The Little Wind Farm that Could: A Comparative Analysis of Lookahead Policies for Energy Storage Problems

Madhumitha Shridharan

Advisor: Professor Warren Powell

Submitted in partial fulfillment

of the requirements for the degree of

Bachelor of Science in Engineering

Department of Operations Research and Financial Engineering

Princeton University

I hereby declare that I am the sole author of this thesis.

I authorize Princeton University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Madhumitha Shridharan

I further authorize Princeton University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Madhumitha Shridharan

Abstract

In this thesis, we explore how to make the best sequential decisions when faced with variations of a stochastic energy storage optimization problem. Each day, smart grid managers need to satisfy the energy demands of a load with wind energy from a wind farm, energy from a rechargable storage device, and energy from an electricity grid. However, this problem is rife with uncertainty, since both the supply of wind energy and grid electricity prices are highly stochastic.

We create seven variations of this problem designed to bring out the features of different policies which make decisions every hour about what combination of energy sources to use to satisfy the power demand, how much energy to store, and how much to buy from the grid in order to maximize profits over a day. For each problem, we evaluate the performance of the policies over a large number of simulated trials to identify the policy which best optimizes revenue when fitted to the problem's electricity market. These problems demonstrate that each class of policies is best for particular problem characteristics. In the process, we combine tools from dynamic programming, approximate dynamic programming, Monte Carlo methods and reinforcement learning.

Acknowledgements

I would like to express my heartfelt gratitude to my wonderful advisor, Professor Warren Powell. I am grateful for his untiring patience for questions, genuine investment in my growth, and contagious enthusiasm for our field. Thank you for guiding me through the intimidating world of graduate school applications, and nurturing my interest and knowledge in the field of Operations Research over a year and a half. Thank you also to Joe Durante and Juliana Nascimento, for your valuable ideas and suggestions.

I also want to thank Professors Christopher Brinton, Mona Singh and William Massey and graduate student Anat Fuchs for offering to mentor me in earlier research projects. The skills and knowledge I have learned from you were very valuable in completing this thesis, and I am sure they will continue to be in any future technical endeavour I set out to do. Thank you to Professor Shkolnikov and Pierre Lamarre for your amazing delivery of ORF 309, my favorite class at Princeton and the turning point that first inspired my interest in bringing mathematical insights to real-world problem settings.

Abby, Brittany, David, Melissa, Morlan, Nicole and Seyoon, thank you for bringing me so much joy throughout these gruelling four years (thanks David for suggesting the title of this thesis!). I will always be grateful for our beautiful friendships. To my family which has always been and forever will be my unrelenting pillar of strength, words can't express how grateful I am, but I'll try: Thank you for your unconditional love and support. You are a gift.

Contents

List of Figures

Chapter 1

Introduction

1.1 The Rise of Renewable Energy

Energy powers the global industrial economy. It is essential for maintaining the health and well-being of people around the world, enabling interdependent infrastructures on which citizens rely, and operating lifeline networks and emergency services. In short, it is the lifeblood of modern civilization.

We live in a time of unpredecented boom in global energy demands. The International Energy Agency (IEA)'s annual World Energy Outlook report projects that global energy consumption will rise by nearly 50 percent between 2018 and 2050 due to rising incomes and an extra 1.7 billion people on the planet, mostly in urban areas in developing economies (2019). However, hydrocarbon-based fuel like coal, gasoline and natural gas that have conventionally satisfied energy demands contribute to climate change and disastrous environmental consequences. This creates the urgent need for clean, zero-carbon energy sources that are renewable and environmentally

sustainable.

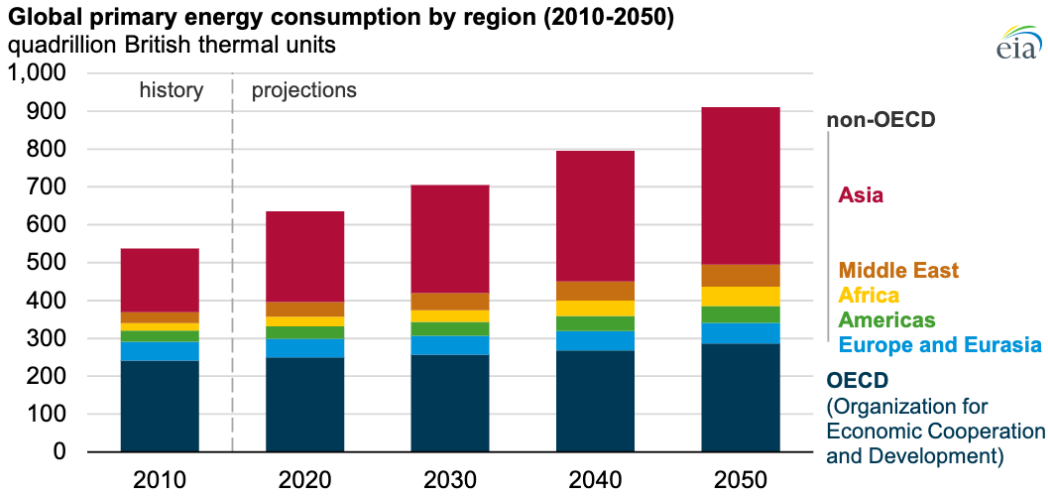

Figure 1.1: Projected rise in global energy consumption from 2010 to 2050. Source: U.S. Energy Information Administration, International Energy Outlook 2019 Reference case

Renewable energy sources like wind and solar offer promising solutions. Besides being clean and sustainable, these sources have also become highly cost efficient in recent years. Users of these sources enjoy marginal costs which are essentially zero, since energy from the wind and sun are free once the initial infrastructure is installed. Moreover, initial infrastructure costs have decreased as well; for example, capital costs for installing solar panels have decreased more than 60% in the last ten years ("The Price of Residential Solar Power" 2018).

1.2 Wind Energy and Storage

In this work, we focus on harnessing the power of wind energy. We live in a time of unprecendented growth in the popularity of wind energy. According to the Global Wind Energy Council, global wind power capacity is expected to increase by 50 percent in the next five years as technology costs fall further and emerging markets drive growth. In 2018 alone, 51.3 gigwatts of new wind installations were added globally, increasing capacity by 9.6 percent. (2019)

Despite this phenomenal progress, greater integration of wind into the grid remains challenging due to its intermitency and unpredictability. Energy sources must be robust to adapt to real-time customer demands: when a consumer turns on an appliance, high-quality electricity must be available to satisfy the demand. Unpredictable power sources hence present a challenge to smart grid managers. Figure 1.2, which was developed from a study of offshore wind, depicts this variability in wind energy (Powell 2019).

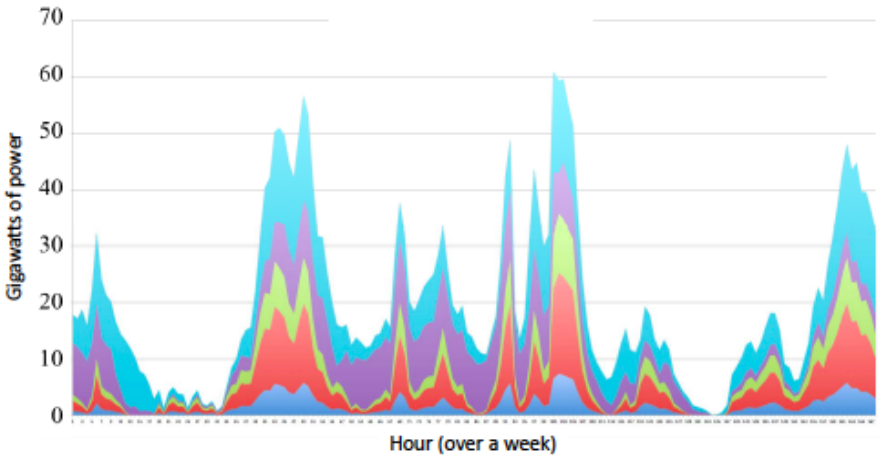

Figure 1.2: Power from five levels of wind generating capacity (Powell 2019)

Energy storage devices offer a promising solution to this problem by storing energy when it is abundant so that it can be used when it is sparse. They smooth out the peaks and valleys in Figure 1 to produce a steady and reliable energy supply to consumers with an unpredictable energy source. This promotes the increase in the renewable portion of electricity delivered to consumers, greatly reducing greenhouse gas emissions associated with fossil fuel - based power generation. In addition, storage improves overall grid performance and management, and gives grid managers greater flexibility in providing power during emergency situations (Powell 2019).

Batteries are one of the most prominent storage options in use today. Battery research is a dynamic field, with institutions vying to be the first to create cheap batteries which offer high energy density. One of the most promising advances in this field is the Tesla Megapack, which offers 60 percent greater energy density compared to the existing Powerpack. In fact, Tesla claims that it can utilize the Megapack to deploy a one gigawatt-hour plant over three acres in under three months, which is approximately a quarter of the time needed to deploy a comparable fossil fuel plant ("With Tesla Megapack, Elon Musk Is Quietly Shifting Hard Toward Clean Energy" 2019). Several other storage technologies for the grid have been developed in recent years as well, like flywheel storage, chemical strorage, compressed-air energy storage, storage as high-temperature heat, and storage in water reservoirs ("pumped storage") (Socolow et al. 2014).

Figure 1.3: The Tesla Megapack. Source: https://www.tesla.com/megapack

1.3 Contributions of this Thesis

In this work, we develop stochastic optimization models for variations of the following energy storage problem: how can we satisfy the energy demands of a building using a stochastic energy supply from a wind farm and a grid with stochastic prices? In our model, we are equipped with a single energy storage device to smooth the energy supply for the building.

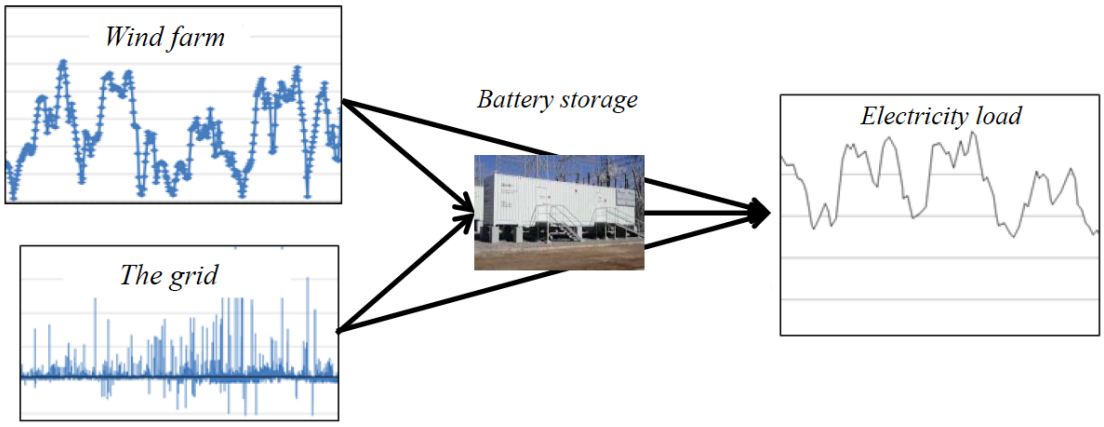

Figure 1.4: Energy storage system, with energy from a wind farm with random supply, the grid with random prices, serving a load with time dependent demands (Powell 2019)

We evaluate the performance of five different policies (PFA, DLA, static BADP, dynamic BADP and SL-PFA) on each problem variation. Overall, the thesis makes the following contributions:

- We use the above described system to show that we can create problem variations that are best suited to each class of policies. We show that by tweaking the problem, we can make any policy work well, and any policy work poorly.
- We design variations of Backward ADP policies which have not been implemented in literature and show that they lead to performance gains.
- We demonstrate that the dynamic Backward ADP policy is the most robust to changes in the problem among all policies tested, and the

PFA policy is the least robust.

- We demonstrate that stochastic lookahead policies have the potential to perform well across time-varying problem variations, provided computationally efficient rollout policies are implemented.

1.4 Organization of the Thesis

We start by presenting the basic model, and then describe how we can change the character of the problem by altering the nature of the underlying data. We then discuss the policy which best optimizes revenue for each problem context.

The next chapter of this thesis provides a literature review of energy storage problems, common modelling techniques and algorithmic solution strategies that have been studied. Chapter 3 presents a basic model of the energy storage problem, the variations of this problem explored in this thesis, strategies used to model the stochastic wind process in all problem variations, and the price process in variations of the problem with daily PJM grid price patterns. The algorithms used in this paper and the benchmark policies they are compared against are presented in Chapter 4. The parameter tuning process for these policies are discussed in Chapter 5. These policies are implemented in the context of each problem variation and numerical results that capture their performance are presented in Chapter 6. These findings are analyzed in Chapter 7, and the paper is concluded in Chapter 8.

Chapter 2

Literature Review

Energy storage optimization is a widely studied field with a plethora of different problem formulations and proposed control policies. In this chapter, we provide a literature review of energy storage problems, common modelling techniques and algorithmic solution strategies that have been studied. We begin with a general overview of policies which have been used to solve different energy storage problems, and then explore strategies to model uncertainty which have been implemented in prior work.

2.1 Policies in Energy Storage Domain

Powell (2019) defines four classes of policy functions which can be applied in sequential decision making problems: policy function approximations (PFA), cost function approximations (CFA), value function approximations (VFA), and direct lookahead policies (DLA). PFAs and CFAs are policies based on policy search, while VFAs and DLAs are policies based on look ahead approx-

imations. A plethora of different formulations of the energy storage problem have been previously studied, and policies from each of the four classes have been applied to them. In particular, given the time varying nature of the energy storage problem, policies based on look ahead approximations have been heavily applied to this domain.

2.1.1 PFAs and CFAs

PFA policies have been studied in the energy storage domain in the form of affine policies. For example, Warrington et al (2012) effectively controlled storage devices and generators in an intraday scheduling problem using affine policies which map a state to an action based on wind forecast errors. Since PFA policies are any policy which maps a state directly to a feasible action, an artificial neural network (ANN) which makes control decisions based on the state of the system is also a PFA. Han et al (2016) applies this technique to allocate energy from a wind farm, storage device, and the power grid to satisfy a time-varying load.

CFA policies in the form of deterministic look-aheads with tunable parameters have been studied as well. Simao et al. (2017) for example used such a policy to robustly control a power system where reserves are explicitly tuned to handle the intermittency of renewables. Similarly, Thalassinakis and Dialynas (2004) used a Monte Carlo simulation method to tune the reserve level and other power system settings.

2.1.2 DLA

Due to their ability to look ahead into the future, DLA policies have been widely studied in the energy storage domain. In particular, rolling horizon procedures like Model Predictive Control (a DLA policy which optimizes over an new deterministic model of the future at each time step in the base model) have been used in various circumstances. For example, Model Predictive Control is often used to maximize the efficiency of heating and cooling systems for large buildings as in Ma et al. (2012). In Arnold and Andersson (2011), Model Predictive Control is used to manage a storage hub with both battery and hot water storage devices to minimize the cost of satisfying loads from a collection of households in the presence of uncertain renewable sources, electricity prices, and natural gas prices.

2.1.3 VFA

We now discuss VFA policies, which are arguably the most widely used policy in problems similar to ours which focus on optimizing the operation of a single energy storage device over a finite time horizon. Prior VFAs implemented have made decisions by optimizing over the base model. For example, in Zhou et al (2013), exact backward dynamic programming is used to find an optimal policy for operating a co-located wind power-storage device system in the presence of stochastic wind and electricity price processes. However, while most energy storage problems tend to have large state spaces to carry information needed to model future exogenous processes, exact backward dynamic programming is often computationally infeasible in problems with

large state spaces. Hence, most algorithmic approaches to optimize over the base model rely on solving an approximation of the base model. For example, Backward approximate dynamic programming (Backward ADP) offers a promising solution to handle the curses of dimensionality as it is based on constructing functions that approximate the values of states, instead of determining a value for every possible state. Backward ADP has been shown to produce solutions close to optimality in realistic battery storage problems in Cheng et al. (2017) and Durante et al. (2017), but otherwise, has not been widely explored in the energy storage domain.

