Second Order Extension of Optimistic Actor Critic

Niklas Kappes [1] Pascal Herrmann [1]

Abstract

Optimistic Actor-Critic (OAC), an optimistic exploration algorithm based on *Soft Actor Critc* (SAC), addresses the problems of pessimistic underexploration and directional uninformedness of existing exploration strategies. We evaluate OAC and analyse its key exploration behaviour. In addition we show that the first order approximation of OAC has weaknesses and because of that, we introduce *Optimistic Actor Critic with Second Order Approximation* (OAC2), a second order extension of OAC.

We highlight its theoretical advantages, which result in an exploration policy achieving higher exploitation ability, so that the exploitation/exploration trade-off of the exploration policy can be better balanced. We evaluate it on different robotic control tasks of the MuJoCo engine and show practical issues, but also discuss further potential for improvements.

1. Introduction

Reinforcement Learning (RL) (Sutton & Barto, 1998) algorithms use different exploration methods to achieve state-of-the-art performance. Current algorithms use an information gain given current beliefs (Russo & Roy, 2014), posterior sampling or also known as Thompson sampling (Chapelle & Li, 2011) and optimistic exploration strategies like the *upper confidence bound* (UCB) (Auer et al., 2002) to explore new actions while enforcing exploitation to achieve higher rewards.

Optimistic Actor Critic (OAC) (Ciosek et al., 2019) introduces a locally approximated exploration policy that enables a more optimistic exploration behaviour. This addresses the problems of pessimistic underexploration and directional uninformedness of existing exploration strategies, which occur for state-of-the-art actor-critic algorithms that use two state-action value function approximations to reduce overestimation bias by computing an approximate lower confidence bound (Hasselt, 2010) (Van Hasselt et al., 2016) and the same policy for exploration and exploitation, as used in *Soft Actor Critic* (SAC) (Haarnoja et al., 2018) and *Twin Delayed Deep Deterministic* (TD3) (Fujimoto et al., 2018). OAC improves the exploration/exploitation trade-off by maximising a linear approximated upper bound on the state-action value function to obtain a better exploration policy (Brafman & Tennenholtz, 2002).

The optimisation problem is constrained by a trust region (Yuan, 2000) and here, a problem of OAC arises. Due to the linear objective, the exploration policy is determined by the boundary of the trust region. But at that boundary, the state-action value function does not necessarily have to have higher values compared to the approximated point and can even have lower values. Therefore OAC does not guarantee an exploration policy that results in a better action based on the current state-value function. Because of this, we extend OAC with a second order approximation of the upper bound. Theoretically, this second order objective leads to an optimal solution within the trust region, which is illustrated in Figure 1.

With the second order extension of OAC the improvements of optimistic exploration and directional informedness are guaranteed based on the nature of OAC. In contrast to OAC, where the exploitation ability of the exploration policy is limited due to the fixed step-size of the exploration update, the second order approximation enables better exploitation, which leads to a better exploitation/exploration trade-off of the exploration policy, controlled by an additional entropy constraint.

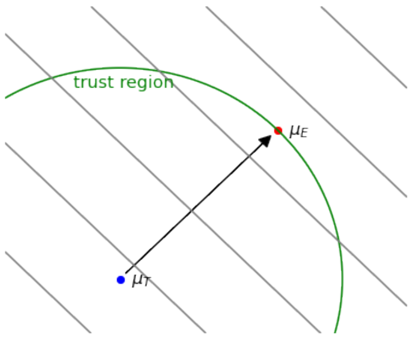
2. Preliminaries

2.1. Reinforcement Learning

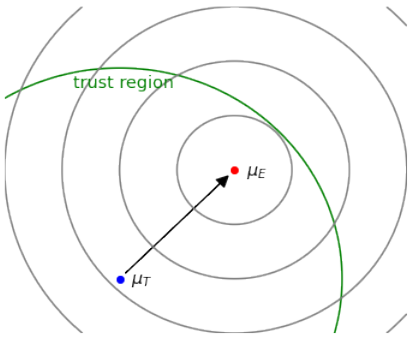
In *Reinforcement learning* (RL), we consider an infinite *Markov Decision Process* (MDP) (Puterman, 2014), defined by the tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where an agent observes the current environment state $s_t \in \mathcal{S}$ and performs an action $a_t \in \mathcal{A}$. This leads to a new environment state $s_{t+1} \sim p(\cdot|s_t, a_t)$, where $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, \infty)$ repre-

*Equal contribution [1]Department of Computer Science, Technische Universität Darmstadt, Germany. Correspondence to: Niklas Kappes <niklas.kappes@stud.tu-darmstadt.de>, Pascal Herrmann <pascal.herrmann1@stud.tu-darmstadt.de>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).



(a) linear objective



(b) quadratic objective

Figure 1. Trust region optimisation with a linear and a quadratic objective function, where π_T corresponds to the current point and π_E shows the optimised solution constrained by the trust region.

sents the probability density of state transitions. For each transition in the environment the agent receives a reward $r : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}$. The overall goal of the agent is to maximise the total expected reward $J = \sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t)]$, where $\rho_\pi(s_t)$ and $\rho_\pi(s_t,a_t)$ denote 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 (Sutton & Barto, 1998).

