Natural Gradient Optimistic Actor Critic

Natural Gradient Optimistic Actor Critic
Master thesis by Niklas Kappes
Date of submission: 31. Oktober 2023

1. Review: João Carvalho
2. Review: Jan Peters
Darmstadt

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Erklärung zur Abschlussarbeit gemäß § 22 Abs. 7 APB TU Darmstadt

Hiermit erkläre ich, Niklas Kappes, dass ich die vorliegende Arbeit gemäß § 22 Abs. 7 APB der TU Darmstadt selbstständig, ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe. Ich habe mit Ausnahme der zitierten Literatur und anderer in der Arbeit genannter Quellen keine fremden Hilfsmittel benutzt. Die von mir bei der Anfertigung dieser wissenschaftlichen Arbeit wörtlich oder inhaltlich benutzte Literatur und alle anderen Quellen habe ich im Text deutlich gekennzeichnet und gesondert aufgeführt. Dies gilt auch für Quellen oder Hilfsmittel aus dem Internet.

Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§ 38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, 31. Oktober 2023

N. Kappes

Abstract

Exploration constitutes a fundamental facet within the realm of Reinforcement Learning, exerting considerable influence on learning efficiency, policy quality, and agent performance. This significance amplifies when exploration must be conducted in real-world scenarios, as a proficient exploration policy becomes pivotal in curtailing runtime expenses and expediting superior outcomes. Despite extensive research efforts, exploration in Reinforcement Learning (RL) remains an ongoing challenge, attracting substantial attention from the research community.

Exploration techniques grounded in uncertainty utilization leverage epistemic uncertainty information to expedite convergence. This thesis introduces a natural gradient-based optimistic exploration method, denoted as Natural Gradient Optimisitic Actor Critic, designed to establish a directional informed exploration strategy. Natural Gradient Optimisitic Actor Critic (NGOAC) effectively mitigates pessimistic underexploration and exhibits reduced sensitivity to hyperparameters, ultimately fostering a stable and resilient learning process. It achieves good results in non-bang-bang optimal environments while maintaining competitive computational efficiency.

Zusammenfassung

Die Exploration bildet einen grundlegenden Aspekt im Bereich von Reinforcement Learning und übt erheblichen Einfluss auf die Lerneffizienz, die Qualität der Strategie und die Leistung der Agenten aus. Diese Bedeutung verstärkt sich, wenn die Erkundung in realen Szenarien durchgeführt werden muss, da hier eine herausragende Explorationsstrategie von entscheidender Bedeutung ist, um die Laufzeitkosten zu senken und schneller bessere Ergebnisse zu erzielen. Trotz umfangreicher Forschungsbemühungen bleibt die Erforschung von RL eine ständige Herausforderung und erregt erhebliche Aufmerksamkeit in der Forschungsgemeinschaft.

Erkundungstechniken, die auf der Nutzung von Unsicherheiten basieren, nutzen epistemische Unsicherheitsinformationen, um die Konvergenz zu beschleunigen. In dieser Arbeit wird eine auf natürlichem Gradienten basierende optimistische Explorationsmethode vorgestellt, die als Natural Gradient Optimisitic Actor Critic bezeichnet wird und darauf ausgelegt ist, eine richtungsinformierte Explorationsstrategie zu etablieren. NGOAC mildert effektiv die pessimistische Untererkundung und zeigt eine verringerte Empfindlichkeit gegenüber Hyperparametern auf, was letztendlich einen stabilen und robusten Lernprozess fördert. Es erzielt gute Ergebnisse in nicht-bang-bang optimalen Umgebungen und behält gleichzeitig wettbewerbsfähige Recheneffizienz bei.

Contents

1. Introduction

Reinforcement Learning has made remarkable progress in recent years, showcasing its potential in a wide range of applications, from game-playing agents to robotic control systems. At the heart of RL lies the fundamental challenge of exploration, a key aspect that distinguishes RL from supervised learning. In RL, agents must not only learn to exploit their current knowledge to maximize rewards but also explore the environment to discover new, potentially more rewarding actions and states.

The exploration problem in RL is both intriguing and multifaceted. It encompasses questions of how an agent should balance between exploiting its current knowledge and exploring the unknown, how it can efficiently sample from the vast state and action spaces, and how it can adapt its exploration strategy to suit different environments and tasks. Furthermore, exploration in RL is a fundamental aspect that impacts an agent's learning efficiency, the quality of its learned policies, and its overall performance. Exploration in RL is a sophisticated challenge, with key issues including the trade-off between exploration and exploitation, the curse of dimensionality in high-dimensional spaces, sparse and delayed rewards, and the uncertainties introduced by stochastic environments. Handling non-stationarity, achieving sample efficiency, and balancing optimism and pessimism in exploration strategies are also vital aspects. Additionally, addressing the heterogeneous demands of different RL tasks and finding the right balance between intrinsic and extrinsic motivations for exploration present ongoing challenges. Tackling these issues is crucial for advancing RL algorithms and making them more effective in complex real-world applications.

While there exist alternative exploration strategies rooted in intrinsic motivation or more advanced methods, this thesis confines its focus to uncertainty-oriented exploration. Uncertainty-based exploration entails the selection of actions that prioritize the agent's comprehension of the environment, typically through quantifying uncertainty measures like variance or entropy. This approach motivates the agent to investigate regions where

outcomes are uncertain, potentially enhancing its policy learning and decision-making capabilities.

An example of an uncertainty-based approach is Optimisitic Actor Critic. OAC optimizes an linearly approximated Upper Confidence Bound derived from the state-action value function. This optimization results in a directional informed exploration policy and mitigates pessimistic underexploration by incorporating optimism, leading to superior results in terms of convergence speed in continuous control task with well tuned hyperparameters. In addition to the substantial benefits offered by OAC, a more in-depth analysis of the optimization problem reveals a notable dependence of the exploration policy on the optimization bound. This dependence can potentially result in the generation of bang-bang-like actions for exploration. This thesis delves into the hyperparameter sensitivity of OAC and explores its impact in various environments, considering both theoretical and experimental perspectives.

The primary objective of this thesis is to enhance the optimization approach, aiming to develop an exploration policy that retains the advantages of OAC while exhibiting reduced sensitivity to hyperparameters, thus avoiding the tendency to generate a bang-bang exploration strategy. An extension of OAC that utilizes the natural gradient, denoted as NGOAC, is introduced. Unlike OAC, Natural Gradient Optimisitic Actor Critic optimizes the Upper Confidence Bound (UCB) without relying on a linear approximation of the objective function. It leverages Hessian information to improve the exploration policy while reformulating the update to only incorporate first-order information. The theoretical merits of NGOAC are exemplified through a straightforward example and subsequently evaluated in the Cartpole, Ant, and Humanoid environments. To underline the necessitie of the natural gradient, OAC-RepTrick is introduced. This approach optimizes a reparameterized UCB using Stochastic Gradient Descent (SGD) to derive an exploration policy. In addition to an extensive hyperparameter search to assess the sensitivity of various exploration strategies, this thesis investigates the impact of a prioritized replay buffer. This buffer incorporates additional online data into the off-policy agent update.

In summary, this thesis makes the following contributions:

- The theoretically and empirically hyperparameter sensitivity of OAC and its inherent tendency to generate a bang-bang exploration strategy in continuous control tasks is demonstrated.
- NGOAC, an optimistic exploration strategy, is proposed, which optimizes the exploration policy using a natural gradient.

The thesis is structured as follows: In Chapter 2, we delve into related work in the fields of Actor Critic methods, experience replay, exploration in RL, and the fundamentals of bang-bang optimal control. Continuing in Chapter 3 with the introduction of maximum entropy RL in combination with Soft Actor Critic (SAC), followed by an in-depth explanation of OAC. In Chapter 4, the bang-bang tendency of OAC is further elucidated and exemplified using a toy example. Additionally, the natural gradient version, NGOAC, is introduced. Chapter 5 encompasses the hyperparameter experiments and investigations of a prioritized replay buffer. Finally, Chapter 6 concludes this thesis and provides directions for future research.

2. Related Work

In Reinforcement Learning, an infinite Markov Decision Process (MDP) [1, 2] defined by the tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where an agent observes the current environment state $s_t \in \mathcal{S}$ and performs an action $a_t \in \mathcal{A}$ is considered. This leads to a new environment state $s_{t+1} \sim p(\cdot|s_t, a_t)$, where $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, \infty)$ represents the probability density of the state transitions. For each transition in the environment the agent receives a reward $r : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$. The overall goal of the agent is to maximize the total expected reward

$$J = \sum_t \mathbb{E}_{(s_t, a_t) \sim \rho_\pi}[\gamma^t \cdot r(s_t, a_t)] \tag{2.1}$$

where $\rho_\pi(s_t)$ and $\rho_\pi(s_t, a_t)$ denote the state and state-action marginals of the trajectory distribution produced by the policy $\pi(a_t|s_t)$ and γ defines the discount factor. Accordingly, a trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, ...)$ is obtained by sequentially generated actions from the policy π [3].

The state-value function for policy π quantifies the current state and is defined as the expected return when starting in state s_t and following π

$$V_\pi(s_t) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \cdot r_{t+k+1} \,\middle|\, s = s_t \right]. \tag{2.2}$$

Likewise the action-value function or Q-function defines the value of performing action a_t in state s_t while following policy π

$$Q_\pi(s_t, a_t) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \cdot r_{t+k+1} \,\middle|\, s = s_t, a = a_t \right] \tag{2.3}$$

2.1. Actor Critic Methods

Actor-Critic Policy Gradients incorporate the value function into the learning process. Represented as the critic, the value function is learned by evaluating the returns of the agent. In contrast to vanilla policy gradients where Equation 2.1 is differentiated directly, actor-critic methods improve the policy, also called actor, by updating the policy parameters in the direction suggested by the critic. The combination of value function estimation and policy gradients reduces the variance of the policy gradient, and therefore, the policy converges with fewer samples. Also, actor-critic methods can suit different problems and objectives because the critic can be represented by different types, such as state-value function (Equation 2.2), state-action-value function (Equation 2.3) [4], or advantage function [5]. Furthermore, by integrating Temporal Difference Learning (TD Learning), the critic's estimation can be bootstrapped, allowing for policy updates at each step, as opposed to episode-based learning [3]. Besides the advantages of using a separate actor and critic network, an inaccurate or inconsistent value estimation can affect the quality of the policy update because the policy gradient depends on the precision of the critic's value function approximation. Moreover the actor and the critic can interfere with each other and influence the learning process as they may suffer from instability and divergence.

A2C is a synchronous and A3C is a asynchronous variant of the advantage actor-critic algorithm. They utilize an advantage function to measure the relative value of each action [5]. DDPG, on the other hand, is a deterministic policy gradient algorithm that employs deep neural networks to represent both the actor and the critic [4]. In contrast, SAC is a soft actor-critic algorithm that not only maximizes the expected return but also places emphasis on maximizing the entropy of the policy [6]. All of these algorithms are well-suited for handling high-dimensional and continuous action spaces. They incorporate a variety of techniques, such as the use of replay buffers, target networks, twin networks, and reparameterization tricks, to enhance their overall performance and stability.

Indeed, there exist a multitude of alternative RL methods beyond policy gradient approaches. Value-based methods involve the estimation of the state- or state-action-value function corresponding to the optimal policy and do not explicitly learn the policy itself. Consequently, the policy is derived by maximizing the underlying value function [7, 8]. In model-based RL the state transition $p(\cdot|s_t, a_t)$ is learned and used to improve the policy that is typically used for planning [9, 10, 11].

2.2. Experience Replay

Experience Replay stores a fixed number of the most recently collected transitions in a replay memory. Further a uniformly sampled batch of experiences is used to update the agent during training. Mixing more and less recent experiences can break the temporal correlation between consecutive experiences and enables the agent to train on rare and informative experiences multiple times [12]. Both improving sample efficiency and stability leads to better performance especially when training a neural network function approximators with stochastic gradient descent. For example in Neural Fitted Q-Iteration [13] and Deep Q-Learning [7, 14] this leads to better overall performance.

Sampling from a replay memory indirectly influences the learning process which can result in superior performance using an informed sampling method. The key component of prioritized replay is the criterion by which the importance of each experience is measured. Prioritized Experience Replay [15] extends classic prioritized sweeping ideas [16] and prioritizes each transition using the Temporal Difference (TD) error such that the experience with the largest absolute TD error is assigned to a higher prioritization. It quantifies how unexpected a transition is since it measures the distance to the next-step bootstrap estimate. Biased sampling can be very helpful in RL since the data distribution depends on the agent's policy and can also lead to improvements when an agent has to deal with sparse rewards. This is shown for example in DDPG [17], Prioritized Dueling DQN [18], UNREAL [19], DQfD [20] and Rainbow [21].

To incorporate the experiences of distributed actors into the learning process, Distributed Prioritized Experience Replay accumulates the experiences of distributed actors which interact with the environment using a shared neural network [22]. The Ape-X architecture decomposes the standard Deep Reinforcement Learning algorithm into two parts, the acting and the learning part as done as in Gorila [23].

Furthermore, Hessel et al. show that prioritization is the most important ingredient contributing to the agent's performance [21].

2.3. Exploration in Reinforcement Learning

Exploration methods in RL can be classified into three main types [24].

Intrinsic motivation-oriented exploration originates from human development where children often employ less goal-oriented exploration but use curiosity to gain knowledge about the world. This psychological inspired behavior intrinsically rewards exploration activities, where typically reward-agnostic information as the prediction error [25, 26], novelty or information gain is used to design intrinsic rewards.

Uncertainty based exploration originates from the Optimisim in the Face of Uncertainty (OFU) principle. Uncertainty is quantified by epistemic and aleatoric uncertainty to measure the sufficiency of learning and the intrinsic stochasticity to derive efficient exploration. To directly explore state-action pairs with a high uncertainty, an exploration bonus can be added based on a specific uncertainty measurement. Optimistic action-selection is achieved by maximizing the optimistic value function or also known as UCB

$$a_t = \arg\max_a Q^{UCB}(s_t, a) \tag{2.4}$$

where the optimistic value function is defines as

$$Q^{UCB}(s_t, a_t) = Q(s_t, a_t) + \beta \cdot Uncertainty(s_t, a_t) \tag{2.5}$$

Here, the level of optimism can be controlled by the hyperparameter β.

In contrast to incorporating an additional exploration bonus, the action selection can be greedy to a sampled value function. The posterior distribution of the Q-function is estimated through parametric or non-parametric posterior, then the sampled Q-function is used to select actions when interacting with the environment for a whole episode.

$$a_t = \arg\max_a Q_\theta(s_t, a) \text{ where } Q_\theta \sim Q^{Posterior} \tag{2.6}$$

Thompson Sampling [27] enables the agent to perform deep exploration [28, 29] and has advantages in long-horizon exploration tasks.

Besides intrinsic motivation- and uncertainty-oriented exploration advanced strategies adapt and utilize the underlying neural network structures to explore with parametric

noise [30, 31, 32] or are constrained to restrictions of the environment and the task itself to achieve safe exploration [33, 34].

In general directed exploration strategies exhibit higher levels of sample efficiency compared to undirected techniques. However, it is essential to acknowledge that some directed exploration strategies such as bonus-based exploration (BBE) suffer from three main limitations. The policy is biased after any finite number of steps in the environment because it is learned on a combination of the task reward and the exploration bonus. Secondly, the agent slowly adapts the policy and the rewards within a single episode which leads to sample inefficiency. Lastly, unseen transitions are not incorporated into the exploration behavior resulting in a lack of optimism and increases the time required to visit unseen and enables the agent to learn a better policy [35].

To mitigate this Decoupled Exploration and Exploitation Policies (DEEP) separately learns an unbiased task policy and an exploration policy also called behavior policy that is used to select actions during training. This leads to strictly better performance and sample efficiency even in dense and sparse environments [36]. Further UFO extends this by incorporating optmism into the action selection process [35].

2.4. Bang-Bang Optimal Control

In continuous control problems, policies are commonly represented as continuous probability distributions, with Gaussian distributions being a popular choice. This representation allows for more refined decision-making compared to simpler policies like discretized controllers. In many standard continuous control tasks, learned agents tend to result in bang-bang policies, even when using Gaussian policies. It's noteworthy that bang-bang controllers are frequently optimal and can achieve performance that is on par with state-of-the-art (SOTA) algorithms [37].

Non-bang-bang optimal tasks can be classified based on either the task's inherent goal or the reward structure it employs. For instance, consider the Cartpole stabilization task from the MuJoCo set of environments [38]. In this task, the inherent goal is to stabilize the pole, which naturally leads to an optimal control policy that necessitates actions with minimal magnitude to achieve stability. Introducing various action costs can alter the reward structure, potentially leading to the emergence of non-bang-bang optimal policies.

$$\max \int_0^T r(s(t)) - c(a(t))dt \tag{2.7}$$

Maximizing only the state reward in Equation 2.8 the agent learns a bang-bang control even under a Gaussian policy. Adding a linear action cost in Equation 2.9 results in minimizing the fuel-type cost inducing a bang-off-bang control. Minimum energy-type cost on the basis of quadratic action costs in Equation 2.10 generally leads to a non-bang-bang optimal control [37].

$$c(a(t)) = 0 \tag{2.8}$$
$$c(a(t)) = |a(t)| \tag{2.9}$$
$$c(a(t)) = a(t)^2 \tag{2.10}$$

3. Preliminaries

This chapter provides a comprehensive introduction to maximum entropy RL within the framework of SAC in Chapter 3.1. Furthermore, it delves into the optimistic exploration strategy OAC in Chapter 3.2, discussing the UCB based on the critic's estimates and the formulation of the optimization problem.

