Safety through Intrinsically Motivated Imitation Learning

Henrique Donancio, Laurent Vercouter

Safety through Intrinsically Motivated Imitation Learning

Henrique Donâncio[1], Laurent Vercouter[1]

[1] Normandie Université, INSA Rouen
LITIS

Abstract

Deep Reinforcement Learning methods require a large amount of data to achieve good performance. This scenario can be more complex, handling real-world domains with high-dimensional state space. However, historical interactions with the environment can boost the learning process. Considering this, we propose in this work an imitation learning strategy that uses previously collected data as a baseline for density-based action selection. Then, we augment the reward according to the state likelihood under some distribution of states given by the demonstrations. The idea is to avoid exhaustive exploration by restricting state-action pairs and encourage policy convergence for states that lie in regions with high density. The adopted scenario is the pump scheduling for a water distribution system where real-world data and a simulator are available. The empirical results show that our strategy can produce policies that outperform the behavioral policy and offline methods, and the proposed reward functions lead to competitive performance compared to the real-world operation.

Keywords

Learning from Demonstrations, Deep Reinforcement Learning, Pump Scheduling Optimization

1 Introduction

Over the past years, Reinforcement Learning (RL) approaches combined with function approximators have been applied in different tasks such as games [1, 2] to control [3, 4]. The appeal of this approach is the ability to support decision-making and leverage scalability for complex domains. However, exploration in large state spaces as found in the real world can be costly, inefficient, and even infeasible. For example, in scenarios such as healthcare systems and autonomous driving, trial and error methods are not an option due to safety constraints. A way to mitigate these problems is using historical interactions with the environment, an approach called Offline Reinforcement Learning[1].

In the Offline RL [5] settings, the experiences of the agent are limited to collected data, without the possibility of further exploration. The reasons for this limitation include the complexity of building accurate simulators and safety constraints for exploring the environment. Even when online data collection is reasonable, the use of prior datasets capable of generalizing to efficient policies can be attractive due to the costs to interact with the environment. Offline RL methods such as [6, 7] rely upon the idea of constraining the policy to the dataset to mitigate overestimation caused when facing out-of-distribution state-action pairs.

On the other hand, in Online RL, agents interact with the environment in an exploration-exploitation trade-off fashion. A widely applied exploration strategy is the epsilon-greedy [8], in which a given probability trade-off between exploring the environment or exploiting the policy learned. The exploration can also be encouraged by intrinsic motivation as curiosity or disagreement in the state's estimation by augmenting the reward function. For instance, some works [9, 10, 11] propose strategies based on counting occurrences of states/actions to encourage exploration of unfamiliar areas of the state-action space. In [12], the reward is augmented based on the disagreement of an ensemble of parametric Q-functions. Finally, methods to perform safe exploration are discussed in [13]. In particular, incorporating demonstrations and restricting the exploration to meaningful states can produce safe policies.

This work proposes the *Safety through Intrinsically Motivated Imitation Learning (SIMIL)* strategy that uses the distribution of historical interactions (*demonstrations*) as a guideline for action selection. The approach works as follows: given a current state, action selection depends on choosing the one that occurs most frequently in the most similar states found in the demonstrations. Later, we augment the immediate reward with an intrinsic motivation [14] according to the state likelihood under some distribution of states. The underlying idea is to constrain the policy to state-action pairs found in expert demonstrations using k-Nearest Neighbors (k-NN) to avoid exhaustive exploration. Also, we encourage states that lie in high-density regions under the demonstrations distribution using Kernel Density Estimation (KDE) [15].

We apply this imitation learning strategy in a scenario of pumping scheduling for water distribution systems (WDS). For that, it is available a dataset of three years of data collected in timesteps of one minute from a real-world operation. The pump scheduling is the process to decide when, and in some cases at which speed, the pump(s) should operate regarding the forecasting of the water demand. Yet,

[1] Some works uses the term *Batch* instead of *Offline*

some requirements must be satisfied, including safety constraints of water level in the tanks and pressure in the network's nodes. Some works have addressed these questions through several methods, including linear optimization, evolutionary and branch-and-bound algorithms, and recently Deep RL [16, 17, 18, 19]. This work uses a Deep Q-Networks (DQN) [20, 1]-based approach to handle the pump scheduling problem. The contributions presented in this work are the following:

- A formulation of the pumping scheduling problem using Partially Observable Markov Decision Process (POMDP) is presented, with definitions of system states/observations, actions, and reward function. These definitions allow the system to operate by achieving the constraints and minimizing the associated costs;
- An imitation learning strategy using real-world/offline data. The empirical results demonstrated that the obtained policies achieved competitive average cumulative rewards compared with fully-offline training.
- To evaluate the proposed scheduling, we compare the results with the real-world water distribution system regarding the electricity consumed, pumps use distribution and the tank level profile. The results showed that our approach achieved competitive performance with real-world operation.

