UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

DEVELOPMENT AND ASSESSMENT OF CONSTRAINED REINFORCEMENT LEARNING-BASED CONTROLLER FOR BUILDING DEMAND RESPONSE

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE

By

JERSON SANCHEZ
Norman, Oklahoma
2023

DEVELOPMENT AND ASSESSMENT OF CONSTRAINED REINFORCEMENT LEARNING-BASED CONTROLLER FOR BUILDING DEMAND RESPONSE

A THESIS APPROVED FOR THE
SCHOOL OF AEROSPACE AND MECHANICAL ENGINEERING

BY THE COMMITTEE CONSISTING OF

Dedicated

to my parents.

ACKNOWLEDGMENTS

I express my deepest gratitude to the boundless grace of God, which guides me throughout each day and every project I start. All of my accomplishments are dedicated to His glory.

I would like to acknowledge the financial support from NSF under award #2238381 and DOE under award #DE-SC0023498.

A special acknowledgment is to my advisor, Dr. Cai, who invested many hours advising me on technical and professional aspects. His patience, understanding and commitment to high standards have shaped me into a better researcher . The lessons learned and experiences gained in Dr. Cai's lab are a timeless source of inspiration. The impact of this experience will benefit me throughout my lifetime.

I am equally grateful to my esteemed committee members, Dr. Song and Dr. Zhang, whose invaluable time and expertise contributed significantly to this endeavor. Their thoughtful engagement and discerning insights enriched the depth and quality of my work.

Finally, my heartfelt gratitude to my parents for their unconditional support and sacrifice through this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Recent advancements in model-free control strategies, such as reinforcement learning (RL) have led to more practical and scalable solutions for building energy system controls These strategies do not require complex models of building dynamics and rely exclusively on data to learn the control policy. Applications of these techniques in heating, ventilation and air-conditioning (HVAC) systems are being studied under different operational scenarios, including demand response programs. Conventional (unconstrained) reinforcement learning controllers often address indoor comfort constraints by incorporating a comfort violation penalty in the reward function. While this approach can result in good performance in terms of energy cost, it often leads to significant constraint violations when a small penalty factor is used. On the other hand, effective enforcement of constraints can be achieved, but at the cost of economic performance degradation. Hence, a clear trade-off between economic performance and constraint satisfaction poses a challenge to overcome. Motivated by this challenge, this thesis presents a constrained RL-based control strategy for building demand response. The proposed strategy handles the constraints explicitly, avoiding the use of arbitrarily set penalty factors that can significantly impact control performance. To demonstrate its efficacy, simulation tests of the proposed strategy, as well as baseline model predictive controllers (MPC) and conventional (unconstrained) policy optimization methods, were conducted. The simulation tests showed that the constrained RL strategy achieved utility cost savings up to 16.1%, similar to the MPC baselines, without requiring any model of the building and with minimum constraint violation. In contrast, the unconstrained RL controllers led to either high utility costs or constraint violations, depending on the penalty factor setting.

CHAPTER 1

INTRODUCTION

Building electrical loads represent 75% of the electricity consumption in the United States [1]. Technologies and solutions for efficient and sustainable building operations have witnessed growing attention in the past few years, with heating, ventilation and air conditioning (HVAC) demand response being one of them. Demand response plays an important role in improving electric power system reliability against uncertainties in renewable generation, peak demand, asset availability, and other grid contingent conditions [2, 3].

The concept of demand response, originally rooted in the power utility sector, has found its way into the realm of building control systems as a means to enhance energy resilience, reduce peak load demands, and mitigate the environmental impact of energy consumption. It provides a mechanism for building systems to adapt in real-time to changing energy supply and demand conditions, allowing them to intelligently and autonomously adjust their energy consumption patterns.

Model predictive control (MPC) is a broadly used technique in demand response control of HVAC systems. While MPC offers a powerful framework that can explicitly consider constraints during control decision making, a decent process model is needed to achieve satisfactory control performance, the development of which would require significant engineering costs, especially for complex systems such as buildings. Reinforcement learning (RL) offers a promising model-free control approach that has been successfully applied in different fields including building energy management [4].

CHAPTER 2

LITERATURE REVIEW

MPC has become a popular approach in improving energy efficiency and optimizing performance of building systems. Researchers have extensively explored the application of MPC in a range of applications which include HVAC systems and building controls for energy cost and occupant comfort optimization [5, 6]. An extensive and detailed review of MPC applications in buildings can be found in [7].

In the context of demand response, MPC has been researched to reduce peak energy demand under a real time pricing (RTP) regime for building climate control and proved to be effective on reducing overall electricity costs [8]. MPC was also used under demand response to optimize the battery energy storage system and HVAC system schedules to minimize annual electricity cost using MPC on an EnergyPlus building model [9]. Studies have also focused on improving MPC performance in buildings. A study compared different MPC formulations for commercial buildings under time of use rates. These proposed formulations showed to effectively reduce energy costs and increase load-shifting capacity under demand response events, while maintaining low peak demands [10]. In order to determine the maximum potential savings with MPC for commercial buildings under demand response, the effects of look-ahead horizon, timing of peak demand and the initial peak demand target have been studied. It was found that a 24 hour look- ahead horizon can achieve up to 78% of the cost savings of a one shot monthly optimization [11]. While MPC is a promising control approach, a complex and accurate model of the system is needed for its implementation. Hence, model-free approaches are an attractive alternative when it is difficult or impossible to obtain an accurate model or the cost and engineering efforts required to obtain one are prohibitive.

RL offers a model-free control approach and numerous studies have researched the application of RL techniques to building controls. These studies have highlighted the advantages of RL over model-based control methods as well as challenges for its real-life implementation. One of these challenges is that the training process can be time-consuming and require large data sizes. Model-based RL is a potential solution to address this problem since data can be generated using the model at no cost. However, this requires a complex model of the building and suffers from high upfront costs for model development, similar to model-based control methods such as MPC. Ensuring the safety of the controls during the training phase so that dangerous actions are not taken is also necessary. Lastly, the potential of improving RL algorithms by using transfer learning for generalization and the use of trained building policies on other similar buildings is an opportunity identified by many studies to accelerate training and enable real-life implementation of RL algorithms [12, 13].

The earliest works in RL for HVAC systems used Q-learning techniques and highlighted the challenges of RL techniques that need to be overcome and the direction the research in this field could take. The use of Q-learning for the operation of thermal energy storage in commercial buildings was investigated in [14]. This study found that for time of use (TOU) rates, the RL controller achieved 8.1% of cost savings at a cost of 3.5% increase in total energy use and 13.9% for real-time pricing at a cost of 6.2% increase of energy use. While Q-learning showed the capability to learn a good policy, the algorithm applicability was found to be limited due to the size of the Q-table increasing considerably with the action and observation space size, which makes the problem intractable. At the same time, the training time for achieving an acceptable control policy was found to be considerable and does not allow for practical application of the algorithm [15, 16].

With the increase in available computation power and the development of new algorithms, such as deep Q-networks (DQN) and policy optimization, better performance could be achieved even for applications in the control of complex systems such as HVAC systems under demand response programs. For instance, early efforts to use deep RL sought to minimize the electricity usage cost of the HVAC system. However, they utilized the DQN algorithm without considering different temperature bounds for occupied and unoccupied hours and the algorithm required hundreds of months of data to converge [17]. In another study, DQN was used to minimize building energy use while maintaining CO_2 concentration levels under an acceptable ppm threshold [18]. Seeking to improve the learning of the DQN algorithm, an action processor to leverage previously known information from rule-based controllers to reduce training time was used [19]. Better performance were shown using algorithms similar or derived from DQN such as deep deterministic policy gradient (DDPG) and asynchronous advantage actor critic (A3C). A study in the context of a locational marginal price in a multi-zone residential HVAC system used DDPG to minimize energy costs [20]. In another study, DDPG was also used in a smart home energy management system [21]. Asynchronous advantage actor-critic methods were applied for demand response subject to demand charges in a multi-zone commercial building [22]. DQN strategies have also been implemented and demonstrated in real buildings, e.g., a DQN strategy pre-trained by a simulation model was deployed in a residential building demonstrating a potential of 10 to 20% cost savings [23].

Policy optimization algorithms unlike value-based algorithms such as DQN directly learn a policy from data samples of the building operation and have also been studied in the context of demand response. A TRPO controller was proposed to be used in residential appliances scheduling in residential buildings. The controller effectively generated a control policy for the appliances although it did not consider occupancy-based comfort temperature bounds [24]. A similar study used a multi-agent TRPO algorithm for an autonomous home energy management system (HEMS). The algorithm is trained to optimize the scheduling of the appliances considering the dynamic retail price and the local power generation and showed better performance than simpler policy optimization algorithms such as REINFORCE. A two-stage RL training framework was proposed integrating evolutionary strategies and proximal policy optimization (PPO) to address a grid-interactive building control problem [25]. Pure PPO algorithms were also tested under different demand response scenarios in a building simulation tool (EnergyPlus) showing energy reduction of up to 22% and peak demand reduction of up to 50% [26]. On a different application, PPO was used to find optimal indoor airflow to minimize exposure of the building occupants to pathogents such as COVID-19 and describes the advantages of using a model-free approach against computationally complex CFD models [27].

All the studies above used unconstrained RL that cannot directly handle constraints, e.g., capacity and indoor comfort constraints, during control decision making. In unconstrained RL implementations, constraints are often addressed heuristically, e.g., by adding a penalty for constraint violations in the reward function. However, the control performance is sensitive to the arbitrarily set penalty factor for indoor discomfort. It is difficult to put a price tag on thermal discomfort as individuals would have different perceptions and tolerances thereof. The local utility rate structures could further complicate the situation as the economic consequence can highly depend on the energy rates as well as the peak-to-off-peak ratio.

Constrained RL has arisen as a promising framework to address the challenges for applications that require constraints satisfaction. Constrained RL application has been studied in other fields. A safety layer on top of a hierarchical approach for safe robot navigation where the safety layer maps the actions proposed by the low-level controller and the constraint violation values to propose a safer action was proposed in [28]. Constrained RL has been proposed for electric vehicle (EV) charging applications as well. The EV charging/discharging scheduling problem in a demand response context as a constrained Markov decision process (CMDP) and uses the constrained policy optimization algorithm to obtain a safe policy was formulated in [29]. This approach achieved the best performance in charging cost and performed 10 times better on satisfying the constraint when compared with the safest baseline. In another field, a flight decision optimization problem aiming to minimize the searching time is formulated as a CMDP [30]. This study used a UAV tracking scheme using Gaussian process regression to estimate reference points and used it jointly with multi-agent Q-learning to make flight decisions. These studies showed that constrained RL offers an effective framework to solve problems that satisfy constraints while achieving good performances.

Contributions of this work

This thesis reports the challenges of applying unconstrained RL algorithms to building HVAC control: (1) sensitivity to the choice of the penalty factor, and (2) challenges in benchmarking these algorithms with other techniques such as MPC-based strategies without guaranteed comfort constraint satisfaction. To overcome the challenges, this thesis presents a constrained RL-based strategy for building demand response and its numerical test results in comparison with baseline unconstrained RL strategies as well as MPC. In addition, this thesis also seeks to show that linear policies can work as well or better than deep neural network-based policies in terms of convergence and performance depending on the problem to be solved. To the author's knowledge, this is the first work that applied constrained RL strategies to tackle these challenges in building demand response.

This thesis is structured as follows. First, the numerical building model used as the emulator that interacts with each RL agent is presented in chapter 3. Secondly, MPC is defined and the MPC baselines are formulated in chapter 4. Next, chapter 5 introduces the general RL concept along with the state-of-the-art algorithms for both unconstrained and constrained policy optimization. Next, chapter 6 reports the simulation test results comparison with the conventional RL and MPC baselines. Concluding remarks are provided in chapter 7.

CHAPTER 3

BUILDING SYSTEM MODEL

This section introduces the building dynamic model used for control testing and MPC synthesis. Note that in the MPC implementation, the same model is assumed for the plant and control synthesis; therefore, the presented control performance represents the theoretical upper bound and mainly serves the benchmarking purpose.

A discrete-time state-space model is used to reproduce building thermal dynamics, based on a thermal network approach [31]. The model uses the cooling rate as the control input and outputs the zone temperature, in the following form:

$$x^{t+1} = \mathbf{A}x^t + \mathbf{B_w}w^t + \mathbf{B_u}Q_z^t, \tag{3.1}$$

$$T_z^{t+1} = \mathbf{C}x^{t+1}, \tag{3.2}$$

where x^t is the state vector containing all nodal temperatures of a thermal network, Q_z^t is the average cooling rate within each time step, T_z^t is the zone temperature, and w^t is the disturbance vector comprised of outdoor temperature, internal heat gains, and solar radiation. The state-space matrices $\mathbf{A}$, $\mathbf{B_w}$, $\mathbf{B_u}$ and $\mathbf{C}$ are dependent on the thermal resistances and capacitances of the thermal network. The model adopts a thermal network approach with the network shown in Figure 3.1, which consists of two main wall branches: the floor branch (Flr) that represents most of the thermal storage in the building construction; the external wall (Ext) that bridges the indoor space to the ambient. The radiative internal heat gain ($Q_{gain,rad}$) is applied to the floor and external wall with a uniform heat flux while the convective internal heat gain ($Q_{gain,conv}$) interacts with the air node directly. A facade branch is used to capture the dynamics of the facade cavity temperature ($T_{fac,space}$). The facade branch also captures fasting couplings between the indoor, facade and outdoor spaces including infiltration and heat transfer through windows. However, the case study building is a typical commercial office space with negligible infiltration. The solar radiation transmitted through the windows into the indoor space ($Q_{sol,trans}$) and the facade space ($Q_{sol,trans,fac}$) are also modeled where the window transmittance is estimated in the training process.

This particular thermal network was obtained after rigorous identifiability analysis following the procedure detailed in [32], which uses norms of the Fisher information matrix to evaluate the informativeness of training data relative to a given model structure. More complex model structures (e.g., with more RC branches) may provide better match to experimental data but suffer from poor structural identifiability. Low-order models come with better identifiability but may not capture all the dynamics. The thermal network structure adopted here achieves a balance between accuracy and identifiability. The training data includes building operation data from the building management system, weather data collected in a weather station on the building rooftop, and sub-circuit power meters that measure the plug loads and lighting power usage. Further details of the training methodology and the identified parameter values can be found in [31]. The resultant building load model uses the cooling rate as input and predicts the zone temperature.

