Neural Processing Letters (2024) 56:80
https://doi.org/10.1007/s11063-024-11461-y

Parameter-Free Reduction of the Estimation Bias in Deep Reinforcement Learning for Deterministic Policy Gradients

Baturay Saglam[1] · **Furkan Burak Mutlu**[1] · **Dogan Can Cicek**[1] · **Suleyman Serdar Kozat**[1]

Accepted: 8 October 2023 / Published online: 2 March 2024

Abstract
Approximation of the value functions in value-based deep reinforcement learning induces overestimation bias, resulting in suboptimal policies. We show that when the reinforcement signals received by the agents have a high variance, deep actor-critic approaches that overcome the overestimation bias lead to a substantial underestimation bias. We first address the detrimental issues in the existing approaches that aim to overcome such underestimation error. Then, through extensive statistical analysis, we introduce a novel, parameter-free Deep Q-learning variant to reduce this underestimation bias in deterministic policy gradients. By sampling the weights of a linear combination of two approximate critics from a highly shrunk estimation bias interval, our Q-value update rule is not affected by the variance of the rewards received by the agents throughout learning. We test the performance of the introduced improvement on a set of MuJoCo and Box2D continuous control tasks and demonstrate that it outperforms the existing approaches and improves the baseline actor-critic algorithm in most of the environments tested.

Keywords Deep reinforcement learning · Actor-critic methods · Estimation bias · Deterministic policy gradients · Continuous control

✉ Baturay Saglam
baturay@ee.bilkent.edu.tr

Furkan Burak Mutlu
burak.mutlu@ee.bilkent.edu.tr

Dogan Can Cicek
cicek@ee.bilkent.edu.tr

Suleyman Serdar Kozat
kozat@ee.bilkent.edu.tr

[1] Department of Electrical and Electronics Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey

1 Introduction

Policy optimization in reinforcement learning (RL) has achieved notable successes in a wide range of sequential decision-making tasks, such as for neural network systems [1–4] or the control of partially observable systems [5, 6]. However, in the deep setting of RL, where deep neural networks approximate value functions and policies, there exist several issues [7]. The systematic estimation bias that prevents the learning agents from attaining maximum performance and applicability of the deep techniques to diverse real-world tasks is one of the difficulties originating from the function approximation [7, 8]. For discrete action spaces, the estimation bias on the value estimates has been widely investigated for the value-based RL algorithms [9–13]. In addition, similar work is done in the continuous action domains with actor-critic techniques for the subtype of the estimation bias, namely, overestimation bias [7]. However, our recent work [14] demonstrated that actor-critic methods that overcome the overestimation bias and accumulated high variance induce an underestimation bias on the action-value estimates.

In continuous control, the estimation bias on the action-value estimates is generally examined under underestimation and overestimation [14]. Overestimation bias, caused by the maximization of the noisy estimates in traditional Q-learning [15], results in a cumulative estimation error on the action values (state-action values or Q-values) throughout the learning stage [7]. As deep neural networks represent the action and value functions in the deep RL setting, such a function approximation noise is inevitable [7]. Due to the temporal-difference (TD) learning [8], this inaccuracy in the value estimation is further amplified [7]. The underestimation bias, in contrast, is an outcome of Q-learning [15] variants that focus on eliminating the accumulated overestimation bias [14]. Although a recent objective function proposal in the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm [7], Clipped Double Q-learning, is shown to eliminate the overestimation bias and accumulated variance, it can nevertheless decrease an RL agent's performance by assigning low values to optimal state-action pairs and thus, may result in suboptimal policies and divergent behaviors [14].

In the Clipped Double Q-learning algorithm [7], two Q-networks with identical structures and different parameters are initialized before the learning process [7]. The minimum of these critics' estimates is utilized to form the objective of Q-networks during learning. Despite the decoupled actor and critics, using the minimum Q-value in learning the targets results in persistent underestimation of the state-action values [14]. Recent works, Weighted Delayed Deep Deterministic Policy Gradient (WD3) [16] and Triplet-average Deep Deterministic Policy Gradient (TADD) [17], focus on this existing underestimation bias in the TD3 algorithm [7] and introduce a linear combination of the functions of action-value estimates in forming the objective of Q-networks. Although the recent objective function proposals [16, 17] are shown to reduce the underestimation bias and improve the TD3 algorithm [7], their theoretical assumptions on the underestimation of Q-values are either on a strict basis or infeasible assumptions that prevent their approach to be adapted to the off-policy learning. Furthermore, our recent work for the underestimation bias problem, Triplet Critic Deep Deterministic Policy Gradient (TCD3) [14], heuristically searches for an alternative for the Q-network objective and proposes a combination of three approximate critics. However, maintaining three deep networks comes with an intensive computational complexity compared to the TD3 algorithm [7].

In this paper, we extend our previous study [14] on the estimation bias such that we first examine the current strategies that aim to overcome the underestimation bias in deterministic policy gradient (DPG) [18] algorithms. We address the detrimental issues in these algorithms

and explain the infeasible assumptions in their theoretical background. We then derive a closed-form expression for the estimation error yielded by the Clipped Deep Q-learning algorithm [7] and our previous work TCD3 [14], without any statistical assumptions that violate the off-policy RL paradigm, which was not introduced in our previous work. Using the derived closed-form expressions, we introduce a new variant of Deep Q-learning [19], Stochastic Weighted Twin Critic Update, that achieves superior performance to our previous work but with using only two critics and hence, having 33% less time complexity. Our approach derives a parameter-free linear combination of the functions of two approximate critics. The weights are sampled from a bias interval corresponding to a significantly smaller underestimation bias than the existing approaches. In addition to our previous work on the underestimation bias, the main contributions of this study can be summarized as follows:

- We first address the issues with the existing algorithms that focus on the underestimation in deterministic policy gradient [18] methods. We explain why the statistical assumptions made in those works cannot be adapted to the off-policy deep RL in continuous action spaces.
- We derive a closed-form expression for the estimation bias in the Clipped Double Q-learning algorithm [7] and TCD3 [14]. Theoretically, we show that if the rewards that the agent receives vary on a large scale, the underestimation of the action-value estimates detrimentally increases.
- Through an extensive statistical analysis of the expected error in the existing approaches and derived closed-form expressions, we introduce a stochastic Q-network update rule in which weights are sampled from a bias interval that is substantially smaller than the expected errors in the existing approaches and TCD3 [14].
- We empirically verify our claims by comparing the actual and estimated Q-values produced by the WD3 [16] and TADD [17] algorithms and demonstrating that an increasing variance of the received reward signals increases the underestimation throughout the learning.
- Our method is not affected by the variance of the reward signals as it samples the weights of the Q-networks from an interval, the lower bound of which is linearly decreased. An extensive set of empirical studies in several challenging OpenAI Gym [20] tasks reflect our theoretical claims and show that the introduced approach outperforms the competing methods and improves our previous study in most of the MuJoCo [21] and Box2D [22] continuous control tasks or provides nearly the same result.
- The source code of our algorithm is publicly available at our GitHub repository[1] to ensure reproducibility.

