Reinforcement Learning for the Design and Operation of Distribution Networks

by

Aisling Karen Pigott

B.Sc., University of Colorado Boulder, 2020

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Civil, Environmental, and Architectural Engineering

2024

Committee Members:

Kyri Baker, Ph.D., Chair

Andrey Bernstein, Ph.D.

Gregor Henze, Ph.D.

Zoltan Nagy, Ph.D.

Wangda Zuo, Ph.D.

Pigott, Aisling Karen (Ph.D., Architectural Engineering)

Reinforcement Learning for the Design and Operation of Distribution Networks

Thesis directed by Prof. Kyri Baker, Ph.D.

ABSTRACT

Recent changes to the distribution electrical network such as increased consumer demand and various renewable resources have made the need to replace and upgrade electrical distribution equipment more dire than ever. This thesis introduces various frameworks for model-free, data-driven reinforcement learning to optimize energy system control across electrical and thermal energy domains. We consider control at various points of the distribution network including aggregator price setting, system operator generator control, and variations of demand response that address grid services beyond peak shaving used in conventional demand response programs (such as voltage regulation). In this thesis, we seek to address the primary concern of the combination of thermal and electrical models, ensuring the feasibility of these results for both the customer and the electric utility. Each control study that includes load shifting validates that the thermal comfort requirements of the end-user are satisfied and does so without adding extraneous hardware. Finally, we introduce a study that considers the use of reinforcement learning algorithms in distribution network design by placing distributed electrical resources at optimal points to support grid-level goals. Preliminary results indicate that reinforcement learning can be valuable for improving grid operation.

Dedication

To Fiona and Ronan, who have taught me the most.

Acknowledgements

Throughout my PhD, I have become increasingly indebted to the people who made it possible for me to achieve such a task. I am deeply grateful to my chair and advisor, Dr. Kyri Baker, for being so generous with her time and advice in creating each manuscript, preparing me for conferences, providing me invaluable connections to the power systems community, and creating a supportive microcosm of academia in the GRIFFIN Lab. Dr. Zuo has supported my Ph.D. from the beginning and welcomed me onto an exciting grant that inspired most of this work; for that, I'm tremendously thankful. To Dr. Henze, thank you for the words of support even in my undergraduate years and when I was first contemplating this path. Many thanks to Dr. Bernstein for introducing me to Q-learning and Dr. Nagy for being a generous collaborator early on.

The Architectural Engineering program at CU has become a significant part of my community over the last eight ye. I'dI'd be remiss not to thank the professors who have profoundly impacted my life, especially Jennifer Scheib, for her tenacious support of my academic and personal achievements. I am thankful for both my undergraduate and graduate cohorts, who were always there to support me and collaborate on the trickiest projects. To Jacob, thank you for spending many hours humoring my impractical ideas for research projects and helping me stay focused on the big picture. Matt and Mostafa, many thanks for always lending a friendly ear and a smile in the office. I must also thank Dr. Constance Crozier for her wisdom, guidance, and humo(u)r.

Most importantly, my parents have supported me throughout graduate school and allowed me to pursue what seemed like an insurmountable task. I owe my family an indefinite amount of thanks for the amount they have taught me well before I set foot on campus.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

Grid infrastructure in much of the US is near or surpassed the end of its design lifespan [5]. In addition to stress from aging and deteriorating equipment, changes to the end-use electric load due to the electrification of common consumer appliances and changes to the generation mix have made the current infrastructure obsolete. These changes include electrification of HVAC equipment and the introduction of electric vehicles (EVs) [38]; increased cooling demand due to climate change [4]; and the mass adoption of distributed energy resources (DERs) with bidirectional power flow such as photovoltaics (PV) and high-powered fast-chargers for EVs [16]. They have elevated the need to replace and upgrade electric infrastructure at the transmission and distribution level (e.g., lines, transformers, capacitor banks). Recently, US policymakers have passed legislation that allows for significant upgrades to electric infrastructure to increase US energy security and reduce greenhouse gas emissions (GHGs) [2]. While hardware upgrades to infrastructure are becoming increasingly urgent to meet reliability and emissions goals, software solutions can be used in addition to physical infrastructure upgrades as a cost-effective solution to mitigate strain and improve the efficiency of electrical installations.

The electric grid is composed of many stakeholders who operate with various levels of control. The amount of oversight given to each consumer (e.g., centralized or decentralized control) varies, as does the individual objective (e.g., minimizing cost or maximizing reliability). While some of these objectives naturally align, others naturally conflict. To that end, stakeholders may seek to influence the actions of one another. For example, utility operators may add time of use or dynamic pricing

to the consumer's utility bill to align their goal of improved reliability with the consumer's goal of reduced cost. Additionally, stakeholders may cooperate if it is in their best interest: consumers in a distribution network might cooperate to avoid overloading distribution equipment and causing a catastrophic blackout.

This proposal introduces several approaches to improving grid operations using reinforcement learning. First, we show preliminary success in multiple facets of operating the power grid, such as price design and voltage regulation. Next, we expand the scope of our research into the impact of the interaction between multiple RL agents, especially those designed by different users, and including RL-based design strategies that can determine the most effective hardware upgrades when resources are limited.

In general, the objectives of the grid can be categorized as cost-saving, efficacy, and/or reliability improvements. Each of these objectives for operating power systems is subject to the physical constraints of generator and equipment dynamics and the transmission system. In this thesis, we consider the efficiency of the system to be the utility provided to the end-user in relation to the energy injected at the generation point. To that end, when considering efficiency goals, we consider additional constraints that ensure the comfort and safety of the occupants, namely the thermodynamic constraints of the coupled HVAC systems, as a measurement of the utility. This reflects that the grid is largely designed to be responsive to demand loads from utility customers.

1.1 Research Questions

The list of research questions related to improving the electric grid is extensive, and many studies have been done to address any of the aforementioned existing objectives for the grid. However, the proposed research in this document should address the following research questions specifically:

- To what extent can control strategies of both grid-side and demand-side infrastructure be implemented to reduce strain on the electric grid (with minimal hardware)?

- Can RL be used to implement hardware design strategies that support grid objectives in addition to operational changes (controls)?
- How would widespread adoption of machine learning based approaches affect the homogeneity or heterogeneity of home energy management?

1.2 Expected Outcomes

In addressing the aforementioned research questions, we primarily expect to find that the control strategies currently implemented in power systems underutilize the existing infrastructure. For example, building owners can leverage their buildings to store energy and shift demand to reduce strain on the grid without compromising their physical comfort in terms of thermal constraints. Grid operators can add secondary controls using RL that anticipate voltage deviations across the network and can do so without changing the ramping limits on existing generators. With limited hardware improvements, namely the inclusion of communication hardware, we will show that the grid can be operated more cost-effectively, safely, and reliably. More specifically, we expect to find:

- That we can improve grid performance using limited data that protects the privacy of the consumer (consumption) data and the grid's topology, such as local voltage measurements and aggregate demand data.
- That we can effectively use RL to support grid-level goals not just in the operational phase but in the design phase to intelligently allocate distributed resources such as capacitor banks and batteries.
- That centralized and homogenized methods of optimizing consumption are idealized scenarios for demand response capabilities, although decentralized and heterogeneous systems can improve existing conditions without forfeiting consumer data privacy.
- That multiple open-source platforms presented here for studying new control paradigms can continue to be used by demand response researchers.

The remainder of this thesis is organized as follows: Chapter 2 is a discussion of modeling methods and machine learning tools used in this proposal, Chapters 3, 4, and 5 are a discussion of the results of ML tools for optimizing grid performance via active control algorithms. Chapter 6 discusses optimizing grid performance through design. Finally, Chapter 7 introduces a competition platform used to teach young engineers about the use of RL in energy systems.

Chapter 2

Methods

2.1 Energy Models

Firstly we acknowledge the complexity of building accurate and precise models. One of our objectives in this thesis is to broaden the scope of the constraints, or models, of each of our previously acknowledged research problems. Particularly in scenarios with multi-domain systems (e.g. mechanical building HVAC systems coupled with the electric grid network) the literature is sparse and contains models that focus on one domain with great specificity and the other as a course model. Therefore in this section, we address the level of complexity of the models comprising our various case studies. Table 2.1 summarizes the energy modeling tools used in the remaining chapters.

Software	**Electrical Domain**	**Thermal Domain**	**Time Domain**	**Language**
DRAGG	Demand-side	Single-zone Building	Minutes	Python
EnergyPlus	Demand-side	Large Buildings	Minutes	–
pandapower	Transmission	-	Steady-state	Python
PowerModelsDistribution.jl	Distribution	-	Steady-state	Julia
Modelica	Demand, Dist., Trans.	Any	Transient	Modelica

Table 2.1: Summary of energy modeling tools used

2.1.1 Thermal Energy Modeling Tools

For residential, low-voltage distribution studies, we have developed the Python package Distributed Resource AGGregator (DRAGG) for linearized R1-C1 thermal models of houses and linearized models of other electric devices such as water heaters, batteries, and PV arrays. The DRAGG package provides several advantages over non-native Python packages, such as Python APIs for EnergyPlus: the availability of a model with convex (linear) constraints allows for computationally simple optimization problems. The homeowners' decision-making can therefore be assumed to be optimal according to cost-savings or other objective measures and modeled accordingly. Furthermore, as the computational cost is low DRAGG implements both multi-threading for parallel computing if the simulation is centralized as well as a message broker (Redis) that allows for individual stakeholders (each homeowner and the aggregator) to operate on separate networked computers (e.g., they may be simulated on any internet-connected device). Read/write privileges for participants are intended to preserve the actual mechanics of the distribution network where homeowners can write their demand to aggregator and the aggregator can read only the total demand of the homeowner rather than the entire set of status values for occupants (i.e. occupancy status or indoor air temperature).

R1-C1 models can provide a similar root mean squared error (RMSE) as higher fidelity thermal models, but typically predict a more dampened response to quick changes in external environmental conditions than higher fidelity models [25]. Therefore they are generally only considered suitable for single-zone thermal models, such as stand-alone single-family homes. Since R1-C1 models are in general only suitable for single-zone thermal models EnergyPlus simulations are used to determine the electrical demand for the remaining large, commercial buildings. However, EnergyPlus is non-native to Python and therefore the demand is typically pre-calculated in an open loop. Several assumptions and adjustments are made in Chapter 5 to approximate a closed-loop control system.

2.1.2 Electrical Energy Modeling Tools

Once the electrical demand is calculated per building, the network's state (generator dispatch, line currents, and bus voltages) can be determined via analysis or optimization. For a general analysis of the system's performance over a long time span, a steady state analysis captures the typical performance and is used by utility operators for dispatching purposes. The Python package pandapower provides AC power flow and AC optimal power flow solutions for balanced systems. For unbalanced systems, as is typical in distribution feeders, we use PowerModelsDistribution.jl to provide the optimal power flow solutions.

Between timesteps at which various grid-level controls such as generator setpoints are dispatched grid resources use local control to balance the grid. Local controls which typically operate on PID or droop control are sufficient to account for small fluxuations in demand or generation. However, drastic changes can require contingency planning to understand how the grid will respond when stressed. Contingency scenarios can include generator outages, line faults, or other blackout conditions.

In contingency scenarios, a transient (sub-second) analysis is necessary. Dynamic power simulations can be done in the Modelica package OpenIPSL. Dynamic simulations from OpenIPSL must be paired with initialization conditions taken from a steady-state analysis (e.g., pandapower). Additionally, since there is an array of available open-source packages for Modelica, Modelica can couple the thermal demand with electric demand and distribution (e.g., using the Buildings library).

In the following chapters, as we will discuss various scenarios that require each energy modeling tool, we will identify key requirements and allowable assumptions of each scenario and how they have led us to choose each energy modeling tool where appropriate.

2.2 Introduction to Reinforcement Learning

Reinforcement learning (RL) is a form of machine learning based on the premise that machine learning agents can learn to make intelligent control decisions with only implicit feedback in the

form of a reward. This contrasts with supervised learning in that the best solution is generally not provided a priori and unsupervised learning in which the agent would uncover a relationship between samples. RL is, therefore, generally preferred over supervised learning when it is difficult to sample with many different scenarios and for control problems in which unsupervised clustering is not necessarily useful. To gain new insights the RL agent "explores" its environment by trying new combinations of state-action pairs, if the new state-action pair is lucrative the RL agent can "exploit" that information at a later timestep. The conundrum of balancing exploration with exploitation is known as the "exploration-exploitation trade-off" and is central to designing an appropriate RL agent.

RL is also preferred over conventional optimization algorithms when the constraints are difficult to model (i.e. non-convex, lengthy, or unknown). Despite many advancements in optimal power flow (OPF) formulations through various relaxation techniques, OPF problems are still time-consuming and resource intensive to solve. Additionally, the most common relaxation technique, DC OPF, creates "solutions" that are entirely infeasible [3]. Therefore in this dissertation, we look to RL to improve the performance of the electrical grid while lowering the computational cost.

2.2.1 Mechanics of Reinforcement Learning

In RL, the agent interacts autonomously with the environment in an agent-environment cycle as shown in Fig 2.1. At every timestep, the agent proposes an action which is then applied to the environment. The environment which is modeled as a black box, as the agent is unaware of any mechanics of the environment, returns a new state. From the new state, the agent gains a new observation (a subset of the environment's new state) and a reward (that may or may not be directly derived from the observable state). The RL agent's goal is to develop a model that correlates the state-action pair directly to the current observed reward as well as the estimated future reward; the combination of these two values is given by $\mathbf{Q}$. Since the future reward is dependent on the future actions selected, let Q^{π} be the Q-value under the policy of action selection π. In other words, the current action should enable success in the future as well as in the present as determined by the

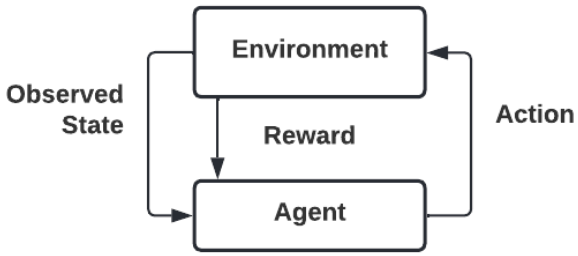


Figure 2.1: Interaction between agent and environment at each timestep

Bellman's operator (Eq. (2.2).

$$Q^{\pi}(s^{(t)}, a^{(t)}) = \mathbb{E}_{\substack{s^{(t+1)} \sim P \\ a^{(t+1)} \sim \pi}}[R(s^{(t)}, a^{(t)}, s^{(t+1)}) + \gamma Q^{\pi}(s^{(t+1)}, a^{(t+1)})] \tag{2.1}$$

$$= \mathbb{E}_{s' \sim P}[R(s^{(t)}, a^{(t)}, s^{(t+1)}) + \gamma V(s^{(t+1)})] \tag{2.2}$$

Given a deterministic transition function between $s^{(t)}$ and $s^{(t+1)}$ the exact future performance could **theoretically** be calculated. (In practice, RL is often limited by the availability of rich state data, which captures all factors that influence the state transition, the ability of the neural networks to capture the nuance of the state transition, and the stochasticity of the state transition.) Given that RL is a technique for learning to control black box environments, the future Q-value is recursively estimated.

2.2.2 Variations of RL

Various improvements to the vanilla RL algorithm have been proposed since its introduction. Deep RL, in which the Q-function and others are modeled via neural networks as opposed to function approximators or tabular scores, is a particularly common variant due to the ability of neural networks to capture complex dynamics.