In this thesis, we explore a new policy which incorporates Backward ADP within a stochastic lookahead policy. That is, we make decisions at each time step in the base model by optimizing using Backward ADP over a lookahead model. We describe the implementation of this policy further in Chapter 4.

2.2 Modelling Uncertainty

From the previous section, it is clear that a critical component of any policy which looks ahead is uncertainty modelling. In this section we delve deeper into how uncertainty in energy storage problems have been modelled in the past when VFAs and DLAs are used.

2.2.1 VFAs

In general, previous research work which have applied VFA-based policies have been forced to make great simplifications to decrease the size of the state space. This is because VFA-based policies are algorithms based on

Bellman's equation, which is only efficiently computable if the state space is small. While the true stochastic process may depend on state variables that have not been considered in the simplified version of the problem, the simplification is necessary to ensure that we arrive at a solution. However, the compromise may steer the problem away from the true behavior of the process, undermining the robustness of the solution obtained.

In particular, simplifications have been applied in the modelling of the wind and price processes. Wind processes (such as wind speed or wind power) have been modelled as a first-order Markov process. For example, both Lohndorf et al. (2013) and Zhou et al.(2013) model wind as an autoregressive process, while Jiang et al. (2014) and Cheng et al. (2017) model it as a first-order Markov chain. Similarly, price processes have been simplified in favor of computation. For example, Sioshansi et al. (2014) assume that we have perfect knowledge about the future prices. Prices have also been modelled using a first-order Markov process assumption, as reflected by the mean-reverting models described in Tseng et al. (2002) and Mokrian et al. (2006). A key exception to these prior research works is Durante et al. (2017), which uses a crossing state hidden semi-markov model to accurately recreate daily wind and price processes while maintaining a relatively simple state variable. In this thesis, we use this technique to model our stochastic price process.

2.2.2 DLA

DLAs driven by deterministic forecasts of stochastic exogenous processes have been used widely in the energy domain. For example, Denholm and Sioshansi (2009) study the benefits of co-locating wind farms and storage

devices by utilizing a deterministic lookahead which makes decisions based on a mixed integer program with a forecast for a two-week horizon into the future. Wallace and Fleten (2003) use deterministic forecasts of wind, solar and loads to plan energy generation. However, these models have been widely criticized in the research community for not accounting for uncertainty. Model predictive control policies use rolling forecasts which update at each time step in the base model, but forecast error cannot be controlled in such models.

Unlike prior work, Ghadimi et al. (2019), uses a series of recursive equations to realistically model the evolution of the amount of wind energy generated using rolling forecasts. Furthermore, the speciality of this model is that it allows us to manipulate forecast error and control the quality of the forecast. In this thesis, we adopt this methodology to simulate rolling forecasts and evaluate how varying forecast errors affects a policy's ability to make sequential decisions.

2.3 Chapter Summary

In this chapter, we have provided a literature review of energy storage problems, common modelling techniques and algorithmic solution strategies that have been studied.

In future chapters of this thesis, we will compare the performance of the above policies in a variety of newly designed energy storage problem contexts. Our work is perhaps most similar to Powell and Meisel (2016) in that we create a series of problem variations that are specifically designed to

bring out the strengths of each class of policies. However, unlike this work, we employ sophisticated techniques to model stochastic daily wind and price processes, leading to a far more robust analysis of policy performance in a variety of real world energy contexts. We also develop a new DLA policy which makes decisions at each time step in the base model by optimizing using Backward ADP over a lookahead model, and show that it is more robust than existing policies for our problem contexts.

Chapter 3

Defining the Model

In this chapter, we present the basic model of the energy storage problem outlined in Chapter 1 by defining the state variables, the decision variables, exogenous information, the transition function, and the objective function. We then describe how we can change the character of the problem by altering the nature of the underlying data. Finally, we discuss strategies employed to model the stochastic wind and price processes in our problem variations.

3.1 Basic Model

In our basic model of our energy storage problem, a stochastic wind energy supply, an energy storage device, and the power grid, with an associated stochastic electricity price, are used in combination to satisfy a power demand. The objective is to control the system, whose configuration is illustrated in Figure 3.1, at minimum cost (or maximum profit). There are four nodes, five decision variables, and three exogenous processes in this energy

storage system configuration.

To represent sequential energy allocation decisions in the context of our energy storage problem, we follow the modelling framework and notation outlined in Powell's Sequential Decision Analytics and Modelling textbook, which includes five key components: the state variable, decision variable, exogenous variable, transition function and objective function (Powell 2019).

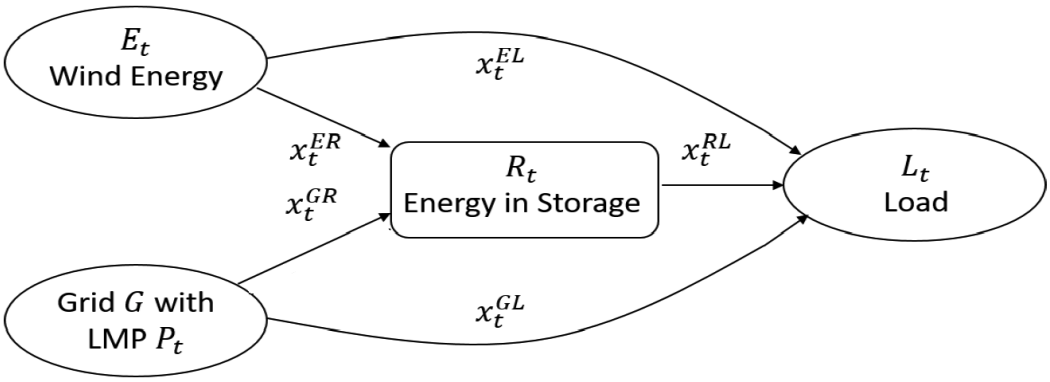

Figure 3.1: An energy storage problem with four energy nodes: wind, the larger power grid, storage, and demand/load; three exogenous variables: wind energy production, the price of electricity, and the demand; and five decision variables representing possible energy allocations at each time step. (Durante 2017)

3.1.1 State Variables

The state variable contains all the information we need to know (from history) to model the evolution of the system. The initial state variable S_0 specifies initial values of any time-varying parameters, deterministic parameters and latent information (such as static forecasts), along with beliefs about

uncertain parameters. The dynamic state variable S_t specifies information that varies over time (Powell 2019).

We begin by defining S_0 and S_t in the context of the energy storage problem. In addition to static forecasts, S_0 contains initial values of wind, price, and storage level. In order to make a decision at time t in our base model, the policies based on look-ahead approximations we implement require us to model the future evolution of wind and price processes after time t. We thus denote the statistics needed to model the evolution of price and wind processes starting from time t as I_t^P and I_t^E respectively. Hence, S_0 also contains I_0^P and I_0^E. Note that since each problem variation has a distinct stochastic price process, $\{I_t^P\}_{t=0}^T$ is problem dependent as well. The techniques used to model the stochastic processes for each problem will be discussed both later in this chapter.

Let $\{f_t^P\}_{t=0}^T$ denote the static forecasts of prices and $\{L_t\}_{t=0}^T$ denote the static hourly load. R_t, E_t,P_t and M_t denote the energy stored, wind supply, grid price and market price at time t respectively. Note that in the problem variations described below, M_t is either equivalent to P_t for all t, or a distinct, static value captured by M_0.

There are services available to forecast wind, but the errors can be quite high. Short term forecasting (0 to 4 hours out) is typically done using "persistence forecasting" where the wind is assumed to stay at the same speed (which is clearly not true in practice). For longer term forecasts ($>$ 4 hours), errors increase steadily, where 12 hour forecasts easily exhibit errors of 100 percent or more. We hence utilize rolling, dynamic forecasts to frequently update these approximations. $f_{tt'}^E$ denotes the forecast of wind energy at time

t' when the forecast is made at time t.

Converting power from AC (as it travels from the grid) to DC (as it is stored in the energy storage system) incurs approximately 5 to 10 percent loss, which is represented by the battery round trip efficiency $\eta \in [0, 1]$.

Our initial state variable is given by:

$$S_0 = (\{f_{0t}^P\}_{t=1}^T, \{f_{0,t}^E\}_{t=1}^T, \{L_t\}_{t=0}^T, R^{max}, \eta, \rho^{charge}, \rho^{discharge}, R_0, E_0, P_0, M_0, I_0^P, I_0^E)$$

where $R^{\max}$ is the maximum capacity of the battery and ρ^{charge} and $\rho^{\text{discharge}}$ are maximum battery charge and discharge rates respectively.

The dynamic state variable S_t contains the current grid price P_t and market price M_t in order to compute the contribution function at time t. It also contains the current supply of wind energy E_t and the current amount of energy stored in the battery R_t in order to track constraints on the decision vector x_t. Finally, it contains $\{f_{tt'}^E\}_{t=0}^T$, I_t^P and I_t^E introduced above to model stochastic processes for both decision making and determining the evolution of the system. S_t for $t > 0$ is thus given by:

$$S_t = (R_t, E_t, P_t, M_t, I_t^P, I_t^E, \{f_{tt'}^E\}_{t'=t}^T)$$

3.1.2 Decision Variables

Our decision variable x_t encapsulates all energy allocation decisons made at time t. Energy is bought from the grid at the current grid price while energy is supplied to the load at the current market price. Wind energy can either be stored or used to satisfy the current energy demand.

Decisions are determined by a pre-determined policy X^π, a measurable function which maps the state of the system S_t to a feasible decision. Note that π contains information that characterizes this function and its tunable parameters. Hence

$$x_t = X^\pi(S_t)$$

Let x_t^{ab} denote the power moving from one device to another, where ab identifes the pair of devices. Our decisions are:

- Power from the wind farm to the battery, x_t^{wr}
- Power from the wind farm to the load, x_t^{wl}
- Power from the grid to the battery, x_t^{gr}
- Power from the grid to the load, x_t^{gl}
- Power from the battery to the load, x_t^{rl}

They are subject to the following constraints:

$$\begin{aligned}
x_t^{w\ell} + x_t^{g\ell} + \eta x_t^{r\ell} &\leq L_t, && (3.1)\\
x_t^{r\ell} &\leq R_t, && (3.2)\\
x_t^{wr} + x_t^{gr} &\leq \frac{1}{\eta}(R^{max} - R_t), && (3.3)\\
x_t^{w\ell} + x_t^{wr} &\leq E_t, && (3.4)\\
x_t^{wr} + x_t^{gr} &\leq \min(u^{charge}, R^{max} - R_t), && (3.5)\\
x_t^{r\ell} &\leq \min(u^{discharge}, R_t), && (3.6)\\
x_t^{wr}, x_t^{w\ell}, x_t^{gr}, x^{g\ell}, x_t^{r\ell} &\geq 0. && (3.7)
\end{aligned}$$

Since we cannot overload the building, constraint (1) limits the total amount of energy transferred to the building from the wind farm, grid and battery combined to the load amount. Constraint (2) limits the amount of energy transferred out of the battery at a time to the amount of energy stored in the battery. Constraint (3) limits the amount of energy transferred into the battery to the amount of capacity left in the battery currently. Capacity (4) limits the amount of energy moved out from the wind farm to the amount of wind energy generated. Constraint (5) limits the amount of energy transferred into the battery to either its charging rate or the capacity left in it, whichever is smaller. Constraint (6) limits the amount of energy transferred out of the battery to either its discharging rate or the energy in it, whichever is smaller. Lastly constraint (7) ensures that all variables are non-negative, since the amount of energy moved between points cannot be negative. Since we assume that the grid can supply infinite power, we do not place constraints on the amount of energy that can be supplied by the grid.

3.1.3 Exogenous Information

Exogenous information variables represent what we learn from an exogenous source after we make each decision. We denote the exogenous information arriving between t and $t+1$ as W_{t+1}.

Letting $\omega \in \Omega$ be a sample path $\{W_t\}_{t=1}^T$, $\mathcal{F}$ be a sigma algebra on Ω and $\mathbb{P}$ be a probability measure on $(\Omega, \mathcal{F})$, we obtain the probability space $(\Omega, \mathcal{F}, \mathbb{P})$. The natural filtration of the stochastic process $\{W_t\}_{t=1}^T$ is denoted $\{\mathcal{F}_t\}_{t=1}^T$, where $\mathcal{F}_t = \sigma(W_1, ..., W_t)$.

Since S_t contains only information from history, it is only a function of

historic variables $\{W_i\}_{i=1}^{t}$. Hence S_t is $\mathcal{F}_t$-measurable by definition. It follows that the decision $x_t = X^\pi(S_t)$ is also $\mathcal{F}_t$-measurable, since it is a measurable function of S_t. Note that the decision we make at time t depends solely on information we know at time t in our model.

The exogenous information arriving between t and $t+1$ is given by $W_{t+1} = (\hat{E}_{t+1}, \hat{P}_{t+1})$, where $\hat{E}_{t+1}$ and $\hat{P}_{t+1}$ denote the error of wind forecast $f^E_{t,t+1}$ and price forecast f^P_{t+1} respectively.

3.1.4 Transition function

The transition function $S^M(S_t, x_t, W_{t+1})$ depicts the evolution of state variables from one time step to the next via a system of equations. Our system is given by:

$$R_{t+1} = R_t + \eta(x_t^{gr} + x_t^{wr}) - x_t^{rl} \tag{3.8}$$

$$E_{t+1} = f^E_{t,t+1} + \hat{E}_{t+1} \tag{3.9}$$

$$P_{t+1} = f^P_{t+1} + \hat{P}_{t+1} \tag{3.10}$$

$$I^P_{t+1} = T^P(I^P_t, \hat{P}_{t+1}) \tag{3.11}$$

$$I^E_{t+1} = T^E(I^E_t, \hat{E}_{t+1}) \tag{3.12}$$

where $T^P(I^P_t)$ and $T^E(I^E_t)$ are functions which capture the evolution of variables used to model future prices and wind respectively. We will detail these functions which depend on the problem variation later in this chapter.

3.1.5 Objective Function

The objective function captures our performance at a particular time, and characterizes the problem of finding optimal policies.

In this problem, we make money by selling electricity to the building, but incur costs when we purchase energy from the grid. Furthermore, we pay a penalty fee of half the market price on the load we were unable to satisfy. Our profit at time t is thus given by

$$C(S_t, x_t) \quad = \quad M_t(x_t^{w\ell} + \eta x_t^{r\ell} + x^{g\ell}) - P_t(x_t^{gr} + x^{g\ell}) - \frac{M_t}{2}(L_t - x^{w\ell} - x^{r\ell} - x^{g\ell})$$

Since we aim to maximise profits with real-time electricity prices and wind supply over a finite horizon, our objective function is then given by

$$max_\pi \mathbb{E}\left[\sum_{t=0}^{T} C(S_t, x_t)|S_0\right]$$

where $S_{t+1} = S^M(S_t, X^\pi(S_t))$ is given by equations (8)-(12). Letting Π be the collection of all measurable mappings from states to decisions, we maximize over the set of all possible policies $\pi \in \Pi$ to determine an optimal policy.

3.2 Problem Variations

We create a series of problem variations that are specifically designed to bring out the strengths of each class of policies. Our sample problems include:

- A stationary problem with constant load, daily PJM grid price patterns

[1], less accurate wind forecasts [2].

- A time-dependent problem with a sinusoidal load, daily PJM grid price patterns, less accurate wind forecasts.
- A time-dependent problem with a sinusoidal load, daily PJM grid price patterns, highly accurate wind forecasts.
- A time-dependent problem with a sinusoidal load, low grid prices most of the time, less accurate wind forecasts.
- A time-dependent problem with a sinusoidal load, high grid prices most of the time, less accurate wind forecasts
- A stationary problem with a constant load, low grid prices most of the time, less accurate wind forecasts.
- A stationary problem with a constant load, high grid prices most of the time, less accurate wind forecasts.

