Machine Learning (2024) 113:5203–5234
https://doi.org/10.1007/s10994-023-06460-4

Multi-agent reinforcement learning for fast-timescale demand response of residential loads

Vincent Mai[1,2] · Philippe Maisonneuve[2,3,4] · Tianyu Zhang[1,2] · Hadi Nekoei[1,2] · Liam Paull[1,2] · Antoine Lesage-Landry[2,3,4]

Received: 25 April 2023 / Revised: 10 August 2023 / Accepted: 17 October 2023 /
Published online: 28 November 2023

Abstract

To integrate high amounts of renewable energy resources, electrical power grids must be able to cope with high amplitude, fast timescale variations in power generation. Frequency regulation through demand response has the potential to coordinate temporally flexible loads, such as air conditioners, to counteract these variations. Existing approaches for discrete control with dynamic constraints struggle to provide satisfactory performance for fast timescale action selection with hundreds of agents. We propose a decentralized agent trained with multi-agent proximal policy optimization with localized communication. We explore two communication frameworks: hand-engineered, or learned through targeted multi-agent communication. The resulting policies perform well and robustly for frequency regulation, and scale seamlessly to arbitrary numbers of houses for constant processing times.

Keywords Multi-agent reinforcement learning · Demand response · Power systems · Renewable integration · Communication · Coordination

1 Introduction

To achieve the United Nations' climate change target of limiting global warming at +1.5°C, global electricity generation must transition from fossil fuels to renewable energy sources such as wind turbines and solar panels. In 2019, according to the International Energy Agency, electricity and heat production accounted for 40% of global emissions (Agency, 2021), as 64% of it is generated from burning fossil fuel (Agency, 2022). The electricity sector must thus move from a conventional, fuel-burning paradigm to a renewable, natural phenomenon-based generation, e.g., wind turbines and solar photovoltaics. Renewable energy generation is subject to short-term, high-amplitude variations, referred to as intermittency. As an example, a cloud passing will lead to a sudden drop in the solar-based generation, followed

Editors: Emma Brunskill, Minmin Chen, Omer Gottesman, Lihong Li, Yuxi Li, Yao Liu, Zonging Lu, Niranjani Prasad, Zhiwei Qin, Csaba Szepesvari, Matthew Taylor.

Extended author information available on the last page of the article

by a sharp increase when the sky becomes clear again. These changes can happen at the scale of a few seconds, and create major challenges for power grid operators: to ensure the stability of the electric grid, a near-perfect balance between the power demand and the generation is critical (Kundur, 2007). In other words, power generation and consumption must be equal at all times. Hence, trading a constant, deterministic generation for an intermittent, uncertain one exacerbates the need for power balancing. At the second timescale, this balancing task is referred to as frequency regulation (Bevrani et al., 2010; Taylor et al., 2016).

On the power generation side, solutions such as excess energy storage in batteries or support from fossil fuel plants require large investments and are not renewable respectively. Alternatively, demand response programs (Siano, 2014) can be introduced to mitigate renewable intermittency (Taylor et al., 2016). The demand response approach aims at adjusting the power demand to meet the supply by coordinating loads temporally. These loads must be flexible, i.e., capable of modulating their consumption while fulfilling their own purpose. This does not apply to, for example, computer monitors, which must be fully powered when they are in use. Thermostatic loads, such as heating, air conditioning or water heaters, are instead ideal candidates: they do not need to be turned on at all times, as long as the temperature of the air/water is within the user's preference range (Callaway, 2009). They are also widely deployed and they represent a significant part of global power consumption (Agency, 2018; Mathieu et al., 2012). The increasing adoption of electric vehicles is another important possible source of flexibility. The frequency regulation objective differs from the peak-shaving problem, where the objective is load shifting over, e.g., a day, to reduce the grid's power consumption peak. Instead, we aim at leveraging the loads' flexibility to balance out high-frequency variations in power generation.

In this paper, we focus on the task of fast timescale demand response for frequency regulation using *residential air conditioners*. This presents several physical and algorithmic constraints: (1) air conditioners are **discretely** powered, i.e., ON or OFF, which limits the control flexibility; (2) they are subject to hardware **dynamic constraints** such as lockout: once turned OFF, they must wait some time before being allowed to turn back ON to protect the compressor; (3) as the context is residential, privacy is important and **communications should be limited**; (4) to provide enough power flexibility to the grid, a large aggregation of loads must be considered: the method must be **scalable**; (5) for easier implementation, the control should also be **decentralized** with **localized communications**; (6) the decisions must be taken at a **few seconds timescale**; and finally (7) the control algorithm should be able to **cope with uncertainty** in the future regulation signal.

These constraints impede the deployment of classical methods. Greedy algorithms are centralized and have difficulty accounting for long-term dynamic constraints (Lesage-Landry et al., 2021). Standard model predictive control is also centralized, and even decentralized versions solve a multi-period combinatorial optimization problem that does not scale with the number of agents (Dong et al., 2018; Chen et al., 2020). We propose to tackle this problem by using multi-agent reinforcement learning (MARL) to learn decentralized and scalable policies (4) with discrete and constrained control (1, 2) and limited and localized communications (3, 5). Once learned, these policies can take the best decisions in real time (6) based on expected value over uncertainty (7). As this problem combines the most important current challenges of MARL, i.e., communication, long-term credit assignment, coordination, and scalability (Gronauer & Diepold, 2021), it is also interesting for MARL algorithm research. We train our agents with Multi-Agent Proximal Policy Optimization (MA-PPO) (Yu et al., 2021) with Centralized Training, Decentralized Execution (CT-DE) (Kraemer and Banerjee 2016). Two local communication frameworks are tested—hand-engineered and learned—and both outperform the baselines. Our main contributions are threefold:

- An open source, multi-agent environment[1] simulating the real-world problem of frequency regulation through demand response at the second timescale. The simulator is compatible with the OpenAI Gym (Brockman et al., 2016) framework.
- Two decentralized, fast-responding agents[1] trained by MA-PPO. The first one has a hand-engineered communication strategy, while the second one learns what data to share through Targeted Multi-Agent Communication (TarMAC) (Das et al., 2019). Both outperform baselines on two-day simulations.
- An in-depth analysis of the dynamics, communications, scalability and robustness of the trained agents.

In the next section, we describe prior work in the field of demand response and MARL. In Sect. 3, we describe the environment and formulate the problem. The classical and learning-based methods are described in Sect. 4. Finally, Sect. 5 presents the experimental results and analyses of the agents' performance, dynamics, robustness, and scalability.

2 Related works

Frequency regulation through demand response is commonly tackled by model predictive control (MPC) (Wu et al., 2018; Lee et al., 2015; Olama et al., 2018; Dong et al., 2018; Maasoumy et al., 2014; Mathieu et al., 2012), where the best action is chosen based on trajectory prediction over a given horizon, sometimes combined with machine learning (Dusparic et al., 2013; Lauro et al., 2015; Ahmadiahangar et al., 2019). Apart from Liu and Shi (2015), these works do not consider short-term dynamic constraints such as lockout. MPC approaches rely on mixed-integer programming, which does not scale sustainably with higher numbers of agents, preventing control at fast timescales. Moreover, these works generally require a centralized entity to access residences' data, leading to confidentiality issues. An alternating direction of multipliers method (ADMM)-based distributed MPC approach was proposed in Chen et al. (2020). This approach did not consider the lockout constraint and is not compatible with fast timescale decision-making as it requires multiple centralized communication rounds at each time step in addition to solving several optimization problems and converting continuous setpoints to binary actions.

To tackle these problems, online optimization (OO) approaches (Lesage-Landry & Taylor, 2018; Zhou et al., 2019) have been used because of their high computational efficiency and scalability. In particular, Lesage-Landry et al. (2021) deploys OO for frequency regulation with binary control settings as is the case for air conditioning (AC) units. However, these methods rely on greedy optimization and their lack of foresight leads to limited performance when facing dynamic constraints. Reinforcement learning (RL) methods have been developed to address the longer timescale power balance problems such as peak shaving through demand response (Aladdin et al., 2020) or coordination of loads and generators (Roesch et al., 2020; Yang et al., 2019). The CityLearn environment (Vazquez-Canteli et al., 2020) proposes a standard environment for multi-agent RL (MARL) for demand response, upon which are developed methods such as Pigott et al. (2021) to regulate the voltage magnitude in distribution networks using smart inverters and intelligent energy

[1] The code is hosted on https://github.com/ALLabMTL/marl-demandresponse.

storage management, and Vazquez-Canteli et al. (2020) for load shaping of grid-interactive connected buildings. The AlphaBuilding ResCommunity environment (Wang et al., 2021) then implements detailed thermal models. Both CityLearn and AlphaBuilding ResCommunity, however, consider longer timescale control, which makes them inadequate for high-frequency regulation and removes the ACs' lockout and binary constraints. The Power-Gridworld (Biagioni et al., 2021) environment, a more flexible alternative to CityLearn, allows fast-timescale simulation but does not provide a detailed thermal model of loads, options for lockout or binary control, or classical baseline approaches to compare with. High-frequency regulation has been addressed by MARL, but only on the power generation side (Xi et al., 2018). We are unaware of any example in the literature deploying MARL for frequency regulation with demand response, with second-timescale control and flexible binary loads such as ACs which are subject to hardware dynamic constraints like a lockout.

More generally, MARL has been developed for collaboration both in virtual environments such as Dota 2 (OpenAI et al., 2019), Hide and Seek (Baker et al., 2020) or Fuchs et al. (2021), and in real-world environments such as traffic light control (Wei et al., 2019), single-house energy management (Ahrarinouri et al., 2021), path-finding (Sartoretti et al., 2019), active voltage control (Wang et al., 2022) or ride-sharing (Qin et al., 2022). MARL problems pose several additional challenges to the RL settings (Gronauer & Diepold, 2021), such as the non-stationarity of the environment, the need to learn coordination and communication, or the scaling of the training and deployment. Multi-agent adaptations of known RL algorithms, such as online PPO (Schulman et al., 2017; Yu et al., 2021), or offline DDPG (Lillicrap et al., 2019; Lowe et al., 2020; Mnih et al., 2015), have led to strong performance in many problems. However, some particular problems, such as the ones requiring communication with large numbers of agents, need specialized algorithms (Jiang and Lu, 2018). MARL communication is an active field of research. Different strategies exist: MAIC models the other agents to maximize the mutual information in the messages (Yuan et al., 2022). TarMAC (Das et al., 2019), instead, uses an attention mechanism to aggregate messages for each agent based on their importance. MASIA agents (Guan et al., 2023) instead share a common aggregator, and then use a focus network to only extract the relevant contents.

A 2-page extended abstract was previously presented for this work (Mai et al., 2023). This article presents the complete work on MARL for demand response of residential loads; it contains the full formulation of the problem, necessary information about its implementation, additional experimental results and their in-depth analysis.

3 Problem formulation

3.1 Environment

The environment is a simulation of an aggregation of N houses, each equipped with a single AC unit. Thermostatic loads modeled as multi-zone units and equipped with more than a single AC (Amin et al., 2020) is a topic for future work. The environment model is updated every 4 s. More details about the environment are given in Appendix C. A notation table is provided in Appendix A.