Other variants are largely aimed at reducing the erratic initialization behavior of the agent. Several of the variants of reinforcement learning used in this paper include Soft-Actor Critic (SAC)

and Proximal Policy Optimization (PPO). These variants include several methods to improve performance, as follows for SAC [24]:

- Value Networks: To avoid calculating the policy-weighted average of the expected Q-value, the agent additionally learns the expected value, known as the "value"-network, V.
- Entropy Regularization: The RL agent is additionally rewarded by the entropy in the system, which promotes RL agents that retain as much exploration as possible throughout. This functionally changes the reward from the user-defined signal to the user-defined signal *plus the observed entropy*. This changes the Q-function from Eq. (2.2) to Eq. (2.4).

$$Q^{\pi}(s^{(t)}, a^{(t)}) = \mathbb{E}_{\substack{s^{(t+1)}\sim P \\ a^{(t+1)}\sim\pi}}[R(s^{(t)}, a^{(t)}, s^{(t+1)}) + H(\pi(\cdot|s^{(t)})) + \gamma Q^{\pi}(s^{(t+1)}, a^{(t+1)})] \tag{2.3}$$

$$= \mathbb{E}_{s^{(t+1)}\sim P}[R(s^{(t)}, a^{(t)}, s^{(t+1)}) + H(\pi(\cdot|s^{(t)})) + \gamma V(s^{(t+1)})] \tag{2.4}$$

The primary change for PPO is the addition of policy clipping during each policy update [51]. The KL divergence of the policy is clipped to reduce the variance during exploratory phases. First, we define the advantage of the action as the difference between the reward from that action and the expected reward from the state; the action is **advantageous** if the observed reward is greater than the expected reward of a random action taken in that state. Given that RL quantifies the reward over an infinite horizon as Q and the expected reward over the infinite horizon as V, the advantage is defined as:

$$A^{\pi_\theta} = Q(s^{(t)}, a \sim \pi_\theta(\cdot|s^{(t)})) - V(s^{(t)}) \tag{2.5}$$

We then use the advantage to determine when and how to update the policy: first clipping the updated policy to be "close" to the old policy and then evaluating the new policy for improvement. The policy only updates if the new policy weight vector θ is an improvement on the existing one, θ_k. In Eq. (2.7) let L define the potential improvement of the proposed policy, clipped by Eq. (2.8).

$$\theta_{k+1} = \arg\max_{\theta} \mathop{\mathbb{E}}_{s,a\sim\pi_{\theta_k}} [L(s, a, \theta_k, \theta)] \tag{2.6}$$

$$L(s, a, \theta_k, \theta) = \min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s, a), \quad g(\epsilon, A^{\pi_{\theta_k}}(s, a))\right) \tag{2.7}$$

$$g(\epsilon, A) = \begin{cases} (1+\epsilon)A & A \geq 0 \\ (1-\epsilon)A & A < 0. \end{cases} \tag{2.8}$$

Both RL variants discussed here prevent the RL agent from getting stuck on a single promising solution, like a local maximum, by being conservative with policy updates. SAC rewards a policy that does not "exploit" new solutions; PPO explicitly prevents the policy from changing too fast.

Primitive forms of RL include the k-armed bandit problem in which an agent learns to select one or more actions to receive the maximum reward value. Environments for k-armed bandits are stateless, and actions taken at time t have no effect on the environment or the state-reward relationship at later timesteps.

Chapter 3

For the Utility Aggregator: Real-Time Pricing Signals

In communities with high renewable resource penetration, eliminating the duck curve has become a top priority for grid operators. As discussed in Chapter 1 conventional strategies for demand response include the utility sending a real-time signal (commonly a price signal that changes electricity cost) to the consumer to encourage a shift in demand. Some electric utilities have implemented time-of-use (TOU) pricing to address this with daily pricing tiers, while others (especially for the commercial market) have implemented dynamic pricing based on the locational marginal price (LMP) where the LMP represents the real cost to the utility of delivering power to a consumer. However, static TOU pricing structures are not adaptable to variations in daily conditions such as PV generation and must be continually updated by the utility to reflect the makeup of the current community [49]. Other pricing structures such as those based on LMP [13] are difficult to determine given that the LMP is not typically calculated for distribution level nodes and has the potential for creating unfairness from marginal losses [28] or discrepancies when the LMP is derived from the DC OPF rather than the AC OPF [60].

In addition, though the consumer might wish to reduce their overall cost, their response to the price signal change is limited by the thermal response of the building and the occupants desire to stay within their preferred thermal bounds. Adding a positive price signal to the consumer's electricity cost when the aggregate demand is dangerously close to the line limit will only work when the home is pre-conditioned and available to load-shed. Therefore price signals must be designed to encourage pre-cooling and pre-heating in order to see the full impact of encouraging load shed

at specifically requested periods.

Prior work has been performed exploring the use real time prices to control distributed energy resources in residential markets [41] and even in using reinforcement learning (RL) to design these price signals[36, 31, 20]. Models of HEMS such as those in [41, 36, 31, 20] assume that community members are willing to cooperate by sharing information about their desired demand or even more revealing information such as how much load they are able to shift. Many of the HEMS models reviewed neglect realistic demand response models by using a set percent of the demand as a load shift resource with a discounted consumption rate [31, 20]. However, if the environment can be fully modelled by a single linear constraint the full advantages of RL – as opposed to more conventional online learning algorithms such as projected gradient descent – are not realized. By utilizing an RL framework we may create a model with constraints that are unknown, non-convex, and changing.

(1) We do not require home owners to include any bid information other than their actual demand and we do not require home owners to commit to their demand in advance. Since the aggregator is naive to individual home constraints and even consumption each home owner receives the same price signal from the aggregator.

(2) We include thermal constraints and linear models for the HVAC and domestic water heaters of each home as in [32, 29]. These models are a convex approximation only for the purposes of MPC but may be replaced with non-convex models or simply measured data when implemented in a real testbed.

(3) We consider the use of transfer training in which an RL agent can be trained off-line in a simulated environment before being deployed in real communities.

In the remainder of this chapter we introduce the mathematical models for the home energy management system and corresponding devices, we then consider the application of these MPC models to RL-control of real time price. The work discussed in this chapter is a result of the publication [46].

3.1 Distributed Resource Aggregator (DRAGG)

In order to determine the limitations of the building owners' response to a price signal the buildings must be modeled. To this end we utilize a linear R1C1 model of a single zone single family home and model predictive control (MPC) to determine the homeowner's actions at each timestep. Conservatively, we use strict bounds on the thermal deadband for both the indoor air temperature and water heater. The MPC model of the homes is therefore formulated as follows:

$$\min \quad \sum_{t=0}^{\tau} \beta^t c_e^{(t)} P_{grid}^{(t)} \tag{3.1}$$

$$\text{s.to} \quad P_{grid}^{(t)} = P_{load}^{(t)} - P_{PV}^{(t)} \tag{3.2}$$

$$P_{load} = P_{HVAC} + P_{wh} + P_{battery} + P_{EV} \tag{3.3}$$

where $c_e^{(0)} = c_{e,base}^{(0)} + c_{e,reward}^{(t)}$ and $c_e^{(t)} = c_{e,base}^{(t)} \forall t > 0$.

The thermal model of the home indoor air temperature is given as[1] :

$$T_{in}^{(t)} = T_{in}^{(t-1)} + \left(\frac{T_{OAT}^{(t)} - T_{in}^{(t-1)}}{R_i C_i} + \frac{k}{C} I^{(t)} + \frac{\eta}{C} Q_{HVAC}^{(t)} \right) \Delta t \tag{3.4}$$

$$T_{in}^{min,(t)} \leq T_{in}^{(t)} \leq T_{in}^{max,(t)} \tag{3.5}$$

$$Q_{HVAC}^{(t)} = S_{heat}^{(t)} Q_{heat} - S_{cool,i}^{(t)} Q_{cool,i} \tag{3.6}$$

$$P_{HVAC}^{(t)} = \eta^{-1} Q_{HVAC}^{(t)} \tag{3.7}$$

where Eq. (3.4) represents the state transition of the indoor temperature dependent on the current outdoor air temperature ($T_{OAT}^{(t)}$), global irradiance ($I^{(t)}$), and thermal power from the HVAC equipment ($Q_{HVAC}^{(t)}$) bounded by the time-varying thermal bounds in (3.5).

Similarly, the state transition of the hot water tank is given in Eq. (3.8) and is bounded in (3.9), where the external environmental temperature is the indoor temperature from (3.4). We assume for the resistive electric heater that the efficiency is 100%. Additionally, the water heater

[1] The thermal model in DRAGG has since been updated since publication to include a radiative heat gain term. This energy model is used in DRAGG-v2.0.0 and later whereas it is omitted in the version published in [46].

accounts for water draws as recorded in [1]. We model the water draws as closed-loop feedback where it occurs between the controlled timesteps (e.g. at time $t - 0.5$).

$$T_{wh,i}^{(t)} = T_{wh,i}^{(t-0.5)} + \left(\frac{T_{in,i}^{(t)} - T_{wh}^{(t-1)})}{R_{wh}C_{wh}} + \frac{1}{C_{wh}} Q_{wh,i}^{(t)} \right) \Delta t \tag{3.8}$$

$$T_{wh,i}^{min} \leq T_{wh}^{(t)} \leq T_{wh}^{max} \tag{3.9}$$

$$T_{wh,i}^{(t-0.5)} = \frac{\zeta_{wh} - d^{(t-1)}}{\zeta_{wh}} T_{wh,i}^{(t-1)} + \frac{d^{(t-1)}}{\zeta_{wh}} T_{tap}^{(t-1)} \tag{3.10}$$

$$Q_{wh}^{(t)} = S_{wh}^{(t)} Q_{wh} \tag{3.11}$$

$$P_{wh}^{(t)} = Q_{wh}^{(t)} \tag{3.12}$$

For both systems the control variable $S_{sys}^{(t)} \in \{0, ..., 1\}$ represents the duty cycle of the system during the timestep. By providing a discrete value we ensure that the system is not cycling so rapidly as to unnecessarily deteriorate the equipment.

Additional models available in DRAGG are home battery systems, electric vehicles (EV), and photovoltaic (PV) arrays. The home battery system and EV models track the state of charge as follows:

$$E^{(t)} = E^{(t-1)} + \left(\eta_c P_c^{(t)} - \eta_d^{-1} P_d^{(t)} \right) \Delta t \tag{3.13}$$

$$0 \leq P_c^{(t)} \leq P_c^{max} \tag{3.14}$$

$$0 \leq -P_d^{(t)} \leq P_d^{max} \tag{3.15}$$

$$0 \leq E^{min} \leq E^{(t)} \leq E^{max} \leq E^{cap} \tag{3.16}$$

where the state of charge in (3.13) is given by $E^{(t)}$ and the power for charging or discharging is given by $P_c^{(t)}$ and $P_d^{(t)}$ respectively. While the charge and discharge variables are separated to accommodate different efficiency values η_c and η_d, the formulation guarantees against the simultaneous charge and discharge while the cost of electricity is positive [23]. Finally, in Eq. (3.16) the state of charge at each timestep is constrained between the desired minimum and maximum charge level.

Additionally when utilized as an EV battery, the discharge is constrained to discharge when

the EV is traveling and neither charge nor discharge when the EV is parked away from home:

$$P_d^{(t)} = P_{d,drive}^{(t)} + P_{d,grid}^{(t)} \tag{3.17}$$

$$P_{d,drive}^{(t)} = E_{trip}(\Delta t_{trip})^{-1} \qquad \forall\, t \in \{t_{departure}, t_{arrival}\} \tag{3.18}$$

$$P_{d,drive}^{(t)} = 0 \qquad \forall\, t \notin \{t_{departure}, t_{arrival}\} \tag{3.19}$$

$$P_{d,grid}^{(t)}, P_c^{(t)} = 0 \qquad \forall\, t_{departure} \leq t \leq t_{arrival}. \tag{3.20}$$

We introduce the variables $P_{d,drive}$ and $P_{d,grid}$ to denote discharge for the purpose of driving and grid respectively and more easily add travel-dependent constraints.

Lastly, DRAGG contains a PV model for curtailment (u). The power produced by the array (P_{PV}) is given by the following:

$$P_{PV}^{(t)} = A_{PV}\, \eta (1 - u^{(t)}) I^{(t)} \tag{3.21}$$

$$0 \leq u^{(t)} \leq 1 \tag{3.22}$$

where A_{PV} is the area of the array, η is the efficiency of the array, and $I^{(t)}$ is the solar irradiance.

DRAGG is open source and available as a Python package via PyPI or at `https://github.com/apigott/dragg`.

3.2 Methods

Using the home energy models in 3.1 we create a community of MPC-controlled houses connected by an aggregator RL agent. The interaction of the community is depicted in Fig 3.1: at each timestep the current and predicted environmental conditions are broadcast to

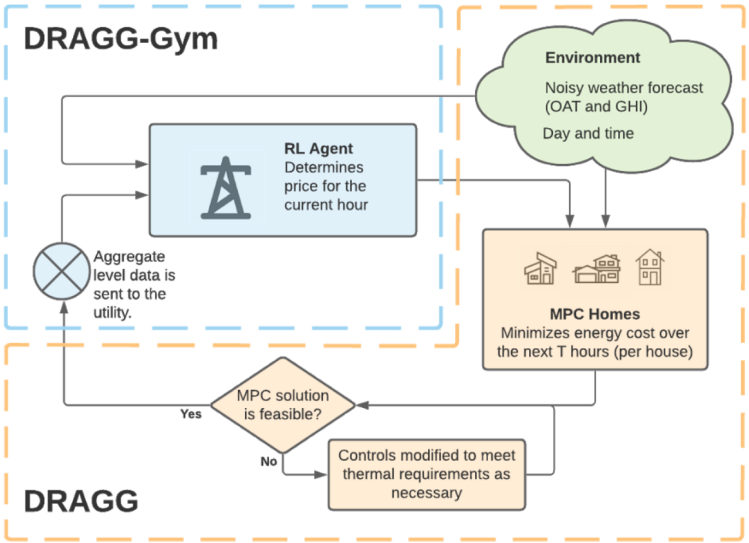


Figure 3.1: Interaction of MPC controlled home energy management systems and RL controlled price design

each HEMS and the RL aggregator and the HEMS attempt to solve an MPC calculation for the forecast conditions. If the HEMS is unable to find a feasible solution to the system-level problem then each device falls back on (1) the lowest-cost solution that satisfies the thermal constraints or (2) the solution that most closely achieves the thermal constraints (in satisfying the thermal constraints makes the device level problem infeasible). When each home is done solving its own optimization problem the aggregator receives the aggregate demand load which it may use in conjunction with the weather forecast to select a price signal.

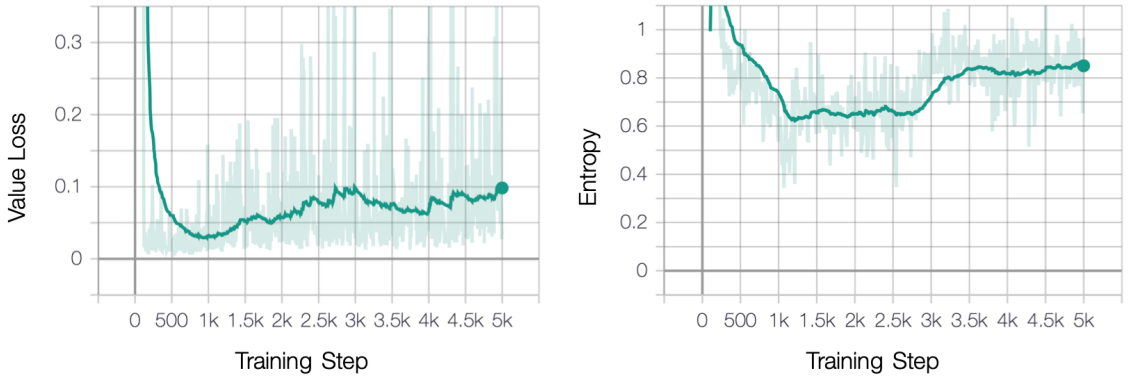
Recall from Section 2.2 that the Q-function is extrapolated from observed rewards at past state-action pairs and the expected future Q-values. Once an action is selected to maximize the expected value of Q the reward price signal is sent to the MPC responsive homes and carried out in the next timestep. The resulting state is evaluated by the RL agent with the reward function in (3.23), based on the aggregate demand.

$$r(\hat{x}_t) = -\left(\frac{p_{grid}^{(t)}}{\bar{p}_{grid}}\right)^2 + \left(\frac{p_{grid}^{(t)}}{\bar{p}_{grid}}\right) \tag{3.23}$$

The reward function presented here promotes solutions where the community consumes half of the expected maximum aggregate demand. Normalizing the aggregate observed demand $p_{grid}^{(t)}$ against the maximum possible demand load allows for the RL agent to be immediately scaled to larger communities without modifying the RL agent. Additionally out of the box formulations of RL such as those provided in [27] are typically tuned for normalized reward functions, so the training process is more numerically stable for reward functions that are in the same order of magnitude.