[1]Electricity prices on the PJM grid are highly volatile. For example, energy prices typically range between $20-25$ per mwh, but can often peak at over $300, and even exceed $1000 in rare weather circumstances. Prices can spike dramatically, and while it is not possible to forecast when spikes will happen, the likelihood of spikes increase with extreme temperatures (very hot or very cold). Although temperature can be accurately forecasted, an independent forecast model is required for prices. Refer to section 3.3 for our methodology for modelling daily PJM grid prices.

[2]Refer to section 3.3 for our methodology for modelling the wind process and for our derivation of wind forecasts for all problem variations

3.3 Uncertainty Quantification

By design, we have two sources of uncertainty in this energy storage problem: the energy produced by wind and the electricity price. In this section, we outline the procedures used to model the stochastic wind process in all problem variations, and the price process in variations of the problem with daily PJM grid price patterns.

3.3.1 Prices

In order to model the stochastic evolution of PJM grid prices, we follow the univariate crossing state HSMM methodology outlined in Durante et al. (2017). This model accurately reflects the distribution of times for which prices are above and below their forecast levels, otherwise known as crossing time. Here we present a summary of how we can use empirical data to train this model.

As mentioned in Chapter 1, electricity prices are correlated with temperature, which is in turn correlated with time of day. It thus displays a daily periodic pattern. However, it is also highly volatile; prices typically range from \$20 - \$25, per mwh, but can often peak at over \$300. We typically observe that the higher the temperature, the higher the price spikes observed. These characteristics are depicted in Figure 3.2, a two-week snapshot of pricing data from Princeton New Jersey in the summer of 2015 (Durante et al. 2017).

In order to train our model in this thesis, we use pricing data which was recorded every day from June 1 to August 31 2015 in 5-minute intervals. We

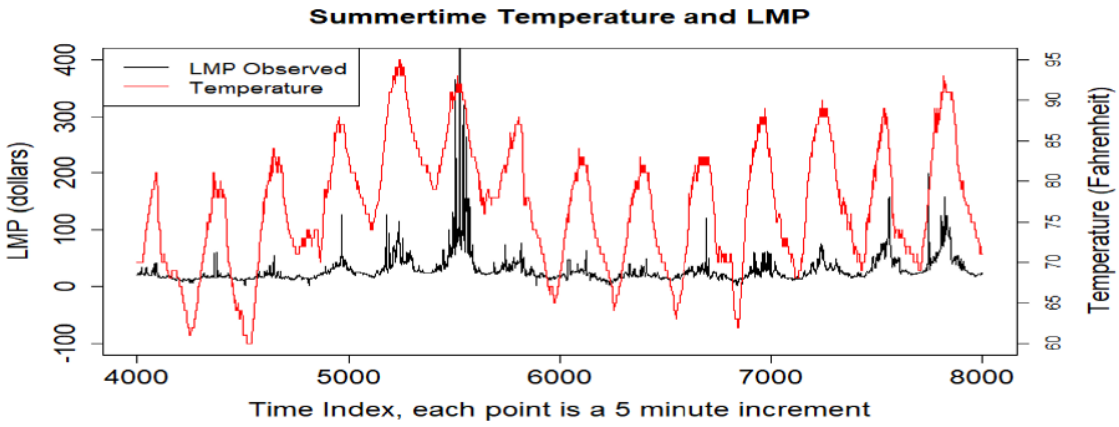

Figure 3.2: Electricity Prices and temperature data for the Princeton, NJ, area during two weeks in July 2015 (Durante et al. 2017)

model price evolution on an hourly basis in this thesis, and so we utilized the highest recorded price every hour to train our model to ensure that we capture price spikes.

We can decompose a price path observed into its seasonal (daily) part, and its random part. Hence

$$P_t = P_t^{seas} + P_t^{random}$$

We first construct our static price forecasts $\{f_t^P\}_{t=0}^T$. We begin by subtracting the seasonal component at each time t from P_t to get P_t^{random}.

$$P_t^{random} = P_t - P_t^{seas}$$

We then construct our static price forecast by computing:

$$f_t^P = \bar{p} + P_t^{seas}$$

where $\bar{p}$ is the average price in the historical dataset. As this forecast is deterministic over our optimization horizon, $\{f_t^P\}_{t=0}^T \in S_0$.

As mentioned, since prices are correlated with temperature, our price distribution at time $t+1$ depends on the temperature forecast for $t+1$. Recall that we utilize a static temperature forecast over our horizon contained in S_0 as it accurately depicts future temperature behavior and simplifies both the modelling framework and computations.

By applying a length 50 moving average filter, we decompose the temperature series into two components: the seasonal component $\{h_t^s\}_{t=0}^T$ and the temperature classification component which captures the average temperature computed by the filter $\{h_t^{tr}\}_{t=0}^T$. We aggregate these continuous variables as follows:

$$y_t^s = \left\{ \begin{array}{ll} 2, & h_t^s \geq 0.75 \times h^{s,max} \\ 1, & h_t^s < 0.75 \times h^{s,max} \end{array} \right\} \tag{3.13}$$

where $h^{s,max} = \max_t h_t^s$.

$$y_t^{tr} = \left\{ \begin{array}{ll} 3, & h_t^{tr} \geq 0.8 \times h^{tr,max} \\ 2, & 0.3 \times h^{tr,max} \leq h_t^{tr} < 0.8 \times h^{tr,max} \\ 1, & h_t^{tr} < 0.3 \times h^{tr,max} \end{array} \right\} \tag{3.14}$$

where $h^{tr,max} = \max_t h_t^{tr}$.

Aggregation (13) separates periods when the temperature is peaking from all other times of the day, while aggregation (14) separates cool, average, and hot days. Since we used a static temperature forecasts, both $\{y_t^s\}_{t=0}^T$ and $\{y_t^{tr}\}_{t=0}^T$ are static series, and are part of S_0.

We now make concrete the information state I_t^P, which contains all information needed to model the evolution of the price process from time t onward.

We begin by defining the forecast error $\hat{P}_t = P_t - f_t^P$ where P_t is the actual price at time t. Let $C_t^P = \mathbb{1}_{\{\hat{P}_t \geq 0\}} \in \{0, 1\}$ denote whether the price process is above or below the forecast by specifying the sign of the error. We call this state the crossing state.

Next we introduce the notions of running up and down crossing times, and complete up and down crossing times. A running up-crossing time of duration d starting at time t is given by:

$$\tau_t^{U,P} = d \text{ if } \left\{ \begin{array}{ll} C_{t-d}^P = 0 & \\ C_{t-d'}^P = 1 & \forall d' \in \{0, 1, ..., d-1\} \end{array} \right\} \tag{3.15}$$

Similarly, a running down-crossing time of duration d at time t is defined as:

$$\tau_t^{D,P} = d \text{ if } \left\{ \begin{array}{ll} C_{t-d}^P = 1 & \\ C_{t-d'}^P = 0 & \forall d' \in \{0, 1, ..., d-1\} \end{array} \right\} \tag{3.16}$$

Although we define $\tau_t^{U,P}$ using the list of crossing state variables $\{C_{t-k}^P\} \forall k \in \{0, 1, ..., d-1, d\}$, we can represent $\tau_t^{U,P}$ compactly during algorithmic execution by letting it be a counter which keeps track of the number of time periods we have been above (or below) the forecast.

On the other hand, a complete crossing time is a running crossing time with a switch in the sign of error at the time. That is, if the price process was previously above the forecast, it moves to be below the forecast, and vice versa. We formalize this by letting $\mathcal{C}^{U,P}$ be the set of all points where errors convert from being negative to being positive, and $\mathcal{C}^{D,P}$ be the set of all points where errors convert from being positive to being negative. That is:

$$\mathcal{C}^{U,P} = \{t | \hat{P}_{t-1} \leq 0 \text{ and } \hat{P}_t > 0\} \text{ and } \mathcal{C}^{D,P} = \{t | \hat{P}_{t-1} \geq 0 \text{ and } \hat{P}_t < 0\}$$

Running crossing times with $t+1 \in \mathcal{C}^{U,P} \cup \mathcal{C}^{D,P}$ are known as complete crossing times. Using historical data, we construct empirical distributions $F^{U,P}$ and $F^{D,P}$ for complete up-crossing and down-crossing times respectively.

We now describe how we generate error at time $t+1$ by conditioning on the crossing state and temperature at time t. For each crossing state-temperature state combination i, we can obtain empirical conditional error CDFs $F_i^{\hat{P}}$ and corresponding error density functions $\mathbf{P}(\hat{P}|i)$ from historical data. Note that conditioning error generation on the crossing state and temperature and avoiding the simplifying assumption that all crossing and temperature states have identical error distributions helps us model the behavior of the error process with greater accuracy.

Within a time period where the crossing state remains unchanged, transitions between error quantities can be captured using a first order Markov Chain. We first partition each conditional error distribution $F_i^{\hat{P}}$ into n^P bins, splitting at the $q_j = \frac{j}{n^P}$ quantile points for $j = 0, 1, ..., n^P - 1$. The error $\hat{P}_t$ belongs to bin $\hat{P}_t^b$ if $q_b \leq F_i^{\hat{P}}(\hat{P}_t) < q_{b+1}$. Next, given $C_t^P = C_{t+1}^P = i$, $\hat{P}_t \in \hat{P}^b$, y_t^s and y_t^{tr} we form conditional empirical distributions for the error at time $t+1$ giving $\mathbf{P}(\hat{P}_{t+1}|C_{t+1}^P = C_t^P = k, \hat{P}_t^b, y_t^s, y_t^{tr})$.

Note that the same error value can belong to different error bins for different crossing state-temperature state combinations. Hence, we always pair the variable $\hat{P}_t^b$ with a crossing state and a temperature state, such that $\hat{P}_t^b$ denotes the bin that the error P_t lies in for the corresponding crossing state-temperature state combination.

Finally, from historical data, we compute error density $\mathbf{P}(\hat{P}_{t+1}|C_t, t+1 \in \mathcal{C}^{U,P} \cup \mathcal{C}^{D,P}, y_t^{tr}, y_t^s)$ for each crossing state-temperature state combination.

This denotes the distribution of the initial error given the process has just transitioned to the new crossing state i.

The information state I_t^P which contains all information needed to model the evolution of the price process from t onwards is thus given by: $I_t^P := (C_t^P, \tau_t^P, \hat{P}_t^b)$. If known, we can completely determine the distribution of the exogenous error at time $t+1$ by

$$\mathbf{P}(\hat{P}_{t+1}|C_t^P = k, \tau_t^P, \hat{P}_t^b, y_t^s, y_t^{tr}) = (1 - F_i^{\tau,P}(\tau_t^P))\mathbf{P}(\hat{P}_{t+1}|k, \hat{P}_t^b, y_t^s, y_t^{tr})$$
$$+F_i^{\tau,P}(\tau_t^P)\mathbf{P}(\hat{P}_{t+1}|k', t+1 \in \mathcal{C}^{U,P} \cup \mathcal{C}^{D,P}, y_t^s, y_t^{tr})$$

where $k' \neq k$.

In section 3.1.4 on the transition function, we introduced $T^P(I_t^P)$ which denotes a function that captures the evolution of the price information state. $T^P(I_t^P)$ is given by the following system of equations:

$$C_{t+1}^P = \mathbb{1}_{\{\hat{P}_{t+1} \geq 0\}} \tag{3.17}$$

$$\tau_{t+1}^P = \left\{ \begin{array}{ll} \tau_t^P + 1, & \text{if } \mathbb{1}_{\{\hat{P}_{t+1} \geq 0\}} = C_t^P, \\ 1, & \text{otherwise} \end{array} \right\} \tag{3.18}$$

$$\hat{P}_{t+1}^b = b \text{ s.t. } q_b \leq F^{\hat{P}}_{\mathbb{1}_{\{\hat{P}_{t+1} \geq 0\}}}(\hat{P}_{t+1}) < q_{b+1} \tag{3.19}$$

3.3.2 Wind

In order to model the stochastic evolution of wind, we follow the methodology outlined in Ghadimi et al. (2019) which creates complex nonstationary behaviors to evaluate a policy's ability to use rolling forecasts in sequential

decision making problems. In particular, this model uses a series of recursive equations to realistically model the evolution of the amount of wind energy generated.

Let $\{f^E_{t,t'}\}$ be the forecast of wind energy generated at time t' made at time t. Assume $\{f^E_{0,t'}\}_{t'=0,...min(H,T)}$ has been determined by a wind forecasting service and is known. In this thesis, we randomly generate $\{f^E_{0,t'}\}_{t'=0,...min(H,T)}$. $\forall t \in \{0, ..., T-1\}$ and $t' \in \{t+1, ..., \min(t+H, T)\}$, we define

$$f^E_{t+1,t'} = f^E_{t,t'} + \hat{E}_{t+1,t'} \tag{3.20}$$

where $\hat{E}_{t+1,t'}$ represents degree of noise whose distribution depends on $f^E_{t,t'}$. To create such noise, we first construct a symmetric matrix $\Sigma \in \mathbb{R}^{H\times H}$ such that $\Sigma(i,j) = e^{-\alpha|i-j|} \forall i, j$, where $\alpha \in \mathbb{R}$. By construction, Σ is a covariance matrix which represents less correlation between the i-th and j-th elements when they are far from each other. We denote the noise vector at time t by $\hat{E}_t$, where

$$\hat{E}_t = \begin{bmatrix} \hat{E}_{t,t+1} \\ \hat{E}_{t,t+2} \\ . \\ . \\ \hat{E}_{t,min(t+H,T)} \end{bmatrix}$$

.

Then we can compute $\hat{E}_{t+1}$ using forecasts $\{f_{t,t'}\}_{t'=t+1,...min(t+H,T)}$ by:

$$\hat{E}_{t+1} = L_{H\times H} \cdot Z$$

where $L_{H\times H}$ is the top-left $H \times H$ block of the lower triangular Cholesky decomposition of Σ and

$$Z_t \sim \mathcal{N}\left(0, \sigma_E \begin{bmatrix} \sqrt{f_{t,t+1}} & 0 & . & . & 0 \\ 0 & \sqrt{f_{t,t+2}} & . & . & 0 \\ . & & & & \\ . & & & & \\ . & & & & \\ 0 & . & . & . & \sqrt{f_{t,min(t+H,T)}} \end{bmatrix}\right)$$

.

The speciality of this model is that it allows us to manipulate σ_E, and thus control the quality of the forecast. In Figures 3.3 to 3.5, we note how increasing σ_E increases the noise in our generated wind forecasts.

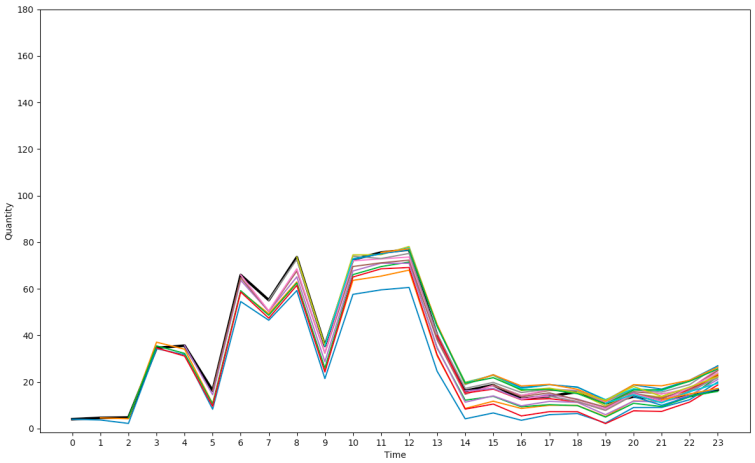

Figure 3.3: Evolution of evolving forecasts of wind power over the course of a 24 hour period with $\sigma_E = 1$. The black line is the actual.

