mathematics

MDPI

Article

Deep Reinforcement Learning for Intraday Multireservoir Hydropower Management

Rodrigo Castro-Freibott [1,*], Álvaro García-Sánchez [2,*], Francisco Espiga-Fernández [2] and Guillermo González-Santander de la Cruz [1]

[1] baobab soluciones, José Abascal 55, 28003 Madrid, Spain; guillermo.gonzalez@baobabsoluciones.es
[2] Industrial Engineering, Business Administration and Statistics Department, Escuela Técnica Superior de Ingenieros Industriales, Universidad Politécnica de Madrid, José Gutierrez Abascal 2, 28006 Madrid, Spain; francisco.espiga.fernandez@alumnos.upm.es
* Correspondence: rodrigo.castro@baobabsoluciones.es (R.C.-F.); alvaro.garcia@upm.es (Á.G.-S.)

Abstract: This study investigates the application of Reinforcement Learning (RL) to optimize intraday operations of hydropower reservoirs. Unlike previous approaches that focus on long-term planning with coarse temporal resolutions and discretized state-action spaces, we propose an RL framework tailored to the Hydropower Reservoirs Intraday Economic Optimization problem. This framework manages continuous state-action spaces while accounting for fine-grained temporal dynamics, including dam-to-turbine delays, gate movement constraints, and power group operations. Our methodology evaluates three distinct action space formulations (continuous, discrete, and adjustments) implemented using modern RL algorithms (A2C, PPO, and SAC). We compare them against both a greedy baseline and Mixed-Integer Linear Programming (MILP) solutions. Experiments on real-world data from a two-reservoir system and a simulated six-reservoir system demonstrate that while MILP achieves superior performance in the smaller system, its performance degrades significantly when scaled to six reservoirs. In contrast, RL agents, particularly those using discrete action spaces and trained with PPO, maintain consistent performance across both configurations, achieving considerable improvements with less than one second of execution time. These results suggest that RL offers a scalable alternative to traditional optimization methods for hydropower operations, particularly in scenarios requiring real-time decision making or involving larger systems.

Keywords: daily optimization; hydropower generation; multireservoir; reinforcement learning; mixed integer linear programming

MSC: 93E20; 68T07

Received: 31 October 2024
Revised: 19 December 2024
Accepted: 27 December 2024
Published: 3 January 2025

Citation: Castro-Freibott, R.; García-Sánchez, Á.; Espiga-Fernández, F.; González-Santander de la Cruz, G. Deep Reinforcement Learning for Intraday Multireservoir Hydropower Management. *Mathematics* **2025**, *13*, 151. https://doi.org/10.3390/math13010151

1. Introduction

Hydropower plays a crucial role in addressing the growing global demand for sustainable energy solutions. Hydro units form the backbone of renewable energy systems in many regions; for example, they account for 22.5% of the installed power in the Iberian market [1]. In modern hydropower systems, multiple reservoirs are often interconnected to form multireservoir stations, as illustrated in Figure 1. The efficient management of these multireservoir systems not only ensures reliable energy supply but also optimizes the use of natural resources.

This is particularly important in markets with high shares of intermittent renewables like wind and solar, in which the marginal price of energy is occasionally zero. As shown

in Figure 2, energy prices in Spain exhibit significant daily variations, highlighting the importance of strategic generation scheduling. Optimizing hydropower generation reduces unnecessary use of water when the price is low, preserving water resources for times of higher prices and lower supply. This makes hydropower optimization an essential tool for balancing the energy mix and achieving sustainability goals.

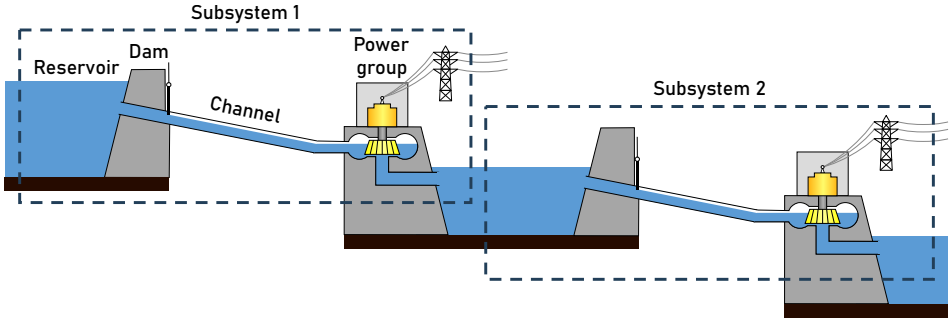


Figure 1. Elements of a multireservoir hydropower system.

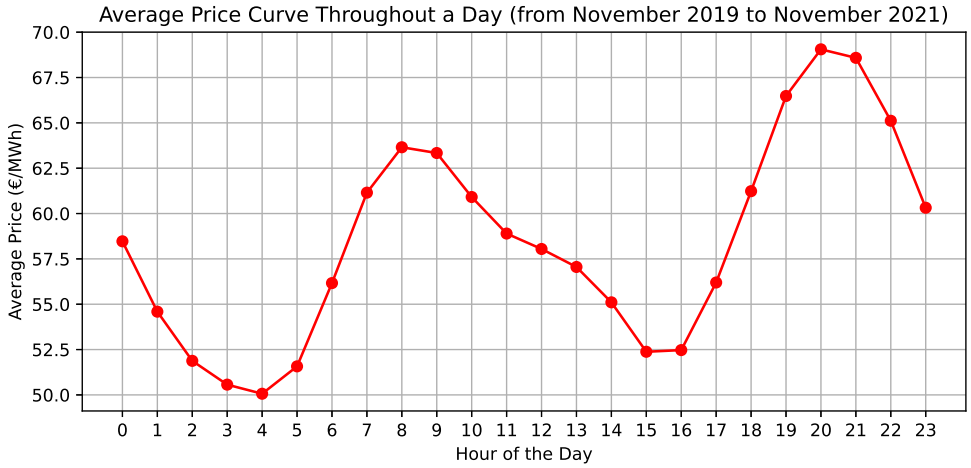


Figure 2. Average energy price at each hour of the day in Spain, from November 2019 to November 2021. Data from [2].

Reinforcement Learning (RL) has emerged as a promising approach for optimizing reservoir operations due to its ability to handle complex sequential decision-making problems. The increasing adoption of RL in this domain reflects both the growing computational capabilities and the need for more adaptive and efficient reservoir management strategies. Recent years have seen a proliferation of studies using RL for various aspects of hydropower management, from long-term planning to real-time control systems.

Most existing studies focus on long-term reservoir management with coarse temporal resolution: daily timesteps over a two-week [3] or 40-day horizon [4], 10-day intervals over a year [5], or even monthly timesteps [6]. This coarse temporal discretization allows these works to ignore short-term dynamics such as dam-to-turbine delays and gate movement restrictions.

Many previous works focus solely on maximizing energy production or minimizing spillage without considering electricity market prices [3,6,7]. Other studies concentrate on water demand satisfaction rather than power generation [8–10]. Some works address entirely different objectives: maintaining water quality [11], designing water networks [12], or optimizing proportional–integral–derivative controllers for dam gates [13,14].

While one study does incorporate energy prices and startup costs, it aggregates decisions to daily totals, avoiding the complexities of intraday operations [4]. Another

study also considers the electricity market, but it takes a different approach by focusing on price bidding strategies rather than direct operational control [1].

A significant limitation in existing literature is the reliance on discrete state-action spaces, with many works employing grid-based discretization for their Q-learning implementations [3,5,10,15]. This approach becomes impractical for intraday fine-grained control due to the curse of dimensionality.

Most studies utilize basic Q-learning with discretized spaces [3,6,15]. While some recent works have begun exploring modern algorithms such as Soft Actor-Critic (SAC), they still focus on long-term scheduling rather than intraday operations [16]. Another work explores alternative approaches using evolutionary neural networks, but maintains the focus on long-term management [17].