The organization of the paper is as follows. Section 2 describes some related works. Section 3 introduces the pump scheduling problem and its formalization as a POMDP; Section 4 presents technical aspects. Section 5 describes the imitation learning strategy proposed. Section 6 describes the conducted experiments and shows the obtained results. Finally, are presented conclusions in Section 7.

2 Related Works

Offline RL methods rely on the capacity to exploit and generalize from static datasets to efficient policies. Although, leveraging the learning process using prior experiences can be challenging due to the distributional shift issue and overly optimism in the face of uncertainty [6, 5]. The Random Ensemble Mixture (REM) [21] used in this paper addresses this problem by using a convex combination of Q-values to mitigate overestimation under the assumption of a diverse and large dataset. Also, it can be adopted orthogonally to other sampling methods allowing further online data collection. Similarly, Jaques and colleagues [7] handle the overestimation issue by applying a dropout-inspired Q-learning and penalizing divergence from prior data distribution through KL-control. Batch Constrained Deep Q-Learning (BCQ) [6, 22] also constrains the policy regarding actions found in the dataset using as a baseline a generative model.

Some other works have used demonstrations as a pre-training step. In [23] is presented Deep Q-learning from Demonstrations (DQfD) that uses demonstrations as a pre-training and later improves the learned policy with self-generated experiences. The pre-training phase applies a supervised loss to ground values from unseen actions regarding the demonstrations. After the pre-training, DQfD interacts with the environment through the learned policy. In [24] the method incorporates the actor-critic algorithm DDPG, and a loss is applied to tie the policy to the offline data.

As an imitation learning strategy, this work interacts with the environment by performing a density-based action selection using the demonstrations as a distribution. Some approaches instead use this distribution to create models to perform exploration. For instance, Model-based Offline Policy Optimization (MOPO) [25] builds a model using supervised learning and then penalizes the uncertainty in further interactions based on the model's error estimation. Therefore, MOPO balances the return and risk in collecting experiences out-of-distribution of the support data. Similarly, the Model-Based Offline Reinforcement Learning (MOReL) [26] proposes to learn a policy for a pessimist MDP (P-MDP) using offline data. This P-MDP partitions the state space according to the uncertainty, applying a reward penalty to unknown areas. In [27] the authors propose a model-based approach that can mix online data collection with prior offline data. For that, a model is built and incrementally improved through Monte Carlo Tree Search rollouts.

Imitation Learning approaches [28] aims to mimic the behavior observed in demonstrations. In [29] is proposed a hierarchical method for action selection with self-improvement over time. The first step is to select a primitive that corresponds to some behavior. The second step is selecting a sub-goal achieved by performing the chosen primitive. Finally, an action generator picks a policy to execute the primitive. The underlying idea is to improve the action generator by practicing. Similar to our work and [30], this approach uses k-NN queries to retrieve a primitive from demonstrations given a current state. The difference lies in the fact that we propose a passive sampling approach, letting the evaluation of the state-action pairs to the learning method.

Our work has a basis on the wealth of literature on imitation learning and offline RL. However, the intersection between these branches remains underexplored. While offline methods rely upon methods to improve the exploitation of static datasets, the exploration often seeks to uncover areas in the state space. Yet, static datasets may not be large and diverse, and exploring unknown states can produce undesirable behaviors. Thus, the premise of our contribution is to increase the sample efficiency. We consider expert demonstrations as an underlying model for action selection and encourage policy convergence to high-density regions under the demonstration distribution through intrinsic motivation.

3 Modeling the pump scheduling problem

In water distribution systems, pump scheduling is a decision process about when operating pumps to supply water while limiting electricity consumption. Therefore some constraints must be respected, including a minimum pressure within the network, safety water level in the tanks, and avoiding frequent switches in pump operation to protect the assets. To that end, distinct strategies can be used according to the particularities of the system. For instance, pumping water in off-peak hours when the price of electricity has different tariffs throughout the day or reducing the tank level in periods of low consumption to preserve the water quality, and so on.

