Physics-Informed RL for HVAC Controls: Imitation to Online Fine-Tuning

Anonymous Authors[1]

Abstract

Reinforcement learning (RL) deployment in real-world, safety-critical systems remains a significant challenge despite advancements in the field. This work analyzes practical obstacles to deploying RL for HVAC systems through a case study on residential heat pump control, utilizing real-world data to apply and extend a model-based RL algorithm. To enhance interpretability and safety, we incorporated a physics-informed system model and faced challenges in parameter estimation due to local minima, even in idealized settings with sufficient exploration. Given the complexities of reward shaping without a simulator, inverse RL was employed to learn cost parameters from an existing controller. We then introduced two calibration strategies to impose user-defined control characteristics via a simple, generic reward function with a single thermal discomfort price parameter. Ultimately, this research aims to advance the practical application of RL in safety-critical systems, offering insights for bridging the gap between simulation and real-world deployment.

1. Introduction

Despite reinforcement learning's (RL) success in simulated domains (Schrittwieser et al., 2020; Plaat et al., 2023; An et al., 2023), its deployment in safety-critical real-world systems like HVAC control is hindered by safety, reliability, and interpretability concerns, particularly without high-fidelity simulators (Berkenkamp et al., 2017; Tran et al., 2019; Zanon & Gros, 2020). Real-world HVAC applications demand occupant comfort, equipment protection, and energy efficiency, yet understanding of RL's practical obstacles in such physical environments is limited due to the rarity of actual deployments (Nagy et al., 2023), with most research confined to high-fidelity simulators (An et al., 2023; Ding et al., 2019; Zhang et al., 2022; Chen et al., 2021) or small-scale lab setups (Chen et al., 2019). As a result, the practical limitations of RL in real-world settings – not only in terms of algorithmic performance, which varies across systems, but also the challenges faced prior to deployment, such as model mismatch, reward design, and behavior alignment – are not yet well-understood.

In this work, we conduct preliminary experiments for those nuances by demonstrating the challenges of transitioning from imitation learning to online fine-tuning for real-world HVAC controls, using data from a residential building with a heat pump system. We apply and extend a state-of-the-art RL algorithm for HVAC control (Gnu-RL (Chen et al., 2019)), which leverages Differentiable Model Predictive Control (MPC) policy (Amos et al., 2018). Our experiments revealed key challenges. With respect to the learned *system dynamics*, we found that fitting a 2R1C thermal circuit model with Gnu-RL to yield physically plausible parameters proved difficult, with optimization often settling on local minima even in simplified simulation environments. We found that action space explorations improved parameter convergence for action-related parameters but, critically, not for all physical parameters of the model; this partial success, even in our idealized settings, challenges the common assumption that sufficient exploration would enable accurate system identification. Regarding *behavior*[1], traditional reward shaping is impractical for real-world systems. We initially used imitation learning to derive cost parameters from an existing controller for a safe starting point, but this failed to replicate desired behavior despite low loss values. Consequently, we developed two calibration strategies – initial and online – employing a simple, generic reward function with a single user-defined "discomfort price." These strategies successfully imposed user-desired characteristics, especially via online learning, offering a scalable and intuitive alternative to manual reward shaping for HVAC controls.

While our observations stem from experiments with a specific RL algorithm in the context of a single real building, they shed light on the complex path towards practical, interpretable model-based RL for HVAC systems. And although

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

[1]'Behavior' here indicates the system's operational outcomes in achieving user-defined objectives (e.g., comfort-energy trade-offs), as dictated by the differentiable MPC's learned cost parameters.

we propose strategies to ameliorate some of the real-world deployment challenges we observed, our main goal is to characterize these challenges well enough to incite critical discussion by the community.

1.1. Background

Offline RL followed by online fine-tuning has been effective in robotics and HVAC control, where methods like Conservative Q-Learning (Kumar et al., 2020) have shown success (Zhang et al., 2022). Offline RL can be model-free or model-based. While model-free RL has been applied to HVAC (e.g., (Zhang et al., 2018), (Ding et al., 2019)), it often demands extensive online data, requiring simulation with a reasonably accurate model. Some offline approaches use historical data to mimic existing controllers (Li et al., 2019; Jia et al., 2019), but model-free RL generally remains sample-inefficient. Model-based RL typically surpasses model-free methods in sample and energy efficiency and occupant comfort for HVAC (Nagy et al., 2018), as it learns system dynamics for planning – crucial for energy efficiency, demand response, and comfort. Examples include MPC variants with neural networks (Ding et al., 2020; Zhang et al., 2022) and Gaussian process models (An et al., 2023). Despite simulation success, these models are often black-box, lacking interpretability, requiring substantial training data, and facing scalability and expert trust challenges for real-world deployment (Nagy et al., 2023).

Thus, to avoid the pitfalls of purely black-box or model-free methods while leveraging historical data, we adopt a model-based RL strategy centered on Differentiable MPC policy (Amos et al., 2018), which enables end-to-end learning of cost and system dynamics in imitation learning. Gnu-RL (Chen et al., 2019) improved this for online learning in HVAC. Differentiable MPC learns parameters defining a nonconvex cost function and state transitions by iteratively linearizing dynamics and forming quadratic cost approximations around a fixed point. It differentiates implicitly through the Karush-Kuhn-Tucker (KKT) conditions of this solution, reducing computational load compared to back-propagation through optimization iterations. This allows end-to-end learning of cost and dynamics, with a Linear Quadratic Regulator (LQR)-based policy solving the locally linearized problem. More background in Differentiable MPC and Gnu-RL is provided in the Appendix Section A.