Our work addresses several critical gaps in the existing literature:

- Fine-grained temporal resolution: Unlike previous works that use daily timesteps [3,4], weekly intervals [16,18], or even longer periods [5,6], we model intraday operations with short intervals, incorporating dynamics such as dam-to-turbine delays.
- Comprehensive operational constraints: We handle complex operational requirements including power group dynamics and gate movement restrictions, which are typically ignored in existing literature [3,4,7,10].
- Advanced action space formulations: We evaluate three distinct action space formulations (continuous, discrete, and adjustments) implemented using modern RL algorithms (A2C, PPO, and SAC), moving beyond the traditional discrete Q-learning approaches used in most studies [3,6,10,15].
- Scalability analysis: We provide a thorough comparison of RL against MILP solutions across different system sizes, demonstrating the scalability advantages of our approach. This contrasts with previous works that either focus on single reservoirs [5,10] or simplify multi-reservoir dynamics [7].
- Real-time decision making: Our framework achieves sub-second execution times, making it suitable for real-time operational control. This separates our work from studies using traditional methods like stochastic dynamic programming that require hours of computation [18], or RL approaches that focus on longer-term planning [4,16].

While previous work has established the potential of RL for hydropower management, significant limitations remain in handling real-world operational complexities and constraints. Our work bridges this gap by developing a comprehensive framework that addresses fine-grained temporal dynamics, complex operational constraints, and scalability requirements, while maintaining practical execution times for real-world applications.

This work builds on the findings of previous research [19] on the Hydropower Reservoirs Intraday Economic Optimization (HRIEO) problem, which evaluated the performance of Mixed-Integer Linear Programming (MILP) and Particle Swarm Optimization (PSO) against a Greedy baseline. While the earlier study highlighted the scalability advantages of PSO over MILP for larger systems, it also revealed the long execution times of both methods, rendering them unsuitable for real-time operational contexts. In contrast, this study investigates Reinforcement Learning (RL) as a compelling alternative for solving HRIEO, particularly when rapid decision making is required, since an RL model can deliver solutions in sub-second execution times.

The remainder of this paper is organized as follows. Section 2 presents the hydropower optimization problem tackled in this study, incorporating all short-term phenomena and operational constraints. Section 3 introduces the two methods used for comparison: a Greedy approach and a Mixed-Integer Linear Programming model. Section 4 outlines our RL framework's design, including the simulation model and the definition of the state, reward and action. This is followed by Sections 5 and 6, which describe the available

data for this study and the validation of the simulation model. Section 7 describes the experimental setting that leads to Section 8, which indicates how the best RL models were identified. The performance of these agents against the benchmark methods is presented in Section 9. Finally, Section 10 discusses the implications of our findings and directions for future research.

2. Problem Statement

The problem addressed in this paper is the Hydropower Reservoirs Intraday Economic Optimization (HRIEO) problem introduced by [19]. This problem addresses the operation of multireservoir hydropower stations within a single day.

A multireservoir station can be divided into several subsystems, as illustrated in Figure 1. Each subsystem contains a reservoir that stores water, a dam with a gate that controls water release, a channel through which water flows, and a power group where energy is generated.

The HRIEO problem involves determining the optimal outflow from each reservoir at every time interval over a 24 h period. The goal is to maximize daily income from energy generation while adhering to the operational constraints of the system. The timing of energy generation is crucial since energy prices fluctuate significantly within the same day, as shown in Figure 2. By storing water during low-price periods and releasing it during peak-price hours, the station can maximize its revenue.

To simplify the model without losing generality, the following assumptions are made. Water released at one subsystem is assumed to be immediately available downstream without delay. The turbined flow at each subsystem's power group is modeled as a moving average of past reservoir outflows, mainly depending on the channel's length. The generated power is determined by the turbined flow using a piecewise linear function called the Power Curve function of the power group. The maximum flow through a channel depends on the reservoir's water height, governed by Bernoulli's principle, and is also modeled with a piecewise linear function called the Flow Limit function of the reservoir.

Solving the HRIEO problem requires forecasts for two inputs: hourly (or quarter-hourly) energy prices and the inflow to each reservoir from rivers or rain. For downstream reservoirs, inflow predominantly comes from the upstream subsystem's turbined water.

Time is discretized into fixed intervals, during which reservoir outflows are decided and assumed constant. Income for each interval is calculated by multiplying energy price by generated power, which depends on the turbined flow. Reservoir volumes are updated at the end of each interval using a balance of inflows and outflows.

The solution of the HRIEO problem includes outflow decisions for every reservoir at each time interval. The objective function evaluates the total income from generated power, subtracting operational costs.

The HRIEO problem includes a gate movement constraint to ensure stable reservoir outflows by restricting frequent adjustments. Once a gate is moved, it must remain in its position for a set number of periods before being moved in the opposite direction. Since the solution optimizes outflows directly, the constraint is implemented by enforcing constant outflows, which ensures no gate movement. While constant gate positions do not guarantee constant outflows due to flow dynamics, constant outflows do imply no gate movement, correctly satisfying the constraint.

The mathematical model for this problem is explained in Section 4.3, while Sections 4.4–4.6 describe how the problem can be modelled within the RL framework.

3. Benchmark Methods

This study introduces a Reinforcement Learning (RL) framework for the Hydropower Reservoirs Intraday Economic Optimization (HRIEO) problem and compares its performance with these two methods:

- A basic Greedy approach. This approach involves keeping the gates of all dams fully open at all times, meaning that each reservoir's outflow is set to its maximum value in every period. Even though this procedure does not make use of the reservoirs' capacity, it represents the previous operational strategy of the hydropower station that motivated this study.
- A Mixed-Integer Linear Programming (MILP) model. This model was developed in [19] by turning the assumptions of the HRIEO problem into MILP variables and constraints.

These benchmarks allow us to compare our RL implementation with both a simple baseline and an exact optimization model. We expect the trained RL models to outperform the baseline while requiring less computational time than the exact MILP optimization.

4. Materials and Methods

4.1. Reinforcement Learning Framework

In this study, we developed a Reinforcement Learning (RL) framework for solving the Hydropower Reservoirs Intraday Economic Optimization (HRIEO) problem. Our code and data are publicly available on GitHub: https://github.com/baobabsoluciones/flowing-basin (accessed on 20 October 2024).

The RL framework involves an environment and an agent. The agent takes as input the state of the environment and outputs an action, which changes the environment's state and generates a reward. The agent is trained throughout multiple interactions with the environment, called episodes, to maximize cumulative rewards.

In the context of the HRIEO problem, the states reflect the hydropower system's variables, the actions are mapped to outflows, and the rewards are proportional to the generated income. Every episode corresponds with a day of operating the hydropower system.

We trained the agent in this environment using three modern RL algorithms: Advantage Actor-Critic (A2C) [20], Proximal Policy Optimization (PPO) [21], and Soft Actor-Critic (SAC) [22]. The following sections describe the implemented environment in more detail.

4.2. Training Environment

Given the complexities of accessing the real system, the RL agent was trained within a simulated environment of the multireservoir hydropower system. This simulator captures the state transitions by accepting actions—specifically, dam outflows—and returning updated states and rewards.

The equations of the simulation model were linearized to ensure a fair comparison of RL with the Mixed-Integer Linear programming (MILP) model. Therefore, the simulation model matches the evaluation function used in [19] for the Particle Swarm Optimization approach. Despite the linearization, accuracy in simulating this environment was verified using real-world data.

4.3. Simulation Model

Essentially, the simulator operates by taking outflows as input and producing income as output. In period t, the simulation model processes the decided outflow ($\hat{q}^{\text{out}}_{d,t}$) for each dam d and updates multiple variables: turbine flow ($q^{\text{turb}}_{d,t}$), actual outflow ($q^{\text{out}}_{d,t}$), volume ($v_{d,t}$), power ($w_{d,t}$), and income ($i_{d,t}$). Figure 3 gives a schematic representation of the simulation model, which is explained in detail in the following paragraphs.

