# AFU study

## Abstract

## Introduction

## Related Work

Several algorithms have been developed to address the challenges of reinforcement learning in both online and offline settings. Among these, Soft Actor-Critic (SAC), Implicit Q-Learning (IQL), and Calibrated Q-Learning (Cal-QL) stand out for their performance and conceptual innovations.

Soft Actor-Critic (SAC) [Biblio] is a widely used off-policy deep reinforcement learning algorithm based on the maximum entropy framework. SAC jointly learns a stochastic policy (actor) and two Q-value functions (critics), with a value function used for stability. The actor is updated to maximize expected reward while also maximizing entropy, encouraging exploration and robustness. SAC is highly sample-efficient and performs well in continuous control tasks, but it relies heavily on accurate Q-value estimation and often struggles in purely offline settings due to value overestimation and distributional shift between the behavior and target policies.

To address some of the challenges in offline reinforcement learning, Implicit Q-Learning (IQL) [Biblio] introduces a conservative update strategy that avoids explicit policy learning. Instead of using the actor-critic paradigm, IQL trains a Q-function and a value function from a fixed dataset, using quantile regression to mitigate overestimation. The policy is then implicitly defined via advantage-weighted regression, without requiring interaction with the environment. This separation makes IQL robust to distribution shift and improves performance in the offline RL setting.

Building on the idea of conservative Q-learning, Calibrated Q-Learning (Cal-QL) [Biblio] further refines Q-value estimation by incorporating calibration techniques. The motivation behind Cal-QL is that many offline RL failures stem from poor calibration of the learned Q-function, which can result in incorrect value targets and unstable learning. Cal-QL introduces a calibration loss that penalizes Q-values inconsistent with observed returns, thereby aligning the learned Q-function more closely with the empirical data distribution. This leads to more reliable policy evaluation and selection, particularly in high-stakes or high-variance environments.

[Expliquer AFU]

## Methods

### Environment

In order to test our algorithms, we wrapped the gymnasium environments in personalized wrapper that contains the following methods: _set_state, get_obs, get_observation_space and get_action_space.

### Algorithms

We reimplemented DDPG, SAC, AFU, IQL and Cal-QL with the BBRL library.

## Experimental study

### Off and On-Policy Training

*First Results.*

We evaluated AFU on the CartPoleContinuous environment and obtained results that closely match those observed using Mr. Perrin's original implementation. [These results are included in the figures attached to this report] – Mettre les figures. AFU significantly outperforms SAC on this task. Notably, AFU achieves these results in only 50k iterations, compared to the 200k required by SAC. As shown in the histograms, AFU also demonstrates greater consistency, suggesting improved stability and reliability during training.

However in the Pendulum environment, despite multiple runs, AFU performs poorly, failing to converge even after 1 million steps. In contrast, SAC reliably converges in under 50k steps. Given these results, we did not proceed with Off-Policy training for AFU on this environment.

Finally, we conducted experiments on LunarLander. This time, AFU performed well and successfully converged. We runned 1 million steps, but analysis of the learning curve indicates that convergence was already achieved by around 200k steps. We provide histograms showing performance distributions, which (albeit potentially exaggerated) suggest that AFU substantially outperforms SAC on this task.

Encouraged by these results, we tested AFU in an Off-Policy setting on LunarLander, using 200k training steps. Unfortunately, the results were similar to those of SAC: neither algorithm achieved meaningful convergence. To verify whether this was due to insufficient data, we extended the training to 1 million steps, but AFU still failed to converge under Off-Policy training.

These results suggest that while AFU performs competitively–and at times better than SAC–in On-Policy settings, its Off-Policy capabilities remain limited, particularly in more complex environments like LunarLander.

*Make AFU more intersting than SAC in an Off-Policy setting.*

Our preliminary results suggest that the initial hypothesis – that AFU can effectively learn from uniformly generated state-action data – is likely incorrect. This observation has led us to explore potential reasons behind this behavior and to design alternative experimental settings that could both validate AFU's utility and demonstrate its Off-Policy potential compared to algorithms such as SAC. We outline two such experiments below.

