RIO

PUC

Dissertação de Mestrado

Reinforcement Learning and Optimization Applied to the Hydrothermal Dispatch Problem

Gabriel Vidigal de Paula Santos

Pontifícia Universidade Católica do Rio de Janeiro
Centro Técnico Científico
Departamento de Engenharia Elétrica

Rio de Janeiro, 29 de abril de 2025

Pontifícia
Universidade
Católica do
Rio de Janeiro

Dissertação de Mestrado

Reinforcement Learning and Optimization Applied to the Hydrothermal Dispatch Problem

Gabriel Vidigal de Paula Santos

Orientação: Prof. Alexandre Street
Coorientação: Dr. Joaquim Dias Garcia

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica pelo programa de Pós-Graduação em Engenharia Elétrica, no Departamento de Engenharia Elétrica da PUC-Rio.

Rio de Janeiro, 29 de abril de 2025

Pontifícia
Universidade
Católica do
Rio de Janeiro

Dissertação de Mestrado

Reinforcement Learning and Optimization Applied to the Hydrothermal Dispatch Problem

Gabriel Vidigal de Paula Santos

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica. Aprovada pela Comissão examinadora abaixo:

Prof. Alexandre Street
Orientador
Departamento de Engenharia Elétrica – PUC-Rio

Dr. Joaquim Dias Garcia
Coorientador
PSR

Prof. Bernardo Freitas
Departamento de Matemática - FGV

Prof. Wouter Caarls
Departamento de Engenharia Elétrica – PUC-Rio

Rio de Janeiro, 29 de abril de 2025

Pontifícia
Universidade
Católica do
Rio de Janeiro

Gabriel Vidigal de Paula Santos
Graduou-se em engenharia elétrica pela PUC-Rio.

Ficha Catalográfica

Santos, Gabriel Vidigal de Paula

Reinforcement Learning and optimization applied to the hydrothermal dispatch problem / Gabriel Vidigal de Paula Santos ; orientador: Alexandre Street ; coorientador: Joaquim Dias Garcia, 2025.

41 f: il. Color. ; 30 cm

Dissertação (mestrado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica.

Inclui bibliografia

1. Engenharia Elétrica – Teses. 2. Despacho hidrotérmico. 3. Aprendizado por reforço. 4. Rastreamento de estado. 5. Fluxo de potência ótimo. I. Street, Alexandre. II. Garcia, Joaquim Dias. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. IV. Título.

CDD: 621.3

Acknowledgments

To my adviser Professor Alexandre Street for the insights provided.

To my co-advisor and boss Joaquim for the guidance, help and patience.

To my boss Raphael for all the help and late night revisions.

To my colleague Rafael for all the discussions, ideas and partnership during these years.

To my colleagues Guilherme, Carolina, Pedro, Vinicius and Bianca for al we learned with each other.

To my parents Ana Carolina and Sergio for all the oportunities that led me here.

Last but not least, to my girlfriend Paula. Without your kind words and understanding, I would not be here.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Abstract

Vidigal, Gabriel; Street, Alexandre (Advisor); Dias Garcia, Joaquim (Co-Advisor). **Reinforcement Learning and Optimization Applied to the Hydrothermal Dispatch Problem**. Rio de Janeiro, 2025. 41p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

The optimal scheduling of power plants in a hydrothermal system is a complex problem due to inflow uncertainty, network constraints, and long-term time coupling. The most commonly used approach to solve it is multistage stochastic optimization, particularly Stochastic Dual Dynamic Programming (SDDP). However, the assumptions of time independence and convexity challenge the application of this method in situations where nonlinear patterns are observed, thus requiring simplifications in practice. In this work, we propose a combination of optimization and Reinforcement Learning (RL) to better approximate the nonlinear problem, considering Alternating Current (AC) network constraints and an unknown data-generating process. For each stage, an optimization problem minimizes operating costs while tracking target volumes for each reservoir. These target values are obtained from a neural network trained via RL. The reward function is defined based on the system's operating cost and simulated using actual data and the AC network model. We benchmark the proposed model against SDDP using data from the Bolivian system. The results show the proposed approach is able to solve large problems in real world systems with costs within 5-25% of the benchmark solution.

Keywords

Hydrothermal Dispatch; Reinforcement Learning; State Tracking; Optimal Power Flow.

Resumo

Vidigal, Gabriel; Street, Alexandre; Dias Garcia, Joaquim. **Aprendizado por Reforço e Otimização Aplicados ao Problema de Despacho Hidrotérmico**. Rio de Janeiro, 2025. 41p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

A programação ótima de usinas em um sistema hidrotérmico é um problema complexo devido à incerteza nas afluências, às restrições da rede e ao acoplamento temporal de longo prazo. A abordagem mais comumente utilizada para resolvê-lo é a otimização estocástica em múltiplos estágios, em particular a Programação Dinâmica Dual Estocástica (SDDP, em inglês). No entanto, as suposições de independência temporal e convexidade desafiam a aplicação desse método em situações onde padrões não lineares são observados, exigindo simplificações na prática. Neste trabalho, propomos uma combinação de otimização com Aprendizado por Reforço (RL, em inglês) para melhorar a aproximação do problema não linear, considerando as restrições da rede em Corrente Alternada (AC, em inglês) e um processo gerador de dados desconhecido. Para cada estágio, um problema de otimização minimiza os custos operacionais enquanto acompanha volumes-alvo para cada reservatório. Esses valores-alvo são obtidos por meio de uma rede neural treinada via RL. A função de recompensa é definida com base no custo operacional do sistema e simulada com dados reais e o modelo AC da rede. Comparamos o modelo proposto com o SDDP utilizando dados do sistema elétrico Boliviano. Os resultados mostram que o método proposto é capaz de resolver problemas de grande porte em sistemas reais com custos dentro de 5-25% da solução de referência.

Palavras-chave

Despacho Hidrotérmico; Aprendizado por Reforço; Rastreamento de Estado; Fluxo de Potência Ótimo.

Table of contents

1 Introduction **12**

2 The Hydrothermal Dispatch Problem **15**
2.1 The SDDP Algorithm and the PAR(p) Model 16
2.2 The Network Models 17

3 The Reinforcement Learning Framework **19**

4 The Proposed Algorithm **22**
4.1 Deep Deterministic Policy Gradient 23
4.2 The Single Stage Optimization Problem 25

5 Computational Experiments **27**
5.1 Parameter tuning 28
5.2 Case Study 1 - The Inflow Model 29
5.3 Case Study 2 - The Network Model 32

6 Conclusion **37**

7 Bibliography **39**

List of figures

Figure 3.1 The RL Framework 19

Figure 4.1 Information Flow in the Model 22

Figure 5.1 Case 1: Total Operating Costs 31
Figure 5.2 Case 1: Total Operating Costs with Confidence Interval 31
Figure 5.3 Case 1: Operating Costs with Quantiles 31
Figure 5.4 Case 1: Operating Costs per Seed 32
Figure 5.5 Case 1: Hydro Generation with Quantiles 32
Figure 5.6 Case 1: Reservoir Volumes with Quantiles 32
Figure 5.7 Case 1: Reservoir Volumes per Seed 33
Figure 5.8 Case 1: Training Loss 33
Figure 5.9 Case 2: Total Operating Costs 34
Figure 5.10 Case 2: Total Operating Costs with Confidence Intervals 34
Figure 5.11 Case 2: Operating Costs with Quantiles 34
Figure 5.12 Case 2: Operating Costs per Seed 35
Figure 5.13 Case 2: Hydro Generation with Quantiles 35
Figure 5.14 Case 2: Hydro Volume with Quantiles 36
Figure 5.15 Case 2: Hydro Generation per Seed 36
Figure 5.16 Case 2: Training Loss 36

List of tables

Table 5.1	System Dimensions and Study Parameters	27
Table 5.2	Modeling choices	28
Table 5.3	Neural network architecture	28
Table 5.4	DDPG Hyperparameters	28
Table 5.5	Case 1 - Experimental Setup	30
Table 5.6	Case 1 - Normalized Total Operating Costs	30
Table 5.7	Case 2 - Normalized Total Operating Costs	33

List of Abreviations

SDDP – Stochastic Dual Dynamic Programming
RL – Reinforcement Learning
AC-OPF – Alternating Current Optimal Power Flow
AC – Alternating Current
IEA – International Energy Agency
TS-LDR – Two-stage Linear Decision Rules
LDR – Linear Decision Rules
TS-DDR – Two-stage Deep Decision Rules
PAR(p) – Periodic Auto-Regressive
DC – Direct Current
MDP – Markov Decision Process
DDPG – Deep Deterministic Policy Gradient
TD3 – Twin Delayed DDPG
SAC – Soft Actor-Critic

1
Introduction

Hydropower plants are the world's largest source of low-carbon electricity. According to a report from the International Energy Agency (IEA) (IEA, 2024), hydropower accounted for approximately one-seventh of global electricity generation in 2023, totaling almost 4,000 TWh. These plants are also a primary source of generation flexibility. However, the time-coupling effect introduced by the storage capacity of large reservoirs necessitates long-term simulations to properly evaluate the opportunity cost of water. This challenge has been successfully addressed using Stochastic Dual Dynamic Programming (SDDP) (PEREIRA; PINTO, 1991), which has been implemented in various forms across countries such as Brazil, Chile, Bolivia, and other Latin American nations, as well as the United States, Norway, and Vietnam. Nevertheless, the use of large-scale optimization models relies on critical assumptions—particularly time independence and convexity—that require significant approximations, as discussed in (BRIGATTO; STREET; VALLADAO, 2017) and (ROSEMBERG et al., 2022). Another key challenge faced by system operators is the accuracy and bias in inflow forecasts (BRIGATTO et al., 2024). As a result, an important and ongoing research question is how to better approximate long-term hydrothermal planning models to reflect the complexities of real-world operations.