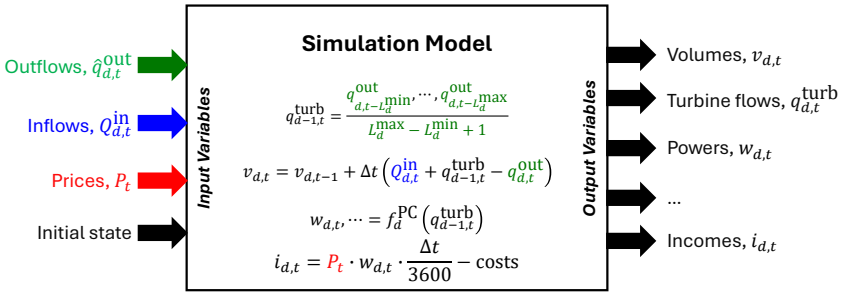


Figure 3. A schematic overview of the simulation model, outlining its most important input variables, output variables, and equations.

First, the turbine flow for each power group d is computed using Equation (1). Due to the time delay between water release and arrival at the power group, the turbine flow is determined by past outflows rather than the current one. More precisely, it is calculated as the moving average of prior outflows, spanning from the maximum lag ($L_d^{\max}$) to the minimum lag ($L_d^{\min}$), which are determined by the channel length.

$$q_{d,t}^{\text{turb}} = \frac{q_{d,t-L_d^{\min}}^{\text{out}} + \cdots + q_{d,t-L_d^{\max}}^{\text{out}}}{L_d^{\max} - L_d^{\min} + 1} \quad \forall d = 1, \ldots, D \tag{1}$$

Second, the actual outflow from each dam d is updated to ensure it complies with the system's constraints. Specifically, the actual outflow is the smallest among three values:

- The decided outflow, $\hat{q}_{d,t}^{\text{out}}$.
- The maximum permissible outflow based on the reservoir volume. This is given by the reservoir's Flow Limit function, $f_d^{\text{FL}}(v_{d,t-1})$, as shown in Figure 4.
- The maximum outflow that keeps the volume at or above the minimum limit, derived by solving Equation (3) for $q_{d,t}^{\text{out}}$ when $v_{d,t} = V_d^{\min}$.

This logic, implemented in Equation (2), ensures that the actual outflow equals the decided value unless it violates outflow or volume constraints.

$$q_{d,t}^{\text{out}} = \min\{\hat{q}_{d,t}^{\text{out}}, f_d^{\text{FL}}(v_{d,t-1}), Q_{d,t}^{\text{in}} + q_{d-1,t}^{\text{turb}} + \frac{v_{d,t-1} - V_d^{\min}}{\Delta t}\} \quad \forall d = 1, \ldots, D \tag{2}$$

Third, the reservoir volume for each dam d is updated using the volume balance equation in Equation (3). Volume increases with the natural inflow and the turbine flow from the preceding dam, and it decreases with the actual outflow.

$$v_{d,t} = v_{d,t-1} + \Delta t \left(Q_{d,t}^{\text{in}} + q_{d-1,t}^{\text{turb}} - q_{d,t}^{\text{out}} \right) \quad \forall d = 1, \ldots, D \tag{3}$$

Fourth, the number of active turbines, the limit zone indicator, and the generated power for each power group d are calculated using Equation (4). These values are given by the Power Curve function of the power group, as shown in Figure 5. This function determines the power generated by the turbine flow, the number of active turbines required, and whether the turbine flow is within a limit zone.

$$w_{d,t}, n_{d,t}, l_{d,t} = f_d^{\text{PC}}(q_{d,t}^{\text{turb}}) \quad \forall d = 1, \ldots, D \tag{4}$$

Finally, the income for each subsystem d is calculated using Equation (5). The income is derived by multiplying the generated power (in MW) by the period duration (in hours) to calculate the energy, which is then multiplied by the electricity price (in €/MWh) to

compute revenue. From this, operational costs are subtracted, which include turbine startup costs and penalties for entering limit zones.

$$i_{d,t} = P_t \cdot w_{d,t} \cdot \frac{\Delta t}{3600} - S \cdot \max\{n_{d,t} - n_{d,t-1}, 0\} - L \cdot l_{d,t} \quad \forall d = 1, \ldots, D \tag{5}$$

It is essential to note that the volume and outflow must remain within predefined thresholds, as defined in Equations (6) and (7). Any values beyond these limits are adjusted accordingly in the simulator.

$$V_d^{\min} \leq v_{d,t} \leq V_d^{\max} \quad \forall d = 1, \ldots, D \tag{6}$$

$$0 \leq q_{d,t}^{\text{out}} \leq Q_d^{\max} \quad \forall d = 1, \ldots, D \tag{7}$$

For integration with RL, the income is mapped to the reward, the simulator's state variables are mapped to the state vector, and the agent's action vector corresponds to the outflows. These mappings are detailed in the following sections.

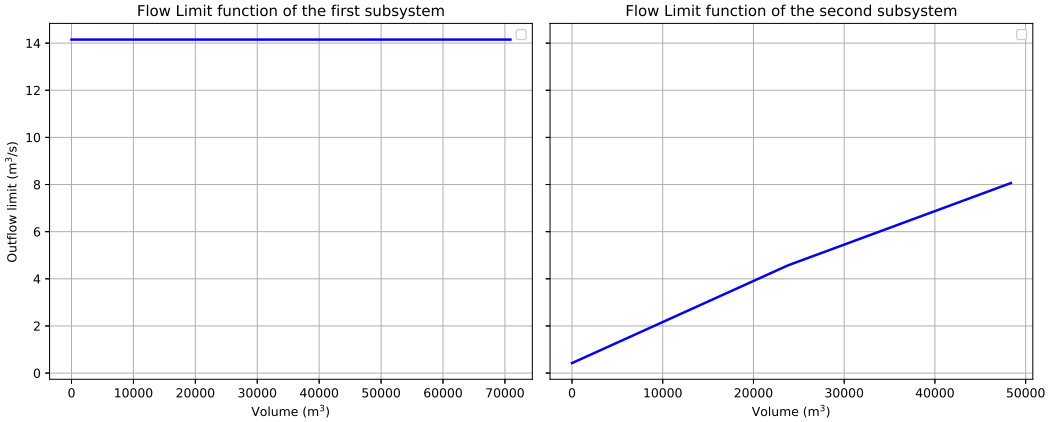


Figure 4. The Flow Limit function of each subsystem, which gives the maximum outflow of the reservoir depending on its current volume. The first subsystem does not have this limitation, so the outflow limit is simply the channel's maximum outflow.

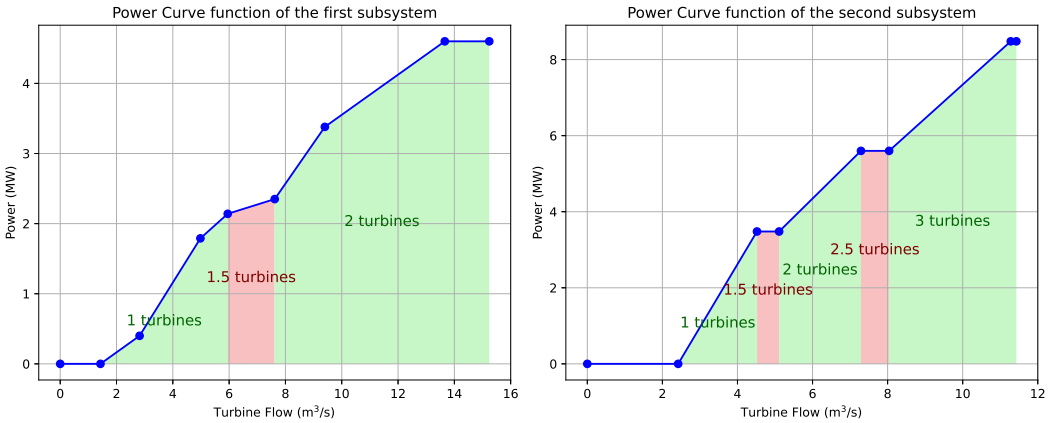


Figure 5. The Power Curve function of each subsystem, which shows the generated power and the number of active turbines based on turbine flow. Red areas correspond to limit zones where the required number of turbines is not certain.