2 Related Work

Prior studies on the approximation error in reinforcement learning have been done by [23, 24] regarding the estimation bias and resulting high variance build-up. This paper focuses on one of the function approximation error outcomes, namely, underestimating the action-values. In the following, we extensively investigate the background of the estimation error phenomenon in deep reinforcement learning.

[1] https://github.com/baturaysaglam/SWTD3.

2.1 Estimation Bias

The estimation error induced by the maximization of Q-values has been extensively studied in discrete action spaces. For Deep Q-learning [19], many techniques were proposed to mitigate the impacts of the overestimation bias caused by the function approximation and policy optimization. Hasselt et al. [9] address the function approximation error for discrete action spaces in their work, Deep Double Q-learning (DDQN), which is one of the successor studies to Deep Q-learning [19]. By employing two independent and identically structured Q-value approximators, DDQN [9] obtains unbiased Q-value estimates. Lan et al. [10] modify Deep Q-learning [19] by utilizing of multiple action-value estimators. Their approach, Maxmin Q-learning, uses multiple action-value estimates selected through partial maximum operators, the minimum of which constructs the Deep Q-learning target [19]. Additionally, methods that employ multi-step returns are shown to overcome the estimation bias [11–13, 25] and have proven to be effective through distributed approaches [12], weighted Q-learning [25], and importance sampling [11–13, 26]. Lastly, [27] proposed a one-step improvement to reduce the contribution of each erroneous estimate by reducing the discount factor in a structured manner.

Although [10] primarily aims to eliminate the overestimation, they show that their method may yield an underestimation [10]. In contrast to our method, their approach is also not generalizable to continuous action domains since they do not consider actor networks; hence, it only operates in discrete action domains. Furthermore, methods that rely on multi-step returns [11–13, 25] introduce a trade-off between the biased action-value estimates and accumulated variance, as shown by Schmitt et al. [26]. Compared with the one-step solutions to the bias-variance trade-off, employing multi-step returns might also be impractical due to the increased memory demands caused by collecting long trajectories, i.e., the temporally correlated experiences used in the multi-step TD-learning [8, 11].

While the estimation bias in discrete action space is overcome by the existing Deep Q-learning [19] variants, they cannot be adapted to the control of the continuous systems since they do not consider the existence of a separate actor network that chooses continuous actions [7]. As infinitely many intractable actions exist in continuous action domains, the maximization of Q-networks cannot be used to select actions. Hence, these mentioned works cannot be used in continuous action domains, in contrast to our introduced algorithm.

2.2 Function Approximation Error

In continuous control, a direct and one-step solution to the overestimation and variance accumulation has been proposed by Fujimoto et al. [7]. It is shown to It is shown to effectively eliminate the function approximation error for the deep setting of RL. Their research demonstrates that the deep function approximation of Q-values causes overestimation bias and cumulative variance in continuous action domains, which causes the approximate gradient of the actor network to diverge from the actual gradient. An extension of temporal-difference learning [11] in the DPG [18] methods, Clipped Double Q-learning [7], on which we build our algorithm, presents a direct remedy to the overestimation problem by employing two identically structured Q-networks. On top of Clipped Double Q-learning [7], the delayed actor updates and target policy regularization through additive policy noise constitute the TD3 algorithm [7]. TD3 [7] overcomes the overestimation build-up by performing the target Q-value computation through a minimum of two approximate critics. Their introduced

update rule, Clipped Double Q-learning [7], is further used in many state-of-the-art continuous control algorithms.

While the improvements introduced by Clipped Double Q-learning [7] can eliminate cumulative estimation error, using the minimum of two critics causes an underestimation bias in the Q-value estimations, as empirically shown [14, 16, 17, 28, 29]. First, Wang et al. [29] focused on the function approximation error in Ensemble Q-learning, in which multiple function approximators are used to estimate the action values. While the estimation bias heavily relies on the ensemble size, determining it is highly nontrivial because of the time-varying nature of the function approximation errors during the learning process. The authors first derived an upper and lower bound on the estimation bias to tackle such a challenge. Based on these bounds, the ensemble size is then adjusted to reduce the estimation bias to nearly zero, effectively mitigating the effects of the time-varying approximation errors. The proposed method, Adaptive Ensemble Q-learning (AdaEQ), has been shown to improve learning performance in the MuJoCo benchmark [21]. Secondly, Pan et al. [30] investigated the use of the Boltzmann softmax operator in updating value functions in actor-critic methods and showed that it has several advantages that make it preferable. They provided a new analysis indicating that the error between the value function under the softmax operator and the optimal can be bounded. Using this finding, they incorporated the softmax operator into the actor-critic setting to form the Softmax Deep Deterministic Policy Gradient (SD2) algorithm, which has been shown to reduce the overestimation bias and improve the Deep Deterministic Policy Gradient (DDPG) [31] algorithm. Next, they extended their technique to double Q-value estimators, i.e., TD3 [7]. They proposed the Softmax Deep Double Deterministic Policy Gradient (SD3) algorithm, which has been demonstrated to produce more accurate value estimations than TD3 [7].

Several techniques have also been proposed for the linear combination of the Q-value estimates by approximate critics to compute the objective for Q-network update [16, 17]. Specifically, the Weighted Delayed Deep Deterministic Policy Gradient (WD3) algorithm [16] uses the linear combination of the minimum and average of two Q-networks, where the hyper-parameter β controls the underestimation. The Triplet-average Deep Deterministic Policy Gradient (TADD) algorithm [17] adopts the same approach yet includes the estimates of an additional Q-network. Lastly, Cicek et al. [28] extended the WD3 algorithm [16] by an adaptive β parameter computed using the reward of the terminal transition in each episode. As the final reward of an episode is not discounted, the Q-value estimate for the terminal transition corresponds to the reward achieved in the terminal step. Using this, the Adaptive-WD3 (AWD3) algorithm [28] updates the β value through the difference between the reward and Q-value estimate for the terminal transition.

In contrast to the studies by Wang et al. [29] and Pan et al. [30], we only use the existing Q-networks in the standard TD3 algorithm [7] to achieve accurate value estimates. While Wang et al. [29] proposed to overcome the function approximation error in ensemble-based methods, our emphasis is on double Q-value estimators. Second, the approach of Pan et al. [30] requires the initialization of an additional actor network, which crucially increases the computational complexity. The AWD3 algorithm [28] is also not generalizable since success or failure is defined on the terminal states only for *some* tasks [32]. Lastly, we note that WD3 and TADD are extensively reviewed along with our previous work, TCD3 [14], in later sections and, we use them in our empirical studies to compare our algorithm.

3 Background

The reinforcement learning paradigm considers an agent interacting with its environment to learn the optimal, reward-maximizing behavior. In this study, we consider the reinforcement learning setting represented by a fully observable, finite-horizon Markov decision process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, p, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively, $\mathcal{R}(s, a, s') \in \mathbb{R}$ is the reward function which can be deterministic or stochastic depending on the environment, $p(s'|s, a)$ is the transition dynamics, and γ is the constant discount factor. At each discrete time step t, the agent observes its state $s \in \mathcal{S}$ and chooses an action $a \in \mathcal{A}$ according to its policy π, which can be stochastic or deterministic. Then, based on its action decision given the observed state, the agent receives a reward $r \sim \mathcal{R}(s, a, s')$ and observes a next state s' such that $s' \sim p(\cdot|s, a)$. The objective of the agent is to maximize the cumulative reward defined as the discounted sum of future rewards $R_t = \sum_{i=t}^{T} \gamma^{i-t} r_i$, where T is the index of the terminal time step, $r_i \sim \mathcal{R}(s_i, a_i, s_{i+1})$. The discount factor $\gamma \in [0, 1)$ downscales the long-term rewards to prioritize the short-term rewards more.