3.3 Results

Experimentally, over 5000 training steps the average reward of the RL agent achieves a 13.7% improvement in the average reward obtained from the first to last episode. Over the testing period (the sixth episode) the RL agent achieves a 2.5% improvement of the average reward over the baseline do-nothing case. (Recall that the reward is bounded on [-1,1].) Where regret is defined



(a) Value Loss

(b) Entropy

Figure 3.2: RL Agent Validation

as the difference between the maximum possible utility of the reward function and the obtained utility of an algorithm, this is a clear improvement of the regret.

In Fig. 3.2a we see that the RL agent's value function approximator begins to converge. In Fig. 3.2b we see that the entropy settles around 0.8, where at values ≥ 1 the action selection is completely random and at 0 the action selection is static for all states. Since the community energy consumption is largely constrained by thermal bounds and the action has only a little sway over the current energy consumption the entropy remains fairly high.

	RL	**Baseline**	**TOU**
Overall Peak Demand (kW)	67.5	69.8	85.6
Average Daily Spread (kW)	16.1	17.4	21.8

Table 3.1: P_{grid} Summary

Table 3.1 shows a 3.4% reduction from the baseline in the observed maximum demand over a 30-day period and, on average, a 7.5% decrease in the average daily load spread (the difference of maximum and minimum daily demand). In [18] a 5% reduction of peak demand across the US was estimated to save $3 billion annually. Eighty percent of these savings came from the reduced use

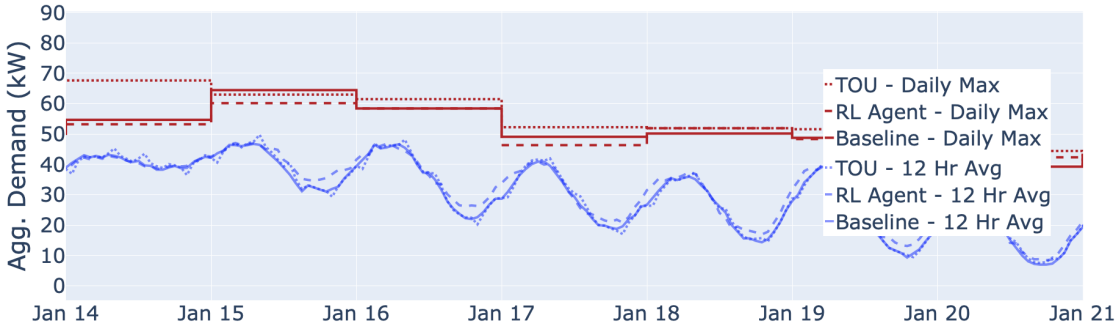


Figure 3.3: 12-Hour moving average and daily max Load

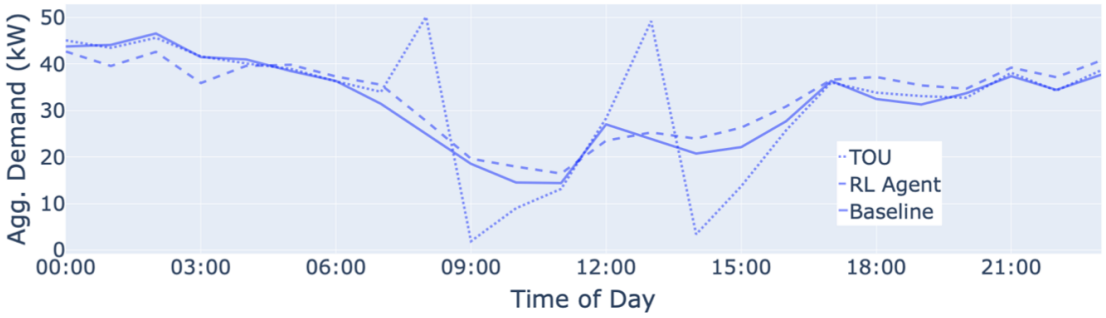


Figure 3.4: Daily Load Profile Averaged over 30 Days

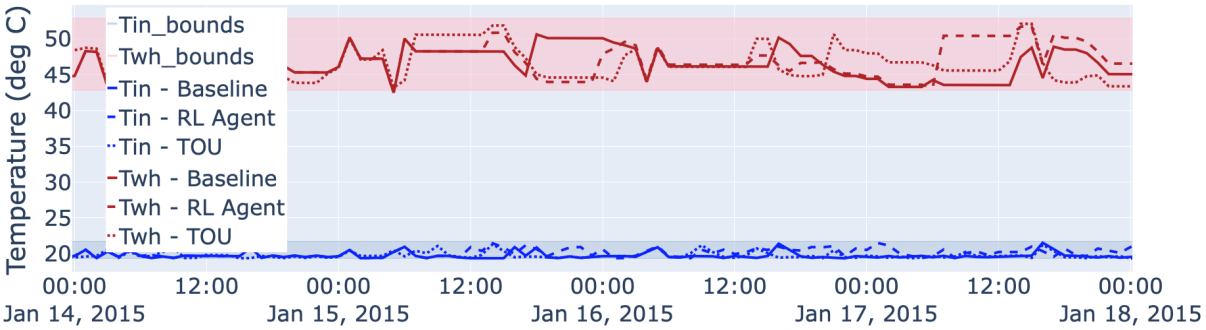


Figure 3.5: T_{in} and T_{wh}

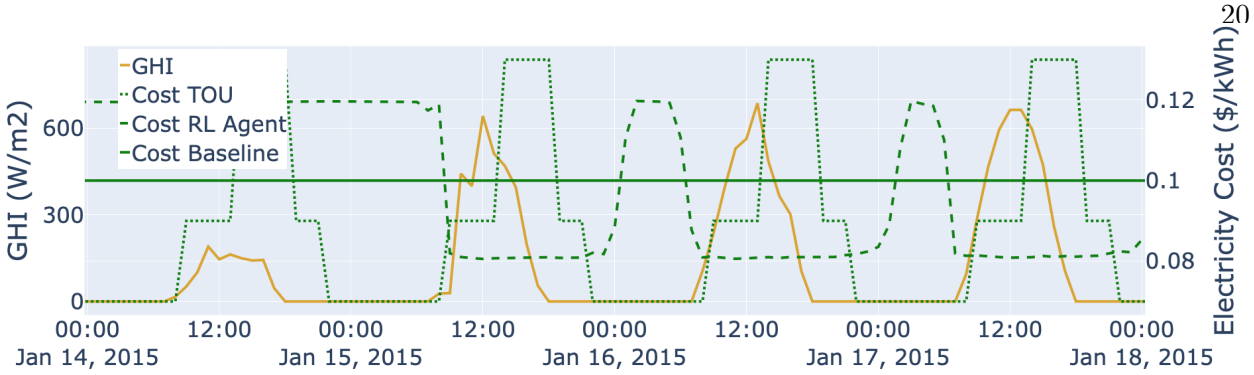


Figure 3.6: Real Time Price and GHI

of peaker plants primarily operated in steep ramping periods. Unlike upgrading transmission lines to accommodate the peak demand in worst-case scenarios, reducing the magnitude of ramping and the number of ramping hours reduces the operational cost each time. Therefore, the RL agent, which performs on average **significantly** better at managing the load spread but is only **slightly** better at reducing the peak demand, can still achieve the majority of the savings associated with load flattening.

Out of 30 simulated days, the maximum daily demand is reduced from the baseline on 16 days and from either the baseline or TOU on 28 days. Fig. 3.3 shows maximum daily demand and 12-hour rolling average for the second week of simulation time, where the peak demand is reduced or maintained for 5 of the 7 days shown. The average daily demand profile in Fig. 3.4 shows the reduction of load spread by decreasing the maximum demand and increasing the minimum demand compared to the constant rate baseline and time-of-use pricing cases.

In Fig. 3.5 we see that the community's total energy consumption is preserved due to the thermal constraints on each HEMS. In preparation for water draws timesteps in the horizon, even the HEMS in the baseline case utilizes the full range of the hot water thermal bounds. From this, we can see that for water heaters, there is a limit to the load shifting regardless of the reward price signal, and this contributes to the high entropy observed in Fig. 3.2b.

In Fig. 3.6 we see that the RL-determined price signal is strongly correlated to the GHI, agnostic to the state of any of the homes. When the RL agent with the performance described

above is initialized on a community with 50% more houses and proportionally more PV, the peak demand is reduced by 1% and the average daily spread is maintained. Comparatively, a new RL agent with no prior training maintains the peak demand but increases the average daily spread by 5%. The use of transfer learning, in which one RL agent is applied to a new environment, creates a stable initialization period without developing a complete model of the community.

Chapter 4

For the Systems Operator: Generator Set-points for Voltage Regulation

Voltage regulation in the power grid is becoming increasingly important with the rise of distributed energy resources (DERs) which provide intermittent and uncontrollable generation. Historically, the systems operator responds to the consumer's demand with local control of the generation plants. In most cases, a turbine governor regulates the speed of a turbine via a PID control loop to match changes in demand. However, especially under new circumstances of increased DERs, this kind of PID control may be inadequate. PID controllers are tuned for generators at the time of installation and may be satisfactory when the system has an overall high level of inertia as with conventional generators — however, inverter-based resources (IBR) are inherently low-inertia and can negatively impact conventional PID tuning. To overcome this, renewable controllers are typically throttled with a "virtual inertia" element [55], which negates some benefits that inverter-based resources could offer, such as fast frequency control. Other control methods include solving the optimal power flow equations, which require complex models of the transmission systems and cannot be solved in real-time. To combat issues with PID governor control (which may be hard to generalize to new DER generation patterns) and problems with OPF (which is historically a slow method of dispatching generators), we introduce a case study for generator dispatch using reinforcement learning (RL). RL models can help generators pursue optimal set-points on a second or sub-second level without solving a computationally expensive optimization problem or requiring an exact model of the current system. Additionally, RL continually adapts to the environment, whereas PID controllers can require re-tuning as the generation profile changes.

In contrast to previous studies which use RL in power systems settings to improve stability in steady-state analyses, we propose using RL in dynamic simulations to improve efficiency. To do so, we leverage a combination of Python for creating deep neural networks and machine learning algorithms with Modelica for dynamic power systems simulations. While Python is generally a robust programming language, it is unsuitable for dynamic simulations of power systems typically modeled with ordinary differential equations. Therefore we introduce Modelica, a physical-based programming language suitable for dynamic simulations of electrical networks and many other energy domains.

Previous works have leveraged the capability of the Modelica language for RL in Python; however, they have mainly focused on the use of functional mockup units (FMUs) [26, 37]. In many cases, FMUs limit the simulation's speed and accuracy and require additional changes to the Modelica model for interfacing with the FMU via an FMU-specific API such as PyFMI [11]. Therefore we interface with the Dymola compiler for Modelica directly through the Dymola API.

The findings of this chapter are a result of the publication [45].

4.1 The DymolaGym Package

DymolaGym extends the OpenAI gym environment and adds the functionality of the Dymola API to ModelicaGym environments [45]. The OpenAI gym ensures that RL agent methods can be benchmarked in an environment to directly compare the RL agents' performance [8]. It does this by standardizing methods such as `step()` and `reset()`, defining the observed **state** values and state-

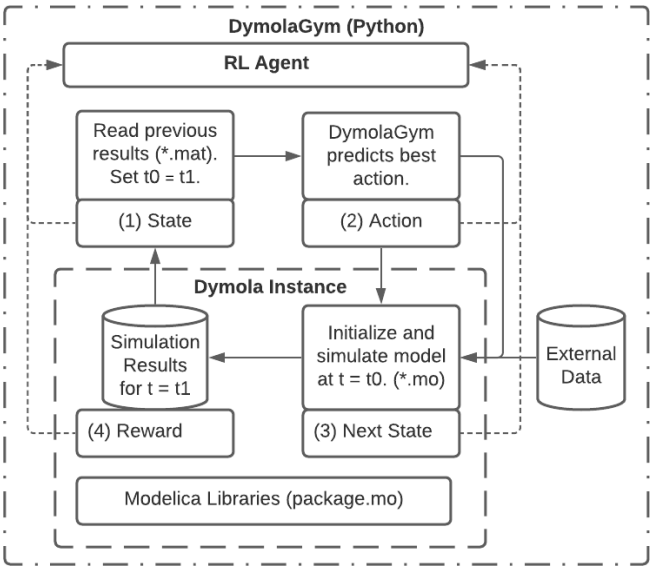


Figure 4.1: Python wrapper of Dymola-powered simulation for use in the OpenAI gym implementation

space and similarly the action-space, and requiring a reward function. Within this framework, DymolaGym requires users to identify **state** values that accurately describe the environment and **action** values that can be used to influence the environment; it uses the user-defined state and action values to build the transition function (`step()`) as required by OpenAI gym. Secondly, users must define the custom reward function that aligns with their intended control scheme. Finally, we offer the user a straightforward method of integrating Python-basedata processingng (e.g., for feeding in static load profiles) which reduces memory usage and computational time compared to doing so in Modelica.

DymolaGym is intended to be a user-friendly package for applying RL control techniques to Modelica simulations. As such DymolaGym environments can be created with minimal changes to the Modelica model: any value declared as a `parameter` in Modelica (e.g., such as the value of a controller's set-point) may be used in DymolaGym as a control action; any value that is assigned to a `variable` in Modelica may be used in DymolaGym as a "state" value (e.g., typical variables in power system models such as algebraic ones (voltage magnitude) or dynamic states (generator angles or internal control states)). The code for Dymola-enabled gym environments ("DymolaGym") is publicly available at `https://github.com/apigott/dymolagym`.

4.1.1 Case Study

In this case study of DymolaGym, we look at the impact of the generator set-point on the network voltage. We use the IEEE 9-bus transmission network, which includes three generators and three load centers. Using the OpenIPSL library for Modelica, we modified the IEEE 9-bus test case and enabled RL-control of two conventional generators (controlled by a single agent) as shown in 4.2. The third generator functions as a slack bus.

Each load is modeled as a ZIP model, which can provide very slight deviations in current and/or power consumption with voltage and are realistic to the existing portfolio of connected devices. These voltage deviations for the load's constant power fraction can provide congestion relief on lines when voltage is high and current is low. Conversely, the constant current fraction of

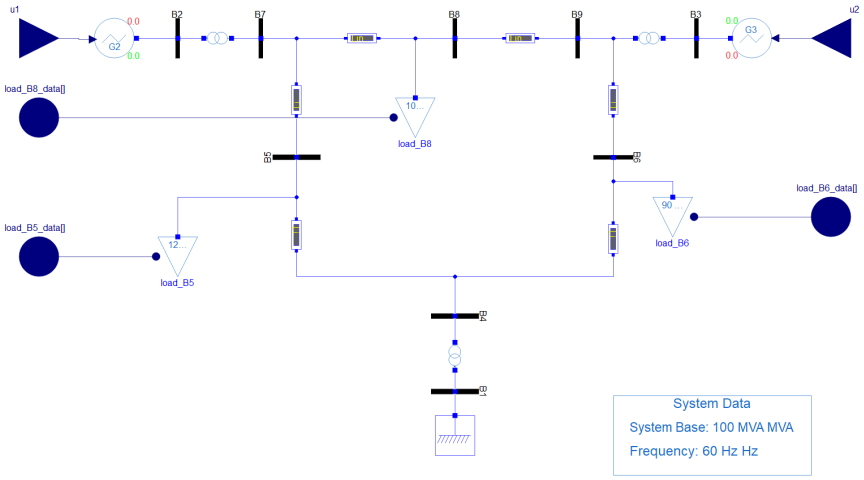


Figure 4.2: Modified OpenIPSL IEEE 9-bus network

the load can, in effect, provide peak shaving for the generation requirements when the voltage is low, a phenomenon known as conservation voltage reduction (CVR).