4.4. State Space Design

The state vector summarizes the relevant information of the environment, containing all variables that are relevant for deciding the optimal outflows.

A key component is the price of energy in future periods, which allows the agent to plan ahead. In addition, the agent is provided with the following information about each subsystem of the station:

- Future natural inflows: The forecasted inflows to the reservoir from natural sources, such as rivers or rainfall, for upcoming periods.
- Subsystem metrics: The volume, turbine flow, power, and number of active turbines in the last period.
- Outflow discrepancies: The difference between the real and assigned outflow in the last period. The assigned outflow was decided by the agent and may be different to the real outflow due to volume limitations.
- Outflow variations: The outflow variations in recent periods, which inform the agent if it violated the gate movement constraint.
- Past outflows: The outflows in recent periods. These variables are included in the state because past outflows directly impact the next period due to channel delays, so they must be included to satisfy the Markov property.

All state variables were normalized to take values between 0 and 1. The subsystem metrics were normalized using their physical minimum and maximum values. Inflow was normalized using its highest historical value, whereas price, like in the reward (see Equation (11)), was normalized using the maximum value of the day.

The vector representing the state is given by Equation (8). In this equation, $\mathbf{P}_{t,t+N} = (P_t, P_{t+1}, \ldots, P_{t+N})$ is the future N energy prices and $\mathbf{s}_{d,t}$ is the vector containing the information about subsystem d, which is defined in Equations (9) and (10). This vector contains the indicated relevant variables:

- The future N natural inflows, $\mathbf{Q}^{\text{in}}_{d,t,t+N} = (Q^{\text{in}}_t, Q^{\text{in}}_{t+1}, \ldots, Q^{\text{in}}_{t+N})$.
- The last volume, turbine flow, power, and number of active turbines: $v_{d,t-1}$, $q^{\text{turb}}_{d,t-1}$, $w_{d,t-1}$, $n_{d,t-1}$.
- The last difference between real and assigned outflow, $\kappa_{d,t-1} = \hat{q}^{\text{out}}_{d,t-1} - q^{\text{out}}_{d,t-1}$.
- The last K outflow variations, $\mathbf{\Delta}_{d,t-K,t-1} = (\delta_{d,t-K}, \ldots, \delta_{d,t-1})$. Here, K is the number of periods forced to remain constant before a change in direction (gate movement constraint), and $\delta_{d,\tau} = q^{\text{out}}_{d,\tau} - q^{\text{out}}_{d,\tau-1}$ is the outflow variation on period τ.
- The past $L^{\max}_d$ outflows, $\mathbf{q}^{\text{out}}_{d,t-L^{\max}_d-1,t-1} = (q^{\text{out}}_{d,t-L^{\max}_d-1}, \ldots, q^{\text{out}}_{d,t-1})$. Outflows from periods further in the past do not influence the next turbine flow, so they are not included.

Note the agent is given the future N energy prices and natural inflows. Therefore, this parameter, N, indicates how much the agent "looks ahead." This work trained agents with different values of N.

$$\mathbf{s}_t = (\mathbf{P}_{t,t+N}, \mathbf{s}_{1,t}, \ldots, \mathbf{s}_{D,t}) \tag{8}$$

$$\mathbf{s}_{d,t} = (\boldsymbol{\sigma}_{d,t-1}, \mathbf{Q}^{\text{in}}_{d,t,t+N}, \mathbf{q}^{\text{out}}_{d,t-L^{\max}_d-1,t-1}) \tag{9}$$

$$\boldsymbol{\sigma}_{d,t-1} = (v_{d,t-1}, q^{\text{turb}}_{d,t-1}, w_{d,t-1}, n_{d,t-1}, \kappa_{d,t-1}, \mathbf{\Delta}_{d,t-K,t-1}) \tag{10}$$

4.5. Reward Function Formulation

The reward function is designed to align with the optimization objective (total income) while incorporating operational constraints (gate movement constraint). To prevent the agent from receiving artificially high rewards on days with higher energy prices, the reward is normalized using each day's maximum energy price.

The mathematical formulation is given in Equation (11), where $i_{d,t}$ represents the income of subsystem d in period t, and $h_{d,t}$ is a binary variable indicating whether the gate

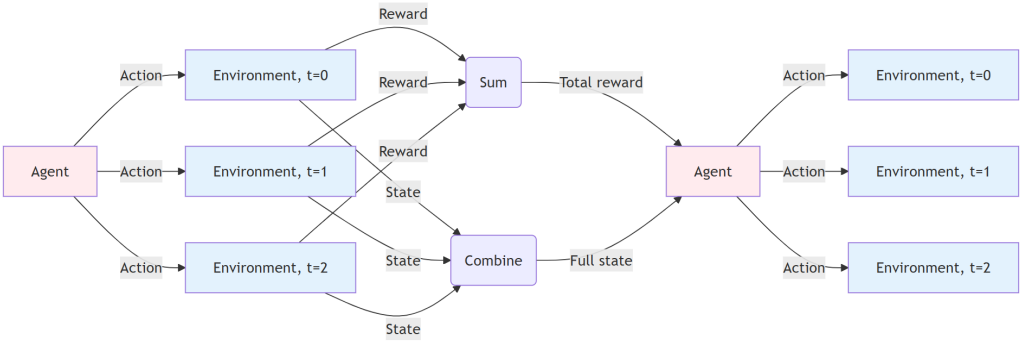
movement constraint was violated, incurring a penalty H. In addition, P_t is the energy price in period t, T is the total number of periods, and D is the total number of subsystems.

$$r_t = \frac{\sum_{d=1}^{D}(i_{d,t} - H \cdot h_{d,t})}{\max_{\tau=1,\ldots,T} P_\tau} \tag{11}$$

4.6. Action Space Design

The design of an appropriate action space is crucial for RL applications. We investigated three distinct action space formulations that differ in both the output space and the interactions with the environment, as illustrated in Figure 6.

(**a**) Flowchart of the continuous and discrete action formulations.



(**b**) Flowchart of the adjustments action formulation.

Figure 6. Flowchart describing how each agent interacts with the environment.

The Continuous and Discrete formulations (Figure 6a) follow the traditional RL paradigm, where the agent sequentially interacts with the environment at each timestep. These approaches differ in their output space: continuous values versus pre-selected discrete options. In contrast, the Adjustments formulation (Figure 6b) introduces a novel iterative refinement mechanism, enabling the agent to modify the entire solution simultaneously. The detailed description of these formulations is:

- Continuous: At every timestep t, the agent outputs normalized outflow values $a_{dt} \in [-1, 1]$ for each subsystem. These are transformed to actual outflows with Equation (12), where $Q_d^{\max}$ is the maximum outflow of channel d.

$$\hat{q}_{d,t}^{\text{out}} = \frac{a_{d,t} + 1}{2} Q_d^{\max}, \quad a_{dt} \in [-1, 1] \quad \forall d \tag{12}$$

- Discrete: At every timestep, the agent selects an outflow value for each subsystem from a pre-defined set, as represented in Equation (13). These sets of values were derived from optimal MILP solutions, focusing on historically successful operating points in each subsystem.

$$\hat{q}_{d,t}^{\text{out}} = a_{d,t}, \quad a_{d,t} \in \{Q_d^1, \ldots, Q_d^{n_d}\} \subset \mathbb{R} \quad \forall d \tag{13}$$

- Adjustments: The agent iteratively modifies the solution until it no longer improves. In the first iteration, the agent changes the solution of the Greedy approach, and it further refines its own solution in subsequent iterations. At every iteration i, the agent outputs an action $a_{dt} \in [-1, 1]$ for every subsystem d and period t, indicating how this

outflow should change with respect to the previous iteration. The resulting outflow is computed with Equation (14) and then clipped between 0 and $Q_d^{\max}$.

$$\hat{q}_{d,t}^{\text{out},i} = \hat{q}_{d,t}^{\text{out},i-1} + a_{d,t} Q_d^{\max}, \quad a_{dt} \in [-1, 1] \quad \forall d, t \tag{14}$$

Since the Adjustments action affects the entire day in each iteration, rather than a single time period, both the state and reward formulations need to be modified. First, the state now incorporates metrics for the full day rather than a single period, with updates occurring after each iteration. Second, the agent's reward is now based on the improvement in total daily reward compared to the previous iteration, instead of rewards from individual periods.