3.1.1 Outdoors temperature

The outdoor temperature $T_{o,t}$ is assumed to be the same for every house, i.e., they are co-located in the same geographical region, and is simulated as sinusoidal with a one-day period. Unless otherwise specified, the maximum temperature of 34 °C is reached at 6 pm and the minimal temperature of 28 °C at 6 am. $T_{o,t}$ is thus always above the target indoor temperature T_T of 20 °C, so that every household can offer its flexibility to the grid. Note that demand response for frequency regulation can easily be extended to heat-pumps, which provide flexibility in situations where heating is required due to low temperatures. Flexibility is thus primarily limited when the outdoor temperature aligns closely with the desired temperature, eliminating the need for temperature control. As this scenario coincides with reduced energy consumption for households, system operators must then rely on other sources of flexibility like battery energy storage, electric vehicles, or fast-ramping conventional power plants.

3.1.2 House thermal model

Each house $i = 1, 2, \dots, N$ is simulated using a second-order model based on Gridlab-D's Residential module user's guide (Betelle Memorial Institute, 2022). At time t, the indoor air temperature $T^i_{h,t}$ and the mass temperature $T^i_{m,t}$ are updated given the house characteristics θ^i_T (wall conductance U^i_h, thermal mass C^i_m, air thermal mass C^i_h and mass surface conductance H^i_m), the outdoor temperature $T_{o,t}$, and the heat $Q^i_{a,t}$ removed by the AC. By default, the thermal characteristics are the same for each house and model a 100 square meter, 1-floor house with standard isolation. During training and deployment, the initial mass and air temperatures are set by adding a positive random noise over the target temperature. Although it is not used by default, the solar gain $Q_{s,t}$ can also be added to the simulation, as seen in Appendix C.1.1.

3.1.3 Air conditioners

Once again based on Gridlab-D's guide (Betelle Memorial Institute, 2022), air conditioner i's heat removal capacity $Q^i_{a,t}$ and power consumption $P^i_{a,t}$ are simulated based on the AC characteristics θ^i_a, which include their cooling capacity K^i_a, their coefficient of performance COP^i_a and the latent cooling fraction L^i_a. The model and parameters are also described in Appendix C.2. Additionally, a hard dynamic constraint is set to protect the compressor: after being turned OFF, it needs to wait a given amount of time before being allowed to turn ON again (Zhang et al., 2013). This constraint is referred to as the lockout. By default, the lockout duration $l^i_{\max}$ is set to 40 s.

3.1.4 Regulation signal

The power system operator sends to the aggregator a signal ρ_t, which covers the complete aggregated load consumption: the systems we cannot control such as computers, washing machines, or lights, and the flexible power consumption, in our case, the ACs. Let, $\rho_t = D_{o,t} + s_t$ where $D_{o,t}$ is the power demand for the non-controllable loads and s_t is the objective aggregated AC power consumption, i.e., the flexible load. We define $D_{a,t}$ as the power needed by the ACs to satisfy their thermal objectives, i.e., to keep the temperature around the target. To focus on the high-frequency variations of the power generation, we assume that s_t is well behaved at low frequencies, i.e., its mean in the 5 min scale is $D_{a,t}$. A 0-mean, high-frequency variation $\delta_{s,t}$ is added to represent renewable intermittency the aggregator wants to mitigate. We model the regulation signal as $s_t = D_{a,t} + \delta_{s,t}$.

The aggregation flexible power consumption is the sum of all of the ACs' consumption: $P_t = \sum_i^N P_{a,t}^i$. The objective is to coordinate the ACs in the aggregation so that P_t tracks s_t, while keeping the indoors temperature as close as possible to the target for each house.

Base signal. To compute the average needed power $D_{a,t}$, we created a dataset of the average power needed over a 5-minute period by a bang-bang controller without lockout – which is optimal for temperature – for all combinations of discrete sets of the relevant parameters. At each time step, we interpolate the average power demand of each AC from this dataset and sum them to compute $D_{a,t}$. More details are available in Appendix C.3. In practice, the base signal would be estimated or obtained from historical data. The aggregator would then consider its value when committing to track a signal s_t. This ensures that the required power adjustment is enough to maintain the houses at acceptable temperatures while providing flexibility to the grid.

Modelling high-frequency variations. The high-frequency variation $\delta_{s,t}$ is modelled with 1-D Perlin noise (Lagae et al., 2010), a smooth, procedurally generated 0-mean noise. The Perlin noise produces $\delta_{p,t} \in [-1, 1]$, and we have $\delta_{s,t} = D_{a,t}\beta_p\delta_{p,t}$ where β_p is an amplitude parameter set to 0.9. Our Perlin noise is defined by 5 octaves and 5 octave steps per period of 400 s; it thus is the sum of noises with periods of 80, 40, 20, 10 and 5 s. More details are given in Appendix C.3.2.

3.1.5 Communication between agents

To achieve coordination between agents, they must be able to communicate. For the agent implementation to be decentralized, flexible, and privacy-preserving, we consider limited and localized communications. This enables, for example, devices communicating with simple radio-frequency emitters, without the need for any further infrastructure. As such, we limit the communication to a number N_c of neighbours. This is in line with the low-deployment investment argument for using demand response for frequency regulation.

3.2 Decentralized partially observable Markov decision process

In this section, we formalize the above environment as a decentralized, partially observable Markov decision process (Dec-POMDP) characterized by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, \mathcal{R}, \gamma \rangle$. Let $\mathcal{S}$ be the global state, $\mathcal{A} = \prod_{i=1}^N \mathcal{A}^i$ the joint action space, and $\mathcal{O} = \prod_{i=1}^N \mathcal{O}^i$ the joint observation space. $\mathcal{O}^i$ partially observes $\mathcal{S}$. $\mathcal{P}$ describes the environment's transition probabilities, $\mathcal{R}$ the reward function for each agent and γ the discount parameter.

3.2.1 State, transition probabilities and actions

The state of the environment $X \in \mathcal{S}$ and its transition probabilities $\mathcal{P}$ are unknown to the agent. They are simulated by the environment dynamics described in Sect. 3.1. Each agent i's action $a_t^i \in \mathcal{A}^i$ is a binary decision to control the AC status. If the remaining lockout time l_t^i is above zero, the ON action will be ignored by the environment. In practice, a backup controller within the AC would prevent the ON decision from being implemented.

3.2.2 Observations and communications

By default, agent i receives observation $o_t^i = \{T_{h,t}^i, T_{m,t}^i, T_T^i, \omega_t^i, l_t^i, s_t/N, P_t/N\}$ at time step t, where $T_{h,t}^i$, $T_{m,t}^i$ and T_T^i are the indoor air, mass, and target temperatures, ω_t^i is the ON or OFF status of the AC, l_t^i is its remaining lockout time, s_t/N is the per-agent regulation signal and P_t/N is the per-agent total consumption of the aggregation.

Each agent i communicates with its N_c neighbours. The messages' sizes are not hard limited but should be small, and their contents are not constrained. We define the set of all of agent i's N_c neighbours as M^i. By default, we organize the agents in a 1-dimensional structure: $M^i = \{i - \lfloor N_c/2 \rfloor, i - \lfloor N_c/2 \rfloor + 1, \dots, i, \dots, i + \lfloor N_c/2 \rfloor - 1, i + \lfloor N_c/2 \rfloor\} \backslash \{i\}$.

3.2.3 Reward

For each agent i, reward r_t^i is computed as the weighted sum of the penalties due to its air temperature difference with the target, which is unique to the agent, and to signal tracking, which is common across all agents. This scenario is therefore cooperative with individual constraints. We normalize the reward with $\alpha_{\text{temp}} = 1$ and $\alpha_{\text{sig}} = 3 \times 10^{-7}$: a 0.5 °C error is penalized as much as a 912 W per-agent error (each agent consumes 6000 W).

$$r_t^i = -\left(\alpha_{\text{temp}} \left(T_{h,t}^i - T_{T,t}^i \right)^2 + \alpha_{\text{sig}} \left(\frac{P_t - s_t}{N} \right)^2 \right)$$

4 Classical and learning-based algorithms

4.1 Classical baselines

To the best of our knowledge, there is no classical baseline that performs well under all the constraints enumerated in Sect. 1. However, simple algorithms can optimize selected objectives, and we use them as baselines for the results of the MARL agent. The important characteristics of the different baselines are summarized in Table 1.

4.1.1 Bang-bang controller

The bang-bang controller (BBC) turns the AC ON when the air temperature $T_{h,t}^i$ is higher than the target T_T^i, and OFF when it is lower. This is a decentralized algorithm, which does not consider demand response but near-optimally controls the temperature. When the lockout duration $l_{\max}^i$ is 0, the BBC optimally controls the temperature, but does not account for

Table 1 Comparison of the classical baselines

	Demand response	Centralized	Handles lockout	Scalable
BBC	No	Yes	Yes	Yes
Greedy myopic	Yes	No	No	Yes
MPC	Yes	No	Yes	No

the signal. As the base signal $s_{0,t}$ is computed to allow optimal temperature control, BBC's signal tracking error is mainly due to the high-frequency variations of the signal.

4.1.2 Greedy myopic

The greedy controller is a centralized algorithm that solves a knapsack problem (Dantzig, 1957) where the size of the collection is the regulation signal, the weight of each AC is its consumption $P^i_{a,t}$, and its value is the temperature difference $T^i_{h,t} - T^i_T$. At each time step, ACs are chosen based on a value priority computed by $(T^i_{h,t} - T^i_T)/P^i_{a,t}$, until the aggregation's consumption P_t is higher than the regulation signal s_t. As it does not plan for the future, the greedy myopic approach quickly runs out of available ACs as most of them are in lockout. However, with a 0-lockout duration $l^i_{\max}$, it is near-optimal to track the signal s_t, and controls the temperature in second priority. We implement the greedy myopic approach as it is better adapted to these settings than the OO approach described in Sect. 2. Indeed, OO only uses past state information and must be implemented in a strictly online fashion. Both frameworks are myopic, and struggle similarly with the lockout constraint.

4.1.3 Model predictive control

Model predictive control, or MPC, is in its nominal form a centralized algorithm modeling the environment and identifying the actions which will lead to the highest sum of rewards over a time horizon of H time steps. As the signal is stochastic, MPC assumes a constant future signal over horizon H, and optimally solves the trajectory with lockout. However, because it is a large-scale combinatorial optimization problem, it scales poorly with the number of agents N and with a horizon H. In the best case the complexity is polynomial, but it is exponential in the worst case. As a result, we were not able to run the MPC for more than 10 agents for $H = 60$ s, and had to increase the time step between each action to 12 s. More details are provided in Appendix D.1.

4.2 Learning-based methods

We deploy two algorithms using deep reinforcement learning, namely MA-DQN and MA-PPO, both using the CT-DE paradigm. These algorithms are online and learn from experimenting with the environment: as this problem is not solved, there is no existing trajectory the agents could learn from in an offline learning setting. While MA-DQN only uses hand-engineered communications, MA-PPO was implemented with two communications paradigms: hand-engineered and learned. Details about the architectures and hyperparameters are provided in Appendix 2.

4.2.1 Centralized training, decentralized execution

The CT-DE paradigm (Kraemer & Banerjee, 2016) assumes that information is shared during the training of the agents, while they execute actions only based on their decentralized observations, maintaining privacy during deployment. This reduces the non-stationarity of the environment (Gronauer & Diepold, 2021) and stabilizes the training. In our case, all agents are homogeneous, which allows the use of parameter sharing (Gupta et al., 2017). As such, all ACs are controlled by identical instances of the same policy trained from the shared experience of all agents.