2.2. Soft Actor Critic

SAC is a model-free RL algorithm, that uses an entropy-regularised reinforcement learning setup, such that an additional entropy term $\mathcal{H}$ of the policy is added, where α defines the trade-off coefficient to improve exploration, similar to a temperature term.

$$J = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t) + \alpha\mathcal{H}(\pi(\cdot|s_t))\right] \tag{1}$$

Here, the min-double-Q trick (Hasselt, 2010) is also used to stabilise the off-policy learning process.

$$\hat{Q}_{LB}(s_t,a_t) = \min\left(Q^1_{\theta_1}(s_t,a_t), Q^2_{\theta_2}(s_t,a_t)\right) \tag{2}$$

Both Q functions $Q^1_\theta(s_t,a_t)$ and $Q^2_\theta(s_t,a_t)$ are learned by minimising the soft Bellman residual (MSBE) using the same target state-action value y', where $\hat{a}_{s+t} \sim \pi_T(\cdot|s_{t+1})$ (Haarnoja et al., 2018).

$$y' = r(s_t,a_t) + \gamma \min_{j=1,2}\big[Q^j_{\theta_j,target}(s_{t+1},\hat{a}_{t+1}) - \alpha\log\pi(\hat{a}_{t+1}|s_{t+1})\big] \tag{3}$$

The policy is optimised using the reparameterisation trick, that allows reformulation of the expectation over actions by sampling from $\varepsilon \sim \mathcal{N}(0,I)$ and squashing a deterministic dependency of the state, policy parameters ϕ and random noise as stated in Equation 4.

$$\hat{a}_\phi(s,\varepsilon) = \tanh\left(\mu_\phi(s) + \sigma_\theta(s) \bullet \varepsilon\right) \tag{4}$$

Therefore the objective in Equation 1 can be rewritten, where the state is sampled from a replay buffer $\mathcal{D}$ and the action is obtained by Equation 4.

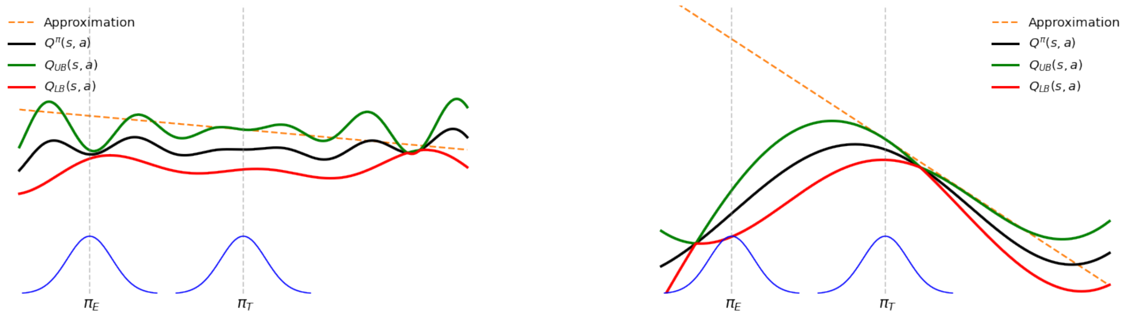
$$J_\pi = \mathbb{E}_{\substack{s\sim\mathcal{D}\\ \varepsilon\sim\mathcal{N}}}\big[\min_{j=1,2} Q^j_{\theta_j}(s,\hat{a}_\phi(s,\varepsilon)) - \alpha\log\pi_\phi(\hat{a}_\phi(s,\varepsilon)|s)\big] \tag{5}$$

2.3. Optimistic Actor Critic

OAC obtains an upper confidence bound $\hat{Q}_{UB}$ using the uncertainty estimate of the two Q-function approximations and a Gaussian to model epistemic uncertainty.

$$\begin{aligned}\mu_Q(s,a) &= \frac{1}{2}(Q^1_{LB}(s,a) + Q^2_{LB}(s,a))\\ \sigma_Q(s,a) &= \frac{1}{2}|Q^1_{LB}(s,a) - Q^2_{LB}(s,a)|\end{aligned} \tag{6}$$

With β_{UB} the level of optimism can be controlled to influence the exploration behaviour of the exploration policy.



(a) beginning of learning (b) end of learning

Figure 2. Comparison of a Gaussian exploration policy π_E with mean μ_E obtained by Equation 10 for different training progresses, where the upper bound of the Q function is locally approximated around a Gaussian target policy π_T with mean μ_T. Q^{π} refers to the true Q function, whereas Q_{LB} and Q_{UB} indicate the lower and upper bound. In Figure 2a a flat Q function encodes a higher level of uncertainty in the beginning and a sharper defined one represents a well explored state-action space in Figure 2b at the end of training, where the best true action can be clearly obtained from.

$$\hat{Q}_{UB}(s,a) = \mu_Q(s,a) + \beta_{UB}\sigma_Q(s,a) \tag{7}$$

To derive a separate exploration policy π_E the upper confidence bound is locally linear approximated around the current target policy π_T, that corresponds to the policy learned by the SAC policy update in Equation 5.

$$\bar{Q}_{UB}(s,a) = a^T\left[\nabla_a \hat{Q}_{UB}(s,a)\right]_{a=\mu_T} + const \tag{8}$$

Maximising this approximated upper bound increases the chances of executing informative and opportunistic actions. Since the optimisation objective is linear, a *Kullback-Leibler* (KL) divergence constraint is added to bound the maximisation, preserve stability of the optimisation and the update.

$$\begin{aligned} \mu_E, \Sigma_E = \arg\max_{\mu,\Sigma} \mathbb{E}_{a\sim\mathcal{N}(\mu,\Sigma)}\left[\bar{Q}_{UB}(s,a)\right] \\ \text{s. t. } \mathcal{KL}(\mathcal{N}(\mu,\Sigma),\mathcal{N}(\mu_T,\Sigma_T)) \leq \delta \end{aligned} \tag{9}$$

For the linear approximation and Gaussian policies, this can be solved in closed-form to achieve the exploration policy defined by the mean μ_E and variance Σ_E.