4.7. Implementation and Training

The environment was implemented using the Gymnasium framework [23], while the RL agents were implemented as multi-layer perceptrons using the Stable Baselines3 Python library [24]. Training proceeded through episodes, each corresponding to a random day from our dataset. All experiments were executed on a computer equipped with a 3 GHz Intel Core i7-9700 CPU and 16 GB of RAM. We used the default hyperparameters from Stable Baselines3 for the A2C, SAC and PPO algorithms after experiments showed limited benefits from parameter tuning.

5. Available Data

We evaluated our Reinforcement Learning (RL) agents using data from a real hydropower system with two cascading reservoirs. Table 1 details the key operational parameters of both subsystems. In addition to the parameters, the simulator requires two characteristic functions for each subsystem: the Flow Limit function and the the Power Curve function. These are shown in Figure 4 and Figure 5, respectively.

Table 1. Characteristics of each subsystem in our dataset.

Variable	First Subsystem	Second Subsystem
Minimum volume	34.045 m^3	17.117 m^3
Maximum volume	70.882 m^3	58.343 m^3
Maximum outflow	14.15 $m^3\,s^{-1}$	11.27 $m^3\,s^{-1}$
Minimum dam-to-turbine delay	15 min	45 min
Maximum dam-to-turbine delay	15 min	75 min
Maximum power	4.6 MW	8.48 MW
Turbines installed	2	3

To assess the scalability of our RL approach, we extended our experiments to a hypothetical six-reservoir system. We constructed this larger system by replicating the characteristics of the original reservoirs: subsystems 3 and 4 were modeled after subsystem 2, while subsystems 5 and 6 replicated the parameters of subsystem 1.

Our dataset comprised three years of operational records sampled at 15-minute intervals, matching our chosen time discretization. These data were combined with energy price data from OMIE [2] to enable income calculation.

6. Simulation Model Validation

The simulation model was validated to ensure it accurately represents the hydropower system by comparing its performance against historical data. The validation involved simulating the system's behavior over 500 randomly selected days, with the simulation fed

historical outflows to predict income. The accuracy was measured using Mean Percentage Error (MPE) and Mean Absolute Percentage Error (MAPE).

The results are shown in Figure 7. The MPE is negative, indicating the model systematically underestimates income. However, both the MPE and the MAPE show narrow 95% confidence bands that indicate predictable and consistent behavior.

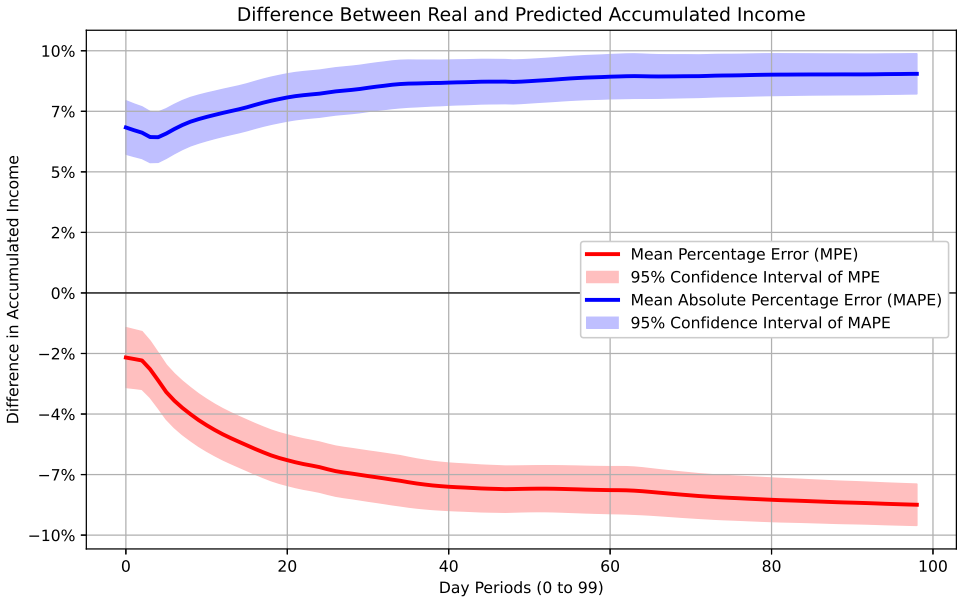


Figure 7. Mean Percentage Error (MPE) and Mean Absolute Percentage Error (MAPE) of the simulation model's prediction of the total income for each period of the day. Values were computed across 500 random days.

The consistent negative bias results from the model's conservative design. Physical system characteristics, such as reservoir maximum volumes, are set below their true values to maintain a safety margin. This reduces the risk of small mistakes in forecasts causing breaches in the actual reservoir limits. However, this careful approach makes the simulation model regularly predict lower incomes compared to historical data.

7. Experimental Setting

The experimental setting was designed to thoroughly evaluate the RL agents under varying levels of complexity. For the two-reservoir configuration, we trained each RL agent on 1000 randomly selected days from this dataset. To accommodate the increased complexity of the six-reservoir system, we extended the training duration to 2000 simulated days in this case.

Given that each day spans 96 quarter-hour periods, plus 3 additional periods at the end to account for the effect of delays, training on 1000 days corresponds to 99,000 timesteps. Exceptionally, the agents with the Adjustments action experience one entire day per timestep (instead of a single period per timestep), so they were trained for 99,000 days to keep the same number of timesteps as the other agents.

After training, the agents were tested on eleven specific days from the dataset, selected for their representativeness. These days included the driest day, the rainiest day, and nine other intermediate cases. Both the Greedy approach and the MILP model were also applied to these eleven problem instances to enable a comparison of the methods. The MILP model was solved using the Gurobi commercial solver (version 11), aiming for a final gap of 1% or lower, with a time limit set to 15 min.

The number of periods of the gate movements constraint was set to 2. Therefore, after any change in outflow (increase or decrease), gates must maintain their position for 30 min (2 periods) before allowing the opposite adjustment. If an RL agent violated this constraint

in one period, it was penalized with $H = €25$. This penalty was calibrated to be high enough to encourage compliance with the constraint while ensuring the agent remained focused on its primary objective of maximizing income. Additionally, the cost of starting a turbine (startup cost) was set at EUR 50, consistent with the value used in the real-world system on which this study is based.

Table 2 summarizes the experimental parameters and their values.

Table 2. Summary of experimental parameters and their values.