Several works propose alternative simplifications to the problem. In (BODUR; LUEDTKE, 2022), a Two-stage Linear Decision Rule (TS-LDR) method is presented. In this approach, reservoir volumes are represented using Linear Decision Rules (LDRs), restricting decisions at each stage to be affine functions of the observed uncertain parameters. The remaining variables are left unchanged, resulting in a two-stage optimization problem that can be approximately solved one stage at a time, given the reservoir volumes. This method does not rely on the stagewise independence assumption but is susceptible to overfitting and constrained by the affine structure of the decision rules. To address this, (NAZARE; STREET, 2023) proposes a regularized version of the method to reduce overfitting and improve out-of-sample performance. Another variation, Two-stage Deep Decision Rules (TS-DDR), is introduced in (ROSEMBERG et al., 2025), and is more suitable for non-

convex environments such as the hydrothermal dispatch problem. In this framework, the affine representation of decision rules is replaced by deep neural networks.

Reinforcement Learning (RL) is a machine learning technique that seeks a policy to maximize the expected cumulative reward over a given time horizon. RL methods have proven effective in sequential decision-making problems under uncertainty (SIVAMAYIL et al., 2023). In particular, they have achieved remarkable success in playing games (SILVER et al., 2017) and tuning large language models (KAUFMANN et al., 2024). Unlike traditional optimization approaches, RL does not rely on assumptions of convexity or stagewise independence, and the learned policies can adopt arbitrary functional forms. However, this flexibility comes at the cost of high computational demands, particularly when handling complex, high-dimensional constraints. To the best of the authors' knowledge, no existing work has applied RL to the hydrothermal dispatch problem while considering alternating current (AC) network constraints and an unknown data-generating process for the inflow.

In this work, we aim to demonstrate that combining optimization and RL can provide a more flexible alternative approach for the hydrothermal dispatch problem by accounting for nonlinear AC network constraints and incorporating a neural network-based inflow model within cost-to-go function approximations. At each stage, the model solves a deterministic optimization problem that minimizes the system's immediate operating cost while tracking target reservoir volumes obtained from a set of neural networks. The optimization problem ensures that all system constraints are satisfied, while the neural networks approximate both the cost-to-go function and the corresponding optimal policy. These networks are trained using RL, with the reward defined in terms of the system's operating cost. Unlike decision rule-based approaches, this method decomposes the problem by solving each stage independently as a smaller, tractable optimization problem.

To evaluate the proposed approach, we used SDDP as a benchmark and evaluated both models on different experimental setups. The first case study was designed to isolate the effects of the inflow model on policy quality. A SARIMA model was trained on historical inflow data from the Bolivian system, and used to generate an artificial history as well as the out-of-sample scenarios. The RL model was then trained

directly on the artificial history, simulating a model trained on historical data and applied to the real world. Since the SARIMA model does not guarantee the time independence required by SDDP, a Periodic Auto-Regressive (PAR(p)) model (MACEIRA; PENNA; DAMÁZIO, 2006) was fitted to the artificial historical data and used to generate inflow scenarios. These PAR(p) scenarios can be represented in a way that satisfies the stagewise independence assumption. Both policies were then evaluated using the out-of-sample data. Another case study aims to isolate the effects of network simplifications on policy performance. The RL model was trained using the full AC network representation, while SDDP employed the convex Direct Current (DC) approximation (MOLZAHN; HISKENS et al., 2019). The resulting policies were evaluated using an Alternating Current Optimal Power Flow (AC-OPF) model.

The main contributions of this work are as follows: (1) We propose a novel approach that combines RL and optimization to solve multistage stochastic optimization problems, such as the hydrothermal dispatch problem, without relying on time independence or convexity assumptions. (2) The single-stage deterministic target-tracking optimization problem enforces the application's hard constraints, while neural networks trained via RL capture the long-term temporal dependencies present in the cost-to-go function. (3) The computational experiments performed evaluate how different inflow models, number of in-sample scenarios and network representations impact the policy quality.

2
The Hydrothermal Dispatch Problem

The hydrothermal dispatch problem aims to determine which plants are dispatched at any given moment in order to meet demand while minimizing the long-term operating costs of the system. Considering the large size of the reservoirs, the impact of a decision made today can sometimes only be seen years into the future. Thus, in order to find adequate solutions to the problem, large horizons must be considered.

Another aspect of the problem is the significant uncertainty regarding inflow, demand, and renewable generation. In order to properly incorporate the uncertainty into the solution, multiple scenarios must be considered, further increasing the problem's complexity. In this work, renewable generation is not explicitly represented. Since it is a non-dispatchable generation, it is represented as a decrease in demand. Going forward, the term demand refers to the net demand after subtracting the renewable generation.

The long horizon and large number of scenarios make solving the entire problem at once an untractable proposition. A common approach used in this work is to break the problem into stages and solve it one at a time. To account for future stages, a cost-to-go term is introduced in the objective function, representing the relation between future costs and the current level of the reservoirs. Learning and representing this relation is the goal of both SDDP and the proposed RL algorithm.

The single-stage problem with the cost-to-go term is described below.

$$C(v_h^t, \omega^t) = \min_{g,\delta,f,u,z,v^{t+1}} \quad \sum_{j\in J} c_j g_j + \sum_{b\in B} c_b \delta_b + \mathbb{E}_{\omega^{t+1}}[C(v_h^{t+1}, \omega^{t+1})] \tag{2-1}$$

s.t.

$$\sum_{j\in J(b)} g_j + \sum_{h\in H(b)} \rho_h(u_h) + \sum_{l\in L^I(b)} f_l - \sum_{l\in L^O(b)} f_l + \delta_b = d_b, \quad \forall b \in B \tag{2-2}$$

$$v_h^{t+1} = v_h^t - u_h - z_h + a_h + \sum_{m\in M(h)} (u_m + z_m), \quad \forall h \in H \tag{2-3}$$

$$0 \le v_h \le \overline{V}_h, \quad 0 \le u_h \le \overline{U}_h, \quad 0 \le z_h, \quad \forall h \in H \tag{2-4}$$

$$0 \le g_j \le \overline{G}_j, \quad \forall j \in J \tag{2-5}$$

$$-\overline{F}_l \leq f_l \leq \overline{F}_l, \quad \forall l \in L \tag{2-6}$$

In the model, H, J, B, and L represent the sets of hydro plants, thermal plants, buses, and transmission lines, respectively. $M(h)$ is the set of upstream plants for hydro plant h, while the sets $J(b)$ and $H(b)$ represent the subsets of plants connected to bus b. $L^I(b)$ and $L^O(b)$ are the sets of transmission lines entering and exiting bus b, respectively.

The parameters c_j and c_b are the thermal plants' operating costs and the buses' deficit cost, respectively. ρ_h is the production factor of hydro plant h, d_b is the demand at bus b, v_h^t is the initial volume of reservoir h, and a_h is the inflow at hydro plant h.

The decision variables of the problem are g_j, u_h, z_h, f_l, δ_b, and v_h^{t+1}, which represent the generation of thermal plants, turbine discharge, and spillage of hydro plants, transmission line flows, bus deficits, and the final reservoir volumes, respectively. The time index is assumed to be t when omitted.

The problem includes load balance constraints per bus in Equation (2-2) and water balance constraints in Equation (2-3). The objective function (2-1) comprises the operating costs of thermal plants, the deficit costs at the buses, and the expected future cost given the final volume. Equations (2-4), (2-5) and (2-6) represent the bounds for the decision variables.

2.1 The SDDP Algorithm and the PAR(p) Model

The SDDP algorithm works by replacing the expected value in the objective function (2-1) with a variable η and adding the following constraints:

$$\eta \geq \psi_k + \boldsymbol{\chi}_k^\top \boldsymbol{v}^{t+1}, \quad \forall k \in K \tag{2-7}$$

where $\boldsymbol{\chi}$ and $\boldsymbol{v}$ are both vectors of size $|H|$, and ψ and $\boldsymbol{\chi}$ values are obtained at each iteration k of the algorithm.