3.1. Soft Actor Critic

SAC is an off-policy maximum entropy algorithm that is based on soft Q-learning [39] and optimizes a stochastic policy. In contrast to the classical RL objective in Equation 2.1, the cumulative reward in entropy regularized RL [40, 41, 42] incorporates the entropy of the policy distribution

$$J_{entropy} = \sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi} \left[\gamma^t \left(r(s_t,a_t) + \alpha\mathcal{H}\left(\pi(\dot{|}s_t)\right)\right)\right] \tag{3.1}$$

Accordingly, the agent tries to maximize the trade-off between the expected return and the entropy of the policy distribution [43]. Maximizing the entropy ensures a minimum randomness of the policy distribution such that the agent acts as randomly as possible while trying to succeed at the task. This highly correlates to the exploration-exploitation tradeoff in RL. Enforcing higher entropy results in a more exploratory training strategy that can accelerate learning later on and prevent the policy from converging to a sub-optimal solution. Concurrently the entropy term in Equation 3.1 limits the exploitation potential of the agent that can mitigate the agent to exploit the optimal solution. This exploration-exploitation dilemma is still an open problem in RL and is addressed by different exploration strategies as described in Chapter 2.3.

SAC is an Actor-Critic algorithm where the policy gradient is computed by optimizing the expected performance that the critic predicts [6]. The critic is trained to model future rewards of the agent and is learned using TD Learning, as described in Chapter 2.1. Maximizing the learned state-action value function approximation inherently induces an overestimation bias, originating from the critics' predictions and, consequently, potential inaccuracies in these predictions. The model is parameterized with two, independently-initialized networks to counteract overestimation in off-policy RL. Using the min double Q trick [44] the target of the state-action value function is defined by the minimum value over both Q-function approximatons

$$y(r_t, s_{t+1}) = r_t + \gamma \left(\min_{j=1,2} Q_{\bar{\theta}_j}(s_{t+1}, a) - \alpha \log \pi_\phi(a|s_{t+1}) \right) \quad \text{with } a \sim \pi_\phi(\cdot|s_{t+1}). \tag{3.2}$$

Accordingly, both Q-value approximatons are optimized by minimizing the quadratic loss function

$$\min_{\theta_j} \mathcal{L}_Q(\theta_j, \mathcal{D}) = \mathbb{E}_{(s_t, a_t, r_t, s_{t+1}) \sim \mathcal{D}} \left[\left(Q_{\theta_j}(s_t, a_t) - y(r_t, s_{t+1}) \right)^2 \right]. \tag{3.3}$$

To enhance the stability of the training process, the update of the critic uses target networks, which are computed based on an exponentially moving average on the weights of the value network [14]. Empirical findings demonstrate that using two soft Q-functions significantly accelerates the training procedure, particularly in the context of addressing more challenging tasks with a high-dimensional action space [45].

The optimization objective of the policy in Equation 3.1 can be reformulated by incorporating the value function learned as per Equation 3.3.

$$\max_{\phi} \mathcal{L}_\pi(\phi) = \mathbb{E}_{a \sim \pi_\phi} \left[\min_{j=1,2} Q_{\bar{\theta}_j}(s, a) - \alpha \log \pi_\phi(a|s) \right] \tag{3.4}$$

In Equation 3.4, the distribution depends on the policy parameter that makes computing the policy gradient unfeasible. The expectation over actions can be reformulated as an expectation over noise using the reparameterization trick [46] such that the distribution only depends on the noise.

The entropy regularization coefficient in Equation 3.1 influences the scaling factor of the reward function. Therefore, a suitable temperature is crucial for training, and a sub-optimal temperature can lead to degraded performance. In [45], an Autotuned Temperature version of SAC is introduced, employing an automatic gradient-based temperature tuning. It adjusts the expected entropy solving a constrained optimization problem such that a minimum expected entropy is fulfilled. This makes SAC easier adaptable to different tasks and reduces the time of hyperparameter tuning.

Broadly, SAC exhibits superior performance compared to SOTA model-free RL algorithms, including both off-policy Deep Deterministic Policy Gradient (DDPG) and on-policy Proximal Policy Optimization (PPO). Furthermore, it also improves sample-efficiency in comparison to DDPG. Particularly in complex and high-dimensional environments, such as the Humanoid task where conventional off-policy algorithms often encounter difficulty or converge to a suboptimal policy [47], the stable learning process in SAC becomes pronounced. The adaptability of SAC extends to real-world learning tasks because of its robustness and sample efficiency. This is evidenced by its successful application in tasks such as acquiring locomotion skills with a Minitaur robot or accomplishing manipulation tasks using a 3-finger dexterous robotic hand, as highlighted in the work by [6].

Illustrated within the context of SAC, it is a general observation that the incorporation of the minimum double Q-trick to augment training stability is not without its caveat concerning exploration potential. This consideration arises from updating both Q-function approximation with the same shared target value. The consistent generation of pessimistic target values mitigates the issue of overestimation bias through bootstrapped TD Learning targets. However, a direct consequence of pessimism is reduced and suboptimal exploration, owing to underestimation bias, as discussed in [48, 49]. This is mitigated within the maximum entropy framework, where a lower bound is imposed on the entropy of the policy distribution to guarantee a minimum level of stochastic exploration behavior. Moreover, this emphasizes the necessity and underscores the demand of exploration methods based on the OFU principle [50]. The inherent limitation of shared pessimistic target values is further investigated in the computation of the Bellman error by [51]. Their study reveals that using independent target values can surpass the performance of SOTA algorithms.

3.2. Optimisitic Actor Critic

In the study conducted by [49], they pinpoint two main issues in the exploration strategy of SAC: a) pessimistic underexploration as a result of consistently updating the critic

with pessimistic target values and b) the presence of directional uninformedness in the exploration behavior using a symmetric Gaussian policy as demonstrated by SAC. To address these concerns, the researchers propose an optimistic exploration extension, denoted as OAC. This approach involves the computation of a local exploration policy, which is based on the current learned policy. The exploration policy is devised by maximizing an Upper Confidence Bound derived from the critic, effectively circumventing the pessimistic underexploration problem.

$$\begin{aligned} \mu_e, \Sigma_e &= \arg\max_{\mu,\Sigma} \mathbb{E}_{a\sim\mathcal{N}(\mu,\Sigma)}\left[\hat{Q}_{UCB}(s,a)\right] \\ &\text{s. t. } \mathcal{KL}(\mathcal{N}(\mu,\Sigma),\mathcal{N}(\mu_T,\Sigma_T)) \leq \delta \end{aligned} \tag{3.5}$$

The optimization objective is constrained by a maximum Kullback-Leibler Divergence (KL) divergence denoted as δ, which quantifies the dissimilarity between the distributions of the exploration policy and the target policy, as expressed in Equation 3.5. This ensures that the exploration policy maintains similarity to the target policy, thus preserving the stability of the optimization process.

The UCB on the Q-value function draws upon both Q-function approximations employed within the context of SAC, with its calculation relying on the assessment of their mean and variance:

$$\begin{aligned} Q_{UCB}(s,a) &= \mu_Q(s,a) + \beta_{UB} \cdot \sigma_Q(s,a) \\ \mu_Q(s,a) &= \frac{1}{2}\left(Q_{\theta_1}(s,a) + Q_{\theta_2}(s,a)\right) \\ \sigma_Q(s,a) &= \frac{1}{2}\left|Q_{\theta_1}(s,a) - Q_{\theta_2}(s,a)\right| \end{aligned} \tag{3.6}$$

To facilitate efficient computation and ensure faster optimization, the UCB defined in Equation 3.6 is linearly approximated at the mean μ_T of the target policy.

$$\hat{Q}_{UCB}(s,a) = a^T \left[\nabla_a Q_{UCB}(s,a)\right]_{a=\mu_T} + const \tag{3.7}$$

In the context of Gaussian policies, which are commonly used within the framework of SAC, it is possible to analytically compute the exploration policy in each step from scratch.

$$\mu_E = \mu_T + \frac{\sqrt{2\delta}}{\left\| \left[\nabla_a \hat{Q}_{UB}(s,a)\right]_{a=\mu_T} \right\|_{\Sigma_T}} \Sigma_T \left[\nabla_a \hat{Q}_{UB}(s,a)\right]_{a=\mu_T} \tag{3.8}$$
$$\Sigma_E = \Sigma_T$$

Accordingly, the optimized exploration policy is not constrained to share the same mean as the target policy, whereby it is not symmetric around the mean of the target policy. This enables exploration to be guided by directional information, thereby enhancing sample efficiency. The theoretical impact of the optimization problem is exemplified in Figure 3.1. Consequently, actions that are sampled from the exploration policy are concentrated on less explored regions of the state-action space, effectively reducing the need to sample actions that have previously been explored. Additional implementation details are expounded upon in Appendix A.1.

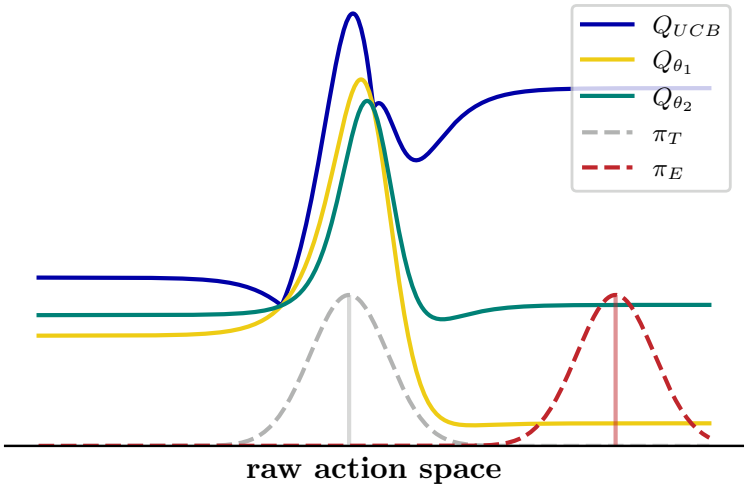


Figure 3.1.: Optimization result of OAC in a toy example. The value function approximations and policy distributions are shown against the raw action space.

Furthermore, the authors in [49] introduce a revised Q-target computation within the SAC framework. Through an locally independent optimistic exploration strategy as illustrated in Equation 3.8, the update of the target policy can adopt a more pessimistic approach,

yielding policy updates that are both more conservative and stable. Instead of employing the minimum double Q trick, the authors generalize the Lower Confidence Bound (LCB)

$$Q_{LCB}(s,a) = \mu_Q(s,a) + \beta_{LB} \cdot \sigma_Q(s,a) \tag{3.9}$$

to calculate the target Q-value in Equation 3.10 and simultaneously the update of the target policy, as presented in Equation 3.4. Consequently, the hyperparameter β_{LB} governs the extent of pessimism in the computation of the target Q-value, much like β_{UB} scales the degree of optimism in the exploration policy update, as detailed in Equation 3.2.

$$\begin{aligned} y(r_t, s_{t+1}) = r_t + \gamma \left(Q_{LCB}(s_{t+1}, a) - \alpha \log \pi_\phi(a|s_{t+1}) \right) \\ \text{with } a \sim \pi_\phi(\cdot|s_{t+1}) \end{aligned} \tag{3.10}$$

The experimental results in Chapter 5.1 demonstrate that OAC attains a state-of-the-art level of sample efficiency in continuous control tasks, as evaluated using the MuJoCo [38] continuous control benchmarks. OAC mitigates pessimistic underexploration through the locally optimized exploration policy, which facilitates the selection of actions from π_E that directly adjusts the critic estimate. Due to the mean shift of the exploration policy relative to the target policy, OAC also avoids directional uninformedness. The optimistic estimation in Equation 3.6 is only employed for optimizing the exploration policy in each step from scratch, ensuring that optimism does not induce overestimation bias as described in [45, 44]. Consequently, the critic and target policy continue to be updated using a lower-bound estimate.

The exploration-exploitation trade-off is also addressed by other works that employ an optimistic exploration strategy based on curiosity estimates derived from Q-function approximations, similar to the approach taken in OAC. The concept of reward-shifting, facilitated through a changing reward shifting constant, proves to be effective in striking a balance between exploration and exploitation. This involves reinforcing conservative exploitation through positive shifts in rewards, while employing negative reward shifts to stimulate curiosity-driven exploration, thus enhancing optimistic exploration [52]. Unlike OAC, it directly adapts the Q-function estimate of the critic, and notably, the exploration policy remains uninformed.

In the pursuit of mitigating the overestimation bias without incurring the downsides of overly pessimistic targets, Generalized Pessimism Learning (GPL) introduces a learnable penalty designed to enact such pessimism. The parameterized uncertainty regularizer

is deployed to ensure accurate estimation of the target action-values with low bias and variance. This regularizer is learned concurrently with the critic through a dual TD Learning approach, achieved by minimizing the bias in the target returns, as quantified by the epistemic uncertainty inherent in the critic's predicted return distributions [48]. Addressing the challenge of pessimistic underexploration involves a direct adaption of the target-values themselves, aimed to compensate the overestimation bias directly within the Q-function approximations. In contrast, OAC involves recalculating the optimistic exploration strategy each step from scratch, which only indirectly influences the critic.

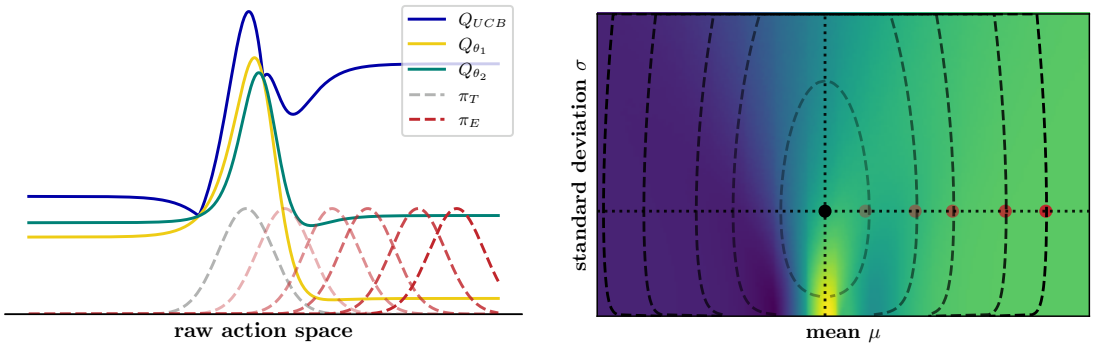
4. Natural Gradient Optimistic Actor Critic

In this chapter, the theoretical hyperparameter sensitivity of OAC is thoroughly explored, revealing its inclination to generate a bang-bang-like exploration strategy, as discussed in Chapter 4.1. Additionally, Chapter 4.2 introduces the natural gradient variant of OAC and shows the advantages of the optimization method.

4.1. Bang-Bang Exploration induced by Optimism

In accordance with the content presented in Chapter 3.2, OAC addresses fundamental concerns pertaining to pessimistic underexploration and directional uninformedness. However, it is noteworthy that the utilization of the linear approximation of the UCB as depicted in Equation 3.7 results in a mean shift of the probability distribution. In Equation 3.8, this shift's direction is determined through the computation of the gradient of the critic, while the length of the shift is primarily influenced by the KL constraint in Equation 3.5. This indicates a strong interdependence between the parameters of the exploration policy and the optimization constraint in a general context, emphasizing the necessity for a sufficiently good choice of the KL bound, as it directly influences the exploration behavior of the agent.

An improper hyperparameter choice can give rise to two distinct exploration behaviors. Firstly, when a small KL bound is chosen, it results in negligible or minimal mean shift, which in turn leads to a lack of optimistic exploration behavior. Consequently, the exploration policy explores the state-action space similar to when employing entropy-based exploration of SAC.
On the other hand, opting for a higher constraint value induces a more substantial mean shift, causing the exploration policy to diverge significantly from the target policy. The exploration policy parameters are defined in the raw action space, where it is a common setting in off-policy RL to map the sampled actions to the correct action space such that



(a) **(b)**

Figure 4.1.: Optimization results of OAC in a toy example for different KL constraints. The different exploration policies (red) are shown in the raw action space on the left (a). The different trust regions and the underlying objective function can be seen on the right (b).

the action boundaries are not violated. In practice, it is typically seen in SAC and similar algorithms that a $\tanh$ function is used for the action space mapping.

The optimization of the exploration policy, subject to substantial update length, results in a mean value characterized by a considerable magnitude in the raw action space. Consequently, the sampled actions exhibit high magnitudes, subsequently undergoing the transformation through a $\tanh$ function for mapping into the action space. This culminates in actions for exploration located either at or in close proximity to the boundaries of the action space and therefore produces a bang-bang like exploration behavior.

This may benefit efficient exploration in bang-bang optimal tasks; however, it tends to result in suboptimal exploration when applied to non-bang-bang optimal tasks as described in Chapter 2.4 and is observable in the experiments in Chapter 5.2.

4.2. Natural Gradient Descent

To mitigate the consequences of optimizing a linear approximation of the UCB, the optimization task can be addressed through the application of natural gradient descent [53]. This yields an exploration policy that 1) reduces sensitivity to the KL constraint, 2)

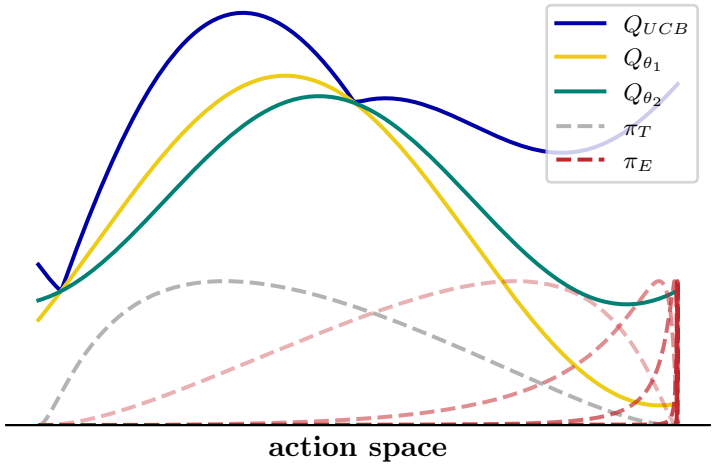


Figure 4.2.: Optimization results of OAC in a toy example for different KL constraints. The value function approximations and policy distributions are shown against the action space, mapped using a $\tanh$ function.

improves the optimization outcome resulting in a solution with higher UCB value and 3) also adapts the covariance of the exploration policy to align with the structure of the objective function.