In this case study, we consider the goal of the grid operator to be voltage regulation, with the RL reward given as follows:

$$r(x^{(t)}) = \sum_{i=1}^{9} (v_i^{(t)} - 1)^2 \tag{4.1}$$

which penalizes voltage deviations from 1 per unit (pu), the nominal voltage. While the most efficient implementation of CVR would suggest promoting voltages under 1 pu we use a common objective that penalizes both under and overvoltages to increase the stability of the network during load fluctuations. Modelica produces sub-second level results but to provide some stability to the RL agent in training, we use the average of the voltage profile over the last minute.

Here we use the measured voltage at all nine buses, the load at all three load centers, and the current real power output of all three generators as observed "state" values. We normalize the "state", action, and reward values to the interval [-1,1] to take advantage of typical training parameters of various RL algorithms.

4.2 Results

In this section, we present two scenarios for voltage control with (1) ±10 MW control and (2) ±50 MW control over the generators' real power set-points, respectively.

In both scenarios, the overall voltage deviations from 1 pu are reduced, as shown in Fig. 4.3. In case 1 ("RL-10") with less control over the generator set-point, the voltage is increased locally at Generator 2, but the voltage deviation is reduced overall from the baseline case. In other words, the RL controller with global oversight ignores the baseline controller goal of local voltage regulation in favor of overall voltage regulation. In case 2 ("RL-50"), the voltage is reduced throughout the network, including at the load buses that already dip under 1 pu voltage (Buses 5 and 6).

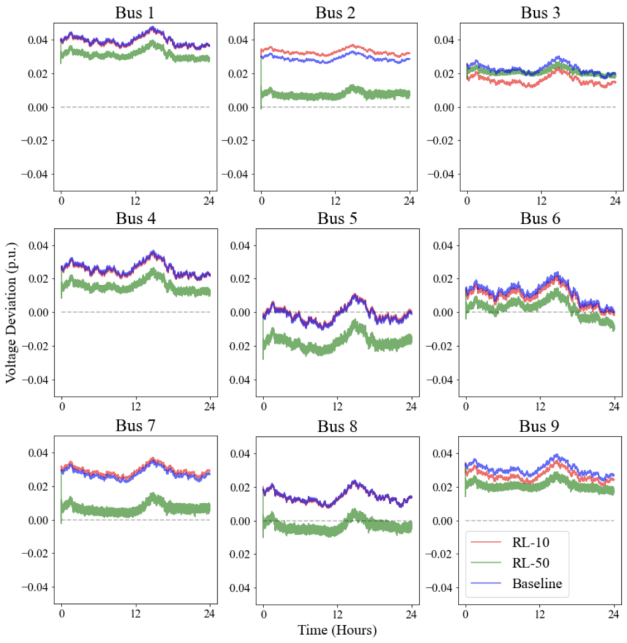


Figure 4.3: Voltage deviation from 1 pu

Fig. 4.4 summarizes the performance of the RL agent with voltage deviation measured by the $\mathcal{L}_2$-Norm (a value of 0 would indicate that all buses achieved perfect regulation at 1 pu, while a larger value would suggest that one or more buses are over/under 1 pu). Additionally, Fig. 4.4 includes the total voltage spread in the system where the voltage spread is **increased** for the "RL-

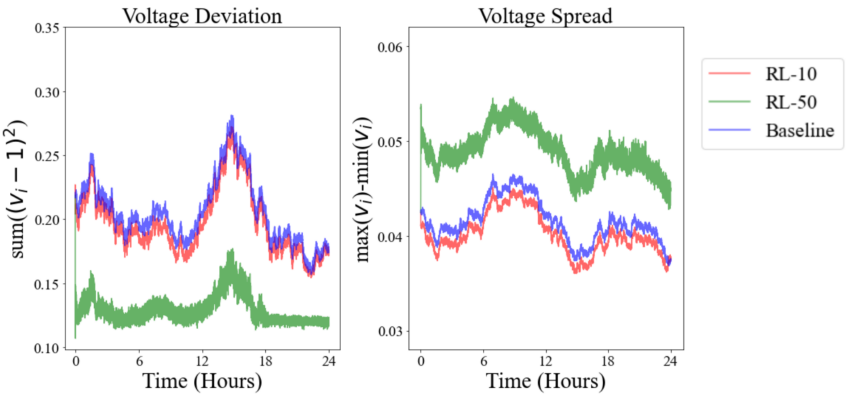


Figure 4.4: Summary of network performance

50" case. In the "RL-10" case, the voltages are, on average, lowered slightly from the baseline case giving it a somewhat lower total deviation; in the "RL-50" case, the voltages are lowered even more for a significantly reduced deviation — which corresponds to the increase of control domain.

The RL-generated actions ramp Generator 3 to the extremes of the control domain, while load following with Generator 2 (Fig. 5.5). The result is that the "RL-50" case operates Generator 3 at a very low fraction of its total capacity. In contrast, the "RL-10" case (due to the inherent limitations of the control domain) operates with both generators splitting the load relatively evenly. In response to the low loading of Generator 3 in the "RL-50" case, the slack bus must become significantly more responsive than the "RL-10" or baseline case.

One possible reason for such a drastic difference in control strategies is that the RL agent with less control ("RL-10") is incapable of reducing the voltages to 1.0 pu and instead selectively reduces the voltage at the highest voltage buses (Bus 1 and 9). Additionally, the second RL case drastically reduces generation from the controllable generators, which might be because the slack bus is tuned to be too responsive ($H = 1$ second, compared to $H = 3.33$ and 2.35 s for the controllable generators, respectively). The difference in the inertia of the controllable generator also likely explains why the RL agents favor G2 for control over G3 — the high inertia makes G2 more stable and, therefore, predictable during the training phase. We might further reduce reliance on the slack bus by increasing the machine's H or increasing the impedance of the connected lines. Additionally, we might see more control over the more responsive generator G3 by introducing

multiple RL agents dedicated to individual generators or allowing the RL agent more "exploration" to overcome the instability of G3 during the training period.

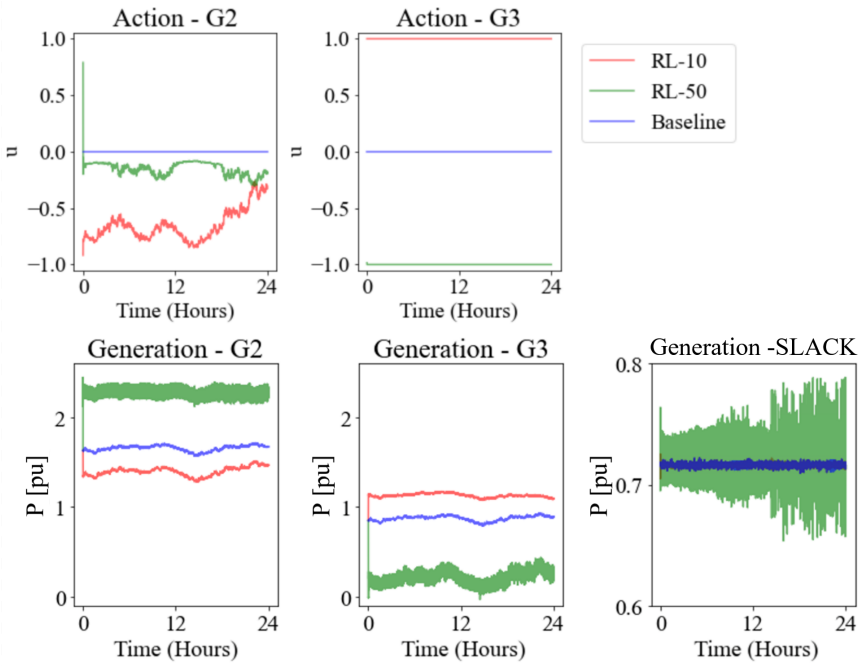


Figure 4.5: Control actions and output of generators

The overall performance of the controller is assessed in Fig. 4.6, which shows that the load roughly matches the demand for both the RL and baseline droop-controlled scenarios (note that the per unit power is plotted with a base unit of 100 MW). A slight difference in load for the

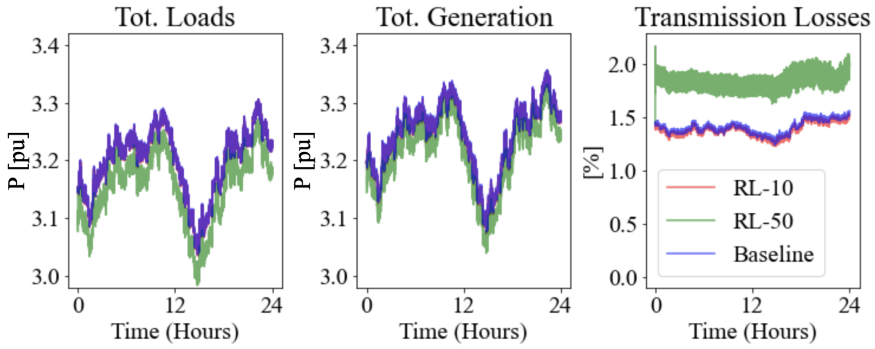


Figure 4.6: Aggregate load and generation of systems with difference given as transmission losses

baseline and RL-controlled versions can be accounted for by the fact that the ZIP loads modeled in OpenIPSL have a constant current component that lowers the power consumption of the load under decreased voltages. Fig. 4.6 compares the transmission losses of the RL agent and the baseline controller.

Although the system losses are higher for the "RL-50" controlled case, the overall generation is reduced by 0.6% throughout the simulation period. Both can be attributed to the overall lower voltage in the network: resistive losses are higher for low-voltage networks due to constant power loads that increase current. In contrast, the constant current part of the loads draws less power at lower voltages. The "RL-10" case, which has a lower voltage spread, reduces the transmission losses but negligibly reduces the total generation.

4.3 Conclusion

In this chapter, we have introduced a reinforcement learning toolkit for Dymola-enabled environments. As a case study, we have provided two trained RL agents that successfully control generator set-points in real time to reduce the voltage deviations throughout the network to various degrees. The results indicate that a learning-based, model-free approach can prove useful for real-time control in dynamical environments. Future work could consider different grid objectives, such as frequency regulation or control of inverter-interfaced generation. In addition, more advanced localized controllers could be compared against the RL agent's performance versus just comparing against the standard droop controllers.

Chapter 5

For the Commercial Building Owner: Multiagent Building Energy Management for Grid-Level Goals

As recent advancements in consumer appliances have increased electric demand (e.g. through electrified HVAC equipment and other consumer appliances), the electric grid has come under increased stress. In distribution networks, the stress is even more apparent as power pushed back from distributed energy resources (DERs) such as electric vehicles or photovoltaics can increase congestion on distribution lines and equipment such as transformers. Furthermore, large shifts in demand can cause voltage irregularities that affect consumer service.

Conventionally demand response is limited to load shedding during peak hours to benefit utilities at the transmission level. However, here we propose the use of grid-aware demand response to provide other grid services such as voltage regulation. While voltage regulation is typically controlled via optimal power flow (OPF), OPF requires precise models of distribution lines which are uncommon for utility companies to maintain. At the distribution scale, three-phase AC OPF is preferred for its accuracy but is non-convex which increases the difficulty of solving the optimization problem.

Using reinforcement learning from the perspective of the consumer we can offer consumers complete privacy. Only the total demand at each timestep is used to simulate power flow in our virtual test bed but in practice, the utility does not need to know any information about demand beyond what they currently use to operate the grid. In line with preserving privacy, the utility has no direct control over consumption so consumers' comfort cannot be impacted.

The findings of this chapter are the result of the publication [47].

5.1 Methods

The framework for energy modeling is based on the CityLearn competition [58] for multi-agent demand management with several changes to support grid-level goals. In CityLearn, the energy models of 10 distinct buildings are analyzed via EnergyPlus. The load profiles in the form of thermal energy and end-use electric loads are then passed to the Python wrapper of CityLearn. In this wrapper, the RL agent can change the allocation of energy used to charge thermal storage devices (hot and chilled water tanks) according to the logic in Alg 1. Most importantly, throughout the simulation the energy supplied to the consumer through a combination of electrical and thermal energy is maintained at each timestep: the demand for the building is broadcast and then met with a combination of energy from the grid or from thermal storage, on top of the building's operational demand the thermal devices can add energy to storage given that they have excess capacity for producing thermal energy and the storage is not at capacity. Therefore consumers who participate in this demand response have no thermal comfort violations.

Changes to the CityLearn environment can be summarized as follows:

- The load profiles were interpolated to sub-hourly intervals as most voltage regulation is performed on a sub-hourly timescale.
- The number of buildings was increased significantly by creating multiple buildings based on a single EnergyPlus energy profile.
- A variable percentage of the buildings were given RL control.
- DC resources (PV and battery) were modified to include P-Q control, as well as curtailment for PV installations (curtailment was not used for this study).
- Building plug loads were given an aggregate, fixed, power factor of 0.95 lagging.

Algorithm 1 Algorithm for determining total power consumption of energy storage devices

Data: Charge/discharge action, u

Result: Electric power consumed

$P_{cons}^{elec} = \eta^{-1} P_{demand}$

$P_{request} = \underline{P} + u(\overline{P} - \underline{P})$

if $u \geq 0$ **then**

  $P_{ch}^{avail} \leftarrow (E_{max} - E^{(t)})(\Delta t)^{-1}$

  $P_{ch}^{avail} \leftarrow \min\{P_{ch}^{avail}, \overline{P} - P_{request}\}$

  $P_{stor} = \min\{P_{request}, P_{ch}^{avail}\}$

else

  $P_{disch}^{avail} \leftarrow E^{(t)}(\Delta t)^{-1}$

  $P_{disch}^{avail} \leftarrow \max\{P_{disch}^{avail}, \overline{P}\}$

  $P_{stor} = \max\{P_{request}, P_{disch}^{avail}\}$

end

$E^{(t+1)} = E^{(t)} - E^{loss} + P_{stor}\Delta t$

$P_{cons}^{elec} += \eta^{-1} P_{stor}$

return P_{cons}^{elec}

As in the previous works, the open-source code for these models is available at `https://github.com/apigott/gridlearn`.

5.2 Results

The set of RL agents was experimentally tuned on the environment until they improved on the baseline as measured by voltage deviations from 1 pu. A lower learning rate was found to be particularly helpful in avoiding oscillatory behavior.