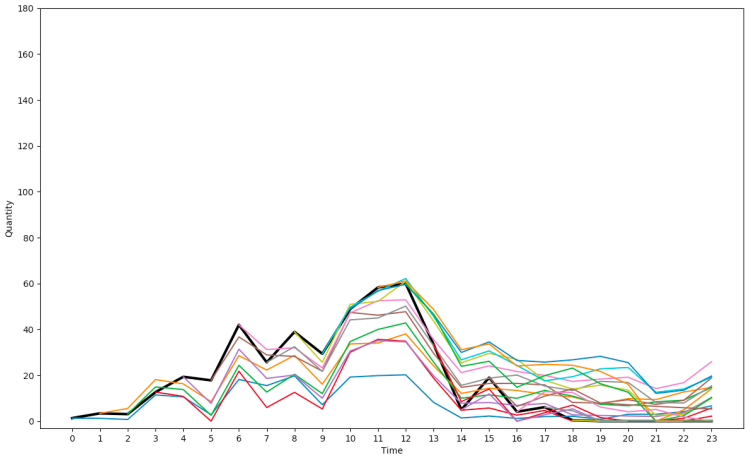

Figure 3.4: Evolution of evolving forecasts of wind power over the course of a 24 hour period with $\sigma_E = 10^2$. The black line is the actual.

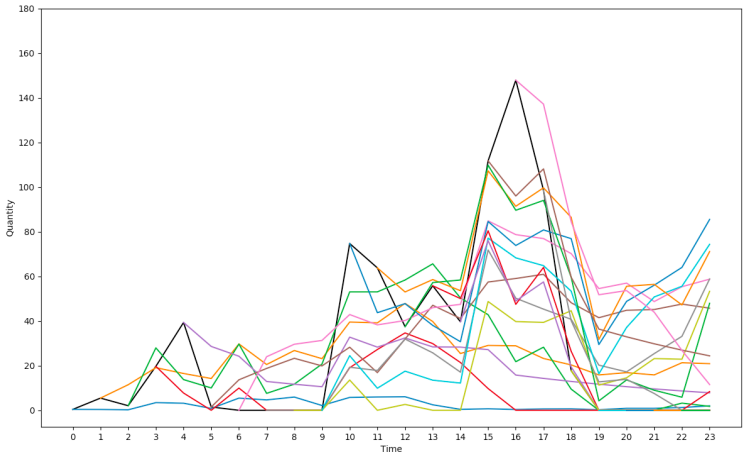

Figure 3.5: Evolution of evolving forecasts of wind power over the course of a 24 hour period with $\sigma_E = 10^4$. The black line is the actual.

The information state I_t^E which contains all information needed to model the evolution of the wind process from t onwards is thus given by: $I_t^E := \{f_{t,t'}\}_{t'=t+1}^{min(H,T)}$. The corresponding function $T^E(I_t^E)$ is given by:

$$f_{t+1,t'}^E = f_{t,t'}^E + \hat{E}_{t+1,t'} \tag{3.21}$$

where $\hat{E}_{t+1,t'}$ is computed using the method described above.

Note that quantities α and σ_E which are used in $T^E(I_E)$ are contained in I_0^E (and thus in S_0) since they are constants.

3.4 Chapter Summary

In this chapter, we presented the basic model of the energy storage problem outlined in Chapter 1 by defining the state variables, the decision variables, exogenous information, the transition function, and the objective function. We then described the variations of this model we created that are specifically designed to bring out the strengths of each class of policies. Finally, we described strategies employed to model the stochastic wind and price processes in our problem variations.

In the next chapter, we will discuss the policies we designed to determine how much energy to transfer between the grid, battery and load at each time step in our problem variations.

Chapter 4

Designing Policies

With modelling under our belt, we now introduce the policies we designed that determined energy allocation decisions in our problem variations. We will follow the universal framework for policy design outlined in Powell (2019).

Powell defines four classes of policy functions which can be applied in sequential decision making problems: policy function approximations (PFA), cost function approximations (CFA), value function approximations (VFA), and direct lookahead policies (DLA). PFAs and CFAs are policies based on policy search, while VFAs and DLAs are policies based on look ahead approximations. We discuss these policies by adhering to the formulations presented in Powell (2019).

4.1 Policy Function Approximations (PFAs)

A policy function approximation is any analytical function mapping a state to an action. These may come in any one of three forms: lookup table (where

we map a discrete state to a discrete action), parametric functions, and nonparametric functions. Critically, unlike the other three classes of policies, PFAs do not require us to solve an optimization problem to determine the next action (Powell 2019).

In this thesis, we utilize a carefully tuned buy-low, sell-high PFA defined by $\theta = (\theta^H, \theta^L)$ with $\theta^H > \theta^L$:

$$
X_t^{PFA}(S_t|\theta) \quad = \quad \left\{ \begin{array}{l} x_t^{EL} = \min\{L_t, E_t\}, \\ x_t^{RL} = \left\{ \begin{array}{ll} \min\{L_t - x_t^{EL}, \min\{R_t, \rho^{dch}\}\}, & \text{if } P_t > \theta^H \\ 0, & \text{if } P_t \leq \theta^H \end{array} \right\} \\ x_t^{GL} = L_t - x_t^{EL} - x_t^{RL} \\ x_t^{ER} = \min\{E_t - x_t^{EL}, \rho^{ch}, R^{max} - R_t\} \\ x_t^{GR} = \left\{ \begin{array}{ll} \min\{\rho^{ch} - x_t^{ER}, R^{max} - R_t - x_t^{ER}\}, & \text{if } P_t < \theta^L \\ 0, & \text{if } P_t \geq \theta^L \end{array} \right\} \end{array} \right\}
$$

where θ is manually tuned via grid search. The θ resulting in the highest average objective function value in each case is chosen.

4.2 Value Function Approximations (VFAs)

A value function approximation captures the value (or benefit) of being in a downstream (or future) state by estimating the contribution of decisions made after landing in a state (Powell 2019). In this work, we explore a VFA-based policy known as Backward Approximate Dynamic Programming. We first summarize a precursor algorithm known as Backward Dynamic Pro-

gramming, and then delve into a presentation of the Backward Approximate Dynamic Programming algorithm.

4.2.1 Backward Dynamic Programming

Let $\mathcal{S}$ be the set of possible states in our problem, and $\mathcal{A}_s$ be the set of possible decisions from state $s \in \mathcal{S}$. Let the reward we obtain when we make decision x_t while in state s_t at time t be $C_t(s_t, x_t)$. Finally, assume we know the probability of seeing exogenous information w after taking action x_t while in state s_t, $f^W(w|s_t, x_t)$.

We are interested in computing $V_t(S_t)$, the value of being in state S_t at time t. If $V_{t+1}(S_{t+1})$ is known for all $S_{t+1} \in \mathcal{S}$, $V_t(S_t)$ is given by:

$$V_t(S_t) = \max_{x_t \in \mathcal{A}_{S_t}} \left(C_t(S_t, x_t) + \mathbb{E}\left[V_{t+1}(S^M(S_t, x_t, w)) | S_t, x_t \right] \right) \tag{4.1}$$

This hallmark equation is known as **Bellman's Equation** and is the foundation for all forms of VFA policies. Rewriting the expectation term of this equation, we obtain:

$$V_t(S_t) = \max_{x_t \in \mathcal{A}_{S_t}} \left(C_t(s_t, x_t) + \sum_{w \in \mathcal{W}} f^W(w|s_t, x_t) V_{t+1}(S^M(s_t, x_t, w)) \right) \tag{4.2}$$

Note that if we can compute the values for all states using this equation, then we can arrive at an optimal policy $X^*(S_t)$ which maximizes the one-step contribution plus the expected value of the downstream state. This is given by:

$$X^*(S_t) = argmax_{x_t} \left(C_t(s_t, x_t) + \sum_{w \in \mathcal{W}} f^W(w|s_t, x_t) V_{t+1}(S^M(s_t, x_t, w)) \right)$$

A brute force computation of Bellman's equation is known as Backward Dynamic Programming.

Algorithm 1 Backward Dynamic Programming Algorithm

Initialize terminal condition $V_{T+1}(S_{T+1}) = 0$, for all states S_{t+1}

for $t = T, T-1, ...0$ **do**

  for all $s \in \mathcal{S}$ **do**

    $V_t(s_t) = max_{x_t}(C_t(s_t, x_t) + \sum_{w \in \mathcal{W}} f^W(w|s_t, x_t)V_{t+1}(S^M(s_t, x_t, w)))$

  end for

end for

4.2.2 Backward Approximate Dynamic Programming

The brute force algorithm to compute values of states described above is often computationally intractable for real-world problems when the state space, decision space and exogenous information space is too large, as is the case with the problem we have at hand. We thus rely on constructing approximations of the values described above instead and use the policy:

$$X_t^\pi(S_t) = argmax_{x_t} \left(C(s_t, x_t) + \sum_{w \in \mathcal{W}} f^W(w|s_t, x_t)\bar{V}_{t+1}(S^M(s_t, x_t, w)) \right)$$

where $\bar{V}_{t+1}(S_{t+1})$ is some approximation of the value of a downstream state.

This policy however, still requires us to compute the expectation $\mathbb{E}\left[V_{t+1}(S^M(S_t, x_t, w))|S_t, x_t\right] = \sum_{w \in \mathcal{W}} f^W(w|s_t, x_t)\bar{V}_{t+1}(S^M(s_t, x_t, w))$.

In order to avoid this, we note that the transition function $S^M(S_t, x_t, W_{t+1})$ can be decomposed into two parts: $S_t^x = S^{M,x}(S_t, x_t)$ and $S_{t+1} = S^{M,W}(S_t^x, W_{t+1})$.

The variable S_t^x is the state variable immediately after we execute a decision (also known as the post-decision state). We can then design the following VFA-based policy that makes decisions based on the values of the post-decision states:

$$X_t^\pi(S_t) = argmax_{x_t}(C(S_t, x_t) + V_t(S_t^x))$$

Our post-decision state is given by

$$S_t^x = (R_t^x, I_t^P, \{f_{tt'}^E\}_{t'=t+1}^T)$$

where $R_t^x = R_t + \eta(x_t^{gr} + x_t^{wr}) - x_t^{rl}$.

We are now left with the task of approximating $V_t(S_t^x)$. In order to handle high dimensional forecasts in the lookahead model of our Backward ADP algorithm, we fix the forecasts into the model, rather than modeling their evolution over time. That is, $\{f_{tt'}^E\}_{t'=t+1}^T$ is a latent variable in the lookahead model constructed at time t. Our post-decision state variable at time t' in the lookahead model constructed at time t is thus given by

$$S_{tt'}^x = (R_{tt'}^x, I_t^P)$$

In this work, we approximate $V_t(S_t^x)$ using a linear model with parameter vector θ_t given by

$$\bar{V}_t(S_t^x|\theta_t) = \sum_{f\in\mathcal{F}} \theta_{t,f}\phi_f(S_t^x)$$

Since the price process varies from problem to problem, I_t^P and S_t^x varies from problem to problem. As a result, our basis functions of the post-decision

states ϕ_f for $f \in \mathcal{F}$ are problem-dependent, and have been customised for the specific characteristics of each energy storage problem. For problems with the PJM grid price process, I_t^P contains the HSMM crossing state variables described in section 3.3.1. The basis functions are thus given by

$$\begin{aligned}
\phi_1(S_t^x) &= (C_t^P - 0.5)\tau_t \\
\phi_2(S_t^x) &= R_t^x \\
\phi_3(S_t^x) &= (R_t^x)^2 \\
\phi_4(S_t^x) &= R_t^x(C_t - 0.5)\tau_t \\
\phi_5(S_t^x) &= (C_t^P - 0.5)P_t^b \\
\phi_6(S_t^x) &= R_t^x(C_t^P - 0.5)P_t^b
\end{aligned}$$

For problems with either mostly low or mostly high prices, we have

$$\begin{aligned}
\phi_1(S_t^x) &= R_t^x \\
\phi_2(S_t^x) &= (R_t^x)^2
\end{aligned}$$

Finally, we learn the parameter vector θ_t and compute $\bar{V}_t$ via Algorithm 2:

Algorithm 2 Backward Approximate Dynamic Programming Algorithm

Initialize terminal condition $\bar{V}_{T+1}(S_{T+1}) = 0$, for all states S_{t+1}

for $t = T, T-1, ...0$ **do**

  for $n = 1, 2, ...N$ **do**

    Sample $s^x_{t-1} \in \mathcal{S}^x_{t-1}$

    Sample W_t and simulate to the next pre-decision state s_t.

    Compute $v_t(s_t) = max_{x_t}(C_t(s_t, x_t) + \bar{V}_t(s^x_t))$

  end for

  Create dataset $(s^{x,n}_{t-1}, v^n_t)^N_{n=1}$ and use this to fit a linear model for $\bar{V}_{t-1}(s^x_{t-1})$.

end for

We refer to Algorithm 2 as the *static* Backward Approximate Dynamic Programming algorithm (BADPS). In the next section on Direct Lookahead Models, we discuss a slight modification to Algorithm 2 we implemented where we repeat the algorithm at each time step in the base model, instead of implementing it once at time 0 of the base model. This is known as the *dynamic* Backward Approximate Dynamic Programming algorithm (BADPD).

This concludes our discussion of the Backward Approximate Dynamic Programming algorithm.

4.3 Direct Lookahead Models (DLAs)

When function approximations of the future are difficult to formulate, we can instead use a direct lookahead model (DLA). The DLA optimizes decisions over some pre-specific horizon H using a future approximation of the base

model. DLAs can be either deterministic or stochastic.

4.3.1 Deterministic Lookahead (DL)

As discussed in Chapter 3, the canonical formulation of our base model is given by

$$max_{\pi} E\left[\sum_{t=0}^{T} C(S_t, x_t)|S_0\right]$$

where $C(S_t, x_t) = M_t(x_t^{w\ell}+\eta x_t^{r\ell}+x^{g\ell})-P_t(x_t^{gr}+x^{g\ell})$ and $S_{t+1} = S^M(S_t, X^{\pi}(S_t), W_{t+1})$.

A deterministic lookahead model is a deterministic approximation of the base model. We hence denote all variables in the lookahead model with tildes, and index them by both the time t at which we are making our decision, and time t' within the lookahead model. In the context of the energy storage problem, we hence define:

- The decision vector to be executed at time t' in the lookahead model being made at time t, $\tilde{x}_{tt'}$
- The cost coefficient vector for $\tilde{x}_{tt'}$, $\tilde{c}_{tt'}$
- The energy in the battery at time t' in the lookahead model generated at time t, $\tilde{R}_{tt'}$

Note that $x_t = \tilde{x}_{tt}$, $c_t = \tilde{c}_{tt}$ and so on.

Our deterministic lookahead policy $X_t^{DL}(S_t)$ is given by the following linear program:

$$X_t^{DL}(S_t) \quad = \quad argmax_{x_t,(\tilde{x}_{tt'},t'=t+1,\ldots,t+H)}\left[C(S_t, x_t) + \sum_{t'=t+1}^{t+H} C(\tilde{S}_{tt'}, \tilde{x}_{tt'})\right]$$

where

$$C(S_t, x_t) = (x_t^{w\ell} + \eta x_t^{r\ell} - x_t^{gr}) f_t^P, \tag{4.3}$$

$$C(\tilde{S}_{tt'}, \tilde{x}_{tt'}) = (\tilde{x}_{tt'}^{w\ell} + \eta \tilde{x}_{tt'}^{r\ell} - \tilde{x}_{tt'}^{gr}) f_t^P \tag{4.4}$$

This problem has to be solved subject to the constraints (1) - (7) in section 3.1.2 for x_t, and the following constraints for $\tilde{x}_{tt'}$ for all $t' = t+1, \ldots, t+H$:

$$\tilde{R}_{t,t'+1} - (\tilde{R}_{tt'} - \tilde{x}_t^{r\ell} + \eta \tilde{x}_{tt'}^{wr} + \eta \tilde{x}_{tt'}^{gr} = 0 \tag{4.5}$$

$$\tilde{x}_{tt'}^{w\ell} + \tilde{x}_{tt'}^{g\ell} + \eta \tilde{x}_{tt'}^{r\ell} \leq L_{tt'} \tag{4.6}$$

$$\tilde{x}_{tt'}^{r\ell} \leq \tilde{R}_{tt'} \tag{4.7}$$

$$\tilde{x}_{tt'}^{wr} + \tilde{x}_{tt'}^{gr} \leq \frac{1}{\eta}(R^{max} - \tilde{R}_{tt'}) \tag{4.8}$$

$$\tilde{x}_{tt'}^{w\ell} + \tilde{x}_{tt'}^{w\ell} \leq f_{tt'}^E \tag{4.9}$$

$$\tilde{x}_{tt'}^{wr} + \tilde{x}_{tt'}^{gr} \leq u^{charge} \tag{4.10}$$

$$\tilde{x}_{tt'}^{r\ell} \leq u^{discharge} \tag{4.11}$$

$$\tilde{x}_{tt'} \geq 0 \tag{4.12}$$

The optimization problem described above is a linear program that can be easily solved by packages in various programming languages. In this thesis, we use the cvxopt package in Python.