4.2.2 MA-DQN

Multi-agent Deep Q-Network (MA-DQN) is the CT-DE adaptation of DQN (Mnih et al., 2015), an off-policy algorithm made for discrete action spaces. A DQN agent mainly consists of a Q-network predicting the Q-value of action-observation pairs $(a^i_t, \tilde{O}^i_t)$ for every possible a^i_t. During training, at time step t, the transition $\Theta^i_t = \{\tilde{O}^i_t, a^i_t, r^i_t, \tilde{O}^i_{t+1}\}$ of every agent is recorded in a common replay buffer. This replay buffer is sampled to train the Q-network to predict $Q(a^i_t, \tilde{O}^i_t)$ supervised with target $T(a^i_t, \tilde{O}^i_t)$ according to Bellman's optimality equation:

$$T(a^i_t, \tilde{O}^i_t) = r^i_t + \gamma \max_a Q(a, \tilde{O}^i_{t+1}).$$

Actions are selected as a^i_t with maximal predicted Q-value given an input $\tilde{O}^i_t$. ϵ-greedy exploration is added during training.

4.2.3 MA-PPO

Multi-agent Proximal Policy Optimization (MA-PPO) (Yu et al., 2021) is the CT-DE adaptation of clipped PPO (Schulman et al., 2017), an on-policy, policy-gradient algorithm. The agent jointly learns a policy $\pi_\theta(a^i_t|\tilde{O}^i_t)$, also called an actor, and a value function $V_\phi(\tilde{O}^i_t)$, also called a critic. At each epoch, the policy is fixed and the transitions $\Theta^i_t = \{\tilde{O}^i_t, a^i_t, \pi_{\theta_t}(a^i_t|\tilde{O}^i_t), r^i_t\}$ for all agents are recorded together for one or several episodes of length H. For each Θ^i_t, a return $G^i_t = \sum_{\tau=0}^{H-t} \gamma^\tau r^i_{t+\tau}$ is computed based on future experience. As the environment is actually infinite-horizon, the return G^i_t is corrected using partial episode bootstrap (Pardo et al., 2018). Then, the new policy parameters θ_{t+1} are trained over the stored memory to optimize the clipped PPO objective $\mathcal{L}(\tilde{O}^i_t, a^i_t, \theta_{t+1}, \theta_t)$, maximizing the advantage $A^{\pi_{\theta_t}}(\tilde{O}^i_t, a^i_t) = G^i_t - V_\phi(\tilde{O}^i_t)$ under the constraint of proximity around the previous policy. The critic parameters ϕ are then trained so that $V_\phi(\tilde{O}^i_t)$ predicts the return G^i_t. The memory is erased and a new epoch starts.

Exploration is handled by the inherent stochasticity of the policy. In the CT-DE setting, V_ϕ, which is only used during training, is given additional information about the states of other agents.

4.2.4 Communications

4.2.4.1 Hand-engineered communications For MA-DQN and the hand-engineered MA-PPO, the messages are designed based on the state of each agent, effectively providing a wider observability of the general state. Agent j's message $m_{j,t}$ contains the current difference between its air and target temperatures $T^j_{h,t} - T^j_T$, its remaining lockout time l^j_t, and its current status ω^j_t. The messages $\{m^i_{j,t}\}_{\forall j \in M_i}$ from agents $j \in M^i$ are concatenated with the observations o^i_t to create the input $\tilde{O}^i_t$ of the neural networks. Message $m^i_{j,t}$ from agent j to agent i is at a fixed place in the $\tilde{O}^i_t$ vector based on its relative position $i - j$. MA-PPO with hand-engineered communication will be referred to as MA-PPO-HE.

4.2.4.2 Targeted multi-agent communication To allow agents to learn to communicate, we implement TarMAC (Das et al., 2019) in MA-PPO. TarMAC is an attention-based targeted communication algorithm where each agent outputs a key, a message and a query. The key is sent along with the message to the other agents, which then multiply it with their query to compute the attention they give to the message. All messages are then aggregated using the attention as a weight. The three modules – key, message, query – are trained. TarMAC allows more flexibility to the agents: it does not restrict the contents of the communication, and it allows agents to communicate with a different number of houses than they were communicating with during training. More details are available in Appendix D.2.1. We refer to this version as TarMAC-PPO.

4.2.4.3 No communication It is also possible to train agents without communication. In this case, it only observes o_t^i. This agent is referred to as MA-PPO-NC.

4.2.5 Agent training

The learning agents were trained on environments with $N_{\text{tr}} = \{10, 20, 50\}$ houses and communicating with $N_{\text{c}_{\text{tr}}} = \{9, 19, 49\}$ other agents. We trained every agent on 16 different seeds: 4 for environment and 4 for network initialization. They were trained on 3286800 time steps, equivalent to 152 days, divided in 200 episodes. Each episode is initialized with each house having a temperature higher than the target, sampled from the absolute value of a 0-mean Gaussian distribution with $\sigma = 5°\text{C}$. We tuned the hyperparameters through a grid search, as shown in Appendix D. The contribution of this paper is to demonstrate that learning-based methods can lead to high performance on the problem of high frequency regulation. We therefore do not compile statistics over the trained agents; instead, for each situation, we select the two best agents over the seeds based on test return, and report the best score from these two on the benchmark environment.

5 Results and analysis

5.1 Metrics of performance

We deploy the agents on a benchmark environment with N_{de} houses on trajectories of 43200 steps, i.e., two full days. The primary metric used to assess their performance is the per-agent root mean square error (RMSE) between the regulation signal s_t and aggregated power consumption P_t. We also measure the temperature RMSEs – one for all agents, one for the maximal temperature error of the aggregation – to ensure effective thermal control. In practice, a RMSE of a few tenth of a °C is considered acceptable for residential house temperature control. Every house's temperature is initialized differently, so we start computing the RMSE when the temperature is controlled, after 5000 steps. For context, a single AC consumes 6000 W when turned ON. Due to the MPC's computing time, its performance is evaluated differently, as explained in Appendix D.1. Unless mentioned otherwise, the results are the mean and standard deviation over 10 environmental seeds.

Table 2 Performance of the different agents, computed over 10 environment seeds

Per-agent		$N_{\mathrm{de}} = 10$			$N_{\mathrm{de}} = 50$			$N_{\mathrm{de}} = 250$			$N_{\mathrm{de}} = 1000$		
		Signal	T	Max T	Signal	T	Max T	Signal	T	Max T	Signal	T	Max T
RMSE		(W)	(° C)	(° C)	(W)	(° C)	(° C)	(W)	(° C)	(° C)	(W)	(° C)	(° C)
No l.o	Greedy	194 ± 1	0.04	0.06	70 ± 1	0.03	0.05	63 ± 1	0.03	0.052	63 ± 1	0.03	0.05
	BBC	806 ± 147	0.02	0.03	392 ± 50	0.02	0.04	310 ± 11	0.02	0.03	272 ± 12	0.02	0.03
40 s l.o	Greedy	2668 ± 14	0.87	0.93	3166 ± 12	1.09	1.15	3313 ± 12	1.16	1.22	3369 ± 15	1.18	1.24
	BBC	830 ± 207	0.05	0.09	426 ± 63	0.05	0.10	318 ± 7	0.05	0.10	296 ± 4	0.05	0.10
	MPC	344 ± 96	0.07	0.12	–	–	–	–	–	–	–	–	–
	MA-DQN	541 ± 86	0.05	0.09	321 ± 24	0.05	0.10	246 ± 8	0.05	0.11	234 ± 4	0.05	0.12
	MA-PPO-HE	253 ± 1	0.04	0.08	161 ± 8	0.04	0.08	127 ± 2	0.04	0.11	122 ± 3	0.05	0.13
	TarMAC-PPO	**247 ± 3**	**0.04**	**0.07**	**158 ± 2**	**0.04**	**0.09**	**115 ± 1**	**0.05**	**0.13**	**101 ± 2**	**0.05**	**0.14**
	MA-PPO-NC	434 ± 2	0.06	0.08	215 ± 1	0.06	0.14	132 ± 1	0.06	0.16	107 ± 1	0.06	0.17

In bold, the best performing agent based on the signal RMSE

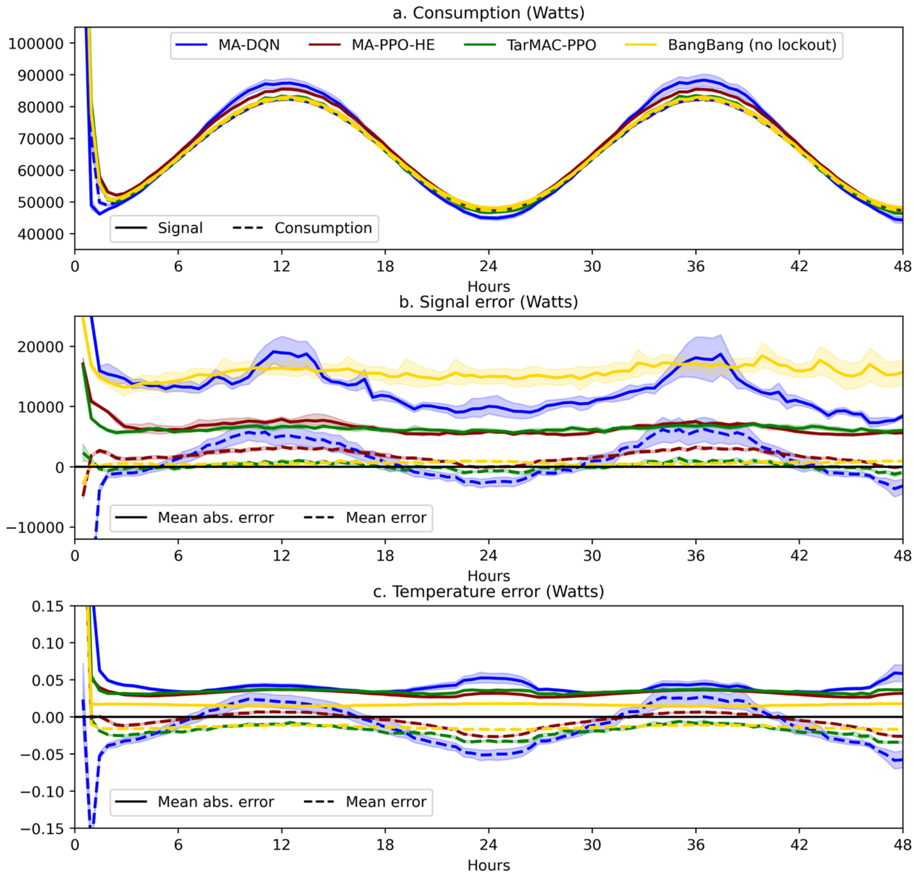


Fig. 1 MA-PPO-HE and TarMAC-PPO outperform MA-DQN and BBC for signal while keeping the temperature error low over 2 days with $N_{\mathrm{de}} = 50$ agents

5.2 Performance of agents

Table 2 shows the performance of different agents in environments with and without lockout with N_{de} of 10, 50, 250, and 1000 houses. To avoid cluttering the table, the temperature uncertainties are not shown, as they were always negligible.