The model is used as a simulation test bed to evaluate the various control strategies. The same model is adopted for the MPC implementation, while in RL-based control, the system dynamics are not known to the agent and the control policy is learned from the recorded interactions between the agent and the plant model that include control commands sent to the building and the measurable observations.

The power used by the HVAC system can be estimated from the cooling rate by

$$P^t = \frac{Q_z^t}{COP}, \tag{3.3}$$

where the coefficient of performance (COP) assumes a constant in this study. Equation 3.1 to Equation 3.3 define the environment which the agent interacts with. The agent applies a temperature setpoint and a set of measurable observations are available to the agent after execution of the set-point. In this study, only a cooling scenario is considered although the methods can be directly applied for heating seasons.

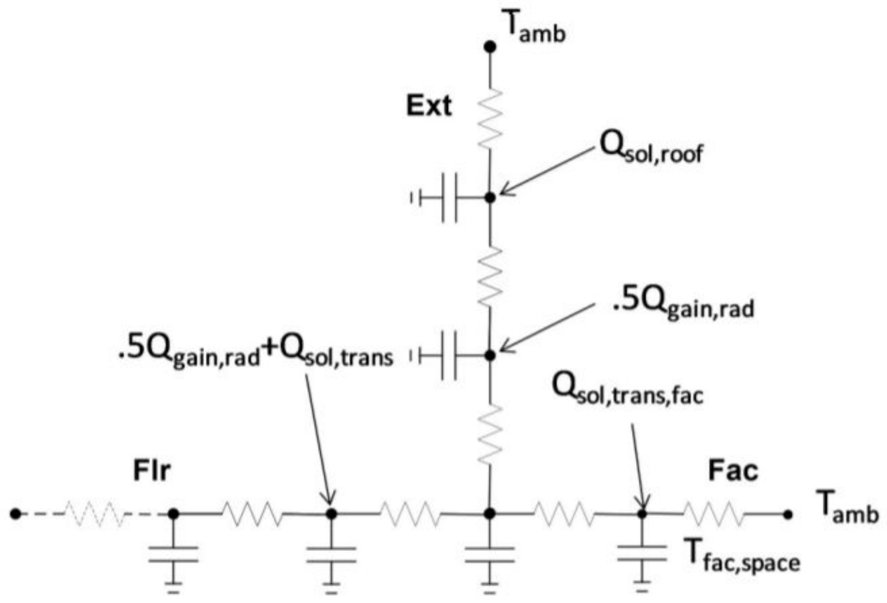

Figure 3.1: Thermal network model

CHAPTER 4

MODEL PREDICTIVE CONTROL

This chapter introduces the MPC method which is broadly used in building control applications including demand response. It serves as a benchmark for evaluating the RL-based control performance in this thesis. This section first describes the general concept of MPC and then lays out the formulation used with the building system model. MPC optimizes system performance by using a dynamic model to predict future behavior over a specified time horizon. Key components include a dynamic model, a cost function, and constraints. At each time step, an optimization problem is solved to find optimal control inputs, considering the dynamic model and constraints over a moving horizon. MPC is advantageous for handling complex systems, managing constraints, and adapting to changes in real-time. For building demand response applications, an MPC uses the forecasts of the outdoor temperature, solar radiation and convective and radiation internal heat gains and identifies the optimal control (zone temperature set-point) trajectory within the look-ahead time horizon (e.g., 24 hours in this thesis). Only the control decision of the first time step is implemented, while this control decision procedure is repeated when next time step arrives. Two control objectives are considered for bench-marking purposes.

4.1 Energy Minimization Control (E-Min)

The first baseline strategy represents the current practice for maximum energy efficiency. This strategy tends to maintain the zone temperature as close to the upper limit of the comfort temperature zone as possible to minimize the HVAC energy consumption of each time step, with the following cost function:

$$W_1 = P^t. \tag{4.1}$$

The energy minimization strategy represents a greedy control policy over the HVAC energy use with only one step ahead prediction. A number of operational constraints should be respected in control decision making. Upper and lower limits are imposed for the zone temperature in order to meet the indoor comfort requirements:

$$T^t_{z,min} \leq T^t_z \leq T^t_{z,max}, \tag{4.2}$$

where $T^t_{z,min}$ and $T^t_{z,max}$ are the lower and upper bounds of the comfort band. These temperature limits can vary with the occupancy status: when the building is occupied, a tighter temperature constraint is needed to ensure comfort, while during unoccupied hours, relaxed temperature bounds can be used to achieve energy or cost savings. The cooling rate at each time step should be bounded by the cooling capacity Q_T, which is assumed to be time-invariant in this study:

$$0 \leq Q^t_z \leq Q_T. \tag{4.3}$$

4.2 Utility Cost Minimization Control (U-Min)

The second baseline strategy minimizes the electric utility cost under time-of-use rates over a look-ahead time horizon (e.g., 24 hours in the case study) and represents the current practice for predictive demand responsive control. This strategy can be implemented by solving an LP with a cost function in the following form:

$$W_2 = \sum_{t=t_i}^{t_f} (u^t \cdot P^t), \tag{4.4}$$

where u^t is the retail energy rate ($\$/\text{kWh}$) that may change with time of the day, t_i and t_f are the first and last time steps of the prediction horizon. The same constraints discussed for the E-min strategy are also present in this cost-minimizing strategy, over the whole look-ahead time horizon.

CHAPTER 5

REINFORCEMENT LEARNING CONTROL

This chapter introduces the RL framework and algorithms used and formulates the optimization problem that each algorithm seeks to solve. First, the Markov decision process (MDP) is defined and its basic elements are described in the context of HVAC system control. Next, important RL definitions used by the proposed algorithms, such as value functions, are defined. It is followed by a technique to formulate the partially observable environment as an MDP. In the second part of this chapter, unconstrained policy gradient methods are described starting with vanilla policy gradient, continuing with TRPO and finishing with PPO. Lastly, a constrained Markov decision process, the backbone of the control strategy proposed in this work, to minimize energy costs while enforcing constraints on the zone temperature is presented.

5.1 Building Control as a Markov Decision Process

This section introduces the fundamental framework used in RL. MDPs represent a system as a set composed by states, actions and transition probabilities following the Markov property that implies that any future state is only dependent of the current state and action. The agents' goal in MDPs is to find a policy that maps actions from states such that an objective is maximized. In the context of the building HVAC system control, the basic elements of the MDP are defined as follows:

5.1.1 States

The states (s_t) are the observations available to the agent obtained through interaction with the environment. At every 15-min time step, a set of observations is available to the agent, which includes the current zone temperature T_z^t, as well as the past six-step zone temperatures $T_z^{t-1}, ..., T_z^{t-6}$; the reason of including the previous zone temperature measurements is discussed in section 5.3. The observation set also contains the hour of the day h^t, and the 24-hour forecasts of the outdoor temperature $T_{out}^t, ..., T_{out}^{t+95}$ as well as the global horizontal solar radiation $q_{sol}^t, ..., q_{sol}^{t+95}$ and the power P^t required to achieve the temperature set-point. Note that these observations are measurable and do not depend on the building system model.

$$s_t = \left\{ T_z^t, \dots, T_z^{t-6}, h^t, T_{out}^t, \dots, T_{out}^{t+95}, q_{sol}^t, \dots, q_{sol}^{t+95}, P^t \right\}.$$

5.1.2 Actions

The actions (a_t) are the control inputs that the agent takes given a state s_t. The action taken by the agent is the zone temperature setpoint T_{sp}^{t+1} for the next decision step

$$a_t = T_{sp}^{t+1}.$$

The action space is a discrete space with p elements equally spaced between the upper and lower zone temperature comfort bounds.

$$A = \{\underbrace{T_{z,\min}, \dots, T_{z,\max}}_{\text{p available actions}}\}.$$

5.1.3 Transition probability

The transition probability ($P(s_{t+1}|s_t, a_t)$) is the probability of being in state s_{t+1} after taking an action a_t being in state s_t. Since the building system environment is deterministic this probabilities are not considered in the problem formulation.

5.1.4 Rewards

The reward (r_t) is a signal that depends on the current and next step states and the action taken. It represents how good or bad the state-action pair is. The goal of the RL agent is to maximize the reward (negative of the cost) under a time-of-use tariff while maintaining the zone temperature within the comfort bounds.

Unconstrained RL: Conventional (unconstrained) RL techniques cannot explicitly handle control constraints. In almost all previous building control applications, the comfort constraints were addressed as a soft penalty cost added to the energy cost with a prescribed weighting factor.The reward associated with the energy cost is defined as:

$$r^t_{cost} = -u_t P^t, \tag{5.1}$$

where u_t is the time of use energy usage price. The reward associated with the comfort violation penalty is considered in the following form:

$$r^t_{com} = -\psi\big(\max(T^t_{z,min} - T^t_z, 0) + \max(T^t_z - T^t_{z,max}, 0)\big), \tag{5.2}$$

where ψ is the constraint violation penalty factor. This comfort penalty is proportional to the cumulative temperature excursions out of the comfort band. The overall reward for the unconstrained RL case is $r_t = r^t_{cost} + r^t_{com}$.

5.2 Reinforcement Learning Key Concepts

This section describes the key concepts that most RL algorithms use. These concepts are well know by RL researchers and deeper theory about these can be found in [33].

5.2.1 Policy

A policy represents a map that take the observed states as inputs and outputs a set of probabilities if the action space is discrete, or a pair containing the mean and standard deviation of the action if the action space is continuous. In this work the policy is a discrete set, hence it is represented as:

$$\pi(s_t, a_t) = P(a_t|s_t), \tag{5.3}$$

where $P(a_t|s_t)$ is the probability of taking action a_t given a state s_t.

5.2.2 Return

A trajectory of an episode consists of the collected states, actions and rewards under the current policy.

$$\tau = \{s_0, a_0, r_0, s_1, a_1, r_1 \ldots s_T, a_T, r_T\}, \tag{5.4}$$

where τ is a trajectory and T is the number of time steps taken of that trajectory. The return of a trajectory is defined as the discounted or undiscounted cumulative reward for the episode.

$$R(\tau) = \sum_{t=0}^{T} \gamma^t r_t, \tag{5.5}$$

where γ is the discount factor which determines the importance of future actions.

5.2.3 Value Function

The value function of a state is defined as the expected discounted or undiscounted cumulative reward collected during an episode τ following a policy π given a state s_t.

$$V^{\pi}(s_t) = \mathbb{E}_{\tau \sim \pi}\left[\sum_{l=0}^{T} \gamma^k r_{t+l+1} | s_t\right]. \tag{5.6}$$

5.2.4 Action Value Function

The action value function of a state and action is defined as the expected discounted or undiscounted cumulative reward collected during an episode following a policy π given a state and the action taken in that state:

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{\tau \sim \pi}\left[\sum_{l=0}^{T} \gamma^k r_{t+l+1} | s_t, a_t\right]. \tag{5.7}$$

Early stage tabular RL techniques use a table to store the action value function for each state and action. While this technique is simple and straightforward to implement, its computation burden dramatically increases with the size of the table which limits its application to complex problems. To address this issue, parameterized functions are used to represent the state value functions, action value functions and policies.

$$V^{\pi}(s_t) \approx V^{\pi}_{\phi}(s_t), \tag{5.8}$$

$$Q^{\pi}(s_t, a_t) \approx Q^{\pi}_{\phi_q}(s_t, a_t), \tag{5.9}$$

$$\pi(s_t, a_t) \approx \pi_{\theta}(s_t, a_t), \tag{5.10}$$

where ϕ, ϕ_q and θ are the parameters of each function. While several types of parameterization can be used to represent these functions, this work uses deep neural networks and linear mapping to represent them.

5.2.5 Advantage Function

An advantage function $A^{\pi}(s_t, a_t)$ quantifies the benefit of taking an action a_t over sampling the action from the policy $\pi(\cdot|s_t)$. A positive advantage $A^{\pi}(s_t, a_t) > 0$ indicates that taking an action a_t while being in state s_t is better than the expected action. A negative advantage $A^{\pi}(s_t, a_t) < 0$ indicates that taking an action a_t while being in state s_t is worse than the expected action. Hence, the advantage function guides the learning process to encourage actions that lead to improved overall performance.

$$A^{\pi}(s_t, a_t) = Q^{\pi}(s_t, a_t) - V^{\pi}(s_t). \tag{5.11}$$

5.3 Partial Observability of the Environment States

In an ideal case, the states of a dynamic system are all measurable during interactions with the environment where the control problem can be formulated as an MDP. In reality, however, not all states are available to the agent, e.g., wall internal temperatures which characterize building thermal dynamics, and only one or a few of the states are observable, e.g., zone temperatures. Control of such systems can be addressed as a partially observable Markov decision process (POMDP) [34] or as a regular MDP by including historical measurements of the observable outputs and inputs in the state vector. The latter is analogous to the state-space realization of a high-order linear system using the high-order time derivatives of the inputs and outputs as the state variables. This MDP formulation has been used for RL implementations for building HVAC controls.[19].

5.4 Policy Gradient Methods

Value-based methods such as deep Q-learning showed remarkable performance in solving popular games such as Atari [35] and has also been studied for HVAC systems controls. These methods learn the value function and derive a greedy policy that chooses the action with the highest action value function. In recent years, policy gradient methods have shown the capacity for solving complex problems such as robot controls [36]. These methods directly learn the policy instead of using the action value function which allows them to have better performance than value-based methods. For a discrete action space, the policy outputs the probability of taking each action given a state.

$$\pi_\theta(s_t, a_t) = P(a_t|s_t). \tag{5.12}$$

Then policy gradient methods seek to maximize the following cost function:

$$\max_\theta \ J(\pi_\theta) = \ \mathbb{E}_{\tau\sim\pi_\theta}\Big[R(\tau)\Big], \tag{5.13}$$

which is solved by gradient descent:

$$\theta_{k+1} = \theta_k + \alpha\nabla_\theta J(\pi_{\theta_k}), \tag{5.14}$$

where α is the learning rate and k is the update step. $\nabla_{\theta_k} J(\theta_k)$ has been shown to in the following form:

$$\nabla_\theta J(\pi_{\theta_k}) = \mathbb{E}_{\tau\sim\pi_{\theta_k}}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)|_{\theta_k}\Phi_t\right], \tag{5.15}$$

where Φ_t can be chosen between the return $R(\tau)$, the action value function $Q^\pi(s_t, a_t)$, the reward-to-go defined as $\hat{R}_t = \sum_{t'=t}^{T} r_{t'}$, or the advantage $A^\pi(s_t, a_t)$.