Since the problem we have at hand is dynamic and time-varying where our forecasts of exogenous processes are rolling, we solve a new linear program in each time step to make decisions on a rolling basis as well. That is, at each time step in the base model, we use the updated forecast as a latent variable in the lookahead model and solve the above linear program with these forecasts to make a decision at that time step. This is illustrated in Figure 4.1.

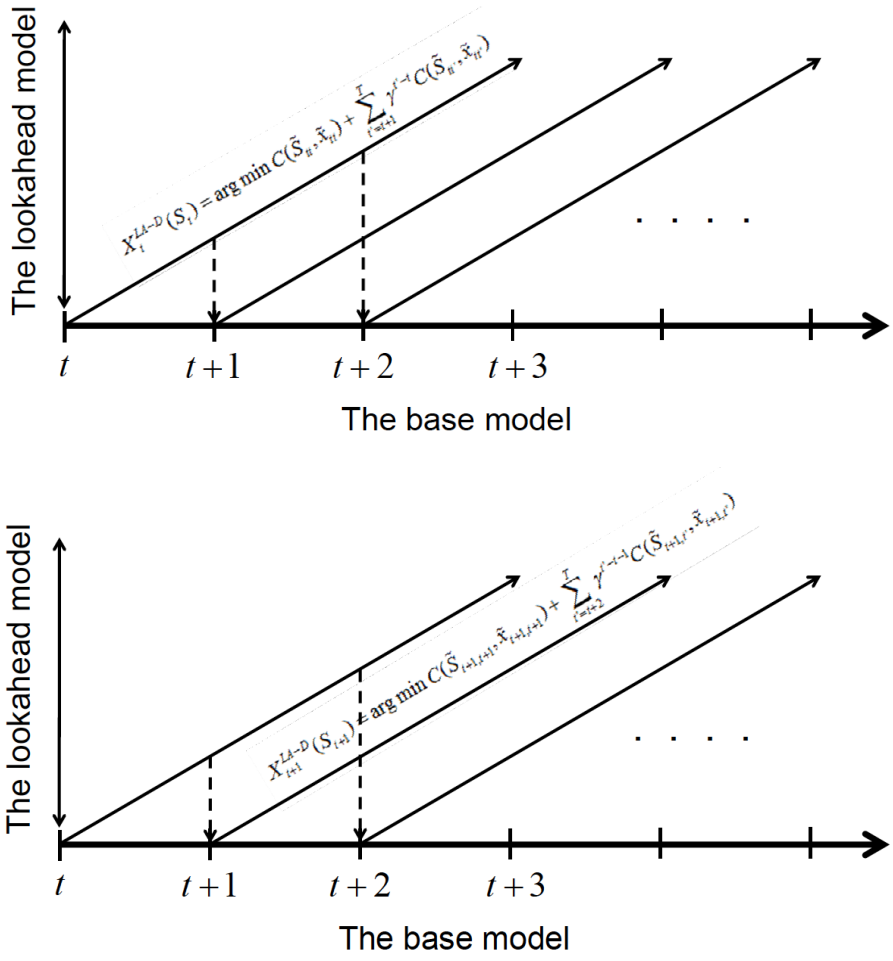

Figure 4.1: Rolling Horizon Procedure for Deterministic Lookahead (Powell 2019)

Observe that should our wind and price forecasts be perfect, we would obtain the optimal decision sequence for the entire horizon for our exogenous information process $(S_0, W_1, ..., W_T)$ at hand. This gives us a powerful way to benchmark our proposed algorithms.

4.3.2 Stochastic Lookaheads (SLs)

While the deterministic lookahead policy uses a deterministic approximation of the base model, the stochastic lookahead policy makes decisions by stochastically approximating exogenous information by using a sampled set of future outcomes.

Recall that the optimal policy is given by:

$$X_t^*(S_t) = argmax_{x_t} \left[C(S_t, x_t) + \mathbb{E} \left[max_{\pi \in \Pi} \left[\mathbb{E} \sum_{t'=t+1}^{T} C(S_{t'}, X_{t'}^{\pi}(S_{t'})) | S_{t+1} \right] | S_t, x_t \right] \right]$$

We replace the full lookahead with an approximation

$$X_t^{SL}(S_t) = argmax_{x_t} \left[C(S_t, x_t) + \tilde{\mathbb{E}} \left[max_{\tilde{\pi} \in \tilde{\Pi}} \left[\tilde{\mathbb{E}} \sum_{t'=t+1}^{T} C(\tilde{S}_{t'}, \tilde{X}_{t'}^{\pi}(\tilde{S}_{t'})) | \tilde{S}_{t+1} \right] | S_t, x_t \right] \right]$$

We can replace the expectations with Monte Carlo samples, which leaves us the challenge of designing the "policy within a policy", which can be any of the four classes of policies. To simplify computations, we use a parameterized approximation $\tilde{X}^{\pi}(\tilde{S}_{tt'}|\tilde{\theta})$ which involves solving:

$$X_t^{SL}(S_t) = argmax_{x_t} \left[C(S_t, x_t) + \tilde{\mathbb{E}} \left[max_{\tilde{\theta}_t} \left[\tilde{\mathbb{E}} \sum_{t'=t+1}^{T} C(\tilde{S}_{t'}, \tilde{X}_{t'}^{\pi}(\tilde{S}_{t'}|\tilde{\theta}_t)) | \tilde{S}_{t+1} \right] | S_t, x_t \right] \right]$$

This requires finding $\tilde{\theta}_t(\tilde{S}_{t,t+1})$, which implies that we have to retune our inner policy for each future state we land on. To further simplify computations, we replace $\tilde{\theta}_t(\tilde{S}_{t,t+1})$ with a fixed parameter θ and then solve the parameterized policy:

$$X_t^{SL}(S_t|\theta) = argmax_{x_t} \left[C(S_t, x_t) + \tilde{\mathbb{E}} \left[max_{\theta} \left[\tilde{\mathbb{E}} \sum_{t'=t+1}^{T} C(\tilde{S}_{t'}, \tilde{X}_{t'}^{\pi}(\tilde{S}_{t'}|\theta)) | \tilde{S}_{t+1} \right] | S_t, x_t \right] \right]$$

In this thesis, we construct two SL policies: the first is the dynamic BADP policy, a lookahead policy where the policy within the policy is a VFA-based static BADP policy. The second is a one-step lookahead with a PFA rollout policy.

Dynamic BADP (BADPD)

This is a lookahead policy where the "policy within the policy" is a VFA-based static BADP policy. Our algorithm is executed as follows:

Algorithm 3 BADPD policy

Initialize terminal condition $\bar{V}_{T+1}(S_{T+1}) = 0$, for all states S_{T+1}.

for $t = T, T-1, ...0$ **do**

  for $\tilde{t} = T, T-1, ...t$ **do**

    for $n = 1, 2, ...N$ **do**

      Sample $s^x_{\tilde{t}-1} \in \mathcal{S}^x_{\tilde{t}-1}$

      Sample $W_{\tilde{t}}$ and simulate to the next pre-decision state $s_{\tilde{t}}$.

      Compute $v_{\tilde{t}}(s_{\tilde{t}}) = max_{x_{\tilde{t}}}(C_t(s_{\tilde{t}}, x_{\tilde{t}}) + \bar{V}_{\tilde{t}}(s^x_{\tilde{t}}))$

    end for

    Create dataset $(s^{x,n}_{\tilde{t}-1}, v^n_{\tilde{t}})^N_{n=1}$ and use this to fit a linear model for $\bar{V}_{\tilde{t}-1}(s^x_{\tilde{t}-1})$.

end for

$x^*_t = argmax_{x_t}(C_t(S_t, x_t) + \bar{V}_t(s^x_t))$

The BADPD policy is a slight modification to the BADPS policy. In the BADPD policy, we execute the BADPS policy on a lookahead model at each time step in the base model, instead of implementing it once at time 0 of the base model as in BADPS.

One-step lookahead with PFA rollout policy (SL-PFA)

In this stochastic lookahead policy, the policy within our policy (also known as a "rollout policy") is the simple PFA policy described in section 4.1, with tunable parameters θ_L and θ_H. To be completely clear, our stochastic lookahead policy is implemented as follows. Letting the number of W_t samples we obtain be N, we have

Algorithm 4 SL-PFA policy

for $t = 0, 1, ...T$ **do**

Initialize $RolloutSum(S_{t+1}) = 0$ for all states S_{t+1}.

for $n = 1, 2, ...N$ **do**

Sample W_t and simulate to the next pre-decision state S_{t+1}.

$RolloutSum(S_{t+1}) = RolloutSum(S_{t+1}) + Rollout(S_{t+1})$

end for

Compute $AveRollout(S_{t+1}) = \frac{RolloutSum(S_{t+1})}{N}$

$x_t^* = argmax_{x_t}(C_t(S_t, x_t) + AveRollout(S_{t+1}))$

end for

4.4 Chapter Summary

In this chapter, we summarized the policies we designed for determining energy allocation decisions in our problem variations. Our policies span three out of four classes of policy functions which can be applied in sequential decision making problems: policy function approximations (PFA), value function approximations (VFA), and direct lookahead policies (DLA).

In the next chapter, we briefly discuss how we tuned these policies to achieve optimal performance.

Chapter 5

Parameter Tuning

Parameter tuning is an important part of policy design, which is often overlooked in literature. In this thesis, the policies with tunable parameters implemented are the PFA, SL and VFA policies. We list below the parameters that require tuning for each policy, and briefly describe each tuning process:

- The tunable parameters for the PFA policy are θ_L and θ_H. For problem variation variations involving PJM grid prices, we tune the PFA using a simple grid search. Illustrated in Figure 5.1 is a 3-D plot of how average contribution over 100 sample paths varies with different (θ_L, θ_H) pairs in a problem. Observe the concavity inherent in the objective function.

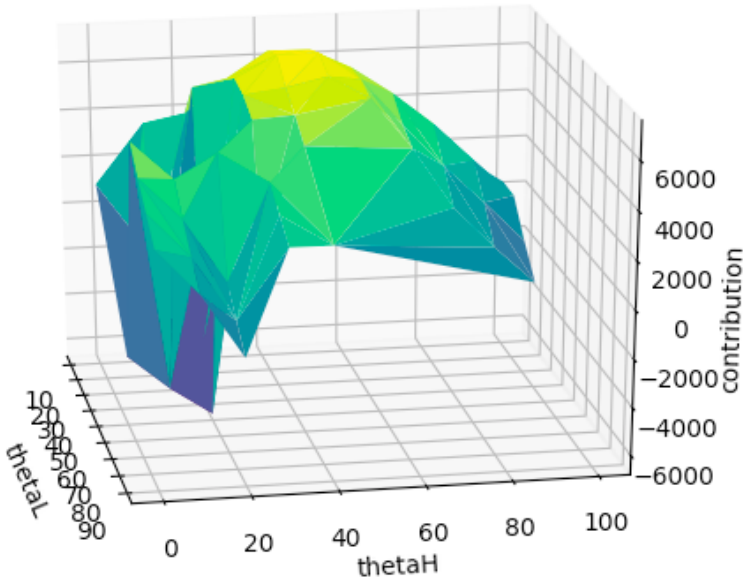

Figure 5.1: Average contribution vs θ_H and θ_L for problem with accurate wind forecasts, PJM prices, sinusoidal load

Note that no tuning is necessary for problem variations with only two possible prices in a sample path.

- The tunable parameters for the one-step lookahead with PFA rollout policy are also θ_L and θ_H. For each problem variation, we tune this policy using a simple grid search.

- The tunable parameter for both the dynamic BADP policy and static BADP policy is the sample size. We plot how increasing sample size affects mean contribution over 100 sample paths, and identify the number of sample paths which immediately precedes the "contribution plateau"- the point where mean contribution does not increase further.

Any fewer samples would compromise the algorithm's performance, and any more would waste computing resources. Figure 5.2 illustrates our tuning process for a problem:

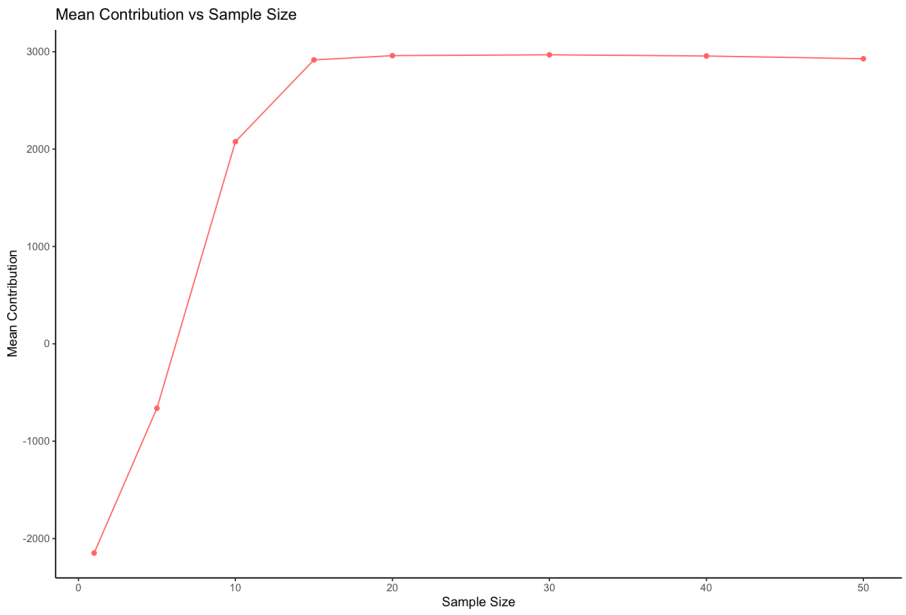

Figure 5.2: Mean contribution of static BADP vs sample size for problem with noisy wind forecasts, PJM prices, sinusoidal load. Note that the optimal number of samples which strikes a balance between policy performance and computing time is 20.

The above graph illustrates why tuning is a delicate process: small changes in parameter values can greatly change contribution values. Figures 5.3 and 5.4 further illustrate the sensitivity of policy performance to changes in parameters using a simple PFA policy applied to problems with with PJM grid prices. In each figure, we fix the θ_L value of the PFA's tuning parameter to be 10, vary θ_H between 20 and 100 and record the policy's average contribution

over multiple sample paths for each θ_H value. We present how average contribution varies with θ_H for 10, 50 and 100 sample paths. Key observations include:

- The policy's performance varies **greatly** with different θ_H values. This illustrates the sensitivity of policy performance to changes in parameters. Further theoretical work is necessary to understand why.
- The greater the number of samples we use, the smoother the contribution vs θ_H curve. This illustrates that increasing the number of sample paths used decreases noise in the sample mean contribution obtained.