Once the RL agent is trained, it can be "transferred" or applied to different climate data or can be used at different time intervals. The results shown in this section are for models trained on Climate Zone 2A (hot, humid) and tested on Climate Zone 3A (warm, humid). Regardless

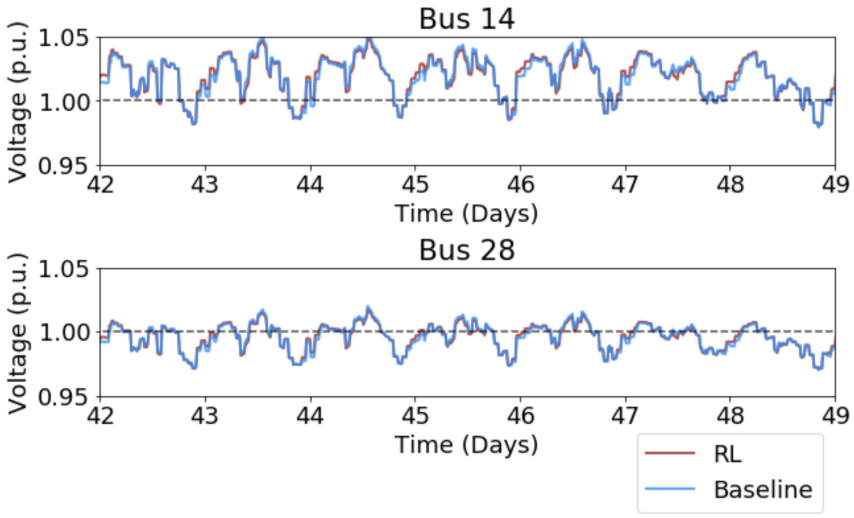


Figure 5.1: Voltages at Buses 14, 28

of the change in climate data, the RL agents are successful in reducing the L2 norm of voltage deviations by 0.5% on average. Although there is not a significant change by the L2-norm metric it is important to note that in most networks the voltages largely fall into an acceptable range. We can see in Figure 5.1 that voltages over 1 pu are reduced (with the largest deviations seeing the greatest reduction) and some voltages under 1 pu are raised which shows that the RL specifically targets the most extreme voltages. In the winter months, where the voltage is generally above 1 pu, the RL agents become biased towards voltage reduction.

In Figure 5.1 the RL agents at times perform worse at Bus 14 or 28. For example, at the beginning of day 43, the voltage is raised for both Bus 14 and Bus 28. In Figure 5.3 we can see that the maximum voltage is temporarily increased even further than the baseline due to this change in voltage. However, in Figure 5.4 we also see that the average voltage hovers around 1 pu and the highest peak values of the maximum voltage are still reduced. This is likely due to the strong coupling between the buses where raising one voltage below 1 pu might also raise the voltage elsewhere in the network to be above 1 pu. A moving average of the L2-Norm of the voltage deviation at all buses is plotted in Figure 5.2. This demonstrates that the decentralized multiagent approach still works to reduce voltage deviations in general.

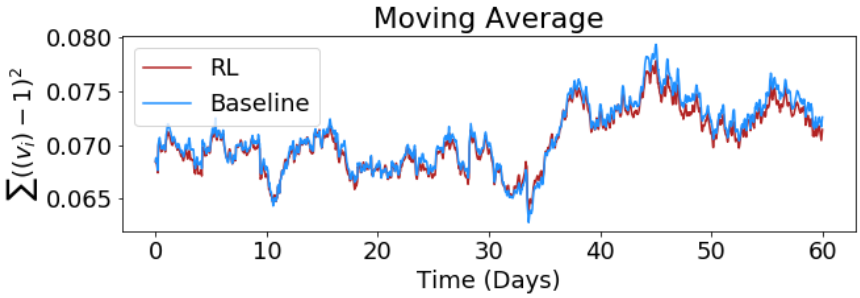


Figure 5.2: Overall grid voltage deviation

Considering the overall voltage deviations (Fig 5.2), we observe that the RL agents generally reduce the overall voltage deviations, especially during periods of more extreme deviations (i.e. Days 40-60). In Figure 5.4 we plot the number of intervals that each voltage is observed. In the baseline case, the number of 15-minute intervals across all 33 buses in which the voltage is at the upper bounds (1.03-1.05 pu) is more than twice the number of intervals that any bus sees a voltage at the lower bound (0.95-0.97 pu). The skew towards rising voltages aligns with earlier studies such as [**?**] and suggests that the most important task for the RL agents is in preventing overvoltages. The number of observed voltages over 1.03 and 1.04 pu as well as voltages under 0.97 and 0.96 pu are summarized in Table 5.1. Note that the RL agent significantly reduces overvoltages that are closer to the operating bounds.

	Baseline	RL	% reduction
$v_i^t > 1.04$	812	532	34.4
$v_i^t > 1.03$	6361	6156	3.2
$v_i^t < 0.97$	2867	2804	2.2
$v_i^t < 0.96$	1035	1018	1.6

Table 5.1: Number of 15-min intervals of over/under voltages

By inspecting the actions selected by each agent in Figure 5.5, we see that all agents converge on similar control strategies. This would likely lead to the RL agents learning to reduce their

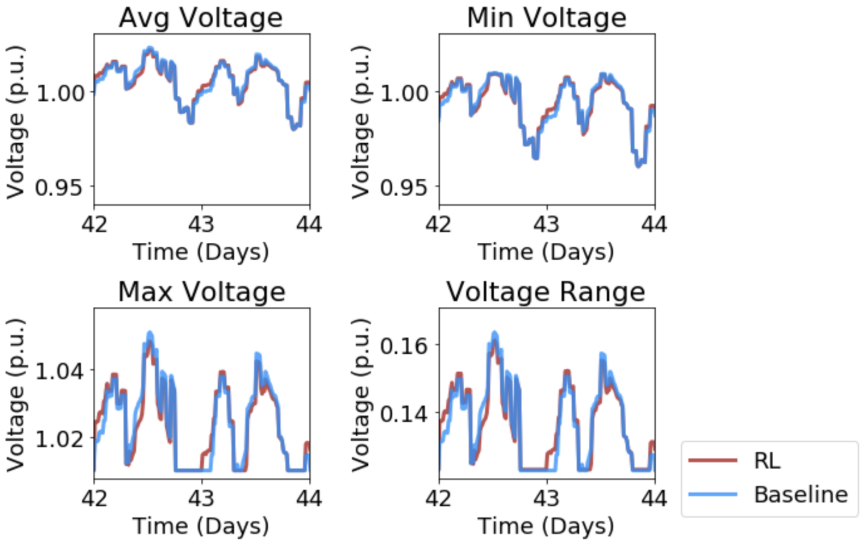


Figure 5.3: Summary of all voltages observed in the network over time

responsiveness as they try to avoid collectively acting too responsively in over or undervoltage scenarios. In general, the RL agents learn to shift their peak demand slightly later than the baseline RBC controllers and learn to be more conservative in their charge/discharge signals. It should be noted that the battery subsystem stopped charging and discharging during the testing period, even though it was frequently charged/discharged while training the agents. Since the battery is a DC resource behind the smart inverter, it likely strengthens the impact of the RL phase lag control. However, the battery might prove too responsive during training to be a legitimately useful resource in the testing phase. The increased reliance on thermal storage might also benefit building owners since charge/discharge cycles add little wear to the HVAC and DHW systems and leaves the battery energy storage free for other resilience-related objectives.

The inverter phase shift observed in Figure 5.5 reveals that the RL agents learn to curtail real power injections in favor of reactive power injections that supplement the grid capacitor banks in providing voltage regulation. As a result, grid operators might allow for increased real power uptake by increasing the size or number of capacitor banks in the grid.

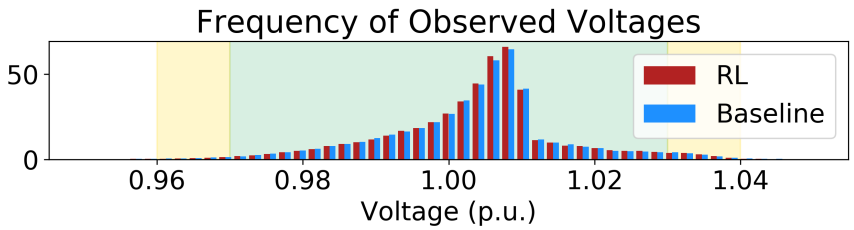


Figure 5.4: Histogram of observed voltages

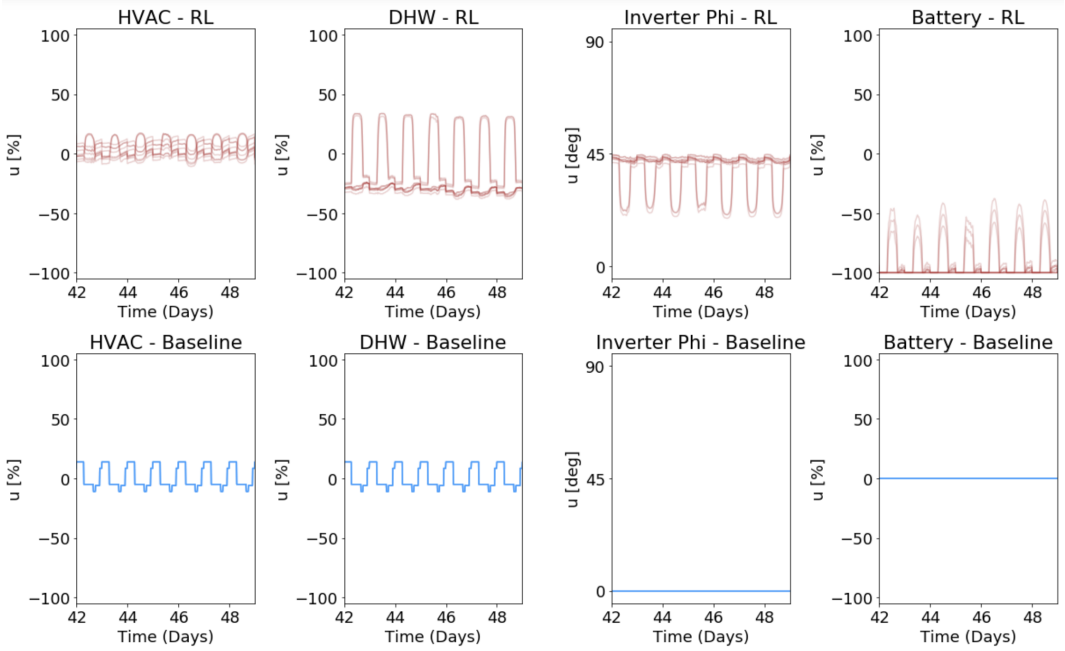


Figure 5.5: Subsystem action selection across buildings

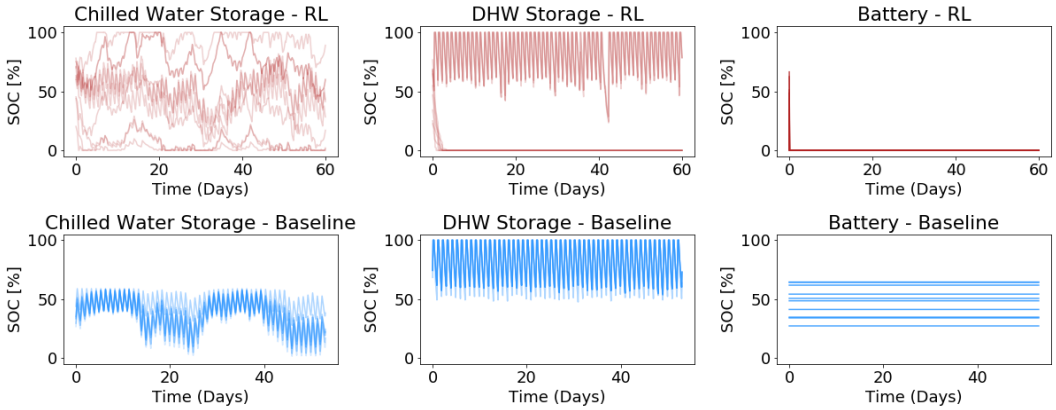


Figure 5.6: Subsystem state of charge across buildings

The resulting state of charge for each of these scenarios is shown in Figure 5.6. As shown in Alg. 1, even though the battery is consistently in a state of "discharging" the actual power discharged from the device is 0 kW, as the battery is quickly drained at the start of the simulation period. To activate the battery as a resource, we speculate (1) that more specific objectives, such as net demand reductions, should be implemented in the reward function and weighted with voltage regulation and/or (2) that the battery resource is too sensitive. The thermal resources must accommodate a regular demand profile and provide demand response. With the total demand limited by each device, the device is not guaranteed to deploy each action fully. We speculate that curtailing the action this way helps the RL learn more stable behavior from the thermal devices.

Chapter 6

For Microgrid Engineers: Distributed Resource Placement for Improved Efficiency

Capacitor banks are often installed by electrical network operators to provide reactive power compensation, power factor correction, and/or help minimize losses in the network. Capacitor banks are typically passive reactive power devices and historically placed near the load. Some capacitor banks have minimal controls, such as switching controlled by local power factor measurements [48]. Many prior publications have focused on improving control logic performance of existing capacitor banks [48, 14, 39]. In [61] reinforcement learning (a form of black box machine learning) is used to determine control of various resources including capacitor switching. However, in this paper we focus on the impacts of capacitor banks that can be improved solely on the placement within a network rather than via switching.

Conventional optimal power flow (OPF) can be used to determine capacitor bank switching, and with adaptations of mixed integer variables can even be used to determine placement. To avoid computational challenges, however, OPF formulations for planning purposes are generally done with assumptions of balanced loading or approximations of the AC power flow equations. In order to take into account the effects of unbalanced loading which could include equipment overloading/degradation, voltage imbalance, and equipment faults, we ideally want to consider the full, nonconvex three-phase AC power flow models. PowerModelsDistribution.jl is one such package that formulates OPFs of distribution networks using three-phase unbalanced AC power flow models and leverages various commercial solvers in Julia [**?**]. Black box machine learning algorithms are

capable of optimizing environments that are difficult to represent with convex constraints but easy to simulate given the parameters are predetermined. We are therefore motivated to create a machine learning (ML) formulation that is less computationally expensive than conventional OPF formulations.

As discussed in Chapter 1, reinforcement learning is most typically used as a control algorithm. RL is typically designed to adapt to the current perceived state and respond with the most likely most advantageous actions, pulled from a probability distribution. However, in a static design problem one of the key elements of reinforcement learning agents which is to adapt to the current state reading becomes irrelevant. In this proposed framework we elect to use the k-armed bandit [54] as a model for determining selection as opposed to reinforcement learning because k-armed bandits are more typically used in stateless environments (those in which the action of the agent is irrelevant to any state not in the immediate future). In the example of our capacitor bank placement problem, giving the machine learning agent information about or related to the time of day might encourage a solution that is temporally determined (i.e., moving capacitors to one part of the grid that it predicts to be a high load during the day and another during the evening). Since the given problem of capacitor bank placement should produce a solution that does not vary over time, and the transition between voltages at one timestep to the next is not linked, the state data is irrelevant to the ML agent.