The agent learns the optimal policy π^* that maximizes the expected return $\mathbb{E}_{s_i \sim p_\pi, a_i \sim \pi}[R_0]$. In actor-critic settings where the action space is continuous, parameterized policies π_ϕ represented by deep neural networks with parameters ϕ are optimized by computing the gradient of the expected return $\nabla_\phi J(\phi)$ through a policy gradient technique. In this study, we consider the deterministic policy gradient algorithm expressed by:

$$\nabla_\phi J(\phi) = \mathbb{E}_{s \sim p_\pi}[\nabla_a Q^\pi(s, a)|_{a=\pi(s)} \nabla_\phi \pi_\phi(s)], \tag{1}$$

where p_π denotes the distribution over the visited states. The expected return after taking the action a given the observed state s under the current policy π is computed by the critic (action-value function or Q-function) $Q^\pi(s, a) = \mathbb{E}_{s \sim p_\pi, a \sim \pi}[R_t|s, a]$ which values the quality of the action decision given the observed state while following the current policy π. The critic evaluates and improves the agent's policy such that it chooses the actions that yield higher future rewards.

In Q-learning [15], the action-value function Q^π is estimated through recursive Bellman optimization [33] given the transition tuple (s, a, r, s'):

$$Q(s, a) \leftarrow Q(s, a) + \lambda \cdot \left[r + \gamma \mathbb{E}_{s', a'}[Q(s', a')] - Q(s, a)\right]; \quad a' \sim \pi(s'), \tag{2}$$

where λ is the learning rate. For large state and action spaces, the action-value function is usually estimated by function approximators $Q_\theta(s, a)$ parameterized by θ, also known as the Q-networks. In the deep setting of Q-learning [15], the Q-network is updated through the temporal-difference learning [11] by a secondary frozen target network $Q_{\theta'}(s, a)$ to construct the objective for behavioral Q-network, also known as Deep Q-learning [19]:

$$y = r + \gamma Q_{\theta'}(s', a'); \quad a' \sim \pi_{\phi'}(s'), \tag{3}$$

where the subsequent action given the observed next state can be obtained from a separate target actor network $\pi_{\phi'}$ for actor-critic settings in continuous control. The target networks are either updated by a small proportion τ at each time step, i.e., $\theta' \leftarrow \tau\theta + (1 - \tau)\theta'$, called soft-update, or periodically to exactly match the behavioral networks called hard-update.

4 The Underestimation Bias in Deterministic Policy Gradients

4.1 An Informative Analysis on the Existing Approaches to the Underestimation Bias

We start by explaining the current approaches to the underestimation bias in the literature and emphasize specific points through remarks. Mainly, we investigate the WD3 [16] and TADD [17] algorithms and the theoretical background of these algorithms. These studies extend the Clipped Double Q-learning algorithm [7] by replacing the Q-networks' objective with a fixed linear combination, as discussed. Let us first consider the WD3 algorithm [16]. In the simplest terms, Q-networks are updated as follows:

$$y = r + \gamma \left(\beta \min_{i=1,2} Q_{\theta_i'}(s', \tilde{a}') + \frac{1-\beta}{2} \sum_{i=1}^{2} Q_{\theta_i'}(s', \tilde{a}') \right), \tag{4}$$

$$J(\theta_i) = \| y - Q_{\theta_i}(s, a) \|^2, \tag{5}$$

where $\tilde{a}'$ is the action chosen by the target policy in the next state s' perturbed by a zero-mean Gaussian noise, i.e., $\tilde{a}' = \pi_{\phi'}(s') + \mathcal{N}(0, \sigma)$, σ is the standard deviation of the perturbation noise, and $J(\theta_i)$ is the loss associated with critic Q_{θ_i}. Here, $\beta \in [0, 1]$ is a parameter that controls the underestimation since the minimum operator yields the underestimation of Q-values [7, 14]. Note that this additive exploratory noise does not alter the expected function approximation error by having a zero mean [7].

The TADD algorithm [17] adopts a similar approach through an additional third critic employed in Q-learning [15]. In addition, the last K parameters of the third critic are stored in a critic network buffer, which is used to construct the objective for the Q-networks:

$$y = r + \gamma \left(\beta \min_{i=1,2} Q_{\theta_i'}(s', \tilde{a}') + \frac{(1-\beta)}{K} \sum_{k} Q_{\theta_{3,k}'}(s', \tilde{a}') \right), \tag{6}$$

$$J(\theta_i) = \| y - Q_{\theta_i}(s, a) \|^2. \tag{7}$$

Taking the average of the last K parameters reduces the variance of the Q-value estimates [17]. In these studies, the errors by the employed Q-networks are represented by probability distributions, which is feasible as the employment of deep neural networks and bootstrapping in the Q-learning introduce noise in the action-value estimates [7, 8]. Based on such a probabilistic representation, these works make two assumptions based on the error distributions. First, it is stated that the error by each of the critics can be represented either by a zero-mean Gaussian or a zero-mean uniform distribution. Second, error distributions by the two critics are independent and identically distributed, as shown by Theorems 1 and 2 in [16] and by Theorem 1 in [17]. Formally, we express the made assumptions in [16, 17] as:

$$N_i \sim \mathcal{N}(0, \tilde{\sigma}), \quad Z_i \sim \text{uniform}[-\delta, \delta]; \tag{8}$$

$$\left(Q_{\theta_i}(s, a) - Q^*(s, a) \right) \sim N_i \vee Z_i : \tag{9}$$

$$\mathrm{P}(N_1 \cap N_2) = \mathrm{P}(N_1) P(N_2), \tag{10}$$

$$\mathrm{P}(Z_1 \cap Z_2) = \mathrm{P}(Z_1) P(Z_2), \tag{11}$$

for some parameters δ and $\tilde{\sigma}$, where Q^* denotes the actual Q-value of the state-action pair (s, a). However, the zero-mean assumption violates the existence of the estimation bias:

$$\mathbb{E}[Q_{\theta_i}(s, a) - Q^*(s, a)] = 0, \tag{12}$$

$$\mathbb{E}[Q_{\theta_i}(s, a)] = Q^*(s, a). \tag{13}$$

The latter equation is satisfied since $Q^*(s,a)$ is the fixed point of the Bellman operator $\mathcal{T}^{\pi^*}$ [33] under the optimal policy π^* [13]. Then, from (13), we infer that each $Q_{\theta_i}(s,a)$ is an unbiased estimator of $Q^*(s,a)$, which contradicts the existence of an estimation bias. Furthermore, the errors of the two critics cannot be entirely independent due to the employment of the opposite critic in learning the targets and the same replay buffer [7]. Therefore, assumptions made in the current approaches to the underestimation bias violate the nature of the Q-learning in off-policy and deterministic policy gradient [18] methods. Finally, we can conclude this section with the following remarks.