The water distribution system used here is located in Worms, Germany, and supplies water for about 120000 citizens[2]. The composition of this system is one station with four pumps (NP1, NP2, NP3, NP4), with distinct settings and fixed speed (ON/OFF). The flow Q through those pumps is proportional to the electricity consumption kW, being NP1 $>$ NP2 $>$ NP3 $>$ NP4. In other words, using pump NP1 supplies more water in the network than pump NP2 but also corresponds to higher electricity consumption. Also, two storage tanks with different capacities are placed and provide water for the end consumers. Among the constraints and requirements established in the operation settings for this system are the following:

- It is desirable to avoid frequent switches and distribute pump operations to protect the assets;
- It is imposed a boundary condition of the tank level and, once achieved, the minimum pressure is guaranteed;
- It is desirable to provide water exchange in the tank during one day of operation to keep the water quality.

The tank is located 47m above the pumps and has a 10m length. Thus, the tank levels considered are in the range of [47, 57]m. We consider only one tank once that the second has the level *stable* along with the operation. The specialists assume a safety operation guarantee with at least 3m filled with water. Besides this, the system does not have sensors measuring the water's quality. Thus, to *ensure* the exchange and preserve the water's quality, we assume that in one operation day, the level must decrease below half of the total capacity. Finally, the upper boundary constraint overlaps the physical limit.

As with many real-world tasks, the scenario of pumping scheduling is partially observable. In other words, the agent has a noisy or incomplete observation of the environment. For instance, most of the state's features are noisy once it has been gathering by sensors. Also, the water demand has variance along the hours, days, and seasons, even following a pattern. A POMDP is an extension of MDP that considers uncertainty regarding the current state of the environment. Formally, the POMDP can be defined as [8]:

$$POMDP = < S, A, P, R, \Omega, O, \gamma > \quad (1)$$

where the set S correspond to the **States** of the environment; A is defined as the set of **Actions** available; P is the **Transition Probability** which defines the probability being in some state $s_t \in S$, taking an action $a_t \in A$, resulting a next state $s'_{t+1} \in S$; the **Reward** $r_t \in R$ is the return to be in some state s_t and perform an action a_t; O is the set of conditional probabilities of take an action a_t in some state s_t and receive an **Observation** $o_t \in \Omega$ about the next state s'_{t+1}; Finally, γ is the **Discount Factor** $\in [0, 1]$ which determines the relevance of immediate rewards over rewards in the future.

The **States** S and the **Observations** Ω are interchangeable in the context of this work as adopted in [31] and represented by:

- The water level in the tank and water consumption;
- The previous action performed (currently being applied);
- The cumulative time that the pumps operated in a horizon length of 24 hours, the month and time t;
- A binary value called water quality indicating whether on the current day of operation the system has reached a certain minimum in the tank level.

Actions A are defined by the set of binary values that represent if some pump is operating (value 1) or not (value 0) once the pumps have fixed speed. At each timestep, only one pump is running or none of them.

Finally, two **Reward** functions are designed to choose the most *efficient* pump at a given time t, as well as respect the boundary conditions of the tank level, preserve the water quality, and make use of different pumps. The immediate rewards are defined by the Equations 2 and 3:

$$r_t = e^{1/(-Q_t/kW_t)} - B * \psi + \log(1/(P + \omega)) \quad (2)$$

$$r_t = -e^{(-1/kW_t)} - B * \psi + \log(1/(P + \omega)) \quad (3)$$

where at the time t, Q_t is the flow rate through the active pump, and kW_t is the respective electricity consumption; B is the achievement of lower/upper restrictions of the water level in the tank. These lower/upper values are defined by specialists in the system and in case of not achievement, $B = 1$ in case of overflow and $B = abs(level\_of\_the\_tank_t - boundary\_condition) \in (0, 1]$ in case of (near) shortage, being $\psi = 10$, otherwise $B = 0$. Also, B has an exception, being -1 strictly for the timestep when the tank level reaches the water quality condition. P is a penalty that increases with accumulated pump run time. The penalty P increases +1 at each timestep of cumulative operating time, and for the Equation 3 it also hold for the

[2]The dataset has been provided by the IoT.H2O project (IC4WATER JPI funding)

action *(NOP)*. In the case of switching to a pump that has already been running throughout the day, ω equals 30 for the respective timestep of the switch, otherwise 1. If no pumps are running, neither $-e^{(-1/kW_t)}$ nor $e^{1/(-Q_t/kW_t)}$ are considered.

Which differentiates the Equation 2 is efficiency regarding the pumps through Q_t/kW_t, when the Equation 3 directly penalize the electricity consumption through the term e^{-1/kW_t}. As a consequence, this leads to a different perception regarding the actions. As the agent tries to maximize these rewards along its trajectory, the result is the emergence of some behavior applying the policy learned through these distinct returns. Thus, by designing two reward functions, we aim to analyze the adequacy of those behaviors regarding the goals established.