Setting	Value
Time discretization	Periods of 15 min
Training (2 dams)	99,000 timesteps (1000 days)
Training (2 dams, Adjustments action)	99,000 timesteps (99,000 days)
Training (6 dams)	198,000 timesteps (2000 days)
Training (6 dams, Adjustments action)	198,000 timesteps (198,000 days)
MILP gap	Stop when reaching 1%
MILP time limit	Time limit of 15 min
Gate movement constraint	Outflow constant for 2 periods (30 min)
Constraint violation penalty	EUR 25 per violation
Turbine startup cost	EUR 50 per startup

8. Agent Selection

Identifying the best Reinforcement Learning (RL) agents required exhaustive experimentation covering multiple alternative state transformations, reward adjustments, action formulations, and RL algorithms. This work experimented with the following alternatives:

- State transformations: In addition to normalizing state variables, as described in Section 4.4, we evaluated Principal Component Analysis (PCA) [25] for dimensionality reduction and quantile-based pseudo-discretization to achieve a uniform state value distribution. Neither approach improved agent performance.
- Look-ahead values: We experimented with different values for the number of future prices and inflows given to the agent: $N = 16$ (the next four hours), $N = 32$, $N = 64$, and $N = 96$ (the whole day). The ideal value depended on the other configurations used.
- Reward adjustments: In addition to the basic reward formula shown in Section 4.5, we explored the idea of adjusting the reward based on the income of the Greedy or MILP approaches for the same instance. However, the agents trained with adjusted rewards had a similar performance as those trained using the basic formula.
- Alternative action formulations: In addition to the Continuous, Discrete and Adjustment actions detailed in Section 4.6, we explored alternative continuous and discrete formulations. Variants included agents specifying outflows over multiple time steps, selecting turbine counts instead of outflows, and using randomized starting points for the Adjustments action. While some excelled in simplified scenarios, the three proposed action spaces performed best in the complex environment with all operational constraints.
- Alternative RL algorithms: We evaluated three alternatives: Advantage Actor-Critic (A2C) [20], Proximal Policy Optimization (PPO) [21], and Soft Actor-Critic (SAC) [22]. For discrete actions, only PPO was applicable, while all three were tested with continuous actions, where SAC produced the best-performing agents.
- Hyperparameter tuning: Using RL Zoo [26], we trained 100 agents with various hyperparameter configurations (e.g., learning rate, neural network architecture). The top-performing agents achieved similar performance to those trained with default parameters.

In total, we trained 199 agents over 945 computing hours, excluding time spent on hyperparameter tuning. All agent configurations and weights are available in our repository: https://github.com/baobabsoluciones/flowing-basin (accessed on 20 October 2024).

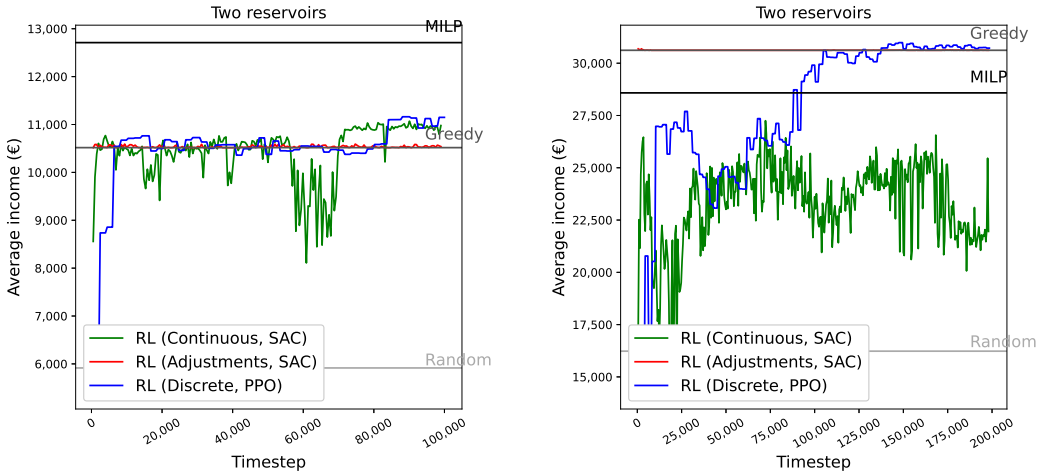
The top three agents from these experiments are:

- Continuous, SAC: Agent using the Continuous action, taking the next $N = 16$ energy prices and inflows as input, and trained with the SAC algorithm.
- Adjustments, SAC: Agent using the Adjustments action, taking all forecasted prices and inflows for the day as input, and trained with the SAC algorithm.
- Discrete, PPO: Agent using the Discrete action, taking the next $N = 32$ energy prices and inflows as input, and trained with the PPO algorithm.

Each of these three agents underwent three independent training runs on two different systems: one with two reservoirs and another with six reservoirs. The following results present the best-performing run from each agent across both systems.

9. Results

The training curves of the top three agents are represented in Figure 8. These graphs show the performance of each agent throughout its training, measured as its average income on the eleven testing instances.



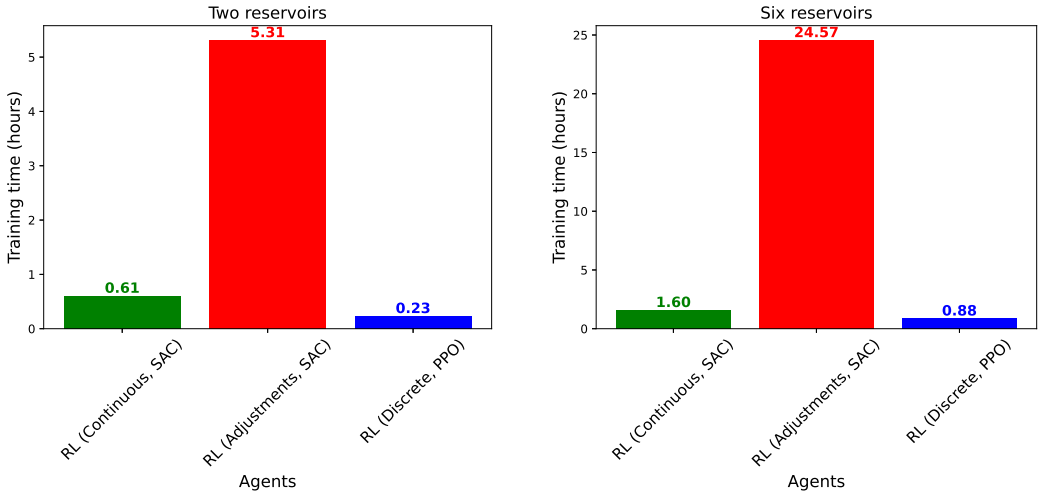
(**a**) Training curves with two reservoirs. (**b**) Training curves with six reservoirs.

Figure 8. Trainingcurve of selected agents: (**a**) System with two reservoirs and (**b**) System with six reservoirs.

By inspecting Figure 8, we notice most agents start with random performance, except for the Adjustments SAC agent, which begins at the Greedy baseline level (because of how the Adjustments action is defined), but shows minimal improvement thereafter. The Discrete PPO agent is the strongest performer, steadily improving beyond the Greedy baseline in both two and six reservoirs. The Continuous SAC agent displays more volatile performance, particularly in the six-reservoir system.

The training times of the agents are plotted in Figure 9. The Continuous SAC and Discrete PPO agents took less than one hour to train in the two-reservoir system, and less than two hours in the six-reservoir system. The Adjustments SAC agent shows notably longer training times because it simulates entire days per timestep, while the other agents process a single period per timestep.

Notably, training times for the six-reservoir system are generally more than three times longer than for the two-reservoir system. This increase is due to several factors: the simulator takes more time to predict states because of the system's added complexity; the larger input and output spaces require bigger neural networks, which use more time for inference and training; and agents are trained for twice as many timesteps, as outlined in Table 2.



(**a**) Training times with two reservoirs. (**b**) Training times with six reservoirs.

Figure 9. Training time of selected agents: (**a**) system with two reservoirs and (**b**) system with six reservoirs.

The average performance of each RL agent and the MILP model (Gurobi) over the Greedy approach is given by Table 3. The table presents the average income, execution time, proportion of periods violating the gate movement constraint, and the average MILP final gap across the eleven testing distances. The average income is given as a percentage increase over Greedy's average income.