Remark 1 The estimation errors by the two critics in the Clipped Double Q-learning algorithm [7] cannot follow a zero-mean probability distribution. If so, then the existence of an estimation bias is violated.

Remark 2 The error distributions by the two critics in the Clipped Double Q-learning algorithm [7] are not independent due to the employment of the opposite critic in learning the targets and the use of the same replay buffer.

4.2 Derivation of the Closed-Form Expression for the Underestimation Bias

By representing the error distributions through Gaussian distributions with non-zero mean and considering the dependence of the Q-networks, i.e., following Remarks 1 and 2, we begin to derive a closed-form expression for the estimation bias in the Clipped Double Q-learning algorithm [7] in a realistic manner. We follow the Gaussian distribution representation for estimation errors throughout the paper in both our statistical analysis and constructing our algorithm. The Gaussian assumption is realistic as Q-networks are deterministic function approximators, that is, each state-action pair corresponds to a single estimation error value. Additionally, using deep neural networks introduces noise in the estimates, which corresponds to the variance of the Gaussian error distributions [32]. This is highlighted in Remark 3.

Remark 3 A practical assumption to represent the Q-value estimation error is to use Gaussian distributions, with the mean corresponding to the actual estimation error value and the standard deviation arising from using deep neural networks.

Fujimoto et al. [7] previously highlighted the presence and effects of overestimation in actor-critic settings through the gradient ascent in the policy updates. However, using the minimum operator to compensate for overestimating Q-values may result in underestimated action-value estimates. We begin by proving through basic assumptions and claims that the underestimation phenomenon exists in DPG [18] algorithms for environments with varying reinforcement signals. We follow the gradient ascent approach in [7] to show such underestimation.

In the TD3 algorithm [7], the policy is updated using the minimum value estimate by two approximate critics, Q_{θ_1} and Q_{θ_2}, parameterized by θ_1 and θ_2, respectively. Without loss of generality, we assume that both critics overestimate the action-values, and the policy is updated with respect to the first approximate critic $Q_{\theta_1}(s,a)$ through the DPG algorithm [18]. The assumption on the overestimation of both Q-networks is valid as the single Q-network in the Deep Deterministic Policy Gradient (DDPG) algorithm [31] already overestimates the Q-values, as shown by [7]. First, let ϕ_{approx} define the parameters from the actor update by the maximization of the first approximate critic $Q_{\theta_1}(s,a)$:

$$\phi_{\text{approx}} = \phi + \frac{\eta}{Z_1}\mathbb{E}_{s\sim p_\pi}[\nabla_\phi \pi_\phi(s)\nabla_a Q_{\theta_1}(s,a)|_{a=\pi_\phi(s)}], \tag{14}$$

where Z_1 is the gradient normalizing term such that $Z^{-1}\|\mathbb{E}[\cdot]\| = 1$, and $\eta > 0$ is the learning rate. As the actor is optimized with respect to $Q_{\theta_1}(s, a)$ and the gradient direction is a local maximizer, there exists ζ sufficiently small such that if $\eta < \zeta$, then the *approximate* value of the policy, π_{approx}, by the first critic will be bounded below by the *approximate* value of the policy by the second critic:

$$\mathbb{E}[Q_{\theta_1}(s, \pi_{\text{approx}}(s))] \geq \mathbb{E}[Q_{\theta_2}(s, \pi_{\text{approx}}(s))]. \tag{15}$$

Note that for the latter equation, there could be a local maximizer for which $\mathbb{E}[Q_{\theta_2}(s, \pi_{\text{approx}}(s))] \geq \mathbb{E}[Q_{\theta_1}(s, \pi_{\text{approx}}(s))]$. However, such a possibility can be neglected in actor-critic algorithms that utilize Clipped Double Q-learning [7] since the actor is always optimized with respect to the first critic Q_{θ_1} [7]. Then, we can treat the function approximation error for both critics as distinct Gaussian random variables:

$$\begin{aligned} Q_{\theta_1}(s, a) - Q^*(s, a) &= N_1 \sim \mathcal{N}(\mu_1, \sigma_1), \\ Q_{\theta_2}(s, a) - Q^*(s, a) &= N_2 \sim \mathcal{N}(\mu_2, \sigma_2). \end{aligned} \tag{16}$$

Following (15) and Remark 1, we have $\mu_1 \geq \mu_2 \geq 0$. As the same experience replay buffer [34] and opposite critics are used in learning the target Q-values and critics, error Gaussian's denoted by (16) are not entirely independent according to Remark 2. Through the first moment of the minimum of two correlated Gaussian random variables [35], the expected estimation error for the Clipped Double Q-Learning algorithm [7] becomes:

$$\mathbb{E}[\min_{i=1,2}\{N_i\}] = \mu_2 + (\mu_1 - \mu_2)\Phi(\frac{\mu_1 - \mu_2}{\theta}) - \theta\psi(\frac{\mu_1 - \mu_2}{\theta}), \tag{17}$$

where $\theta := \sqrt{\sigma_1^2 + \sigma_2^2 - 2\rho\sigma_1\sigma_2}$, ρ is the correlation coefficient between N_1 and N_2, and $\Phi(\cdot)$ and $\psi(\cdot)$ are the cumulative distribution function (CDF) and probability density function (PDF) of the standard normal distribution, respectively. Due to the delayed actor updates, the mean function approximation errors by both critics are not very distant due to the decoupled actor and first critic. Since the policy updates are delayed, we can assume $\mu_1 \approx \mu_2$, as extensively used in the reinforcement learning literature [14]. Using this, (17) reduces to:

$$\mathbb{E}[\min_{i=1,2}\{N_i\}] = \mu_1 - \frac{\theta}{\sqrt{2\pi}}, \tag{18}$$

since $\Phi(0) = 0.5, \psi(0) = 1/\sqrt{2\pi}$. Hence, if $\sigma_1, \sigma_2 > \sqrt{\pi/(1-\rho)}\mu_1$, then the action-value estimate will be underestimated:

$$\mathbb{E}[\min_{i=1,2}\{Q_{\theta_i}(s, a)\} - Q^*(s, a)] < 0. \tag{19}$$

From $\sigma_1, \sigma_2 > \sqrt{\pi/(1-\rho)}\mu_1$ condition, if the pair of critics are highly correlated, underestimation does not exist. However, a moderate correlation exists between the pair of critics due to the delayed policy updates, which increases the underestimation possibility [7].