4 Deep Reinforcement Learning

The DQN [20, 1] combines Q-Learning [32] with Deep Neural Networks. The state-input can be, for instance, a set of images or continuous values, and the output is an estimation of how good is be in that state s and perform an action a, called Q-value. During the learning process, DQN tries to approximate the optimal Q* for each state-action pair performing updates through the Bellman equation. This approach achieves higher scalability compared to other methods once that is not necessary to keep a vast search space. Later, Hausknecht and Stone [31] introduced long short-term memory (LSTM) in this structure to handle partially observable environments, and van Hasselt and colleagues adopt a Double DQN [33] to tackle the optimistic nature of the original Q-Learning.

4.1 Learning Process

Using a simulator of the environment and real-world data of the water consumption at determined time t, the simulator can calculate at timestep t the values of flow Q, pressure H, and electricity consumed kW, as well the tank level at $t+1$. The dynamic of this simulator is first to define the state s and then use some strategy to choose an action to be performed. Once this action is applied, a reward is given, and the next state is perceived, constituting a transition $T = < state, action, reward, next\ state >$.

During this process, new transitions feed the *Experience Replay*. The *Experience Replay* [34] and the *Target Network* are two techniques applied in [1], to improve the performance of DQN. The former break the correlation of data, and the latter makes the learning process more stable. Transitions stored in the *replay memory* consist of a batch. Then, this batch is split into mini-batches and shuffled to break the correlation between the data. Finally, the states of these mini-batches are inputs in the neural network, which aims to approximate Q*(s, a) through the Bellman Equation 4 [8].

$$Q^*(s,a) = E[R(s,a) + \gamma max_{a'} Q'(s',a')], \qquad (4)$$

where an expectation is defined regarding the future returns, discounting it through the factor $\gamma \in [0,1]$. The Q-Learning approach establishes a convergence for the optimality, updating the Bellman equation through Equation 5.

$$Q(s,a) = Q(s,a) + \alpha[R(s,a) + \gamma max_{a'} Q'(s',a') - Q(s,a)]. \qquad (5)$$

In order to update the Q(s, a), every state-action pair is recorded and updated iteratively in the tabular form of Q-learning [32]. This approach suffers from a problem called *curse of dimensionality* [8] as the number of possible states and actions grows. This can be even more complicated when considered continuous values, that must be discretized in some way. For that, DQN combines Q-learning with neural networks as a function approximator with weights θ to estimate the Q-values. This is accomplished by minimizing the loss δ at each time step i, as shown in Equation 6.

$$\delta_i(\theta_i) = E[R(s,a) + \gamma max_{a'} Q'(s',a',\theta_{i-1}) - Q(s,a,\theta_i)]^2, \qquad (6)$$

where the weights θ_{i-1} are those fixed in the target network that in turn, are periodically updated copying weights θ. The frequency that the target network updates can be seen as a hyperparameter, being with the replay memory properties an object of study in the performance of DQN and variants [35].

The problem of traditional Q-learning is that it tends to overestimate state-action pairs out of the distribution when exploiting a fixed dataset [6]. REM mitigates this using an ensemble of models to improve the generalization through the Equation 7.

$$\delta_i(\theta_i) = E[R(s,a) + \gamma max_{a'} \sum_k \alpha_k Q'_k(s',a',\theta^k_{i-1}) - \sum_k \alpha_k Q_k(s,a,\theta^k_i)]^2, \qquad (7)$$

where for each mini-batch, α is a set of weights randomly generated such that $\sum_k \alpha_k = 1$. Thus, REM is a convex combination of Q-values, converging for itself [21].

4.2 Sample Efficiency

The performance of the family of DQN-based approaches is strongly correlated with sample efficiency. This section describes the strategies adopted to provide richer observation of the current state, improve training performance, and make better use of samples.

4.2.1 State stacking

In the original approach of DQN, n last previous states (frames) are concatenated [1]. Thus, the input provides a richer observation of the current state, such as the system's dynamic.

4.2.2 Training data scale

The state-input values have different ranges that differ substantially. It is applied normalization in both states and re-

ward values for the range [0, 1] using Equation 8. The feature is the value x, and max/min was defined considering historical observations.

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{8}$$

4.2.3 Prioritized Experience Replay (PER)

Schaul and colleagues present in [36] an improvement regarding the Experience Replay, prioritizing samples more "unexpected". In other words, samples that provide the highest values $|\delta_i|$ through the Equations 6 are those much to learn from [37]. Then, every transition in the mini-batch is associated with the correspondent magnitude of the loss, such as $T = < state, action, reward, next\ state, |\delta| >$. Finally, to balance the bias introduced by the prioritization of samples, PER applies Importance Sampling (IS) weights.