The basis of the optimization problem is taken from the PowerModelsDistribution.jl formulation, which writes the OPF problem in the JuMP syntax. We then modify the AC polar coordinate constraints to model the possibility of a capacitor bank at each bus. Lastly, we add our mixed-integer constraints to indicate the presence of a capacitor bank at each node subject to a limited number of capacitor banks available. In this case, we constrain the number of capacitor banks to the number provided in the original IEEE test case. These constraints are formulated as follows:

$$\min_{\alpha} \quad \sum_{t=0}^{\tau}\sum_{n=1}^{N}(v_n^{(t)}-1)^2 \tag{6.1a}$$

$$\text{s.t.} \quad p_n = \sum_{mk}|v_n||v_m|(g_{nmk}\cos\theta_{nm} + b_{nmk}\sin\theta_{nm}) \tag{6.1b}$$

$$q_n = \sum_{mk}|v_n||v_m|(g_{nmk}\sin\theta_{nm} - b_{nmk}\cos\theta_{nm}) \tag{6.1c}$$

$$p_n = p_n^{gen,(t)} - p_n^{load,(t)} + \alpha_n^{(t)} p_b^{DER,(t)} \ \forall t \in [0,\tau] \tag{6.1d}$$

$$q_n = q_n^{gen,(t)} - q_n^{load,(t)} + \alpha_n^{(t)} q_b^{DER,(t)} \ \forall t \in [0,\tau] \tag{6.1e}$$

$$\sum_{n=0}^{N}\bar{\alpha}_n = M \tag{6.1f}$$

$$\alpha_n^{(t)} \leq \bar{\alpha}_n \ \forall n \in [1,N], \ \forall t \in [0,\tau] \tag{6.1g}$$

$$V^{min} \leq |v_n^{(t)}| \leq V^{max} \tag{6.1h}$$

$$\theta_{nm}^{min} \leq \theta_{nm}^{(t)} \leq \theta_{nm}^{max} \tag{6.1i}$$

where voltage at bus n and time t is given by $v_n^{(t)}$ and bounded by V^{min} and V^{max}. Each line k connecting buses m and n is represented with line conductance (g_{nmk}) and line susceptance (b_{nmk}); the voltage angle difference between buses n and m is given by θ_{nm}. Real (p) and reactive (q) power injections are given with regards to each kind of resource (e.g. generators, ^{gen}, or capacitor banks, ^{DER}). Let constraints (6.1b) and (6.1c) be the real and reactive power losses on each line, respectively. Constraints (6.1d) and (6.1e) balance the power coming in and out of each node, the $\alpha_n^{(t)}$ term represents real and reactive power source or sink from the distributed energy resource. Therefore $\alpha_n^{(t)} = 1$ indicates that there is a resource placed at node n **and** it is enabled; $\alpha_n^{(t)}$ indicates that there is no resource placed at node n **or** that it is disabled at time t. We assume that the source/sink value is constant in order to eliminate a term that is the product of two variables. The constraint in (6.1f) uses the binary variable $\bar{\alpha}_n$ to limit the total number of resources equal to the allowable number of resources to be placed, M; $\bar{\alpha}_n = 1$ indicates that at any time a resource is enabled at any time t in the simulation and therefore a resource is permanently placed at that node. We use (6.1g) to represent cases where the OPF is allowed to disable resources at certain

timesteps even when physically connected. If $\bar{\alpha}_n = 0$ (i.e. $\max(\alpha_n^{(t)}) = 0$) then a resource is never enabled at node n and does not need to be physically connected at that node. Physical bounds on the voltage are imposed in (6.1h) and (6.1i).

6.0.1 Methods

Extrapolating the k-armed bandit problem to a problem in which multiple actions can be selected we use the following algorithm to transform the continuous action space to a discrete one. As in Algorithm 2 we use a vector of actions, $a_i \in [0, 1]$ to vote on battery placement at each node. Acknowledging that this kind of projection from continuous to discrete is a relaxation in conventional optimization, in RL environments it is just another layer of black box mechanics for the agent to learn.

Algorithm 2 Algorithm for determining placement of a set of distributed resources

Data: list of actions, $a \in \mathbb{R}^n$; number of possible locations, n; number of resources to place, m

Result: list of selected node indices, $l \in \mathbb{Z}^b$

Initialize a new set of selected locations, $\mathcal{L}$

for i **in 1:**m **do**

  $l_i = \arg\max(a)$

  Add l_i to the set $\mathcal{L}$

end

return $\mathcal{L}$

6.0.2 Results

As mentioned previously, we consider the 13-bus network and the 123-bus network. The optimality of the resulting capacitor placement in the smaller network was compared between the optimized solution obtained using a brute force method and the proposed ML-based method. For the 123-bus network, the mixed-integer, non-convex optimization problem is too challenging to solve directly, and thus we compared the proposed ML-based method with the default capacitor

placement given in the network.

6.0.2.1 IEEE 13 Bus Network

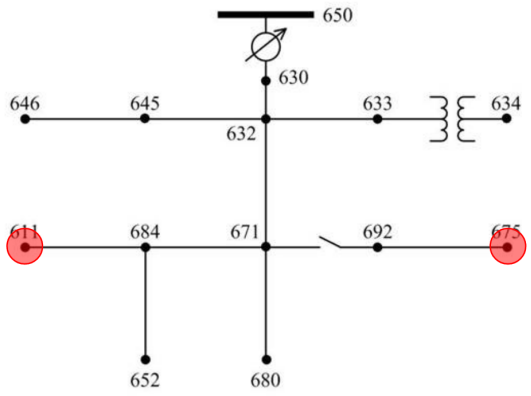


Figure 6.1: One line diagram of IEEE 13-bus network with original placement of 2 capacitor banks

The IEEE 13-bus network (Fig 6.1) is a distribution grid test bed designed based on a 4.16kV radial distribution grid in North America. The network is comprised of 13 buses, of which nine are load buses, two have capacitor banks, and one is the distribution substation that represents the connection to the external transmission grid. The original subcommittee report describes the network [30] as "short and relatively high loaded for a 4.16kV" distribution network. Because the distribution grid is highly loaded, the voltage drop is significant and creates an opportunity to observe the effects of various capacitor bank placements in the network.

The ML solution, the optimal benchmark solution, and the IEEE 13-bus design solution all use two capacitor banks in order to directly compare the results.

ML Solution The ML agent tasked to determine the appropriate placement for the two capacitor banks is trained on a network with approximated load data. The load data is taken from the metered data of buildings in a university campus distribution network. The data is further scaled to experimentally provide minor voltage irregularities outside the ideal range. The metered data is only provided in real power, so we assume the power factor. During the training period, the total power is scaled to create minor voltage irregularities and then perturbed $\pm 12.5\%$; the power factor at each bus is perturbed between 0.91 and 0.94 lagging. The data is randomized during the training period at each bus and timestep. Randomization at each timestep ensures that load curves are not exactly replicated in relation to other buses in the network and creates a more generalized

representation of the impacts of various placements.

The agent is trained on 200,000 timesteps of data, taken at a maximum fidelity of 3-minute intervals. Throughout the training interval, the ML agent converges on a selection of bus placements, and the reward generally increases with more training. Fig 6.2 shows that several nodes are selected with an increasing frequency (e.g., bus 8, while other buses are eliminated as top picks. We show the progression of node selection at 20k training step checkpoints, terminating in the "Selected" node case, which indicates the final selection of the ML agent. This is implemented by eliminating the exploration aspect of the ML agent's prediction function (i.e., the agent only uses information it has already observed instead of trying new combinations to gain more knowledge).

We compare the "Selected" node case with the "Baseline" or the original IEEE 13-bus network design. In contrast with the other checkpoint tests, both cases have a fixed resource placement at every timestep. In that way, these two cases are the only truly feasible solutions because the checkpoint voltage profiles represent a scenario where the resources may be mobilized throughout time. Fig 6.2 shows the underlying "top" picks for resource placement at each checkpoint and the actual distribution of exploration in the darker shaded region.

We effectively terminate the training period of the ML agent at 200k training steps, where

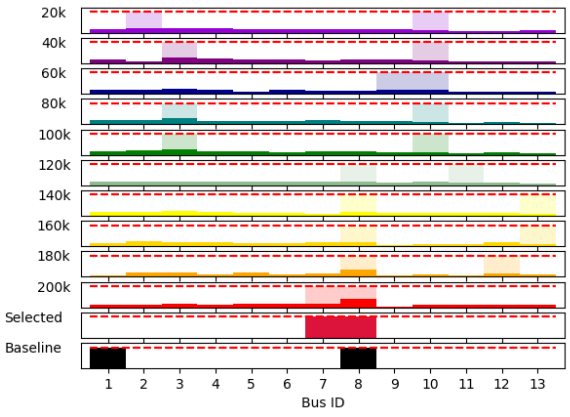


Figure 6.2: Histogram of selected nodes for placement throughout increasing training period lengths (20k to 200k steps)

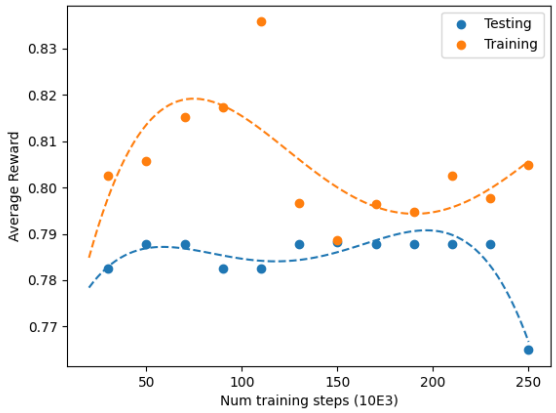


Figure 6.3: Reward improvement over training duration on training and test data sets

we can see in the extended training period shown Fig 6.3 that the ML agent tested on the training data set and a testing data set converge in performance. The sharp divergence in performance after this cutoff indicates that the ML agent overfits the training dataset at the cost of poor performance on the testing data set. The results of the agent trained on 200k data points compared with the baseline IEEE design are shown in Fig 6.4.

Optimization Solution

During experimentation, we found the above optimization is too resource-intensive to solve using the computing resources available to us. We thus utilize a brute force technique to compute the objective value for all combinations of capacitor bank placements since, in the 13-bus network, the search space is still tractable. The parameters of the brute force solver sequentially set the variable $\bar{\alpha}_n = 1$ (i.e., a resource is placed at bus n) at each possible combination of buses. (The brute force solution requires a search of 78 combinations in the IEEE 13-bus without excluding resource placement at the distribution substation or any other buses in the network.)

The solver was parameterized with a `mip_gap` of 0.001 and set to terminate at a maximum of 1 local optimal solution. Several trials of solving the optimization formulation with four time linked power flows failed to find a feasible solution after 112 minutes of computation on the machine specifications in Table 6.1, the same computer used for all ML training simulations.

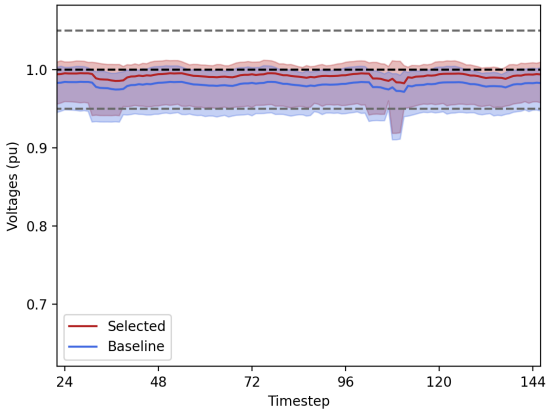


Figure 6.4: Voltage deviation performance of the baseline vs selected capacitor placement

A brute force search of the action space was performed as an alternative method of determining the true optimal solution and benchmarking the ML agent. The distribution of the objective values (minimization) for all possible solutions is given by Fig 6.5. The value produced by the ML solution falls just outside of the 25th percentile and is 6.2% greater than the global minimum (a 65.5% improvement from the maximum or worst-case placement).

The original IEEE configuration was run as a time-linked OPF on PowerModelsDistribution.jl (the on/off status of the two specified capacitor banks being the only optimization variables).

6.0.2.2 IEEE 123 Bus Network

ML Solution

While the IEEE 13-bus is relatively small, which makes it an ideal test feeder for understanding the feasibility of this method, the proposed machine learning solution easily scales to the larger 123-bus network. The 123-bus network is also a low voltage 4.16 kV distribution grid. The original subcommittee report notes that the relatively low voltage for such an extensive distribution grid creates “voltage drop problems that must be solved with the application of voltage regulators and shunt capacitors” [30].

RAM	64 GB
Processor	Intel Xeon W-2235 × 12
Graphics	AMD Radeon Pro W5500
Operating System	Fedora Linux 36
Julia	1.7.3
JuMP	1.13.0
Ipopt	1.1.0
PowerModelsDistribution.jl	0.14.4
Python	3.10.4
Stable-Baselines3	1.6.0

Table 6.1: PC hardware and software specifications

In running the ML scenario, we use the architecture described in the previous section to select buses to place n capacitors. The size and number of capacitors are the same as in the original IEEE 13-bus network.

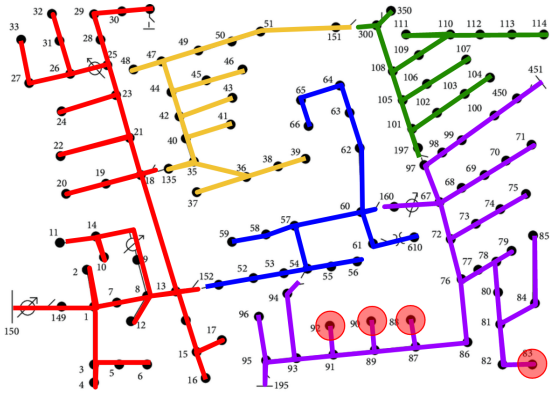


Figure 6.6: One line diagram of IEEE 123-bus network with original placement of 4 capacitor banks

ML solutions are noted for being biased towards the initialization point (occasionally referred to as a "primacy bias" [43]), which is potentially constraining in a problem with several local optima. We present three sets of solutions with different initialization points to examine the outcomes created by this effect. Furthermore, these additional initialization arrangements impact the selection of buses that we include in the allowable solution set. These arrangements are denoted as follows:

(1) "Sorted" in reference to the alphanumeric sorting of the list provided by Julia. Note that sorting bus names as strings (in cases where the bus label is an alphanumeric value, ex. "150r" sorts values by the first character rather than the numeric value of the string.

(2) "Randomized" in reference to generating a random initialization list of buses via Julia's

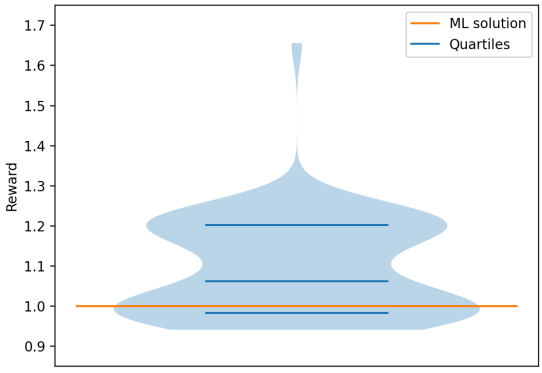


Figure 6.5: Distribution of brute force solutions

`Random.shuffle` function.

(3) "Districted" in reference to randomizing the initialization points across various districts of the distribution network and concatenating the lists of buses. These districts are defined where a subsection of the grid could be isolated via switching points and are shown in Fig 6.6.

In each of these cases, when reducing the available options for resource placement, we select every n-th element of the list, which ensures that in scenarios with a low fraction of buses available, these buses are evenly distributed throughout the network (particularly in the "Districted" cases).

Fig 6.7 and Fig 6.8 shows the progression of nodes selected by the ML agent throughout training; in Fig 6.7, the buses are presented in the same order that the ML agent perceived them, while in Fig 6.8 the buses are presented in the alphanumeric order that they are labeled in the original OpenDSS file. In Fig 6.7, the ML agent is biased towards the buses presented at the beginning of the ML action space due to the processing done on the ML action to convert it to a discrete action. The ML agent is initialized with each element in the pre-processed action vector identically ($a_n = a_0 \; \forall \; n$); when the action is projected onto discrete space with the NumPy `argmax` function, the projection, as a result, becomes biased towards the lower indexed values.

To observe and offset the impacts of the bias, we selected two other arrangements of the

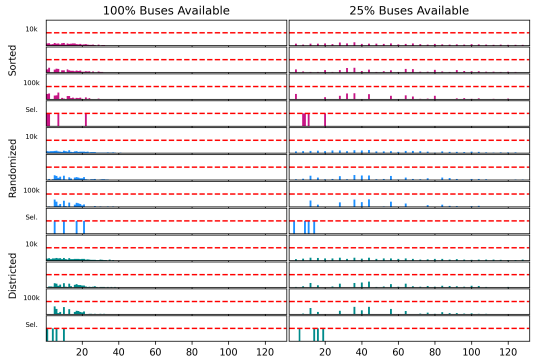


Figure 6.7: Capacitor bank selection ordered by action index as perceived by the ML agent

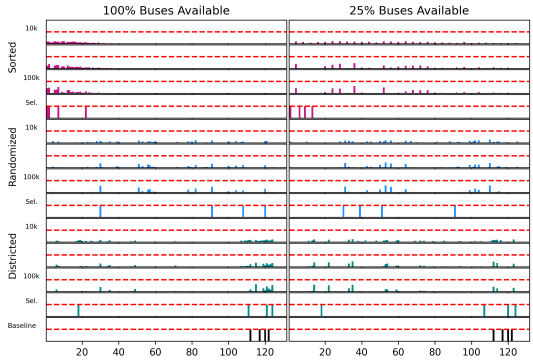


Figure 6.8: Capacitor bank selection ordered by index of bus created in IEEE 123

bus indexes. This perturbation is necessary in the larger 123-bus case study where the ML agent initializes at a point heavily skewed towards the lower indexed buses (Fig 6.7) but not the 13-bus network where the ML agent is relatively uniform in exploring all 13 buses at the beginning stages (Fig 6.2).