The per-agent signal RMSE generally goes down when N_{de} increases. This is due to the lower relative discretization error, but also because, with more agents, errors have more chances to cancel each other, as explained in Appendix E. As expected, BBC controls the temperature well, but does not track the signal. Without lockout, the greedy myopic shows near-optimal signal tracking, where errors are due to discretization. It also maintains good control of the temperature. With lockout, however, it fails, as it runs out of available agents. The MPC gives good results for 10 agents, but its performance

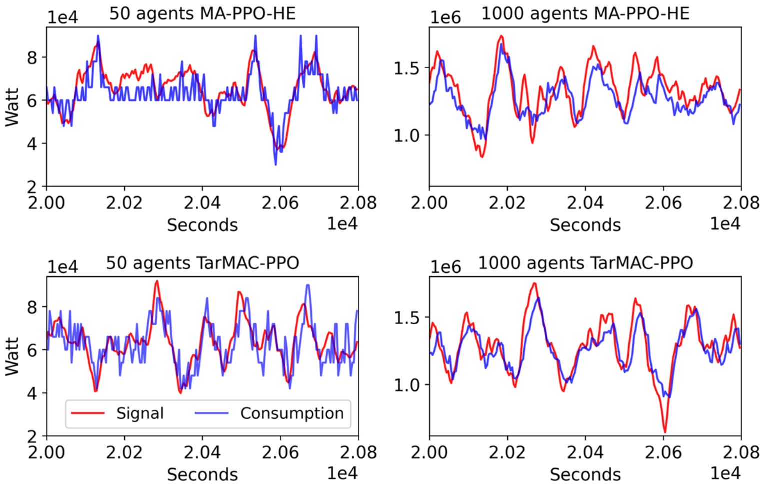


Fig. 2 Both MA-PPO policies scale seamlessly in the number of agents: signal and consumption on 800 s for $N_{\mathrm{de}} = 50$ and 1000

is limited by the lower control frequency of 12 s. It could not be run on $N_{\mathrm{de}} = 50$ for computing time reasons. MA-DQN controls the temperature well but is only slightly better than BBC on the signal. Both MA-PPO agents show significantly better performance, and TarMAC-PPO outperforms MA-PPO-HE at high N_{de}. The results without communication will be discussed in Sect. 5.5.

Figure 1 shows the behaviour of each agent over two days for 50 houses. Every point on the curves is averaged over 10 min. The mean error captures the error's bias by averaging the differences such that positives and negatives cancel each other, while the mean absolute error is the mean of the absolute differences. The signal and consumption curves start very high due to the initial situation, and then follow the sinusoidal pattern of the outdoor temperature. Without lockout, the BBC shows low temperature and signal mean error, with a significant signal mean absolute error, as it does not track high-frequency variations of the signal.

The MA-DQN agent has a smaller signal mean error and mean absolute error, especially at night when the amplitude of the signal variations is lower. During the day, the signal mean absolute error is still significant. Both MA-PPO agents, on the other hand, have a near-0 mean error in signal and temperature. Their signal mean absolute error is also significantly lower than the others, because they are able to track the high-frequency variations.

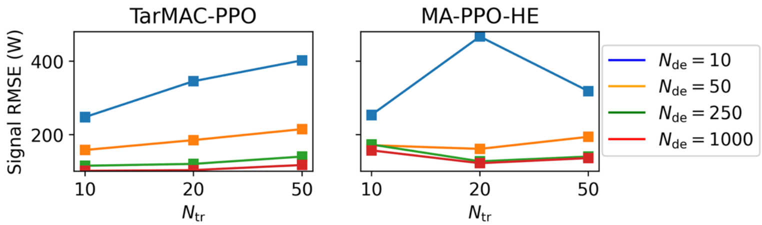
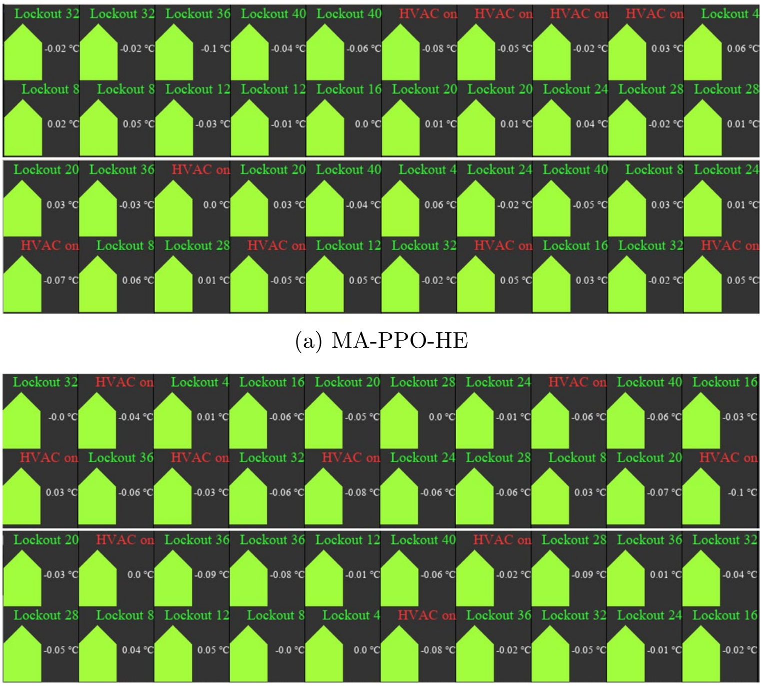


Fig. 3 Training with more agents N_{tr} does not lead to better performance, even when deployed on large N_{de}



(a) MA-PPO-HE

(b) TarMAC-PPO

Fig. 4 State of 20 houses controlled with two different PPO agents. The number on the top right is the remaining lockout time. (**a**) Two different agents of MA-PPO-HE with $N_{c_{de}} = 19$ show a "20-house" (up) and a "3-house" (down) pattern. (**b**) Two different TarMAC-PPO agents show no such pattern

5.3 Scalability with number of agents

As shown in Table 2, the PPO agents, and TarMAC-PPO especially, scale gracefully with the number of agents. Figure 2 shows the consumption and signal over 800 s for agents deployed over $N_d = 50$ and 1000 over 800 seconds. For $N_d = 50$, the agents do not perfectly match the signal. However, the same agent does better on 1000 houses. Indeed, as the environment is homogeneous, the local strategy scales smoothly by averaging out

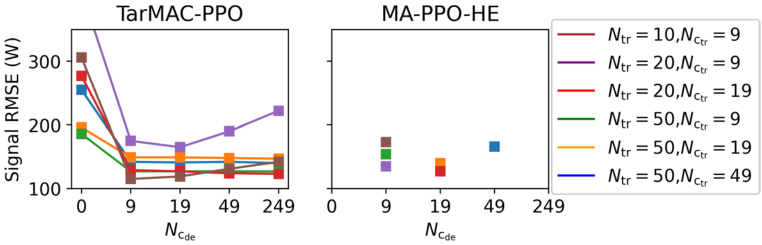
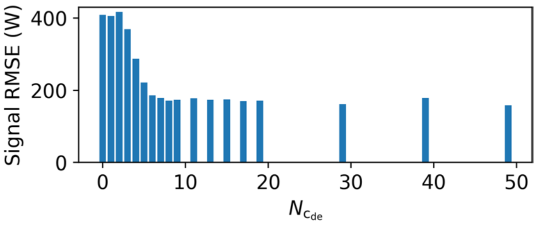


Fig. 5 TarMAC-PPO's performance does not increase after $N_{c_{de}}=9$, while MA-PPO-HE is better with $N_{c_{de}}=19$, for $N_{de}=250$ agents

Fig. 6 A TarMAC-PPO agent performs well as long as it communicates with $N_{c_{de}}=7$ agents or more, when deployed on $N_{de}=50$ houses



errors. The best performing agents for TarMAC-PPO were trained on environments with $N_{tr}=10$ houses. With MA-PPO-HE, it is often the agents trained on $N_{tr}=20$ that had the best results. Training with $N_{tr}=50$ probably makes the credit assignment harder as shown in Fig. 3. The ability of our MARL agents to be deployed on a large scale, despite being trained with a small number of houses, supports the notion of a low deployment investment for demand response in frequency regulation, as training the agent in the real world becomes a viable option. Moreover, it helps alleviate privacy concerns that may arise during real-world training with CT-DE.

5.4 PPO agents' dynamics

As visualized in Fig. 4, both MA-PPO-HE and TarMAC-PPO policies keep the ACs in lockout or ON, and never OFF. This is optimal for temperature control: an agent needing to be OFF to warm up after lockout, would not have had the time to warm up during the lockout and was thus ON for too long beforehand. The agents turn ON as soon as they can, but control when they turn OFF based on the context and the messages of other agents.

A fascinating feature of the learned policies is the cyclic behaviour used by MA-PPO-HE agents for coordination. As shown in Fig. 4, the ACs turn ON one after the other based on their positions in the aggregation, with a repetitive pattern. This happened for each MA-PPO-HE agent we trained, although the pattern period or moving direction was different. These patterns enable agent coordination thanks to the stable message structure, i.e., the fixed relative position of agent j's message to agent i in the $\tilde{O}_t^i$ vector. The TarMAC-PPO agents, on the other hand, do not follow a pattern in their collective behavior. Indeed, aggregated messages do not contain information about the structure of the neighbours. The coordination is done through flexible message contents.

5.5 Communications

The agents need communications to coordinate and get the best results. Intuitively, the more agents to communicate with, the better the performance because the observability of the environment is improved. In practice, this is not always the case, as shown in Fig. 5. For TarMAC-PPO, communicating with 9 neighbours often leads to the best performance. Higher values of $N_{c_{de}}$ can lead to a reduction of the weight of important messages in the aggregation. This is an important result, because a higher number of agents to communicate with would increase the computational cost of TarMAC-PPO accordingly. Instead, our approach can scale well while keeping communications limited to the local neighbours. Similarly, for MA-PPO-HE, communicating with 19 agents yields better results than with 49. Indeed, in MA-PPO-HE, the agents must have $N_{c_{de}} = N_{c_{tr}}$. During training, communicating with more agents increases the credit assignment difficulty as it increases the input size with non-controllable elements. It is also clear in Fig. 5 that agents trained to communicate do not cope well when not communicating. Figure 6 shows the performance of a TarMAC agent trained with $N_{tr} = 10$ and $N_{c_{tr}} = 9$ on an environment with $N_{de} = 50$ agents, when changing the number $N_{c_{de}}$ of neighbours it can communicate with. The performance is bad at low communication but stabilizes around 7 or 8 agents.

It is, however, possible to train an agent without communication to do better than Bang-Bang control, as shown by the performance of MA-PPO-NC in Table 2. Without coordinating with the others, an agent can learn to act well on average to minimize the signal error. When there are only a few agents, as when $N_{de} = 10$ or 50, this does not perform very well. However, the performance gap decreases when N_{de} increases: a good average policy will do well when applied on many agents. Another way to see this is that, with large N_{de}, each agent's importance becomes negligible in the final result. As such, the group can be seen as a single average agent, and the problem can be posed as a mean field game (Yang et al., 2018; Subramanian et al., 2018). This result is particularly interesting in the case where communication between agents poses a technical or a privacy issue.

5.6 Robustness

All the results presented were produced under certain assumptions, such as homogeneous houses and ACs, consistent outdoor temperature and signal profiles, and faultless communication. If such agents were to be deployed in the real world, they would be confronted with situations where these conditions are not satisfied. In this section, we evaluate the robustness of our trained agents to different disturbances in the deployment conditions.