5 Imitation Learning

The imitation learning strategy *Safety through Intrinsically Motivated Imitation Learning (SIMIL)* present in this work assumes that offline data is available and online data collection is feasible. The underlying idea is to use the offline dataset distribution as a model to constrain the action selection and enhance the sample efficiency while encouraging the policy's convergence to states that lie in high-density regions under the same prior distribution.

The imitation learning strategy works as a follows: given a current state s_t and demonstrations $\mathcal{D}$, select the action a mostly applied in the *k-most* similar states to s_t in $\mathcal{D}$. For that, we make use of k-Nearest Neighbors (k-NN), where the parameter *k* can be chosen such that minimizes the distance $min \sum_{\mathcal{D}} d(\tau, \tau^{\mathcal{D}})$, regarding trajectories $\tau^{\mathcal{D}} \in \mathcal{D}$. The objective is to keep new transitions tied to the previously collected data, mitigating overestimation facing unseen state-action pairs. Finally, a reward bonus $\rho\eta(s_t)$ is added to the immediate reward according to the *Kernel Density Estimation* (KDE) for s_t through Equation 9, being ρ the importance factor for the bonus. Thus, we encourage policy convergence to states with high density under prior dataset distribution.

$$\eta(s_t) = \frac{1}{N} \sum_{i=1}^{N} K\left(\frac{s_t - s_i^{\mathcal{D}}}{h}\right). \tag{9}$$

In Equation 9, $K(s_t) \geq 0$ is the kernel that estimates the density for the current state s_t over the states $s^{\mathcal{D}}$ found in the demonstrations. The parameter h is the bandwidth that trade-off the results between balance and variance. In this work, we adopt the *k-NN* based on *Manhattan distance* once it can provide suitable metric for real-values without parameter tuning and KDE with a *gaussian kernel* from Scikit-learn [15]. The Algorithm 1 summarizes the strategy proposed.

In particular, we reduce the dimensionality of the state's representation for the meaningful features regarding the current status of the WDS and skip some of them for the *k-NN* queries. That is because the timesteps are strongly correlated, and skipping some of them reduces the computational overhead due to *k-NN* query. Thus, the state representation used to calculate the reward bonus and perform k-NN queries has the reduced form of $\phi(s_t) = < tank\ level, water\ consumption, current\ time, month >$.

Algorithme 1 : Safety through Intrinsically Motivated Imitation Learning (SIMIL)

Input : set of Q-Networks with weights θ^Q, set of Target Q'-Networks with weights $\theta^{Q'} \leftarrow \theta^Q$, replay memory $\mathcal{D}'$, demonstrations $\mathcal{D}$, frequency which update target net λ, importance factor ρ;

Output : Policy π

for $t \in \{1, 2, ...\}$ **do**
- Sample state s_t
- Select action a_t using *k-NN*(s_t) in $\mathcal{D}$
- Play (s_t, a_t), observe the reward r_t and the next state s'_t
- Calculate $\eta(s_t)$, sum it to a final reward $r'_t = r_t + \rho\eta(s_t)$
- Store transition (s_t, a_t, r'_t, s'_t) into $\mathcal{D}'$
- $s_t \leftarrow s'_t$

end

for $t \in \{1, 2, ...\}$ **do**
- Sample a mini-batch of *n* transitions from $\mathcal{D}'$
- Calculate loss $\delta(\theta^Q)$
- Perform a gradient descent step to update θ^Q
- **if** *t* ***mod*** $\lambda = 0$ **then**
  - Update the set of weights $\theta^{Q'} \leftarrow \theta^Q$
- **end**

end

6 Policy Evaluation

6.1 Experimental Setup

In this work, we aim to evaluate if (1) the proposed imitation learning strategy can generate policies that outperform offline methods baselines; (2) the proposed POMDP can obtain policies that offer a competitive performance relative to that observed in the real world. To this end, we conducted the experiments using the real-world dataset divided into one year for the learning process and one year for the evaluation. Both Offline RL methods and SIMIL use the same amount of data for learning. For accurate comparisons, all samples interact with the simulator for both training and evaluation. This means that the evaluation of the offline dataset is done through interactions with the simulator. We compare the policies BCQ, REM, and SIMIL + REM using 5 models for each reward function due to the stochasticity in the learning process [38].