The solve time of the IEEE 13-bus timelinked OPF scenario was prohibitively computationally expensive even in the lower action space of 13 buses vs 123. Therefore we did not attempt to solve the 123-bus solution with conventional OPF strategies to create a baseline evaluation of performance. Instead, we direct the reader to the benchmark of the originally proposed capacitor bank placements in the IEEE 123-bus test feeder ("Baseline") and a scenario with no capacitor

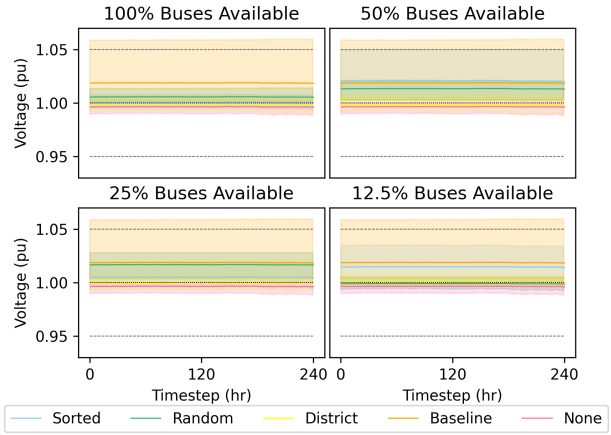


Figure 6.9: Voltage profile created by ML selected capacitor bank placement in the IEEE 123-bus network

banks ("None").

Each of the three ML-trained agents shows a marked improvement over the baseline behaviors of the original IEEE design and the trivial solution of **no** capacitor banks. The summary of voltages observed throughout the network is given in Fig 6.9 with upper and lower bounds in the shaded region and the average voltage highlighted. In the "Baseline" case with capacitors placed in the original locations per the IEEE design the behavior shows a slight overvoltage (averaged around 1.02 p.u.), and each trained ML agent reduces the voltage while maintaining it closer than the baseline to an ideal value of 1.0 p.u.

Fig 6.10 shows the normalized reward values under 12 ML scenarios (four different sizes of node subsets and three different methods for selecting those subsets), the baseline scenario, and the trivial "None" scenario. Due to the quadratic reward function over- and undervoltages are penalized at the same rate. In all scenarios, even those that only consider a fraction of the buses in the allowable action space.

If the ML agent finds the global optimal point we could expect the ML agent's performance to monotonically increase with the size of the available action space. However, the ML agent is more likely to converge on a local optimal when the problem is highly non-convex.

From the results in Fig 6.10 the ML largely outperforms both baselines but significantly

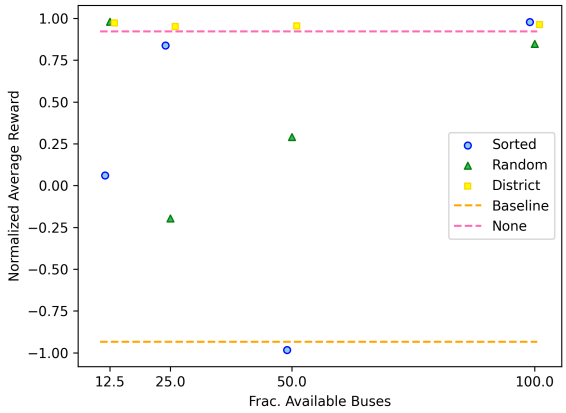


Figure 6.10: Average reward values achieved by various selection methods and ML agents

outperforms the baseline IEEE design with two capacitor banks, and moderately outperforms the baseline with no capacitor banks. We hypothesize that the ML could more significantly outperform the baseline design strategies if the network had a higher loading and lower average voltage. Conversely, the IEEE 123 network case study shows that the ML agent is successful with reducing overvoltages of the original design (in addition to correcting the undervoltages of the IEEE 13-bus case shown earlier).

Recall that the voltage at each bus in the network is taken into consideration even when all buses are not all eligible for capacitor placement (that is, the reward function is consistently a function of network-wide voltages). We also therefore observe that the ML agent is successful optimizing performance even when circumstances (e.g., construction or space limitations) only allow for a limited number of practical placement options.

Chapter 7

For the Homeowner: Home Energy Management in a Competitive Environment

RL and multi-agent RL (MARL) have become increasingly promising alternatives to conventional control strategies in the built environment. However, as we move towards real-world deployment of the various control strategies that have been developed, the underlying assumption of MARL studies: that all users would use a single proposed algorithm or training technique posed by a single manufacturer, should be examined further. For example, in [34], smart thermostats are shown to coordinate pre-cooling and pre-heating behaviors and exacerbate peak loads in practice. In this chapter we introduce a student based competition for home energy management entitled Gaming for Novel Optimization of Managing Energy Systems ("GNOMES4Homes" or "GNOMES"). The intent of GNOMES is twofold: first and foremost we intend to use GNOMES as an educational tool for students learning about building energy systems and machine learning. Secondly, GNOMES poses an opportunity for us as observers to establish several diverse strategies to achieve the same objective.

Competition-based learning (CBL) has been shown to improve learning outcomes and engage students in novel ways [10, 59]. GNOMES aimed to achieve three main learning objectives: 1) To get students from diverse backgrounds excited about using computational skills to address energy problems; 2) To make machine learning concepts, such as reinforcement learning, more accessible to students from all backgrounds; and 3) To incentivize students to learn about residential energy use and their own impact on the electric grid infrastructure. The competition was sponsored by the National Science Foundation (NSF) with up to $10,000 of cash prizes.

7.1 Competition Architecture

For the purposes of teaching students without prior experience in Python or home energy systems and to level the playing field, we developed a ready-to-use toolkit for prototyping home energy management controls. Each player is tasked to optimize the performance of their own single-family home. All efforts have been made to focus this competition on the theory of home energy management and make the programming portion as accessible as possible, which is described below.

7.1.1 Environment

The GNOMES competition is built on the DRAGG platform. DRAGG, as described in Chapter 3, provides a platform for distributed computing-enabled MPC home energy models but has been extended for this competition with the release of DRAGG-v2.0.0 and greater. While improving on DRAGG for this competition, electric vehicles (EVs) have been added to all home models. In addition, in line with adding EVs with temporal constraints, home occupancy schedules have been enabled, allowing a default setback temperature when the home is unoccupied and requiring the EV to be charged before departure.

The remaining changes are competition-specific and are as follows:

- User overrides have been added to the HVAC, water heater, and EV charging controller. The MPC time horizon is set to zero to prioritize the user-designed algorithm at the current step rather than the MPC.

- The home environment is given an alternative process when original solve fails due to the thermal constraints. Instead of brute-forcing a heuristic solution the optimization objective is changed to minimize the difference of the comfort constraint violations.

- Each building asynchronously posts updates to its status. Therefore, MARL agents in a single environment can update simultaneously from different processes without knowledge of the other buildings at that time step.

Manual overrides provided by the player's controller are designed to not interfere with normal operations of the home. Temperature setbacks are enforced outside the preferred thermal deadband to preserve occupant safety and prevent conditions dangerous to mechanical systems (e.g., freezing of water pipes). EVs are enforced to charge before planned departures that align with regular daily travel. We utilize the MPC structure of DRAGG to implement the player's control actions by first attempting to minimize the difference between the desired and implemented control. e.g., If the desired control is to pre-cool the home and the home is near the lower threshold of the home, and thus no more cooling is allowed, the HVAC will remain off; if the desired control is to turn off the HVAC but the home is near the threshold of the deadband the HVAC will turn on at the minimum rating required to meet the thermal threshold requirement. At some timesteps, even the maximum rating is not enough power to maintain the desired or required temperature. This typically happens when the expected value of the environment parameters (e.g. outdoor air temperature) fails to accurately predict the actual value. In this case, the optimization objective is changed to minimize the difference between the resulting temperature and the thermal boundaries.

7.1.2 Scoring

Each team had access to the same home model with the same set of training weather data. At biweekly intervals, we simulated each team's "checkpoint" submission with testing data on the same neighborhood of houses and updated the competition rankings. The training and test data spans a meteorological year to assess performance in all seasons. The scoring metric for players is given in two parts:

- Contribution to peak. The contribution to the peak demand of the community is a metric to incentivize demand that does not align with neighboring consumers. High aggregate peak demand creates the need for peaker plants with conventional fuel sources (e.g., gas) and creates excess wear and tear on distribution equipment operating near its capacity limit.

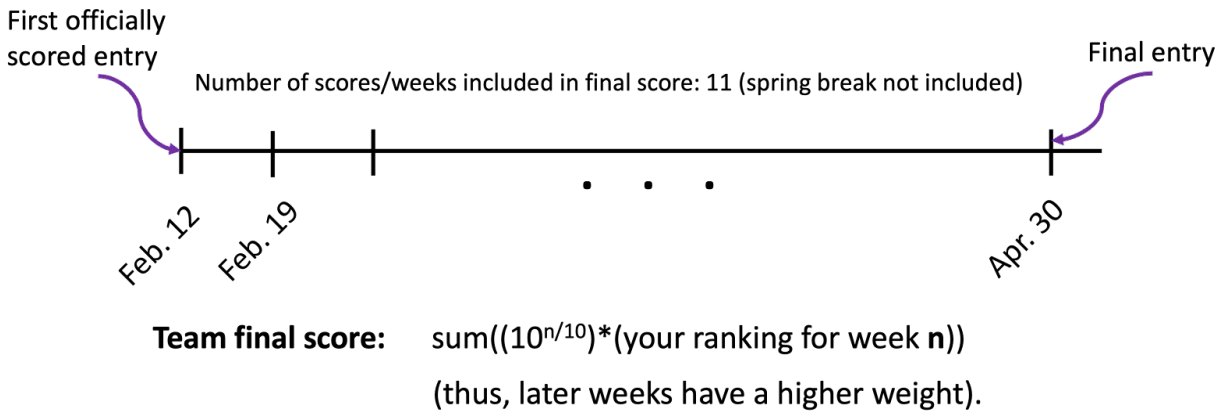


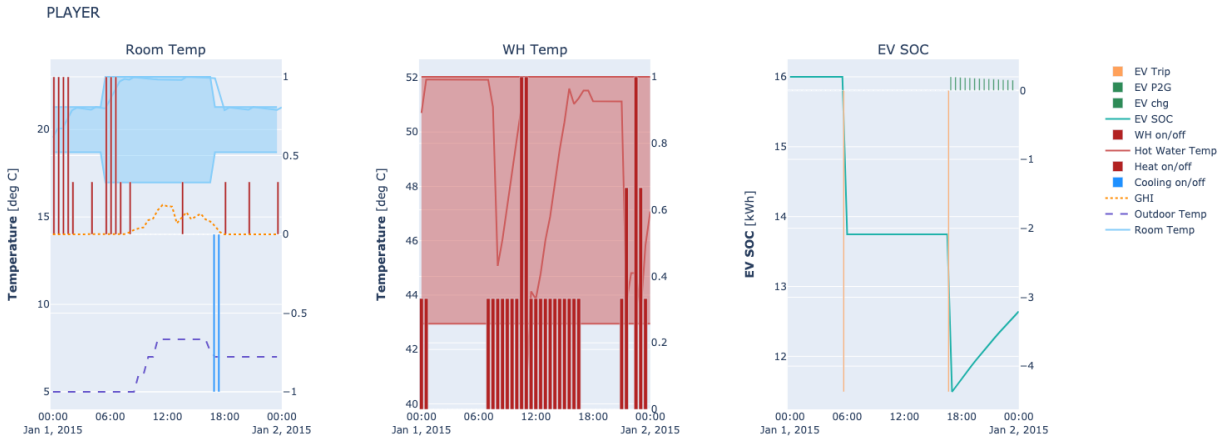
Figure 7.1: Timeline of the frequency of official score calculation and the final scoring composition.

- L-2 norm of the demand profile. The L-2 norm of the demand profile promotes energy savings and flattening of the individual consumer's load profile.

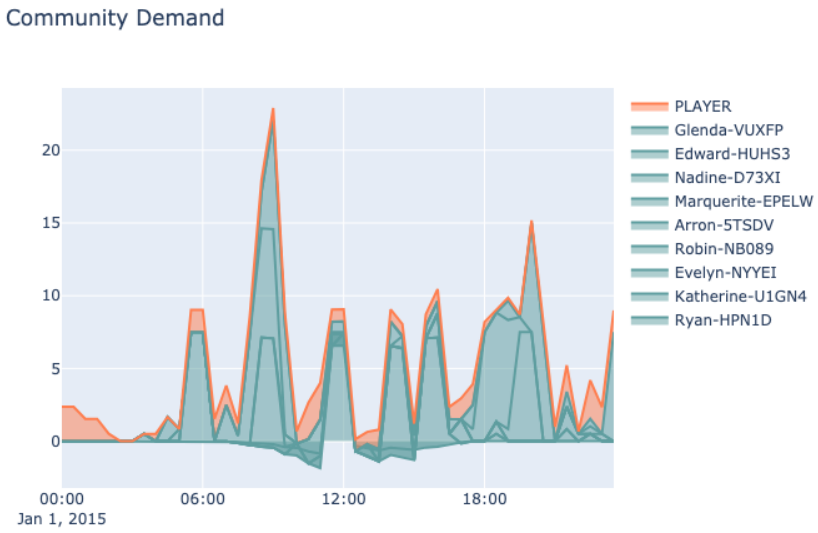
Additionally, the participants were graded on their overall composite score throughout the competition (based on rankings of the two primary categories) and on innovative methods for achieving performance goals (the "Golden Gnome Award"). The formula used to determine the final score is given in Fig 7.1.

The submission template has a training data set for an entire calendar year of environmental conditions (outdoor air temperature, global horizontal irradiance, day of week). The testing data set is taken from another arbitrary year and consists of the same environmental conditions. While the models of the players' homes are consistent between training and testing simulations, including the occupancy schedules, temperature bounds, resistance, and capacitance values of the home and water heater, the remainder of the community is randomly initialized to have similar but not identical properties in the testing phase. During each simulation checkpoint we change the environmental data to test performance in various seasons.

Users are encouraged to submit their agents earlier than the deadline in order to receive feedback on their performance using the test data. Feedback is given in the form of plots containing the players' homes' thermal performance and the community aggregate demand. The amount of



(a) Thermal performance of home energy system



(b)

Figure 7.2: Community aggregate demand profile

```python
def predict(home):
    if home.obs_dict['occupancy_status'] == 0:  # no one home
        hvac_action = 0  # neutral, leave the hvac alone
        wh_action = -1  # turn off the water heater
        if home.obs_dict['t_in_current'] < 65:
            hvac_action = 1
            ev_action = 1  # charge the car
        return [hvac_action, wh_action, ev_action]
    return home.action_space.sample()  # choose a random action
```

Figure 7.3: Minimum working example of a valid agent prediction function using if/else logic

feedback given reflects data that is available for the agent's state space and preserves the black box nature of the environment by obscuring the thermal dynamics. An example of these figures is given in Fig 7.2. Official scoring metrics are run at 5AM MST by pulling the relevant files from the players' GitHub repositories. The test simulation is identical to the sandbox simulation provided to the players with the exception of pulling a different weather file. The 5AM MST scores are the only scores that are ranked in the GNOMES scoreboard.