In contradistinction to the formulation of the optimization objective in Equation 3.5 in OAC, the UCB as defined in Equation 3.6 is not linearly approximated. This distinction manifests in the following formulation of the optimization objective.

$$\begin{aligned} &\mu_e, \Sigma_e = \arg\max_{\mu,\Sigma} J(\mu, \Sigma) \\ &\text{s. t. } \mathcal{KL}(\mathcal{N}(\mu, \Sigma), \mathcal{N}(\mu_T, \Sigma_T)) \leq \delta \\ &\text{with } J(\mu, \Sigma) = \mathbb{E}_{a \sim \mathcal{N}(\mu,\Sigma)}\big[Q_{UCB}(s, a)\big] \end{aligned} \tag{4.1}$$

This can be solved using the natural gradient with respect to the natural distribution parameters, that corresponds to the vanilla gradient with respect to the expectation parameters [54]. Therefore the natural gradient step for a Gaussian policy can be expressed as

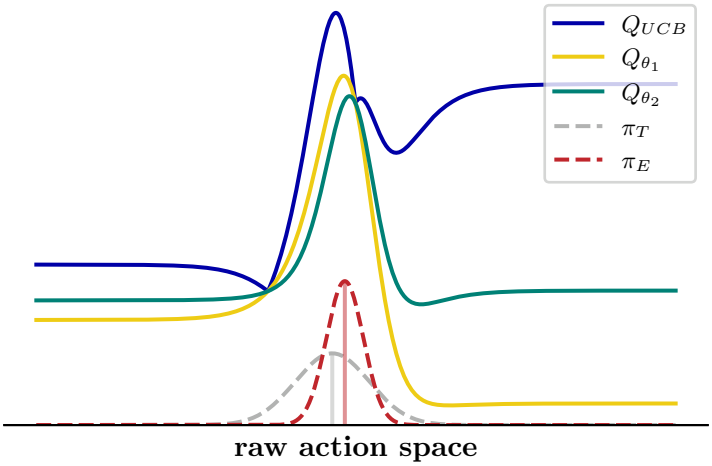


Figure 4.3.: Optimization result of NGOAC in a toy example. The value function approximations and policy distributions are shown against the raw action space.

$$\begin{aligned} \Sigma^{-1} &= \Sigma_{\pi}^{-1} - 2\beta \left[\nabla_{\Sigma} J(\mu, \Sigma)\right] \\ \mu &= \mu_{\pi} + \beta\Sigma \left[\nabla_{\mu} J(\mu, \Sigma)\right]. \end{aligned} \tag{4.2}$$

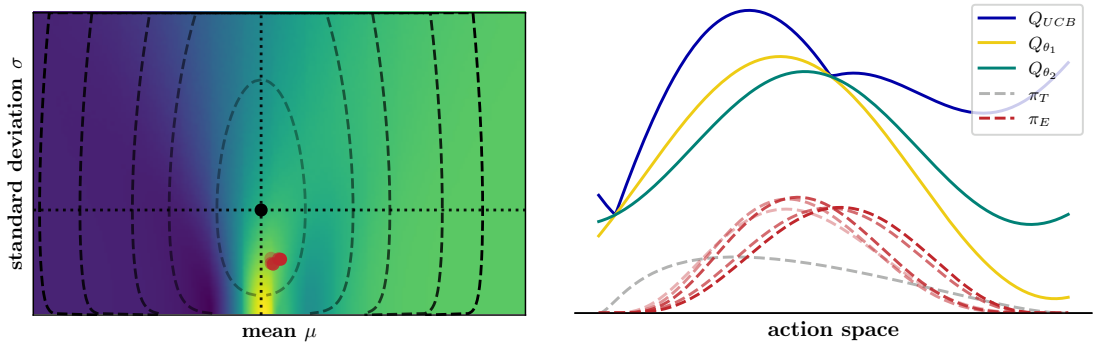
Furthermore, the the gradients for the mean and covariance can be computed using the gradient and hessian of the objective function [55]. It is noteworthy that the hessian information can be derived from the first-order information employing Stein's Lemma [56, 57].

$$\begin{aligned} \nabla_{\mu} J(\mu, \Sigma) &= \mathbb{E}\left[\nabla_a Q_{UCB}(s, a)\right] \\ \nabla_{\Sigma} J(\mu, \Sigma) &= \frac{1}{2}\mathbb{E}\left[\nabla_a^2 Q_{UCB}(s, a)\right] = \frac{1}{2}\mathbb{E}\left[\Sigma^{-1}(a-\mu)\nabla_a Q_{UCB}(s, a)^T\right] \end{aligned} \tag{4.3}$$

The update length in Equation 4.2 is determined by satisfying the KL constraint for a policy update along the direction of the natural gradient. This assurance is achieved through bracketing line search. Further implementation details can be found in Appendix A.2

Utilizing second-order gradient information to update both the policy's mean and standard deviation results in a more refined optimum compared to OAC, as illustrated in Figure 4.3, where the exploration policy is positioned close to the maximum of the UCB. Furthermore,

NGOAC maintains directional informedness and supports optimistic exploration, akin to OAC.



(a) **(b)**

Figure 4.4.: Optimization results of NGOAC in a toy example for different KL constraints. The different trust regions and the underlying objective function can be seen on the left (a). The different exploration policies (red) are shown in the raw action space on the right (b).

In theory, the optimized value of the expected UCB is not explicitly determined by the trust region bound. Consequently, NGOAC avoids exhibiting a pronounced bang-bang exploration tendency exemplified in Figure 4.4 and is, therefore, less sensitive to a good hyperparameter choice. Similar experimental results can be seen in Chapter 5.2.

5. Experiments

The subsequent experimental trials were executed utilizing NVIDIA's physics simulation environment for RL, Isaac Gym [58], that enhances the learning speed by leveraging its parallel training capabilities. The foundational implementation of the agents, with particular emphasis on the one pertaining to SAC, is drawn from the RL Games repository [59]. The experiments are launched through the utilization of the experiment launcher developed by the Intelligent Autonomous Systems (IAS) group [60]. It simplifies the process of launching experiments, enabling execution either on a local machine or on a cluster running SLURM Workload Manager [61]. In order to conduct a comparative analysis of the different exploration strategies, the evaluation is performed within the Cartpole, Ant, and Humanoid environments. These environments are available in IsaacGymEnvs, which provides tensor-based Gym environments for GPU-accelerated RL [58].

The objective of the **Cartpole** environment is to maintain the balance of a pole by exerting forces in either the left or right direction on a cart [62]. The pole is affixed to the cart through a joint that remains unactuated, while the cart itself travels along a frictionless track. Both the cart's position and the pole's angle have defined limits, and the episode terminates if these limits are surpassed. At each time step, including the terminal step, a reward of $+1$ is earned, with the cumulative reward threshold set at 500.

The **Ant** environment features a 3D robot configuration - comprising a central torso with free rotational movement and four legs, each with two body parts. It was initially introduced by Schulman et al. [63] and is part of the MuJoCo set of environments [38]. The primary objective is to effectively coordinate the movement of the four legs to propel the robot forward. This is achieved by applying torques to the eight hinges that connect the two body parts of each leg and the central torso. In total, the action space is eight-dimensional, representing each action by the torque applied to the eight hinge joints. The agent observes the positional values and velocities of different body parts, summing up to a 27-dimensional observation space. The reward consists of three parts, a fixed value representing the health of the Ant that depends on the termination of the episode, a reward

that is correlated to the forward movement of the Ant, a negative reward penalizing high actions and external contact forces.

The **Humanoid** environment features a 3D bipedal robot designed to emulate a human [64]. This robotic model comprises a torso (abdomen) along with a pair of legs and arms. Each leg consists of three body parts, while the arms are composed of two body parts, representing the knees and elbows, respectively. The primary objective is to achieve forward motion as fast as possible while maintaining balance and stability, akin to human walking, and preventing any instances of falling over. The agent exerts torques on the hinge joints, leading to an action space characterized by 17 dimensions. At each time step, the agent receives observations of the 376-dimensional state of the Humanoid. This results in a task that is notably high-dimensional and inherently complex. Much like the Ant environment, the reward of the Humanoid environment also consists of three parts. It includes a health-related reward, a reward component associated with forward movement, and a negative reward component designed to penalize large actions and external contact forces. In general this three-fold reward is designed to encourage effective locomotion while discouraging wasteful or undesirable behaviors.

As elucidated in Chapter 2.4, tasks that are non-bang-bang optimal can be justified by their task goals, as exemplified in the Cartpole environment. In this context, the pursuit of a bang-bang optimal control policy would result in a fast termination of the episode, primarily due to the rapid exceeding of either the cart's position or the pole's angle beyond their specified limits.
In stark contrast to the Cartpole environment, the primary objective in the Ant and Humanoid environments centers around achieving a maximal state reward, with the goal of attaining the highest possible forward movement speed. In these scenarios, the optimal solution is characterized by a "bang-bang" controller, as outlined by Seyde et al. [37]. This controller involves rapid and extreme control actions to optimize the forward motion of the agents in these environments. The incorporation of a minimum energy-type cost [65], which is represented by penalizing large actions in the reward structures of both environments, tends to steer the control policies toward non-bang-bang optimality. However, the degree to which this cost is integrated into the reward depends significantly on the parameter used to control the cost's importance. Typically, this parameter is fixed and governs the balance between optimizing for forward movement and minimizing energy expenditure.
Accordingly, the experiments are conducted in environments that, on the one hand, do not exhibit bang-bang optimality by default, exemplified by the Cartpole. On the other hand, the Ant and Humanoid environment are initially bang-bang optimal, but incorporate

quadratic action costs, which in turn transform the optimal control policy also into a non-bang-bang optimal one.

5.1. Performance Benchmark

The performance of NGOAC is directly compared against the performance of SAC and OAC. Furthermore, an evaluation is conducted to assess the enhancement of the natural gradient when compared to a reparameterized optimization problem as described in Chapter 5.1.1. Additionally, a bang-bang optimal exploration policy is introduced in Chapter 5.1.2 to check if the task is bang-bang optimal. If so, a bang-bang exploration strategy would lead to a fast convergence to the optimal policy.

5.1.1. Reparameterized Optimisitic Actor Critic

Besides using the natural gradient (Chapter 4), the optimization problem of OAC in Equation 3.5 can be solved using the equivalent Lagrange function.

$$\mathcal{L}(\mu, \Sigma, \lambda) = J(\mu, \Sigma) - \lambda \cdot \mathcal{KL}(\mathcal{N}(\mu, \Sigma), \mathcal{N}(\mu_T, \Sigma_T)) + \lambda \cdot \delta \tag{5.1}$$

Treading λ as a fixed hyperparameter results in a simplified optimization problem that can be optimized using SGD as described in Equation 5.2. Additional implementation details can be found in Appendix A.3.

$$\mu_E, \Sigma_E = \arg\max_{\mu,\Sigma} \mathbb{E}_{a\sim\mathcal{N}(\mu,\Sigma)}\big[Q_{UCB}(s,a)\big] - \lambda \cdot \mathcal{KL}(\mathcal{N}(\mu, \Sigma), \mathcal{N}(\mu_T, \Sigma_T)) \tag{5.2}$$

In this case, the parameter λ quantifies the extend of the KL constraint's influence, given its treatment as a soft constraint. Consequently, dissimilar to NGOAC, there exists no guarantee that the exploration policy remains strictly within the trust region. Theoretically, this could lead to a potential bang-bang exploration similar to that observed in OAC.

Conversely, augmenting the impact of the KL divergence by increasing the parameter λ adjusts the objective function in Equation 5.1 such that the optimum is shifted towards the target policy and consequently, limits the computed update of the target policy. Under

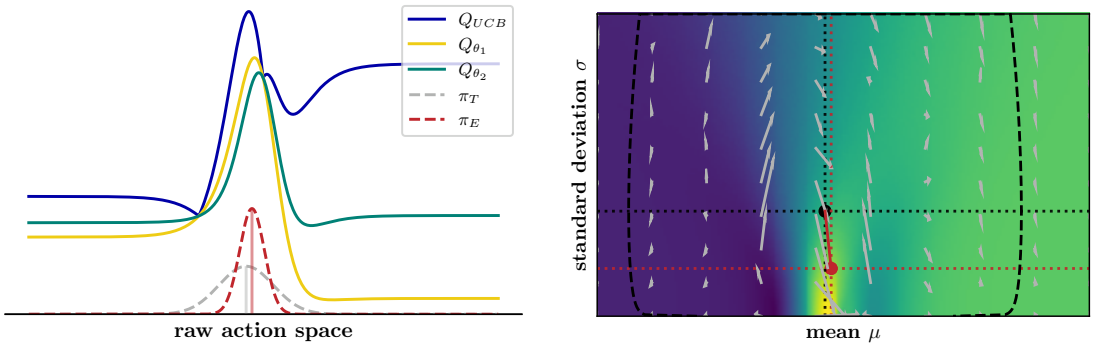


Figure 5.1.: Optimization results of OAC-RepTrick in a toy example for different KL constraints. The different exploration policies (red) are shown in the raw action space on the left (a). The different trust regions and the underlying objective function can be seen on the right (b).

extreme circumstances, the exploration strategy becomes nearly indistinguishable from the target policy and disregards the potential benefits of optimistic exploration.

The experimental results in Chapter 5.2 show that the bang-bang tendency is very unlikely as the objective function is well formed inside the action space. Therefore a lower value of λ can be chosen despite the theoretical fact, that it can produce a bang-bang like exploration strategy. In Figure 5.1 the optimization result of the reparameterized objective function is exemplified using a toy example. Similar to the natural gradient version of OAC in Figure 4.4, the exploration policy converges to the maximum of the UCB, is directional informed, avoids pessimistic underexploration and does not tend to explore the state-action space with a bang-bang exploration strategy.

The reparameterized version of OAC challenges the necessity of the natural gradient and forms a computationally less expensive alternative. Furthermore, a bang-bang exploration strategy is used to investigate the advantage of the uncertainty-based exploration strategy.

5.1.2. Bang-Bang exploration

In general a bang-bang exploration should benefit the learning process in tasks, where the optimal policy itself is a bang-bang policy, as described in Chapter 2.4. Therefore, SAC is

modified to explore the state-action space with actions at the limit of the action space to check if uncertainty based exploration methods like OAC and its natural gradient version outperform a simple bang-bang exploration strategy.

The exploration actions are determined by choosing the closest action boundary to a sampled action from the target policy, as outlined in Equation 5.3.

$$\begin{aligned} a_E = &\min_{a \in \{a_{min}, a_{max}\}} (a - a_\pi)^2 \\ \text{with } a_\pi &\sim \pi_\phi(\cdot\, s_{t+1}) \end{aligned} \tag{5.3}$$

As the alterations are limited to the exploration behavior and, consequently, the actions stored in the replay buffer, the updates for the critics and the target policy remain unchanged.

5.1.3. Results

The experiments are carried out with a total of 25 different random seeds for each algorithm. The selection of hyperparameters, particularly for OAC, NGOAC, and the reparameterized version of OAC, is determined through an extensive grid search, detailed in Chapter 5.2 and in Appendix C. A comprehensive overview of the entire experimental setup is available in Appendix B.

In Figure 5.2, the non-bang-bang optimality of the Cartpole task is substantiated by the performance of the SAC-Bang-Bang exploration strategy. By restricting exploration actions to the maximum and minimum action values, the agent struggles to learn an effective policy, despite its capability to represent a non-bang-bang policy. This outcome underscores the distinct nature of the Cartpole task, where a non-bang-bang policy is more suitable for achieving successful learning outcomes.
With a well-chosen set of exploration hyperparameters, OAC effectively solves the task and performs comparably to the baseline performance of SAC. In this context, the utilization of the UCB does not exert a substantial influence on the results. This observation is underpinned by the selection of a very small KL bound, which restricts the impact of the optimistic exploration strategy, leading to a very similar exploration strategy to SAC. This is evidenced by the analysis of the influence of the KL bound in Figure 5.6a, where higher values of δ induce a bang-bang exploration strategy. However, such a strategy is suboptimal in the Cartpole environment, as demonstrated by the performance of SAC-Bang-Bang.

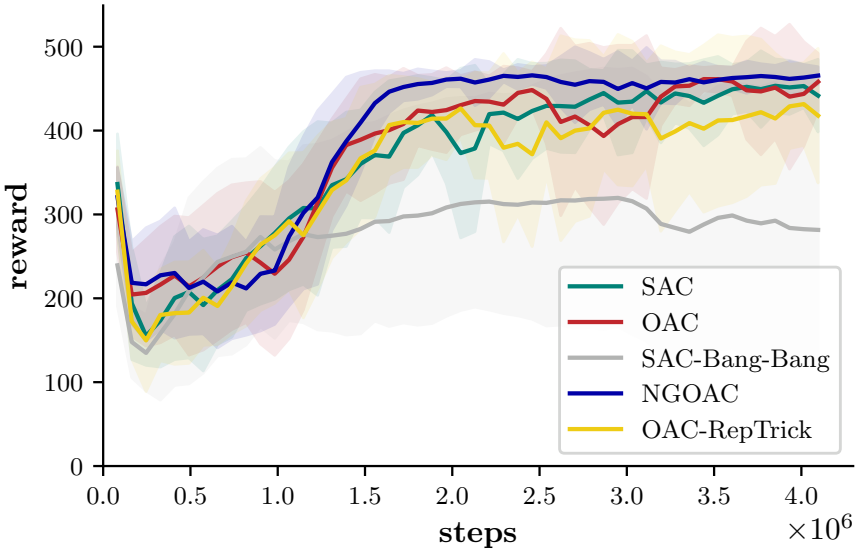


Figure 5.2.: Performance benchmark in the Cartpole environment.