Expectations are estimated by a sample mean over trajectories and time steps in this study. If the agent collects N trajectories of T steps each, the expected value of the objective function in Equation 5.19 can be estimated as follows:

$$\frac{1}{NT}\sum_{N}\sum_{t=0}^{T}\nabla_\theta \log \pi_\theta(a_t|s_t)|_{\theta_k}\Phi_t. \tag{5.16}$$

5.4.1 Generalized Advantage Estimation

The large number of samples required and the unstability of the improvement become a challenge for policy gradient methods. [37] identifies a correct estimation of the advantages as the key to reducing variance and proposes to use general advantage estimation (GAE), a method to reduce the variance of the estimated policy gradient at the cost of some bias. The advantages are estimated using GAE as follows:

$$\hat{A}_t^{\pi,GAE} = \sum_{l=0}^{T}(\gamma\lambda_{GAE})^l \delta_{t+l}^{V^\pi}, \tag{5.17}$$

$$\delta_{t+l}^{V^\pi} = r_t + \gamma V^\pi(s_{t+1}) - V^\pi(s_t), \tag{5.18}$$

where $\delta_{t+l}^{V^\pi}$ is the discounted TD residual which represents the difference between predicted and actual value function $V^\pi(s_t)$. The variance and bias trade-off is controlled by two parameters λ_{GAE} and γ. Most of the studies that implement a policy optimization algorithm used GAE to estimate the advantages. However, the ideal λ_{GAE} value for different applications may differ. This thesis found that a value of $\lambda_{GAE} = 0.52$ would provide better estimation of the advantages for building control applications.

5.4.2 Vanilla Policy Gradient

One of the simplest policy optimization algorithms is vanilla policy gradient (VPG). The This algorithm takes m trajectories with the current policy and uses them to estimate the advantages and policy gradient. Then it uses gradient descent steps to update the parameterized policy and updates the value function parameters at the end of each step using the mean squared error (MSE) [38] as described in Algorithm 1. While this method is straightforward to implement, it suffers from a proclivity to fall in local optima and has poor sample efficiency and high variance. This thesis does not test this algorithm but it is discussed in this section as the foundation of the RL algorithms used in this study.

Algorithm 1 RL - Vanilla Policy Gradient

```
1: Initialize policy and value function networks weights $\theta_0$ and $\phi_0$.
2: Initialize learning rates $\beta_1, \beta_2$.
3: for k = 0, 1, 2, 3, ... do
4:     Collect set of trajectories $D_m = \{\tau_i\}$ with current policy $\pi_{\theta_k}$
5:     Compute advantages $\hat{A}^{\pi_{\theta_k}}$
6:     Estimate the policy gradient as
           $\hat{g}_k = \frac{1}{|D_m|} \sum_{\tau \in D_m} \sum_{t=0}^{T} \nabla_\theta log \pi_\theta(a_t|s_t)|_{\theta_k} \hat{A}^{\pi_{\theta_k}}(s_t, a_t)$
7:     Perform an update on the policy network using gradient descent
           $\theta_{k+1} = \theta_k + \beta_1 \hat{g}_k$
8:     Fit value function using regression and MSE using learning rate $\beta_2$ for each iteration:
           $\phi_{k+1} = argmin_\phi \frac{1}{|D_m|T} \sum_{\tau \in D_m} \sum_{t=0}^{T} (V_{\phi_k}^{\pi_{\theta_k}}(s_t) - \hat{R}_t)^2$
9: end for
```

5.4.3 Trust Region Policy Optimization (TRPO)

The goal of RL is to obtain a policy π that maximizes the finite or infinite horizon discounted total return $J(\pi_\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[R(\tau)]$. To overcome the proclivity to falling in local optimas of VPG algorithms, TRPO [39] seeks to maximize a surrogate objective function subject to a constraint that limits the size of a policy update during each iteration measured by the KL divergence. This measure quantifies the difference between two policies. This sets an upper bound on how much a policy can change at every update step.

$$\max_\theta \quad \mathbb{E}_{\tau \sim \pi_{\theta_k}} \left[\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} \hat{A}^{\pi_{\theta_k}}(s, a) \right] \tag{5.19}$$

$$\text{s.t.} \quad \mathbb{E}_{\tau \sim \pi_\theta} \left[D_{KL}(\pi_\theta(\cdot|s) || \pi_{\theta_k}(\cdot|s)) \right] \leq \delta, \tag{5.20}$$

where $\hat{A}^{\pi_{\theta_k}}$ is the advantage that can be calculated using the GAE method [37] and δ is the upper bound imposed on the KL divergence between two consecutive policy updates. This problem is usually simplified using a linear approximation of the objective function and a quadratic approximation of the KL divergence constraint. TRPO approximates this by taking the 1st and 2nd order Taylor expansions of the objective function and constraint respectively as follows:

$$L(\theta_k, \theta_{k+1}) \approx g^T(\theta_{k+1} - \theta_k), \tag{5.21}$$

$$\overline{D}_{KL}(\theta_{k+1}||\theta_k) \approx \frac{1}{2}(\theta_{k+1} - \theta_k)^T H(\theta_{k+1} - \theta_k), \tag{5.22}$$

where $L(\theta_k, \theta_{k+1})$, is the approximation of the optimization goal in Equation 5.19, g is the estimated objective function gradient and H is the estimated hessian of the KL divergence or Fisher information matrix and δ is a constant that limits the KL divergence between new and old policies. Then the approximate optimization problem is:

$$\max_{\theta} \quad g^T(\theta - \theta_k) \tag{5.23}$$

$$\text{s.t.} \quad \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \leq \delta. \tag{5.24}$$

Analytically solving this problem gives:

$$\theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{g^T H^{-1} g}} H^{-1} g. \tag{5.25}$$

TRPO adds a backtracking line search to this update:

$$\theta_{k+1} = \theta_k + \alpha^j \sqrt{\frac{2\delta}{g^T H^{-1} g}} H^{-1} g, \tag{5.26}$$

where α is the backtracking coefficient and j is the smallest nonnegative integer such that the new policy satisfies the constraint and produces a positive advantage. The inverse of matrix

H is hard to compute for high dimensional networks so TRPO uses the conjugate gradient method to solve

$$Hx = g. \tag{5.27}$$

Then x can be used to reformulate the update:

$$\theta_{k+1} = \theta_k + \alpha^j \sqrt{\frac{2\delta}{x_k^T H_k x_k}} x_k. \tag{5.28}$$

TRPO starts collecting trajectories under the current policy π_{θ_k}. With these trajectories, it estimates the rewards-to-go, advantages and value function for the current state. With these estimations, it computes the gradient and uses Equation 5.21 to Equation 5.28 to calculate the policy parameters update. Finally, it fits the value function parameters using the value function network and the rewards-to-go estimated values as described in Algorithm 2.

While TRPO overcomes some of the challenges of VPG algorithms, there are also some drawbacks with TRPO. First, it is computationally expensive since it requires solving a system involving the Fisher information matrix. While this can be improved by using approximations or iterative methods, errors may be introduce as consequence of these procedures.

5.4.4 Proximal Policy Optimization (PPO)

PPO seeks to solve the same problem presented in the TRPO case in a simpler manner. Two versions of PPO algorithms were proposed in the original work [40], while PPO-clip is more broadly used. PPO-clip uses a clip function to limit the incentive of policy change to stabilize training as follows:

$$\max \ \mathbb{E}_{\tau \sim \pi_{\theta_k}} L(s, a, \theta_k, \theta), \tag{5.29}$$

Algorithm 2 RL - Trust Region Policy Optimization

```
1:  Initialize policy and value function networks weights $\theta_0$ and $\phi_0$.
2:  Initialize $\alpha$, $\beta$, $\delta$, $K$
3:  for k = 0, 1, 2, 3, ... do
4:      Collect set of trajectories $D_m = \{\tau_i\}$ with current policy $\pi_{\theta_k}$
5:      Compute rewards-to-go $\hat{R}_t$
6:      Compute Advantage estimates $\hat{A}_t^{\pi_{\theta_k}}$
7:      Estimate the policy gradient as
            $\hat{g}_k = \frac{1}{|D_m|} \sum_{\tau \in D_m} \sum_{t=0}^{T} \nabla_\theta log\pi_\theta(a_t|s_t)|_{\theta_k} \hat{A}_t^{\pi_{\theta_k}}$
8:      Use the conjugatte gradient algorithm to compute:
            $\hat{x} \approx \hat{H}_k^{-1} \hat{g}_k$
9:      Update the policy by backtracking line search with:
            $\theta_{k+1} = \theta_k + \alpha^j \sqrt{\frac{2\delta}{\hat{x}_k^T \hat{H}_k \hat{x}_k}} \hat{x}_k$
10:     Fit value function using regression and MSE using learning rate $\beta$ for each iteration:
            $\phi_{k+1} = argmin_\phi \frac{1}{|D_m|T} \sum_{\tau \in D_m} \sum_{t=0}^{T} (V_{\phi_k}^{\pi_{\theta_k}}(s_t) - \hat{R}_t)^2$
11: end for
```

where

$$L(s, a, \theta_k, \theta) = \min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} \hat{A}^{\pi_{\theta_k}}(s, a), h\Big(\epsilon, \hat{A}^{\pi_{\theta_k}}(s, a)\Big)\right), \tag{5.30}$$

and the clip-range ϵ is a hyper-parameter that determines how far the new policy candidate is allowed to deviate from the old one and the function h is defined as:

$$h\Big(\epsilon, \hat{A}^{\pi_{\theta_k}}\Big) = \begin{cases} (1+\epsilon)\hat{A}^{\pi_{\theta_k}}, & \hat{A}^{\pi_{\theta_k}} > 0 \\ (1-\epsilon)\hat{A}^{\pi_{\theta_k}}, & \hat{A}^{\pi_{\theta_k}} < 0. \end{cases} \tag{5.31}$$

PPO starts in a similar manner as TRPO, collecting trajectories under the current policy and estimating the rewards-to-go, advantages and state value function. Note that both TRPO and PPO can use a parameterized network to approximate the value function $V_\beta(s)$ and compute advantages at each timestep. Next, it uses Equation 5.29 to calculate the policy network parameters update and lastly it fits the value function similarly with TRPO as described in Algorithm 3. PPO is an effective policy gradient method and avoids some of the drawbacks of TRPO since it does not rely on approximations or computationally expensive expressions. However, the performance is sensitive to the choice of the clip-range hyper-parameter ϵ. An

Algorithm 3 RL - Proximal Policy Optimization

```
1: Initialize policy and value function networks weights $\theta_0$ and $\phi_0$.
2: Initialize $\epsilon$, $\beta$, $\delta$, $K$
3: for k = 0, 1, 2, 3, ... do
4:     Collect set of trajectories $D_m = \{\tau_i\}$ with current policy $\pi_{\theta_k}$
5:     Compute rewards-to-go $\hat{R}_t$
6:     Compute Advantage estimates $\hat{A}_t^{\pi_{\theta_k}}$
7:     Update the parameters using the PPO loss function
           $\theta_{k+1} = \arg\max_{\theta} \frac{1}{|D_m|T} \sum_{\tau \in D_m} \sum_{t=0}^{T} L(s_t, a_t, \theta_k, \theta)$
8:     Fit value function using regression and MSE using learning rate $\beta$ for each iteration:
           $\phi_{k+1} = argmin_{\phi} \frac{1}{|D_m|T} \sum_{\tau \in D_m} \sum_{t=0}^{T} (V_{\phi_k}^{\pi_{\theta_k}}(s_t) - \hat{R}_t)^2$
9: end for
```

improper setting of this hyper-parameter can result in unstable training and the policy can get stuck in local optima quickly for high clip ranges.

5.5 Constrained Markov Decision Process (CMDP)

A CMDP is an extension of a MDP with constraints added to the formulation. These constraints limit the set of feasible policies for the MDP. A CMDP is defined as a set of states, actions, transition probabilities and cost functions. The cost function is the only additional element compared to unconstrained MDP, which represents a penalty for constraint violations.

5.5.1 Cost

The cost for a time step c_t quantifies the violations of the constraints that the system must satisfy. In the context of the building HVAC system it is defined in the same way as the comfort reward r_{com} for the unconstrained scenario:

$$c_t = -\phi\big(\max(T_{z,min}^t - T_z^t, 0) + \max(T_z^t - T_{z,max}^t, 0)\big). \tag{5.32}$$

5.6 Constrained Reinforcement Learning Key Concepts

CMDP expands MDP formulation and uses additional key concepts related to the constraint violations.

5.6.1 Cost Return

The cost return of a trajectory is defined as the discounted or undiscounted cumulative constraint violation for the episode.

$$C(\tau) = \sum_{t=0}^{T} \gamma^t c_t. \tag{5.33}$$

5.6.2 Cost Value Function

The cost value function of a state is defined as the expected discounted or undiscounted cumulative cost collected during an episode τ following a policy π given a state s_t.

$$V_C^{\pi}(s_t) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{l=0}^{T} \gamma^k c_{t+l+1} | s_t\right]. \tag{5.34}$$

5.6.3 Cost Action Value Function

The cost action value function of a state is defined as the expected discounted or undiscounted cumulative cost collected during an episode τ following a policy π given a state s_t and action a_t.

$$Q_C^{\pi}(s_t, a_t) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{l=0}^{T} \gamma^k c_{t+l+1} | s_t, a_t\right]. \tag{5.35}$$

These functions are also parameterized as:

$$V_C^\pi(s_t) \approx V_{C,\omega}^\pi(s_t), \tag{5.36}$$

$$Q_C^\pi(s_t, a_t) \approx Q_{C,\omega_q}^\pi(s_t, a_t), \tag{5.37}$$

where ω and ω_q are the parameters of the value function and action value function respectively.

5.6.4 Cost Advantage Function

The cost advantage function is defined in a similar way as the reward advantage function:

$$A_C^\pi(s_t, a_t) = Q_C^\pi(s_t, a_t) - V_C^\pi(s_t). \tag{5.38}$$

5.7 Constrained Policy Optimization

An inherent challenge of non-constrained RL algorithms (e.g., TPRO and PPO) is the inability to address constraints, while for most control applications, operational constraints exist to ensure safe and quality services. A remedy is to incorporate a penalty for possible constraint violations in the reward function but it is difficult to find an appropriate weighting factor between the original merit and constraint satisfaction. While setting a high penalty factor helps better enforce constraints, it limits the improvement in the original objective. On the other hand, setting a low penalty factor prioritizes maximization of the original merit at the expense of significant constraint violations.