Table 3 Performance under faulty communication (5 seeds)

		$N_{\text{de}} = 10$			$N_{\text{de}} = 50$			$N_{\text{de}} = 250$			$N_{\text{de}} = 1000$		
Per-agent		Signal	T	Max T	Signal	T	Max T	Signal	T	Max T	Signal	T	Max T
RMSE		(W)	(° C)	(° C)	(W)	(° C)	(° C)	(W)	(° C)	(° C)	(W)	(° C)	(° C)
MA-PPO-HE	$p_d = 0$	253 ± 1	0.04	0.08	161 ± 8	0.04	0.08	127 ± 2	0.04	0.11	122 ± 3	0.05	0.13
	$p_d = 0.1$	504 ± 2	0.07	0.14	207 ± 1	0.04	0.11	138 ± 2	0.05	0.13	118 ± 1	0.05	0.14
	$p_d = 0.5$	597 ± 2	0.10	0.19	274 ± 1	0.06	0.15	148 ± 1	0.06	0.151	115 ± 2	0.06	0.17
TarMAC-PPO	$p_d = 0$	247 ± 3	0.04	0.07	158 ± 2	0.04	0.09	115 ± 1	0.05	0.13	101 ± 2	0.05	0.14
	$p_d = 0.1$	246 ± 2	0.04	0.07	158 ± 2	0.04	0.09	115 ± 2	0.05	0.12	101 ± 1	0.05	0.14
	$p_d = 0.5$	248 ± 2	0.04	0.07	159 ± 3	0.04	0.09	115 ± 2	0.05	0.13	101 ± 1	0.05	0.14

Table 4 Performance under house and AC heterogeneity

	MA-PPO-HE		MA-PPO-HE-T	
Per-agent	Signal	Max T	Signal	Max T
RMSE	(W)	(°C)	(W)	(°C)
Homogeneous	161 ± 8	0.08	–	–
House thermal	285 ± 8	0.17	222 ± 7	0.11
AC cooling	292 ± 3	0.15	181 ± 3	0.14
Lockout duration	324 ± 9	0.15	246 ± 4	0.09
	TarMAC-PPO		TarMAC-PPO-T	
Homogeneous	158 ± 2	0.09	–	–
House thermal	184 ± 2	0.12	174 ± 2	0.11
AC cooling	187 ± 2	0.16	185 ± 9	0.16
Lockout duration	192 ± 3	0.09	251 ± 4	0.08

Table 5 Robustness on environment changes (5 seeds)

	MA-PPO-HE		TarMAC-PPO	
Per-agent	Signal	Max T	Signal	Max T
RMSE	(W)	(°C)	(W)	(°C)
Same as training	161 ± 8	0.08	158 ± 2	0.09
Solar gain	190 ± 6	0.09	174 ± 2	0.10
Outdoor T. +4℃	203 ± 4	0.11	198 ± 2	0.11
Outdoor T. −4℃	170 ± 1	0.09	184 ± 2	0.12
Signal average +30%	401 ± 2	0.11	302 ± 2	0.14
Signal average −30%	337 ± 4	0.10	317 ± 1	0.11
Signal noise amplitude +30%	188 ± 5	0.08	179 ± 3	0.09
Signal noise frequency +100%	200 ± 4	0.08	198 ± 5	0.09

5.6.1 Faulty communications

As previously demonstrated, communications are key for good performance of the agents. In this robustness test, we simulate defective communications. At every time step, each message m^i_j is defective with a probability p_d. In the case of TarMAC-PPO, this leads to the message not being received. For MA-PPO-HE, every element of the message is set to 0. We tested the best agents for $N_{\text{de}} = 10$, 50, 250, and 1000 houses with $p_d = 0.1$ and 0.5, as seen in Table 3. MA-PPO-HE agents' coordination is based on their stable communication structure. As a result, it copes badly with defective communications. Interestingly, when N_{de} is higher, the impact decreases, even leading to better signal performance at $N_{\text{de}} = 1000$. This may be because the MA-PPO-HE coordination leads to locally biased policies, which do not benefit from the averaging

Table 6 Computation time (s) for action selection, for 100 s of simulation. We report the time per-agents for a decentralized system and for the whole system otherwise

Agent	Decentralized	$N_{\text{de}} = 10$	$N_{\text{de}} = 1000$
TarMAC-PPO	Yes	0.002	0.001
MA-PPO-HE	Yes	0.006	0.006
MA-DQN	Yes	0.003	0.002
BBC	Yes	0.00001	0.00001
Greedy myopic	No	0.1	3.7
MPC – $H = 40$s	No	92.6680	–

effect reducing the relative error when N_{de} increases. The TarMAC-PPO handles temporary defects in communication very well, as its messages are aggregated. This is the case even with $p_d = 0.5$ and when the agent communicates with $N_{c_{\text{tr}}} = 9$ neighbours only.

5.6.2 Heterogeneous houses and ACs

In reality, different houses have different thermal characteristics. The ACs also do not always have the same rated power or lockout duration. We deployed the best trained MA-PPO-HE and TarMAC-PPO agents for 50-house environments that do not comply with these assumptions, to evaluate their robustness to separate disturbances. We also trained new agents on environments with these conditions, to allow the agents to learn to cope with heterogeneity. The relevant characteristics were observed by both agents as part of o_t^i, and of the messages m_j^i in MA-PPO-HE. These agents are referred to with the -T suffix. The thermal characteristics heterogeneity was simulated by adding a Gaussian noise to each element of θ_h^i for each house, with a standard deviation of 50% of the original value (the final values cannot be negative). For the ACs cooling capacities K_a^i, a value between 10, 12.5, 15, 17.5, and 20 kW was uniformly selected for each house. Finally, heterogeneity in the lockout duration $l_{\max}$ was tested by sampling uniformly between 32, 36, 40, 44, and 48 s.

The results are shown in Table 4. TarMAC-PPO is much more robust to heterogeneity in agents than MA-PPO-HE. This is because in MA-PPO-HE the coordination scheme is based on the stable dynamics of the agent's neighbours, especially with the lockout duration. TarMAC-PPO is instead more flexible with respect to different dynamics. For both agents, it is possible to reduce the effect of heterogeneity by training the agents on such environments and allowing them to observe the characteristics. This is different for heterogenity on the lockout duration, where TarMAC-PPO did not seem able to train satisfactorily on such conditions. An interesting observation is that the best TarMAC-PPO results were obtained when communicating with $N_{c_{\text{tr}}} = 49$ agents. With heterogeneous agents, more neighbours are needed for a representative input.

5.6.3 Other environments

We also tested our agents on environments differing from the training environment, with different outdoor temperature T_o, solar gain Q_s, too low or high average signal D_a, and higher or faster signal variations δ_s. As can be seen in Table 5, both agents are quite robust to such changes, with TarMAC-PPO usually leading to better results. When the signal is misbehaved, i.e., it is too low or too high to allow correct control of the temperature, there is a tradeoff between the signal and the temperature objectives. MA-PPO-HE gives higher priority to temperature, leading to higher signal RMSE.

5.7 Processing time

In Table 6, we report the processing time for action selection of the baseline and trained agents. The results are shown for 25 times steps (100 s of simulation), except for the MPC which simulated 100 s with 10-time steps. They were computed on the 12-core, 2.2 GHz Intel i7-8750 H CPU of a laptop computer.

As the decentralized, learned agents only need a single forward pass in a relatively small neural network, the time for action selection is sufficiently low for control when using 4-second time steps. Centralized approaches such as greedy myopic scale badly with many agents. MPC, already simplified with time steps of 12 s instead of 4, and a short horizon of 40 s, takes an unacceptable amount of time for more than 10 agents.

6 Conclusion

In this paper, we tackle the problem of high-frequency regulation with demand response by controlling discrete and dynamically constrained residential loads equipped with air conditioners with a decentralized, real-time agent trained by MA-PPO. We test two frameworks for local communication—fixed hand-engineered messages and learned targeted communication. The policies trained with few agents perform significantly better than baselines, scale seamlessly to large numbers of houses, and are robust to most disturbances. Our results show that MARL can be used successfully to solve some of the complex multi-agent problems induced by the integration of renewable energy in electrical power grids. Future works towards the application of such algorithms on real power systems could include gathering real-world data for real-world scenarios testing, sim2real transfer, integration of more flexible loads such as electric vehicles, as well as power grid safety issues. Exploring additional communication methods within MARL for addressing decentralized, fast time-scale demand response presents another important avenue for future investigations.

Appendix A: Notation

Table 7 contains the different notations we use in this paper.

Table 7 Notation table

Number of agents	N	Number of houses in cluster (general)
	N_{tr}	Number of houses in training environment
	N_{de}	Number of houses in test environment
	$N_{\mathrm{c}_{\mathrm{tr}}}$	Number of agents for communication during training
	$N_{\mathrm{c}_{\mathrm{de}}}$	Number of agents for communication at deployment
Temperatures	T_h	Indoor air temperature
	T_m	Indoor mass temperature
	T_o	Outside temperature
	T_T	Target indoor temperature
Signal and power	s_0	Base signal
	ρ	Power system operator signal
	s	Regulation signal
	D_a	Average power needed by the ACs
	D_o	Power needed by non flexible loads
	δ_s	Signal variation
	δ_p	Perlin noise
	β_p	Variation amplitude parameter
	P	Total cluster power consumption
AC state	ω	Status (ON or OFF)
	l	Time left for lockout
	P_a	Power consumption
	Q_a	Heat removed by the AC
House thermal model	θ_h	House thermal characteristics
	U_h	Outside walls conductance
	C_m	House thermal mass
	C_h	Air thermal mass
	H_m	Mass surface conductance
	θ_s	House lightning characteristics
	Q_s	Solar gain
AC model	θ_a	AC characteristics
	K_a	Cooling capacity
	COP_a	Coefficient of performance
	L_a	Latent cooling fraction
	$l_{\max}$	lockout duration
POMDP	$a, \mathcal{A}$	Action, action space
	$o, \mathcal{O}$	Observation, observation space
	$S, \mathcal{S}$	State, state space
	$r, \mathcal{R}$	Reward, reward function
	$\mathcal{P}$	Transition probabilities
	M	Set of communicating agents
	m_j^i	Message from j to i
	$\tilde{O}$	Concatenated observation and messages
	γ	Discount factor
	$\alpha_{\mathrm{temp}}, \alpha_{\mathrm{sig}}$	Weights in the reward function

Table 7 (continued)

Algorithms	H	Horizon
	Θ	Transition
	$Q(a, o)$, $T(a, o)$	Q-value prediction (with Q or target network)
	π_θ	Policy parameterized by θ
	V_ϕ	Critic parameterized by ϕ
	G	Return
	A^π	Advantage for policy π

Appendix B: Carbon emissions of the research project

As a significant amount of electricity has been used to train and run the models for this work, we publish its estimated carbon footprint.

Experiments were conducted using a private infrastructure, which has a carbon efficiency of 0.049 kg CO$N_{\mathrm{de}} = 10$eq/kWh. A cumulative of 10895 days, or 261480 h, of computation was mainly performed on CPU of type Intel Xeon Processor E5-2683 v4 (TDP of 120W). We assume on average a power usage of half the TDP for CPUs.