One approach to model time dependent uncertainty in SDDP is using a Markov Chain with a different cost-to-go function for each possible Markov state at each stage (GJELSVIK; BELSNES; HAUGSTAD, 1999), which greatly increases problem complexity. The approach used in this work, which is more common to represent inflows, requires that the time dependent uncertainty affects only the right hand side of constraints and can be represented as an auto-regressive stochastic process with stagewise independent noise. The PAR(p) model, shown in Equation (2-8) is used.

$$a_h^t = \sum_{i \in I} \phi_h^i a_h^{t-i} + \epsilon_h^t, \quad \forall h \in H \tag{2-8}$$

The $|I|$ past values of inflow a^{t-i} are stored as additional state variables, and the only uncertainty is in the stagewise independent noise term, ϵ. It is relevant to mention that this representation increases the complexity of the problem, as the number of dimensions of the state-space is increased, and the size of the vector $\boldsymbol{\chi}$ in Equation (2-7) grows with it.

2.2 The Network Models

The hydrothermal dispatch problem, as presented, uses only Equations (2-2) and (2-6) to represent the electrical network. Equations (2-9) to (2-16) show the more realistic, non-convex AC model:

$$E_r^A = 0, \quad \forall r \in B^R \tag{2-9}$$

$$\sum_{k \in J(i)} g_k + \sum_{k \in H(i)} \rho_k(u_k) + \delta_i - d_i - \sum_{k \in S(i)} Y_k^A (E_i^M)^2 = \sum_{(i,j) \in L(i)} \Pi_{ij}^P + \sum_{(i,j) \in L^R(i)} \Pi_{ij}^P, \quad \forall i \in B \tag{2-10}$$

$$\sum_{k \in S(i)} Y_k^R (E_i^M)^2 + \sum_{(i,j) \in L(i)} \Pi_{ij}^Q + \sum_{(i,j) \in L^R(i)} \Pi_{ij}^Q = 0, \quad \forall i \in B \tag{2-11}$$

$$\Pi_{ij}^P = Y_{ij}^G (E_i^M)^2 - Y_{ij}^G \left(E_i^M E_j^M \cos(E_i^A - E_j^A) \right) - Y_{ij}^B \left(E_i^M E_j^M \sin(E_i^A - E_j^A) \right), \quad \forall (i,j) \in L \cup L^R \tag{2-12}$$

$$\Pi_{ij}^Q = -(Y_{ij}^B + Y_{ij}^C)(E_i^M)^2 + Y_{ij}^B \left(E_i^M E_j^M \cos(E_i^A - E_j^A) \right) - Y_{ij}^G \left(E_i^M E_j^M \sin(E_i^A - E_j^A) \right), \quad \forall (i,j) \in L \cup L^R \tag{2-13}$$

$$(\Pi_{ij}^P)^2 + (\Pi_{ij}^Q)^2 \leq (\overline{\Pi}_{ij})^2, \quad \forall (i,j) \in L \cup L^R \tag{2-14}$$

$$\underline{\theta}_{ij} \leq E_i^A - E_j^A \leq \overline{\theta}_{i,j}, \quad \forall (i,j) \in L \tag{2-15}$$

$$\underline{E}_i \leq E_i^M \leq \overline{E}_i, \quad \forall i \in B \tag{2-16}$$

In this model, B^R, L^R, and $S(i)$ represent the sets of reference buses, the set of branches in reverse orientation, and the shunts of bus i,

respectively. Y_{ij}^G, Y_{ij}^B and Y_{ij}^C represent the branch conductance, susceptance and pi-section parameter, respectively. Y_k^A and Y_k^R represent the shunt admittance parameters. The variables Π_{ij}^P and Π_{ij}^Q are the branch active and reactive flows, and E_i^M and E_i^A represent the bus voltage magnitude and angle, respectively.

Equation (2-9) shows the reference bus voltage angle constraint. Equations (2-10) and (2-11) show the AC bus active and reactive load balances, replacing Equation (2-2) in the dispatch problem. Equations (2-12) and (2-13) represent Ohm's law for the power flow. Equations (2-14), (2-15), and (2-16) show the branch apparent power bounds, the branch voltage angle difference bounds, and the bus voltage magnitude bounds, respectively. These bounds replace Equation (2-6) in the original problem.

The linearized DC approximation is obtained by keeping both constraints (2-2) and (2-6) from the original model, and adding constraints (2-9) and (2-17):

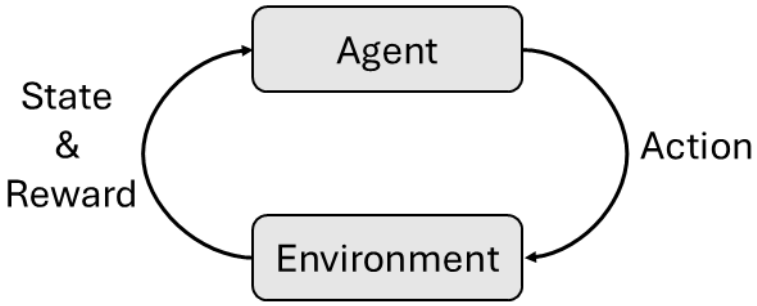
$$f_l = \frac{E_{b^O(l)}^A - E_{b^I(l)}^A}{x_l} \quad \forall l \in L \tag{2-17}$$

Where $b^I(l)$ and $b^O(l)$ represent the input and output buses of a given line, and x_l is the line's reactance.

3
The Reinforcement Learning Framework

The RL framework has two main components: the environment and the agent (WATKINS; DAYAN, 1992). At each time step, the agent observes the current state of the environment and selects an action according to its policy. The environment's state evolves according to some transition probability to the next state, and the agent receives a reward signal for the current time step, as shown in Figure 3.1. This process is repeated over the entire problem horizon, which is referred to as one episode.

Figure 3.1: The RL Framework



The goal of an RL algorithm is to find the optimal policy for a given problem, defined as the policy that maximizes the cumulative reward over an entire episode. The cumulative reward, also referred to as return, is defined recursively by Equation (3-1):

$$G_t = r_t + \gamma \cdot G_{t+1} \tag{3-1}$$

Where γ is a discount factor applied to future rewards, particularly important in infinite horizon problems to avoid the return function tending to infinity. However, discounting future rewards can also be used in finite horizon problems—for example, to represent interest rates when dealing with financial rewards.

The problem of multistage stochastic hydrothermal dispatch discussed in this work is a finite horizon problem. A finite horizon means an optimal policy must know the current stage of the problem, as the best action at the first time step can differ greatly from the best action at the end of an episode. At the start of the dispatch problem, for example, it might be optimal to save water for the future, while in the last

stages, the optimal decision would always be to use all the available water, as there is no future to be considered. This is not the case for many problems in which the return function is constant in time, such as balancing an inverted pendulum for as long as possible (ISRAILOV et al., 2023), an infinite horizon problem.

Two main ideas are considered to achieve this. The first, which we call the *single-net* approach, consists of encoding the current stage t as part of the state vector, as done in (GRONDMAN et al., 2013). The second, *multi-net*, trains a different policy for each stage, as done in (LEI et al., 2021). While the *single-net* agent learns a more complex policy with an extended state-space, the *multi-net* model requires more training data, as each of its neural networks trains on data from a single time step. Both methods are discussed further in Section 4.1.

This framework is well suited (GRONDMAN et al., 2013) to solve problems modeled as Markov Decision Processes (MDP) (BELLMAN, 1958). The problem of hydrothermal dispatch can be modeled as a fully observable MDP, in which the agent's observation contains full information about the environment's state. For the remainder of this work, observation and state will be used interchangeably from the agent's perspective.

RL algorithms can be divided into two groups: model-based and model-free. The key difference is whether the agent has access to (or learns) a model of the environment (ACHIAM, 2018). Model-free algorithms, such as the one used in this work, can be divided into value-based and policy-based algorithms.

A value-based method learns an approximation of the value function, representing the expected return for a given state-action pair. This results in implicitly learning a policy by choosing the actions that maximize the learned value function. Policy-based methods, on the other hand, explicitly learn the policy. Value-based methods are usually more sample efficient but also less stable (TSITSIKLIS; ROY, 1997). There are also methods that combine both approaches, leveraging the strengths of each one, as they are not incompatible. As shown in (SCHULMAN; CHEN; ABBEEL, 2018), they can even be equivalent in some settings.

A topic not often discussed is the similarity between value-based RL and SDDP. Both methods use a forward simulation step and a backpropagation step to iteratively estimate a value (or cost) function in an uncertain, long-term problem. SDDP uses piecewise linear convex functions, while RL has used many different approximations (SUT-

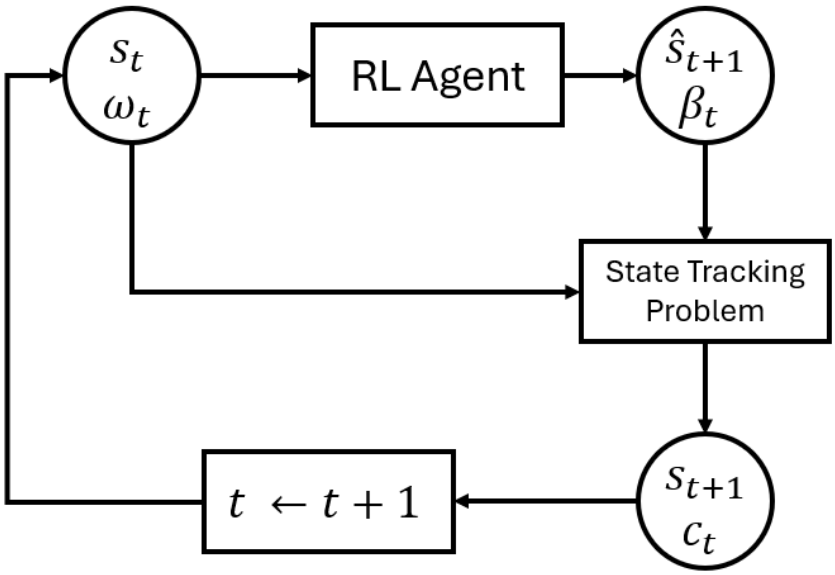
TON; BARTO, 2018). Deep RL, which is the type present in this work, uses neural networks as universal function approximators (HORNIK; STINCHCOMBE; WHITE, 1989). Interestingly, if the neural networks only contain ReLU activation functions (SUTTON; BARTO, 2018) they are piecewise linear, but nonconvex, functions.