Although the improvements by Fujimoto et al. [7] aim to reduce the estimation error growth, the variance of the Q-values cannot be eliminated as they adhere to the variance of the future value estimates and rewards [7]. Furthermore, the Bellman equation [33] in function approximation settings cannot be exactly satisfied [7], which results in erroneous Q-value estimates as a function of the actual TD error expressed by (16). Then, we can show that the variance of the value estimates increases as the agent receives reward signals that vary on a large scale due to the exploration [32]. As shown in [7], the Q-value estimates can

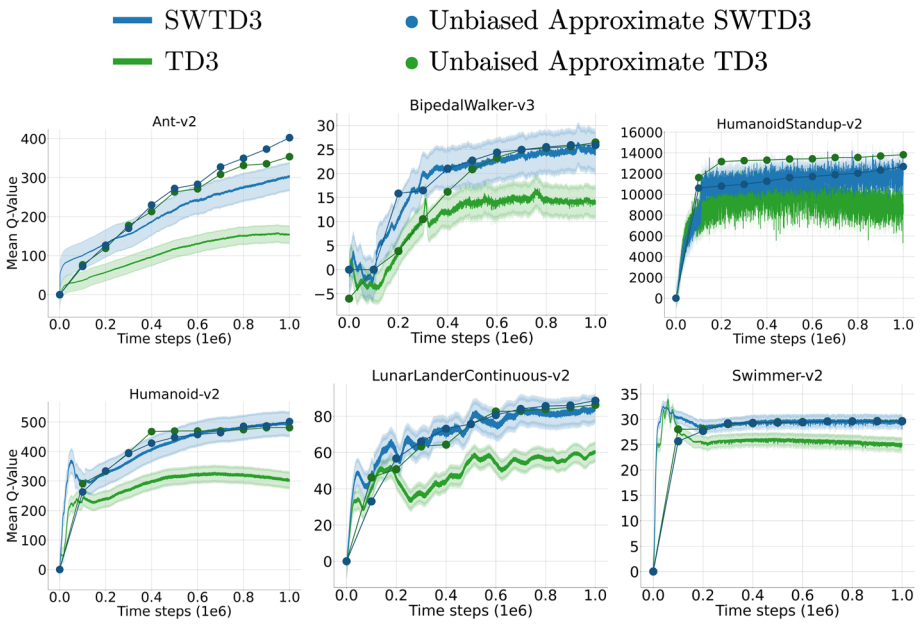

Fig. 1 Measuring estimation bias of fine-tuned TD3 versus SWTD3 while learning on MuJoCo and Box2D environments over 1 million time steps. Estimated and unbiased approximate Q-values are computed through Monte Carlo simulation for 1000 samples

be expressed in terms of the expected sum of discounted future rewards:

$$Q_{\theta_i}(s,a) = \mathbb{E}_{s_i \sim p_\pi, a_i \sim \pi}[\sum_{i=t}^{T} \gamma^{i-t} r_i] + \mu_i \sum_{i=t}^{T} \gamma^{i-t}. \tag{20}$$

If the expected estimation errors by both critics are constant, varying reinforcement signals increase the variance of the Q-value estimates resulting in an increasing underestimation bias. Since extensive exploration is a mandatory requirement for continuous action spaces [32], the variance of the reinforcement signals usually increases throughout the learning phase. Therefore, the underestimation bias on the value estimates becomes unavoidable. Moreover, the estimation error is not accumulated in the underestimation case due to the TD learning [8, 11]. Thus, the underestimation bias is far preferable to the overestimated Q-values in the actor-critic setting [7]. Nevertheless, underestimated action-values may discourage agents from choosing good state-action pairs for an extended period and reinforce agents to value suboptimal state-action pairs more frequently [14].

Remark 4 A varying set of reinforcement signals increase the variance of the Q-value estimates, which results in an increasing underestimation bias.

We can show the existence of the underestimation bias in practice by comparing the unbiased and estimated approximate Q-values while an agent under the TD3 algorithm [7] is learning on a set of OpenAI Gym [20] continuous control tasks over a training duration of 1 million time steps. The simulation results are reported in Fig. 1. We randomly select 1000 state-action pairs at every step and obtain the estimated Q-values by the first Q-network. The unbiased approximate Q-values are obtained at every 100,000 time steps by computing

the discounted sum of rewards starting from a randomly sampled 1000 states following the current policy. The Monte-Carlo simulation [36] is used over the randomly selected states and state-action pairs to obtain the average unbiased and estimated approximate Q-values. Note that the label "Mean Q-value" in Fig. 1 refers to the mean deviation of the Monte Carlo samples.

From Fig. 1, we observe an apparent underestimation bias throughout the learning phase such that the estimated Q-values are smaller than the unbiased ones except for a small proportion of the initial time steps. The underestimation bias arises depending on the environment and either grows or settles to a fixed level. These simulation results verify our claims; the approximate critics overestimate the actual Q-values at the initial steps. However, when the agent starts exploring the environment and encounters varying rewards, the variance of the value estimates increases, and the underestimation bias starts growing. For BipedalWalker and LunarLanderContinuous, the underestimation bias becomes fixed after a duration. This is due to the span of the reward space. If the agent encounters a sufficiently large subspace at the beginning of the learning, the underestimation bias cannot become larger. However, suppose the agent does not receive a significantly large subspace. In that case, the underestimation bias keeps growing even with the delayed target and actor updates as in the rest of the environments. Although the continuous, multi-dimensional, and large state-action spaces contribute to the growth of error, the scale of the current RL benchmarks is still very small compared to real-world tasks [32]. Hence, the underestimation bias will be more detrimental and inevitable when larger-scaled tasks are introduced.

To overcome the shown underestimation bias, we start by deriving the expected error induced by the update rule in TCD3 [14] and reducing the number of Q-networks to two while obtaining the same expected error. Then, through an extensive analysis of the WD3 [16] and TADD [17] algorithms, we introduce our novel, hyper-parameter-free modification on the target Q-value update that can further reduce the underestimation bias while preventing the overestimation.

5 Algorithm

5.1 Methodology

To construct our algorithm to overcome the estimation error problem in continuous control deep RL, we start with an extensive statistical analysis for further comparison with the existing methods of WD3 [16], TADD [17], and our previous approach TCD3 [14]. For this, we first consider the Q-network update rule in our previous work, the TCD3 algorithm [14]:

$$y = r + \gamma \min\left(\max_{i=1,2}(Q_{\theta_i'}(s', \pi_{\phi'}(s'))), Q_{\theta_3'}(s', \pi_{\phi'}(s'))\right), \tag{21}$$

where we employed an additional third critic Q_{θ_3} with corresponding estimation error distribution $N_3 \sim \mathcal{N}(\mu_3, \sigma_3)$. As the first critic is used to optimize the policy and due to the randomness in transition sampling, the same probability distribution can represent the errors corresponding to the second and third critics, i.e., $N_3 \sim \mathcal{N}(\mu_2, \sigma_2)$. We previously showed that this update rule can upper- and lower-bound the Q-value estimates by taking the minimum of the maximum of the first two critics and the third critic. Now, let us derive the expected function approximation error induced by the Q-value target expressed by (21).