2. Methodology

Before discussing deployment challenges, this section outlines extensions made to Gnu-RL (Chen et al., 2019) and Differentiable MPC (Amos et al., 2018) for our controller. Gnu-RL typically fits a simple linear model for its system dynamics function, without explicitly mapping the parameters to physical phenomena, thus limiting interpretability. We address this by using a function structure with the first law of thermodynamics as the prior to fitting physically relevant parameters θ_{state}. Secondly, Gnu-RL assumes manual configuration of the LQR cost parameters (θ_{cost}), hindering scalability. We first learn θ_{cost} from existing controller data (inverse RL) for a safe initial policy, then refine them via calibration strategies using a non-quadratic reward function.

Physics-Based 2R1C Model for System Dynamics. To enhance interpretability, we integrated a physics-based 2R1C thermal resistance-capacitance (RC) model into the Gnu-RL policy, replacing its default linear dynamics model. This 2R1C architecture (see Appendix B.1 for full equations, e.g., Eq. (3)) was chosen for consistency with a prior study on the same testbed. The model describes the indoor temperature (x_k) dynamics based on inputs from the heat pump and backup heater (u_k), as well as disturbances like outdoor temperature and solar gain (d_k). The model is discretized using the Zero-Order Hold (ZOH) method, assuming inputs are piecewise constant over each time step. The resulting discrete-time state-space model is detailed by Eq. (5) in Appendix B.1).

This formulation allows fitting physically meaningful parameters $\theta_{state} = \{C, R_m, R_{out}, T_m, \eta, \alpha\}$ (thermal capacitances, resistances, etc.). The Differentiable MPC framework uses an internal LQR to track a target temperature setpoint, $x_{k,\text{target}}$, with learnable cost parameters $\theta_{cost} = \{O, R_1, R_2\}$ for state and control efforts. The overall parameters $\theta = \{\theta_{state}, \theta_{cost}\}$ are learned by minimizing an imitation learning loss (see Eq. (6) in Appendix B.1) that penalizes Mean Squared Error (MSE) deviations in predicted states ($\mathcal{L}_{state}$) and actions ($\mathcal{L}_{action}$) from historical data. Within this Differentiable MPC framework, θ_{state} is thus learned via system identification (minimizing $\mathcal{L}_{state}$), while θ_{cost} is adapted (as detailed below and in Appendix B.2) to optimize overall control behavior against the non-quadratic reward. It also enforces box constraints on control inputs (e.g., $P_{HP}^{\text{min/max}}$).

Further Calibration of Cost Parameters. As discussed in Section 3.2, fitting LQR cost parameters (θ_{cost}) solely through imitation learning from the existing (suboptimal) controller failed to yield a reasonable control behavior. Thus, we introduce a non-quadratic reward function designed to capture user preferences regarding energy cost and thermal comfort more effectively. This reward function[2], detailed in Appendix B.2 (Eq. (7)), primarily requires a single user-defined parameter: the thermal discomfort price (w_c), which allows intuitive specification of the trade-off between energy savings and comfort. Other components include monetary cost for energy consumption (w_e) and peak power charge (w_d), which can be retrieved from local utilities.

[2]It aims to minimize peak power, energy consumption and temperature deviations.

We employ two strategies to calibrate the LQR's θ_{cost} to align with this non-quadratic generic reward. First, **Online Calibration** updates both system dynamics parameters (θ_{state}) and cost parameters (θ_{cost}) concurrently during deployment. θ_{state} is learned by minimizing state prediction errors against real-world observations, while θ_{cost} is updated by maximizing the proposed reward function (Eq. (7)) via gradient ascent through the Differentiable MPC policy, which leverages its internal (differentiable) system model based on the current θ_{state}. Second, **Initial Calibration** updates θ_{cost} before deployment (or when cost coefficients w_d, w_e, w_c change) by iteratively solving the Differentiable MPC policy with an initial condition and maximizing the cumulative reward over its predicted trajectory (generated using its internal differentiable model). During subsequent deployment, θ_{state} adapts online, but θ_{cost} remains fixed unless a recalibration is triggered. These calibration strategies aim to bridge the gap between the LQR's quadratic cost and more complex, user-defined objectives. The detailed procedures are provided in Appendix B.2.

3. Results

Imitation learning was done using one month of operational data (Nov/01–Nov/29/2023) and validated on two weeks of data (Dec/15–Dec/30/2023) collected from the -Anonymous Location- testbed. This data was recorded while the building was under the control of a reactive controller. The raw 5-minute resolution data, which includes indoor/outdoor temperatures and power consumption for the heat pump and backup heat, was resampled to an hourly resolution. Figure 5 illustrates temperature measurements of the training data. In the rest of this section, we look into the two major components of Gnu-RL (system dynamics and behavior), and propose potential directions for the observed challenges.

3.1. System Dynamics

In RL and differentiable control, hyperparameter tuning is often guided by minimizing a predefined loss function, with the assumption that achieving the lowest validation loss corresponds to a well-generalized model. However, in the context of system identification for thermal networks, this assumption often does not hold (Atam & Helsen, 2016), given issues such as limited operational excitations and inherent model non-convexities. To validate this further, we examined the hyperparameters (lr=0.05, λ=0.001) yielding the lowest state loss (an MSE of 0.048 $^\circ C^2$), even lower than the more complex model used in the prior MPC study. However, these small loss values can be misleading, as they corresponded to learned parameters that lack physical plausibility (see Appendix C.1 for more details), mainly because of the existence of many local minima where not all of them are physically valid.