6.2 Results

To analyze the performance, we call the set of policies obtained using the Equations 2 and 3 by Π_1 and Π_2 respec-

Policy	*Electricity Consumption (kW)*
REM Π_1	-1.11 $\pm$ 9.78
SIMIL + REM Π_1	-4.05 $\pm$ 1.97
BCQ Π_1	-3.54 $\pm$ 2.71
REM Π_2	4.08 $\pm$ 7.93
SIMIL + REM Π_2	-3.33 $\pm$ 5.77
BCQ Π_2	-1.40 $\pm$ 3.33

Table 1: Average electricity consumption (%) $\pm$ standard deviation compared to real-world operation.

Policy	*NOP*	*NP1*	*NP2*	*NP3*	*NP4*
Real-world	30.47	8.30	43.42	8.31	9.50
REM π_1^*	11.38	4.93	0.87	82.82	0.0
SIMIL + REM π_1^*	17.05	0.17	28.54	5.29	48.95
BCQ π_1^*	22.87	17.79	8.13	51.09	0.12
REM π_2^*	32.64	25.85	0.04	41.47	0.0
SIMIL + REM π_2^*	28.08	3.12	36.04	4.89	27.87
BCQ π_2^*	37.11	37.48	0.06	25.35	0.0

Table 2: Action distribution (%)

tively. We show in Figure 1 the min, max, and average cumulative reward along with the episodes using the 5 policies obtained. The results show that SIMIL has lower variance and competitive performance relative to cumulative rewards compared to fully-offline policies. The lower peaks in performance are mainly due to not meeting the tank level safety constraints.

The three sub-goals: electricity consumption, distribution of pump usage, and tank level are the counterparts of the policy. Thus, a suitable policy performs with lower electricity consumption/higher efficiency, reduces switches, and distributes the pump operation while respecting the tank level constraints. We show in Figure 2 the performance of policies π^* with a better average cumulative reward for Offline RL and SIMIL. Tables 1 and 2 present a comparison between the policies using as baseline the real-world statistics for the evaluation data. Table 1 compares the electricity consumption for Π regarding real-world operation while Table 2 shows the action distribution for π^*. The results show that SIMIL policies achieve competitive results with real-world operations considering electricity consumption. Finally, generally, the policies presented an operation in the safety range of tank levels.

7 Conclusions

This work presents *Safety through Intrinsically Motivated Imitation Learning (SIMIL)*, an imitation learning strategy using density-based action selection and intrinsic motivation to constrain policies to expert demonstrations. Our contribution lies in the idea that SIMIL, while retrieving expert demonstrations behavior, also allows the possibility of extrapolating it in favor of states that lies in high-density regions. That could represent a means to deploy safe deep RL approaches in real-world applications. Finally, the results show that SIMIL can lead to policies that could even outperform fully-offline methods.

We present a real-world problem called pumping scheduling for water distribution utilities as an evaluation scenario. The contributions of this work extend to this domain. The proposed reward functions lead to policies that satisfy the safety constraints, protect the assets and lead to electricity savings. The authors hope that this representation of the pumping scheduling problem can help other researchers in different WDS settings.

Acknowledgments

We would like to thank Harald Roclawski (TU Kaiserslautern), Aloysio P. M. Saliba (UFMG), Benjamin Dewals (ULiège), Anika Theis (TU Kaiserslautern), Thomas Pirard (ULiège), and Thomas Krätzig (Dr. Kraetzig) for their comments and suggestions regarding this work. The authors acknowledge FAPEMIG, Federal Ministry of Education and Research of Germany, Agence Nationale de la Recherce de France and Fonds de la Recherce Scientifique Belge, for funding this research by the project IoT.H2O (ANR-18-IC4W-0003) on the IC4Water JPI call.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[2] G. Lample and D. S. Chaplot, "Playing fps games with deep reinforcement learning," 2017.

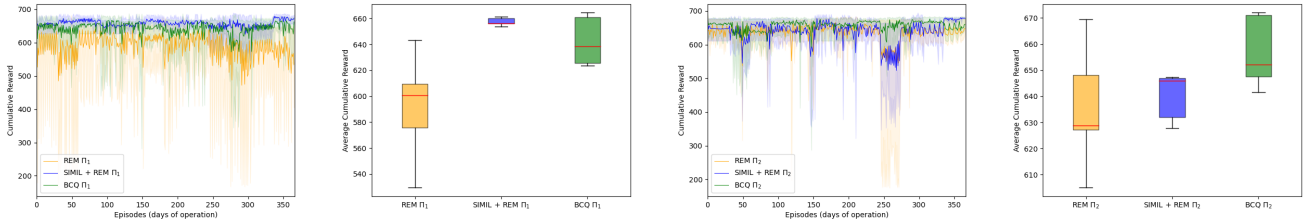
[3] Y. Yang, J. Hao, M. Sun, Z. Wang, C. Fan, and G. Strbac, "Recurrent deep multiagent q-learning for autonomous brokers in smart grid," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pp. 569–575, 7 2018.