First, expand $\min(\max(N_1, N_2), N_3)$ in terms of the maximum of error Gaussian's:

$$\min(\max(N_1, N_2), N_3) = \frac{1}{2}\max(N_1, N_2) + \frac{1}{2}N_3 - \frac{1}{2}|\max(N_1, N_2) - N_3|. \tag{22}$$

It is not trivial to compute the expectation of the latter term in the right-hand side of (22). However, we can rewrite (22) in terms of the maximum of three correlated Gaussian's and use the derivation for its expectation for equal means case from [37]. For this purpose, let $N_{\max} = \max(\max(N_1, N_2), N_3) = \max(N_1, N_2, N_3)$. Then, the expected value of (22) can be expressed as:

$$\min(\max(N_1, N_2), N_3) + N_{\max} = \max(N_1, N_2) + N_3, \tag{23}$$

$$\mathbb{E}[\min(\max(N_1, N_2), N_3)] = \mathbb{E}[\max(N_1, N_2)] + \mathbb{E}[N_3] - \mathbb{E}[N_{\max}]. \tag{24}$$

Under the assumption made in Sect. 4 that $\mu_1 \approx \mu_2 = \mu_3$, let us define $\mu := \mu_1 = \mu_2 = \mu_3$. Now, we can directly import the special case for the expectation of the maximum of correlated Gaussian's from [37]. The equal means case states that if $N_i \sim \mathcal{N}(\mu, \sigma_i)$, then the expected value of the maximum of three Gaussian's can be expressed as:

$$\mathbb{E}[\max(N_1, N_2, N_3)] = \mu + \frac{1}{2\sqrt{2\pi}}(\theta_{1,2} + \theta_{1,3} + \theta_{2,3}), \tag{25}$$

where $\theta_{i,j} := \sqrt{\sigma_i^2 + \sigma_j^2 - 2\rho\sigma_i\sigma_j}$. Due to the same experience replay [34] used in updating the Q-networks and decoupled actor and the first critic, without loss of generality, we can further assume that $\theta := \theta_{1,2} = \theta_{1,3} = \theta_{2,3}$. Then, (25) reduces to:

$$\mathbb{E}[N_{\max}] = \mathbb{E}[\max(N_1, N_2, N_3)] = \mu + \frac{3\theta}{2\sqrt{2\pi}}. \tag{26}$$

Furthermore, using the exact distribution of $\mathbb{E}[\max(N_1, N_2)]$ from [35], similar to (17), we have:

$$\mathbb{E}[\max_{i=1,2}\{N_i\}] = \mu_2 + (\mu_1 - \mu_2)\Phi(\frac{\mu_1 - \mu_2}{\theta}) + \theta\psi(\frac{\mu_1 - \mu_2}{\theta}). \tag{27}$$

Using the assumptions made, we can simplify (27) into:

$$\mathbb{E}[\max(N_1, N_2)] = \mu + \frac{\theta}{\sqrt{2\pi}}. \tag{28}$$

Inserting (26), (28) and $\mathbb{E}[N_3] = \mu$ into (24), we derive:

$$\mathbb{E}[\min(\max(N_1, N_2), N_3)] = \mu - \frac{\theta}{2\sqrt{2\pi}}. \tag{29}$$

Replacing μ with μ_2, we can express the expected function approximation error for $\min(\max(Q_1, Q_2), Q_3)$ in terms of the expected error for the Clipped Double Q-learning [7] denoted by (17) as:

$$\mathbb{E}[\min(\max(N_1, N_2), N_3)] = \frac{\mathbb{E}[\min_{i=1,2}\{N_i\}] + \mu_2}{2}. \tag{30}$$

This expected estimation bias is slightly less than the average of the underestimation in TD3 [7] and overestimation in the DDPG algorithm [31]. As the variance of the value estimates by two correlated critics are greater than the expected function approximation error, (30) is

still an underestimation. We can further reduce this underestimation by replacing μ_2 with μ_1 in (30) as $\mu_1 \geq \mu_2 \geq 0$:

$$y = r + \frac{\gamma}{2}\left(\min_{i=1,2}(Q_{\theta_i'}(s', \pi_{\phi'}(s'))) + Q_{\theta_1'}(s', \pi_{\phi'}(s'))\right). \tag{31}$$

Note that although we assume $\mu := \mu_1 = \mu_2 = \mu_3$ to benefit from the special case for the expectation of maximum of correlated Gaussian's across (26)-(29), we use $\mu_1 \geq \mu_2$ to derive the latter equation. While the actor network is optimized with respect to Q_{θ_1}, there can be $\mu_2 \geq \mu_1$ for some states since Q_{θ_1} and Q_{θ_2} are not independent due to the use of the same experience replay buffer [7]. Nevertheless, this possibility remains for the minority of the encountered states since the actor is *always* optimized with respect to Q_{θ_1} [7]. Therefore, the relation $\mu_1 = \mu_2 + \kappa$ holds for $\kappa \geq 0$ in reality since we consider the expectation over the visited states. However, κ becomes smaller compared to μ_1 and μ_2 in the expectation when we account for the possibility of $\mu_2 \geq \mu_1$. Nonetheless, to obtain the same expected function approximation error of TCD3 but with two Q-networks, we approach (30) in a realistic manner by using the precise fact of $\mu_1 \geq \mu_2$ to derive (31). Observe that the expected value of (21) and (31) are the same. We eliminate the computational burden introduced by employing the third Q-network while attaining the same expected error. Hence, the computational complexity is reduced by 33%.

Let us show the expected error by the WD3 [16] and TADD [17] algorithms. Update rules in these methods were previously expressed in (4) and (6), respectively. Using the Gaussian error distributions in (16) and the expectation of the minimum of two correlated Gaussians in (18), the expected error of WD3 [16] is expressed as:

$$\begin{aligned}\mathbb{E}[\beta \min_{i=1,2} N_i + \frac{1-\beta}{2}\sum_{i=1}^{2}\mu_i] &= \beta\mu_1 - \beta\frac{\theta}{\sqrt{2\pi}} + \frac{1-\beta}{2}\sum_{i=1}^{2} N_i \\ &\approx \beta\mu - \beta\frac{\theta}{\sqrt{2\pi}} + (1-\beta)\mu, \\ &= \mu - \beta\frac{\theta}{\sqrt{2\pi}}.\end{aligned} \tag{32}$$

Note that (32) is satisfied as $\mu \approx \mu_1 \approx \mu_2 = \mu_3$. Similarly, TADD [17] yields the following expected error:

$$\begin{aligned}\mathbb{E}[\beta \min_{i=1,2} N_i + \frac{(1-\beta)}{K}\sum_{k=1}^{K} N_{3,k}] &= \beta\mu_1 - \beta\frac{\theta}{\sqrt{2\pi}} + \frac{1-\beta}{K}\sum_{k}\mu_{3,k}, \\ &\approx \beta\mu - \beta\frac{\theta}{\sqrt{2\pi}} + (1-\beta)\mu, \\ &= \mu - \beta\frac{\theta}{\sqrt{2\pi}},\end{aligned} \tag{33}$$

where again, the latter equations are satisfied by $\mu \approx \mu_1 \approx \mu_2 = \mu_3$. Essentially, from (32) and (33), we observe that the estimation bias in the WD3 [16] and TADD [17] algorithms are the same. Moreover, by (30), we can infer that the following equations hold:

$$\begin{aligned}\epsilon_{\text{TCD3}} &< \epsilon_{\text{WD3}} = \epsilon_{\text{TADD}} \quad \text{if} \quad \beta, < 0.5, \\ \epsilon_{\text{TCD3}} &\geq \epsilon_{\text{WD3}} = \epsilon_{\text{TADD}} \quad \text{if} \quad \beta \geq 0.5,\end{aligned} \tag{34}$$

where ϵ denotes the estimation bias. We highlight our theoretical findings in the following Remarks.

Remark 5 The Q-network update rule in the WD3 [16] and TADD [17] algorithms yield the same estimation bias.

Remark 6 If $\beta \geq 0.5$, the expected estimation bias induced by TCD3 [14] is larger than the bias induced by WD3 [16] and TADD [17], and vice versa.