Online learning in a simple environment. This challenge led us to investigate whether online learning could drive the learned parameters toward their true values under idealized conditions: in an environment with the same underlying system dynamics function structure but different parameter values. To explore this, we conducted experiments in a **simulated 2R1C network** where ground-truth parameters were predefined, and online learning was performed with real-world external inputs (i.e., T_{out} and Q_{sol}).

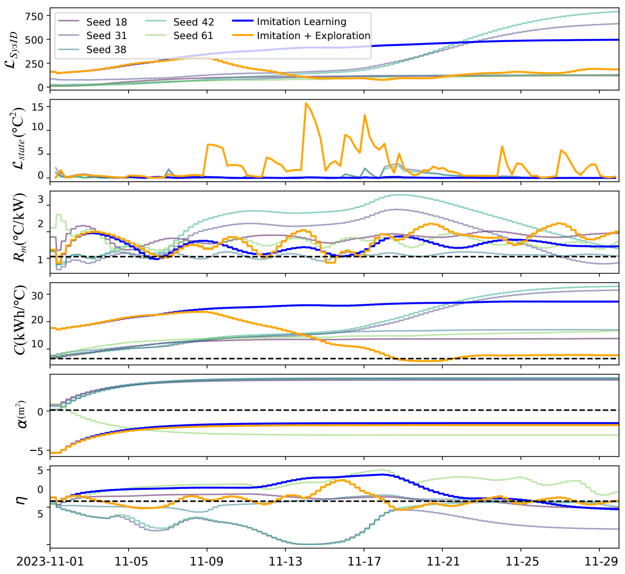

Figure 1. Parameter trajectory during online learning in a simple 2R1C environment; parameters of which shown in dashed lines.

Figure 1 illustrates system identification performance. The top plot shows SysID loss ($\mathcal{L}_{SysID} = \|\theta_{state,true} - \theta_{state,pred}\|_2^2$), comparing predicted model parameters ($\theta_{state,pred}$) to true environment parameters ($\theta_{state,true}$, dashed lines in bottom plots). The second plot shows state prediction MSE ($\mathcal{L}_{state}$) loss, while other plots track select 2R1C parameters. We compare three conditions: 'Imitation Learning' (parameters initialized by an imitation agent, then updated via environment interaction but no exploration), 'Imitation + Exploration' (adds decaying Gaussian noise to actions for exploration), and 'Seeds' (randomly initialized parameters from reasonable bounds).

Online fine-tuning rapidly reduces $\mathcal{L}_{state}$ (next-state prediction error). However, inferred parameters often diverge from true values, with $\mathcal{L}_{SysID}$ increasing as models reach local minima. For instance, η becomes negative (implying heating cools) and solar aperture becomes negatively large (implying solar gain cools), despite low state prediction error. The 'Imitation + Exploration' condition improves convergence for parameters directly related to actions (C and η), reducing $\mathcal{L}_{SysID}$, though $\mathcal{L}_{state}$ is slightly higher.

Note that convergence is only observed for parameters interacting multiplicatively with the explored actions. Thus, while control and RL literature commonly assumes that exploration (or system excitation) can enable accurate system dynamics fitting, our work demonstrates that, even in these idealized settings, only parameters directly related to actions consistently converge to their true values.

3.2. Behavior

Differentiable MPC assumes that trajectories to fit θ come from an expert controller, but in HVAC systems, such an expert is unavailable. Instead, prior work like Gnu-RL used data from a suboptimal existing controller to fit state dynamics and manually configured the cost function θ_{cost}. However, manual configuration of a quadratic cost function is not scalable and biases the fitting of the system dynamics model. In this work, we investigate whether initially fitting θ_{cost} using data from an existing controller can provide a *safe yet suboptimal* policy that serves as a baseline for further refinement through online learning, reducing reliance on manual cost tuning.

Our attempts at behavior cloning of the existing controller, even under idealized assumptions of perfect system dynamics, proved challenging. Despite achieving a low loss in matching the expert actions (MSE of 0.11 (in kW^2)), the resulting imitation learning agent failed to replicate the baseline controller's behavior, leading to large user discomfort. A detailed analysis of this is provided in Appendix C.2.

Imposing user-defined control characteristics. To address this challenge, we explored methods to represent the efficiency-comfort trade-off more interpretably and adaptably. We proposed using a reward function to guide the learning of θ_{cost}, as detailed in Section 2. While reward shaping is a known challenge in RL, our approach requires only one user-defined parameter: the thermal discomfort price w_c, enabling users to intuitively specify their preferred trade-off between energy savings and comfort. We tested two calibration strategies – initial calibration and online calibration – to map the reward function characteristics to θ_{cost}. To isolate these from state dynamics fitting challenges, we evaluated them in a simulated 2R1C environment, with parameters derived from imitation learning (lr = 0.05, $\lambda = 1$ for equal state-action loss weighting).

The results in Figure 2 illustrate distinct control behaviors with statistics such as Predicted Percentage Dissatisfied (PPD) (ASHRAE, 2004) and energy consumption. The imitation learning agent fails by causing significant thermal discomfort. Initial calibration rapidly achieves and maintains the setpoint, while online calibration gradually adjusts over several days to user-specified preferences. It is crucial that in the final cycle, both methods apply a stepped increase, minimizing backup heat usage when adjusting to

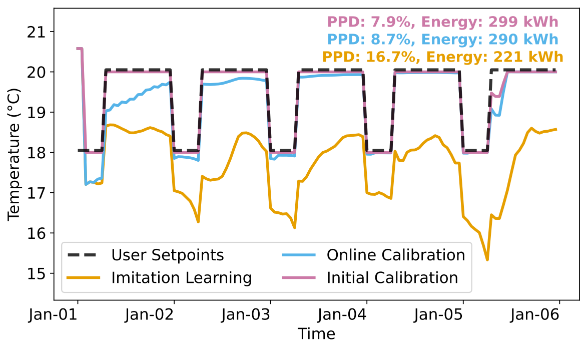

Figure 2. Simulation with varying cost parameters under assumption of perfect state dynamics modeling.

the shift in setpoints under colder weather, providing greater efficiency. Each approach has trade-offs: online calibration continuously adapts to system variations (though no deviation is observed here due to zero state loss) but may introduce early discomfort, whereas initial calibration prevents early discomfort but risks long-term deviation from user preferences as system dynamics evolve. These findings highlight the importance of selecting calibration strategies based on application-specific control objectives.