[4] T. Wei, Y. Wang, and Q. Zhu, "Deep reinforcement learning for building hvac control," in *Proceedings of the 54th Annual Design Automation Conference 2017*, DAC '17, (New York, NY, USA), 2017.

[5] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *CoRR*, vol. abs/2005.01643, 2020.

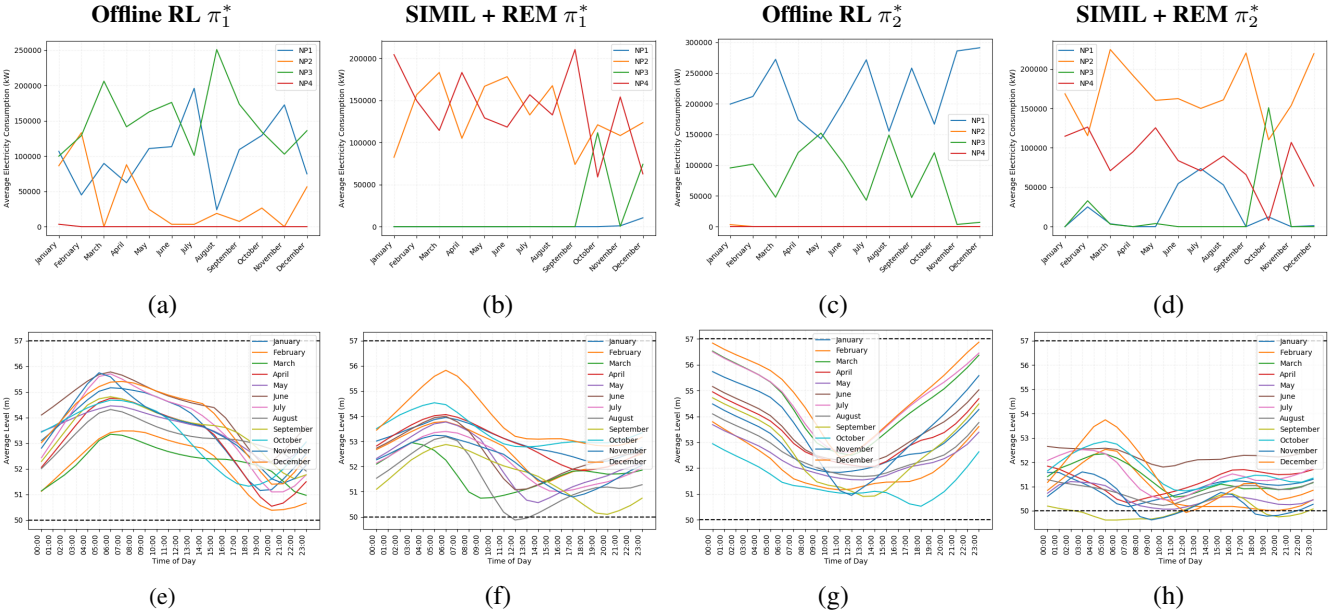
[6] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97 of *Proceedings of Machine Learning Research*, pp. 2052–2062, PMLR, 09–15 Jun 2019.

[7] N. Jaques, A. Ghandeharioun, J. H. Shen, C. Ferguson, À. Lapedriza, N. Jones, S. Gu, and R. W. Picard, "Way off-policy batch deep reinforcement learning of implicit human preferences in dialog," *CoRR*, vol. abs/1907.00456, 2019.



(a) (b) (c) (d)

Figure 1: (a) and (c) are respectively the min, max, and average cumulative reward for the set of policies Π_1 and Π_2. (b) and (d) shows the average of the cumulative rewards along with the episodes for Π_1 and Π_2.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 2: Shows the performance of the policies that achieve the highest average cumulative reward for the sets Π_1 and Π_2 for Offline RL and SIMIL + REM. (a), (b), (c), and (d) present the average electricity consumption per day. (e), (f), (g), and (h) show the average tank level per time of day.

[8] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. 2018.

[9] H. Tang, R. Houthooft, D. Foote, A. Stooke, O. Xi Chen, Y. Duan, J. Schulman, F. DeTurck, and P. Abbeel, "#exploration: A study of count-based exploration for deep reinforcement learning," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.