Although the WD3 [16] and TADD [17] approaches violate Remarks 1 and 2, utilizing a β parameter enables the control of the underestimation bias. However, having a fixed β is a task-specific greedy approach that cannot prevent the increasing underestimation bias as the variance of the reward signals increases throughout the learning. To overcome such an issue, we uniformly sample the β parameter from an interval, the lower bound of which linearly decreases throughout the learning, consistent with the increasing variance of the reinforcement signals. To specify the upper and lower bounds for such sampling interval, we leverage the findings in our previous work [14]. In [14], we showed that the estimation error by the Triplet Critic Update remains an underestimation, the absolute value of which is significantly smaller than that of Clipped Double Q-learning [7]. Although the estimation bias is not completely eliminated, the existing yet significantly decreased underestimation could dramatically improve the performance since underestimation is more preferable to overestimation [7]. As our previous work [14] corresponds to $\beta = 0.5$ in (32) and (33), we set the upper and lower bound of the interval to 0.5 at the beginning of the learning. Then, we linearly decrease the lower bound so that the contribution of an increasing variance of the rewards is also decreased throughout the learning. As we do not know the exact values of μ_i and θ, we are sure that $\theta = 0$ yields overestimation, the final lower bound cannot be 0 but should be a small number, slightly larger than 0. For this, we set the final lower bound of the bias interval to a small number $\alpha = 0.05$. Formally, we obtain the β parameter as:

$$\beta'^{(0)} \leftarrow 0.5, \tag{35}$$

$$\beta^{(0)} \sim [\beta'^{(0)}, \beta'^{(0)}], \tag{36}$$

$$\beta^{(t)} \sim \text{uniform}[\beta'^{(t)}, \beta'^{(0)}], \tag{37}$$

$$\beta'^{(t+1)} \leftarrow \beta'^{(0)} - \frac{\beta'^{(0)} - \alpha}{T} \times (t+1), \tag{38}$$

where $\beta^{(t)}$ is the sampled β value at time step t, $\beta'^{(t)}$ is the lower bound of the sampling interval at time step t, and T is the number of total training iterations. One concern with this update rule is that, as the exact estimation error cannot be known in theory, it may result in overestimation for some time steps. In addition, estimation error accumulates through subsequent updates in which Q-values are overestimated [7]. Nevertheless, the accumulated error will be clipped once a β value that yields underestimation is sampled. Therefore, due to the randomness, the estimation error does not accumulate over a significant number of time steps throughout the learning, and the RL agents can tolerate such slightly overestimated Q-values [14].

This forms our parameter-free update rule, Stochastic Weighted Twin Critic Update. As a result, our modification offers accurate Q-value estimates without introducing hyperparameters and networks. We summarize our introduced approach in Algorithm 1 and the resulting algorithm built on the TD3 algorithm [7], Stochastic Weighted Twin Delayed Deep Deterministic Policy Gradient (SWTD3), in Algorithm 2.

Remark 7 Due to the decreased lower bound of the β sampling interval and hence the mean of the β distribution, the introduced Q-network update rule is not affected as much as when β is fixed.

Remark 8 The estimation error induced by Stochastic Weighted Twin Critic Update may result in overestimation for some training iterations, especially in the later stages of learning, since the lower bound of the β interval becomes very small. However, if a β value corresponding to the underestimation is sampled, the overestimation will be clipped. Hence, estimation error does not accumulate over a significant number of time steps in the SWTD3 algorithm throughout learning due to its stochastic nature.

Algorithm 1 Stochastic Weighted Twin Critic Update (SWT)

Require: $Q_{\theta'_1}, Q_{\theta'_2}, s', \tilde{a}, \beta'^{(0)}, \beta', t, T$

$\beta \sim \text{uniform}[\beta', \beta'^{(0)}]$

$y \leftarrow r + \gamma \left(\beta \min_{i=1,2} (Q_{\theta'_i}(s', \pi_{\phi'}(s')) + (1-\beta) Q_{\theta'_1}(s', \pi_{\phi'}(s')) \right)$

$\beta' \leftarrow \beta'^{(0)} - \frac{\beta'^{(0)} - \alpha}{T} \times (t+1)$

return y, β'

Algorithm 2 SWTD3

Initialize critic networks Q_{θ_1}, Q_{θ_2}, and actor network π_ϕ with randomly initialized parameters θ_1, θ_2, ϕ
Initialize target networks $\phi' \leftarrow \phi, \theta'_1 \leftarrow \theta_1, \theta'_2 \leftarrow \theta_2$
Initialize replay buffer $\mathcal{B}$
Initialize the lower bound of the β sampling interval $\beta'^{(0)} \leftarrow 0.5$
for $t = 1$ **to** T **do**
  Select action with exploration noise $a \sim \pi_\phi(s) + \mathcal{N}(0, \sigma)$, and observe reward r and new state s'
  Store transition tuple (s, a, r, s') in $\mathcal{B}$
  Sample mini-batch of K transitions (s, a, r, s') from $\mathcal{B}$
  $\tilde{a} \leftarrow \pi_{\phi'}(s') + \text{clip}(\mathcal{N}(0, \sigma), -c, c)$
  $y, \beta' \leftarrow \textbf{SWT}(Q_{\theta'_1}, Q_{\theta'_2}, s', \tilde{a}, \beta'^{(0)}, \beta', t, T)$
  Update critics $\theta_i \leftarrow \text{argmin}_{\theta_i} \sum (y - Q_{\theta_i}(s, a))^2 / K$
  if t mod d **then**
    Update ϕ by the deterministic policy gradient:
    $\nabla_\phi J(\phi) = \frac{1}{K} \sum \nabla_a Q_{\theta_1}(s, a)|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$
    Update target networks:
    $\theta'_i \leftarrow \tau\theta_i + (1-\tau)\theta'_i$
    $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$
  end if
end for

5.2 Algorithmic and Complexity Comparison with the Existing Strategies

We investigate how our method differs from the previously examined approaches to the underestimation bias. First, we derive our method by assuming positively biased Q-value estimators and dependence of the approximate critics, which are mandatory and realistic in

practice. These requirements were previously summarized in Remarks 1 and 2, respectively. Second, our method does not introduce any hyper-parameter to be tuned in contrast to the WD3 [16] and TADD [17] algorithms that require the β parameter to be tuned, which controls the underestimation.

As we explained previously, the TD3 [7] and WD3 [16] algorithms maintain two critics while TADD [17] trains three critics. Although the Q-network objective computation requires the estimation of target Q-networks, the behavioral Q-networks must be maintained as the soft or hard update is used to update the corresponding target networks. Moreover, the TADD algorithm [17] uses estimations of K target Q-networks in constructing the Q-network objective. Nevertheless, the time complexity of backpropagation through a network matches or is larger than the forward propagation. Hence, we consider the time complexity as the only number of backpropagated Q-networks. Therefore, SWTD3, TD3 [7] and WD3 [16] match in terms of the run time and are bounded by the time complexity of TADD [17]. The following Remarks are made to conclude this comparison.

Remark 9 Our method introduces an analytical solution to the underestimation bias for deterministic policy gradients by considering biased Q-value estimators and dependence of the Q-networks in Clipped Double Q-learning [7], contrasting with the WD3 [16] and TADD [17] studies.