Overall, a fundamental duality complicates the joint identification of θ_{state} and θ_{cost}: system identification requires excitation outside nominal operating regimes (Atam & Helsen, 2016), while fitting cost parameters requires stable controller data. Our work suggests that exploration in actions does not result in convergence to true system dynamics. Further, the proposed way of imposing user-defined control characteristics may serve as an alternative to manual-reward engineering, which is typically necessary in many RL problems yet not practical in deployment.

4. Conclusions

In this work, we investigate the challenges of applying Differentiable MPC to real-world HVAC control, focusing on system identification and behavior alignment. Our results reveal that minimizing loss functions alone is insufficient for learning physically plausible system dynamics, as parameters often converge to unrealistic values despite low prediction errors. Online fine-tuning fails to drive all parameters toward realistic physical values even in simple settings. Furthermore, we demonstrate that imitation learning fails to replicate the behavior of the existing controller. We proposed two calibration strategies for successfully introducing user-defined control characteristics with minimal reward shaping. These findings underscore the challenges in deploying scalable and interpretable solutions to bridge the gap between theoretical advancements in model-based RL and practical deployment in real-world systems.

Impact Statement

This paper presents work whose goal is to advance the field of Reinforcement Learning for Safety-Critical Systems. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Amos, B., Jimenez, I., Sacks, J., Boots, B., and Kolter, J. Z. Differentiable mpc for end-to-end planning and control. *Advances in Neural Information Processing Systems*, 31, 2018.

An, Z., Ding, X., Rathee, A., and Du, W. CLUE: Safe Model-Based RL HVAC Control Using Epistemic Uncertainty Estimation. In *Proceedings of the 10th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '23, pp. 149–158, New York, NY, USA, November 2023. Association for Computing Machinery. ISBN 9798400702303. doi: 10.1145/3600100.3623742. URL https://dl.acm.org/doi/10.1145/3600100.3623742.

ASHRAE. *Thermal Environmental Conditions for Human Occupancy*. American Society of Heating, Refrigerating and Air-Conditioning Engineers, Atlanta, GA, standard 55-2004 edition, 2004. Defines the Percentage of People Dissatisfied (PPD) and Predicted Mean Vote (PMV) models for thermal comfort assessment.

Atam, E. and Helsen, L. Control-oriented thermal modeling of multizone buildings: Methods and issues: Intelligent control of a building system. *IEEE Control systems magazine*, 36(3):86–111, 2016.

Berkenkamp, F., Turchetta, M., Schoellig, A., and Krause, A. Safe model-based reinforcement learning with stability guarantees. *Advances in Neural Information Processing Systems*, 30, 2017.

Chen, B., Cai, Z., and Bergés, M. Gnu-RL: A Precocial Reinforcement Learning Solution for Building HVAC Control Using a Differentiable MPC Policy. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pp. 316–325, New York NY USA, November 2019. ACM. ISBN 978-1-4503-7005-9. doi: 10.1145/3360322.3360849. URL https://dl.acm.org/doi/10.1145/3360322.3360849.

Chen, B., Donti, P. L., Baker, K., Kolter, J. Z., and Bergés, M. Enforcing Policy Feasibility Constraints through Differentiable Projection for Energy Optimization. In *Proceedings of the Twelfth ACM International Conference on Future Energy Systems*, pp. 199–210, Virtual Event Italy, June 2021. ACM. ISBN 978-1-4503-8333-2. doi: 10.1145/3447555.3464874. URL https://dl.acm.org/doi/10.1145/3447555.3464874.

Ding, X., Du, W., and Cerpa, A. Octopus: Deep reinforcement learning for holistic smart building control. In *Proceedings of the 6th ACM international conference on systems for energy-efficient buildings, cities, and transportation*, pp. 326–335, 2019.

Ding, X., Du, W., and Cerpa, A. E. Mb2c: Model-based deep reinforcement learning for multi-zone building control. In *Proceedings of the 7th ACM international conference on systems for energy-efficient buildings, cities, and transportation*, pp. 50–59, 2020.

Jia, R., Jin, M., Sun, K., Hong, T., and Spanos, C. Advanced building control via deep reinforcement learning. *Energy Procedia*, 158:6158–6163, 2019.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191, 2020.

Li, Y., Wen, Y., Tao, D., and Guan, K. Transforming cooling optimization for green data center via deep reinforcement learning. *IEEE transactions on cybernetics*, 50(5):2002–2013, 2019.

Nagy, A., Kazmi, H., Cheaib, F., and Driesen, J. Deep reinforcement learning for optimal control of space heating. *arXiv preprint arXiv:1805.03777*, 2018.