4
The Proposed Algorithm

The method proposed in this work trains an RL agent to output target volume values for the reservoirs at each stage. These target values are then used in a modified version of the dispatch problem, the single-stage state tracking optimization problem. This problem seeks to minimize both the immediate operating costs as well as the distance between the final volumes and the target values provided by the RL agent. The weight of the tracking penalty in the problem is also given by the RL agent. The resulting operating costs are used by the RL algorithm, training the agent to minimize the total operating cost over the entire horizon.

This method leverages the RL framework to solve the multistage stochastic problem, while the state tracking problem ensures that all constraints are met and guarantees optimal short-term operation. Both the RL agent and the state tracking problem are further described in the following sections. Figure 4.1 shows a diagram of the method.

Figure 4.1: Information Flow in the Model



Where s_t denotes the system's state (the reservoir volumes and past inflows), and ω_t represents the sampled uncertainties (the inflows for all hydro plants, and the energy demand at each bus). Given these inputs, the RL agent selects the set of target reservoir volumes for the next stage $\hat{s}_{t+1}$, as well as the weight of the state tracking penalty for each reservoir β_t. All of these values are then used in the state tracking

problem to obtain the immediate operating costs c_t and the final reservoir volumes s_{t+1}. The loop continues for the entire problem horizon, with the final volume of stage t being used as the initial volume of stage $t+1$. All the values shown in the image are stored to be used in the RL agent's training. The RL agent and state tracking problem are described in more detail in sections 4.1 and 4.2, respectively.

While the SDDP approach used as a benchmark in this work requires the inclusion of the inflow lags as state variables to guarantee convergence, the same is not true for the RL model. The decision to include the inflow lags as state variables is due to computational comparisons, that show the inclusion of these additional state variables improved the results, reducing the final costs up to 5%, while the increased complexity resulted in training times up to 12% higher.

4.1 Deep Deterministic Policy Gradient

To train the RL agent, we use an algorithm called Deep Deterministic Policy Gradient (DDPG) (LILLICRAP et al., 2015). The algorithm uses different neural networks to represent the policy and the value function of the problem, making it both a value-based and policy-based algorithm. The policy is represented by a neural network called the actor, which, given the system's state, outputs an action (SILVER et al., 2014). The value function is represented by another neural network, called the critic, which maps the state of the system and a given action to an estimate of the total cost until the end of the horizon for that state-action pair. The actor is trained to minimize the total cost, while the critic is trained to estimate the total cost from a state and action. In the proposed model, the RL state consists of $\{s_t, \omega_t\}$, the action is composed by $\{\hat{s}_{t+1}, \beta_t\}$ and the reward is given by $\{-c_t\}$.

Let Q_t be the critic, μ_t the actor, and θ_{Q_t} and θ_{μ_t} the parameters of the respective neural networks for each stage. The RL action is given by Equation 4-1.

$$\{\hat{s}_{t+1}, \beta_t\} = \mu_t(s_t, \omega_t; \theta_{\mu_t}) \quad \forall t \tag{4-1}$$

To train the networks, a sample of size N is drawn from the stored data to update the networks. The gradient-based parameter update

Equations 4-2 and 4-3 are shown below.

$$\theta_{Q_t} \leftarrow \theta_{Q_t} - \alpha_Q \nabla_{\theta_{Q_t}} \left(\frac{1}{N} \sum_{k=1}^{N} \left(Q_t(s_t^k, \omega_t^k, \hat{s}_{t+1}^k, \beta_t^k; \theta_{Q_t}) - c_t^k - \gamma Q_{t+1}^T(s_{t+1}^k, \omega_{t+1}^k, \hat{s}_{t+2}^k, \beta_{t+1}^k; \theta_{Q_{t+1}^T}) \right)^2 \right) \quad (4\text{-}2)$$

$$\theta_{\mu_t} \leftarrow \theta_{\mu_t} - \alpha_\mu \nabla_{\theta_{\mu_t}} \left(\frac{1}{N} \sum_{k=1}^{N} Q_t(s_t^k, \omega_t^k, \mu_t(s_t^k, \omega_t^k; \theta_{\mu_t}); \theta_{Q_t}) \right) \quad (4\text{-}3)$$

Where α_Q and α_μ are the learning rates of the critic and actor, γ is the discount factor, and the stage t is given for each sample k. It is important to note that $\hat{s}_{t+2}^k$ and β_{t+1}^k are not obtained from the stored data, but instead sampled during the update phase, as shown in Equation (4-4).

$$\{\hat{s}_{t+2}^k, \beta_{t+1}^k\} = \mu_t^T(s_{t+1}^k, \omega_{t+1}^k; \theta_{\mu_{t+1}^T}) \quad (4\text{-}4)$$

The critic's weights are updated in the direction that minimizes the squared error between two cost estimates. One estimate is the network's prediction based on the current state, and the other is the immediate observed cost plus the network's prediction from the next state onward. Intuitively, the second value can be thought of as estimating the same quantity as the first, but with one additional data point, the immediate cost. The actor, in turn, is updated in a direction that minimizes the expected cost estimated by the critic.

Q_t^T and μ_t^T, present in Equations (4-2) and (4-4), are called the target critic and target actor networks, respectively. Their purpose is to avoid self-dependency, where the network weight updates sample the networks themselves. DDPG employs this approach to reduce training instability, a characteristic issue with value-based methods. The target networks are delayed copies of the original networks, and their weights are updated according to Equations (4-5) and (4-6).

$$\theta_{Q_t^T} \leftarrow \lambda \theta_{Q_t^T} + (1 - \lambda)\theta_{Q_t} \quad (4\text{-}5)$$

$$\theta_{\mu_t^T} \leftarrow \lambda \theta_{\mu_t^T} + (1 - \lambda)\theta_{\mu_t} \quad (4\text{-}6)$$

Where λ is a hyperparameter with a value typically close to 1.

Another important aspect of the algorithm is whether the single-net or multi-net approaches are used. In the single-net approach, the current stage is represented via one-hot encoding, by adding a vector to the RL state. The vector has one entry for each stage in the problem, and

all its values are zero except for the one corresponding to the current stage, which equals one. Furthermore, in the single-net approach, we can consider $\mu_t = \mu$ and $Q_t = Q$ for all values of t in the problem horizon.

By using a different neural network for each stage, the multi-net approach can learn a more specialized policy for a given problem, resulting in costs that were, on average, 7% smaller in the computational comparisons performed. However, this method adds another dimension in which the number of stages impacts problem complexity, by increasing the number of neural networks that need to be trained. For problems with larger horizons, such as the 60-stage problems used in this work's case studies, training times increased by up to a factor of three. For this reason, all cases presented in this work use the single-net approach.

4.2 The Single Stage Optimization Problem

Given the target reservoir volumes and tracking penalty weights determined by the actor, a deterministic optimization problem is solved at each stage to minimize the system's operating cost and the difference between the final reservoir volumes and the target volumes, as outlined below.

$$min \quad \sum_{j \in J} c_j g_j + \sum_{b \in B} c_b \delta_b + \sum_{h \in R} \beta_h^H \Delta_h^H + \sum_{h \in R} \beta_h^S \Delta_h^S \tag{4-7}$$

s.t.

$$\Delta_h^H \geq v_h^{t+1} - \hat{v}_h^{t+1} \qquad \forall h \in R \tag{4-8}$$

$$\Delta_h^H \geq \hat{v}_h^{t+1} - v_h^{t+1} \qquad \forall h \in R \tag{4-9}$$

$$\Delta_h^S \geq v_h^{t+1} - \hat{v}_h^{t+1} - \frac{\overline{V}_h}{\zeta^S} \qquad \forall h \in R \tag{4-10}$$

$$\Delta_h^S \geq \hat{v}_h^{t+1} - v_h^{t+1} - \frac{\overline{V}_h}{\zeta^S} \qquad \forall h \in R \tag{4-11}$$

(2-2) - (2-6)

At the start of the training, all neural networks are initialized with random values; thus, the target volumes $\hat{v}^{t+1}$ and hard tracking penalty weight β^H are unreliable. In the case of the tracking penalty specifically, having values close to zero means the optimal strategy is always to use all the available water, and the target volumes would not affect the problem's solution. This effect significantly slows training.