The reparameterized variant of OAC, OAC-RepTrick, exhibits performance on par with OAC and SAC. However, it is worth noting that OAC-RepTrick converges to a slightly smaller mean reward while displaying an increased variance in the rewards when compared to OAC and SAC.
In the Cartpole environment, NGOAC emerges as the top-performing algorithm, with the converged mean reward surpassing that of OAC and SAC only by a slight margin. Notably, the natural gradient version is found to be less sensitive to the fine-tuning of hyperparameters, as further detailed in Chapter 5.2. Consequently, a higher value for the KL bound is selected, allowing the agent to fully leverage the uncertainty-based exploration strategy and engage in more optimistic exploration. This allows NGOAC to achieve significantly faster convergence compared to the other exploration methods, reaching the optimal policy after approximately 1.7 million steps. In contrast, both OAC and SAC require around 3 million steps to converge. Moreover, NGOAC exhibits the smallest variance in the reward, highlighting the robustness of its exploration strategy against the inherent stochasticity of the task.

The results of the Ant environment are shown in Figure 5.3. With its bang-bang optimal main goal of moving forward, a bang-bang exploration strategy, epxressed as SAC-Bang-Bang, leads to mediocre performance, even though it performs not as good as SAC.

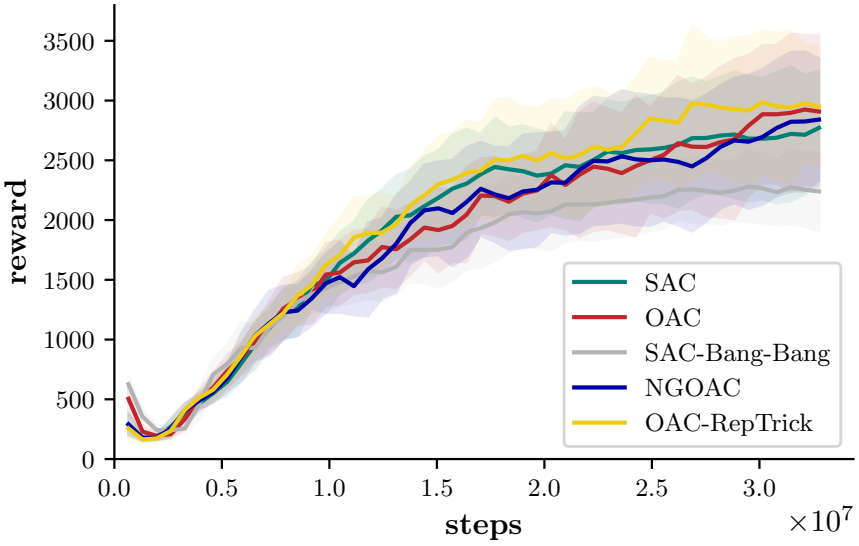


Figure 5.3.: Performance benchmark in the Ant environment.

The different exploration strategies employed by OAC, NGOAC, and OAC-RepTrick exhibit very similar performance to SAC. In this context, it is challenging to distinguish a clear difference in their performance, as both their mean rewards and the variance of the return are nearly identical.

Similar outcomes are observed in the Humanoid environment, as depicted in Figure 5.4. The bang-bang exploration strategy of SAC-Bang-Bang demonstrates noteworthy performance, aligning with the task's primary objective of the 3D bipedal robot, which is to achieve forward motion in a minimal amount of time. However, when compared to SAC, it converges to a lower mean reward. This disparity can be attributed to the limitation of the bang-bang actions in providing sufficient information regarding the tripartite structure of the reward, which is essential for adapting the optimal policy in this in fact non-bang-bang optimal task. The performance of the natural gradient and the reparameterized version cannot be distinguished from that of SAC, as all three approaches yield similar results. Notably, OAC stands out by outperforming the other exploration strategies. In contrast to the Ant environment, the best results of OAC in the Humanoid environment are achieved by utilizing a higher KL bound, as investigated in Chapter 5.2. While this can introduce a tendency toward bang-bang exploration, it proves to be beneficial in the Humanoid environment. In general, all the results obtained from the various exploration strategies

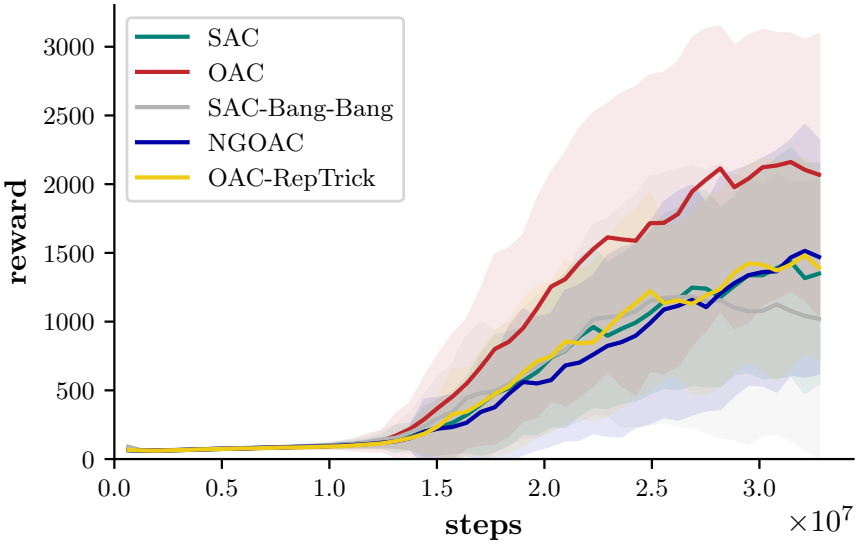


Figure 5.4.: Performance benchmark in the Humanoid environment.

exhibit high variance, underscoring the complexity of the tasks at hand.

5.2. Hyperparameter Search

The influence of the exploration hyperparameters for OAC and NGOAC are investigated in all three environments using a hyperparameter search. To obtain quantitative results a

Hyperparameter		Search space
KL bound	δ	$[.1, 1, 10, 100, 200, 1000]$
Level of optimism	β_{UB}	$[0, 1, 2, 4, 8, 16, 32, 64]$

Table 5.1.: Grid search of the exploration hyperparameter for OAC and NGOAC.

grid search along the KL constraint δ and the level of optimism β_{UB} is performed according to Table 5.1. Each search pair is run for five seeds and the mean reward of the last five evaluation episodes is reported. The full experimental setup can be found in Appendix B.

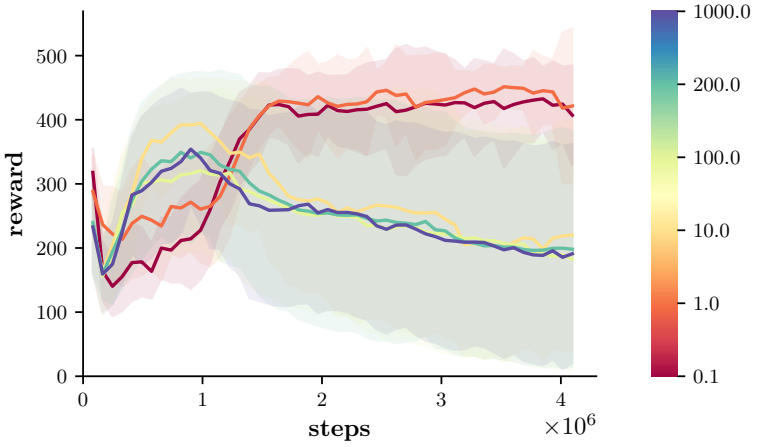


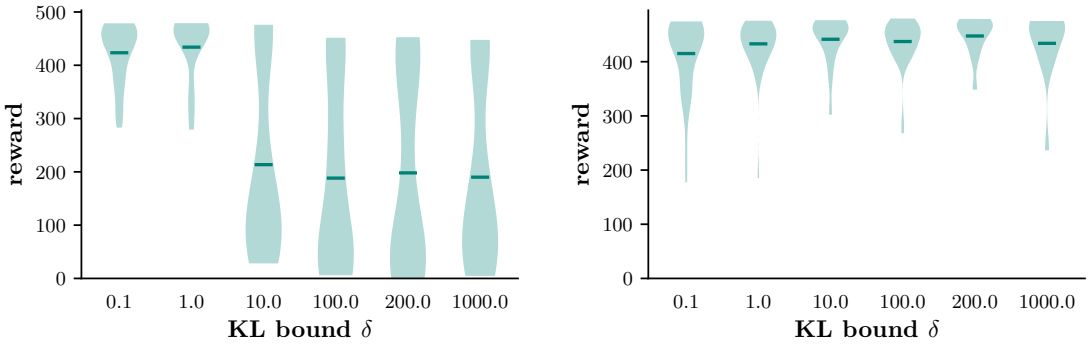
Figure 5.5.: Reward of OAC in the Cartpole environment for different values of the KL bound δ

5.2.1. Kullback-Leibler Divergence Bound

Cartpole

In OAC the KL bound directly influences the exploration strategy and consequently the outcome of the training process. For the low-dimensional Cartpole environment the reward in Figure 5.6 clearly indicates that a higher value of the constraint not only results in a smaller mean reward but also increases the variance of the return.

Further insights in Figure 5.5 show that a higher KL bound seems to lead to fast exploration in the first episodes during learning. But after about one million steps the learned policy converge to a suboptimal solution and fails to learn a good policy on the long term. Only the exploration policies that are optimized enforcing a smaller trust region succeed the Cartpole task.



(a) OAC **(b)** NGOAC

Figure 5.6.: Comparison of the reward of OAC (left) and NGOAC (right) in the Cartpole environment for different values of the KL bound δ. The reward is averaged over the last five evaluation episodes.

This is contrasted with the result of NGOAC in Figure 5.6b. Optimizing the optimistic objective function using the natural gradient leads to good results for all values of the trust region constraint. Indeed, a small KL constraint can limit the update length such that the state-action space is explored using a similar distribution as the target distribution. This can be the case for $\delta = 0.1$ as NGOAC results in the worst mean reward. Actually, this does not seem to influence the training process as shown in Fig 5.7. It can bee seen,

that the training curve for all hyperparameter values seem to converge to the same policy with an identical learning speed.

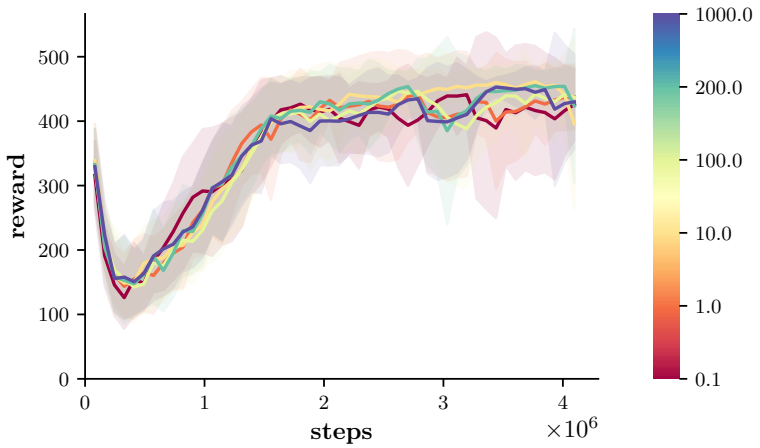
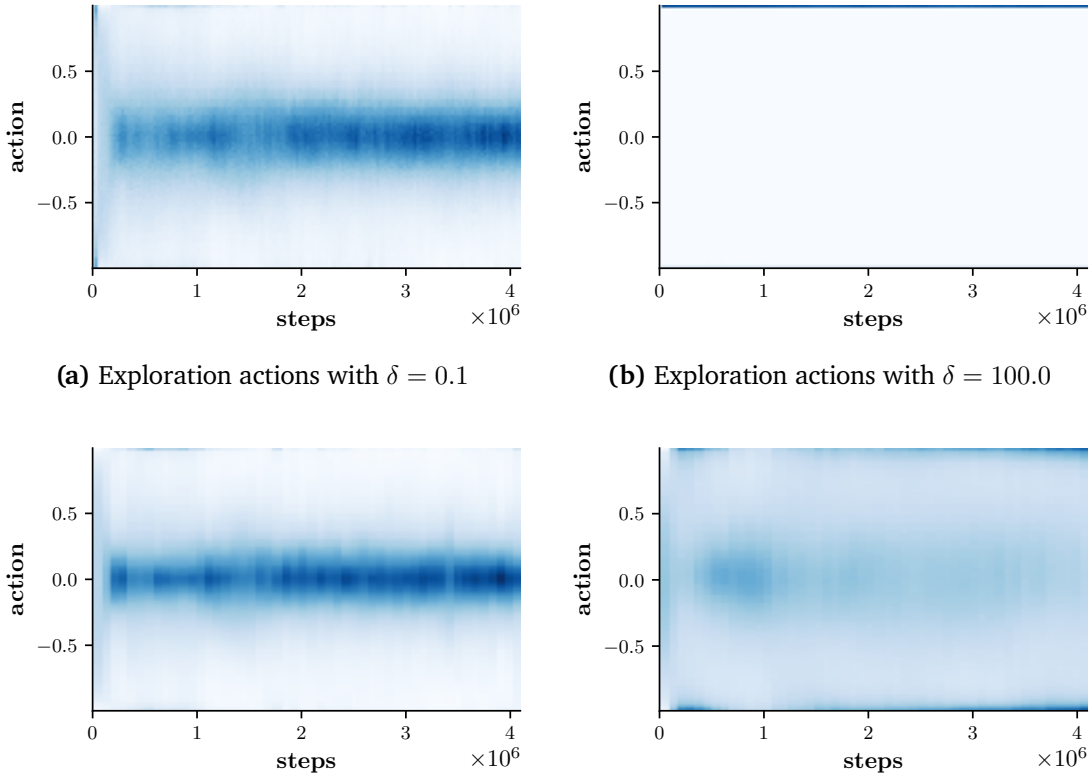


Figure 5.7.: Reward of NGOAC in the Cartpole environment for different values of the KL bound δ

As described in Chapter 4.1, OAC results in a bang-bang-like exploration strategy for sufficiently high values of the KL bound. This can be seen in Fig 5.8a as the actions used for exploration clearly embody a bang-bang exploration strategy. Accordingly the learned policy tends to use actions near the boundary to solve the task (see Figure 5.8c). The Cartpole can be categorized as a non-bang-bang optimal control task, given that the optimal control policy embodies actions of near-zero magnitude to achieve pole stabilization. Furthermore, it is crucial to emphasize that the pole's angular orientation is constrained, and exceeding these limits results in the termination of the interaction and, consequently, in a low reward. A lower KL bound typically does not result in the development of a bang-bang exploration strategy, as theoretically depicted in Chapter 4.1. This facilitates the agent's ability to acquire a meaningful policy that leads to succeed the task. In the context of Figure 5.8c, the learned policy generates actions of near-zero magnitude, ultimately leading toward a higher reward, as evidenced in Figure 5.5.

NGOAC exhibits greater robustness across a broader range of KL constraint values, primarily due to the fact that the exploration policy is not directly determined by the optimization bound, as elucidated earlier and depicted in Figure 5.6b. With a higher KL bound, NGOAC gains the capability to select actions within a more extensive region surrounding the current learned policy, but is not confined to the trust region bound as its exploration policy is



(a) Exploration actions with $\delta = 0.1$

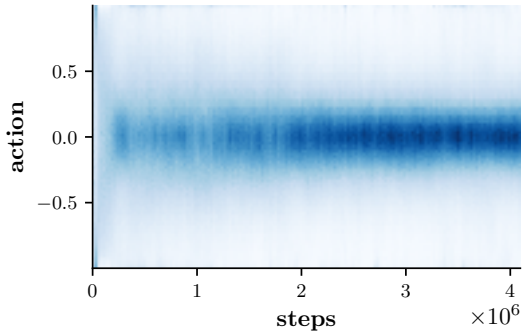
(b) Exploration actions with $\delta = 100.0$

(c) Evaluation actions with $\delta = 0.1$

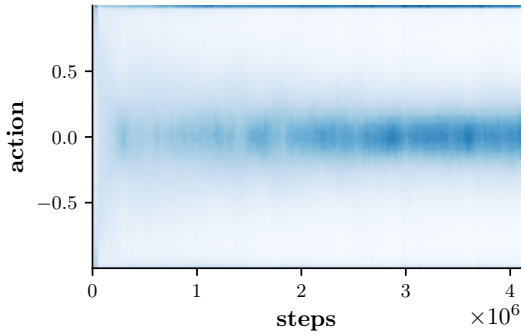
(d) Evaluation actions with $\delta = 100.0$

Figure 5.8.: Action histogram of OAC in the Cartpole environment with different values for δ. The top row shows the actions during training, and the figures at the bottom correspond to the actions during evaluation.

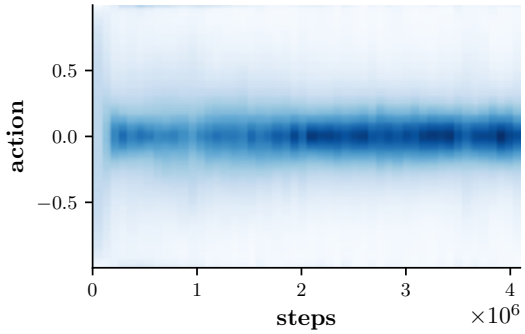
determined by the information provided by the UCB. Even with a higher KL bound NGOAC does not lead to the adoption of a bang-bang exploration strategy, as evidenced in Figure 5.9. A higher optimization constraint incorporates sufficient information into the training process, allowing the agent to accomplish the task effectively. In this case, the learned policy exhibits a similar action distribution as the one associated with a lower optimization constraint, both of which converge towards an optimal policy. This convergence is also reflected in the reward patterns, as depicted in Figure 5.7.