Nagy, Z., Henze, G., Dey, S., Arroyo, J., Helsen, L., Zhang, X., Chen, B., Amasyali, K., Kurte, K., Zamzam, A., Zandi, H., Drgoňa, J., Quintana, M., McCullogh, S., Park, J. Y., Li, H., Hong, T., Brandi, S., Pinto, G., Capozzoli, A., Vrabie, D., Bergés, M., Nweye, K., Marzullo, T., and Bernstein, A. Ten questions concerning reinforcement learning for building energy management. *Building and Environment*, 241:110435, August 2023. ISSN 03601323. doi: 10.1016/j.buildenv.2023.110435. URL https://linkinghub.elsevier.com/retrieve/pii/S0360132323004626.

Plaat, A., Kosters, W., and Preuss, M. High-accuracy model-based reinforcement learning, a survey. *Artificial Intelligence Review*, 56(9):9541–9573, 2023.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

Tran, H.-D., Cai, F., Diego, M. L., Musau, P., Johnson, T. T., and Koutsoukos, X. Safety verification of cyber-physical systems with reinforcement learning control. *ACM Transactions on Embedded Computing Systems (TECS)*, 18(5s): 1–22, 2019.

Zanon, M. and Gros, S. Safe reinforcement learning using robust mpc. *IEEE Transactions on Automatic Control*, 66 (8):3638–3652, 2020.

Zhang, C., Kuppannagari, S. R., and Prasanna, V. K. Safe building hvac control via batch reinforcement learning. *IEEE Transactions on Sustainable Computing*, 7(4):923–934, 2022.

Zhang, Z., Chong, A., Pan, Y., Zhang, C., Lu, S., and Lam, K. P. A deep reinforcement learning approach to using whole building energy model for hvac optimal control. In *2018 Building Performance Analysis Conference and SimBuild*, volume 3, pp. 22–23, 2018.

A. Background in Differentiable MPC and Gnu-RL

Though Differentiable MPC policy allows for nonconvex dynamics, Gnu-RL fits the following linear state-space model:

$$x_{k+1} = Ax_k + B_u u_k + B_d d_k = \underbrace{\begin{bmatrix} A & B_u \end{bmatrix}}_{F} \underbrace{\begin{bmatrix} x_k \\ u_k \end{bmatrix}}_{\tau_k} + \underbrace{B_d d_k}_{f_k}. \tag{1}$$

In the context of applying Differentiable MPC to policy learning in HVAC controls, Gnu-RL uses the following quadratic cost function:

$$\begin{aligned} C_k(x_k, u_k) &= \frac{1}{2} x_k^T O_k x_k + p_k^T x_k + \frac{1}{2} u_k^T R_k u_k + s_k^T u_k \\ &= \frac{1}{2} \underbrace{\begin{bmatrix} x_k^T & u_k^T \end{bmatrix}}_{\tau_k^T} \underbrace{\begin{bmatrix} O_k & 0 \\ 0 & R_k \end{bmatrix}}_{C_k} \underbrace{\begin{bmatrix} x_k \\ u_k \end{bmatrix}}_{\tau_k} + \underbrace{\begin{bmatrix} p_k^T \, s_k^T \end{bmatrix}}_{c_k^T} \underbrace{\begin{bmatrix} x_k \\ u_k \end{bmatrix}}_{\tau_k}, \end{aligned} \tag{2}$$

The dynamics function (Eq. 1) and cost function (Eq. 2) use the following variables. The system considers a scalar state x_k, representing the thermostat temperature, and a control action u_k, defined as the difference between the supply and the mixed air temperature. In addition, it accounts for uncontrollable disturbances d_k, such as weather conditions and internal thermal gains. The model learns scalar parameters A and B_u that form the matrix $F = \begin{bmatrix} A & B_u \end{bmatrix}$ in the dynamics equation. Lastly, Gnu-RL only learns the parameter B_d, using disturbances (d_k) by weather forecasts.

B. Details of Our Approach

B.1. Physics-Based 2R1C Model and System Dynamics

The system dynamics are represented by a physics-based 2R1C thermal RC model. The continuous-time energy balance for the indoor temperature T is given by:

$$C\frac{dT}{dt} = \frac{T_m - T}{R_m} + \frac{T_{out} - T}{R_{out}} + \underbrace{COP(T_{out})P_{HP} + \eta P_{BH}}_{\dot{Q}_c} + \underbrace{\alpha Q_{sol}}_{\dot{Q}_e} \tag{3}$$

where:

- C: Thermal capacitance of the indoor air.
- T: Indoor air temperature.
- T_m: Effective thermal mass temperature (assumed constant).
- T_{out}: Outdoor air temperature.
- R_m: Thermal resistance between indoor air and the thermal mass.
- R_{out}: Thermal resistance between indoor air and the outdoor environment.
- P_{HP}: Electrical power consumed by the heat pump.
- P_{BH}: Electrical power consumed by the backup heater.
- $COP(T_{out})$: Coefficient of Performance of the heat pump, a function of T_{out}, taken from -AnonymousPublication-.
- η: Efficiency of the backup heater.
- Q_{sol}: Solar irradiance.
- α: Solar aperture coefficient (effective window area).
- $\dot{Q}_c$: Controlled heat input.

- $\dot{Q}_e$: Uncontrolled heat input from solar gains.