A soft tracking penalty is added to avoid this problem, with a fixed weight β^S, not dependent on the RL agent. This soft penalty is controlled by parameter ζ^S, which was set to 10 in all case studies performed. Thus, the penalty is only active when the final volume differs from the target by more than 10% of the reservoir's maximum volume. This greatly speeds up the initial training without impacting the problem's solution once the β^H values are trained.

In the model, $R \subset H$ represents the set of hydro plants that contain reservoirs. The added decision variables Δ_h^H and Δ_h^S represent the absolute tracking errors for the hard and soft penalties, respectively. Equations (4-8) and (4-9) represent the absolute error for the hard penalty, while Equations (4-10) and (4-11) represent the same for the soft tracking penalty.

5
Computational Experiments

Two different experiments were performed, comparing four different instances of the proposed algorithm in total, and an open-source implementation of the SDDP algorithm (DOWSON; KAPELEVICH, 2021) was used as a benchmark. The system used in the experiments was the 28-bus Bolivian system, whose dimensions are shown in Table 5.1. All experiments considered a 5-year horizon divided in 60 monthly stages, and all the results presented are from out-of-sample simulations with 10,000 scenarios. For the RL method, the experiments were repeated five times using different random seeds when initializing the neural network weights.

Table 5.1: System Dimensions and Study Parameters

Number of hydro plants	11
Number of thermal plants	23
Number of buses	28
Number of branches	31
Number of loads	26
Number of stages	60
Number of out-of-sample-scenarios	10000

The first case aims to study the effect of the inflow model on the policy quality. Since SDDP requires stagewise independent uncertainty, it was trained with data from a PAR(p) model, represented in the optimization problem by Equation (2-8). The out-of-sample scenarios were generated with a SARIMA model, and the RL instances used in this study were trained on scenarios generated by either the PAR(p) or the SARIMA model. All policies were compared on the same out-of-sample scenarios.

The second case aims to study the effect of the network representation on policy quality. The SDDP algorithm was trained with the linearized DC model, shown in Equations (2-2), (2-6), (2-9) and (2-17), while the proposed method was trained with the non-convex AC model, shown in Equations (2-9) - (2-16). Both policies were trained with scenarios from the same PAR(p) model and tested using the out-of-sample SARIMA scenarios. In the out-of-sample simulation, AC network constraints were used for all models.

5.1 Parameter tuning

Before running the final experiments, it was necessary to determine the optimal neural network architecture, DDPG hyperparameters and algorithm choices such as the tradeoff between single-net and multi-net. For this purpose, a grid search was performed using a smaller system than the 28-bus system presented. Due to the large number of parameters to optimize, the search was divided in two blocks to circumvent the curse of dimensionality. The first block considered the different modeling choices and neural network architectures, shown in Table 5.2 and Table 5.3, while the second block considered the DDPG hyperparameters shown in Table 5.4.

Table 5.2: Modeling choices

Parameter	Values considered	Value selected
Time-step representation	single-net, multi-net	single-net
Inflow lags in the state	included, not included	included

Table 5.3: Neural network architecture

Parameter	Values considered	Value selected
Number of hidden layers	1, 2, 3, 4	2
Neurons per layer	32, 64, 128, 256	64

Table 5.4: DDPG Hyperparameters

Parameter	Values considered	Value selected
Batch size	32, 64, 128, 256, 512	64
Update frequency	1, 2, 4, 8, 16, 32, 64	1
Target network update weight	0.95, 0.99, 0.995, 0.999	0.995

In Table 5.4, the batch size corresponds to the value N shown in equations (4-2) and (4-3), while the episodes per udpate represent how many episodes are solved between each neural network update. The target network update weight is the parameter λ shown in equations (4-5) and (4-6).

5.2 Case Study 1 - The Inflow Model

The main goal of this case study is to evaluate whether the ability to train the model on uncertainty scenarios originated from any distribution provides an advantage to the RL method. In a real world setting, this could be exploited by training the model directly on the inflow history, for instance. However, given the relatively small amount of historical data available, this setup would not provide a satisfactory number of out-of-sample scenarios for comparing the models. For this reason, in this work, a SARIMA model fitted on the inflow history was used to generate both the out-of-sample scenarios, as well as an artifical inflow history comprised of 100 years.

The SDDP benchmark, as previously mentioned, requires stagewise independent uncertainty. To satisfy this requirement, a PAR(p) model was trained on the artificial history, and used to generate the in-sample scenarios for the SDDP training. The first RL instance evaluated in this experiment used the artificial history as its in-sample data, organizing the 100 years into 95 sequential 5-year scenarios. In the setup described, this model was trained with scenarios originating from the same random process as the out-of-sample scenarios, simulating a model trained on actual historical data and then evaluated in the real world. In the tables and plots shown, it is labeled the "RL (artificial history)" model.

This first RL instance, while having the advantage of using scenarios from the true data-generating process, has a key weakness: it doesn't use an inflow model directly during training, and thus has a very limited number of scenarios. To evaluate the relevance of this weakness, a second RL instance was trained using an unlimited amount of scenarios from the same SARIMA model. It is essential to note that this instance has no real-world counterpart: it is equivalent to having an inflow model that perfectly represents the true data-generating process found in nature, and thus it has an unfair advantage. Its results are presented only to evaluate the advantages found in using a model that can generate an arbitrary number of scenarios, possibly leading to a more robust policy. In the results, this instance is labeled "RL (perfect model)".

The third RL instance evaluated was trained using scenarios from the same PAR(p) scenario as SDDP, and is labeled "RL (PAR(p))". Table 5.5 summarizes the experimental setup.

Table 5.5: Case 1 - Experimental Setup

Label	Algorithm	Inflow model	Scenarios
SDDP	SDDP	PAR(p)	unlimited
RL (artificial history)	RL	SARIMA	95
RL (perfect model)	RL	SARIMA	unlimited
RL (PAR(p))	RL	PAR(p)	unlimited

The main metric to compare the different policies is the total operating cost over the entire horizon. Table 5.6 shows the normalized operating cost of each policy in the out-of-sample simulation.

Table 5.6: Case 1 - Normalized Total Operating Costs

Model	Avg.	Scenario std. dev.	Seed std. dev.
SDDP	1.000	$\pm$ 0.058	-
RL (artificial history)	1.085	$\pm$ 0.131	$\pm$ 0.017
RL (perfect model)	1.038	$\pm$ 0.085	$\pm$ 0.017
RL (PAR(p))	1.145	$\pm$ 0.072	$\pm$ 0.020

Figure 5.1 shows the average total operating cost for each of the five random seeds used in the RL models. Figure 5.2 zooms in on the plot to show the 99% confidence interval around the average.

The comparisons between the RL models show the expected results: training the model with scenarios from the true data-generating process yields better results than using a PAR(p) model. Notably, this is true even when using only the limited number of scenarios obtained from the artificial history. Looking at the average costs for both models trained with SARIMA scenarios, we can also see that having a model to generate an unlimited number of scenarios also results in lower costs.

When comparing to SDDP, however, all RL models obtained higher costs. Although the models trained with the SARIMA scenarios are within 10% and 5% of the benchmark costs, SDDP was able to surpass even the "RL (perfect model)", which has an unfair advantage.

The shaded area in some plots represents the interval between the 10th and 90th percentiles of the data, while the faded lines represent the average for a given random seed in the RL model. The bold lines represent the average across all scenarios and seeds. Figure 5.3 and Figure 5.4 show the system's operating cost across the stages. All models show a peak in costs at some point during the year, corresponding to a lower hydro generation, as shown in Figure 5.5.

Figure 5.6 and Figure 5.7 show the evolution of the aggregated reservoir volumes over time. The models trained with the PAR(p) sce-