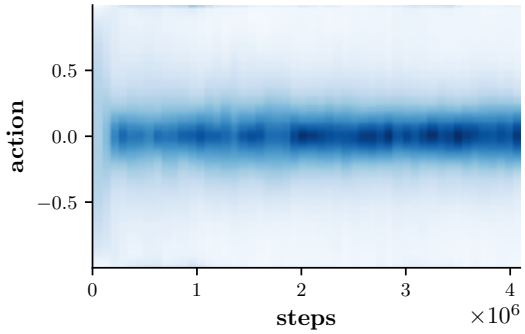
(a) Exploration actions with $\delta = 0.1$



(b) Exploration actions with $\delta = 100.0$



(c) Evaluation actions with $\delta = 0.1$



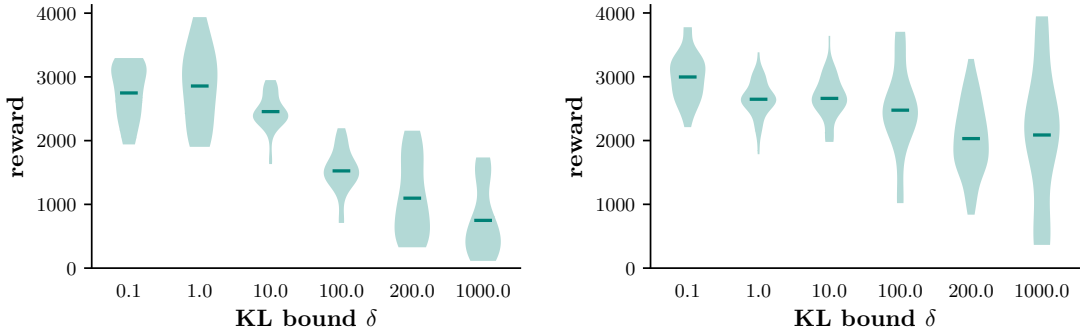
(d) Evaluation actions with $\delta = 100.0$

Figure 5.9.: Action histogram of NGOAC in the Cartpole environment with different values for δ. The top row shows the actions during training, and the figures at the bottom correspond to the actions during evaluation.

Because of the stabilization task and the early termination when the pole angle exceeds the limit, the Cartpole environment is a good example for an inherently non-bang-bang optimal task. This is also satisfied by the evaluation of the action histograms during training and evaluation as seen above, where an agent using a bang-bang policy fails to converge to a good solution.

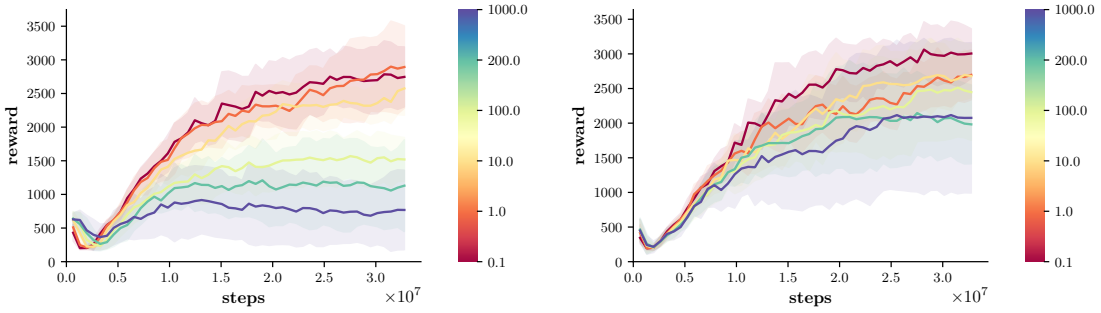
Ant

In the Ant environment, it can be observed that OAC performs comparable to the Cartpole environment. This similarity arises due to a strong correlation between the exploration policy and the KL bound. Consequently, a reduced optimization constraint yields good results, whereas a higher constraint value corresponds to decreased rewards, as depicted in Figure 5.10a and Figure 5.11a.



(a) OAC **(b)** NGOAC

Figure 5.10.: Comparison of the reward of OAC (left) and NGOAC (right) in the Ant environment for different values of the KL bound δ. The reward is averaged over the last five evaluation episodes.



(a) OAC **(b)** NGOAC

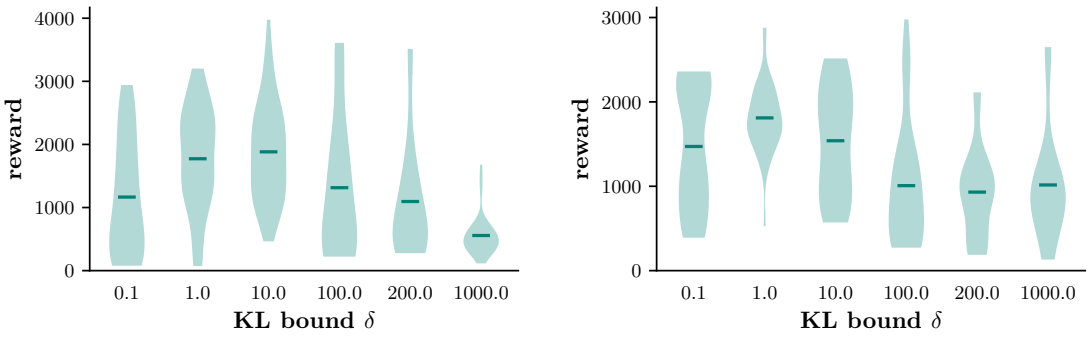
Figure 5.11.: Reward of OAC (a) and NGOAC (b) in the Ant environment for different values of the KL bound δ

The similarity between the results in the Ant and Cartpole environment extends to the behavior of NGOAC. When compared to OAC, it exhibits greater resilience across a broader range of optimization bounds. Although the mean reward remains within a moderately rewarding range for higher hyperparameter values, there is a slight inclination toward lower rewards. Furthermore, it is evident that an increased constraint bound results in a higher reward variance, as exemplified in Figure 5.11b.

Humanoid

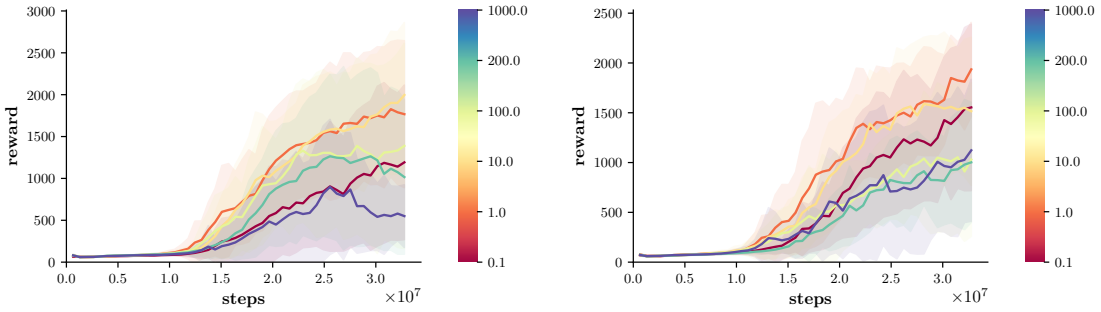
In the high-dimensional Humanoid environment the results for OAC are different to the ones of the previous two evaluated environments. It can be clearly seen in Figure 5.10a and in Figure 5.11a that for very low and very high values of the KL bound the agent fails to learn a good policy. A low constraint bound limits the incorporation of optimism in the face of uncertainty because the exploration policy behaves nearly the same as the learned target policy. This phenomena is also indicated in the lower dimensional Ant environment as shown in Figure 5.10a.

The significance of making an appropriate choice of the optimization constraint is evident even in the case of the NGOAC, as illustrated in Figure 5.12b. Notably, for higher KL values, the agent does not exhibit as pronounced bad behavior as observed with the OAC.



(a) OAC **(b)** NGOAC

Figure 5.12.: Comparison of the reward of OAC (left) and NGOAC (right) in the Humanoid environment for different values of the KL bound δ. The reward is averaged over the last five evaluation episodes.



(a) OAC **(b)** NGOAC

Figure 5.13.: Reward of OAC (a) and NGOAC (b) in the Humanoid environment for different values of the KL bound δ.

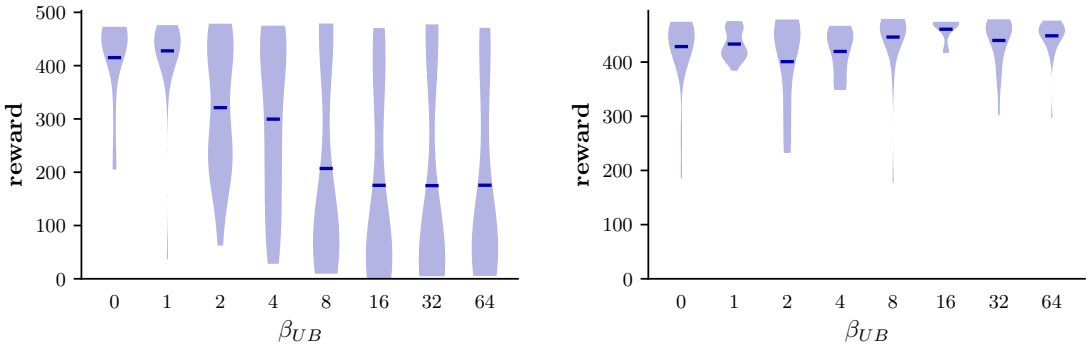
In comparison to both the Cartpole and Ant environments, it is evident that the variance in the rewards is significantly more significant. This phenomenon is visually depicted in Figure 5.13 and can be attributed to the higher-dimensional state-action space inherent in the more challenging tasks of the Humanoid. This increased dimensionality leads to a less precisely defined state-action value function.

5.2.2. Level of Optimism

The degree of optimism is introduced through the hyperparameter β_{UB} and directly impacts the UCB. This influence on the UCB is essential for optimizing the exploration policy, as detailed in Equation 3.5 and Equation 4.1. Given that the UCB is rooted in the Optimisim in the Face of Uncertainty principle, a higher value for β_{UB} empowers the agent to explore regions within the state-action space where the variance of both critics is elevated. This exploration occurs even if the means of these regions are lower than in other areas.

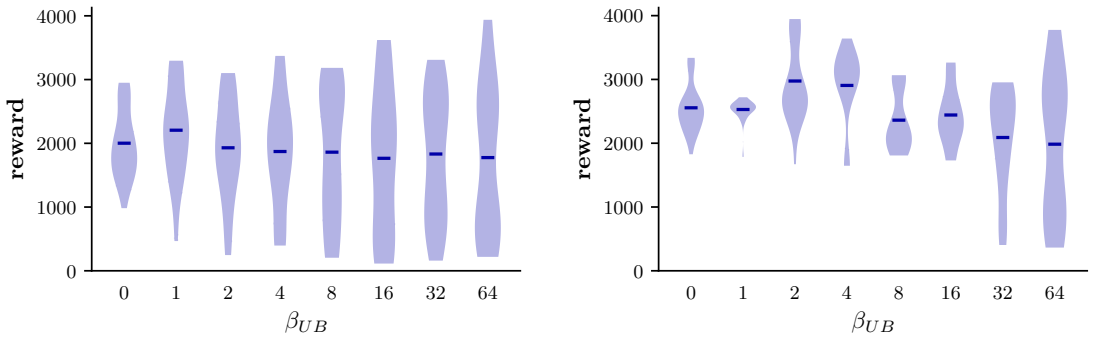
In the Cartpole environment, the grid search conducted along the optimism level closely resembles the outcomes associated with the results of the KL bound, as previously discussed in Chapter 5.2.1. This similarity is visually depicted in Figure 5.14 when focusing solely on the mean values.

Nevertheless, it is important to note that the results, particularly for OAC, exhibit a substantial variance in rewards, encompassing both the maximum and minimum reward values of the environment. In general the grid search encompasses all combinations of hyperparameters, including the KL bound δ and the level of optimism β_{UB}. Consequently, the rewards associated with each set of hyperparameters are averaged over the other hyperparameter. As a result, it becomes challenging to definitively assert whether the level of optimism has a direct and causative impact on the agent's success, as its influence



(a) OAC **(b)** NGOAC

Figure 5.14.: Comparison of the reward of OAC (a) and NGOAC (b) in the Cartpole environment for different values of the level of optimism β_{UB}. The reward is averaged over the last five evaluation episodes.



(a) OAC **(b)** NGOAC

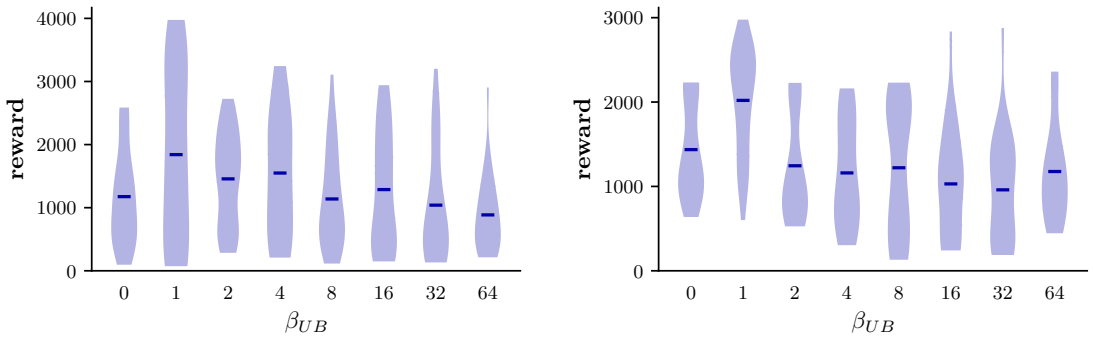
Figure 5.15.: Comparison of the reward of OAC (a) and NGOAC (b) in the Ant environment for different values of the level of optimism β_{UB}. The reward is averaged over the last five evaluation episodes.

is intertwined with the effects of the other hyperparameter based on the performed grid search.

Indeed, the action histogram, which aggregates the sum of actions across all KL bound values for each β_{UB}, provides no additional insights into a possible dependency between the level of optimism and the reward. For a comprehensive examination, these results are available in the Appendix C.1, along with the reward for each hyperparameter value.

In the case of NGOAC, it appears that the level of optimism may not have a clear and direct influence on the agent's success. However, due to the substantial variance observed in the results, as depicted in Figure 5.14b, it becomes challenging to make precise qualitative statements regarding this relationship, at least within the range covered by the performed grid search. The high variance in outcomes underscores the complexity of assessing the impact of the level of optimism on the agent's performance in this context.

In the high-dimensional tasks of the Ant environment (Figure 5.15) and the Humanoid environment (Figure 5.16), the variance in rewards doesn't provide a basis for drawing any qualitative conclusions regarding the hyperparameter sensitivity of OAC and NGOAC. Furthermore, when considering the stochasticity of the experiments, the mean rewards for the various values of β_{UB} fall within a similar range.



(a) OAC **(b)** NGOAC

Figure 5.16.: Comparison of the reward of OAC (a) and NGOAC (b) in the Humanoid environment for different values of the level of optimism β_{UB}. The reward is averaged over the last five evaluation episodes.

5.2.3. Results

Table 5.2 presents the best hyperparameter pairs of the grid search for OAC, considering both the KL bound and the level of optimism. As previously discussed, it's evident that across all environments, the majority of the best-performing hyperparameter pairs involve a lower value for the optimization constraint δ. This observation highlights a consistent trend where lower values of δ tend to be associated with improved performance in various environments. Additionally, it's worth noting that among the best-performing hyperparameter pairs, the top five encompass a diverse range of values for β_{UB}. This observation suggests that the level of optimism, as represented by β_{UB}, does not exert a significant and deterministic influence on the agent's performance. Instead, it appears that other hyperparameters, such as the optimization constraint δ, play a more prominent role in determining the agent's success.

The best hyperparameter pairs for NGOAC (Table 5.3) reaffirm the observations discussed earlier. These hyperparameter pairs encompass a wide spectrum of KL bound values within each environment. This characteristic aligns with the notion that NGOAC, as a less hyperparameter sensitive optimization method, is less inclined to generate a bang-bang exploration policy. The inclusion of a wide range of KL bound values in these best hyperparameter pairs underscores the robustness of NGOAC in avoiding extreme exploration policies. Much like in the case of OAC, the natural gradient variant, NGOAC, results in a relatively low sensitivity to the selection of the level of optimism. This is

Cartpole			Ant			Humanoid		
β_{UB}	δ	reward	β_{UB}	δ	reward	β_{UB}	δ	reward
8	1	472.46	16	1	3482.42	1	100	3237.31
0	10	470.12	64	1	3355.51	4	10	2940.09
1	10	467.39	4	1	3101.25	1	200	2846.45
16	1	467.26	1	0.1	3075.84	8	1	2472.41
32	0.1	457.25	32	0.1	3057.30	4	1	2283.93

Table 5.2.: Best hyperparameter pairs of OAC

substantiated by the inclusion of β_{UB} values that span the full range, from the minimum to the maximum, within the best hyperparameter pairs. This comprehensive coverage of β_{UB} values among the optimal pairs underscores the limited impact of the level of optimism on the agent's performance, as described in Chapter 5.2.2.

For the sake of completeness, the comprehensive results of the grid search are available in their entirety in Appendix C.3.