The continuous-time dynamics (Eq. (3)) are first represented in a state-space form:

$$\dot{x}(t) = A_c x(t) + B_{uc}(T_{out,t})u(t) + B_{dc}d(t) \tag{4}$$

where $x(t) = T(t)$ is the indoor temperature. The continuous-time system matrix A_c, the continuous-time input matrix $B_{uc}(T_{out,t})$ (which depends on the outdoor temperature $T_{out,t}$ due to the Coefficient of Performance COP), and the continuous-time disturbance matrix B_{dc} are defined based on the physical parameters as:

$$\begin{aligned} A_c &= \left[-\frac{1}{R_m C} - \frac{1}{R_{out} C}\right] \\ B_{uc}(T_{out,t}) &= \begin{bmatrix} \frac{COP(T_{out,t})}{C} & \frac{\eta}{C} \end{bmatrix} \\ B_{dc} &= \begin{bmatrix} \frac{1}{R_m C} & \frac{1}{R_{out} C} & \frac{\alpha}{C} \end{bmatrix} \end{aligned}$$

This continuous-time system is then discretized using the ZOH method, assuming that the control inputs u_k and disturbances d_k are constant over each sampling interval of duration Δt. This results in the discrete-time linear state-space model $x_{k+1} = Ax_k + B_u[T_{out,k}]u_k + B_d d_k$, where x_k denotes $x(k\Delta t)$:

$$\underbrace{T_{k+1}}_{x_{k+1}} = \underbrace{A}_{A} \underbrace{T_k}_{x_k} + \underbrace{B_u[T_{out,k}]}_{B_u[T_{out,k}]} \underbrace{\begin{bmatrix} P_{HP,k} \\ P_{BH,k} \end{bmatrix}}_{u_k} + \underbrace{B_d}_{B_d} \underbrace{\begin{bmatrix} T_m \\ T_{out,k} \\ Q_{sol,k} \end{bmatrix}}_{d_k} \tag{5}$$

The discrete-time matrices A, $B_u[T_{out,k}]$, and B_d are derived from the continuous-time matrices A_c, $B_{uc}(T_{out,k})$, and B_{dc} as follows:

$$\begin{aligned} A &= \exp(A_c) \\ B_u[T_{out,k}] &= (A_c)^{-1}(\exp(A_c) - I)B_{uc}(T_{out,k}) \\ B_d &= (A_c)^{-1}(\exp(A_c) - I)B_{dc} \end{aligned}$$

where I is the identity matrix (a scalar 1 in this single-state case) and $\exp(\cdot)$ denotes the matrix exponential (scalar exponential here). It is noteworthy that the time step Δt is involved in the practical computation of these discrete matrices within the model's implementation. Specifically, intermediate matrices are typically formed using parameters scaled by Δt (e.g., an intermediate system matrix equivalent to $A_c/\Delta t$ is formulated from the model parameters R_m, C, R_{out} and Δt). The final discrete matrices A, B_u, B_d are then obtained by applying transformations (such as $A = \exp((A_c/\Delta t)\Delta t)$), which mathematically simplify to the forms $A = \exp(A_c)$, etc., shown above.

The state vector x_k comprises the indoor temperature T_k at discrete time step k. The control inputs u_k are the heat pump power $P_{HP,k}$ and backup heat power $P_{BH,k}$. The disturbance vector d_k includes the thermal mass temperature T_m (assumed constant), outdoor air temperature $T_{out,k}$, and solar gains $Q_{sol,k}$. Since $COP(T_{out,k})$ varies with the outdoor temperature, the input matrix B_u is recomputed at each time step k based on predicted $T_{out,k}$ values.

The parameters fitted for the system dynamics model, denoted θ_{state}, are: $\{C, R_m, R_{out}, T_m, \eta, \alpha\}$.

For trajectory tracking within the Differentiable MPC framework, the quadratic cost function minimized by the LQR at each step is shown in Eq. 2 in its general form. We set $p_k = -O_k x_{k,target}$ to track a target setpoint $x_{k,target}$, and $s_k = 0$. This simplifies the cost parameters to be learned, θ_{cost}, to three values: $O \in \mathbb{R}$ (state cost weight) and $R = \mathrm{diag}(R_1, R_2)$, where $R_1, R_2 \in \mathbb{R}$ are the control cost weights for P_{HP} and P_{BH} respectively, forming the diagonal elements of the control weighting matrix R_k in Eq. (2). The imitation learning loss function is:

$$\mathcal{L}_{imit}(\theta) = \frac{1}{M} \sum_{k}^{M} \|x_{k+1} - x^*_{k+1}\|_2^2 + \lambda \|u_k - u^*_k\|_2^2 \tag{6}$$

where x^*_{k+1} and u^*_k are the Differentiable MPC policy's predicted states and actions, respectively, x_{k+1} and u_k are from the historical data, M is the batch size, and λ weights the action imitation error relative to the state prediction error.

It penalizes deviations in both predicted next states ($\mathcal{L}_{state} = \frac{1}{M}\sum_k^M \|x_{k+1} - x_{k+1}^*\|_2^2$) and chosen control actions ($\mathcal{L}_{action} = \frac{1}{M}\sum_k^M \|u_k - u_k^*\|_2^2$).

Differentiable MPC supports box constraints on control inputs: $P_{HP}^{min} \leq P_{HP,k} \leq P_{HP}^{max}$ and $P_{BH}^{min} \leq P_{BH,k} \leq P_{BH}^{max}$.