Table 3. Performance comparison of different methods for two- and six-reservoir systems.

Two Reservoirs				
Method	**Income (Over Greedy)**	**Time (s)**	**Violations (%)**	**Final Gap**
Greedy	+0.0%	<1	15.00	
MILP	+20.9%	592	0.00	3.3%
RL (Continuous, SAC)	+3.4%	<1	14.97	
RL (Adjustments, SAC)	+0.6%	<1	16.48	
RL (Discrete, PPO)	+4.3%	<1	7.02	
Six Reservoirs				
Method	**Income (Over Greedy)**	**Time (s)**	**Violations (%)**	**Final Gap**
Greedy	+0.0%	<1	5.60	
MILP	−6.7%	775	0.00	43.3%
RL (Continuous, SAC)	−15.5%	<1	14.92	
RL (Adjustments, SAC)	+0.1%	<1	6.60	
RL (Discrete, PPO)	+0.1%	<1	6.69	

Studying Table 3, we notice the MILP always satisfies the constraint, since it is included within the formulation of the model. The MILP's performance is superior (+20.9% over Greedy with only 3.3% final gap) in the two-reservoir system, but it requires 592 s of computational time. In contrast, RL agents achieve a moderate improvement (+4.27% with the Discrete PPO agent) in less than one second, enabling quicker decision making. When scaling to the six-reservoir system, the MILP's performance degrades below the Greedy baseline (−6.7% with a final gap of 43.3%), while RL agents maintain a slight improvement (+0.1% with both the Discrete PPO and the Adjustments SAC agents).

The disaggregated results are presented in Table 4, which shows the performance over the Greedy approach in every problem instance tested, from the driest day (lowest inflow)

to the rainiest day (highest inflow). We notice the relative performance of each approach depends on the conditions of the solved instance:

- MILP: For two reservoirs, this approach gives near optimal solutions (3.3% average final gap, as seen in Table 3). These solutions are significantly better than the Greedy baseline in dry conditions (+464% in driest day). However, when ample water is available, the optimal solution involves leaving the gates open, aligning with the Greedy approach (+0% on rainiest day). In the six-reservoir system, the approach maintains strong performance in dry conditions (+258%) but significantly underperforms in moderate conditions (−19%), leading to a lower average income (−6.7%).
- Continuous, SAC: With two reservoirs, this agent underperforms the Greedy approach in dry instances (−30%) but achieves moderate improvements under normal to high inflows (+19%), leading to an overall higher average income (+3.4%). However, the agent frequently underperforms with six reservoirs, resulting in an average income that is considerably lower than the Greedy approach (−15.5%).
- Adjustments, SAC: Due to its action space design, this agent uses the Greedy solution as its starting point and rarely deviates from it. In both system sizes, it achieves only small improvements (+1%) under dry conditions, except for one notable spike (+66%) with two reservoirs.
- Discrete, PPO: With two reservoirs, this agent maintains a stable performance over the Greedy approach on both dry days (+139%) and moderate days (+10%) and, although its performance diminishes for rainy days (−10%), it achieves a higher overall average income (+4.3%). The same trend appears with six reservoirs, where the agent outperforms on dry days (+12%) and moderate days (+27%) but underperforms on rainy days (−23%), leading to an improvement in average income of only +0.1%.

In summary, we have seen that the MILP approach demonstrates superior optimization capabilities in simpler, low-inflow scenarios, but struggles to scale effectively to larger systems, particularly under moderate conditions. RL agents, while achieving only moderate improvements over the Greedy baseline, offer significant advantages in computational speed and scalability. Among them, the Discrete PPO agent stands out for its consistent performance across varying inflow conditions and system sizes, making it the most versatile choice for real-time decision making in multi-reservoir systems.

Table 4. Performance comparison of the best agents against the Greedy approach across eleven selected days, ranging from the driest to the rainiest. Values under the Greedy column represent incomes (in EUR), while all other cells show the percentage difference (%) in income relative to the Greedy approach.

	Two Reservoirs					Six Reservoirs				
Day	**Greedy**	**MILP**	**Cont. SAC**	**Adj. SAC**	**Disc. PPO**	**Greedy**	**MILP**	**Cont. SAC**	**Adj. SAC**	**Disc. PPO**
Driest	212	+464%	+0%	+1%	−1%	1176	+258%	−12%	+1%	+12%
	158	+1038%	−30%	+66%	+139%	1362	+289%	+1%	+0%	+7%
	1874	+116%	+16%	+0%	+86%	8745	+38%	−1%	+0%	+13%
	16,913	+31%	−1%	+3%	+18%	52,107	−38%	−44%	+0%	+0%
	27,133	+10%	+0%	+0%	−9%	78,465	−10%	−35%	+1%	+4%
Middle	4503	+47%	+8%	+0%	+10%	10,561	−19%	−11%	+0%	+27%
	19,295	+17%	+5%	+0%	+7%	49,734	−37%	+7%	+0%	+14%
	6144	+52%	+19%	+0%	+21%	20,765	+36%	+4%	+0%	−4%
	9604	+14%	+3%	+0%	+5%	29,663	+13%	−2%	+0%	−7%
	18,065	+7%	+6%	+0%	+0%	53,303	+3%	−1%	+0%	−8%
Rainiest	11,784	+0%	+0%	+0%	−10%	30,914	+7%	−12%	+0%	−23%
Average	10,517	20.9%	+3.4%	+0.6%	+4.3%	30,618	−6.7%	−15.5%	+0.1%	+0.1%

10. Discussion

Our experimental results provide additional insights into the application of Reinforcement Learning (RL) to the Hydropower Reservoirs Intraday Economic Optimization (HRIEO) problem, emphasizing the trade-offs between computational efficiency, performance, and scalability.

By evaluating scalability across larger systems rather than focusing solely on single reservoirs [5,10], our findings uncover a critical insight. While the MILP approach consistently outperformed in the two-reservoir system, its performance declined significantly when scaled to the six-reservoir system (falling 19% below the Greedy baseline under moderate weather conditions). In contrast, the selected RL agent sustained performance well above the baseline (+27% in the same scenario) while requiring less than a second of computation, compared to several minutes for MILP. This highlights the superior scalability and computational efficiency of RL compared to exact optimization methods.

Our analysis of RL algorithms and action spaces underscores the importance of algorithm selection and action formulation in achieving optimal performance. Unlike previous studies relying on Q-learning with discretized spaces [3,6,15], we test multiple algorithms and action formulations, reaching the best results with Proximal Policy Optimization (PPO) and a discrete action space. This consistently outperformed other RL configurations and, in the six-reservoir system, it achieved considerably better results than the MILP model under moderate inflow scenarios.

The incorporation of hydropower operational constraints in the RL framework is also a significant methodological contribution. While earlier works either ignored operational constraints [10] or used simplified models [4], our study demonstrates that RL agents can effectively learn to respect complex operational constraints through appropriate reward shaping. The discrete PPO formulation achieved particularly low violation rates, suggesting that discretization can aid in constraint satisfaction.

These results suggest that RL, particularly with discrete action spaces and modern algorithms like PPO, offers a viable and scalable alternative to traditional optimization methods for hydropower operations, especially in scenarios requiring real-time decision making or involving larger systems.

Future research should explore advanced reward shaping techniques for more effective constraint enforcement and investigate the integration of stochastic elements to handle uncertainties in inflows and energy prices. Additionally, hybrid approaches that combine the exact optimization strengths of MILP with the scalability and adaptability of RL could further enhance performance in complex, real-world hydropower optimization scenarios.