Furthermore, the grid search has also been conducted for the reparameterized extension of OAC, as detailed in Appendix C.2. Notably, the performance of this reparameterized version

Cartpole			Ant			Humanoid		
β_{UB}	δ	reward	β_{UB}	δ	reward	β_{UB}	δ	reward
64	10	473.79	64	0.1	3665.76	1	100	2543.68
16	200	470.91	2	1000	3567.46	1	1000	2524.69
16	100	470.12	2	100	3430.24	1	10	2507.57
16	1000	467.88	0	0.1	3278.47	64	0.1	2162.46
2	200	467.05	4	100	3258.78	2	0.1	2149.48

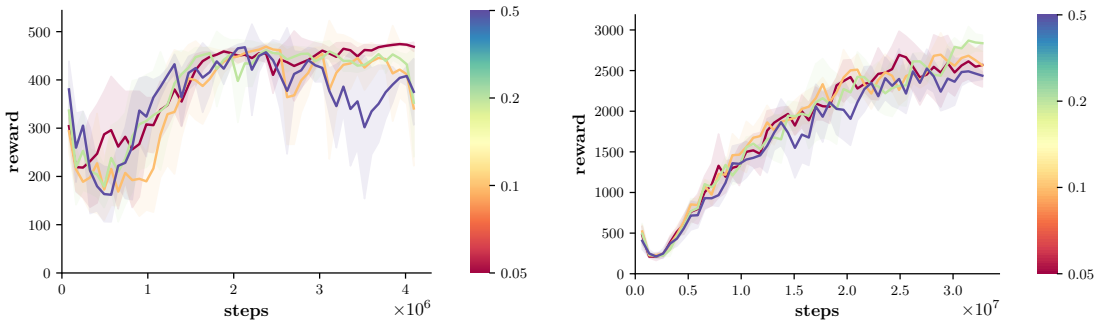
Table 5.3.: Best hyperparameter pairs of NGOAC

closely resembles that of NGOAC, as no discernible correlation between the rewards and hyperparameter values is apparent across all grid searches. Consequently, it can be inferred that the exploration policy remains largely unaffected by the choice of the hyperparameters λ and β_{UB}. Hence, it is noteworthy that OAC RepTrick exhibits a propensity to avoid generating a bang-bang exploration policy in the non-bang-bang optimal Cartpole tasks. This is justified in the evaluation of λ in Figure C.6. In general, λ defines the loss associated with the KL divergence that is incorporated into the optimization objective outlined in Equation 5.1. For both extreme values of the grid search over λ the agent converges to an optimal policy, producing actions characterized by a low magnitude and effectively addresses the task of stabilizing the cart pole.

5.3. Prioritized Experience Replay

The introduced exploration methods focus on optimizing an optimistic objective function to derive an exploration policy. Consequently, the actions used to explore the state-action space are determined based on the current approximation of the state-value function. In theory, it can be justifiable to directly incorporate these sampled actions into the off-policy update of the agent, as they provide valuable exploration information based on the current critic approximation. Prior research has explored similar ideas, showing that incorporating on-policy data into the update of an off-policy agent using experience replay can lead to improved performance, as demonstrated by Schmitt et al. [66]. To achieve this, an informed sampling method is employed to select actions from the replay buffer. This prioritized replay buffer is designed to facilitate the agent in effectively leveraging the informed exploration strategy, ultimately resulting in faster convergence during the learning process.

The sampled actions used for the agent update are categorized into two parts. Online actions are drawn from a smaller replay buffer of the size defined by the hyperparameter $onlineSize$ containing actions from the most recent steps in the environment, while offline actions are sampled from the entire replay buffer.



(a) Cartpole **(b)** Ant

Figure 5.17.: Reward of OAC in the Cartpole (a) and Ant (b) environment for different prioritized replay buffers.

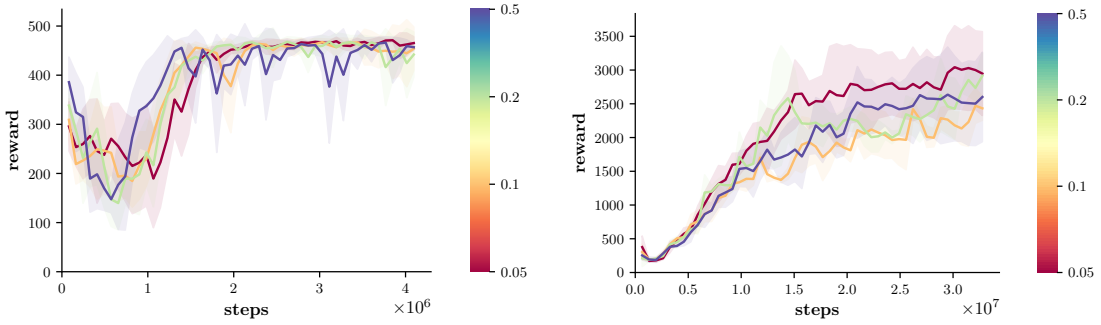
$$
\begin{aligned}
a_B &= a_{B,offline} + a_{B,online} \\
a_{B,online} &\sim replayBuffer[idx - onlineSize : idx] \\
a_{B,offline} &\sim replayBuffer[0 : max]
\end{aligned}
\tag{5.4}
$$

The number of actions drawn from both replay buffers is determined by the batch size and a factor α, which specifies the proportion between offline and online samples.

$$
\begin{aligned}
onlineBatchSize &= \alpha \cdot batchSize \\
offlineBatchSize &= batchSize - onlineBatchSize
\end{aligned}
\tag{5.5}
$$

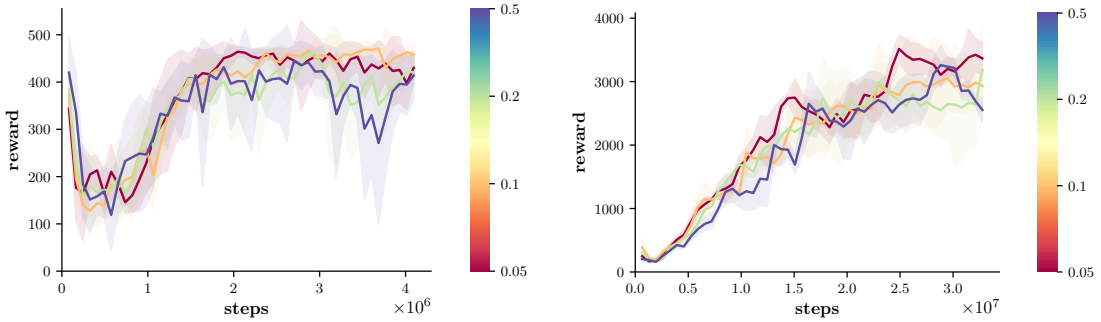
The subsequent experiments are conducted for OAC, NGOAC, and OAC-RepTrick with different online-offline proportions, each across five seeds. For more comprehensive experimental details, please refer to the information provided in Appendix D.

Employing a prioritized replay buffer for OAC to prioritize the latest actions in the agent's update process does not yield any additional improvement. As shown in Figure 5.17a for the Cartpole environment, a lower value of the online-offline fraction α results in a more stable outcome after the policy has converged to the optimal policy. Conversely, a higher value of α leads to slightly inferior results. In the Ant environment, as depicted in Figure 5.17b, there is no noticeable distinction in the performance between different values of α.



(a) Cartpole **(b)** Ant

Figure 5.18.: Reward of NGOAC in the Cartpole (a) and Ant (b) environment for different prioritized replay buffers.



(a) Cartpole **(b)** Ant

Figure 5.19.: Reward of OAC-RepTrick in the Cartpole (a) and Ant (b) environment for different prioritized replay buffers.

This observation also holds for the Humanoid environment, where the detailed results are available in Appendix D.

Like OAC, the incorporation of online data into the offline update of SAC does not result in any performance improvement for both the natural gradient extension and the reparameterized version. This lack of improvement is evident in Figure 5.18 for NGOAC and in Figure 5.19 for OAC-RepTrick. Further results pertaining to the Humanoid environment can be found in Appendix D.

The results are quite evident: a prioritized replay buffer that emphasizes the latest sampled actions of the exploration policy does not accelerate the convergence speed and does not surpass the base version, as demonstrated in Chapter 5.1. It is noteworthy that for OAC in the Cartpole environment, emphasizing recent actions can lead to post-convergence instabilities.

6. Conclusion

OAC relies on an uncertainty-based exploration strategy to enhance both convergence speed and agent performance. However, a significant challenge arises due to its sensitivity to the KL bound, which can lead to the generation of a bang-bang exploration policy. This issue is theoretically demonstrated in a simple example and further investigated in the Cartpole task by evaluating the action histogram.
Experimental results emphasize the sensitivity to the hyperparameter, which can make OAC challenging to apply in other environments without a comprehensive hyperparameter search. In general, non-bang-bang optimal tasks are characterized by either a task goal that necessitates a non-bang-bang policy or the inclusion of a quadratic action cost term in the reward structure. In environments that are non-bang-bang optimal by default, such as the Cartpole, OAC may encounter difficulties. Nevertheless, it excels in inherently bang-bang optimal tasks, like Ant and Humanoid.

To address these challenges, the natural gradient variant of OAC is introduced. NGOAC optimizes the UCB with a natural gradient while relying solely on first-order gradient information, maintaining computational efficiency comparable to OAC. Theoretical strengths of NGOAC are demonstrated through a toy example where the optimum is better approximated and the exploration policy is not directly determined by the KL constraint. As a result, a higher value of the optimization constraint enables a more flexible optimization of the exploration policy, allowing it to better utilize the optimistic exploration information. Building upon the foundation of OAC, the natural gradient extension also enables directional informed exploration, mitigates pessimistic underexploration, and, importantly, does not inherently tend to produce a bang-bang exploration strategy.
Experimental results showcase that NGOAC is in general less sensitive to the hyperparameters, especially to the KL bound of the optimization problem, leading to good results across a wide range of hyperparameter values. Notably, in the Cartpole environment, NGOAC outperforms SAC and OAC in terms of exploration speed, stability, and robustness.

To assess the need for the natural gradient in optimizing the UCB, an alternative approach involves optimizing a reparameterized objective function using SGD. In theory, OAC-RepTrick provides a superior optimization solution compared to OAC as it also incorporates gradient information into the update of the policy's variance. Much like the natural gradient, it avoids generating a bang-bang-like exploration policy. However, it differs in that the attained optimum is not constrained to remain within the trust region defined by the KL bound. Furthermore, OAC-RepTrick introduces an additional hyperparameter that governs the extent of trust region violation. The empirical findings support the notion that optimizing the reparameterized objective function is less susceptible to variations in hyperparameters. Nevertheless, in the Cartpole environment, where NGOAC demonstrates superior performance in terms of convergence speed and stability, OAC-RepTrick exhibits performance levels similar to OAC and the baseline results of SAC.

The potential advantages of a prioritized replay buffer in incorporating a greater proportion of online data into the off-policy update procedure of SAC have been explored. The experimental results unequivocally demonstrate that the implementation of a prioritized replay buffer does not yield any benefits; it neither accelerates the convergence speed nor enhances the performance of OAC and its variants. In fact, it can potentially introduce instabilities into the learning process.

In future research, there is an opportunity to delve deeper into the approximation of the UCB. One avenue to explore involves employing ensemble methods to obtain a more accurate estimation of the epistemic uncertainty associated with the state-value function. Additionally, the work by [51] underscores the significance of ensemble independence when estimating uncertainties for Offline RL. In the context of SAC, this can be achieved by employing an ensemble of independent min-double Q function approximators. Since uncertainty estimation significantly impacts the quality of the UCB, these efforts have the potential to yield further advancements in optimistic exploration. Moreover, with a precisely defined objective function, multiple optimization iterations can be executed to obtain a better optimum, consequently yielding an enhanced exploration strategy.

Bibliography

[1] M. L. Puterman, "Markov decision processes," *Handbooks in operations research and management science*, vol. 2, pp. 331–434, 1990.

[2] R. Bellman, "A markovian decision process," *Journal of mathematics and mechanics*, pp. 679–684, 1957.

[3] R. Sutton and A. Barto, "Reinforcement learning: An intro," 1998.

[4] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[5] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, pp. 1928–1937, PMLR, 2016.

[6] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, pp. 1861–1870, PMLR, 2018.

[7] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[8] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, pp. 279–292, 1992.

[9] S. Levine and V. Koltun, "Guided policy search," in *International conference on machine learning*, pp. 1–9, PMLR, 2013.

[10] S. Levine, "Reinforcement learning and control as probabilistic inference: Tutorial and review," *arXiv preprint arXiv:1805.00909*, 2018.

[11] A. Abdolmaleki, R. Lioutikov, J. R. Peters, N. Lau, L. Pualo Reis, and G. Neumann, "Model-based relative entropy stochastic search," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

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

[13] M. Riedmiller, "Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method," in *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16*, pp. 317–328, Springer, 2005.

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

[15] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[16] A. W. Moore and C. G. Atkeson, "Prioritized sweeping: Reinforcement learning with less data and less time," *Machine learning*, vol. 13, pp. 103–130, 1993.

[17] Y. Hou, L. Liu, Q. Wei, X. Xu, and C. Chen, "A novel ddpg method with prioritized experience replay," in *2017 IEEE international conference on systems, man, and cybernetics (SMC)*, pp. 316–321, IEEE, 2017.

[18] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*, pp. 1995–2003, PMLR, 2016.

[19] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu, "Reinforcement learning with unsupervised auxiliary tasks," *arXiv preprint arXiv:1611.05397*, 2016.

[20] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, *et al.*, "Deep q-learning from demonstrations," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[21] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[22] D. Horgan, J. Quan, D. Budden, G. Barth-Maron, M. Hessel, H. Van Hasselt, and D. Silver, "Distributed prioritized experience replay," *arXiv preprint arXiv:1803.00933*, 2018.

[23] A. Nair, P. Srinivasan, S. Blackwell, C. Alcicek, R. Fearon, A. De Maria, V. Panneershelvam, M. Suleyman, C. Beattie, S. Petersen, *et al.*, "Massively parallel methods for deep reinforcement learning," *arXiv preprint arXiv:1507.04296*, 2015.

[24] J. Hao, T. Yang, H. Tang, C. Bai, J. Liu, Z. Meng, P. Liu, and Z. Wang, "Exploration in deep reinforcement learning: From single-agent to multiagent domain," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[25] B. C. Stadie, S. Levine, and P. Abbeel, "Incentivizing exploration in reinforcement learning with deep predictive models," *arXiv preprint arXiv:1507.00814*, 2015.

[26] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International conference on machine learning*, pp. 2778–2787, PMLR, 2017.

[27] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," *Advances in neural information processing systems*, vol. 24, pp. 2249–2257, 2011.

[28] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped dqn," *Advances in neural information processing systems*, vol. 29, 2016.

[29] I. Osband, B. Van Roy, D. J. Russo, Z. Wen, *et al.*, "Deep exploration via randomized value functions.," *J. Mach. Learn. Res.*, vol. 20, no. 124, pp. 1–62, 2019.

[30] M. Fortunato, M. G. Azar, B. Piot, J. Menick, I. Osband, A. Graves, V. Mnih, R. Munos, D. Hassabis, O. Pietquin, *et al.*, "Noisy networks for exploration," *arXiv preprint arXiv:1706.10295*, 2017.

[31] T. Rückstiess, F. Sehnke, T. Schaul, D. Wierstra, Y. Sun, and J. Schmidhuber, "Exploring parameter space in reinforcement learning," *Paladyn*, vol. 1, pp. 14–24, 2010.

[32] M. Plappert, R. Houthooft, P. Dhariwal, S. Sidor, R. Y. Chen, X. Chen, T. Asfour, P. Abbeel, and M. Andrychowicz, "Parameter space noise for exploration," *arXiv preprint arXiv:1706.01905*, 2017.

[33] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International conference on machine learning*, pp. 22–31, PMLR, 2017.

[34] A. Hans, D. Schneegaß, A. M. Schäfer, and S. Udluft, "Safe exploration for reinforcement learning.," in *ESANN*, pp. 143–148, 2008.

[35] W. F. Whitney, M. Bloesch, J. T. Springenberg, A. Abdolmaleki, and M. A. Riedmiller, "Rethinking exploration for sample-efficient policy learning," *ArXiv*, vol. abs/2101.09458, 2021.

[36] W. F. Whitney, M. Bloesch, J. T. Springenberg, A. Abdolmaleki, K. Cho, and M. Riedmiller, "Decoupled exploration and exploitation policies for sample-efficient reinforcement learning," *arXiv preprint arXiv:2101.09458*, 2021.

[37] T. Seyde, I. Gilitschenski, W. Schwarting, B. Stellato, M. Riedmiller, M. Wulfmeier, and D. Rus, "Is bang-bang control all you need? solving continuous control with bernoulli policies," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27209–27221, 2021.

[38] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033, IEEE, 2012.

[39] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement learning with deep energy-based policies," in *International conference on machine learning*, pp. 1352–1361, PMLR, 2017.

[40] M. Toussaint, "Robot trajectory optimization using approximate inference," in *Proceedings of the 26th annual international conference on machine learning*, pp. 1049–1056, 2009.

[41] R. Fox, A. Pakman, and N. Tishby, "Taming the noise in reinforcement learning via soft updates," *arXiv preprint arXiv:1512.08562*, 2015.

[42] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, *et al.*, "Maximum entropy inverse reinforcement learning.," in *Aaai*, vol. 8, pp. 1433–1438, Chicago, IL, USA, 2008.

[43] B. D. Ziebart, *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University, 2010.

[44] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*, pp. 1587–1596, PMLR, 2018.

[45] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[46] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[47] S. Gu, T. Lillicrap, Z. Ghahramani, R. E. Turner, and S. Levine, "Q-prop: Sample-efficient policy gradient with an off-policy critic," *arXiv preprint arXiv:1611.02247*, 2016.

[48] E. Cetin and O. Celiktutan, "Learning pessimism for reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 6971–6979, 2023.

[49] K. Ciosek, Q. Vuong, R. Loftin, and K. Hofmann, "Better exploration with optimistic actor critic," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[50] T. Moskovitz, J. Parker-Holder, A. Pacchiano, M. Arbel, and M. Jordan, "Tactical optimism and pessimism for deep reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12849–12863, 2021.

[51] K. Ghasemipour, S. S. Gu, and O. Nachum, "Why so pessimistic? estimating uncertainties for offline rl through ensembles, and why their independence matters," *Advances in Neural Information Processing Systems*, vol. 35, pp. 18267–18281, 2022.