B.2. Cost Parameter Calibration Details

To move beyond direct imitation and allow for user-defined control objectives, we introduce a non-quadratic reward function to guide the learning of the LQR cost parameters $\theta_{cost} = \{O, R_1, R_2\}$. This reward function evaluates the Differentiable MPC policy's planned trajectory over a lookahead horizon $L = 24$ hours: $x_{k+1}^*, \ldots, x_{k+L}^*$ and $u_k^*, \ldots, u_{k+L-1}^*$. The reward function is defined as:

$$R_k = -w_d \max_{\ell \in [0, L-1]} (P_{HP,k+\ell}^* + P_{BH,k+\ell}^*) \\ - \Delta t \sum_{\ell=0}^{L-1} \left[w_e (P_{HP,k+\ell}^* + P_{BH,k+\ell}^*) + w_c \left| x_{k+\ell+1}^* - x_{target,k+\ell+1} \right| \right] \quad (7)$$

where:

- w_d: Peak power demand price (\$/kW), set to \$0.8/kW. This term considers the maximum of the total power (sum of heat pump and backup heat power) over the lookahead horizon.
- w_e: Electrical energy price (\$/kWh), set to \$0.15/kWh, applied to the total power consumed.
- w_c: Thermal discomfort price (\$/°C/h), set to \$0.2/°C/h, the primary user-defined parameter.
- $P_{HP,k+\ell}^* + P_{BH,k+\ell}^*$ is the total power input at future time step $k + \ell$ in the planned trajectory from $u_{k+\ell}^*$.
- Δt: Duration of a single control interval (e.g., in hours), ensuring consistent units for energy calculation.

Online Calibration Procedure: During online calibration, both system dynamics parameters θ_{state} and cost parameters θ_{cost} are updated.

1. θ_{state} are optimized by minimizing the state prediction loss $\mathcal{L}_{state} = \frac{1}{M}\sum_k^M \|x_{k+1} - x_{k+1}^*\|_2^2$ based on interactions with the real environment. This is a non-convex least squares problem.
2. θ_{cost} are adjusted by maximizing the reward R_k (Eq. (7)) using gradient ascent. The gradients are computed with respect to θ_{cost} through the Differentiable MPC policy.

This ensures the quadratic LQR cost function (used internally by Differentiable MPC) gradually aligns with the objectives of the non-quadratic reward function.

Initial Calibration Procedure: This offline process calibrates θ_{cost} using initial system conditions before deployment or when cost coefficients (w_d, w_e, w_c) change.

1. Using the current (or initial) θ_{state}, the Differentiable MPC policy is solved to obtain a planned trajectory ($x_{k+1}^*, \ldots, x_{k+L}^*$ and $u_k^*, \ldots, u_{k+L-1}^*$).
2. The cumulative reward (Eq. (7)) for this trajectory is evaluated.
3. Gradients of this cumulative reward with respect to θ_{cost} are computed, and θ_{cost} is updated via gradient ascent, with gradients flowing through the MPC's internal model and decision process.
4. Steps 1-3 are repeated iteratively until θ_{cost} converges or a set number of iterations is reached.

During actual deployment after initial calibration, θ_{state} continues to be updated online, but θ_{cost} remains fixed unless a recalibration is triggered by changes in w_d, w_e, or w_c.

C. Experiments

C.1. Details of Imitation Learning

Our approach to learning a stable differentiable MPC policy involves two key components: (a) learning the system dynamics, represented by the 2R1C thermal model in (B.1), and (b) learning the desired control behavior, represented by the cost function in (2). We systematically tested various hyperparameters to optimize both state and action losses, as shown in Figure 3. While the loss trajectories converge, the resulting system dynamics parameters often lack physical plausibility, and the learned behavior fails to match the existing controller's performance. This dual challenge highlights the limitations of relying solely on loss minimization and underscores the need for additional constraints to ensure both physically meaningful dynamics and behavior alignment.

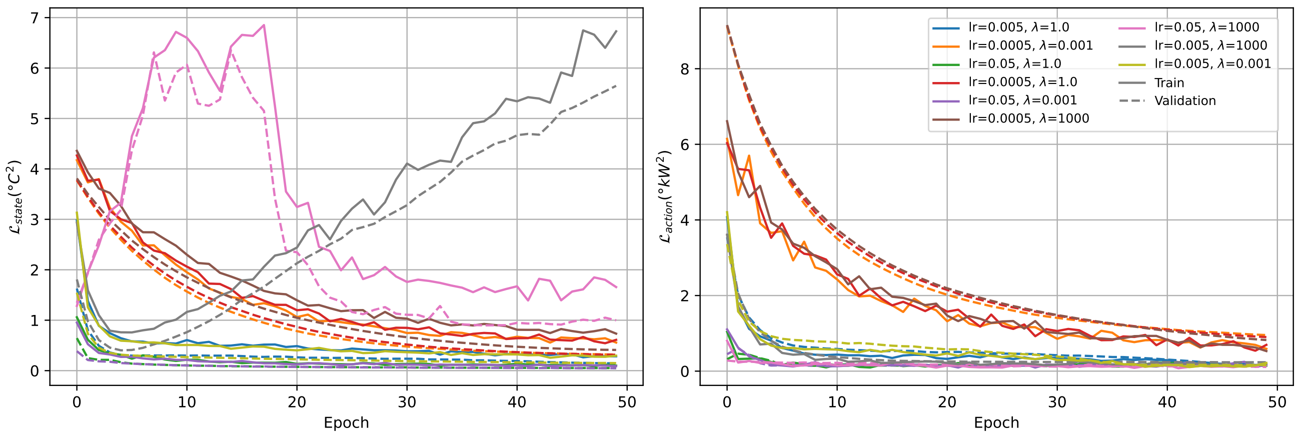

Figure 3. State and Action loss curves for different hyperparameter configurations. Dashed lines are from validation set, while solid lines are from training data.

To account for the best scenarios, for the analysis on state dynamics, we take the results coming from the lowest validation state loss (lr=0.05, λ=0.001) that reached the lowest state loss of MSE 0.0479 at epoch 48, note that this value is even lower than the overengineered model used in the prior MPC study. For the analysis on the behavior, we look at the hyperparameters that minimized action loss, ensuring we chose the agent closely mimicked the existing controller coming from lr=0.05 and $\lambda = 1000$, achieving a minimum MSE of 0.111 at epoch 49. However, we observe that the parameters fitted are actually far from reality, it ends up having an unusually high C, meaning really large capacitances, and a negative α, meaning solar irradiance causes cooling.