$$\begin{aligned} \mu_E &= \mu_T + \sqrt{2\delta}\frac{\Sigma_T\left[\nabla_a \hat{Q}_{UB}(s,a)\right]_{a=\mu_T}}{||\left[\nabla_a \hat{Q}_{UB}(s,a)\right]_{a=\mu_T}||_{\Sigma_T}} \\ \Sigma_E &= \Sigma_T \end{aligned} \tag{10}$$

In addition OAC extends the actor update of SAC with a modification of the lower bound to obtain more conservative and stable policy updates, where β_{LB} treats the level of pessimism similar to the upper bound in Equation 7.

$$\hat{Q}'_{LB} = \mu_Q(s,a) + \beta_{LB}\sigma_Q(s,a) \tag{11}$$

OAC avoids pessimistic underexploration, since the exploration policy π_E is not symmetric with respect to the mean of the target policy π_T. Thereby it also solves the problem of directional uninformedness, that can be seen in Figure 2. Because the exploration policy is achieved every time the agent draws an action from scratch, it does not influence the critic directly and is only used for exploration (Ciosek et al., 2019).

3. Analysis of Exploration in OAC

3.1. Optimistic exploration

To evaluate the pure exploration benefits of OAC, we first omit the changes on the lower bound in Equation 11 by setting $\beta_{LB} = -1$, as it makes it harder to distinguish between the impact of the optimised exploration policy based on the usage of the upper bound and the more pessimistic lower bound in comparison to SAC. As shown in Figure 5 OAC achieves better exploration as SAC and beyond better performance on the evaluated environments. Especially in the first period of the training process the benefits can be clearly seen.

In theory a trust region optimisation with a linear objective only results in an optimal update direction, whereas the length or in this case the shift between the target and exploration policy is fixed by the maximum allowed KL-divergence as shown in Figure 1, since the constraint is always active (Ciosek et al., 2019).

Accordingly this results in a sub-optimal exploration policy and furthermore makes the exploration sensitive in relation to the maximum shift parameter δ in Equation 9 as it determines the shift between the mean of the exploration and target policy in Equation 10.

In the beginning of the learning of the agent, the Q function is more shallow due to the random initialisation of the underlying neural network. Assuming a good converging learning process, with progressive learning the Q function gets more defined and converges to the optimal / true Q function exemplified in Figure 2. Contemporaneously the mean of the target policy converges to the maximum of the optimal Q function due to the actor update in SAC (Equation 5). In addition the entropy shrinks naturally, especially in the automated entropy adjusted version of SAC (Haarnoja et al., 2019).

This leads to better exploration in the beginning, but to exploration of known suboptimal actions at the end of training, when the action space is sufficiently good explored. Because the exploration policy indirectly influences the target policy, this can increase potentially instabilities, when the agent converges to a good solution.

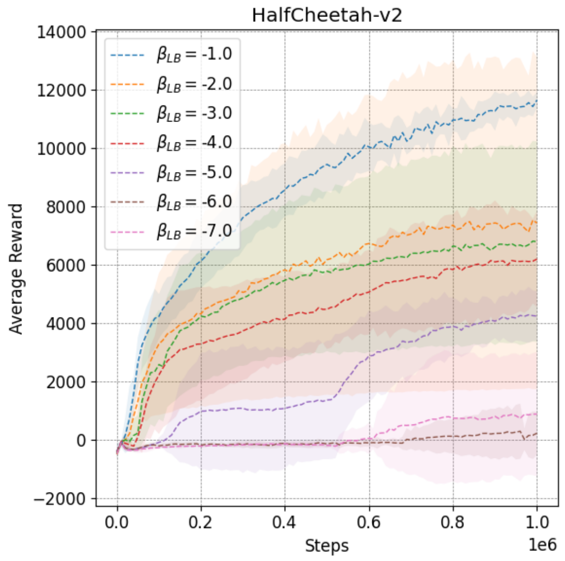


Figure 3. OAC evaluated in the *HalfCheetah-v2* environment with different lower bounds generated by $\beta_{LB} \in [-7, -6, ..., -1]$, evaluated for 5 random seeds and hyperparameters described in Appendix B. The x-axis shows the number of steps while the y-axis shows the average reward.

3.2. Examination of β_{LB}

(Ciosek et al., 2019) introduced an extension of the lower confidence bound by using the hyperparameter β_{LB} in Equation 11, that replaces the min-double-Q trick in Equation 2. In general lowering the lower bound $\beta_{LB} < -1$ for the target Q function value of the MSBE in Equation 3 and for the policy update in Equation 5 amplifies the phenomenon of pessimisitc underexploration, described in (Ciosek et al., 2019). Accordingly it makes it harder to learn the true Q function, since the lower bound becomes much smoother and prevents the agents to exploit a good solution after several training steps.

As (Ciosek et al., 2019) found an optimal hyperparameter value $\beta_{LB} = -3.65$, this seems to contradict the pessimistic underexploration described above. In contrast, our results in Figure 3 for the *HalfCheetah-v2* environment (Todorov et al., 2012) best describes the impact of the adaption of the lower bound, whereby similar results are shown Figure 6 for the *Ant-v2* and *Hopper-v2* environments.

It can be seen, that $\beta_{LB} = -1$ performs best in all three environments. For the *Hopper-v2* and *HalfCheetah-v2*, the performance drops monotonically with decreasing values of β_{LB}, while *Ant-v2* shown not the exact, but a similar behaviour. With this in mind, we can not comprehend the results of (Ciosek et al., 2019), since $\beta_{LB} = -3.65$ does not correspond to the best performance. Hence we do not make use of the adaption of the lower bound for the following experiments, keeping the original lower bound from Equation 2, that is equivalent to $\beta_{LB} = -1$ (Ciosek et al., 2019).