Figure 5.3: Average contribution vs θ_H for problem with accurate wind forecasts, PJM prices, sinusoidal load

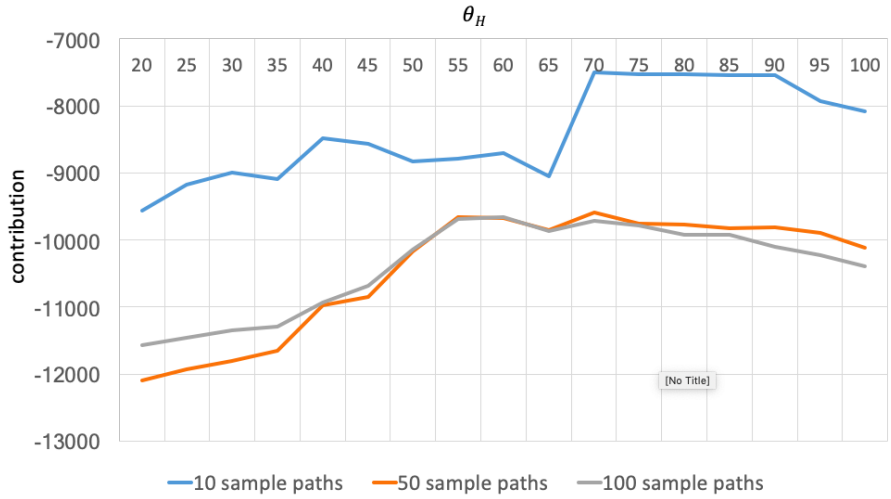

Figure 5.4: Average contribution vs θ_H for problem with noisy wind forecasts, PJM prices, sinusoidal load

5.1 Chapter Summary

In this chapter we listed the parameters that require tuning and the tuning process for the PFA, SL and VFA policies. The deterministic lookahead policies we implemented do not require tuning.

In the next chapter, we implement these policies in various problem contexts and describe their performances on each problem.

Chapter 6

Evaluating Policy Performance

For each problem variation, the policies described in the previous section are simulated using a set of 100 sample paths of wind and prices generated according to the problem context. We describe the relative performance of the policies on each problem in this chapter.

In each problem, we assume the state of charge in the battery, R_t, is discretized evenly from 0 MWh to $R^{max} = 400$, $\rho^{charge} : \rho^{discharge} = 1 : 8$, η is set to 1.0, and the battery starts out with a 0 percent charge. Each simulation has an optimization horizon of $T = 24$ as we make an energy allocation decision every hour over the course of one day. Furthermore, we assume a constant market price $M = 20$.

Figures 6.1-6.7 report each policy's performance on each problem as the average contribution over 100 trials as a percent of the average contribution earned using the posterior optimal policy for that problem. In the next chapter, we will analyse these results and discuss how they show that we can design realistic variations of our storage problem which make each of our

policies work well sometimes, and poorly sometimes.

6.1 Problem 1: Accurate wind forecasts, PJM prices, sinusoidal load

This is a time-dependent problem with sinusoidal load, daily PJM grid price patterns and very accurate wind forecasts. The error of our wind forecasts, $\sigma_E = 1$, and our load $L_t = 25sin(0.3t - 2.5) + 40$. Refer to section 3.3 for a discussion of how daily grid price patterns are modelled and how $\{f_t^P\}_{t=0}^T$ is derived.

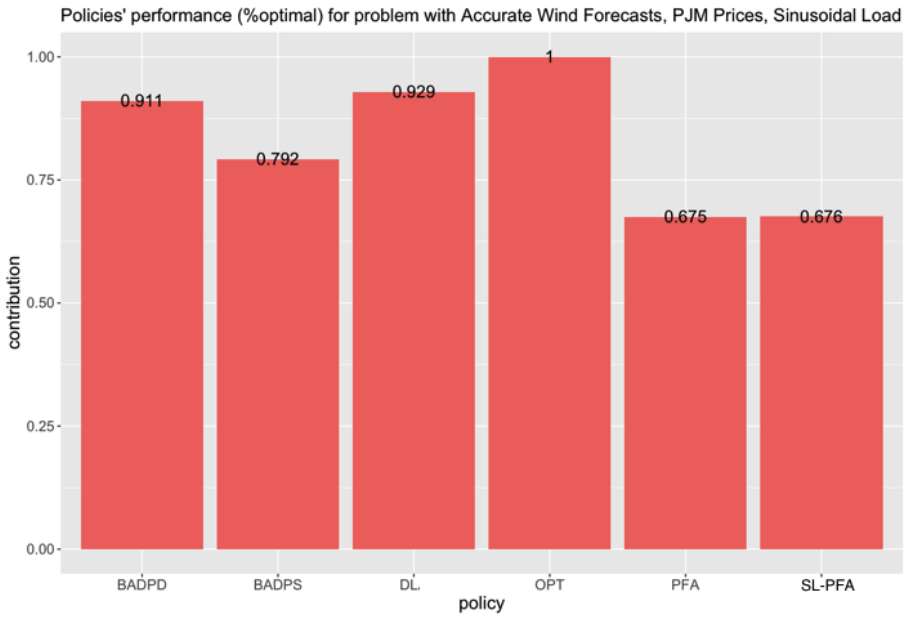

Figure 6.1: Average contribution of each policy for 100 sample paths for Problem 1

The DL policy performs best, followed by BADPD and then BADPS.

The PFA and SL policies perform the worst on this problem.

6.2 Problem 2: Noisy wind forecasts, mostly high prices, constant load

This is a stationary problem with a constant load, high grid prices most of the time and less accurate wind forecasts. At each time step t, $P_t = 90$ with probability 0.8 and $P_t = 10$ with probability 0.2. We hence use a constant weighted price forecast $f^P = 90 \cdot 0.8 + 10 \cdot 0.2 = 74$. The error of our wind forecasts, $\sigma_E = 10^4$, and our load $L_t = 25sin(0.3t - 2.5) + 40$.

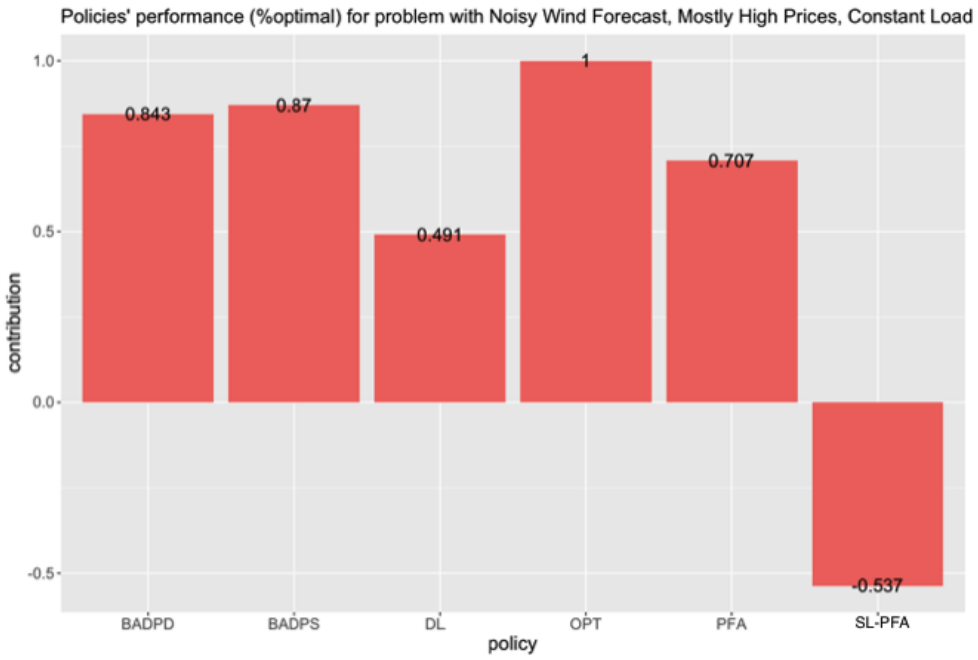

Figure 6.2: Average contribution of each policy for 100 sample paths for Problem 2

The BADPS policy performs best, followed by BADPD and then PFA. The DL and SL policies perform the worst on this problem.

6.3 Problem 3: Noisy wind forecasts, mostly low prices, constant load

This is a stationary problem with a constant load, low grid prices most of the time and less accurate wind forecasts. At each time step t, $P_t = 90$ with probability 0.2 and $P_t = 10$ with probability 0.8. We hence use a constant weighted price forecast $f^P = 90 \cdot 0.2 + 10 \cdot 0.8 = 26$. The error of our wind forecasts, $\sigma_E = 10^4$, and our load $L_t = 25sin(0.3t - 2.5) + 40$.

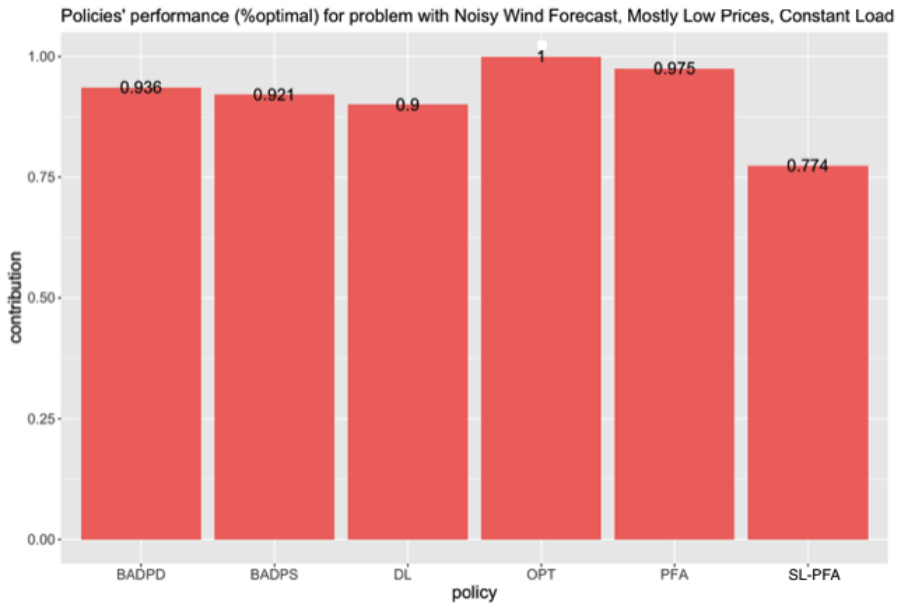

Figure 6.3: Average contribution of each policy for 100 sample paths for each problem.

The PFA policy performs best, followed by BADPD and then BADPS. The DL and SL policies perform the worst on this problem.

6.4 Problem 4: Noisy wind forecasts, PJM prices, constant load

This is a stationary problem with constant load, daily PJM grid price patterns and less accurate wind forecasts. The error of our wind forecasts, $\sigma_E = 10^4$, and our load $L_t = 20$. Refer to section 3.3 for how daily grid price patterns are modelled and how $\{f_t^P\}_{t=0}^T$ is derived.

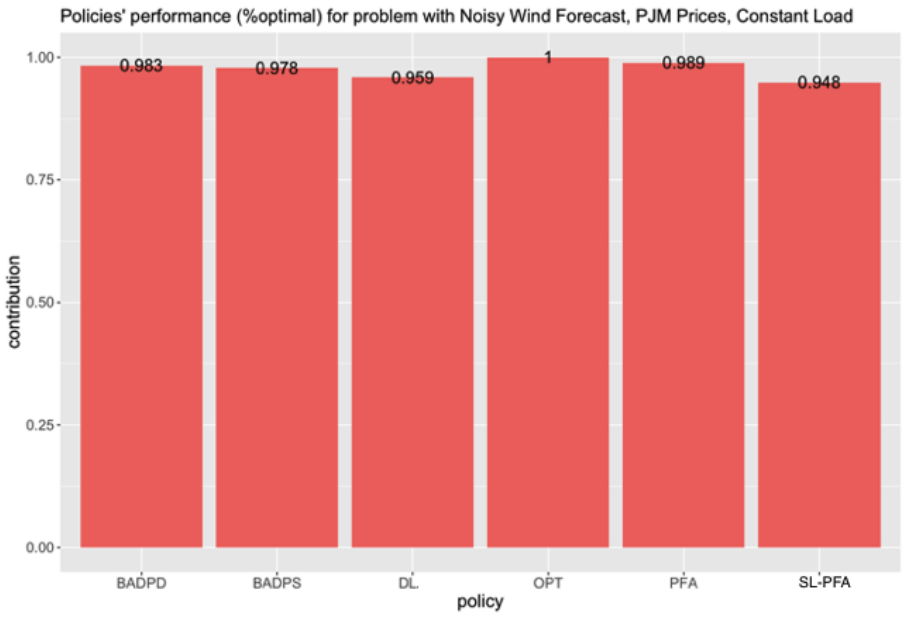

Figure 6.4: Average contribution of each policy for 100 sample paths for each problem.

The PFA policy performs best, followed by BADPD and then BADPS. The DL and SL policies perform the worst on this problem.

6.5 Problem 5: Noisy wind forecasts, PJM prices, Sinusoidal load

This is a time-dependent problem with sinusoidal load, daily PJM grid price patterns and less accurate wind forecasts. The error of our wind forecasts, $\sigma_E = 10^4$, and our load $L_t = 25sin(0.3t - 2.5) + 40$. Refer to section 3.3 for how daily grid price patterns are modelled and how $\{f_t^P\}_{t=0}^T$ is derived.

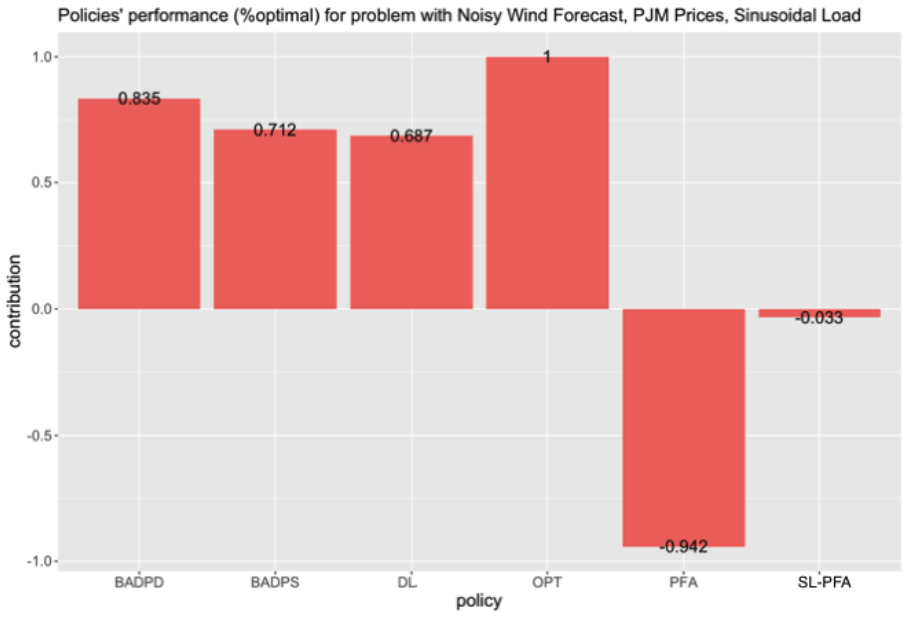

Figure 6.5: Average contribution of each policy for 100 sample paths for each problem.

The BADPD policy performs best, followed by BADPS and then DL. The SL and PFA policies perform the worst on this problem.

6.6 Problem 6: Noisy wind forecasts, Mostly high prices, Sinusoidal load

This is a time-dependent problem with sinusoidal load, high grid prices most of the time and less accurate wind forecasts. At each time step t, $P_t = 90$ with probability 0.8 and $P_t = 10$ with probability 0.2. We hence use a constant weighted price forecast $f^P = 90 \cdot 0.8 + 10 \cdot 0.2 = 74$. The error of our wind forecasts, $\sigma_E = 10^4$, and our load $L_t = 25sin(0.3t - 2.5) + 40$.