Imitation learning with varying hyperparameters. Figure 4 shows the converged parameter values for different hyperparameters, as one can see, most of them converge to unrealistic physical parameters such as having an unusually high C, meaning really large thermal capacitances, and a negative α, meaning solar irradiance causes cooling.

C.2. Mimicking the existing controller.

We first analyze the imitation learning agent's performance under the assumption of a perfect system dynamics model, where each action results in the predicted state transition (i.e., $\mathcal{L}_{state} = 0$). To ensure the agent closely mimicked the existing controller, we selected hyperparameters minimizing action loss. The best-performing agent, trained with a $lr = 0.05$ and $\lambda = 1000$, achieved a minimum MSE of 0.111 (in kW^2) at epoch 49. This analysis mirrors behavior cloning, with minimal state distribution shift, as the agent was tested on its training data.

Figure 5 shows the imitation learning agent's performance in this ideal setting, alongside energy and comfort metrics. PPD quantifies user discomfort (ASHRAE, 2004), while energy consumption is derived from power measurements. The imitation learning agent results in significantly higher PPD, with indoor temperatures reaching 28°C, indicating considerable discomfort and poor mimicry of the training data (blue curves). Thus, despite being trained to match the existing controller's actions, the imitation learning agent exhibits unstable behavior and fails to replicate the baseline controller's performance. This discrepancy arises from two factors: (1) Model mismatch and environmental disturbances: even in controlled settings, modeling errors and unmodeled disturbances prevent perfect behavioral cloning, limiting the agent's generalization; and (2)

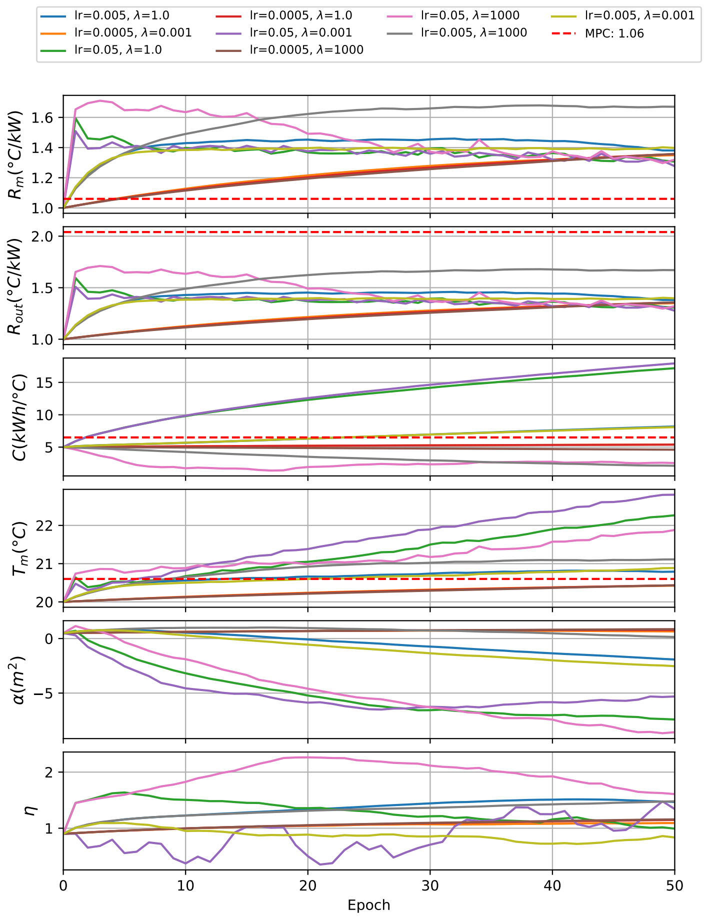

Figure 4. Parameter trajectories across epochs for different hyperparameters.

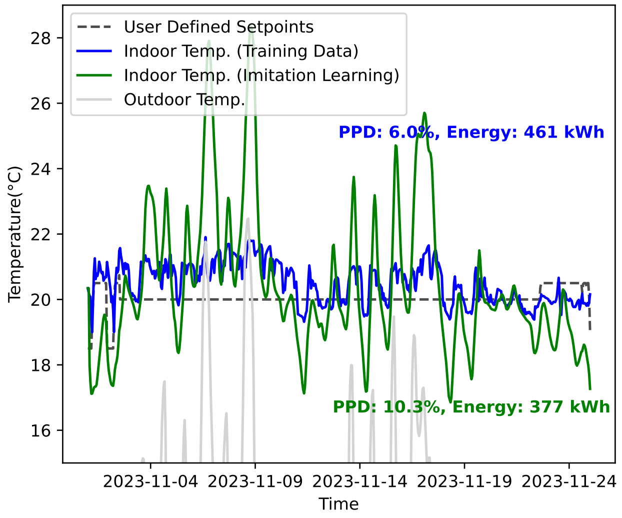

Figure 5. Imitation learning agent being tested on the data it was trained on, assuming deterministic state transitions. It indicates that imitation learning fails to mimic the behavior of the existing controller.

Structural misalignment: the baseline controller uses a reactive PID strategy, while the imitation learning agent operates within a differentiable MPC framework with a 24-hour lookahead. This fundamental difference makes it challenging for the imitation learning agent to mimic the baseline while retaining the benefits of optimal control.