1. Varying the Degree of Policy-Driven Behavior via an Epsilon Parameter

A straightforward experimental modification involves introducing a tunable parameter $\varepsilon \in [0, 1]$. At each step of data generation, a uniform random variable is sampled. If the sample is less than $\varepsilon$, a state-action pair is drawn uniformly at random, mimicking our prior Off-Policy setting. Otherwise, the action taken is the one prescribed by the current policy, corresponding to the On-Policy setting.

This setup allows us to generate a range of datasets interpolating between the On-Policy regime ($\varepsilon = 0$) and fully random Off-Policy regime ($\varepsilon = 1$). We propose to plot the performance across

varying $\varepsilon$ values, using statistical summaries such as whisker plots (e.g., Q1, IQM, Q3) to visualize trends. If AFU is truly more Off-Policy capable than SAC, its performance degradation should be less pronounced as $\varepsilon$ increases. This experiment offers a simple yet effective framework for assessing the Off-Policy robustness of learning algorithms. [Résultats à inclure dans le rapport]

2. State-Space Constraints and the Role of Accessible States

To better understand the mechanisms behind AFU's behavior, we return to the tabular case. In this setting, it is possible to explicitly construct a lookup table that maps each state to the optimal action over an infinite horizon. When the state space is sufficiently small, exhaustive exploration via random state-action pairs can, in principle, lead to convergence—even for algorithms that are not inherently Off-Policy.

For example, in a simple 4x5 maze with 4 actions per state, the total number of state-action pairs is only 80. After sufficient repetitions (e.g., 80k samples), learning the optimal action mapping becomes feasible via brute force. In deep reinforcement learning, such a lookup table is no longer feasible due to the continuous and high-dimensional nature of the state and action spaces. However, the underlying goal — learning a function that approximates such a mapping — remains intact.

This perspective sheds light on our results in CartPoleContinuous. Despite the continuous nature of the problem, its effective state-space dimensionality is low. The first state variable (cart position) exhibits a high degree of symmetry and repetition, and we further constrained velocities to lie between $-8$ and $+8$. These simplifications reduce the practical complexity of the environment. Thus, brute-force learning may still succeed, even when relying on random data.

Importantly, much of the theoretical state-space is not reachable during normal episodes. For example, while velocity can range from $-\infty$ to $+\infty$, in practice, only a narrow subspace of values is observed. We refer to this subspace as the accessible state space. Uniformly sampling across the full space may therefore introduce states that are highly unrealistic and ultimately irrelevant to learning a useful policy.

This observation generalizes. Consider a tabular environment with 100B states, where the optimal trajectory only spans 10 states. Random sampling would rarely encounter meaningful states, and learning would stagnate. In LunarLander, we observed behaviors in which the agent exits the visible screen or exhibits unusual dynamics such as rotating mid-air — scenarios that are rarely encountered under a standard policy. Learning accurate Q-values for such states may come at the expense of learning for states within the accessible region.

Based on this, we propose a second experiment. First, define bounds on each dimension of the state space to better approximate the accessible subspace. Training the algorithm solely within these constraints may yield better convergence. We can then extend this by performing random walks through the environment: sample an initial state within the constrained bounds, and then take random actions for each step. This would naturally constrain the set of states encountered during training.

Such an experimental setup simulates learning from behavior that is entirely unrelated to the agent's own policy. A critic capable of learning accurate Q-values in this setting – as AFU purports to do – should outperform critics like SAC, which rely on the actor for accurate Q-value estimation. This setup thus offers a clearer test of the Off-Policy learning capacity of AFU.

The same $\varepsilon$-based framework from the first experiment could be employed here, offering another way to measure degradation in performance as the data distribution diverges from the policy. [Résultats à inclure dans le rapport]

# Conclusion