CPO seeks to maximize the original merit function but restricts the set of feasible policies so that a discounted constraint violation return $J_C(\pi) = \mathbb{E}_{\tau\sim\pi}[C(\tau)]$ is limited by a set upper bound d [41]. Among the proposed algorithms in the literature, constrained policy optimization (CPO) [42] has been widely studied, which seeks to solve the optimization problem:

$$\max_{\theta} \quad \mathbb{E}_{\tau\sim\pi_{\theta_k}}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}\hat{A}^{\pi_{\theta_k}}(s,a)\right] \tag{5.39}$$

$$\text{s.t.} \quad J_C(\pi_{\theta_k}) + \frac{1}{1-\gamma}\mathbb{E}_{\tau\sim\pi_{\theta_k}}\left[\hat{A}_C^{\pi_{\theta_k}}(s,a)\right] \leq d \tag{5.40}$$

$$\mathbb{E}_{\tau\sim\pi_\theta}\left[D_{KL}(\pi_{\theta_k}||\pi_{\theta_k})\right] \leq \delta. \tag{5.41}$$

The constraint advantage and return are defined in a similar manner to those of the reward function. A positive slack variable d is introduced to stabilize training. CPO uses linear approximations on the objective and the first constraint and a second-order approximation on the KL divergence constraint to solve the optimization problem.

$$\max_{\theta} \quad g^T(\theta - \theta_k) \tag{5.42}$$

$$\text{s.t.} \quad c + b^T(\theta-\theta_k) \leq 0 \tag{5.43}$$

$$\frac{1}{2}(\theta-\theta_k)^T H(\theta-\theta_k) \leq \delta, \tag{5.44}$$

where b is the gradient of the constraint values estimated in a similar way as g but using the constraint advantages $\hat{A}_C^{\pi_{\theta_k}}$ and c is the difference between the total constraint violation and its the upper limit ($c = J_C(\pi_{\theta_k}) - d$). This problem can be solved analytically to avoid the use of solvers and reduce computational time. The analitycal solutions is as follows:

$$\theta_{k+1} = \theta_k + \frac{1}{\lambda}H^{-1}(g - b\nu), \tag{5.45}$$

where λ and ν are the dual variables of the optimization problem whose analytical solution can be found in [42]. CPO starts collecting trajectories using the currrent policy. Then it estimates the gradients for the reward and constraint functions as well as the total costs $J_C(\pi_k)$. CPO recognizes two scenarios. When the problem is feasible it solves the constrained optimization problem shown in Equation 5.42 to Equation 5.44. Due to approximation errors, the approximate solution may not be feasible for the original problem. In this scenario, a different optimization problem that only reduces the constraint violation value is used to obtain the update step. The analytical solution of this new problem is:

$$\theta_{k+1} = \theta_k - \sqrt{\frac{2\delta}{b^T H^{-1} b}} H^{-1} b. \tag{5.46}$$

Lastly it performs backtracking line search to ensure the new parameters satisfy the constraints and result in a positive policy improvement. CPO uses two parameterized value networks, for the constraint and reward value functions, respectively. As a final step it fits the reward and constraint value functions $V_{\phi}^{\pi_k}$ and $V_{C,\omega}^{\pi_k}$ using the estimations collected during the update step as described in Algorithm 4.

Algorithm 4 RL-Constrained Policy Optimization

```
 1: Initialize policy and value functions networks weights $\theta_0$, $\phi_0$, $\omega_0$.
 2: Initialize $\gamma$, $d$, $\delta$, $\beta$.
 3: for k = 0,1,2,3,... do
 4:     Collect set of trajectories $D_m = \{\tau_i\}$ with current policy $\pi_{\theta_k}$.
 5:     Compute rewards-to-go $\hat{R}_t$ and costs to go $\hat{C}_t$.
 6:     Compute Advantage estimates $\hat{A}^{\pi_{\theta_k}}$ and $\hat{A}_C^{\pi_{\theta_k}}$
 7:     if Optimization problem is feasible then
 8:         Update parameteres using Equation 5.45.
 9:     else
10:         Update parameters to only reduce constraint violation using Equation 5.46.
11:     end if
12:     Fit value function using regression and MSE using learning rate $\beta$:
            $\phi_{k+1} = argmin_{\phi} \frac{1}{|D_m|T} \sum_{\tau \in D_m} \sum_{t=0}^{T} (V_{\phi_k}^{\pi_{\theta_k}}(s_t) - \hat{R}_t)^2$
            $\omega_{k+1} = argmin_{\omega} \frac{1}{|D_m|T} \sum_{\tau \in D_m} \sum_{t=0}^{T} (V_{C,\omega_k}^{\omega_{\theta_k}}(s_t) - \hat{C}_t)^2$
13: end for
```

CHAPTER 6

CASE STUDY RESULTS

Simulation tests were conducted to assess the performance of the constrained RL algorithm in comparison with the different baseline strategies. This chapter first introduces the characteristics of the case study building and the operation requirements which include the comfort bounds and demand response regime. Secondly, it shows the results obtained with the unconstrained RL algorithms followed by the results obtained by the constrained RL algorithm. A parametric study of the penalty factor and the constraint violation upper bound is also presented to illustrate the sensitivity of the algorithms to these parameters.

6.1 Case Study Building Description

The case study building is a single-zone office space with a total floor area of approximately 100 m^2. The considered zone is part of a larger building with four identical office spaces all serving graduate students. Each zone is equipped with a dedicated HVAC system. Cross-zone thermal coupling is negligible due to the high-grade insulation used in the separating walls. The office space is attached to a south-facing double façade with a gap of 1 m.

The TOU retail energy rates used in the simulations were obtained from El Paso Electric Co. [43] and are shown in Table 6.1. A lower energy rate of \$0.07/kWh is involved during non-peak hours while electricity is charged at a much higher rate of \$0.22/kWh during on-peak hours. The zone temperature bounds change with the occupancy of the building, with 9AM to 6PM being the occupied period. During occupied hours the upper and lower temperature bounds are 21.5°C and 23.5°C, while during unoccupied hours are 20.5°C and

24.5°C, respectively.

Table 6.1: Summer time of use tariff

Electricity price ($/kWh)	**Hours**
0.222	12:00 to 18:00
0.077	Rest of day

6.2 MPC Baselines Results

The MPC baselines used a look-ahead horizon of 24 hours with a decision implemented every 15-min time step. The MPC baselines were formulated using the CVX package in MATLAB [44] and solved using Gurobi [45].

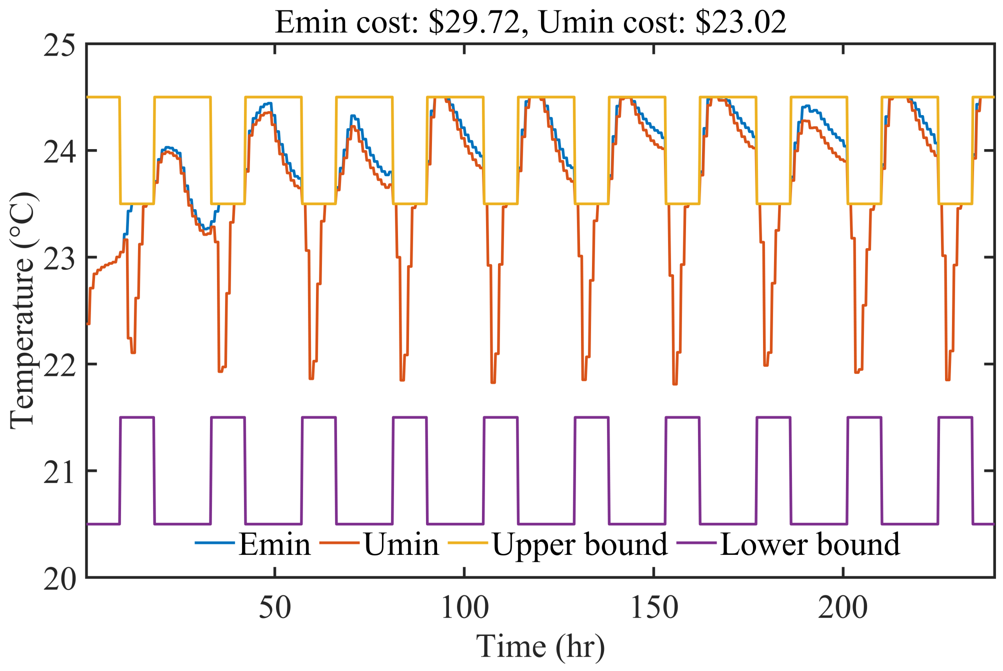

Figure 6.1: Zone temperature under MPC baselines.

Figure 6.1 and Figure 6.2 present the simulation results of the two MPC baseline strategies. The energy minimization baseline (E-min) maintains the zone temperature at the upper

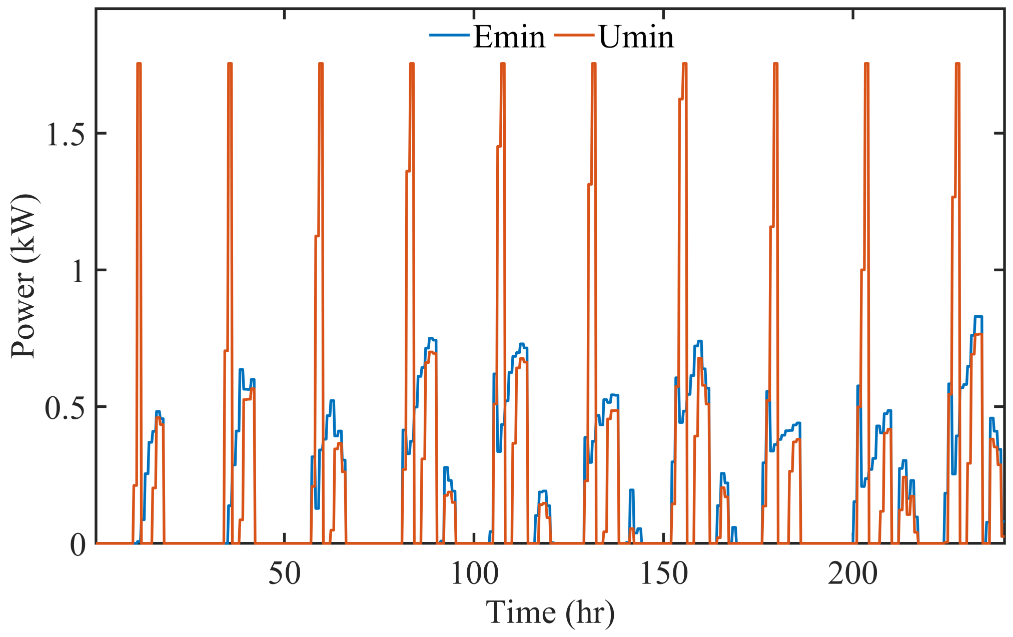

Figure 6.2: Power profile under MPC baselines.

bound when mechanical cooling is called for, resulting in minimum energy usage. During unoccupied hours the temperature floats and the HVAC system remains off. The utility-cost-minimizing (U-min) strategy engages a pre-cooling action before each on-peak period. The pre-cooling action maintains a lower zone temperature prior to on-peak hours so that "cooling" energy is stored in the building thermal mass; during on-peak hours, the zone temperature is adjusted upwards to allow the stored cooling energy to be released, resulting in shifting of building electricity use to low-cost hours to reduce utility cost. While this strategy presents utility cost savings, it increases the total energy used by the HVAC system. Table 6.5 shows the cost break down for the MPC, unconstrained RL and constrained RL strategies. Compared to strategy E-min, the cost-minimizing MPC strategy achieves cost savings of 22.5%, with a total energy rebound of 4.6%. This represents the best economic performance that can possibly be obtained under the same prediction horizon setting without any comfort constraint violation. Actual MPC performance would be worse due to potential control-plant model mismatches.

6.3 Unconstrained RL Baseline Results

The RL strategies were trained using two years of simulation data following an on-policy scheme. The RL baseline simulations were conducted using the Stable Baselines 3 OpenAI library [46].

6.3.1 Simulation Setup

Episodic training was utilized with each episode or trajectory consisting of 1 days (96 steps). TRPO and PPO with different constraint violation penalty factors, i.e. , ψ = 0.01, ψ = 0.1, ψ = 1 and ψ = 10, were chosen as RL baselines to illustrate the effect of the weighting factor on the control performance.

Policy Network: Policy networks use the observation set as input, 2 hidden layers of 64 neurons and an output layer with 17 possible action logits. After each layer a hyperbolic tangent ($\tanh$) activation function is used. A final softmax layer is used to predict the probability of choosing an action given an observation set.

Value Function Network: Value function networks use the observation set as input, 2 hidden layers of 64 neurons, and output the of the estimated value function. After each layer, a hyperbolic tangent ($\tanh$) activation function is used.

Hyperparameters: The hyperparameters used in the simulation of TRPO and PPO algorithms are shown in Table 6.2 and Table 6.3.

Table 6.2: TRPO baseline hyperparameters

Hyperparameter	**Value**
δ	0.003
λ_{GAE}	0.52
α	0.8
β	0.001
K	25
γ	1
T	96
$\lvert D_m\rvert$	1

Table 6.3: PPO baseline hyperparameters

Hyperparameter	**Value**
ϵ	0.003
λ_{GAE}	0.52
β	0.001
γ	1
T	96
$\lvert D_m\rvert$	1

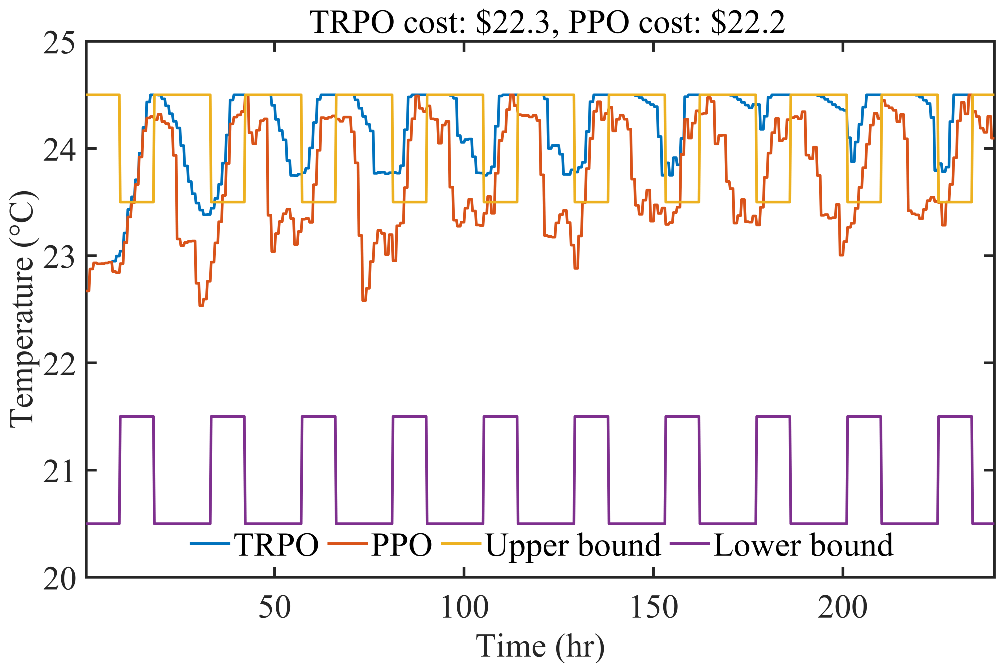

Figure 6.3: Zone temperature for unconstrained RL strategies with ψ = 0.01.