The total emissions are estimated to be 628 kg CO$N_{\mathrm{de}} = 50$eq of which 0% were directly offset. This is equivalent to 2550 km driven by an average car, or 314 kg of burned coal.

These estimations were conducted using the MachineLearning Impact calculator (Lacoste et al., 2019).

Appendix C: Environment details

C.1: Detailed house thermal model

The air temperature in each house evolves separately, based on its thermal characteristics θ_h, its current state, the outdoor conditions such as outdoor temperature and solar gain, and the status of the air conditioner in the house. The second-order model is based on Gridlab-D's Residential module user's guide (Betelle Memorial Institute, 2022).

Using Gridlab-D's module, we model an 8×12.5 m, one level rectangular house, with a ceiling height of 2.5 m, four 1.8m^2, 2-layer, aluminum windows, and two 2 m^2 wooden doors, leading to the values presented in Table 8.

To model the evolution of the house's air temperature $T_{h,t}$ and its mass temperature $T_{m,t}$, we assume that this temperature is homogeneous and do not consider the heat propagation in the house. We define the following variables:

Table 8 Default house thermal parameters θ_h

U_h	2.18×10^2 W/K
C_m	3.45×10^6 J/K
C_h	9.08×10^5 J/K
H_m	2.84×10^3 W/K

$$
\begin{aligned}
a =&C_mC_h/H_m\\
b =&C_m(U_h + H_m)/H_m + C_h\\
c =&U_h\\
d =&Q_{a,t} + Q_{s,t} + U_hT_{o,t}\\
dT_{h,t}/dt =&\big(H_mT_{m,t} - (U_h + H_m)T_{h,t}\\
&+U_hT_{o,t} + Q_{h,t} + Q_{s,t}\big)/C_h.
\end{aligned}
$$

The following coefficient are then computed:

$$
\begin{aligned}
r_1 =&(-b + \sqrt{b^2 - 4ac})/2a\\
r_2 =&(-b - \sqrt{b^2 - 4ac})/2a\\
A_1 =&(r_2T_{h,t} - dT_{h,t}/dt - r_2d/c)/(r2 - r1)\\
A_2 =&T_{h,t} - d/c - A_1\\
A_3 =&(r_1C_h + U_h + H_m)/H_m\\
A_4 =&(r_2C_h + U_h + H_m)/H_m.
\end{aligned}
$$

These coefficients are finally applied to the following dynamic equations:

$$
\begin{aligned}
T_{h,t+1} =&A_1e^{r_1\delta t} + A_2e^{r_2\delta t} + d/c\\
T_{m,t+1} =&A_1A_3e^{r_t\delta t} + A_2A_4e^{r_2\delta_t} + d/c.
\end{aligned}
$$

C.1.1: Solar gain

It is possible to add the solar gain to the simulator. It is computed based on the CIBSE Environmental Design Guide (CIBSE, 2015).

The house's lighting characteristics θ_S, which include the window area and the shading coefficient of 0.67 are needed to model the solar gain, $Q_{s,t}$.

Then, the following assumptions are made:

- The latitude is 30°.
- The solar gain is negligible before 7:30 am and after 5:30 pm at such latitude.
- The windows are distributed evenly around the building, in the 4 orientations.
- All windows are vertical.

This allows us to compute the coefficients of a fourth-degree bivariate polynomial to model the solar gain of the house based on the time of the day and the day of the year.

C.2: Detailed air conditioner model

Once again based on the Gridlab-D Residential module user's guide (Betelle Memorial Institute, 2022), we model the air conditioner's power consumption $P_{a,t}$ when turned ON, and the heat removed from the air $Q_{a,t}$, based on its characteristics θ_H, such as cooling capacity K_a, coefficient of performance COP_a, and the latent cooling fraction L_a.

COP_a and L_a are considered constant and based on default values of the guide: $COP_a = 2.5$ and $L_a = 0.35$. We have:

$$Q_{a,t} = -\frac{K_a}{1 + L_a}$$

$$P_{a,t} = \frac{K_a}{COP_a}.$$

We set K_a to 15 kW, or 50 000 BTU/hr, to be able to control the air temperature even with high outdoor temperatures. This is higher than most house ACs, but allows to have sufficient flexibility even at high outdoor temperatures (a 5kW AC would have to be always ON to keep a 20°C temperature when it is 38°C outside). This choice does not significantly affect our results: with lower outdoor temperatures, the problem is equivalent with lower AC power.

C.3: Regulation signal

C.3.1: Interpolation for the base signal

As described in Sect. 3.1.4, we estimate $D_{a,t}$ by interpolation. A bang-bang controller is ran without lockout for 5 min, and we compute the average power that was consumed. This gives a proxy for the amount of power necessary in a given situation.

A database was created by estimating $D_{a,t}$ for a single house for more than 4 million combinations of the following parameters: the house thermal characteristics θ_h, the differences between its air and mass temperatures $T_{a,t}$ and $T_{m,t}$ and the target temperature T_T, the outdoor temperature $T_{o,t}$, and the AC's cooling capacity K_a. If the solar gain is added to the simulation, the hour of the day and the day of the year are also considered.

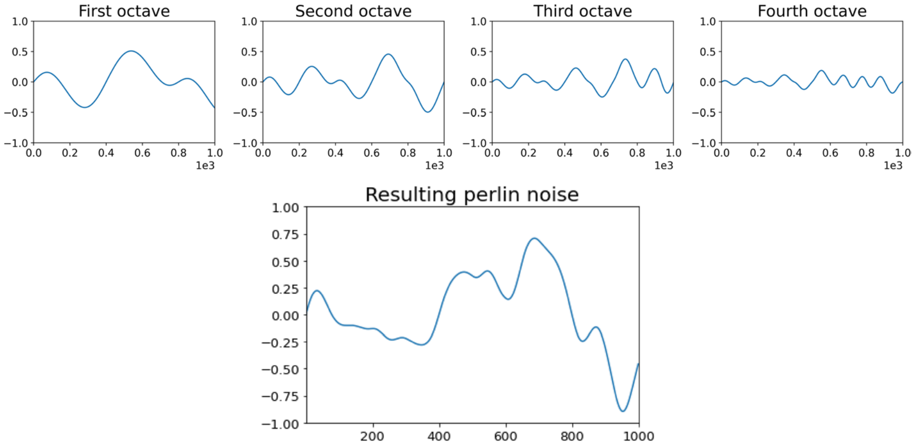


Fig. 7 Illustration of how several octaves add up to form Perlin noise. The frequency of the octaves increases as their amplitude decreases

When the environment is simulated, every 5 min, $D_{a,t}$ is computed by summing the interpolated necessary consumption of every house of the cluster. The interpolation process is linear for most parameters except for the 4 elements of θ_h and for K_a, which are instead using nearest neighbours to reduce the complexity of the operation.

C.3.2: Perlin noise

1-D Perlin noise is used to compute $\delta_{\Pi,t}$, the power generation high-frequency element. Designed for the field of computer image generation, this noise has several interesting properties for our use case.

Perlin noise is most of the time generated by the superposition of several sub-noises called octaves. It is possible to restrict the span of the values that they can take. Thus, it is possible to test the agents in an environment taking into account several frequencies of non-regular noise, but whose values are restricted within realistic limits. Moreover, the average value of the noise can be easily defined and does not deviate, which ensures that for a sufficiently long time horizon, the noise average is 0.

Each octave is characterized by 2 parameters: an amplitude and a frequency ratio. The frequency represents the distance between two random deviations. The amplitude represents the magnitude of the variation. Normally the frequency increases as the amplitude decreases. This way, high-amplitude noise is spread over a wider interval and lower amplitude noise is more frequent and compact. This is illustrated in Fig. 7.

In our case, we use 5 octaves, with an amplitude ratio of 0.9 between each octave and a frequency proportional to the number of the octave.

Appendix D: Algorithm details

D.1: Model Predictive Control

Our MPC is based on a centralized model. At each time step, information about the state of the agents is used to find the future controls that minimize the reward function over the next H time steps. The optimal immediate action is then communicated to the agents. At each time step, the algorithm calculates the ideal control combination for the H-time step horizon.

The cost function for both the signal and the temperature to minimize being the RMSE, the problem is modeled as a quadratic mixed-integer program. The solver used to solve the MPC is the commercial solver Gurobi 9.5.1 (Gurobi Optimization, 2022) together with CVXPY 1.3 (Diamond & Boyd, 2016). Gurobi being a licensed solver, its exact internal behavior is unknown to us and it acts as a black box for our MPC. However, we know that it solves convex integer problems using the branch and bound algorithm. The speed of resolution depends mainly on the quality of the solver's heuristics.

The computation time required for each step of the MPC increases drastically with the number of agents and/or H. To be able to test this approach with enough agents and a rolling horizon allowing to have reasonable performance, it was necessary to increase the time step at which the agents make decisions to 12 s (instead of 4 for other agents).

It was impossible to launch an experiment with the MPC agent for 48 h in a reasonable time. To compensate, we launched in parallel 200 agents having been started at random simulated times. In order to reach quickly the stability of the environment, the noise on the temperature was reduced to 0.05°C. We then measured the average RMSE over the first 2 h of simulation for each agent.

Despite this, it was impossible to test the MPC with more than 10 agents while keeping the computation time reasonable enough to be used in real time. That is to say, in a time shorter than the duration between two-time steps.

At each time step, the MPC solves the following optimization problem:

$$\min_{a\in\{0,1\}^{N\times H}} \sum_{t\in H} \alpha_{\text{sig}} \left(\sum_{i\epsilon N} P_{i,t} - s_0 \right)^2 + \alpha_{\text{temp}} \sum_{i\epsilon N} \left(T_{h,t,i} - T_{t,t,i} \right)^2,$$

such that it obeys the following physical constraints of the environment:

$$\begin{aligned} T_{h,t,i}, T_{m,t,i} &= F_1(a_{i,t}, T_{h,t-1,i}, T_{m,t-1,i}) \,\forall\, t \in H, i \in N \\ P_{h,t,i} &= a_{i,t} F_2(\theta_a^i)\&\forall\, t \in H, i \in N, \end{aligned}$$

and the lockout constraint:

$$l_{max}(a_{i,t} - \omega_{i,t-1}) - \sum_{k=0}^{l_{max}} (1 - \omega_{i,t-k}) \leq 0 \,\forall\, t \in H, i \in N,$$

where F_1 and F_2 are convex functions that can be deduced from the physical equations given in Sect. 3.

D.2: Learning-based methods

D.2.1: TarMAC and MA-PPO

The original implementation of TarMAC (Das et al., 2019) is built over the Asynchronous Advantage Actor-Critic (A3C) algorithm (Mnih et al., 2016). The environments on which it is trained have very short episodes, making it possible for the agents to train online over the whole memory as one mini-batch.

This is not possible with our environment where training episodes last around 16000 time steps. As a result, we built TarMAC over our existing MA-PPO implementation. The same loss functions were used to train the actor and the critic.

The critic is given all agents' observation as an input.