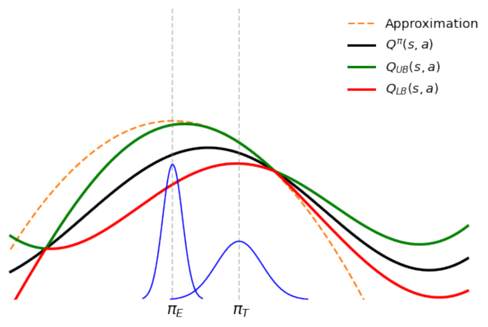


Figure 4. The upper bound of the Q function is approximated quadratically. Q^π refers to the true Q function, whereas Q_{LB} and Q_{UB} indicate the lower and upper bound. μ_T is the target policy around which the upper bound is approximated. μ_E is the exploration policy.

4. Optimistic Second Order Approximation

4.1. Motivation

Based on the insights of trust region optimisation exemplified in Figure 1, we can apply the quadratic objective on

Algorithm 1 Optimistic Actor Critic with Second Order Approximation (OAC2)

```
Require: $\theta_1, \theta_2, \phi$ {Initial parameters $\theta_1, \theta_2$ of the critic and $\phi$ of the target policy $\pi_T$}
 1: $\check{\theta}_1 \leftarrow \theta_1, \check{\theta}_2 \leftarrow \theta_2, \mathcal{D} \leftarrow \emptyset$ {Initialize target network weights and replay pool}
 2: for each iteration do
 3:   for each environment step do
 4:     $\bar{Q}_{UB} \approx \hat{Q}_{UB}$ {second-order approximation (12)}
 5:     if $\bar{Q}_{UB}$ is concave then
 6:       $\pi_E \leftarrow$ (14) {Maximisation of quadratic approximation}
 7:     else
 8:       $\pi_E \leftarrow$ (10) {Maximisation of linear approximation}
 9:     end if
10:     $a_t \sim \pi_E(a_t|s_t)$ {Sample from the exploration policy}
11:     $s_t \sim p(s_{t+1}|s_t, a_t)$ {Sample transition from the environment}
12:     $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t R(s_t, a_t), s_{t+1})\}$ {Store the transition in the replay pool}
13:   end for
14:   for each training step do
15:     for $i \in \{1, 2\}$ do {Update two bootstraps of the critic}
16:       update $\theta_i$ with $\hat{\nabla}_{w_i} ||\hat{Q}^i_{LB}(s_t, a_t) - R(s_t, a_t) - \gamma \min(\check{Q}^1_{LB}(s_{t+1}, a), \check{Q}^2_{LB}(s_{t+1}, a))||^2_2$
17:     end for
18:     update $\phi$ with $\nabla_\phi \hat{J}^\alpha_{\hat{Q}'_{LB}}$ {Policy gradient update}
19:     $\check{\theta}_1 \leftarrow \tau\theta_1 + (1-\tau)\check{\theta}_1, \check{\theta}_2 \leftarrow \tau\theta_2 + (1-\tau)\check{\theta}_2$ {Update target network}
20:   end for
21: end for
output $\theta_1, \theta_2, \phi$ {Optimised parameters}
```

OAC. As OAC obtains the exploration policy by optimising a linear approximation in Equation 8, we use a quadratic objective. The impact of a second order approximation instead of a linear approximation in Equation 8 can be exemplified by comparing Figure 2b and 4. As mentioned before, the optimisation of the linear approximation only lead to an optimal direction, but does not scale the direction length of the updated, that determines the exploration policy. Accordingly the update length is defined by the active KL-divergence constraint.

In Figure 2b, this would lead to a worse action than just using the target policy action, since the value of the Q-function is smaller. The quadratic approximation in Figure 4, on the other hand, approximates the Q-function very well around the target policy. Accordingly to the optimisation of the quadratic objective, this not only leads to a optimal update direction, but also to a better update length, since the optimal approximated Q value lies within the KL-divergence constraint. Therefore the exploration policy obtained by the quadratic approximation naturally exploits the true optimal Q value better.

4.2. Second Order Approximation

First of all we derive the second order approximation of the upper bound given in Equation 7. Therefore we introduce the following abbreviations to facilitate the derivation: Let H be the second order and h the first order derivative of $\hat{Q}_{UB}$ w.r.t a, both for a given state s and evaluated for $a = \mu_T$. Then, the second order approximation is given as follows.

$$\begin{aligned} \bar{Q}_{UB}(s,a) =& \frac{1}{2}(a^T H a - 2a^T H \mu_T + \mu_T^T H \mu_T) + \\ & a^T h + \mu_T^T h + \Big[\hat{Q}_{UB}(s,a)\Big]_{a=\mu_T} \\ \text{with } H =& \Big[\nabla_a^2 \hat{Q}_{UB}(s,a)\Big]_{a=\mu_T} \\ h =& \Big[\nabla_a \hat{Q}_{UB}(s,a)\Big]_{a=\mu_T} \end{aligned} \tag{12}$$

4.3. Optimisation of the Second Order Approximation

The optimisation objective keeps similar to the optimisation of the linear approximation in Equation 9, where we maximise the expectation of the approximated upper bound $\bar{Q}_{UB}(s, a)$.