[10] D. Abel, A. Agarwal, F. Diaz, A. Krishnamurthy, and R. E. Schapire, "Exploratory gradient boosting for reinforcement learning in complex domains," *CoRR*, vol. abs/1603.04119, 2016.

[11] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, "Unifying count-based exploration and intrinsic motivation," in *Advances in Neural Information Processing Systems*, vol. 29, Curran Associates, Inc., 2016.

[12] D. Pathak, D. Gandhi, and A. Gupta, "Self-supervised exploration via disagreement," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97 of *Proceedings of Machine Learning Research*, pp. 5062–5071, PMLR, 09–15 Jun 2019.

[13] J. García and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 42, pp. 1437–1480, 2015.

[14] S. Singh, A. Barto, and N. Chentanez, "Intrinsically motivated reinforcement learning," in *Advances in Neural Information Processing Systems*, vol. 17, MIT Press, 2005.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python,"

Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[16] L. H. M. Costa, B. de Athayde Prata, H. M. Ramos, and M. A. H. de Castro, "A branch-and-bound algorithm for optimal pump scheduling in water distribution networks," *Water resources management*, vol. 30, no. 3, pp. 1037–1052, 2016.

[17] S.-C. Georgescu and A.-M. Georgescu, "Pumping station scheduling for water distribution networks in epanet," *UPB Sci. Bull, Series D*, vol. 77, no. 2, pp. 235–246, 2015.

[18] F. T. Abiodun and F. S. Ismail, "Pump scheduling optimization model for water supply system using awga," in *2013 IEEE Symposium on Computers Informatics (ISCI)*, pp. 12–17, 2013.

[19] G. Hajgató, G. Paál, and B. Gyires-Tóth, "Deep reinforcement learning for real-time optimization of pumps in water distribution systems," *Journal of Water Resources Planning and Management*, vol. 146, p. 04020079, nov 2020.

[20] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing atari with deep reinforcement learning," *CoRR*, vol. abs/1312.5602, 2013.

[21] R. Agarwal, D. Schuurmans, and M. Norouzi, "An optimistic perspective on offline reinforcement learning," in *Proceedings of the 37th International Conference on Machine Learning*, vol. 119 of *Proceedings of Machine Learning Research*, pp. 104–114, PMLR, 13–18 Jul 2020.

[22] S. Fujimoto, E. Conti, M. Ghavamzadeh, and J. Pineau, "Benchmarking batch deep reinforcement learning algorithms," *arXiv preprint arXiv:1910.01708*, 2019.

[23] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, G. Dulac-Arnold, J. Agapiou, J. Leibo, and A. Gruslys, "Deep q-learning from demonstrations," 2018.

[24] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6292–6299, 2018.

[25] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine, C. Finn, and T. Ma, "Mopo: Model-based offline policy optimization," in *Advances in Neural Information Processing Systems*, vol. 33, pp. 14129–14142, 2020.

[26] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims, "Morel: Model-based offline reinforcement learning," vol. 33, pp. 21810–21823, 2020.

[27] J. Schrittwieser, T. Hubert, A. Mandhane, M. Barekatain, I. Antonoglou, and D. Silver, "Online and offline reinforcement learning by planning with a learned model," 2021.

[28] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Comput. Surv.*, vol. 50, apr 2017.

[29] D. C. Bentivegna, C. G. Atkeson, and G. Cheng, "Learning tasks from observation and practice," *Robotics and Autonomous Systems*, vol. 47, no. 2, pp. 163–169, 2004. Robot Learning from Demonstration.

[30] L. Cardamone, D. Loiacono, and P. L. Lanzi, "Learning drivers for torcs through imitation using supervised methods," in *Proceedings of the 5th International Conference on Computational Intelligence and Games*, CIG'09, p. 148–155, 2009.

[31] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *AAAI Fall Symposium on Sequential Decision Making for Intelligent Agents (AAAI-SDMIA15)*, November 2015.

[32] C. J. C. H. Watkins and P. Dayan, "Q-learning," in *Machine Learning*, pp. 279–292, 1992.

[33] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," 2016.

[34] L. J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Mach. Learn.*, vol. 8, pp. 293–321, 1992.

[35] W. Fedus, P. Ramachandran, R. Agarwal, Y. Bengio, H. Larochelle, M. Rowland, and W. Dabney, "Revisiting fundamentals of experience replay," in *Proceedings of the 37th International Conference on Machine Learning*, vol. 119 of *Proceedings of Machine Learning Research*, pp. 3061–3071, PMLR, 13–18 Jul 2020.

[36] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *4th International Conference on Learning Representations, ICLR 2016*, 2016.

[37] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," 2018.

[38] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," 2018.