Author Contributions: Conceptualization, G.G.-S.d.l.C., Á.G.-S., R.C.-F. and F.E.-F.; methodology, Á.G.-S., F.E.-F. and R.C.-F.; software, R.C.-F. and F.E.-F.; validation, G.G.-S.d.l.C., Á.G.-S. and F.E.-F.; formal analysis, R.C.-F.; investigation, R.C.-F.; resources, Á.G.-S. and G.G.-S.d.l.C.; data curation, R.C.-F.; writing—original draft preparation, R.C.-F.; writing—review and editing, Á.G.-S. and F.E.-F.; visualization, R.C.-F.; supervision, Á.G.-S. and F.E.-F.; project administration, Á.G.-S.; funding acquisition, Á.G.-S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported both by the project Project IA4TES (Advanced Intelligent Technologies for Sustainable Energy Transition) with file number TSI-100408-2021 from the 2021 AI R&D Missions Program, within the framework of the Spain Digital Agenda 2025 and the National Artificial Intelligence Strategy, funded by the Recovery, Transformation, and Resilience Plan and co-financed with European funds from the Recovery and Resilience Facility (RRF), Next Generation EU and the Spanish Agencia Estatal de Investigación for the support provided by the Ministerio de Ciencia e Innovación of Spain (Grant Ref. PID2022-137748OB-C31 funded by MCIN/AEI/10.13039/501100011033) and "ERDF A way of making Europe".

Data Availability Statement: The original code and data presented in the study are openly available in GitHub at https://github.com/baobabsoluciones/flowing-basin (accessed on 20 October 2024).

Conflicts of Interest: Authors Rodrigo Castro-Freibott and Guillermo González-Santander de la Cruz were employed by the company baobab soluciones. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest. The company baobab soluciones had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

RL	Reinforcement Learning
MILP	Mixed-Integer Linear Programming
A2C	Advantage Actor-Critic
PPO	Proximal Policy Optimization
SAC	Soft Actor-Critic
HRIEO	Hydropower Reservoirs Intraday Economic Optimization
MPE	Mean Percentage Error
MAPE	Mean Absolute Percentage Error
OMIE	Operador del Mercado Ibérico de Energía
CPU	Central Processing Unit
RAM	Random Access Memory
PCA	Principal Component Analysis

References

1. Sousa, J.C.; Saraiva, J.T. Simulation of hydro power plants in the iberian market using an agent-based model and q-learning. In Proceedings of the 2020 17th International Conference on the European Energy Market (EEM), Stockholm, Sweden, 16–18 September 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6.
2. OMIE. OMIE: Operador del Mercado Ibérico de Energía. 2023. Available online: https://www.omie.es/ (accessed on 2 February 2023).
3. Zarghami, M. Short Term Management of Hydro-Power System Using Reinforcement learning. Ph.D. Thesis, École de Technologie Supérieure, Montreal, QC, Canada, 2018.
4. Matheussen, B.V.; Granmo, O.C.; Sharma, J. Hydropower optimization using deep learning. In Proceedings of the Advances and Trends in Artificial Intelligence, from Theory to Practice: 32nd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2019, Graz, Austria, 9–11 July 2019; Proceedings 32; Springer: Berlin/Heidelberg, Germany, 2019; pp. 110–122.
5. Xu, W.; Meng, F.; Guo, W.; Li, X.; Fu, G. Deep reinforcement learning for optimal hydropower reservoir operation. *J. Water Resour. Plan. Manag.* **2021**, *147*, 04021045. [CrossRef]
6. Lee, J.H.; Labadie, J.W. Stochastic optimization of multireservoir systems via reinforcement learning. *Water Resour. Res.* **2007**, *43*, 5627. [CrossRef]
7. Xu, W.; Zhang, X.; Peng, A.; Liang, Y. Deep reinforcement learning for cascaded hydropower reservoirs considering inflow forecasts. *Water Resour. Manag.* **2020**, *34*, 3003–3018. [CrossRef]
8. Bouchart, F.; Chkam, H. A Reinforcement Learning Model For The Operation Of Conjunctive Use Schemes. *WIT Trans. Ecol. Environ.* **1998**, *26*, 11. [CrossRef]
9. Castelletti, A.; Corani, G.; Rizzoli, A.E.; Soncini-Sessa, R.; Weber, E. Reinforcement learning in the operational management of a water system. In *IFAC Workshop on Modeling and Control in Environmental Issues*; Keio University: Yokohama, Japan, 2002; pp. 325–330.
10. Mahootchi, M.; Tizhoosh, H.; Ponnambalam, K. Opposition-Based Reinforcement Learning in the Management of Water Resources. In Proceedings of the 2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning, Honolulu, HI, USA, 1–5 April 2007; pp. 217–224. [CrossRef]
11. Rieker, J.D.; Labadie, J.W. An intelligent agent for optimal river-reservoir system management. *Water Resour. Res.* **2012**, *48*, 11958. [CrossRef]

12. Romero, M.; Yamanaka, V.; Morales, E. Multi-objective optimization of water-using systems. *Eur. J. Oper. Res.* **2007**, *181*, 1691–1707. [CrossRef]
13. Zhang, G.; Hu, W.; Cao, D.; Jing, S.; Huang, Q.; Chen, Z. Deep Reinforcement Learning Based Optimization Strategy for Hydro-Governor PID Parameters to Suppress ULFO. In Proceedings of the 2020 5th International Conference on Power and Renewable Energy (ICPRE), Shanghai, China, 12–14 September 2020; pp. 446–450. [CrossRef]
14. Enyekwe, I.; Bai, W.; Lee, K.; Nag, S. Deep Reinforcement Learning-Based Governor for Pumped Storage Hydropower. In Proceedings of the 9th World Congress on Electrical Engineering and Computer Systems and Sciences (EECSS'23), London, UK, 3–5 August 2023. [CrossRef]
15. Castelletti, A.; Galelli, S.; Restelli, M.; Soncini-Sessa, R. Tree-based reinforcement learning for optimal water reservoir operation. *Water Resour. Res.* **2010**, *46*, 8898. [CrossRef]
16. Riemer-Sørensen, S.; Rosenlund, G.H. Deep Reinforcement Learning for Long Term Hydropower Production Scheduling. *arXiv* **2020**, arXiv:2012.06312.
17. Dariane, A.B.; Moradi, A.M. Comparative analysis of evolving artificial neural network and reinforcement learning in stochastic optimization of multireservoir systems. *Hydrol. Sci. J.* **2016**, *61*, 1141–1156. [CrossRef]
18. Helseth, A.; Fodstad, M.; Mo, B. Optimal Medium-Term Hydropower Scheduling Considering Energy and Reserve Capacity Markets. *IEEE Trans. Sustain. Energy* **2016**, *7*, 934–942. [CrossRef]
19. Castro-Freibott, R.; Gerbolés, C.G.C.; García-Sánchez, A.; Ortega-Mier, M. MILP and PSO approaches for solving a hydropower reservoirs intraday economic optimization problem. *Cent. Eur. J. Oper. Res.* **2024**. [CrossRef]
20. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.P.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous Methods for Deep Reinforcement Learning. *arXiv* **2016**, arXiv:1602.01783.
21. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal Policy Optimization Algorithms. *arXiv* **2017**, arXiv:1707.06347.
22. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018.
23. Towers, M.; Kwiatkowski, A.; Terry, J.; Balis, J.U.; De Cola, G.; Deleu, T.; Goulão, M.; Kallinteris, A.; Krimmel, M.; KG, A.; et al. Gymnasium: A Standard Interface for Reinforcement Learning Environments. *arXiv* **2024**, arXiv:2407.17032.
24. Raffin, A.; Hill, A.; Gleave, A.; Kanervisto, A.; Ernestus, M.; Dormann, N. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *J. Mach. Learn. Res.* **2021**, *22*, 1–8.
25. Pearson, K. On lines and planes of closest fit to systems of points in space. *London Edinburgh Dublin Philos. Mag. J. Sci.* **1901**, *2*, 559–572. [CrossRef]
26. Raffin, A. RL Baselines3 Zoo. 2020. Available online: https://github.com/DLR-RM/rl-baselines3-zoo (accessed on 14 April 2024).