The actor's architecture is described in Fig. 8. Agent i's observations are passed through a first multi-layer perceptron (MLP), outputting a hidden state x. x is then used to produce a key, a value, and a query by three MLPs. The key and value are sent to the other agents, while agent i receives the other agents' keys and values. The other agents' keys are multiplied using a dot product with agent i's query, and passed through a softmax to produce the attention. Here, a mask is applied to impose the localized communication constraints and ensure agent i only listen to its neighbours. The attention is then used as weights for the values, which are summed together to produce the communication vector for agent i. For multi-round communication, the communication vector and x are concatenated and passed through another MLP to produce a new x, and the communication process is repeated for the number of communication hops. Once done, the final x and communication vector are once more concatenated and passed through the last MLP, the actor, to produce the action probabilities.

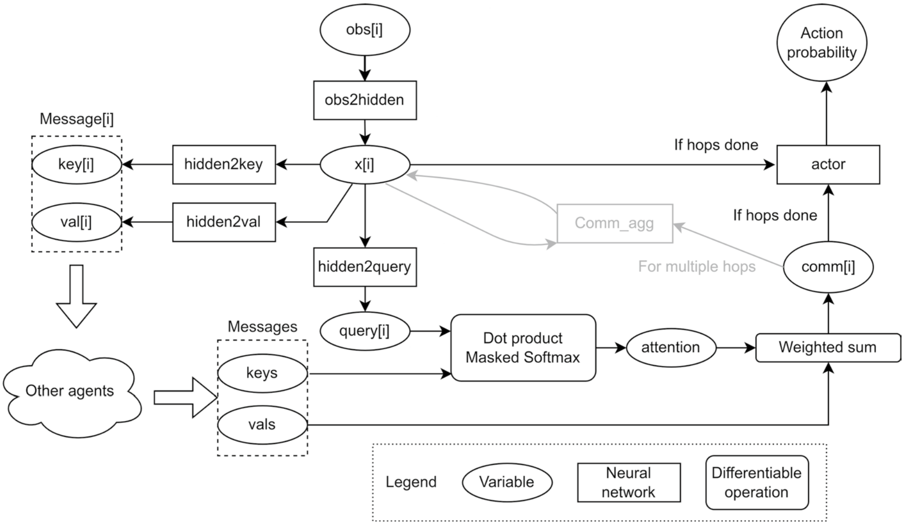


Fig. 8 Architecture of the TarMAC-PPO actor

We take advantage of the centralized training approach to connect the agents' communications in the computational graph during training. Once trained, the agents can be deployed in a decentralized way.

In order to maintain the privacy constraint of local communications, meaning that information about an agent is only shared with its immediate neighbours, two design choices have been made. First, the agents do not retain any memory of past messages received, relying solely on their present observations to generate messages. Second, the communication is limited to a single hop, ensuring that messages travel only to the neighbouring agents.

D.2.2: Neural networks architecture and optimization

For MA-DQN as well as for MA-PPO-HE, every neural network has the same structure, except for the number of inputs and outputs. The networks are composed of 2 hidden layers of 100 neurons, activated with ReLU, and are trained with Adam (Kingma & Ba, 2017).

For TarMAC-PPO, the actor's *obs2hidden*, *hidden2key*, *hidden2val*, *hidden2query* and *actor* MLPs (as shown in Fig. 8) all have one hidden layer of size 32. *obs2hidden* and *actor* are activated by ReLU whereas the three communication MLPs are activated by hyperbolic tangent. The hidden state x also has a size of 32.

The centralized critic is an MLP with two hidden layers of size 128 activated with ReLU. The input size is the number of agents multiplied by their observation size, and the output size is the number of agents.

For all networks, the inputs are normalized by constants approximating the mean and standard deviation of the features, to facilitate the training. The networks are optimized using Adam.

Table 9 Training hyperparameters

Hyperparameter	TarMAC-PPO	MA-PPO	MA-DQN
Learning rate	0.001	0.001	0.0001
Mini-batch size	256	512	256
Clip parameter	0.2	0.2	–
Max grad norm	0.5	0.5	–
Number epochs	200	200	–
Number updates	10	10	–
Number episodes	200	200	–
Discount factor γ	0.99	0.99	0.99
Key vector size	4 or 8	–	–
Comm. vector size	8	–	–
Number comm. rounds	1	–	–
Buffer capacity	–	–	65536
ϵ decay	–	–	0.995
Min ϵ	–	–	0.01

D.2.3: Hyperparameters

We carefully tuned the hyperparameters through grid searches. Table 9 shows the hyperparameters selected for the agents presented in the paper.

Appendix E: $N_{\text{de}} = 250$ and per-agent RMSE

In this section, we discuss the relation between the per-agent signal RMSE of an aggregation of N homogeneous agents if N is multiplied by an integer $k \in \mathbb{N}$.

We consider the aggregation of size kN as the aggregation of k homogeneous groups g_j of N agents which consumes a power $P^j_{g,t} = \sum_i^N P^i_{a,t}$. We have: $P_t = \sum_i^{kN} P^i_{a,t} = \sum_j^k P^j_{g,t}$.

We assume that each group tracks an equal portion of the signal $s^j_t = s_t/k$. We assume that the tracking error $P^j_{g,t} - s^j_t$ follows a 0-mean Gaussian of standard deviation σ_g. This Gaussian error is uncorrelated to the noise of other groups.

It follows from the properties of Gaussian random variables that the aggregation signal error $P_t - s_t$ follows a Gaussian distribution of mean $\mu_k = 0$ and standard deviation $\sigma_k = \sqrt{k}\sigma_g$ for all $k \geq 1$ with $k \in \mathbb{N}$.

Hence the signal's RMSE of a group of kN agents, which is a measured estimation of σ_k, is approximately $\sqrt{k}$ times the RMSE of a group of N agents, which estimates σ_g. Finally, the per-agent RMSE is computed as the group's RMSE divided by the number of agents. We therefore have that the per-agent RMSE of kN agents is approximately $\sqrt{k}/k = 1/\sqrt{k}$ times the RMSE of N agents.

This discussion provides an intuitive explanation for the diminution of the relative RMSE when the number of agents increases. However, it is based on the assumption that the error of each group is not biased, which is not necessarily true with our agents. This explains why the RMSEs are not 10 times lower passing from $N_{\text{de}} = 10$ to $N_{\text{de}} = 1000$.

Author contributions Problem identification and formalization: VM and ALL Simulator implementation: VM and PM Classical baselines implementations: PM Machine learning algorithms: VM, TZ, HN Experiments: VM and PM Analysis: Everyone Manuscript redaction: VM Manuscript review: Everyone Supervision and guidance: ALL and LP.

Funding Natural Sciences and Engineering Research Council of Canada (NSERC): Vincent Mai (VM), Hadi Nekoei (HN), Liam Paull (LP). Institut de Valorisation des Données: Philippe Maisonneuve (PM), Antoine Lesage-Landry (ALL). Microsoft Research and Samsung: Tianyu Zhang (TZ).

Data availability The simulator is open-sourced at https://github.com/ALLabMTL/marl-demandresponse.

Code availability The code to reproduce the results are available at https://github.com/ALLabMTL/marl-demandresponse.

Declarations

Conflict of interest None

References

Agency, I. E. (2018). The Future of Cooling, url: https://www.iea.org/reports/the-future-of-cooling

Agency, I. E. (2021). Greenhouse Gas Emissions from Energy: Overview.

Agency, I. E. (2022). Energy Statistics Data Browser – Data Tools. Available on: https://www.iea.org/data-and-statistics/data-tools/energy-statistics-data-browser (Accessed on Sept 15).

Ahmadiahangar, R., Häring, T., Rosin, A., Korõtko, T., Martins, J. (2019). Residential load forecasting for flexibility prediction using machine learning-based regression model. In *2019 IEEE International Conference on Environment and Electrical Engineering and 2019 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I &CPS Europe),* pp. 1–4. https://doi.org/10.1109/EEEIC.2019.8783634

Ahrarinouri, M., Rastegar, M., & Seifi, A. R. (2021). Multiagent reinforcement learning for energy management in residential buildings. *IEEE Transactions on Industrial Informatics, 17*(1), 659–666. https://doi.org/10.1109/TII.2020.2977104

Aladdin, S., El-Tantawy, S., Fouda, M. M., & Tag Eldien, A. S. (2020). Marla-sg: Multi-agent reinforcement learning algorithm for efficient demand response in smart grid. *IEEE Access, 8*, 210626–210639. https://doi.org/10.1109/ACCESS.2020.3038863

Amin, U., Hossain, M., & Fernandez, E. (2020). Optimal price based control of hvac systems in multizone office buildings for demand response. *Journal of Cleaner Production, 270*, 122059.

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., Mordatch, I. (2020). *Emergent tool use from multi-agent autocurricula*. arXiv:1909.07528 [cs, stat]

Betelle Memorial Institute: Residential Module User's Guide. In *GridLAB-D Wiki*. Available at: http://gridlab-d.shoutwiki.com/wiki/Main_Page (Accessed: September 15, 2022) (Accessed 2022). http://gridlab-d.shoutwiki.com/wiki/Main_Page

Bevrani, H., Ghosh, A., & Ledwich, G. (2010). Renewable energy sources and frequency regulation: survey and new perspectives. *IET Renewable Power Generation, 4*(5), 438–457.

Biagioni, D., Zhang, X., Wald, D., Vaidhynathan, D., Chintala, R., King, J., Zamzam, A. S. (2021). *PowerGridworld: A Framework for Multi-Agent Reinforcement Learning in Power Systems*. arXiv https://doi.org/10.48550/ARXIV.2111.05969.https://arxiv.org/abs/2111.05969

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W. (2016). OpenAI Gym.

Callaway, D. S. (2009). Tapping the energy storage potential in electric loads to deliver load following and regulation, with application to wind energy. *Energy Conversion and Management, 50*(5), 1389–1400.

Chen, B., Francis, J., Pritoni, M., Kar, S., Bergés, M. (2020). Cohort: Coordination of heterogeneous thermostatically controlled loads for demand flexibility. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation,* pp. 31–40. https://doi.org/10.1145/3408308.3427980.arXiv:2010.03659 [cs, eess]. url: http://arxiv.org/abs/2010.03659

Chen, B., Francis, J., Pritoni, M., Kar, S., Bergés, M. (2020). Cohort: Coordination of heterogeneous thermostatically controlled loads for demand flexibility. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation,* pp. 31–40.

CIBSE (2015). Guide A: Environmental Design, 8th edn. Chartered Institution of Building Services Engineers.

Dantzig, G. B. (1957). Discrete-variable extremum problems. *Operations Research, 5*(2), 266–288. https://doi.org/10.1287/opre.5.2.266

Das, A., Gervet, T., Romoff, J., Batra, D., Parikh, D., Rabbat, M., Pineau, J. (2019). Tarmac: Targeted multi-agent communication. In *Proceedings of the 36th International Conference on Machine Learning,* pp. 1538–1546. PMLR, url: https://proceedings.mlr.press/v97/das19a.html

Diamond, S., Boyd, S. (2016). CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*. To appear.

Dusparic, I., Harris, C., Marinescu, A., Cahill, V., Clarke, S. (2013). Multi-agent residential demand response based on load forecasting. In *2013 1st IEEE Conference on Technologies for Sustainability (SusTech),* pp. 90–96 https://doi.org/10.1109/SusTech.2013.6617303

Fuchs, A., Walton, M., Chadwick, T., Lange, D. (2021). *Theory of mind for deep reinforcement learning in hanabi*. arXiv:2101.09328 [cs].

Fuchs, A., Walton, M., Chadwick, T., Lange, D. (2021). *Theory of mind for deep reinforcement learning in hanabi*. arXiv:2101.09328 [cs].