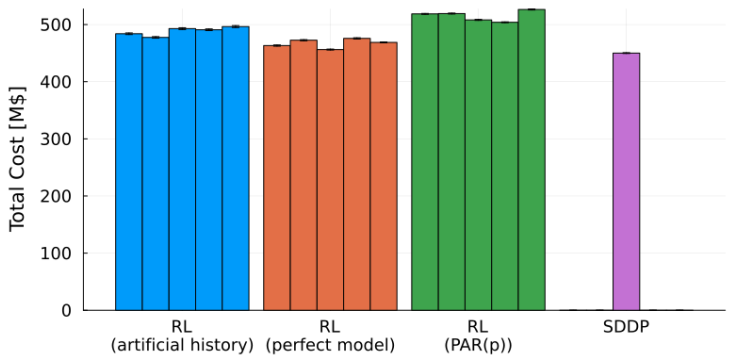


Figure 5.1: Case 1: Total Operating Costs

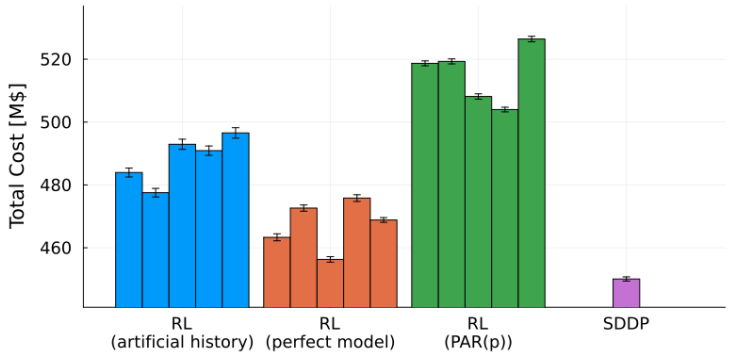


Figure 5.2: Case 1: Total Operating Costs with Confidence Interval

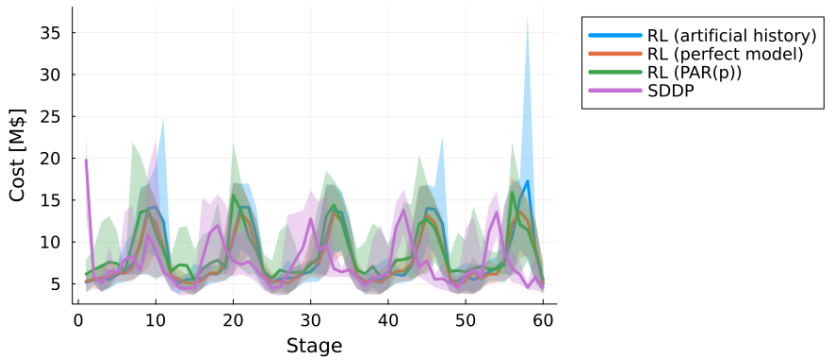


Figure 5.3: Case 1: Operating Costs with Quantiles

narios show a higher volume on average across the entire horizon, when compared to the models trained with the SARIMA scenarios. Looking specifically at the final volume, it is worth noting that for some seeds, the RL models reached the end of the horizon with empty reservoirs, while for others they did not. This is one example of the importance of evaluating multiple seeds, as the neural network initialization can have a considerable impact on the final operation.

The training loss at each iteration is shown in log scale in Figure 5.8 for both RL models and each random seed, for an eight-hour training

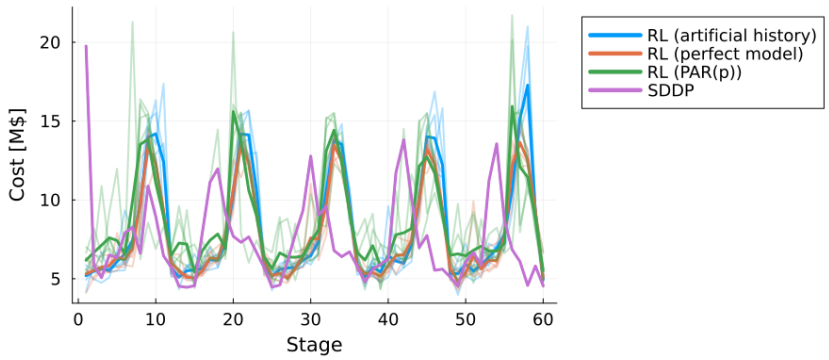


Figure 5.4: Case 1: Operating Costs per Seed

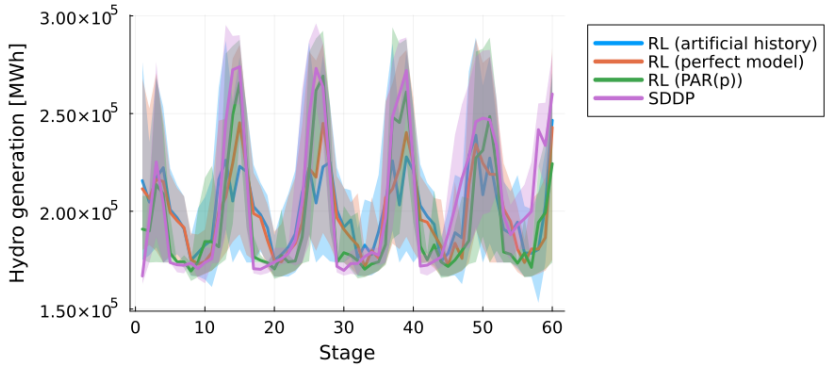


Figure 5.5: Case 1: Hydro Generation with Quantiles

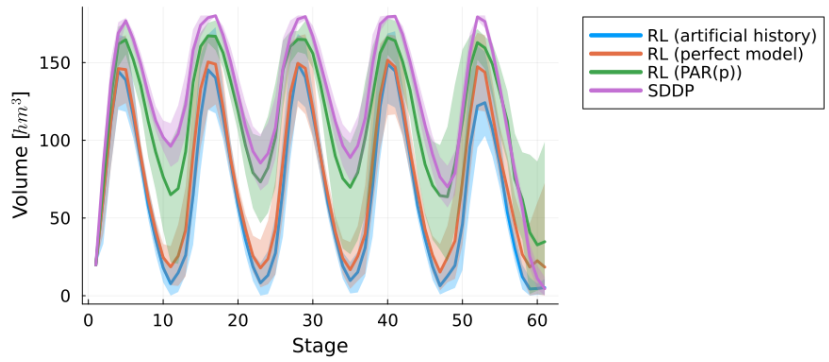


Figure 5.6: Case 1: Reservoir Volumes with Quantiles

session. It is important to note that the models were trained with different in-sample data sets, so the training losses are not directly comparable, as each model was evaluated with its own data during training. The same, of course, is not true for the out-of-sample cost comparisons.

5.3 Case Study 2 - The Network Model

To evaluate the impact of the network model on policy quality, the SDDP algorithm was trained with the linearized DC model described in

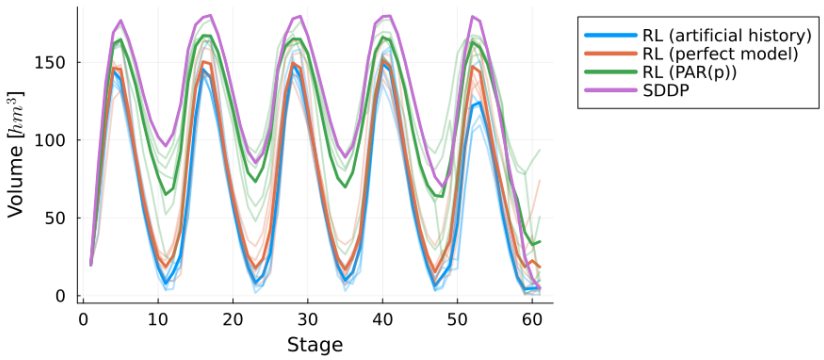


Figure 5.7: Case 1: Reservoir Volumes per Seed

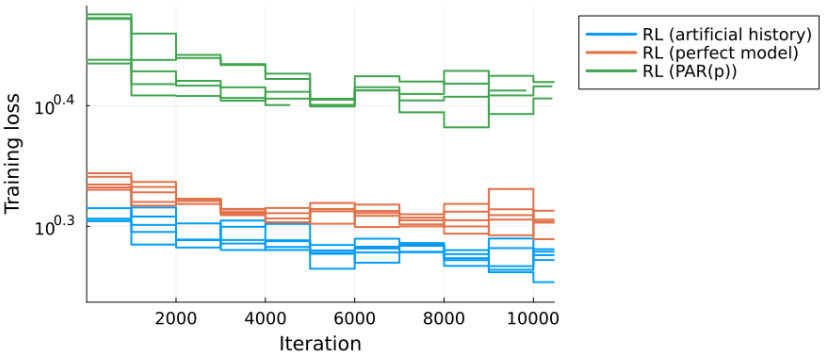


Figure 5.8: Case 1: Training Loss

Equations (2-2), (2-6), (2-9) and (2-17), while the RL model was trained with the AC model, shown in Equations (2-9) to (2-16). In the out-of-sample comparison, both policies were applied to a problem containing all the AC network constraints. Although the AC model can represent reactive loads, none were used in these studies, so the linearized DC model could represent the system as effectively as possible for a fair comparison.

In this case study, it is helpful to think of the monthly stages as representing typical operating points of the system, rather than aggregated energy values. In this setting, representing the AC constraints is not only valid but also helps to align the planning problem with operational reality.

The normalized operating costs are shown in Table 5.7.