6.3.2 Simulation Results

Figure 6.3 to Figure 6.6 present the simulation test results under the two unconstrained RL strategies subject to different constraint violation penalty factors. As expected, the unconstrained RL strategy with the lowest comfort penalty factors leads to the lowest energy cost, with cost savings up to 26.2% relative to E-min, but at the expense of significant comfort issues. The RL strategies with high comfort penalty factors are able to regulate the zone temperature within the comfort zone but lead to high HVAC energy costs (savings of only less than 3% relative to E-min). The high-comfort-penalty RL strategies execute very mild pre-cooling actions for load shifting for most cases, which is the major cause of the high energy cost as can be seen in Table 6.5. It is evident from these results that performance achievable with the unconstrained RL strategies is highly dependent on the comfort penalty factor setting. This also makes the comparison with the MPC benchmarks challenging.

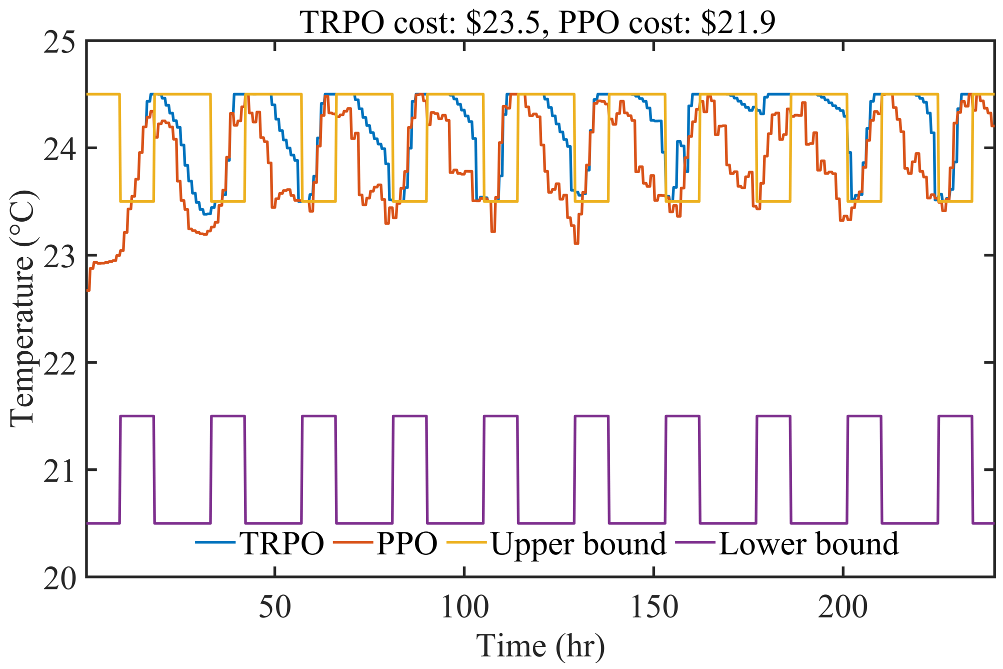

Figure 6.4: Zone temperature for unconstrained RL strategies with $\psi = 0.1$.

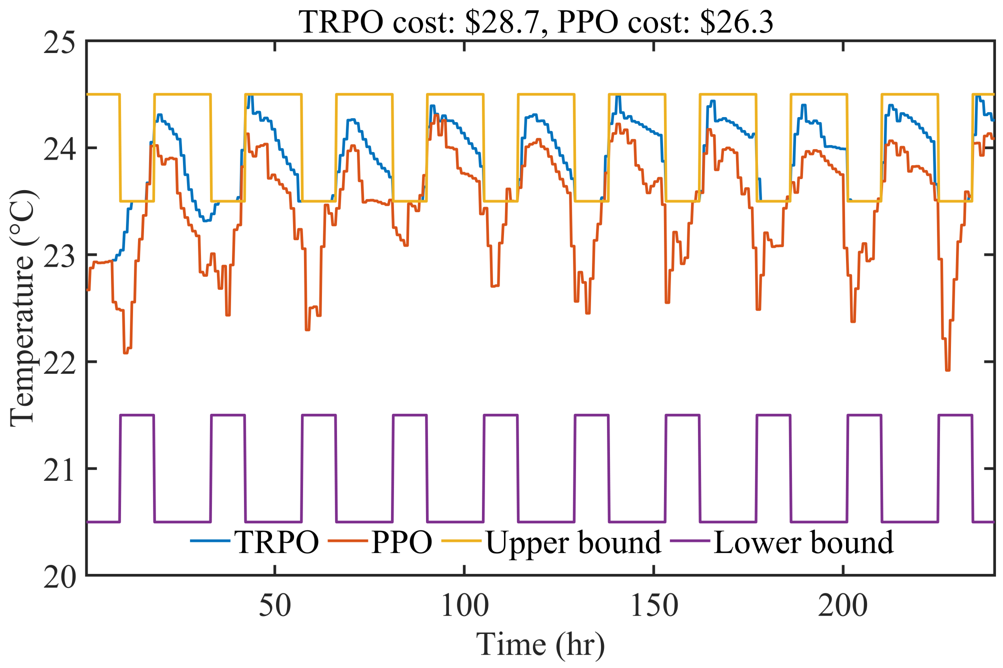

Figure 6.5: Zone temperature for unconstrained RL strategies with $\psi = 1$.

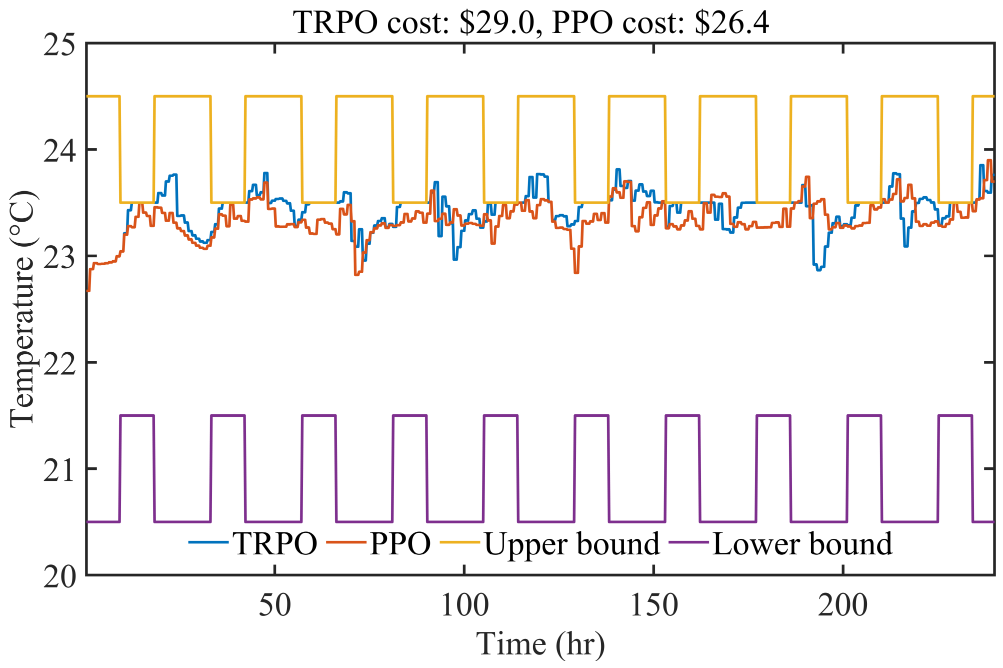

Figure 6.6: Zone temperature for unconstrained RL strategies with ψ = 10.

6.3.3 Parametric Study of Penalty Factor

The simulations performed in the previous subsection consider penalty factors that are one order of magnitude away from each other. Hence, a high-granularity analysis of the effect of the penalty factor on performance is needed. Table 6.5 shows that better balances between cost and constraint violations may be found with penalties between 0.1 and 1. For this purpose, ten penalty factors between $\psi = 0.1$ and $\psi = 1$ are simulated and the average performance over 10 random seeds is evaluated.

Figure 6.7 shows the performance of the PPO algorithm under different penalty factors. The cost performance of CPO is shown in yellow dashed line and the comfort violation performance is shown in green dashed line, as benchmarks. Detailed results of the CPO algorithm performance will be covered in section 6.4. The red line shows the comfort violation average while the blue line shows the average electricity use cost for the PPO cases. It can be seen that unconstrained RL with penalty factors of $\psi \leq 0.4$ outperforms the CPO strategy in terms of cost. However, these cases have a comfort violation of at least 30 °C-step which is equivalent to 15 times the violations of CPO case. On the other hand, none of these cases could outperform the CPO control strategy in terms of constraint satisfaction. As it will be shown in next section, constrained RL with linear mapping performed better than any of the PPO cases simulated in this subsection. These results confirm that CPO-based strategies can successfully enforce constraints subject to a minor tolerance, without any need for fine-tuning of a penalty factor.

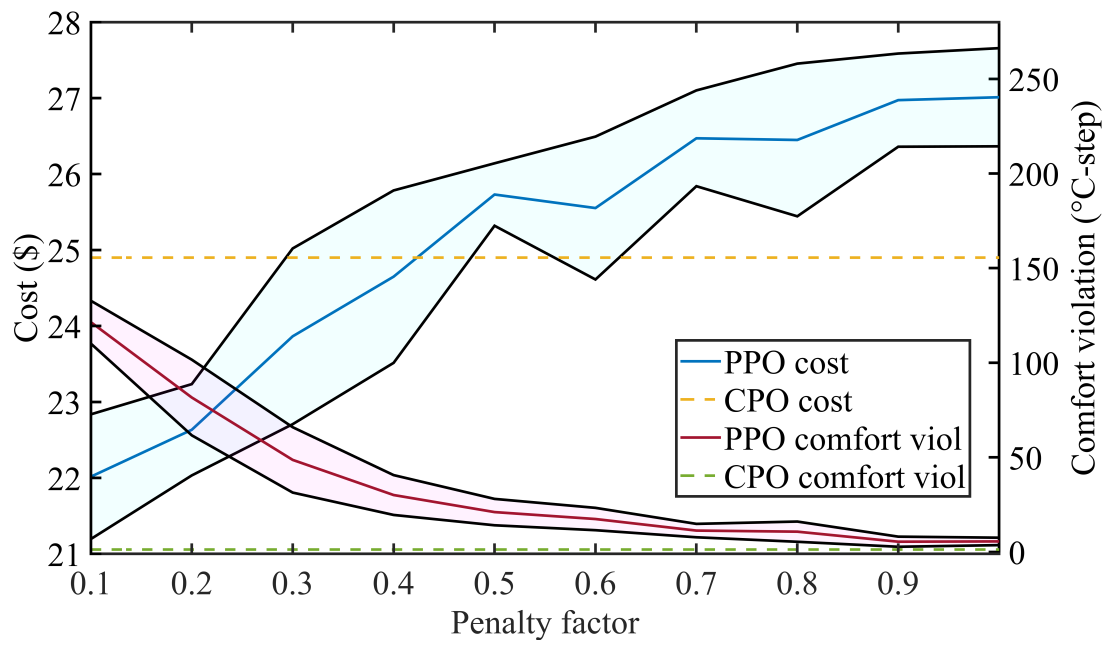

Figure 6.7: PPO performance under different penalty factors.

6.4 Constrained RL Results

The constrained RL strategy was trained using two years of simulation data following an on-policy scheme. The CPO algorithm was implemented using PyTorch [47].

6.4.1 Simulation Setup

Episodic training was utilized with each episode or trajectory consisting of 1 days (96 steps). The algorithm is trained on 10 different random seeds to assess average performance.

Policy Network: The policy network used the observation set as input, 2 hidden layers of 64 neurons and an output layer with 17 possible action logits. After each layer a hyperbolic tangent ($\tanh$) activation function was used. A final softmax layer was used to predict the probability of choosing an action given an observation set.

Value Function Networks: The value function and cost value function networks used the observation set as input, 2 hidden layers of 64 neurons and output the value of the estimated value function. After each layer, a hyperbolic tangent ($\tanh$) activation function was used.

Hyperparameters: The hyperparameters used in the simulation of the CPO algorithm are shown in Table 6.4. The value function and cost value function share the same learning rate β.

Table 6.4: CPO baseline hyperparameters

Hyperparameter	**Value**
d	1
δ	0.003
λ_{GAE}	0.52
α	0.8
β	0.001
K	25
γ	1
T	96
$\lvert D_m \rvert$	1

6.4.2 Simulation Results

Figure 6.8 shows the average learning curve for the constrained RL algorithm, while the 60-episode average constraint function value is shown in Figure 6.9. A slight decrease in performance can be seen at the end of training as well as a relatively high variance of the final performance. This is possibly caused by approximation errors in the policy optimization where the obtained approximate solution may be infeasible for the original problem. When the infeasibility issue happens, the update will solely reduce constraint violation shown in Equation 5.46 causing performance degradations. Two representative control solutions out of those associated with the 10 different seeds are discussed below, corresponding to the best- and worst-performing strategies. Figure 6.10 and Figure 6.11 shows the hourly averaged control behaviors for one of the best-performing seeds. Deep pre-cooling actions are seen under this scenario, resulting in potential savings of 20.25% and minimal constraint violations. On the other hand, a less-aggressive (worst-performing) strategy shown in Figure 6.12 and Figure 6.13 results in limited pre-cooling with potential savings of 11.13% and slightly higher constraint violation. However, the violations are still considerably lower than those seen in the unconstrained RL case. The total average cost and constraint violations are shown in Table 6.5. The potential average savings are 12.4% with an average constraint violation of 4.24 °C-step. For all seed averages, it can be seen in Figure 6.9 that the constraint violation approaches the set limit $d = 1$ and succeeds at enforcing the constraint satisfaction within the set limit. Note that the constraint violations can be further reduced by lowering the slack (d) setting, but a positive slack is needed for stable training. These results clearly demonstrate the superior performance of constrained RL over the unconstrained counterparts for predictive control of building flexible loads while ensuring indoor comfort.