Gronauer, S., & Diepold, K. (2021). Multi-agent deep reinforcement learning: A survey. *Artificial Intelligence Review*. https://doi.org/10.1007/s10462-021-09996-w

Guan, C., Chen, F., Yuan, L., Zhang, Z., Yu, Y. (2023). *Efficient communication via self-supervised information aggregation for online and offline multi-agent reinforcement learning* (arXiv:2302.09605) https://doi.org/10.48550/arXiv.2302.09605.arXiv:2302.09605 [cs]

Gupta, J.K., Egorov, M., Kochenderfer, M. (2017) In: Sukthankar, G., Rodriguez-Aguilar, J.A. (eds.) *Cooperative Multi-agent Control Using Deep Reinforcement Learning. Lecture Notes in Computer Science,* vol. 10642, pp. 66–83. Springer, Cham. https://doi.org/10.1007/978-3-319-71682-4_5

Gurobi Optimization. (2022). LLC: Gurobi Optimizer Reference Manual. https://www.gurobi.com

Jiang, J., Lu, Z. (2018). Learning attentional communication for multi-agent cooperation. In *Advances in Neural Information Processing Systems,* vol. 31. Curran Associates, Inc., url: https://proceedings.neurips.cc/paper/2018/hash/6a8018b3a00b69c008601b8becae392b-Abstract.html

Dong, J., Olama, M., Kuruganti, T., Nutaro, J., Winstead, C., Xue, Y., Melin, A. (2018). Model predictive control of building on/off hvac systems to compensate fluctuations in solar power generation. In *2018 9th IEEE International Symposium on Power Electronics for Distributed Generation Systems (PEDG),* pp. 1–5. https://doi.org/10.1109/PEDG.2018.8447840

Kingma, D. P., Ba, J. (2017). *Adam: A method for stochastic optimization*. arXiv:1412.6980 [cs] arXiv:1412.6980

Kraemer, L., & Banerjee, B. (2016). Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing, 190*, 82–94. https://doi.org/10.1016/j.neucom.2016.01.031

Kundur, P. (2007). Power system stability. Power system stability and control, pp. 7–1.

Lacoste, A., Luccioni, A., Schmidt, V., Dandres, T. (2019). *Quantifying the carbon emissions of machine learning* (arXiv:1910.09700) https://doi.org/10.48550/arXiv.1910.09700.arXiv:1910.09700 [cs].

Lagae, A., Lefebvre, S., Cook, R., DeRose, T., Drettakis, G., Ebert, D. S., Lewis, J. P., Perlin, K., & Zwicker, M. (2010). A survey of procedural noise functions. *Computer Graphics Forum, 29*(8), 2579–2600. https://doi.org/10.1111/j.1467-8659.2010.01827.x

Lauro, F., Moretti, F., Capozzoli, A., Panzieri, S. (2015). Model predictive control for building active demand response systems. Energy Procedia 83, 494–503. https://doi.org/10.1016/j.egypro.2015.12.169.Sustainability in Energy and Buildings: Proceedings of the 7th International Conference SEB-15.

Lee, Y. M., Horesh, R., & Liberti, L. (2015). Optimal hvac control as demand response with on-site energy storage and generation system. *Energy Procedia, 78*, 2106–2111.

Lesage-Landry, A., Taylor, J. A. (2021). Callaway, D.S.: Online convex optimization with binary constraints. IEEE Transactions on Automatic Control.

Lesage-Landry, A., & Taylor, J. A. (2018). Setpoint tracking with partially observed loads. *IEEE Transactions on Power Systems, 33*(5), 5615–5627.

Lillicrap, T.P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D. (2019). *Continuous control with deep reinforcement learning*arXiv:1509.02971 *[cs, stat].*

Liu, M., & Shi, Y. (2015). Model predictive control of aggregated heterogeneous second-order thermostatically controlled loads for ancillary services. *IEEE Transactions on Power Systems, 31*(3), 1963–1971.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., Mordatch, I. (2020). *Multi-agent actor-critic for mixed cooperative-competitive environments*. arXiv:1706.02275 [cs].

Maasoumy, M., Sanandaji, B. M., Sangiovanni-Vincentelli, A., Poolla, K. (2014). Model predictive control of regulation services from commercial buildings to the smart grid. In *2014 American Control Conference,* pp. 2226–2233 IEEE.

Mai, V., Maisonneuve, P., Zhang, T., Nekoei, H., Paull, L., Lesage-Landry, A. (2023). Multi-agent reinforcement learning for fast-timescale demand response of residential loads. In *AAMAS'23:*

Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS'23).

Mathieu, J. L., Koch, S., & Callaway, D. S. (2012). State estimation and control of electric loads to manage real-time energy imbalance. *IEEE Transactions on Power Systems, 28*(1), 430–440.

Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T.P., Harley, T., Silver, D., Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. arXiv:1602.01783 [cs]. arXiv: 1602.01783

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*(75407540), 529–533. https://doi.org/10.1038/nature14236

Olama, M. M., Kuruganti, T., Nutaro, J., & Dong, J. (2018). Coordination and control of building hvac systems to provide frequency regulation to the electric grid. *Energies*. https://doi.org/10.3390/en11071852

OpenAI, Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., Pinto, H.P.d.O., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., Zhang, S. (2019). Dota 2 with large scale deep reinforcement learning arXiv: 1912.06680

Pardo, F., Tavakoli, A., Levdik, V., Kormushev, P. (2018). Time limits in reinforcement learning. In: Dy, J., Krause, A. (eds.) *Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research,* vol. 80, pp. 4045–4054. PMLR. url: https://proceedings.mlr.press/v80/pardo18a.html

Pigott, A., Crozier, C., Baker, K., Nagy, Z. (2021). *GridLearn: Multiagent Reinforcement Learning for Grid-Aware Building Energy Management*. https://doi.org/10.48550/ARXIV.2110.06396

Qin, Z., Zhu, H., Ye, J. (2022). *Reinforcement learning for ridesharing: An extended survey*. arXiv: 2105.01099 [cs].

Roesch, M., Linder, C., Zimmermann, R., Rudolf, A., Hohmann, A., & Reinhart, G. (2020). Smart grid for industry using multi-agent reinforcement learning. *Applied Sciences*. https://doi.org/10.3390/app10196900

Sartoretti, G., Kerr, J., Shi, Y., Wagner, G., Kumar, T.K.S., Koenig, S., Choset, H. (2019). Primal: Pathfinding via reinforcement and imitation multi-agent learning. *IEEE Robotics and Automation Letters 4*(3), 2378–2385. https://doi.org/10.1109/LRA.2019.2903261.arXiv:1809.03531 [cs].

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). *Proximal policy optimization algorithms*. https://doi.org/10.1007/s00038-010-0125-8

Siano, P. (2014). Demand response and smart grids-a survey. *Renewable and Sustainable Energy Reviews, 30*, 461–478.

Subramanian, J., Seraj, R., Mahajan, A. (2018). Reinforcement learning for mean field teams. In *Workshop on Adaptive and Learning Agents at the International Conference on Autonomous Agents and Multi-Agent Systems*.

Taylor, J. A., Dhople, S. V., & Callaway, D. S. (2016). Power systems without fuel. *Renewable and Sustainable Energy Reviews, 57*, 1322–1336.

Vazquez-Canteli, J. R., Dey, S., Henze, G., Nagy, Z. (2020). Citylearn: Standardizing research in multi-agent reinforcement learning for demand response and urban energy management. https://doi.org/10.48550/arXiv.2012.10504.arXiv:2012.10504 [cs].

Vazquez-Canteli, J.R., Henze, G., Nagy, Z. (2020). Marlisa: Multi-agent reinforcement learning with iterative sequential action selection for load shaping of grid-interactive connected buildings. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation. BuildSys '20,* pp. 170–179. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3408308.3427604

Wang, J., Xu, W., Gu, Y., Song, W., Green, T. C. (2022). Multi-agent reinforcement learning for active voltage control on power distribution networks (arXiv:2110.14300). https://doi.org/10.48550/arXiv.2110.14300.arXiv:2110.14300 [cs].

Wang, Z., Chen, B., Li, H., & Hong, T. (2021). Alphabuilding rescommunity: A multi-agent virtual testbed for community-level load coordination. *Advances in Applied Energy, 4*, 100061.

Wu, X., He, J., Xu, Y., Lu, J., Lu, N., & Wang, X. (2018). Hierarchical control of residential hvac units for primary frequency regulation. *IEEE Transactions on Smart Grid, 9*(4), 3844–3856. https://doi.org/10.1109/TSG.2017.2766880

Xi, L., Jianfeng, H., & Y., Xu, Y., Liu, L., Zhou, Y., Li, Y. (2018). Smart generation control based on multi-agent reinforcement learning with the idea of the time tunnel. *Energy, 153*, 977–987. https://doi.org/10.1016/j.energy.2018.04.042

Yang, Y., Hao, J., Zheng, Y., Hao, X., Fu, B. (2019). Large-scale home energy management using entropy-based collective multiagent reinforcement learning framework. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems. AAMAS '19,* pp. 2285–2287. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC.

Yang, Y., Luo, R., Li, M., Zhou, M., Zhang, W., & Wang, J. (2018). *Mean field multi-agent reinforcement learning*. https://doi.org/10.48550/arXiv.1802.05438

Yu, C., Velu, A., Vinitsky, E., Wang, Y., Bayen, A., Wu, Y. (2021). The surprising effectiveness of ppo in cooperative, multi-agent games. arXiv:2103.01955 [cs].

Yuan, L., Wang, J., Zhang, F., Wang, C., Zhang, Z., Yu, Y., & Zhang, C. (2022). Multi-agent incentive communication via decentralized teammate modeling. *Proceedings of the AAAI Conference on Artificial Intelligence, 36*(99), 9466–9474. https://doi.org/10.1609/aaai.v36i9.21179

Zhang, W., Lian, J., Chang, C.-Y., & Kalsi, K. (2013). Aggregated modeling and control of air conditioning loads for demand response. *IEEE Transactions on Power Systems, 28*(4), 4655–4664.

Zhou, X., Dall'Anese, E., & Chen, L. (2019). Online stochastic optimization of networked distributed energy resources. *IEEE Transactions on Automatic Control, 65*(6), 2387–2401.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Vincent Mai[1,2] · Philippe Maisonneuve[2,3,4] · Tianyu Zhang[1,2] · Hadi Nekoei[1,2] · Liam Paull[1,2] · Antoine Lesage-Landry[2,3,4]

✉ Hadi Nekoei
nekoeihe@mila.quebec

Vincent Mai
vincent.mai@umontreal.ca

Philippe Maisonneuve
philippe.maisonneuve@polymtl.ca

Tianyu Zhang
tianyu.zhang@mila.quebec

Liam Paull
liam.paull@umontreal.ca

Antoine Lesage-Landry
antoine.lesage-landry@polymtl.ca

[1] Université de Montréal, 3150 Jean Brillant St, Montreal H3T 1N8, QC, Canada

[2] Mila-Quebec AI Institute, 6666 Rue Saint-Urbain, Montreal H2S 3H1, QC, Canada

[3] Polytechnique Montréal, 2500, chemin de Polytechnique, Montreal H3T 1J4, QC, Canada

[4] GERAD, 2920, Chemin de la Tour, Montreal H3T 1J4, QC, Canada