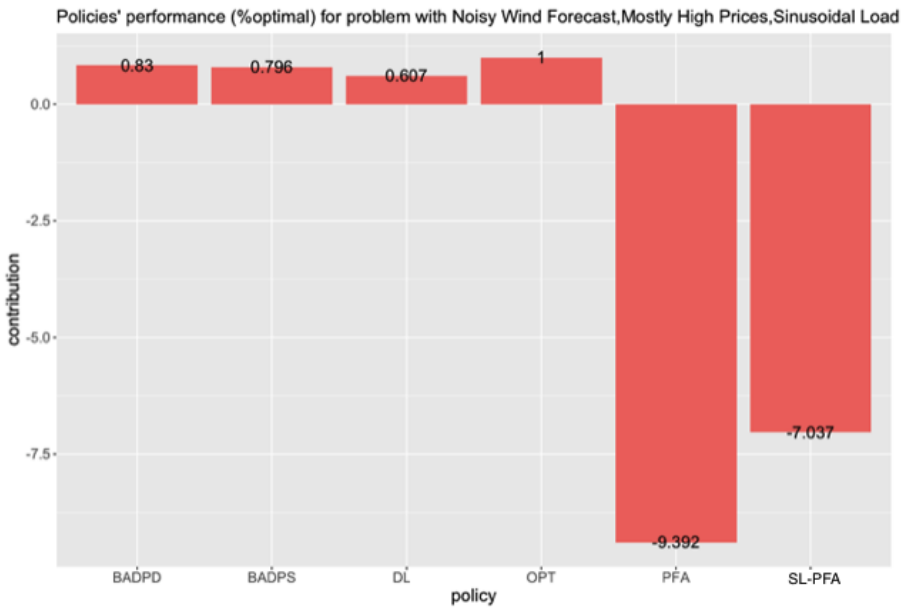

Figure 6.6: Average contribution of each policy for 100 sample paths for each problem.

The BADPD policy performs best, followed by BADPS and then DL. The SL and PFA policies perform the worst on this problem.

6.7 Problem 7: Noisy wind forecasts, mostly low prices, sinusoidal load

This is a time-dependent problem with sinusoidal load, low grid prices most of the time and less accurate wind forecasts. At each time step t, $P_t = 90$ with probability 0.2 and $P_t = 10$ with probability 0.8. We hence use a constant weighted price forecast $f^P = 10 \cdot 0.8 + 90 \cdot 0.2 = 26$. The error of our wind forecasts, $\sigma_E = 10^4$, and our load $L_t = 25sin(0.3t - 2.5) + 40$.

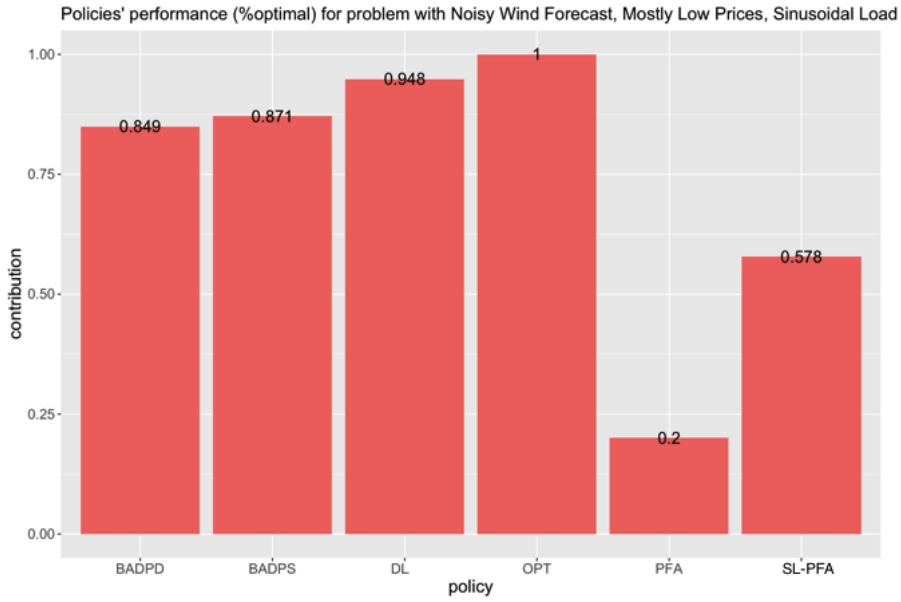

Figure 6.7: Average contribution of each policy for 100 sample paths for each problem.

The DL policy performs best, followed by BADPS and then BADPD. The SL and PFA policies perform the worst on this problem.

6.8 Summary of Results

Table 6.1 summarizes the performance of each policy on each problem variation:

-	BADPD	BADPS	DL	PFA	SL-PFA
Problem 1	0.911	0.792	0.929	0.675	0.676
Problem 2	0.843	0.870	0.491	0.707	-0.537
Problem 3	0.936	0.921	0.900	0.975	0.774
Problem 4	0.983	0.978	0.959	0.989	0.948
Problem 5	0.835	0.712	0.687	-0.942	-0.033
Problem 6	0.830	0.796	0.607	-9.392	-7.037
Problem 7	0.849	0.871	0.948	0.200	0.578
Average Contribution	**0.884**	0.849	0.789	-0.970	-0.662
Worst Case Contribution	**0.830**	0.712	0.491	-9.392	-7.037

Table 6.1: Performance of policies on problem variations

This table illustrates that we can design variations of our storage problem which make each of our policies work well sometimes, and poorly sometimes. Overall, however, the dynamic backward ADP policy with both the highest, lowest contribution and highest average contribution is the most robust across all problem variations.

In the next chapter, we will analyse these results and develop explanations for the performance of the various policies on the problem variations.

Chapter 7

Discussion

In this chapter, we will analyse which problems make each policy perform well, which render it ineffective, and why. Our analysis will also point us to theoretical and empirical extensions that would be instructive to explore in future.

7.1 Policy Function Approximation (PFA)

The PFA policy outperforms every other policy in two out of three stationary, constant load problems. It performs third best on the noisy wind forecast, mostly high prices, constant load problem, only behind both flavors of BADP policies. This comes as no surprise, since a myopic policy like the PFA would naturally work well in a stationary context when there is no need to plan ahed for future changes in demand. On the other hand, policies which lookahead may fit unnecessarily complex value functions to a simple, stationary problem where a buy low, sell high policy would work best, and

may thus underperform.

On the other hand, the PFA is the worst performing policy in all four non-stationary, sinusoidal load problems. Our load has been designed to model actual daily load patterns, where demand increases over the course of the day and then decreases later at night. A myopic policy like the PFA would be unable to forecast the need to store energy in the battery earlier in the day when prices and demand are low, so that energy can be used to satisfy high demand and generate revenue later in day. This can clearly be seen in Figure 7.1 which record a PFA's various decisions each hour, compared to the optimal policy's decisions each hour for the same non-stationary sample path. Note that while the optimal policy makes a consistent effort to store energy in the battery at earlier time steps when load is low in preparation for a high load later on, the PFA does not exhibit the same prescience.

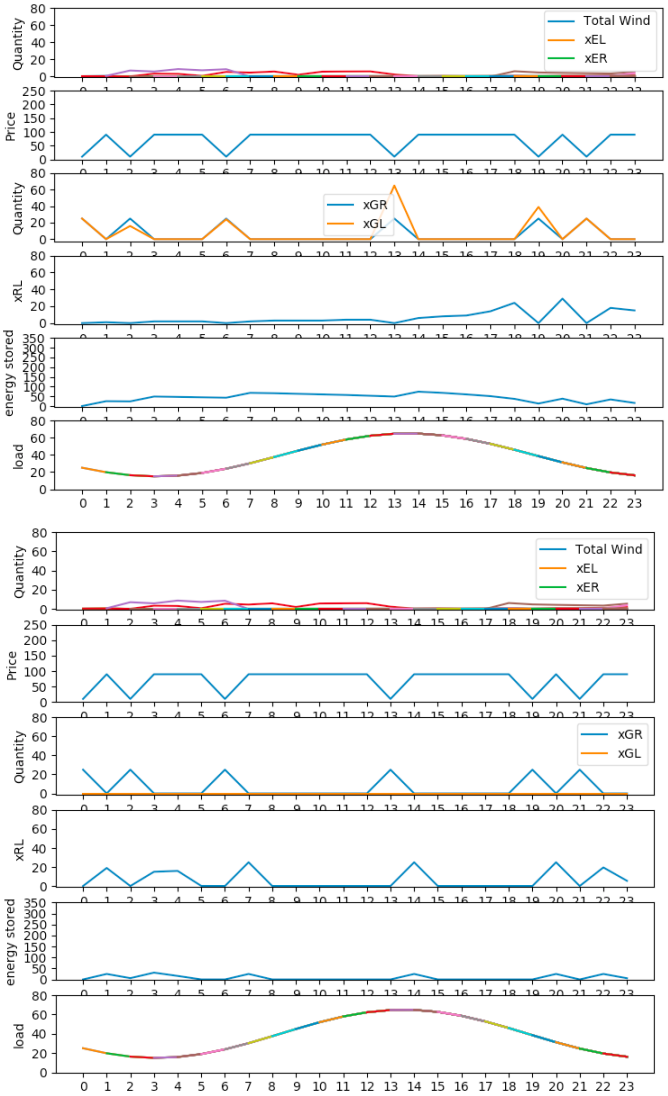

Figure 7.1: Decisions of optimal policy (top) and PFA (bottom) on sample path over 24 time steps. Observe the steadily increasing "energy stored" value from time steps 0 to 7 for the posterior optimal solution: the PFA does not exhibit the same trend.

7.2 Deterministic Lookahead (DL)

The DL policy outperforms every other policy when wind forecasts are accurate. This is natural; the DL makes decisions by optimizing over a deterministic approximation of the future based on forecasts, so as long as forecasts are accurate, the DL's decisions will be too.

On the other hand, the DL is in the bottom half of policies in terms of performance on four out of six problems with noisy wind forecasts. Its performance is particularly compromised on problems where prices are high most of the time, and low sometimes, with its contribution value being less than 50 percent of optimal for the problem with noisy wind forecasts, mostly high prices and constant load. In this problem, the DL uses a constant price forecast $f^P = 74$ for all time steps in the base model, and so does not pick up on all critical, rare time steps where prices are extremely low (i.e. 10) and buying from the grid is essential. Unlike a simple buy low-sell high policy, it misses low price points in a sample path to stock up on energy from the grid. Figure 7.2 illustrates that while a PFA consistently picks up on all three price dips in a sample path and maximises energy bought from the grid at these points, a DL misses one of them. While this might not seem costly, the scarcity of price dips in this sample path where energy can be bought profitably renders the DL's contribution to be merely 15 percent of the PFA's contribution for this sample path. A constant price forecast is thus not as effective as a simple myopic PFA at capturing the nuances of price change in this context, as shown below:

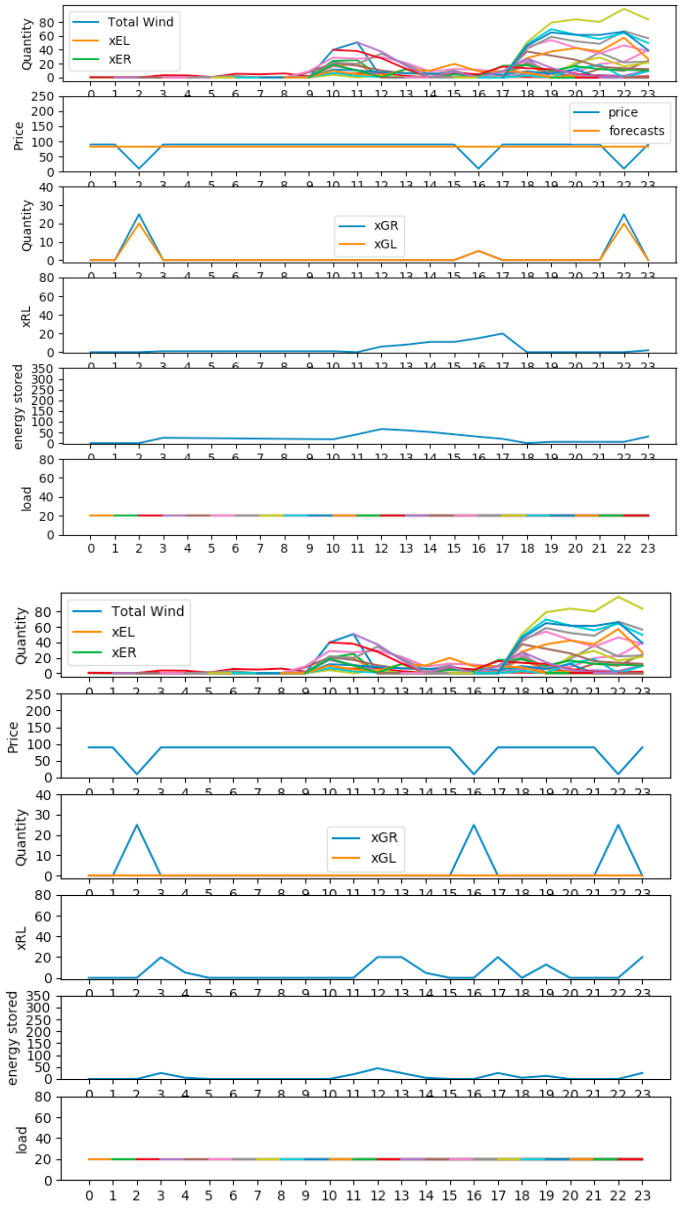

Figure 7.2: Decisions of DL (top) and PFA (bottom) on sample path over 24 time steps. Observe that there is a significant spike in x_{GR} at time step 16 for the PFA when there is a dip in grid price, while the DL has a spike much smaller in magnitude at the same time step.

An interesting exception to this trend is the class of problems with mostly low prices, where a DL seems to perform well despite noisy forecasts. For example, on the problem with noisy wind forecasts, mostly low prices and sinusoidal load, the DL outperforms every other policy. We reason that this is due to the design of the problem: there is no need for the system to specifically pick up the price points which are high out of a sequence of low prices in this variation, since there is no critical action that the policy needs to execute at these points to perform well (this is unlike the problem with high prices most of the time, where the policy *needs* to pick up on the low price points in order to be effective). All that is required of the DL policy is to be able to *not act* (i.e. not buy) at high price points, which is a much easier task than requiring the policy to *act* at high price points. The problem variation was thus likely too easy for the DL, explaining its stellar performance.

7.3 Dynamic Backward Approximate Dynamic Programming (BADPD) and Static Backward Approximate Dynamic Programming (BADPS)

BADP policies are robust in general: in every problem variation, a BADP policy is either the best, or second best performing policy. This in itself is surprising and requires further theoretical enquiry: why does a policy where approximation errors supposedly accumulate over time steps in the lookahead

model in each backward pass perform so well? With little existing literature on this topic, future work can delve into theoretical explorations of BADP policies.

Empirically, the dynamic BADP policy outperforms the static BADP policy in five out of seven problems. This is natural, since the dynamic BADP policy updates value function approximations at each time step in the base model in accordance with evolving latent wind forecasts in lookahead models, while the static BADP policy uses a single set of value function approximations obtained at time 0 using the wind forecast at time 0 to make all future decisions.