Table 5.7: Case 2 - Normalized Total Operating Costs

Model	Avg.	Scenario std. dev.	Seed std. dev.
SDDP (AC)	1.000	$\pm$ 0.062	-
RL (AC)	1.251	$\pm$ 0.159	$\pm$ 0.012

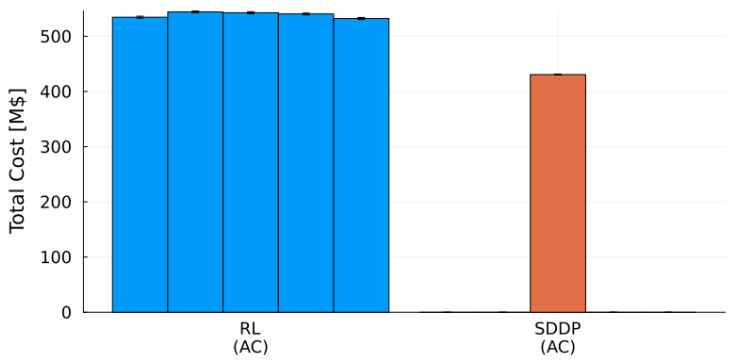


Figure 5.9: Case 2: Total Operating Costs

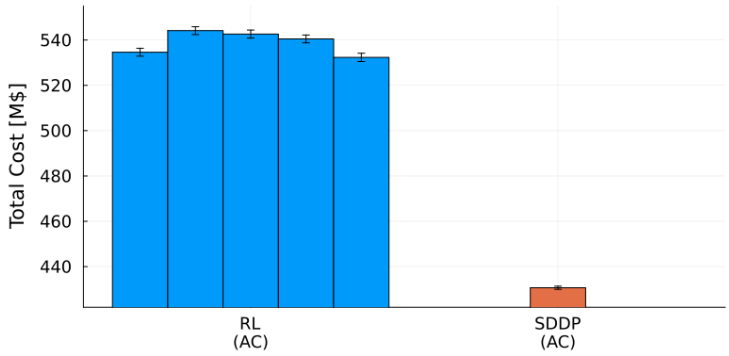


Figure 5.10: Case 2: Total Operating Costs with Confidence Intervals

Figure 5.9 shows the average total operating cost for each of the five random seeds used in the RL models. Figure 5.10 zooms in on the plot to show the 99% confidence interval around the average. In this case, the RL model obtained considerably more expensive operation costs, with lower variability between random seeds.

Figure 5.11 and Figure 5.12 show the system's operating cost across the stages.

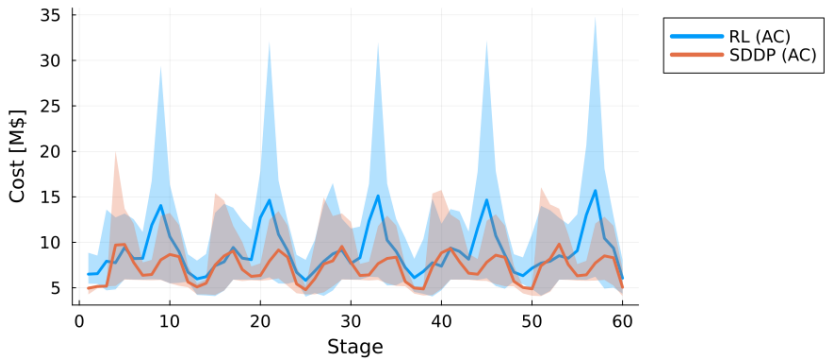


Figure 5.11: Case 2: Operating Costs with Quantiles

Figure 5.13 shows the hydro generation for the models. In the peri-

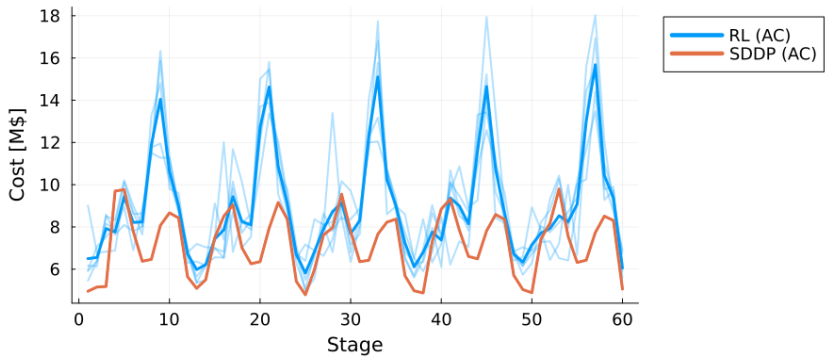


Figure 5.12: Case 2: Operating Costs per Seed

ods of high hydro generation, it is notable that even the 10th percentile of SDDP generation sometimes surpasses the RL average. However, in the periods of low hydro generation, and thus higher costs, both the average generation and the lower percentile are very similar between the models, suggesting the higher RL costs might come from a small amount of very expensive scenarios.

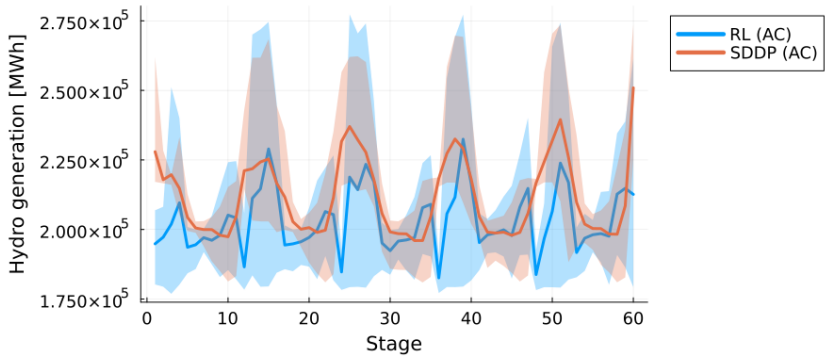


Figure 5.13: Case 2: Hydro Generation with Quantiles

Figure 5.14 and Figure 5.15 show the evolution of the aggregated reservoir volumes over time. SDDP shows a lower average volume across most of the horizon, indicating a more agressive use of the water.

Figure 5.16 shows the log training loss during training for each random seed. It is worth noting that, even though the model was trained three times as long as the RL models in the previous case, the total number of iterations is smaller. This is due to the increased complexity of the AC network model, which results in most of the computational power being used to solve the state tracking problem and not updating the network weights.

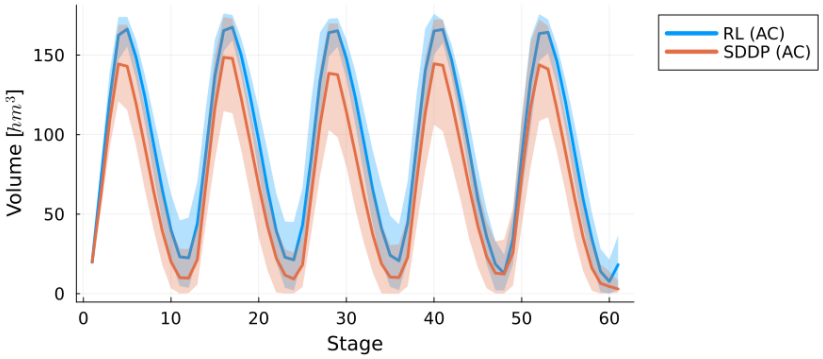


Figure 5.14: Case 2: Hydro Volume with Quantiles

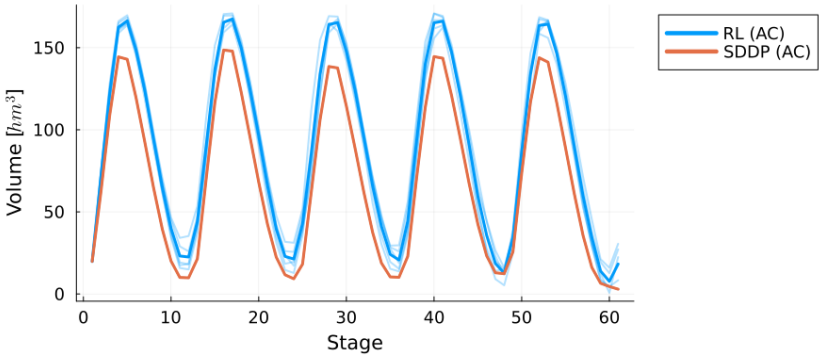


Figure 5.15: Case 2: Hydro Generation per Seed

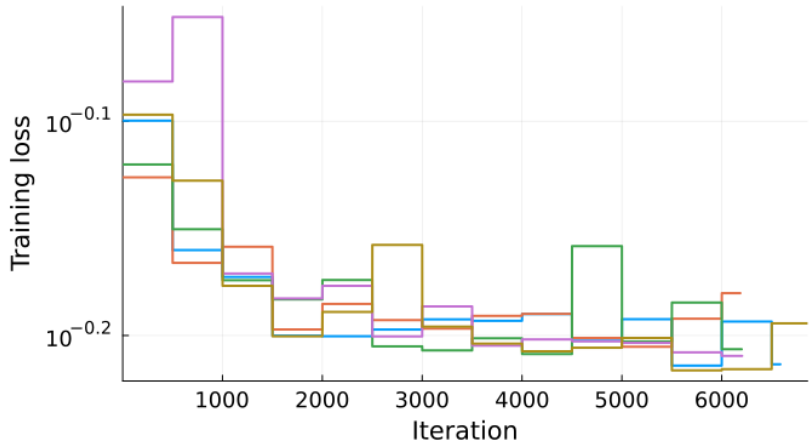


Figure 5.16: Case 2: Training Loss

6
Conclusion

This work demonstrated that combining optimization and Reinforcement Learning (RL) provides a viable method to solve real-world hydrothermal dispatch problems with no assumptions on convexity or time independence. The proposed approach effectively leverages the hard constraint representation of the target-tracking optimization problem and the flexibility of the RL framework. The model does not rely on assumptions of time independence or convexity, providing an alternative method to align long-term planning with operational reality. However, the policies obtained still resulted in higher costs when compared to SDDP.