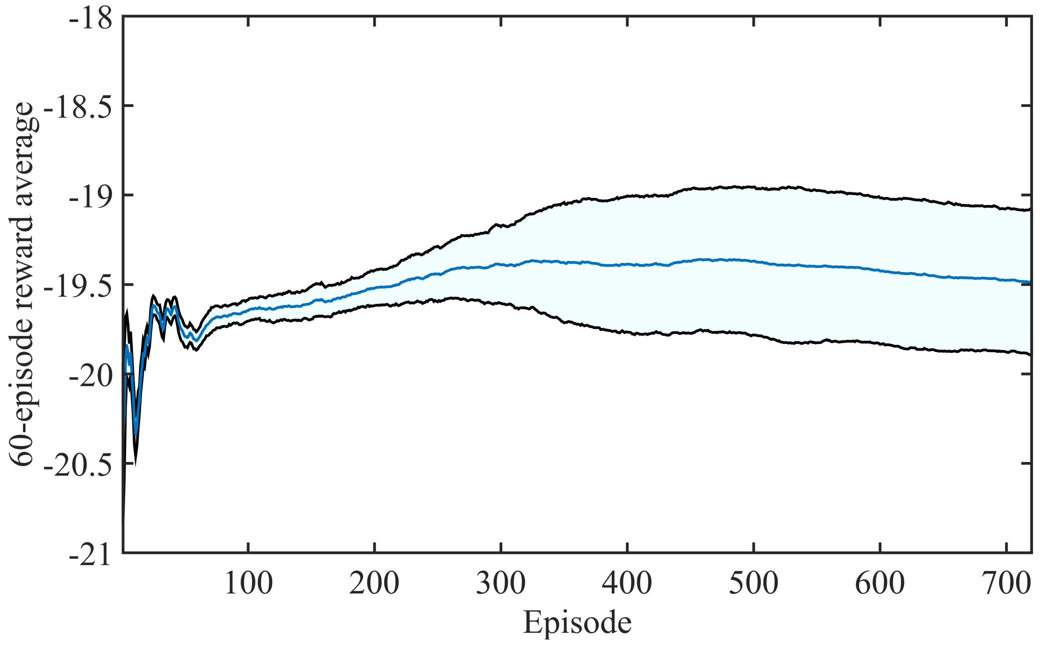

Figure 6.8: CPO learning curve for rewards.

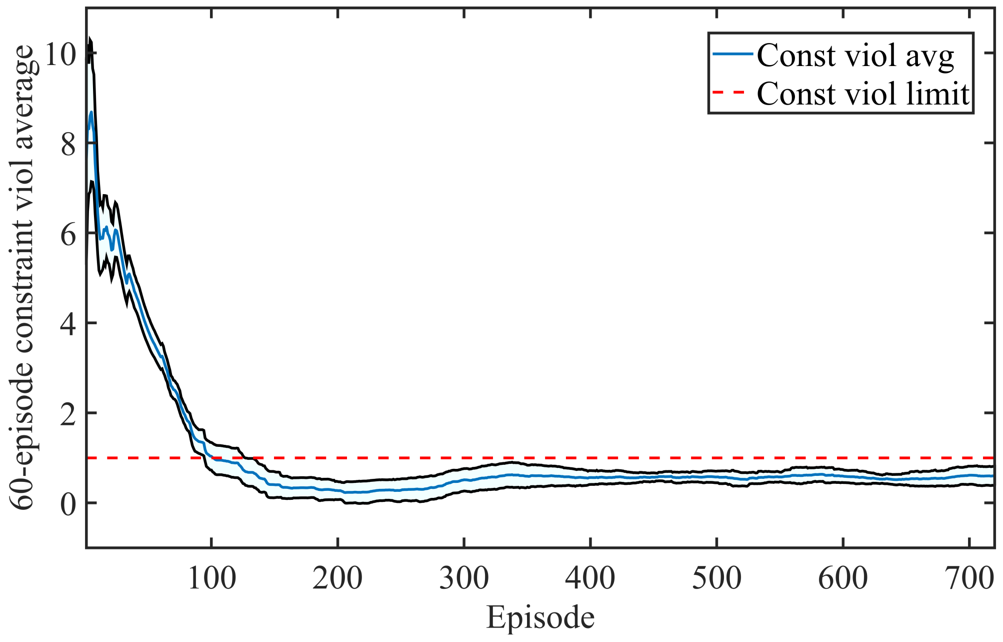

Figure 6.9: CPO learning curve for constraint violations.

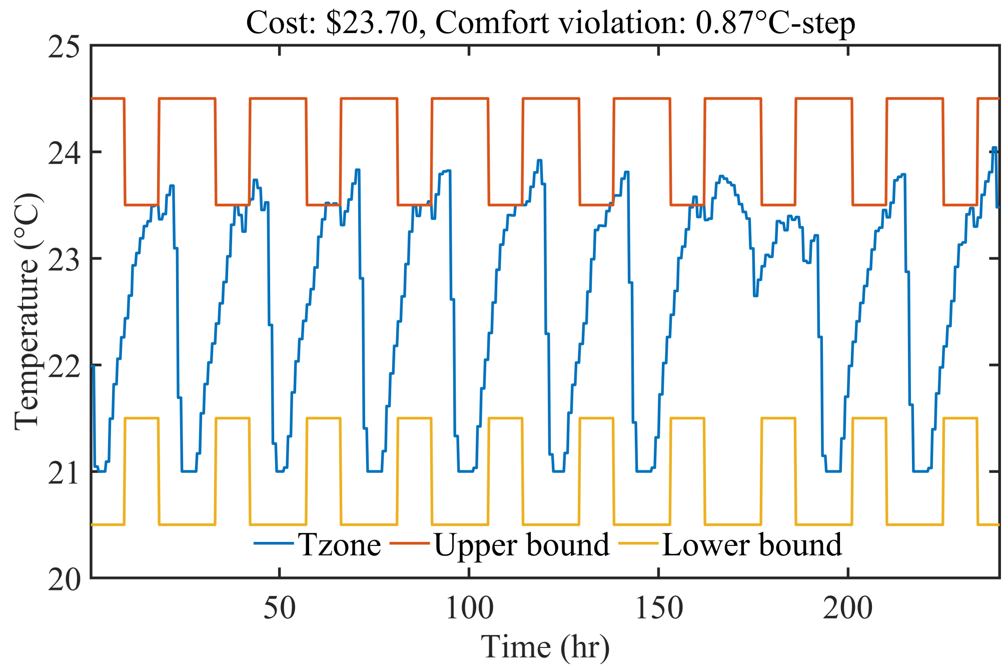

Figure 6.10: Zone temperature for best CPO control strategy.

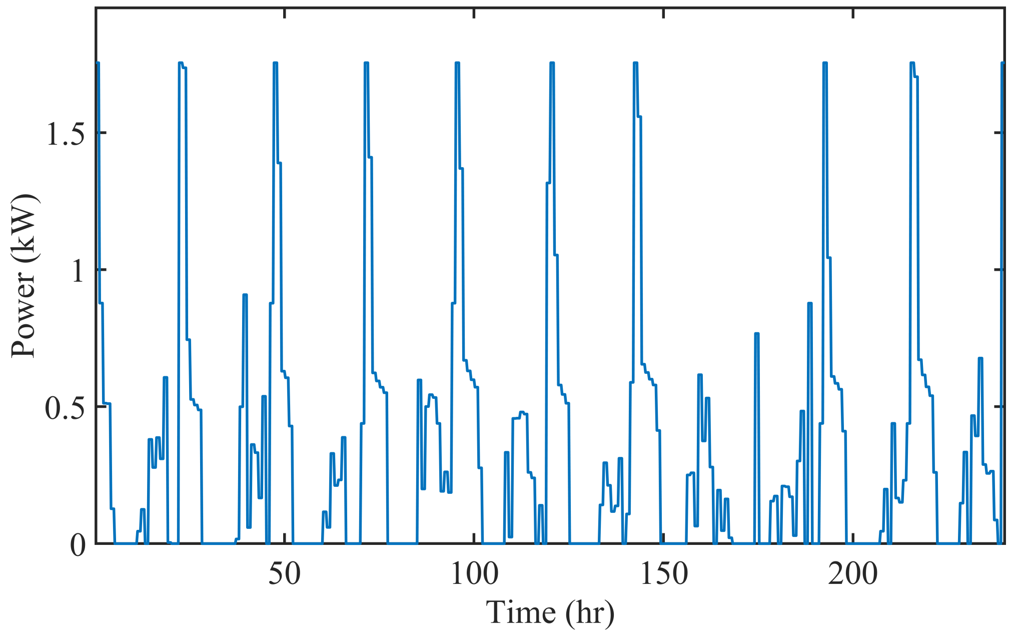

Figure 6.11: Power profile for best CPO control strategy.

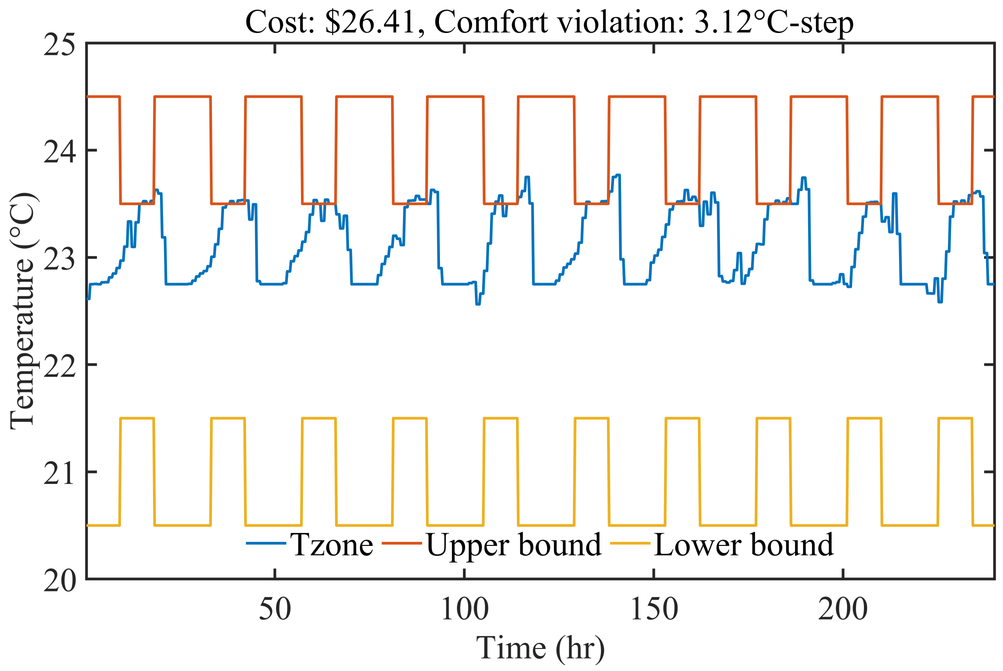

Figure 6.12: Zone temperature for worst CPO control strategy.

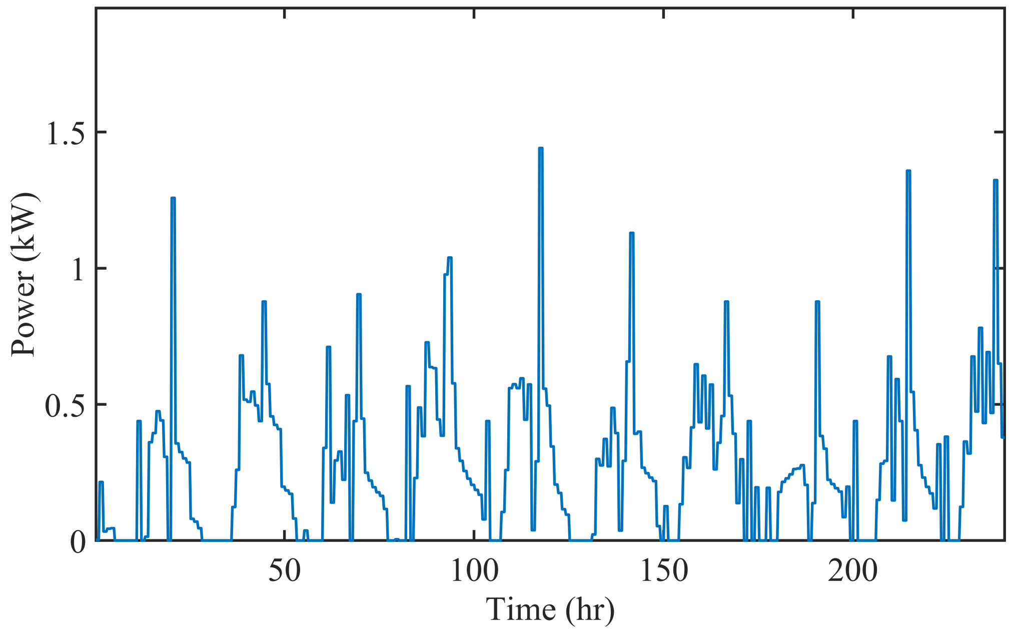

Figure 6.13: Power profile for worst CPO control strategy.

6.5 CPO with Linear Policy Mapping

While in the literature, neural networks are widely used to parameterize the policies of RL agents, some studies have found that for many control applications, complex networks may not work well ~~due to the poor data efficiency~~ and linear mappings may perform better due to the nature of the system dynamics being linear in the range of operation [48]. This section seeks to empirically test this statement and presents the simulation results for the CPO algorithm using only linear mappings. The policy, value function and cost value function are represented by linear maps, hence activation functions are not needed. The hyperparameters used for these experiments are the same as the neural network CPO shown in Table 6.4.

6.5.1 Simulation Results

Figure 6.14 shows the average learning curve for the CPO with linear mappings, while the 60-episode average constraint function value is shown in Figure 6.15. It can be seen that CPO with linear mappings yields a more stable training and a lower variance. The performance also improves monotonically as more training data is collected. Figure 6.16 and Figure 6.17 shows one of the best-performing control responses. This strategy yields potential savings of 22.7% which are slightly higher than the utility minimization MPC strategy; this is caused by minor constraint violations which have an accumulative value of 2 °C-step during the whole month. On the other hand, the worst-performing strategy shown in Figure 6.18 and Figure 6.19 achieves potential savings of 8.5% while accumulating only 0.65 °C-step of comfort violation during the month. The COP with linear mappings results in a much better average performance with average potential savings of 16.1%, which is 30% better than the neural network-based CPO average savings, and constraint violation of only 1.3 °C-step in 30 days, which is a 75% reduction compared with that of the neural network-based CPO.