Proposition 1 (Optimization objective for the second order approximation)**.**
The optimisation objective for the second order approximation is:

$$J = \mu^T \frac{1}{2} H\mu + \frac{1}{2} tr(\Sigma H) + \mu^T (h - H\mu_T) + \frac{1}{2}\mu_T^T H\mu_T - \mu_T^T h + \left[\hat{Q}_{UB}(s,a)\right]_{a=\mu_T}. \tag{13}$$

Proof. See Appendix A.1. □

To ensure that the exploration policy does not collapse to zero variance, as it is the case when optimising the original problem in Equation 9 using a quadratic objective, we add an additional entropy constraint (Abdolmaleki et al., 2015). Through this, the exploration policy would take different actions, when the underlying state-action function approximations do not change, e.g. after few training steps. Accordingly this prevents the exploration policy from purely exploitation and controls the exploration/exploitation trade-off of the exploration policy by guaranteeing a minimum entropy ϵ.

$$\begin{aligned} \mu_E, \Sigma_E = \arg\max_{\mu,\Sigma} \mathbb{E}[\bar{Q}_{UB}(s,a)] \\ \text{s. t. } \mathcal{KL}(\mathcal{N}(\mu,\Sigma), \mathcal{N}(\mu_T,\Sigma_T)) \leq \delta, \\ \mathcal{H}(\mathcal{N}(\mu,\Sigma)) \geq \epsilon \end{aligned} \tag{14}$$

The stated optimisation problem can not be solved completely in closed form, in contrast to the closed form solution of OAC in Equation 10. Therefore it can only be solve numerically using any constrained nonlinear optimiser.

Furthermore, this can be improved by optimising the dual formulation, as the exploration policy can be expressed in terms of Lagrange multipliers (Abdolmaleki et al., 2015).

Proposition 2 (Optimisation of the dual formulation)**.** *The exploration policy resulting from Equation 14 has the form $\pi_E = \mathcal{N}(\mu_E, \Sigma_E)$, where*

$$\begin{aligned} \mu_E &= (\lambda\Sigma_T^{-1} - 2R)^{-1}(\lambda\Sigma_T^{-1}\mu_T + r) \\ \Sigma_E &= (\lambda\Sigma_T^{-1} - 2R)^{-1}(\lambda + \omega) \end{aligned}$$

with the Lagrangian multipliers λ and ω obtained by optimising the dual formulation in Equation 15.

Proof. See Appendix A.2. □

4.4. Non-concave Maximisation

Since we use a quadratic approximation, the curvature of the optimisation objective in Equation 14 can become non-concave, especially for high-dimensional state-action spaces. In this case the optimal solution is defined by the active KL-divergence constraint, assuming that the entropy of the target policy satisfies the entropy constraint. This leads to the closed form solution of the original OAC, using a linear approximation on the upper bound.

In Table 1 can be seen, that the objective is more likely to be non-concave. Especially in the higher-dimensional *Ant-v2* environment, the objective is almost always non-concave, through with the optimisation is only solved using the linear approximation.

Table 1. Number of optimisation problems with concave objective as a percentage of the total number of steps in the evaluated environments for OAC2.

ENVIRONMENT	CONCAVE OBJECTIVE
HALFCHEETAH-V2	$5.95 \pm 0.49\%$
HOPPER-V2	$15.11 \pm 0.33\%$
ANT-V2	$0.21 \pm 0.20\%$

The second order extension of OAC including the curvature evaluation of the objective function is defined as *Optimistic Actor Critic with Second Order Approximation* (OAC2) in Algorithm 1.

If the exploration policy is optimised using the dual formulation accordingly to the Proposition 2, then the Lagrange multiplier λ in Equation 15 can be increased, such that F becomes positive definite (Abdolmaleki et al., 2015). Therefore the objective of the dual optimisation stays convex and the benefits of the second order approximation can be applied more often in practise.

4.5. Evaluation of OAC2

We evaluate OAC2 in comparison to SAC and OAC in Figure 5, whereby we optimise the original maximisation problem stated in Equation 14 and do not make use of the dual formulation.

For a better comparison, we train SAC, OAC and OAC2 on a one-legged jumping robot (*Hopper*), a two-legged cheetah robot (*HalfChetah*) and a four-legged walking robot (*Ant*) environment of the Multi-Joint dynamics with Contact (*MuJoCo*) physics engine (Todorov et al., 2012).

All agents are trained for 1 million steps in the environment with the same lower bound $\beta_{LB} = -1$ for the policy update to ensure the comparability of the pure exploration behaviour between the optimistic exploration methods OAC and OAC2 and the implementation of SAC as described in Section 3. Additionally we use the same tuned hyperparameters for the upper bound β_{UB} and the shift coefficient δ of (Ciosek et al., 2019). The entropy hyperparameter ϵ of the second order optimisation extension of OAC2 in Equation 14 is also tuned and can be found in the Appendix B, accordingly to the other parameters of the experiments.

Accordingly to (Ciosek et al., 2019), OAC outperforms SAC

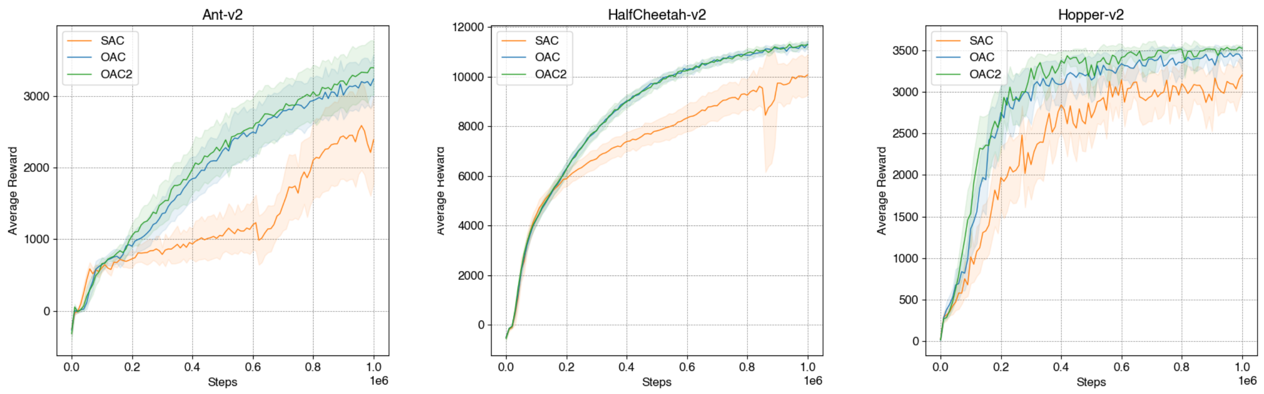


Figure 5. Comparison of OAC2, OAC and SAC, evaluated in the *Ant-v2*, *HalfCheetah-v2* and *Hopper-v2* environment for 25 random seeds[2] and hyperparameters described in Appendix B. The x-axis shows the number of steps while the y-axis shows the average reward.