In case study 1, we showed that the proposed approach benefits from employing a more general inflow model rather than relying on the PAR(p) model required by SDDP. When comparing the RL (artificial history) and the RL (PAR(p)) models, a cost reduction of 5% was observed when training with the artificial history. In case study 2, SDDP performed considerably better than the RL model. The lower number of training iterations suggests that the RL model might benefit from more training time in the AC problem. However, considering the training already took considerably longer than SDDP or the RL models in case 1, work in increasing training efficiency would be important.

Within the limitations of this work, which include all assumptions made about the proposed model and the analyzed data, the experiments conducted allow us to draw the following conclusions: (1) The PAR(p) model, commonly used to represent inflows in long-term hydrothermal dispatch problems, is a significant approximation that can impact the resulting policy and increase system costs. (2) Leveraging an inflow model to generate an unlimited number of scenarios during training can lead to better policies. (3) The proposed approach obtained promising results, but still requires further work to improve on the policies obtained by SDDP.

We highlight some possible avenues for future work to improve model results and computational efficiency. Some parallelization could be explored to improve training time. Multiple scenarios of the state-tracking problem can be solved simultaneously and then sampled during the network update step. This would be particularly beneficial in

AC cases, where the optimization problem is the most time-consuming part of the solution. Another possibility for increasing sample efficiency—and thereby reducing training time—is to combine the single-net and multi-net approaches. By first training the single-net agent and then copying its weights into the multi-net agent for further training, it may be possible to leverage the sample efficiency of the first method to accelerate the training of the more accurate second method.

More modern RL algorithms, such as Twin Delayed DDPG (TD3) (FUJIMOTO; HOOF; MEGER, 2018) and Soft Actor-Critic (SAC) (HAARNOJA et al., 2018), were developed as successors to DDPG to address some of its limitations. Repeating the case studies with these algorithms in place of DDPG would be a relevant research direction. Another possibility on the RL algorithm side is to leverage the fact that a model is available in the optimization problem and use model-based RL algorithms. These algorithms—such as the one used in AlphaZero (SILVER et al., 2017)—can consider multiple actions by solving the optimization problem for each. Given the resulting next state and associated costs, the agent can use this information to plan the next action more effectively.

7
Bibliography

ACHIAM, J. **Spinning Up in Deep Reinforcement Learning**. 2018. Cited in page 20.

BELLMAN, R. Dynamic programming and stochastic control processes. **Information and Control**, v. 1, n. 3, p. 228–239, 1958. ISSN 0019-9958. Disponível em: https://www.sciencedirect.com/science/article/pii/S0019995858800030. Cited in page 20.

BODUR, M.; LUEDTKE, J. R. Two-stage linear decision rules for multi-stage stochastic programming. **Mathematical Programming**, v. 191, n. 1, p. 347–380, Jan 2022. ISSN 1436-4646. Disponível em: https://doi.org/10.1007/s10107-018-1339-4. Cited in page 12.

BRIGATTO, A. et al. Assessing the optimistic bias in the natural inflow forecasts: A call for model monitoring in brazil. **arXiv preprint arXiv:2410.13763**, 2024. Cited in page 12.

BRIGATTO, A.; STREET, A.; VALLADAO, D. M. Assessing the cost of time-inconsistent operation policies in hydrothermal power systems. **IEEE transactions on power systems**, IEEE, v. 32, n. 6, p. 4541–4550, 2017. Cited in page 12.

DOWSON, O.; KAPELEVICH, L. SDDP.jl: a Julia package for stochastic dual dynamic programming. **INFORMS Journal on Computing**, v. 33, p. 27–33, 2021. Cited in page 27.

FUJIMOTO, S.; HOOF, H. van; MEGER, D. **Addressing Function Approximation Error in Actor-Critic Methods**. 2018. Disponível em: https://arxiv.org/abs/1802.09477. Cited in page 38.

GJELSVIK, A.; BELSNES, M. M.; HAUGSTAD, A. An algorithm for stochastic medium-term hydrothermal scheduling under spot price uncertainty. In: **Proceedings of 13th Power Systems Computation Conference**. [S.l.: s.n.], 1999. Cited in page 16.

GRONDMAN, I. et al. Solutions to finite horizon cost problems using actor-critic reinforcement learning. In: **The 2013 International Joint Conference on Neural Networks (IJCNN)**. [S.l.: s.n.], 2013. p. 1–7. Cited in page 20.

HAARNOJA, T. et al. **Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor**. 2018. Disponível em: https://arxiv.org/abs/1801.01290. Cited in page 38.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. **Neural Networks**, v. 2, n. 5, p. 359–366, 1989. ISSN 0893-6080. Disponível em: https://www.

sciencedirect.com/science/article/pii/0893608089900208. Cited in page 21.

IEA. **Renewables 2024**. https://www.iea.org/reports/renewables-2024, 2024. Cited in page 12.

ISRAILOV, S. et al. Reinforcement learning approach to control an inverted pendulum: A general framework for educational purposes. **Plos one**, Public Library of Science San Francisco, CA USA, v. 18, n. 2, p. e0280071, 2023. Cited in page 20.

KAUFMANN, T. et al. **A Survey of Reinforcement Learning from Human Feedback**. 2024. Disponível em: https://arxiv.org/abs/2312.14925. Cited in page 13.

LEI, L. et al. Dynamic energy dispatch based on deep reinforcement learning in iot-driven smart isolated microgrids. **IEEE Internet of Things Journal**, v. 8, n. 10, p. 7938–7953, 2021. Cited in page 20.

LILLICRAP, T. P. et al. Continuous control with deep reinforcement learning. **arXiv preprint arXiv:1509.02971**, 2015. Cited in page 23.

MACEIRA, M. E. P.; PENNA, D. D. J.; DAMÁZIO, J. M. Geração de cenários sintéticos de energia e vazão para o planejamento da operação energética. **Cadernos do IME-Série Estatística**, v. 21, p. 11–11, 2006. Cited in page 14.

MOLZAHN, D. K.; HISKENS, I. A. et al. A survey of relaxations and approximations of the power flow equations. **Foundations and Trends® in Electric Energy Systems**, Now Publishers, Inc., v. 4, n. 1-2, p. 1–221, 2019. Cited in page 14.

NAZARE, F.; STREET, A. Solving multistage stochastic linear programming via regularized linear decision rules: An application to hydrothermal dispatch planning. **European Journal of Operational Research**, v. 309, n. 1, p. 345–358, 2023. ISSN 0377-2217. Disponível em: https://www.sciencedirect.com/science/article/pii/S0377221722010074. Cited in page 12.

PEREIRA, M. V. F.; PINTO, L. M. V. G. Multi-stage stochastic optimization applied to energy planning. **Mathematical Programming**, v. 52, n. 1, p. 359–375, May 1991. ISSN 1436-4646. Disponível em: https://doi.org/10.1007/BF01582895. Cited in page 12.

ROSEMBERG, A. et al. **Efficiently Training Deep-Learning Parametric Policies using Lagrangian Duality**. 2025. Disponível em: https://arxiv.org/abs/2405.14973. Cited in page 12.

ROSEMBERG, A. W. et al. Assessing the cost of network simplifications in long-term hydrothermal dispatch planning models. **IEEE Transactions on Sustainable Energy**, v. 13, n. 1, p. 196–206, 2022. Cited in page 12.

SCHULMAN, J.; CHEN, X.; ABBEEL, P. **Equivalence Between Policy Gradients and Soft Q-Learning**. 2018. Disponível em: https://arxiv.org/abs/1704.06440. Cited in page 20.

SILVER, D. et al. **Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm**. 2017. Disponível em: https://arxiv.org/abs/1712.01815. Cited 2 times in pages 13 and 38.

SILVER, D. et al. Deterministic policy gradient algorithms. In: XING, E. P.; JEBARA, T. (Ed.). **Proceedings of the 31st International Conference on Machine Learning**. Bejing, China: PMLR, 2014. (Proceedings of Machine Learning Research, 1), p. 387–395. Disponível em: https://proceedings.mlr.press/v32/silver14.html. Cited in page 23.

SIVAMAYIL, K. et al. A systematic study on reinforcement learning based applications. **Energies**, v. 16, n. 3, 2023. ISSN 1996-1073. Disponível em: https://www.mdpi.com/1996-1073/16/3/1512. Cited in page 13.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: an introduction, 2nd edn. Adaptive computation and machine learning**. [S.l.]: The MIT Press, Cambridge, 2018. Cited in page 21.

TSITSIKLIS, J.; ROY, B. V. An analysis of temporal-difference learning with function approximation. **IEEE Transactions on Automatic Control**, v. 42, n. 5, p. 674–690, 1997. Cited in page 20.

WATKINS, C. J. C. H.; DAYAN, P. Q-learning. **Machine Learning**, v. 8, n. 3, p. 279–292, May 1992. ISSN 1573-0565. Disponível em: https://doi.org/10.1007/BF00992698. Cited in page 19.