[52] H. Sun, L. Han, R. Yang, X. Ma, J. Guo, and B. Zhou, "Exploit reward shifting in value-based deep-rl: Optimistic curiosity-based exploration and conservative exploitation via linear reward shaping," *Advances in Neural Information Processing Systems*, vol. 35, pp. 37719–37734, 2022.

[53] S.-I. Amari and S. C. Douglas, "Why natural gradient?," in *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, vol. 2, pp. 1213–1216, IEEE, 1998.

[54] M. Khan, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, and A. Srivastava, "Fast and scalable bayesian deep learning by weight-perturbation in adam," in *International conference on machine learning*, pp. 2611–2620, PMLR, 2018.

[55] M. Opper and C. Archambeau, "The variational gaussian approximation revisited," *Neural computation*, vol. 21, no. 3, pp. 786–792, 2009.

[56] C. M. Stein, "Estimation of the mean of a multivariate normal distribution," *The annals of Statistics*, pp. 1135–1151, 1981.

[57] W. Lin, M. E. Khan, and M. Schmidt, "Stein's lemma for the reparameterization trick with exponential family mixtures," *arXiv preprint arXiv:1910.13398*, 2019.

[58] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[59] D. Makoviichuk, "Rl games: High performance rl library." `https://github.com/Denys88/rl_games`. [Accessed 11-09-2023].

[60] IAS TU Darmstadt, "Experiment launcher." `https://git.ias.informatik.tu-darmstadt.de/common/experiment_launcher`. [Accessed 08-10-2023].

[61] SchedMD, "Slurm workload manager." `https://slurm.schedmd.com/documentation.html`. [Accessed 11-10-2023].

[62] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE transactions on systems, man, and cybernetics*, no. 5, pp. 834–846, 1983.

[63] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[64] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4906–4913, IEEE, 2012.

[65] D. Bertsekas, *Dynamic programming and optimal control: Volume I*, vol. 4. Athena scientific, 2012.

[66] S. Schmitt, M. Hessel, and K. Simonyan, "Off-policy actor-critic with shared experience replay," in *International Conference on Machine Learning*, pp. 8545–8554, PMLR, 2020.

A. Implementation Details

The implementation of the OAC and its variants are based on SAC as described in Algorithm 1. They only differ in the computation of the exploration policy. Further implementation details can be found in the following. In the case of SAC, the exploration policy is the same as the target policy $\pi_E = \pi_T$.

Algorithm 1 Soft Actor Critic

Require: θ_1, θ_2, ϕ ▷ Initial parameters
$\bar{\theta}_1 \leftarrow \theta_1, \bar{\theta}_2 \leftarrow \theta_2$ ▷ Initialize target network weights
$\mathcal{D} \leftarrow \emptyset$ ▷ Initialize replay buffer
for each iteration **do**
  for each environment step **do**
    $\pi_E \leftarrow$ exploration strategy ▷ Compute exploration policy
    $a_t \sim \pi_E(a_t|s_t)$ ▷ Sample action
    $s_t \sim p(s_{t+1}|s_t, a_t)$ ▷ Sample transition from the environment
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r(s_t, a_t), s_{t+1})\}$ ▷ Store transition in the replay pool
  end for
  for each training step **do**
    $\theta_i \leftarrow \theta_i - \lambda_Q \nabla_{\theta_i} \mathcal{L}_Q(\theta_i, \mathcal{D})$ for $i \in \{1, 2\}$ ▷ update Q function parameters
    $\phi \leftarrow \lambda_\pi \nabla_\phi J_\pi(\phi)$ ▷ update policy weights
    $\bar{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\bar{\theta}_i$ for $i \in \{1, 2\}$ ▷ Update target network weights
  end for
end for
return θ_1, θ_2, ϕ ▷ Optimized parameters

A.1. Optimisitic Actor Critic

The computation of the exploration policy in OAC that is used in the SAC framework in Algorithm 1 can be found in Algorithm 2.

Algorithm 2 OAC - exploration policy

Require: Q_1, Q_2, π_T

$\Delta_\mu = \left[\nabla_a \hat{Q}_{UB}(s,a)\right]_{a=\mu_T}$ ▷ Compute gradient

$\mu_E = \mu_T + \frac{\sqrt{2\delta}}{\|\Delta_\mu\|_{\Sigma_T}} \Sigma_T \Delta_\mu$ ▷ Update mean

$\Sigma_E = \Sigma_T$ ▷ Update variance

return $\pi_E \leftarrow (\mu_E, \Sigma_E)$ ▷ Exploration policy

A.2. Natural Gradient Optimisitic Actor Critic

The exploration policy of NGOAC is optimized as decribed in Chapter 4 and the implementation details can be found in Algorithm 3.

Algorithm 3 NGOAC - exploration policy

Require: Q_1, Q_2, π_T

$\Delta_\Sigma = \nabla_\Sigma J(\mu, \Sigma) = \mathbb{E}\left[\nabla_a Q_{UCB}(s,a)\right]$

$\Delta_\mu = \nabla_\mu J(\mu, \Sigma) = \frac{1}{2}\mathbb{E}\left[\Sigma^{-1}(a-\mu)\nabla_a Q_{UCB}(s,a)^T\right]$

$\beta \leftarrow 1$ ▷ Set initial update length

for each linesearch step **do**

  $\Sigma^{-1} = \Sigma_\pi^{-1} - 2\beta\Delta_\Sigma$ ▷ Update variance

  $\mu = \mu_\pi + \beta\Sigma\Delta_\mu$ ▷ Update mean

  if $\mathcal{KL}(\mathcal{N}(\mu,\Sigma), \mathcal{N}(\mu_T, \Sigma_T)) <= \delta$ **then**

    break

  end if

  $\beta \leftarrow$ bracketing search algorithm

end for

return $\pi_E \leftarrow (\mu, \Sigma)$ ▷ Exploration policy

A.3. Optimisitic Actor Critic - RepTrick

The reparameterized version of OAC requires a fixed hyperparameter λ that defines the influence of the KL bound as described in Chapter 5.1.1 and also the learning rate α of the gradient update.

Algorithm 4 OAC RepTrick - exploration policy

Require: $Q_1, Q_2, \pi_T, \lambda, \alpha$

$\Delta_\mu = \nabla_\mu \left[\mathbb{E}_{a\sim\mathcal{N}(\mu,\Sigma)}\left[Q_{UCB}(s,a)\right] - \lambda \cdot \mathcal{KL}(\mathcal{N}(\mu,\Sigma), \mathcal{N}(\mu_T,\Sigma_T))\right]$

$\Delta_\Sigma = \nabla_\Sigma \left[\mathbb{E}_{a\sim\mathcal{N}(\mu,\Sigma)}\left[Q_{UCB}(s,a)\right] - \lambda \cdot \mathcal{KL}(\mathcal{N}(\mu,\Sigma), \mathcal{N}(\mu_T,\Sigma_T))\right]$

$\mu_E \leftarrow \mu_T + \alpha \cdot \Delta_\mu$

$\Sigma_E \leftarrow \Sigma_T + \alpha \cdot \Delta_\Sigma$

return $\pi_E \leftarrow (\mu, \Sigma)$ ▷ Exploration policy

B. Experimental Setup

If not otherwise stated, the experiments are conducted using the following settings for the environments in Table B.1 and for SAC in Table B.2. The setup of OAC in Table B.3, for NGOAC in Table B.4 and OAC-RepTrick in Table B.5 extend the settings of SAC.

Environment	Number of parallel environments
Cartpole	512
Ant	4096
Humanoid	4096

Table B.1.: Number of parallel environments

Hyperparameter		Value
Maximum number of epochs	$epochs_{max}$	1000
Steps per episode	n_{steps}	8
discount factor	γ	0.99
Learning rate entropy	$\alpha_{entropy}$	5e-3
Learning rate actor	α_{actor}	5e-4
Learning rate critic	α_{critic}	5e-4
Exponential target smoothing	τ_{critic}	0.005
Batch size	B	4096
Actor updates per step	$n_{updates}$	12
Warmup episodes	n_{warmup}	5
Replay buffer size	s_{buffer}	1000000

Table B.2.: Experiment setting of SAC

Hyperparameter		Value
Optimization constraint	δ	s. hyperparameter search
Level of optimism	β_{UB}	s. hyperparameter search

Table B.3.: Experiment setting of OAC

Hyperparameter		Value
MC samples	n_{mc}	100
Bracketing searches	$n_{bs,max}$	1000
Optimization constraint	δ	s. hyperparameter search
Level of optimism	β_{UB}	s. hyperparameter search

Table B.4.: Experiment setting of NGOAC

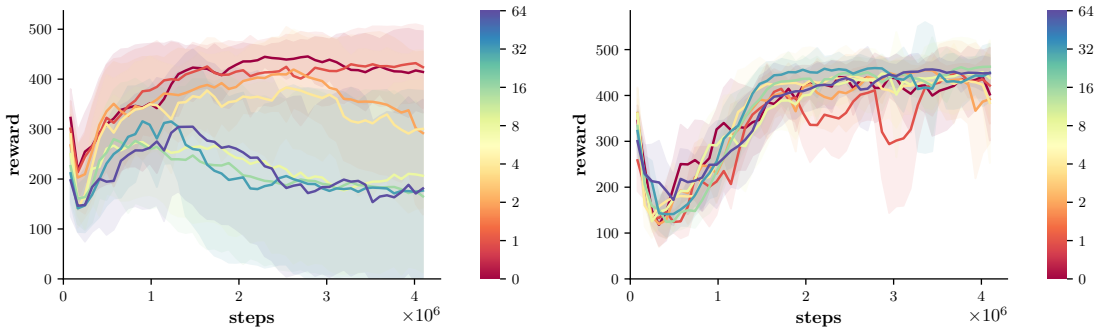
Hyperparameter		Value
MC samples	n_{mc}	100
Learning rate exploration policy	α_{expl}	0.05
Constraint loss	λ	s. hyperparameter search
Level of optimism	β_{UB}	s. hyperparameter search

Table B.5.: Experiment setting of OAC-RepTrick

C. Hyperparameter Search

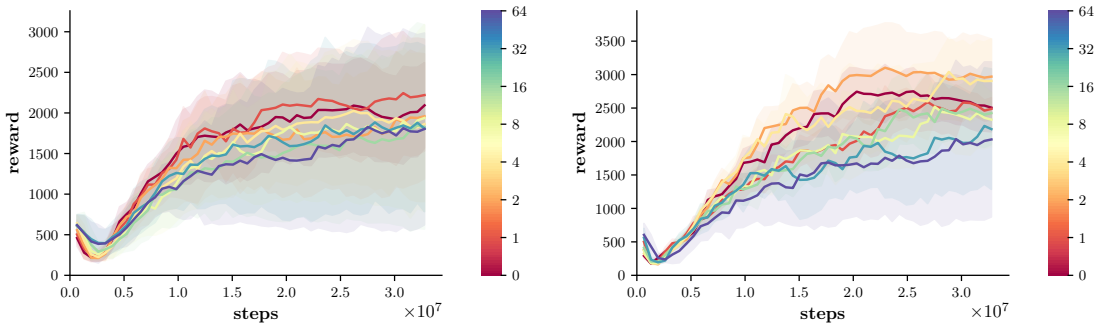
C.1. Level of Optimism for OAC and NGOAC

The reward for different values of β_{UB} are illustrated for both, OAC and NGOAC, in the following Figures for the Cartpole (Figure C.1), Ant (Figure C.2) and Humanoid (Figure C.3). They



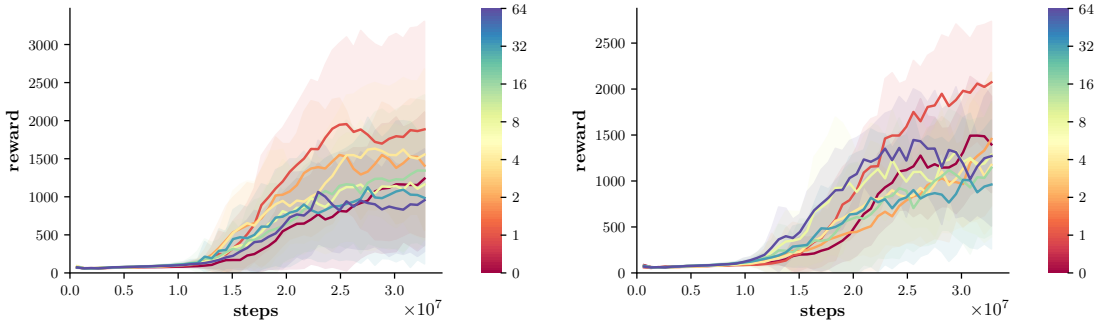
(a) OAC **(b)** NGOAC

Figure C.1.: Reward of OAC (a) and NGOAC (b) in the Cartpole environment for different values of the level of optimism β_{UB}.



(a) OAC **(b)** NGOAC

Figure C.2.: Reward of OAC (a) and NGOAC (b) in the Ant environment for different values of the level of optimism β_{UB}.



(a) OAC **(b)** NGOAC

Figure C.3.: Reward of OAC (a) and NGOAC (b) in the Humanoid environment for different values of the level of optimism β_{UB}.

C.2. OAC RepTrick

In this section, the grid search over the hyperparameters of OAC RepTrick is depicted in the subsequent figures. In the context of hyperparameter sensitivity, the reparameterized optimization approach demonstrates resilience across all hyperparameter pairs within each of the three environments.

C.2.1. Cartpole

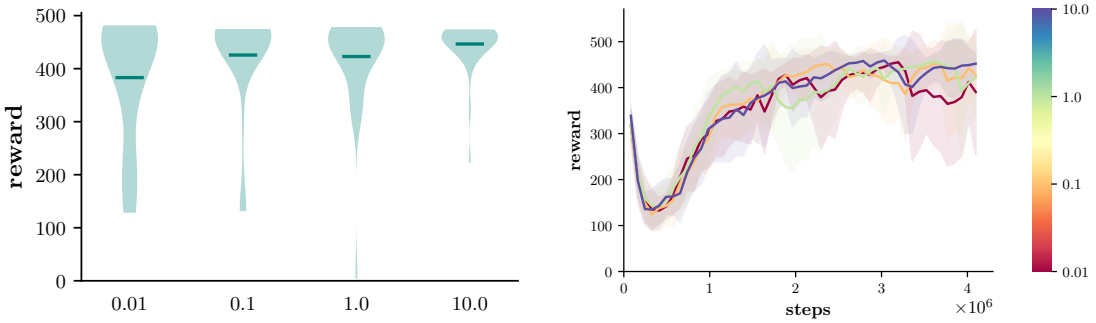


Figure C.4.: Reward of OAC-RepTrick in the Cartpole environment for different values of λ. The reward over the steps is shown on the right and the averaged reward over the last five evaluation episodes is shown on the left.

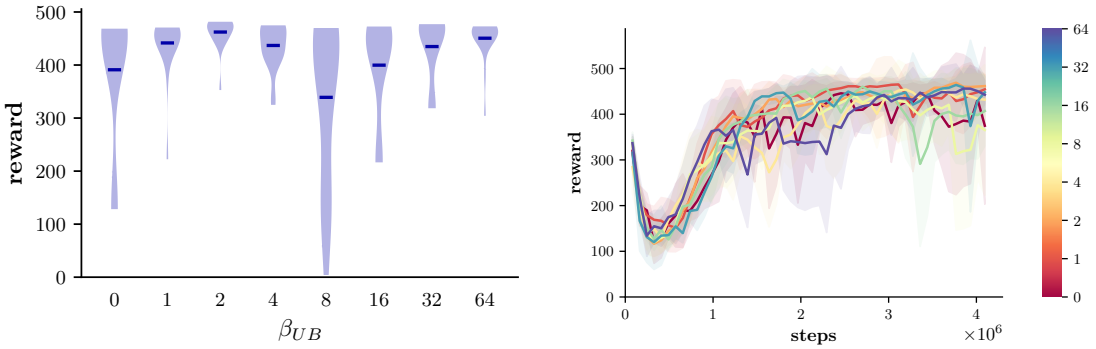
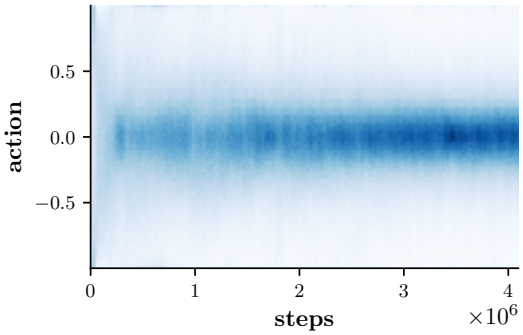
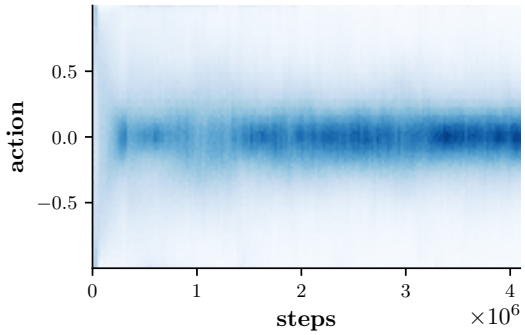


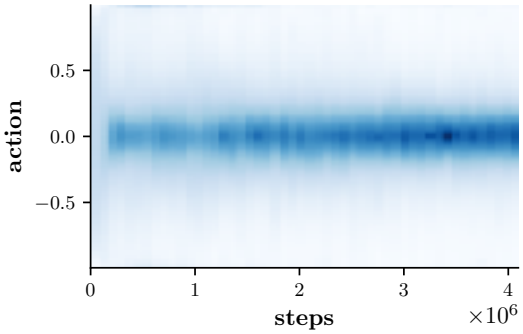
Figure C.5.: Reward of OAC-RepTrick in the Cartpole environment for different values of β_{UB}. The reward over the steps is shown on the right and the averaged reward over the last five evaluation episodes is shown on the left.