after 1 million steps for all three environments. Especially in the first iterations the benefits of OAC can be clearly seen, even if OAC does not guarantee a better exploration policy due to the linear objective of the trust region optimisation that can result in worse action at the optimisation boundary. This can be justified by the additional randomness of OAC, which is encoded in the optimistic upper bound, since unexplored and low-expected Q-function values can also lead to faster exploration of the state-action space and thus to an increase in performance.

Based on the results, OAC2 seems to perform a bit better than OAC due to the second order approximation which guarantees a better exploration policy compared to OAC and also ensures fast exploration of the state-action space by using an optimistic upper bound. Nevertheless the influence of the theoretical improvements of the second order approximation is not as much as expected, which can be seen in the very similar behaviour of OAC2 and OAC. Especially in the *HalfCheetah-v2* environment the differences are not noticeable. This corresponds to the low number of concave optimisation steps in Table 1, whereby the second order approximation is only used a few times. In the *Hopper-v2*, the second order approximation is used for about 15% of the steps in the environment and accordingly the improvement is small, but noticeable. Regarding the evaluation of the concave objective in Table 1, the improvement of OAC2 in the *Ant-v2* environment is higher than expected as the second order approximation is only applied in 0.21% of the steps. This is based on the higher- dimensional state-action space of the environment in comparison to the low dimensional *hopper-v2*.

Overall, the impact of the second order approximation can not be clearly examined, but shows the potential for improvement despite the low level of application of the second order approximation.

5. Conclusion

In this paper we extended OAC by a second order approximation of the upper confidence bound on the Q function and introduced OAC2. While the theoretical potential of the second order approximate is manifestly, in practice, the poor applicability due to the curvature of the optimisation objective, as seen in Table 1, prevents a clear conclusion as to whether OAC2 improves OAC. Nevertheless, small improvements can be seen in all three evaluated environments.

This problem can be improved by optimising the dual function to gain more control over the curvature of the optimisation objective. This would increase the number of usages of the second order approximation in higher dimensional state-action spaces and therefore the impact of OAC2 on the entire learning process. Accordingly, the theoretical advantages of OAC2 are more often applied and we expect higher improvements in the experiments.

References

Abdolmaleki, A., Lioutikov, R., Peters, J., Lau, N., Reis, L., and Neumann, G. Model-based relative entropy stochastic search. In *Advances in Neural Information Processing Systems (NIPS / NeurIPS)*. mit press, 2015. URL `http://www.ausy.tu-darmstadt.de/uploads/Team/GerhardNeumann/Abdolmaleki_NIPS2015.pdf`.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time

[2]*Notice: In HalfCheetah-v2, SAC is visualised using only 10 random seeds.*

analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

Brafman, R. I. and Tennenholtz, M. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct): 213–231, 2002.

Chapelle, O. and Li, L. An empirical evaluation of thompson sampling. *Advances in neural information processing systems*, 24:2249–2257, 2011.

Ciosek, K., Vuong, Q., Loftin, R., and Hofmann, K. Better exploration with optimistic actor-critic, 2019.

Fujimoto, S., Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596. PMLR, 2018.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., and Levine, S. Soft actor-critic algorithms and applications, 2019.

Hasselt, H. Double q-learning. *Advances in neural information processing systems*, 23:2613–2621, 2010.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2017.

Puterman, M. L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Russo, D. and Roy, B. V. Learning to optimize via information directed sampling. *CoRR*, abs/1403.5556, 2014. URL http://arxiv.org/abs/1403.5556.

Sutton, R. and Barto, A. Reinforcement learning: An intro, 1998.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Yuan, Y.-x. A review of trust region algorithms for optimization. In *Iciam*, volume 99, pp. 271–282, 2000.

Supplementary Material

A. Proofs

A.1. Proof of Proposition 1

Proposition 1 (Optimization objective for the second order approximation)**.**
The optimisation objective for the second order approximation is:

$$J = \mu^T \frac{1}{2} H\mu + \frac{1}{2} tr(\Sigma H) + \mu^T (h - H\mu_T) + \frac{1}{2}\mu_T^T H \mu_T - \mu_T^T h + \Big[\hat{Q}_{UB}(s,a)\Big]_{a=\mu_T}.$$

Proof. The second order approximation of Q_{UB} is:

$$\bar{Q}_{UB}(s,a) = \frac{1}{2}(a^T H a - 2a^T H\mu_T + \mu_T^T H \mu_T) + a^T h + \mu_T^T h + \Big[\hat{Q}_{UB}(s,a)\Big]_{a=\mu_T}.$$

Using the trace, this can be rearranged to

$$\begin{aligned}\bar{Q}_{UB}(s,a) &= \frac{1}{2}(tr(a^T H a) - 2a^T H\mu_T + \mu_T^T H \mu_T) + a^T h + \mu_T^T h + \Big[\hat{Q}_{UB}(s,a)\Big]_{a=\mu_T} \\ &= \frac{1}{2}(tr(aa^T H) - 2a^T H\mu_T + \mu_T^T H \mu_T) + a^T h + \mu_T^T h + \Big[\hat{Q}_{UB}(s,a)\Big]_{a=\mu_T}.\end{aligned}$$