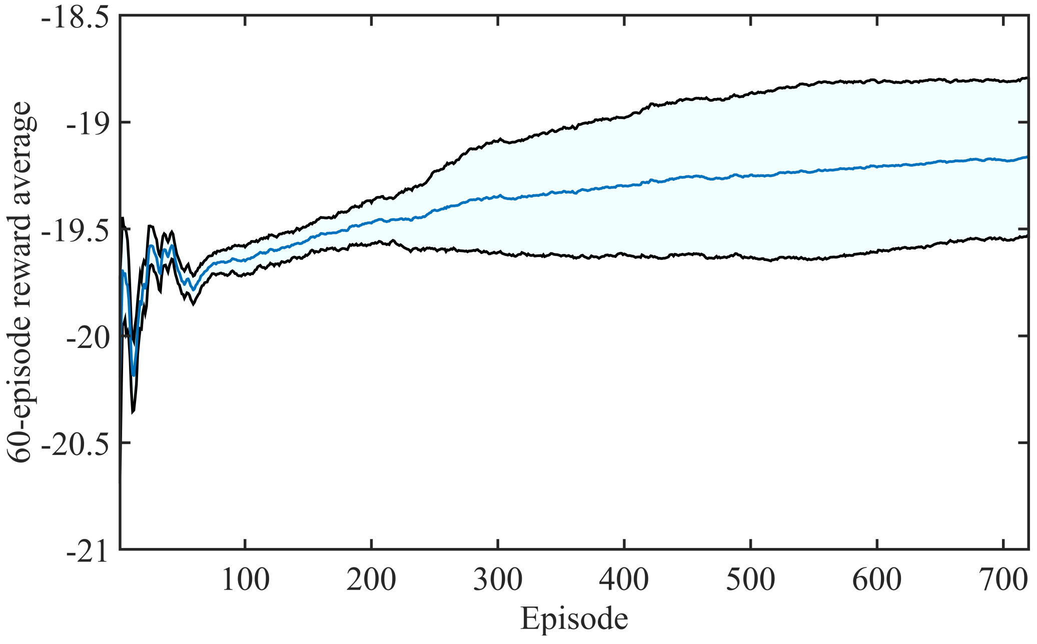

Figure 6.14: CPO algorithm learning curve with linear mapping.

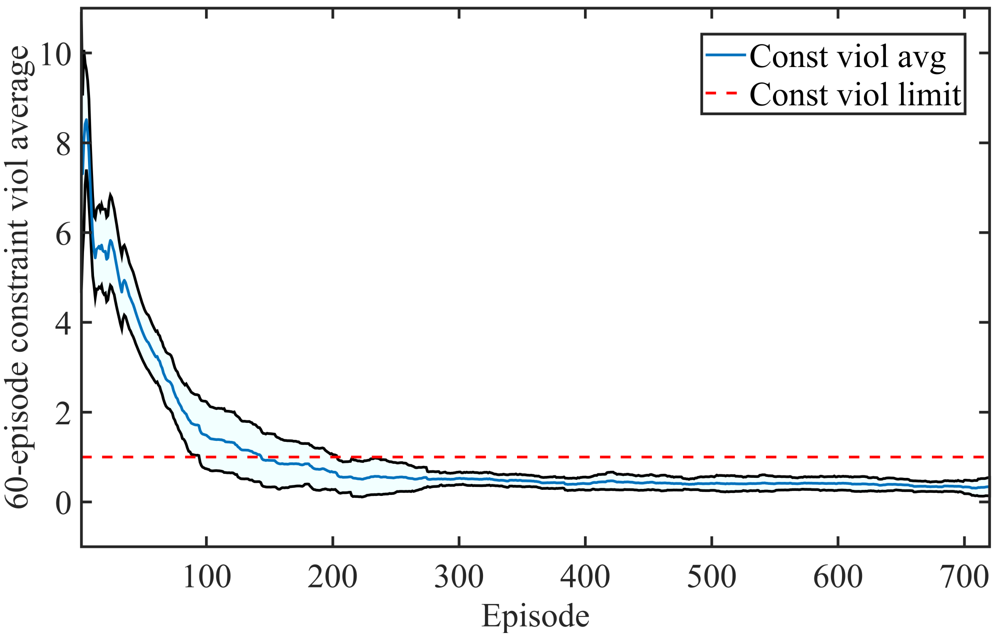

Figure 6.15: CPO algorithm constraint violation learning curve with linear mapping.

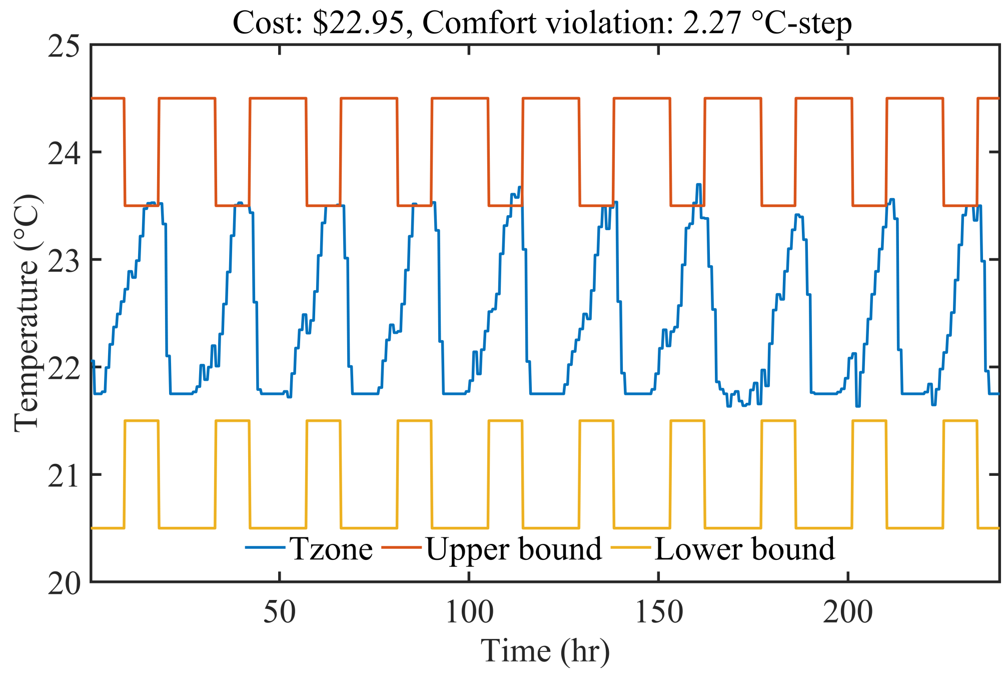

Figure 6.16: Zone temperature for best CPO control strategy with linear mapping.

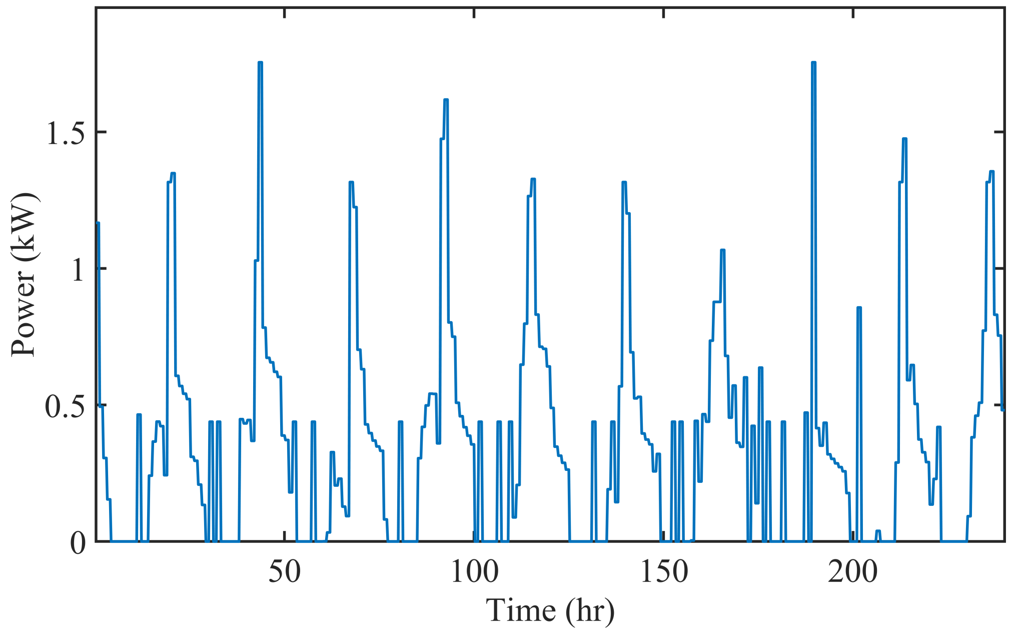

Figure 6.17: Power for best CPO control strategy with linear mapping.

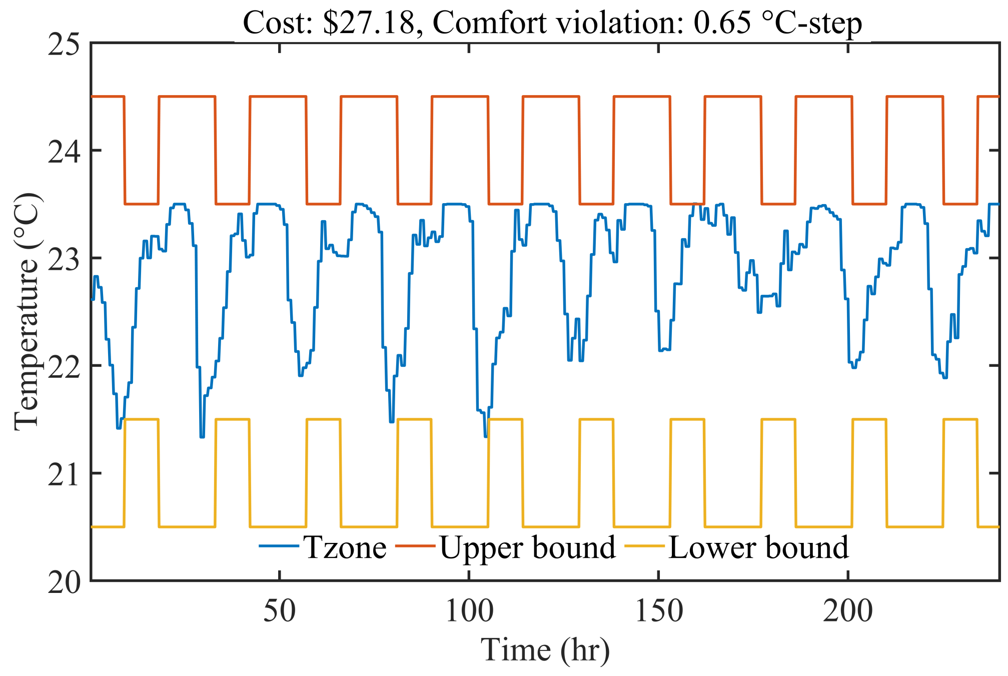

Figure 6.18: Zone temperature for worst CPO control strategy with linear mapping.

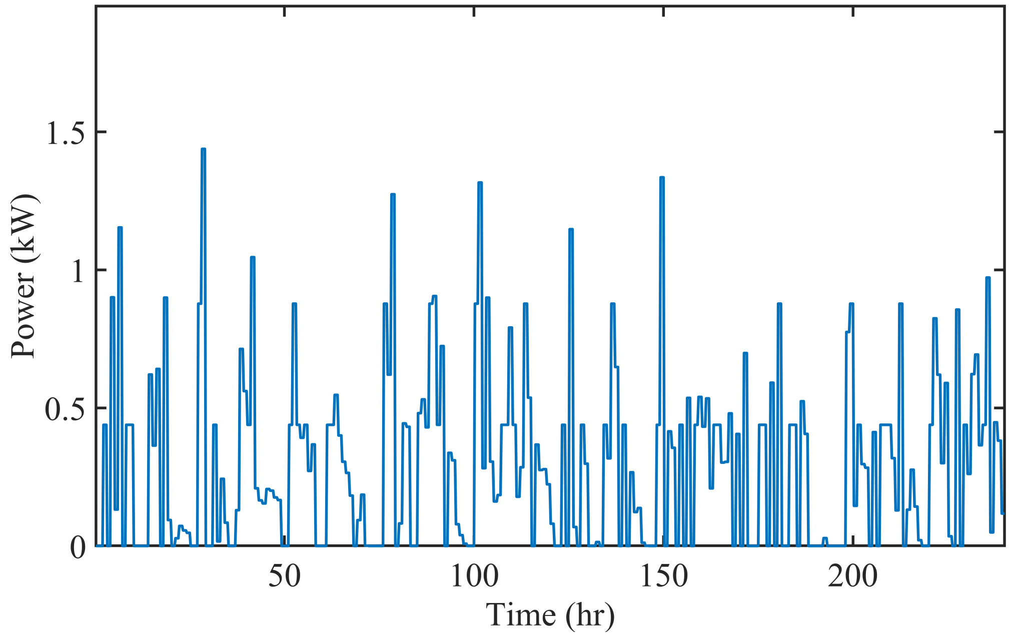

Figure 6.19: Power for worst CPO control strategy with linear mapping.

6.6 Parametric Study of Constraint Upper Limit

A logical assumption is that if the slack term is set to zero, then the control strategy obtained will strictly enforce constraints. However, since the CPO implementation solves an approximate version instead of the original policy optimization problem, the obtained solution may not be feasible for the original problem, leading to possible constraint violations. When the solution is infeasible, the control tends to minimize the constraint violations instead of the actual reward, which may result in sub-optimal performance cost-wise. This section seeks to evaluate the performance of the constrained RL algorithm under different constraint violation slack settings.

6.6.1 Simulation Setup

A series of constraint violation slack settings are considered in this parametric study. Eleven different values are tested from 0 to 5 and are evaluated under 10 different random seeds.