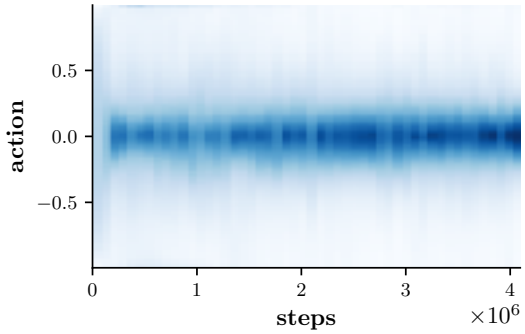
(a) Exploration actions with $\lambda = 0.01$



(b) Exploration actions with $\lambda = 10.0$



(c) Evaluation actions with $\lambda = 0.01$



(d) Evaluation actions with $\lambda = 10.0$

Figure C.6.: Action histogram of OAC-RepTrick in the Cartpole environment with different values for λ. The top row shows the actions during training, and the figures at the bottom correspond to the actions during evaluation.

C.2.2. Ant

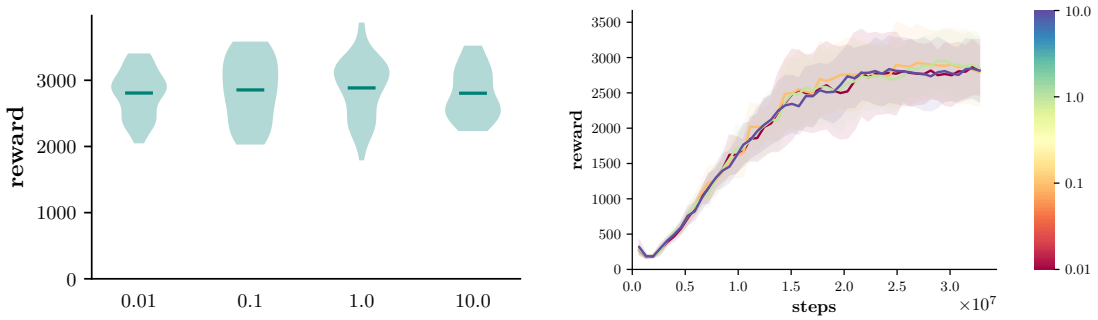


Figure C.7.: Reward of OAC-RepTrick in the Ant environment for different values of λ. The reward over the steps is shown on the right and the averaged reward over the last five evaluation episodes is shown on the left.

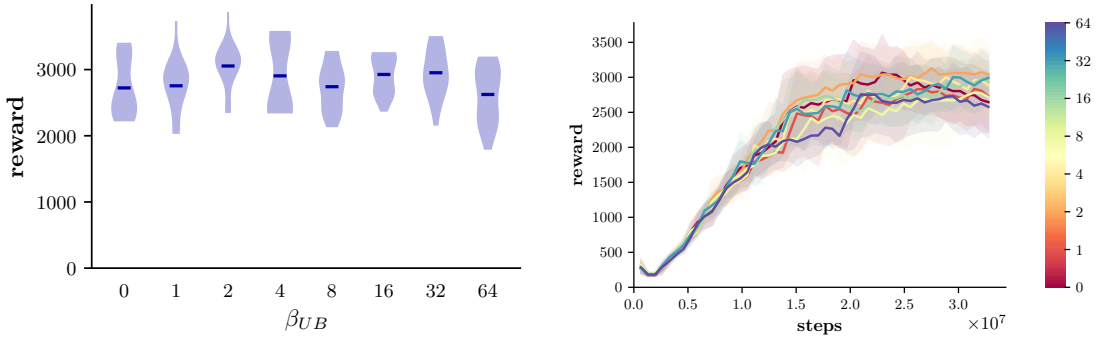


Figure C.8.: Reward of OAC-RepTrick in the Ant environment for different values of β_{UB}. The reward over the steps is shown on the right and the averaged reward over the last five evaluation episodes is shown on the left.

C.2.3. Humanoid

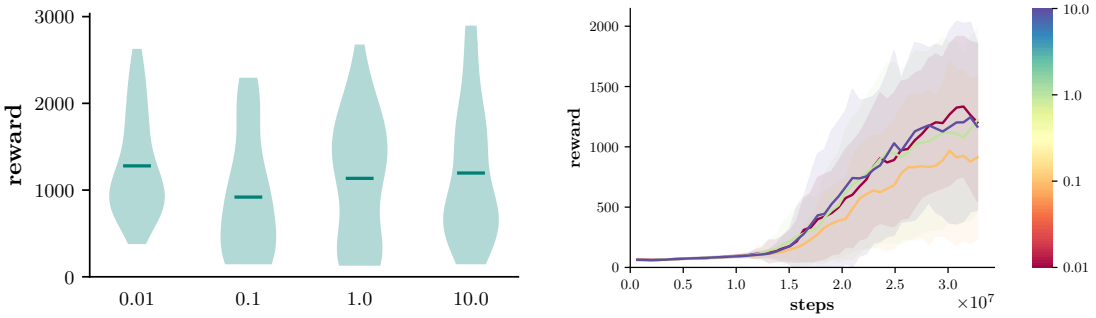


Figure C.9.: Reward of OAC-RepTrick in the Humanoid environment for different values of λ. The reward over the steps is shown on the right and the averaged reward over the last five evaluation episodes is shown on the left.

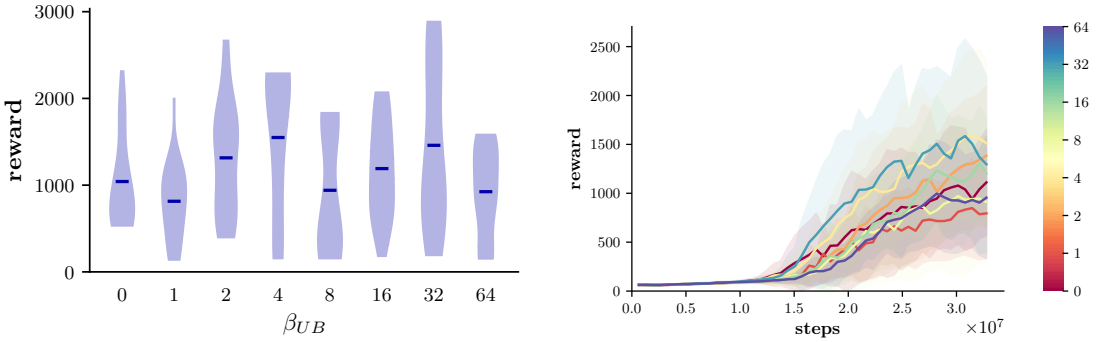


Figure C.10.: Reward of OAC-RepTrick in the Humanoid environment for different values of β_{UB}. The reward over the steps is shown on the right and the averaged reward over the last five evaluation episodes is shown on the left.

Cartpole			Ant			Humanoid		
β_{UB}	λ	reward	β_{UB}	λ	reward	β_{UB}	λ	reward
2	0.01	477.04	4	0.1	3496.05	32	10	2037.12
4	0.1	471.36	4	10	3383.76	2	1	1879.80
2	0.1	470.95	32	0.1	3249.92	4	0.01	1820.67
32	0.1	470.82	2	1	3227.60	32	1	1797.80
64	10	467.79	0	0.01	3196.13	4	10	1790.08

Table C.1.: Best hyperparameter pairs of OAC-RepTrick.

C.3. Detailed Results

In this section the results of all hyperparameter runs are stated.

C.3.1. Cartpole

β_{UB} \ δ	0.1	1.0	10.0	100.0	200.0	1000.0
0	430.8	424.8	470.1	446.1	294.2	424.2
1	434.1	428.0	467.4	440.1	436.1	360.8
2	433.6	332.7	307.3	244.3	440.2	168.7
4	444.7	443.1	92.7	215.3	278.3	323.2
8	455.9	472.5	139.6	66.7	40.6	66.7
16	365.2	467.3	71.7	52.3	22.0	73.9
32	457.3	449.0	73.9	18.4	30.9	19.7
64	366.1	454.7	85.1	22.8	42.0	82.6

Table C.2.: Mean rewards of OAC in the Cartpole environment

β_{UB} \ δ	0.1	1.0	10.0	100.0	200.0	1000.0
0	461.0	374.4	405.2	403.1	464.4	463.3
1	442.9	432.3	431.8	424.8	434.1	433.6
2	334.1	456.8	427.9	389.8	467.0	329.2
4	392.2	440.9	453.8	436.3	374.1	420.1
8	408.0	448.1	448.8	454.5	458.0	459.1
16	452.4	436.7	465.6	470.1	470.9	467.8
32	409.5	410.9	424.3	463.9	464.1	466.5
64	420.2	463.3	473.7	455.4	446.9	431.3

Table C.3.: Mean rewards of NGOAC in the Cartpole environment

β_{UB} \ λ	0.01	0.1	10.0	100.0
0	193.7	453.3	466.3	450.8
1	453.0	457.4	442.3	413.7
2	477.0	470.9	453.4	447.3
4	432.7	471.3	399.2	444.1
8	330.3	195.3	371.6	458.8
16	243.6	454.7	467.0	433.5
32	467.7	470.8	345.3	455.4
64	466.2	431.0	437.1	467.7

Table C.4.: Mean rewards of OAC-RepTrick in the Cartpole environment

C.3.2. Ant

β_{UB} \ δ	0.1	1.0	10.0	100.0	200.0	1000.0
0	2473.5	2208.2	2540.3	1535.7	1687.4	1564.4
1	3075.8	2904.9	2350.1	1885.1	1881.5	1135.2
2	2907.1	2084.8	2543.9	1822.6	1099.2	1120.2
4	2410.5	3101.2	2273.1	1495.0	1435.0	512.4
8	2900.0	2907.4	2650.7	1470.0	962.5	275.8
16	2351.8	3482.4	2400.8	1613.7	544.8	194.7
32	3057.3	2811.1	2401.6	1464.6	729.1	528.0
64	2807.3	3355.5	2480.9	914.6	437.6	659.6

Table C.5.: Mean rewards of OAC in the Ant environment

β_{UB} \ δ	0.1	1.0	10.0	100.0	200.0	1000.0
0	3278.4	2511.7	2613.6	2583.1	2100.8	2334.5
1	2560.7	2440.3	2540.3	2615.0	2582.2	2420.3
2	2789.3	2648.9	2499.7	3430.2	2833.1	3567.4
4	2986.6	3031.3	3204.9	3258.7	1918.8	3156.5
8	2895.1	2980.7	2017.4	2239.5	2113.0	1822.9
16	3088.6	2295.1	2754.6	2446.7	1836.8	2107.1
32	2731.5	2546.5	2820.4	2115.5	1704.8	618.7
64	3665.7	2654.5	2718.5	1166.4	1053.4	368.6

Table C.6.: Mean rewards of NGOAC in the Ant environment

β_{UB} \ λ	0.01	0.1	10.0	100.0
0	3196.1	2271.2	2894.7	2531.3
1	2804.5	2393.7	3093.7	2729.8
2	3074.4	2908.9	3227.5	3039.6
4	2389.3	3496.0	2444.2	3383.7
8	2733.2	3016.1	2648.2	2513.8
16	2777.0	2752.6	3188.2	2988.6
32	2731.5	3249.9	2960.1	2868.8
64	2754.1	2701.8	2707.3	2333.1

Table C.7.: Mean rewards of OAC-RepTrick in the Ant environment

C.3.3. Humanoid

β_{UB} \ δ	0.1	1.0	10.0	100.0	200.0	1000.0
0	592.0	760.6	1796.2	2013.9	1111.5	776.4
1	489.6	1698.9	2218.0	3237.3	2846.4	556.4
2	2233.2	1300.5	1950.6	887.6	1523.5	857.7
4	791.9	2283.9	2940.0	1360.3	1204.8	387.1
8	1515.0	2472.4	1236.3	615.9	536.3	484.3
16	1929.9	1689.4	2254.4	617.2	524.7	395.9
32	1038.9	2268.6	1217.1	250.0	434.4	436.3
64	788.5	1868.3	1339.4	447.9	482.5	512.9

Table C.8.: Mean rewards of OAC in the Humanoid environment

β_{UB} \ δ	0.1	1.0	10.0	100.0	200.0	1000.0
0	1985.8	1688.7	1913.1	1093.3	1066.6	870.1
1	1037.3	1610.2	2507.5	2543.6	1893.7	2524.6
2	2149.4	1833.1	651.4	958.1	1171.9	704.7
4	569.0	2093.1	1621.6	1312.1	425.1	938.8
8	1947.3	1854.0	1929.0	536.3	863.2	195.8
16	1210.6	2025.7	1097.9	352.2	361.9	948.6
32	628.5	1885.4	1446.3	370.4	372.6	1053.8
64	2162.4	1497.2	917.5	539.8	1097.9	844.9

Table C.9.: Mean rewards of NGOAC in the Humanoid environment

β_{UB} \ λ	0.01	0.1	10.0	100.0
0	1044.2	876.5	1502.8	669.1
1	1018.7	847.2	701.5	673.3
2	1353.7	577.7	1879.8	1297.3
4	1820.6	1427.4	1030.9	1790.0
8	1548.6	575.9	676.6	875.6
16	1426.9	1730.0	999.2	596.1
32	1161.4	770.8	1797.8	2037.1
64	944.5	713.3	749.3	1316.3

Table C.10.: Mean rewards of OAC-RepTrick in the Humanoid environment

D. Prioritized Replay Buffer

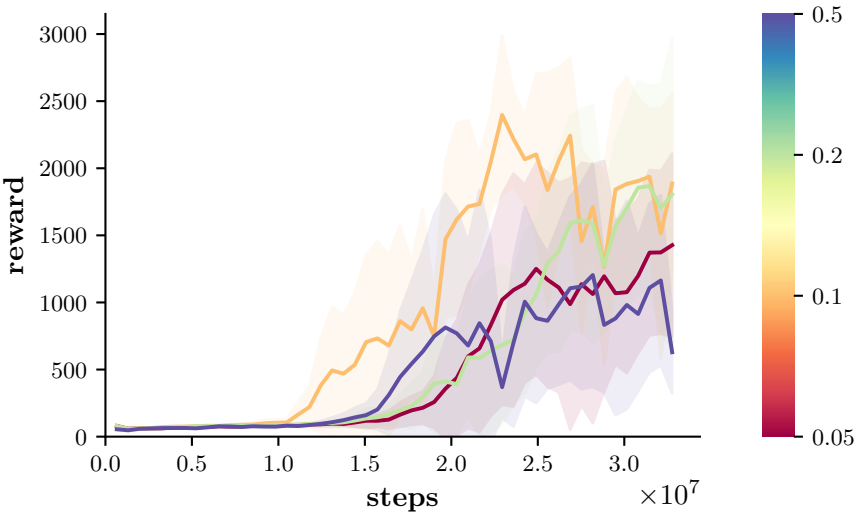


Figure D.1.: Reward of OAC in the Humanoid environment for different prioritized replay buffers.

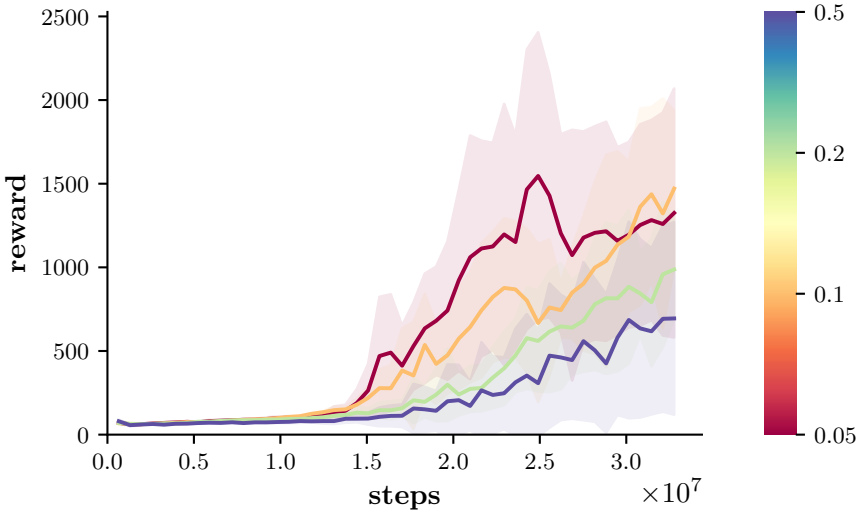


Figure D.2.: Reward of NGOAC in the Humanoid environment for different prioritized replay buffers.

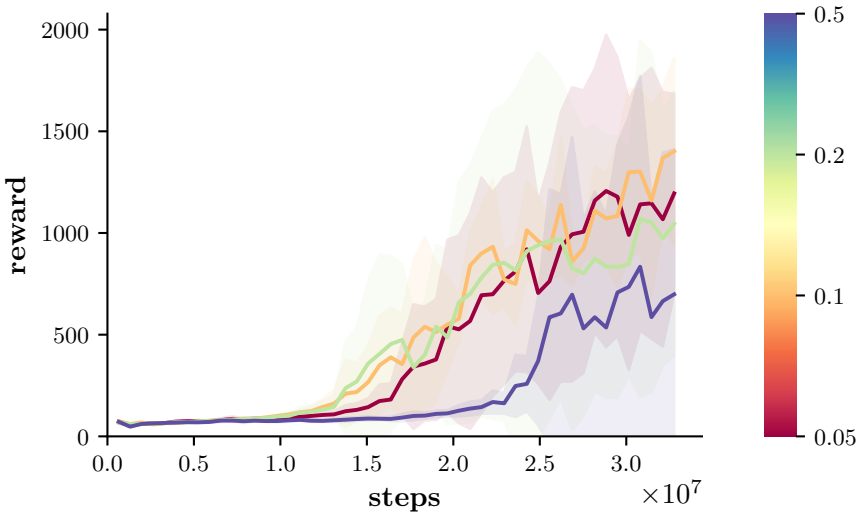


Figure D.3.: Reward of OAC-RepTrick in the Humanoid environment for different prioritized replay buffers.