The objective is then to maximise this approximation. Here, we use the linearity of the trace and the expectation.

$$\begin{aligned}J = \mathbb{E}[\bar{Q}_{UB}(s,a)] &= \frac{1}{2}(tr(\mathbb{E}[aa^T]H) - 2\mu^T H\mu_T + \mu_T^T H \mu_T) + \mu^T h + \mu_T^T h + \Big[\hat{Q}_{UB}(s,a)\Big]_{a=\mu_T} \\ &= \frac{1}{2}(tr(\mu\mu^T H) + tr(\Sigma H) - 2\mu^T H\mu_T + \mu_T^T H \mu_T) + \mu^T h - \mu_T^T h + \Big[\hat{Q}_{UB}(s,a)\Big]_{a=\mu_T} \\ &= \mu^T \frac{1}{2} H\mu + \frac{1}{2} tr(\Sigma H) + \mu^T (h - H\mu_T) + \frac{1}{2}\mu_T^T H \mu_T - \mu_T^T h + \Big[\hat{Q}_{UB}(s,a)\Big]_{a=\mu_T}\end{aligned}$$

□

A.2. Proof of Proposition 2

Proposition 2 (Optimisation of the dual formulation)**.**
The exploration policy resulting from Equation 14 has the form $\pi_E = \mathcal{N}(\mu_E, \Sigma_E)$, where

$$\begin{aligned}\mu_E &= (\lambda\Sigma_T^{-1} - 2R)^{-1}(\lambda\Sigma_T^{-1}\mu_T + r) \\ \Sigma_E &= (\lambda\Sigma_T^{-1} - 2R)^{-1}(\lambda + \omega)\end{aligned}$$

with the lagrangian multipliers λ and ω obtained by optimising the dual formulation in Equation 15.

Proof. Consider the rewritten second order approximation of the upper confidence bound of Equation 12:

$$\begin{aligned}\bar{Q}_{UB}(s,a) =& a^T \underbrace{\frac{1}{2}\left[\nabla_a^2 \hat{Q}_{UB}(s,a)\right]_{a=\mu_T}}_{R} a + a^T \left(\underbrace{\left[\nabla_a \hat{Q}_{UB}(s,a)\right]_{a=\mu_T} - \left[\nabla_a^2 \hat{Q}_{UB}(s,a)\right]_{a=\mu_T} \mu_T}_{r}\right) + \\ & \underbrace{\left[\hat{Q}_{UB}(s,a)\right]_{a=\mu_T} - \left[\nabla_a \hat{Q}_{UB}(s,a)\right]_{a=\mu_T} \mu_T + \frac{1}{2}\left[\nabla_a^2 \hat{Q}_{UB}(s,a)\right]_{a=\mu_T} \mu_T^2}_{r_0} \\ =& a^T R a + a^T r + r_0\end{aligned}$$

Given the multivariate gaussian policies with dimension p

$$\pi = \mathcal{N}(\mu, \Sigma) = ((2\pi)^p \det(\Sigma))^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

and the entropy

$$\mathcal{H}(\mathcal{N}(\mu,\Sigma)) = \frac{p}{2} + \frac{p}{2}\log(2\pi) + \frac{1}{2}\log(\det(\Sigma)).$$

Accordingly the KL divergence between the a exploration π_E and target π_T policy is states as:

$$\mathcal{KL}(\mathcal{N}(\mu,\Sigma), \mathcal{N}(\mu_T,\Sigma_T)) = \frac{1}{2}\left[\operatorname{tr}\left(\Sigma_T^{-1}\Sigma\right) + (\mu_T-\mu)^T\Sigma_T^{-1}(\mu_T-\mu) - p + \log\left(\frac{\det \Sigma_T}{\det \Sigma}\right)\right]$$

Using the original optimisation problem in Equation 14 with the second order approximation and an additional entropy constraint, we can reformulate the Lagrangian function, introducing the Lagrangian multipliers λ and ω.

$$\begin{aligned}\mathcal{L}(\mu,\Sigma,\lambda,\omega) =& \mathbb{E}_{a\sim\mathcal{N}(\mu,\Sigma)}\left[a^T R a + a^T r + r_0\right] + \lambda\left(\delta - \frac{1}{2}\left[\operatorname{tr}\left(\Sigma_T^{-1}\Sigma\right) + (\mu_T-\mu)^T\Sigma_T^{-1}(\mu_T-\mu) - p + \log\left(\frac{\det\Sigma_T}{\det\Sigma}\right)\right]\right) + \\ & \omega\left(\frac{p}{2} + \frac{p}{2}\log(2\pi) + \frac{1}{2}\log(\det\Sigma) - \epsilon\right) \\ =& \mu^T R\mu + \operatorname{tr}(\Sigma R) + \mu^T r + r_0 + \lambda\left(\delta - \frac{1}{2}\left[\operatorname{tr}\left(\Sigma_T^{-1}\Sigma\right) + (\mu_T-\mu)^T\Sigma_T^{-1}(\mu_T-\mu) - p + \log\left(\frac{\det\Sigma_T}{\det\Sigma}\right)\right]\right) + \\ & \omega\left(\frac{p}{2} + \frac{p}{2}\log(2\pi) + \frac{1}{2}\log(\det\Sigma) - \epsilon\right)\end{aligned}$$