As the bar graphs in the previous chapter suggest, the difference in performance between the policies are most prominent in the problem with accurate wind forecasts, since updating forecasts improves the performance of the dynamic BADP policy when forecast noise is low. In one sample path for this problem, wind energy declines from time steps 13 to 17, while energy demand peaks during this time period. In Figure 7.3, we see how a dynamic BADP policy is able to better predict the later decline in wind energy compared to the static BADP policy by storing more energy in the battery in earlier time steps:

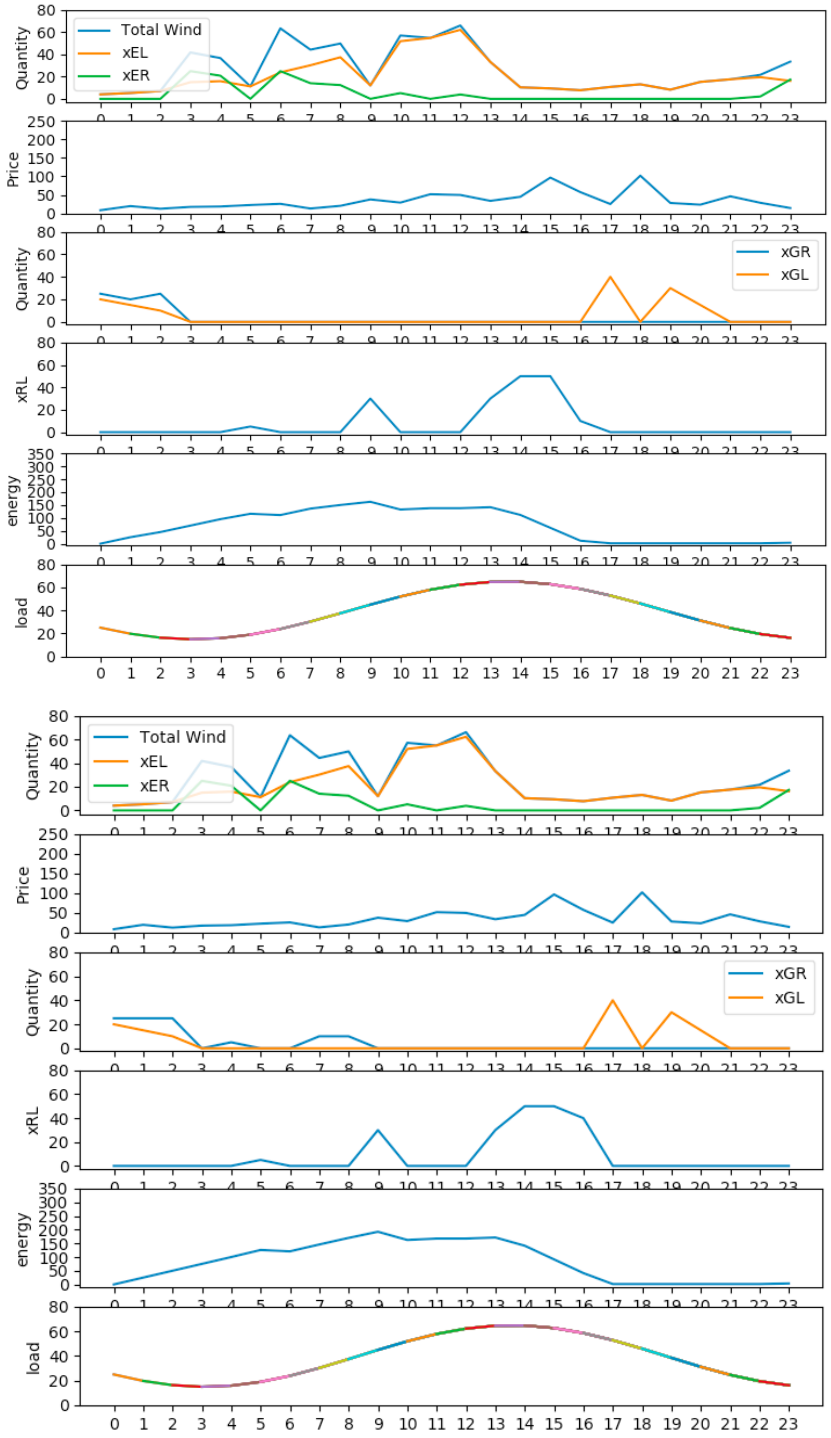

Figure 7.3: Decisions of static BADP (top) and dynamic BADP (bottom) on sample path with accurate wind forecasts over 24 time steps.

7.4 Stochastic lookahead with PFA rollout policy (SL-PFA)

The stochastic lookahead with a PFA rollout policy performs poorly in all formulations of the storage problem. We suspect that its poor performance can be attributed to the fact that a simple buy-low sell high PFA roll out policy is ineffective at approximating the value of different post-decision states, especially given the poor general performance of the PFA.

We emphasize that the poor performance of the SL-PFA policy in this work does not imply that SL-PFA policies add no value. Observe that the PFA policy outperforms the SL-PFA policy on all problems with a constant load, since as described earlier, policies which lookahead may fit unnecessarily complex value functions to a simple, stationary problem where a buy low, sell high policy would work best. However, in problems where looking ahead adds value (i.e. problems with time varying loads, where load is low earlier in the day and high later in the day), the SL-PFA policy outperforms the myopic PFA policy. This is illustrated in Figure 7.4, which compares the decisions made over the course of 24 hours by the SL-PFA policy to that of the PFA policy in a sample path with time-varying load. Observe that, unlike the PFA policy, the SL-PFA policy increases the amount of energy in the battery at earlier time steps in anticipation of the high demand at later time steps.

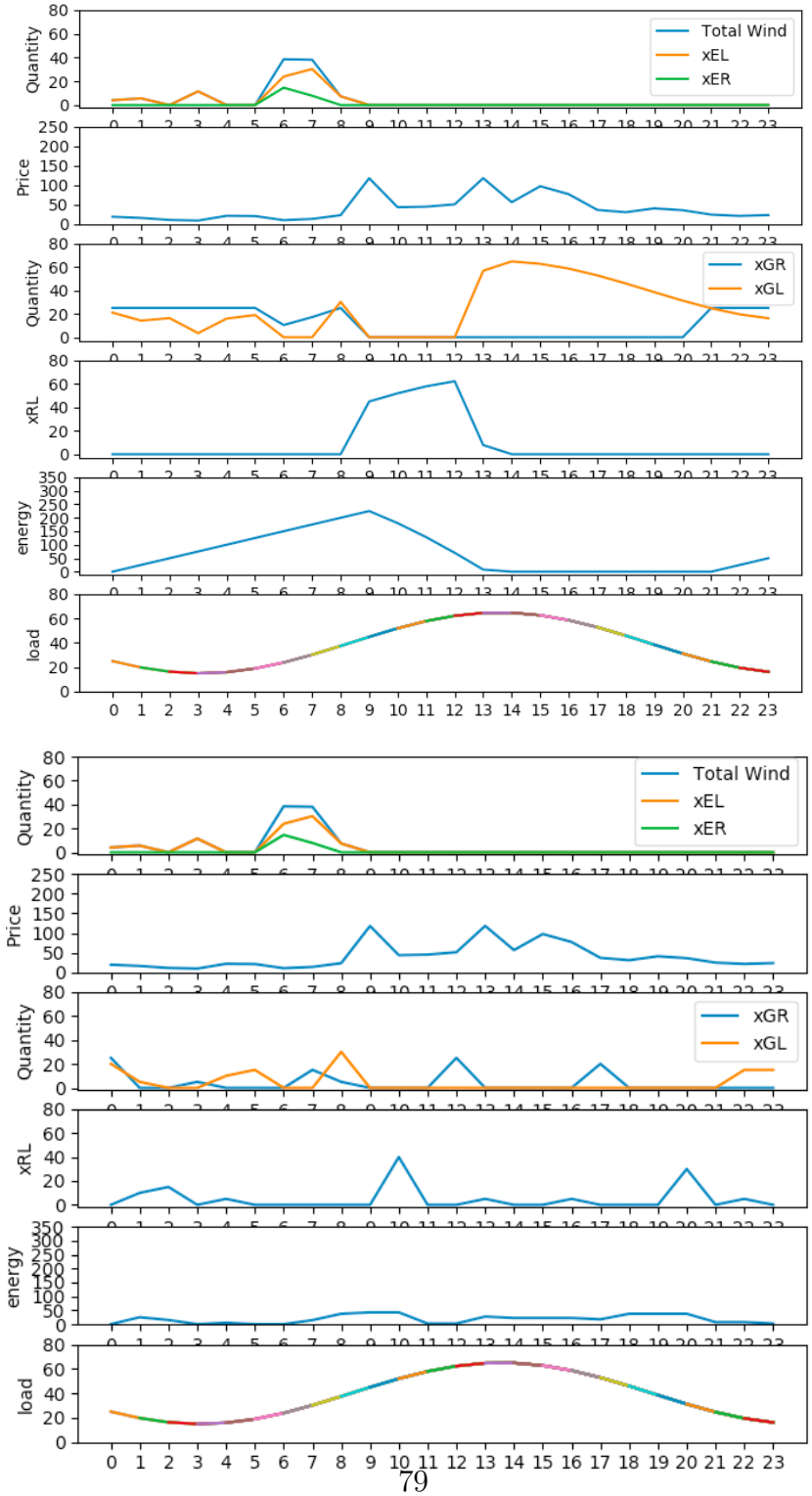

Figure 7.4: Decisions of SL-PFA (top) and PFA (bottom) on sample path with time-varying load over 24 time steps.

In order to perfect the lookahead feature of the SL-PFA policy and improve its performance, we can improve the rollout policy employed by the SL-PFA policy. The dynamic BADP policy for example, uses a more robust VFA as the policy within the policy. While the present version of our SL policy implemented our rollout policy from a node one step into the future, it might be necessary to carefully construct a deeper tree with more nodes (i.e. Monte Carlo Tree Search) in order to have a SL-PFA policy which performs well. However, more advanced SL-PFA methods like MCTS are extremely computationally expensive, and their computation costs might undermine any improvements in performance compared to other faster, robust stochastic lookahead policies like dynamic Backward ADP.

Future work can be focused on improving the SL-PFA policy, by for example, using a deterministic lookahead or VFA roll-out policy. This work illustrates the complexity of designing effective stochastic lookahead policies for the energy storage problem, and illustrates that much more work needs to be done in this arena.

7.5 Summary

Overall, every policy works well on some problem variations, and poorly on others. However, given that the dynamic BADP policy has the highest, worst case contribution across all problems, we conclude that the dynamic BADP policy is the most robust policy among all policies tested.

In the following concluding chapter, we offer a comprehensive summary of our findings, and discuss possible future directions we can take.

Chapter 8

Conclusion

In this work, we develop stochastic optimization models for variations of the following energy storage problem: how can we satisfy the energy demands of a building using a stochastic energy supply from a wind farm and a grid with stochastic prices? In our model, we are equipped with a single energy storage device to smooth the energy supply for the building.

Our results show that we can create problem variations that are either best or badly suited to each class of policies. The seven problems we created can be classified in the following ways:

- Problems with constant load vs problems with time-varying load
- Problems with accurate wind forecasts vs problems with noisy wind forecasts
- Problems with PJM grid prices vs problems with mostly high prices vs problems with mostly low prices

Our results show that PFA policies perform well on problems with constant load, and perform poorly on problems with time-varying load. This is because a myopic policy like the PFA would work well in a stationary context when there is no need to plan ahed for future changes in demand. On the other hand, they would be unable to forecast the need to store energy in the battery earlier in the day when prices and demand are low, so that energy can be used to satisfy high demand and generate revenue later in day in problems with a time-varying sinusoidal load.

In general, DL policies perform well on problems with accurate wind forecasts, and in general, poorly on problems with noisy wind forecasts. This is natural; the DL makes decisions by optimizing over a deterministic approximation of the future based on forecasts, and thus depends heavily on forecast accuracy. As long as forecasts are accurate, the DL's decisions will be too. A key exception however, is the class of problems with mostly low prices, where a DL seems to perform well despite noisy forecasts. We reason that this is due to the design of the problem being too easy for the DL: there is no need for the system to specifically pick up the price points which are high out of a sequence of low prices in this variation, since there is no critical action that the policy needs to execute at these points to perform well.

Unlike PFA and DL, BADP policies are robust in general. This in itself is surprising and requires further theoretical enquiry: why does a policy where approximation errors supposedly accumulate over time steps in the lookahead model in backward passes perform so well? Furthermore, we note that the dynamic BADP policy outperforms the static BADP policy on most problems, since the dynamic BADP policy updates value function approximations

at each time step in the base model in accordance with evolving latent wind forecasts in lookahead models, while the static BADP policy does not.

Finally, the SL-PFA policy performs poorly on all problems, but performs better than the myopic PFA on problems with time-varying load where looking ahead to plan for future energy demands adds value. We predict that with more sophisticated roll-out policies, or by constructing state nodes further into the horizon, this policy can perform even better. Granted, these will likely incur greater computational costs.

Overall, given that the dynamic BADP policy has the highest, worst case contribution across all problems, we conclude that the dynamic BADP policy is the most robust policy among all policies tested. That is, the dynamic BADP policy works most reliably across all problem variations.

Finally, in future, we can focus on the following research enquiries:

- Exploring the theoretical basis of why BADP, a policy where approximation errors supposedly accumulate over time steps in backward passes performs well across a wide variety of problem variations
- Developing problem variations where BADP policies fail, and exploring why
- Improving the computational efficiency and performance of the SL-PFA policy by either increasing the lookahead horizon or improving the roll-out policy

Bibliography

[1] Arnold, M., Andersson, G. (2011). Model predictive control of energy storage including uncertain forecasts. Power Systems Computation Conference (PSCC), Stockholm, Sweden, 23, 24-29.

[2] Cheng, B., Asamov, T., Powell, W. B. (2017). Low-rank value function approximation for co-optimization of battery storage. IEEE Transactions on Smart Grid.

[3] Durante, J. (2018). Backward Approximate Dynamic Programming with Hidden Semi-Markov Stochastic Models in Energy Storage Optimization. Submitted to Operations Research (Energy Area).

[4] Ghadimi, S., Perkins, R. and Powell, W. B. (2019). Reinforcement Learning via Parametric Cost Function Approximation for Multistage Stochastic Programming. $http://www.optimization-online.org/DB_HTML/2018/12/6999.html$. Accessed 1 Sep 2019.

[5] Han, J. E, W. (2016). Deep learning approximation for stochastic control problems. arXiv preprint. arXiv:1611.07422.

[6] Lohndorf, N., Wozabal , D., Minner, S. (2013). Optimizing trading decisions for hydro storage systems using approximate dual dynamic programming. Operations Research, 61(4), 810-823.

[7] Ma, Y., Kelman, A., Daly, A., Borrelli, F. (2012). Predictive control for energy efficient buildings with thermal storage: Modelling, stimulation, and experiments. IEEE Control Systems 32(1), 44-64.

[8] Mokrian, P., Stephen, M. (2006). A stochastic programming framework for the valuation of electricity storage. 26th USAEE/IAEE North American Conference, 24-27.

[9] Ohlenforst, K., Sawyer, S., Dutton, A., Backwell, B., Fiestas, R., Lee, J., Qiao, L. (2019). Global Wind Report 2018. Retrieved September 2019, from https://gwec.net/wp-content/uploads/2019/04/GWEC-Global-Wind-Report-2018.pdf

[10] Pham, T. V., Powell, W. B., Salas, D. F., Scott, W. R. (2014). A comparison of approximate dynamic programming techniques on benchmark energy storage problems: Does anything work? 2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL), 1-8.

[11] Powell, W. B. (2019). Sequential Decision Analytics and Modeling. In: Sequential Decision Analytics and Modeling. http://tinyurl.com/sequentialdecisionanalytics. Accessed 1 Sep 2019.

[12] Powell, W. B. (2019). Stochastic Optimization and Learning. Hoboken, NJ: John Wiley and Sons, Inc.

[13] Simao, H. P., Powell, W. B., Archer, C. Kempton, W. (2017). The challenge of integrating offshore wind power in the US electric grid. Part II: Simulation of electricity market operations. Renewable Energy, 103, 418-431.

[14] Sioshansi, R., Madaeni, S. H., Denholm, P. (2014). A dynamic programming approach to estimate the capacity value of energy storage. IEEE Transactions on Power Systems 29(1):395-403.

[15] Thalassinakis, E. J., Dialynas, E. N. (2004). A monte-carlo simulation method for setting the under frequency load shedding relays and selecting the spinning reserve policy in autonomous power systems. IEEE Transactions on Power Systems 19(4), 2044-2052.

[16] Tseng, C. L., Barz, G. (2002). Short-term generation asset valuation: a real options approach. Operations Research 50(2):297-310.

[17] Wallace, S. W. and Fleten, S. E. (2003). Stochastic programming models in energy. Handbooks in operations research and management science 10, 637-677.

[18] Warrington, J., Goulart, P. J., Mariethoz, S., Morari, M. (2012). Robust reserve operation in power systems using affine policies. Decision and Control (CDC), 2012 IEEE 51st Annual Conference on, 1111-1117 (IEEE).

[19] Zhou, Y., Scheller-Wolf, A., Secomandi, N., Smith, S. (2018). Managing wind-based electricity generation in the presence of storage and transmission capacity. Production and Operations Management, 28(4), 970989.