7.1.3 Agent Creation

GNOMES was deployed at CU Boulder as an undergraduate student competition in the Spring 2023 semester for those wanting to learn more about RL and/or the built environment. To that end, we created a submission template for students with little to no Python experience to make (a) rule-based controllers and/or (b) trained reinforcement learning agents. Both agents follow the same logical structure provided by the minimum working example provided in Fig 7.3

The full tutorial, as well as competition rules, are available here: `https://cugriffinlab.github.io/gnomes-submission`

7.2 Competition Results

GNOMES was hosted during the Spring 2023 semester, and limited to undergraduate students at CU Boulder. Participants were recruited through in-person visits to engineering classes, email campaigns to the Engineering Honors Program, and fliers posted around the Engineering Center. Based on a survey administered to the three regular participants (which we recognize is not a significant sample size), each team indicated that they heard about the competition in a different manner. Participants were incentivized to join with cash prizes for various performance-based categories, as well as a special innovation category. The cash prizes were significant (e.g., four prizes ranging from $2000-$4000). At the beginning of the competition, we had ten teams register and provide GitHub usernames. Throughout the course of the competition, however, we only had three active participants. These teams comprised students from the aerospace engineering, architectural engineering, and computer science departments. Participants ranged from sophomores to seniors, and all indicated that they had little or no experience with reinforcement learning and home energy management.

The winning strategy was provided by team "Green Foxes" with an RL-based approach; the other two competitors only explored the rule-based approach. Green Foxes made minimal changes to the supplied example files so future iterations could attempt to provide a lower performing example. Alternatively, we considered different testing datasets that would require more specific agent training to accommodate extreme weather events such as heat waves, cold snaps, or even grid changes like power outages.

Chapter 8

Conclusions and Future Work

In this thesis, we have presented five novel frameworks for reinforcement learning-based agents that improve the performance of distribution networks measured by efficiency and reliability. We explore improvements to the distribution network made through different perspectives, including grid operators and customers of the grid, both with and without the additional challenges of distributed generation, which is a hallmark of recent changes to the grid in the 21st century. We recognize that there is a significant amount of literature available studying both the impacts on the electric grid or energy customers from new incentive programs, efficiency goals, and/or reliability concerns. However we also note a significant gap in multi-disciplinary literature that covers the realistic impact of changes to the distribution network on utility customers and vice versa. In every chapter, we seek to close the gap in the literature by making more thorough and realistic models of the built environment via improved energy system models and electrical distribution models.

Each of the presented frameworks is accompanied by the open-sourced code for the given black box environments. We hope that the tools developed herein are useful for future generations of researchers in either building systems or power systems for rapidly prototyping distribution network control architecture for their own research purposes. In particular, we hope that two Python packages released as part of these works, DRAGG and DymolaGym, will continue to build a foundation for future work in energy management for a wide variety of research. Within this thesis alone, we have already expanded on DRAGG as a part of the GNOMES competition; the simplified R1C1 MPC thermal models are ideal for rapidly prototyping large-scale implementations

of new control paradigms and their impact on the broader electrical networks. Beyond this thesis, we have collaborated on using DymolaGym in thermal-electrical network optimization.

We already envision several extensions of the work presented here. We hope that GNOMES, in particular, can be further expanded to represent the reality of multi-player environments with competing and differing strategies. The computational architecture for GNOMES already supports parallel learning environments with multiple players feeding into the same aggregator. Additionally, we believe that the work presented for capacitor bank placement can be extended to place other resources like batteries for resiliency objectives or additional lines in transmission and distribution planning. State-dependent resources like batteries are potentially more suited to state-blind black box learning algorithms because they **require** time-linked simulations but will necessitate the inclusion of active control logic in addition to the existing ML-enabled placement algorithm.

Bibliography

[1] Heat pump water heater model validation study. https://ecotopewebstorage.s3.amazonaws.com/2015_001_1_HPWHModelVal.pdf, 2015. Prepared for the Northwest Energy Efficiency Alliance.

[2] 117th U.S. Congress. H.R.5376. https://www.congress.gov/bill/117th-congress/house-bill/5376/text, 2022.

[3] Kyri Baker. Solutions of dc opf are never ac feasible. In Proceedings of the Twelfth ACM International Conference on Future Energy Systems, e-Energy '21, page 264–268, New York, NY, USA, 2021. Association for Computing Machinery.

[4] Nicholas O. Bell, Jose I. Bilbao, Merlinde Kay, and Alistair B. Sproul. Future climate scenarios and their impact on heating, ventilation and air-conditioning system design and performance for commercial buildings for 2050. Renewable and Sustainable Energy Reviews, 162:112363, 2022.

[5] Vicki Bennett, Kate Bowman, and Sarah Wright. Quadrennial Technology Review: An assessment of energy technologies and research opportunities. Technical Report DOE-SLC-6903-1, Salt Lake City Corporation, Salt Lake City, UT, September 2018.

[6] Khaidem Bidyanath, Sanasam Dhanabanta Singh, and Shuma Adhikari. Implementation of genetic and particle swarm optimization algorithm for voltage profile improvement and loss reduction using capacitors in 132 kv manipur transmission system. Energy Reports, 9:738–746, 2023. 2022 9th International Conference on Power and Energy Systems Engineering.

[7] Henrik Bode, Stefan Heid, Daniel Weber, Eyke Hüllermeier, and Oliver Wallscheid. Towards a scalable and flexible simulation and testing environment toolbox for intelligent microgrid control. 2020.

[8] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. arXiv preprint arXiv:1606.01540, 2016.

[9] CAISO. What the duck curve tells us about managing a green grid. https://www.caiso.com/Documents/FlexibleResourcesHelpRenewables_FastFacts.pdf, Last accessed 11/16/2020.

[10] Iván Cantador and José M. Conde. Effects of competition in education : a case study in an e-learning environment. 2010.

[11] Wuzhu Chen, Michaela Huhn, and Peter Fritzson. A Generic FMU Interface for Modelica. In 4th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools, EOOLT 2011, pages 19–24, 2011.

[12] Chan-Jin Chung. Learning through competitions - competition based learning (cbl). 2008.

[13] Commonwealth Edison. Real-time hourly prices. Technical report, `https://hourlypricing.comed.com/live-prices/`, Last accessed May 2020.

[14] Rezi Delfianti, Ontoseno Penangsang, Adi Soeprijanto, Nasyith Hananur Rohiem, Novian Patria Uman Putra, and Titiek Suheta. Application of particle swarm optimization algorithm for scheduling of capacitor bank switching to improve voltage. In 2021 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), pages 245–249, 2021.

[15] Paul L. Denholm, Jacob Nunemaker, Wesley J. Cole, and Pieter J. Gagnon. The potential for battery energy storage to provide peaking capacity in the united states. 6 2019.

[16] Shady A. El-Bataway and Walid G. Morsi. Distribution transformer's loss of life considering residential prosumers owning solar shingles, high-power fast chargers and second-generation battery energy storage. IEEE Transactions on Industrial Informatics, 15(3):1287–1297, 2019.

[17] Damien Ernst, Mevludin Glavic, Florin Capitanescu, and Louis Wehenkel. Reinforcement learning versus model predictive control: A comparison on a power system problem. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 39(2):517–529, 2009.

[18] Ahmad Faruqui, Ryan Hledik, Sam Newell, and Hannes Pfeifenberger. The power of 5 percent. The Electricity Journal, 20(8):68 – 77, 2007.

[19] David M. Fobes, Sander Claeys, Frederik Geth, and Carleton Coffrin. Powermodelsdistribution.jl: An open-source framework for exploring distribution power flow formulations. Electric Power Systems Research, 189:106664, 2020.

[20] Elham Foruzan, Leen-Kiat Soh, and Sohrab Asgarpoor. Reinforcement learning approach for optimal distributed energy management in a microgrid. IEEE Transactions on Power Systems, 33(5):5749–5758, 2018.

[21] Kaitlyn Garifi, Kyri Baker, Dane Christensen, and Behrouz Touri. Control of energy storage in home energy management systems: Non-simultaneous charging and discharging guarantees, 2018.

[22] Kaitlyn Garifi, Kyri Baker, Dane Christensen, and Behrouz Touri. Stochastic home energy management systems with varying controllable resources. In IEEE Power Energy Society General Meeting (PESGM), 2019.

[23] Kaitlyn Garifi, Kyri Baker, Dane Christensen, and Behrouz Touri. Convex relaxation of grid-connected energy storage system models with complementarity constraints in DC OPF. IEEE Transactions on Smart Grid, pages 1–1, 2020.

[24] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. 2018. [Online] Available: https://arxiv.org/abs/1801.01290.

[25] Hassan Harb, Neven Boyanov, Luis Hernandez, Rita Streblow, and Dirk Müller. Development and validation of grey-box models for forecasting the thermal response of occupied buildings. Energy and Buildings, 117:199–207, 2016.

[26] Stefan Heid, Daniel Weber, Henrik Bode, Eyke Hüllermeier, and Oliver Wallscheid. OMG: A Scalable and Flexible Simulation and Testing Environment Toolbox for Intelligent Microgrid Control. Journal of Open Source Software, 5(54):2435, October 2020.

[27] Ashley Hill et al. Stable baselines. `https://github.com/hill-a/stable-baselines`, 2018.

[28] Shira Horowitz and Lester Lave. Equity in residential electricity pricing. The Energy Journal, 35(2):1–24, 2014.

[29] Xin Jin, Kyri Baker, Dane Christensen, and Steven Isley. Foresee: A user-centric home energy management system for energy efficiency and demand response. Applied Energy, 205:1583 – 1595, 2017.

[30] William Kersting. Radial distribution test feeders. volume 6, pages 908 – 912 vol.2, 02 2001.

[31] Byung-Gook Kim, Yu Zhang, Mihaela van der Schaar, and Jang-Won Lee. Dynamic pricing and energy consumption scheduling with reinforcement learning. IEEE Transactions on Smart Grid, 7(5):2187–2198, 2016.

[32] Xiao Kou, Fangxing Li, Jin Dong, Michael Starke, Jeffery Munk, Teja Kuruganti, and Helia Zandi. A distributed energy management approach for residential demand response. In 3rd Int. Conf. on Smart Grid and Smart Cities, 2019.

[33] Ole Kröger, Carleton Coffrin, Hassan Hijazi, and Harsha Nagarajan. Juniper: An open-source nonlinear branch-and-bound solver in julia. In Integration of Constraint Programming, Artificial Intelligence, and Operations Research, pages 377–386. Springer International Publishing, 2018.

[34] Zachary E. Lee and K. Max Zhang. Unintended consequences of smart thermostats in the transition to electrified heating. Applied Energy, 322:119384, 2022.

[35] Helen Lo, Seth Blumsack, Paul Hines, and Sean Meyn. Electricity rates for the zero marginal cost grid. The Electricity Journal, 32(3):39 – 43, 2019.

[36] Renzhi Lu and Seung Ho Hong. Incentive-based demand response for smart grid with reinforcement learning and deep neural network. Applied Energy, 236:937 – 949, 2019.

[37] Oleh Lukianykhin and Tetiana Bogodorova. ModelicaGym: Applying Reinforcement Learning to Modelica Models. In Proceedings of the 9th International Workshop on Equation-based Object-oriented Modeling Languages and Tools, EOOLT '19, pages 27–36, New York, NY, USA, November 2019. Association for Computing Machinery.

[38] Trieu T. Mai, Paige Jadun, Jeffrey S. Logan, Colin A. McMillan, Matteo Muratori, Daniel C. Steinberg, Laura J. Vimmerstedt, Benjamin Haley, Ryan Jones, and Brent Nelson. Electrification futures study: Scenarios of electric technology adoption and power consumption for the united states.

[39] Sujit Mandal and Venkat S. Kolluri. Coordinated capacitor bank switching using svc controls. In 2008 IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century, pages 1–7, 2008.

[40] Mohammad A.S. Masoum, Marjan Ladjevardi, Akbar Jafarian, and Ewald F. Fuchs. Optimal placement, replacement and sizing of capacitor banks in distorted distribution networks by genetic algorithms. IEEE Transactions on Power Delivery, 19(4):1794–1801, 2004.

[41] Hugo Morais, Tiago Pinto, Zita Vale, and Isabel Praça. Multilevel negotiation in smart grids for vpp management of distributed resources. IEEE Intelligent Systems, 27(6):8–16, 2012.

[42] Cory Mosiman and Aisling Pigott. Distributed resource aggregation (DRAGG). `https://github.com/corymosiman12/dragg`, 2020.

[43] Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron C. Courville. The primacy bias in deep reinforcement learning. In International Conference on Machine Learning, 2022.

[44] Ronald Ortner and Daniil Ryabko. Online regret bounds for undiscounted continuous reinforcement learning, 2013.

[45] Aisling Pigott, Kyri Baker, Sergio Dorado-Rojas, and Luigi Vanfretti. Dymola-enabled reinforcement learning for real-time generator set-point optimization. IEEE Power and Energy Society Innovative Smart Grid Technologies, 2022.

[46] Aisling Pigott, Kyri Baker, and Cory Mosiman. Deep Q-learning for aggregator price design. IEEE Power and Energy Society General Meeting, 2021.

[47] Aisling Pigott, Constance Crozier, Kyri Baker, and Zoltan Nagy. Gridlearn: Multiagent reinforcement learning for grid-aware building energy management. Electric Power Systems Research, 213:108521, 2022.

[48] Qais Atef Qawaqneh. Impact of 11kv capacitor bank switching at distribution power network from power quality perspective. In 2020 19th International Conference on Harmonics and Quality of Power (ICHQP), pages 1–6, 2020.

[49] Ashwin Ramdas, Kevin McCabe, Paritosh Das, and Sigrin Benjamin. California time-of-use (tou) transition: Effects on distributed wind and solar economic potential. 3 2020. NREL/TP-6A20-73147. `https://www.nrel.gov/docs/fy19osti/73147`.

[50] Mark Ruth, Annabelle Pratt, Monte Lunacek, Saurabh Mittal, Hongyu Wu, and Wesley Jones. Effects of home energy management systems on distribution utilities and feeders under various market structures. In Proc. CIRED 23rd International Conference and Exhibition on Electricity Distribution, 2015.

[51] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[52] Manajit Sengupta, Yu Xie, Anthony Lopez, Aron Habte, Galen Maclaurin, and James Shelby. The national solar radiation data base (NSRDB). Renewable and Sustainable Energy Reviews, 89:51 – 60, 2018.

[53] Mohamed H. Shwehdi, Somaia Raja Mohamed, and Durairaj Devaraj. Optimal capacitor placement on west–east inter-tie in saudi arabia using genetic algorithm. Computers Electrical Engineering, 68:156–169, 2018.

[54] Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction, chapter 2. The MIT Press, second edition, 2017.

[55] Ujjwol Tamrakar, Dipesh Shrestha, Manisha Maharjan, Bishnu Bhattarai, Timothy Hansen, and Reinaldo Tonkoski. Virtual inertia: Current trends and future directions. Applied Sciences, 7(7), 6 2017.

[56] U.S. Department of Energy. Sizing a new hot water heater. `https://www.energy.gov/energysaver/water-heating/sizing-new-water-heater`, Last accessed 11/5/2020.

[57] Hado van Hasselt. Double Q-learning. Advances in Neural Information Processing Systems, pages 2613–2621, 2010.

[58] José R. Vázquez-Canteli, Jérôme Kämpf, Gregor Henze, and Zoltan Nagy. Citylearn v1.0: An openai gym environment for demand response with deep reinforcement learning. In ACM BuildSys, page 356–357, 2019.

[59] Tom Verhoeff. The role of competitions in education. 12 1997.

[60] Zhifang Yang, Anjan Bose, Haiwang Zhong, Ning Zhang, Jeremy Lin, Qing Xia, and Chongqing Kang. Lmp revisited: A linear model for the loss-embedded lmp. IEEE Transactions on Power Systems, 32(5):4080–4090, 2017.

[61] Ying Zhang, Xinan Wang, Jianhui Wang, and Yingchen Zhang. Deep reinforcement learning based volt-var optimization in smart distribution systems. IEEE Transactions on Smart Grid, 12(1):361–371, 2021.