The optimal mean and variance of the exploration policy can be obtained by maximising the Lagrangian function:

$$\begin{aligned}\nabla_\mu \mathcal{L}(\mu,\Sigma,\lambda,\omega) &= 2\mu R + r - \lambda\Sigma_T^{-1}(\mu-\mu_T) \overset{!}{=} 0 \\ \implies \mu^\star &= (\lambda\Sigma_T^{-1}\mu_T + r)(\lambda\Sigma_T^{-1} - 2R)^{-1}\end{aligned}$$

$$\begin{aligned}\nabla_\Sigma \mathcal{L}(\mu,\Sigma,\lambda,\omega) &= R + \frac{\lambda}{2}\left(\Sigma^{-1} - \Sigma_T^{-1}\right) + \frac{\omega}{2}\Sigma^{-1} \overset{!}{=} 0 \\ \implies \Sigma^\star &= (\lambda\Sigma_T^{-1} - 2R)^{-1}(\lambda+\omega)\end{aligned}$$

By making use of the following substitutions $F = (\lambda\Sigma_T^{-1} - 2R)^{-1}$ and $f = (\lambda\Sigma_T^{-1}\mu_T + r)$ we can obtain the dual formulation as followed:

$$g(\lambda,\omega) = \mathcal{L}(\mu^\star,\Sigma^\star,\lambda,\omega) = \lambda\delta - \omega\beta + \frac{1}{2}\left(f^T F f - \lambda\mu_T^T\Sigma_T^{-1}\mu_T - \lambda\log|2\pi\Sigma_T| + (\lambda+\omega)\log|2\pi(\lambda+\omega)F|\right) \quad (15)$$

Accordingly the exploration policy $\pi_E = \mathcal{N}(\mu^\star,\Sigma^\star)$ can be expressed as $\pi_E(a) = \mathcal{N}(Ff, F(\lambda+\omega))$. $\square$

B. Environment Setup and Hyperparameters

The following hyperparameters are used for the MuJoco environments *HalfCheetah-v2*, *Ant-v2* and *Hopper-v2*, that are used to evaluate the different algorithms in the result plots.

Table 2. SAC Hyperparameters

HYPERPARAMETER		VALUE
HORIZON		100
DISCOUNT FACTOR	γ	0.99
TARGET SMOOTHING COEFFICIENT	τ	0.005
EPOCHS		100
STEPS/EPISODE		1000
EPISODES EVALUATION		10
BATCH SIZE		256
WARMUP TRANSITION		10000
MAX REPLAY SIZE		10^6
CIRITIC NETWORK		[256, 256] RELU
ACTOR NETWORK		[256, 256] RELU
OPTIMIZER		ADAM [3]
LEARNING RATE ACTOR	α_π	3×10^{-4}
LEARNING RATE CRITIC	α_Q	3×10^{-4}

Table 3. OAC Hyperparameters - extending the hyperparameters of SAC in Table 2

HYPERPARAMETER		VALUE	INTERVAL FOR SEARCH[4]
SHIFT MULTIPLIER	$\sqrt{2\delta}$	6.86	[0, 12]
UPPER CONFIDENCE BOUND	β_{UB}	4.66	[0, 7]
LOWER CONFIDENCE BOUND	β_{LB}	-1.0	[-7, -1]

Table 4. OAC2 Hyperparameters - extending the hyperparameters of SAC in Table 2 and OAC in Table 3

HYPERPARAMETER		VALUE	INTERVAL FOR SEARCH
ENTROPY	ϵ	-6.0	[-8, 0]

[3](Kingma & Ba, 2017)
[4](Ciosek et al., 2019)

C. Baselines and additional results

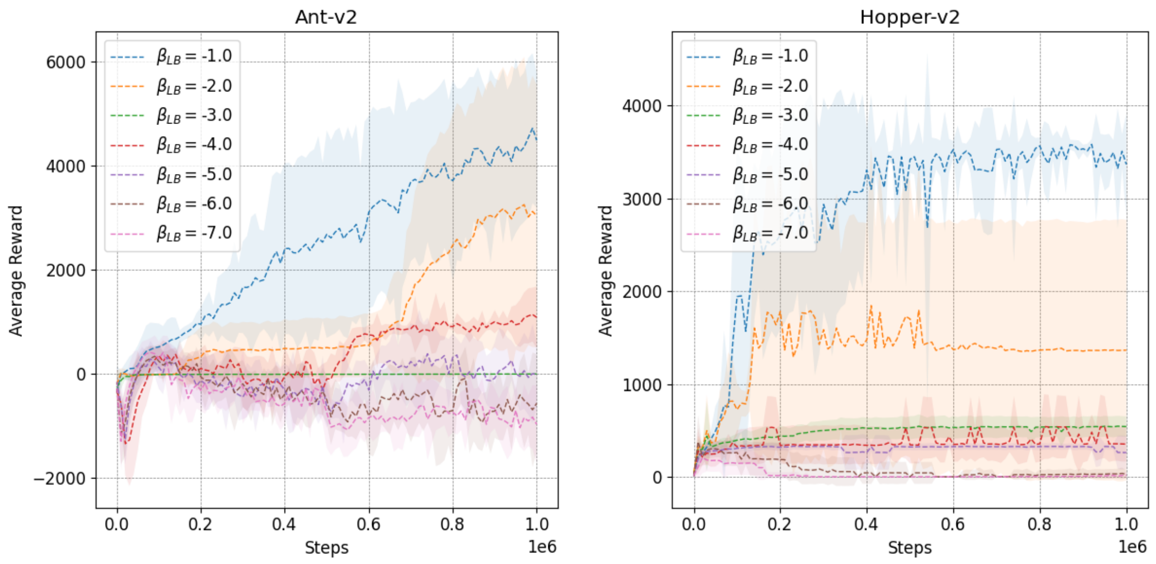


Figure 6. OAC evaluated in the *Ant-v2* and *Hopper-v2* environment with different lower bounds generated by $\beta_{LB} \in [-7, -6, ..., -1]$, evaluated for 5 random seeds and hyperparameters described in Appendix B. The x-axis shows the number of steps while the y-axis shows the average reward.