6.6.2 Simulation Results

Figure 6.20 shows the average cost and constraint violation averages under different constraint violation slack settings. When this setting approaches zero, the training becomes unstable since the agent is only seeking to reduce the constraint violation without any improvement in the cost performance. While a slack term of zero results in unstable training, small values such as $d = 0.1$ proved to result in performances that are comparable to the case study performed in section 6.5. It can also be seen that with the highest value tested, i.e. $d = 5$, the performance is comparable with the utility minimization MPC strategy while still having lower constraint violations than the unconstrained cases with similar electricity usage cost. However, the slack term should remain low since increasing it will allow more constraint violations. In conclusion, the algorithm only requires the slack term to be slightly higher than zero to perform well, and even higher values that allow for more constraint violations produce strategies that outperform the unconstrained cases.

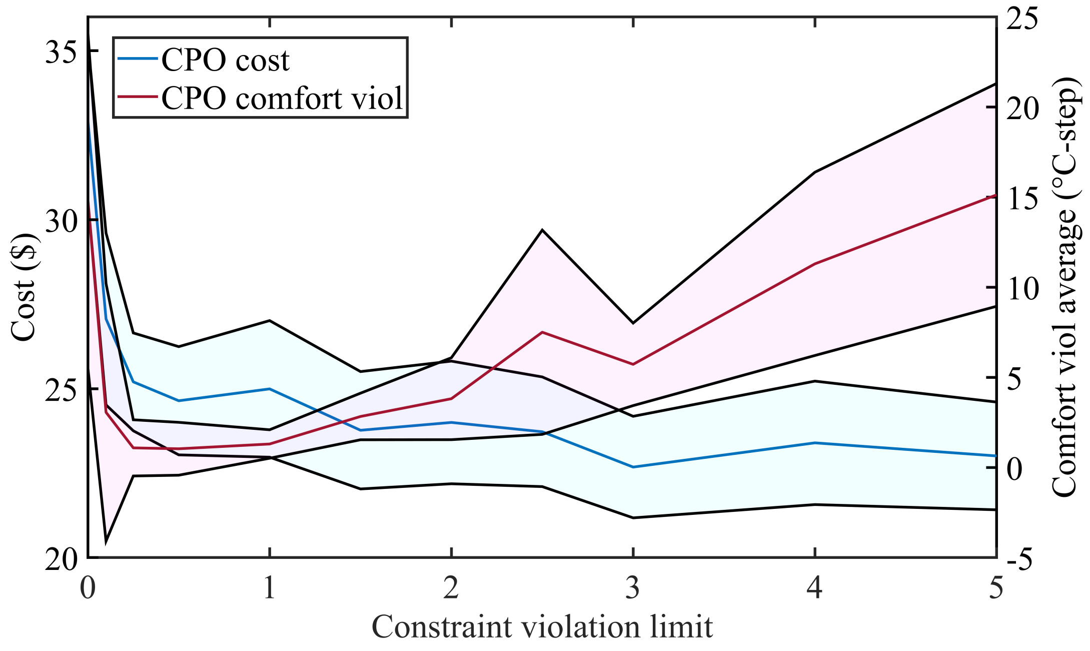

Figure 6.20: CPO performance under different constraint violation limits.

Table 6.5: Cost and Power Break-down

	Energy use (kWh)	**Energy cost ($)**	**Cost savings (%)**	**Comfort violation (°C-step)**
Emin	184.1	29.7	-	-
Umin	192.7	23.0	22.5	-
$\mathbf{TRPO_{0.01}}$	168.2	23.2	21.8	238
$\mathbf{TRPO_{0.1}}$	168.60	23.1	22.2	231.2
$\mathbf{TRPO_{1}}$	186.6	28.6	3.7	3
$\mathbf{TRPO_{10}}$	193	28.8	3	0.1
$\mathbf{PPO_{0.01}}$	185.2	21.9	26.2	155.4
$\mathbf{PPO_{0.1}}$	190.2	22.0	25.9	121.4
$\mathbf{PPO_{1}}$	204.9	27.0	9.0	5.5
$\mathbf{PPO_{10}}$	217	27.2	8.4	0.36
$\mathbf{CPO}$	220.2	26.0	12.3	4.2
$\mathbf{CPOLinear}$	240.1	24.9	16.1	1.3

CHAPTER 7

CONCLUSIONS

This thesis presented a constrained RL-based control strategy for demand responsive control of building thermal loads. The performance was assessed through comparisons to MPC and unconstrained RL baselines with different comfort penalty factors. The test results show that the constrained RL-based strategy was able to reduce the electricity cost by 16.1%, relative to an energy-minimizing controller, with very minor temperature excursions out of the comfort zone, while the unconstrained RL strategies led to either high energy costs or significant constraint violations, depending on the comfort penalty settings. The constrained RL controller achieved performances equivalent to 71% of the MPC utility minimization control benchmark, demonstrating its superior performance over unconstrained RL techniques for building demand response.

Two major observations can be made about the implementation of the proposed algorithm. First, the CPO algorithm was shown to be sensitive to violation slack d settings, especially when it approaches zero. This is due to the possibility of the approximate solution being infeasible for the original problem, which is the case for most update steps where the algorithm is mainly trying to reduce constraint violations instead of the control reward. A small slack term is effective in enforcing the constraints and results in performances that are better than the unconstrained cases. Secondly, it was shown that using linear mapping instead of deep neural networks for the policy and value functions could yield better results in terms of electricity cost and total constraint violations. The selection of an appropriate network architecture is crucial to achieve satisfactory performance and depends on the nature of the application.

7.1 Future work

Further development is necessary to allow the implementation of these algorithms for field applications. One of the weaknesses is the requirement of long training times because the on-policy implementation only updates the policy once every day, and simulation tests show that more than 24 months of training time is needed. Transfer learning and imitation learning are promising approaches to overcome this challenge. Transfer learning has been proven to be effective in reducing training time since it transfers the parameters of already trained buildings into untrained ones. This has been shown to improve learning speed even when the buildings have different physical characteristics such as layout, materials, weather conditions, or even different HVAC equipment [49]. Imitation learning algorithms also show good potential since they can harvest operational data and provide a better starting point for the constrained RL algorithm [50]. The integration of these techniques with constrained RL algorithms will be addressed in future work.

Another promising research direction is the use of constrained off-policy RL algorithms since they have the potential to use existing operational data and require less training time than on-policy algorithms [51].

REFERENCES

[1] *Eia, use of electricity explained*, https://www.eia.gov/energyexplained/index.php?page=electricity_use, Accessed: 2019-01-30.

[2] U. D. of Energy, *Benefits of demand response in electricity markets and recommendations for achieving them.* 2006.

[3] Q. Meng, Y. Li, X. Ren, C. Xiong, W. Wang, and J. You, "A demand-response method to balance electric power-grids via HVAC systems using active energy-storage: Simulation and on-site experiment", *Energy Reports*, vol. 7, pp. 762–777, 2021.

[4] M. Han *et al.*, "A review of reinforcement learning methodologies for controlling occupant comfort in buildings", *Sustainable Cities and Society*, vol. 51, p. 101 748, 2019.

[5] Y. Yao and D. K. Shekhar, "State of the art review on model predictive control MPC in heating ventilation and air-conditioning HVAC field", *Building and Environment*, vol. 200, p. 107 952, 2021.

[6] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control", *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 269–296, 2020.

[7] J. Drgoňa *et al.*, "All you need to know about model predictive control for buildings", *Annual Reviews in Control*, vol. 50, pp. 190–232, 2020.

[8] F. Oldewurtel, A. Ulbig, A. Parisio, G. Andersson, and M. Morari, "Reducing peak electricity demand in building climate control using real-time pricing and model predictive control", in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 1927–1932.

[9] D. T. Vedullapalli, R. Hadidi, and B. Schroeder, "Combined hvac and battery scheduling for demand response in a building", *IEEE Transactions on Industry Applications*, vol. 55, no. 6, pp. 7008–7014, 2019.

[10] O. V. Cutsem, M. Kayal, D. Blum, and M. Pritoni, "Comparison of mpc formulations for building control under commercial time-of-use tariffs", in *2019 IEEE Milan PowerTech*, 2019, pp. 1–6.

[11] D. K. Jie Cai James E. Braun and J. Hu, "General approaches for determining the savings potential of optimal control for cooling in commercial buildings having both energy and demand charges", *Science and Technology for the Built Environment*, vol. 22, no. 6, pp. 733–750, 2016.

[12] L. Yu, S. Qin, M. Zhang, C. Shen, T. Jiang, and X. Guan, “A review of deep reinforcement learning for smart building energy management”, *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12 046–12 063, Aug. 2021.

[13] Z. Wang and T. Hong, “Reinforcement learning for building controls: The opportunities and challenges”, *Applied Energy*, vol. 269, p. 115 036, 2020.

[14] G. P. Henze and J. Schoenmann, “Evaluation of reinforcement learning control for thermal energy storage systems”, *HVAC&R Research*, vol. 9, no. 3, pp. 259–275, 2003.

[15] S. Liu and G. Henze, “Experimental analysis of simulated reinforcement learning control for active and passive building thermal storage inventory –part 1: Theoretical foundation”, *Energy and Buildings*, vol. 38, pp. 142–147, Feb. 2006.

[16] S. Liu and G. Henze, “Evaluation of reinforcement learning for optimal control of building active and passive thermal storage inventory”, *Journal of Solar Energy Engineering-transactions of The Asme - J SOL ENERGY ENG*, vol. 129, May 2007.

[17] T. Wei, Y. Wang, and Q. Zhu, “Deep reinforcement learning for building hvac control”, in *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2017, pp. 1–6.

[18] K. U. Ahn and C. S. Park, “Application of deep q-networks for model-free optimal control balancing between different hvac systems”, *Science and Technology for the Built Environment*, vol. 26, no. 1, pp. 61–74, 2020.

[19] Z. Jiang *et al.*, “Building hvac control with reinforcement learning for reduction of energy cost and demand charge”, *Energy and Buildings*, vol. 239, p. 110 833, 2021.

[20] Y. Du *et al.*, “Intelligent multi-zone residential hvac control strategy based on deep reinforcement learning”, *Applied Energy*, vol. 281, p. 116 117, 2021.

[21] L. Yu *et al.*, “Deep reinforcement learning for smart home energy management”, *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2751–2762, 2020.

[22] X. Zhang, D. Biagioni, M. Cai, P. Graf, and S. Rahman, “An edge-cloud integrated solution for buildings demand response using reinforcement learning”, *IEEE Transactions on Smart Grid*, vol. PP, pp. 1–1, Aug. 2020.

[23] K. Kurte *et al.*, “Evaluating the adaptability of reinforcement learning based hvac control for residential houses”, *Sustainability*, vol. 12, no. 18, 2020.

[24] H. Li, Z. Wan, and H. He, “Real-time residential demand response”, *IEEE Transactions on Smart Grid*, vol. 11, no. 5, pp. 4144–4154, 2020.

[25] X. Zhang, R. Chintala, A. Bernstein, P. Graf, and X. Jin, *Grid-interactive multi-zone building control using reinforcement learning with global-local policy search*, 2020.

[26] D. Azuatalam, W.-L. Lee, F. de Nijs, and A. Liebman, "Reinforcement learning for whole-building hvac control and demand response", *Energy and AI*, vol. 2, p. 100 020, 2020.

[27] A. H. Hosseinloo, S. Nabi, A. Hosoi, and M. A. Dahleh, "Data-driven control of covid-19 in buildings: A reinforcement-learning approach", *IEEE Transactions on Automation Science and Engineering*, pp. 1–, 2023.

[28] F. Schmoeller Roza, H. Rasheed, K. Roscher, X. Ning, and S. Günnemann, "Safe robot navigation using constrained hierarchical reinforcement learning", in *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2022, pp. 737–742.

[29] H. Li, Z. Wan, and H. He, "Constrained ev charging scheduling based on safe deep reinforcement learning", *IEEE Transactions on Smart Grid*, vol. 11, no. 3, pp. 2427–2439, 2020.

[30] Y.-J. Chen, D.-K. Chang, and C. Zhang, "Autonomous tracking using a swarm of uavs: A constrained multi-agent reinforcement learning approach", *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13 702–13 717, 2020.

[31] J. Cai and J. Braun, "An inverse hygrothermal model for multi-zone buildings", *Journal of Building Performance Simulation*, vol. 9, no. 5, pp. 510–528, 2016.

[32] J. Cai, J. E. Braun, D. Kim, and J. Hu, "A multi-agent control based demand response strategy for multi-zone buildings", in *2016 American Control Conference (ACC)*, 2016, pp. 2365–2372.

[33] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Second. The MIT Press, 2018.

[34] R. D. Smallwood and E. J. Sondik, "The optimal control of partially observable markov processes over a finite horizon", *Operations Research*, vol. 21, no. 5, pp. 1071–1088, 1973.

[35] V. Mnih *et al.*, *Playing atari with deep reinforcement learning*, 2013.

[36] J. Peters and S. Schaal, "Policy gradient methods for robotics", in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 2219–2225.

[37] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, *High-dimensional continuous control using generalized advantage estimation*, 2018.

[38] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation”, in *Proceedings of the 12th International Conference on Neural Information Processing Systems*, ser. NIPS’99, Denver, CO: MIT Press, 1999, pp. 1057–1063.

[39] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, “Trust region policy optimization”, *CoRR*, vol. abs/1502.05477, 2015.

[40] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms”, *CoRR*, vol. abs/1707.06347, 2017.

[41] *Constrained Markov Decision Processes*. 1999.

[42] J. Achiam, D. Held, A. Tamar, and P. Abbeel, “Constrained policy optimization”, *CoRR*, vol. abs/1705.10528, 2017.

[43] E. P. Electric, *Small general service rate*, https://www.epelectric.com/customers/rates-and-regulations/business-rates-and-information/texas-rate-tariffs-rules-and-regulations/texas-rate-tariffs, Nov. 2021.

[44] M. Grant and S. Boyd, *CVX: Matlab software for disciplined convex programming, version 2.1*, http://cvxr.com/cvx, Mar. 2014.

[45] *Gurobi solver*, Online at https://www.gurobi.com/.

[46] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-baselines3: Reliable reinforcement learning implementations”, *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.

[47] A. Paszke *et al.*, “Automatic differentiation in pytorch”, 2017.

[48] B. Recht, *A tour of reinforcement learning: The view from continuous control*, 2018.

[49] S. Xu, Y. Wang, Y. Wang, Z. O’Neill, and Q. Zhu, “One for many”, in *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, ACM, Nov. 2020.

[50] S. Dey, T. Marzullo, X. Zhang, and G. Henze, “Reinforcement learning building control approach harnessing imitation learning”, *Energy and AI*, vol. 14, p. 100 255, 2023.

[51] G. Kalweit, M. Huegle, M. Werling, and J. Boedecker, *Deep constrained q-learning*, 2020.