Remark 10 Our method does not introduce any hyper-parameter to be optimized, in contrast to WD3 [16] and TADD [17], in which the underestimation control parameter β requires to be tuned for each continuous task.

Remark 11 The time complexity of the TD3 [7], WD3 [16], and SWTD3 algorithms match and are bounded by the time complexity of the TADD algorithm [17].

Lastly, the previously mentioned Weighted Q-learning algorithm (WQ-L) [25] employs an update rule similar to ours, that is, the weighted average of two value estimators is used to construct the Q-network target. However, WQ-L [25] uses a hypothetical approach based on the Kullback–Leibler divergence to determine the weight values. In contrast, we consider the increasing variance of reinforcement signals to uniformly sample the weight value from an interval, the lower bound of which linearly decreases throughout learning. Furthermore, a single Q-value estimator is updated per learning step in the WQ-L algorithm [25], chosen uniformly. This cannot be adapted to actor-critic algorithms for continuous control. First, the actor's performance is assessed under the first Q-network. Thus, the first Q-network should have credible knowledge about the action-values. Second, the first Q-network is updated through the learning target constructed by its Q-value estimates in conjunction with the second Q-network's estimates. For the first Q-network to gain reliable knowledge about action-values, estimates of the second Q-networks should also be reliable since they both form the Q-network target y. This cannot be achieved under the WQ-L algorithm for actor-critic methods since one Q-network is updated at a time, which would substantially slow the learning of the actor network. In summary, update rules in our approach and WQ-L have similar formulations, yet the selection of the weight parameter and the algorithms' operation domain are significantly different.

6 Experiments

We evaluate the performance of our estimation bias correction approach by first demonstrating the estimated and actual Q-values of SWTD3 versus TD3 [7], WD3 [16], and TADD [17].

Table 1 WD3 and TADD environment specific weight values

Environment	WD3	TADD
Ant-v2	0.75[a]	0.95[a]
BipedalWalker-v3	0.5[b]	0.5[b]
HalfCheetah-v2	0.45[a]	0.95[a]
Hopper-v2	0.50[a]	0.95[a]
HumanoidStandup-v2	0.30[b]	0.30[b]
Humanoid-v2	0.30[b]	0.30[b]
InvertedDoublePendulum-v2	0.75[a]	0.95[a]
InvertedPendulum-v2	0.75[a]	0.95[a]
LunarLanderContinuous-v2	0.45[b]	0.45[b]
Reacher-v2	0.15[a]	0.95[a]
Swimmer-v2	0.45[b]	0.20[a]
Walker2d-v2	0.45[a]	0.95[a]

[a] As given in the paper
[b] Fine-tuned

Then, we evaluate the learning performances of RL agents under the SWTD3, TD3 [7], WD3 [16], and TADD [17] algorithms in MuJoCo [21] and Box2D [22] continuous control tasks interfaced by OpenAI Gym[2] [20]. We also consider our previous work, TCD3 [14], in our comparative evaluations for discussion. For reproducibility and a fair evaluation procedure, we directly follow the same set of tasks from MuJoCo [21] and Box2D [22] without modifying the environment dynamics.

6.1 Implementation Details and Experimental Setup

To implement the TD3 algorithm [7], we use the author's GitHub repository.[3] The implementation of TD3 [7] is the fine-tuned version of the algorithm. This version of TD3 [7] differs from the one introduced in [7] such that the number of hidden units in all networks is reduced to 256, the batch size is increased from 100 to 256, learning rates for the behavioral actor and critic Adam optimizers [38] are decreased from 10^{-3} to 3×10^{-4}, and 25,000 time steps of pure exploratory policy is employed in all environments. Furthermore, we built our modification on the TD3 [7] implementation such that the target Q-value computation is replaced by Algorithm 1. To ensure stability over updates and for consistency with our theoretical approach, the actor in SWTD3 is always optimized with respect to the first critic, as in the TD3 [7] and TCD3 [14] algorithms.

To implement the baseline algorithms, WD3 [16] and TADD [17], we use the TD3 algorithm's repository. We follow the same parameter, network, and Q-value update structures in [16] and [17] such that we replace the target Q-value computation and initialize an additional Q-network if required. For the pre-defined weight parameter β, we use the values for the environments presented in the respective papers. We manually fine-tune the β value over a training duration of 1 million time steps for ten random seeds for the rest of the environments. The values with the highest average of the last ten evaluation return over ten random seeds are chosen to train WD3 [16] and TADD [17] algorithms. Table 1 presents the used environment-

[2] https://gym.openai.com/.

[3] https://github.com/sfujim/TD3.

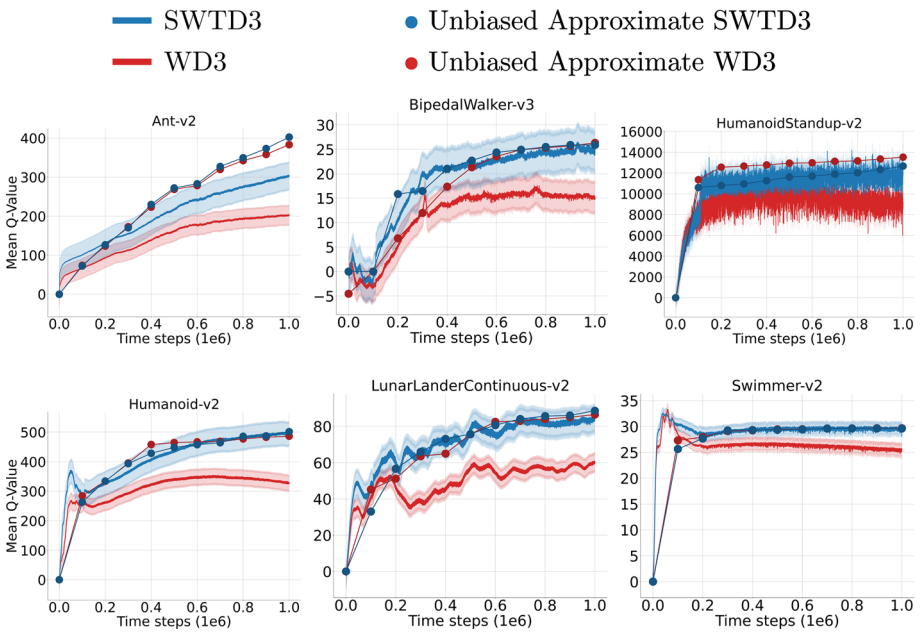

Fig. 2 Measuring estimation bias produced WD3 versus SWTD3 while learning on MuJoCo and Box2D environments over 1 million time steps. Estimated and unbiased approximate Q-values are computed through Monte Carlo simulation for 1000 samples

specific weight parameter β values for the WD3 [16] and TADD [17] algorithms. Values that we fine-tune and presented in [16, 17] are marked.

Each task in the Q-value comparisons is run for 1 million time steps, and curves are derived through the same procedure explained in Sect. 4. We perform evaluations on every task by running the algorithms over 1 million time steps and evaluating the agent's performance in a distinct evaluation environment without exploration noise and learning at every 1000 time steps. Each evaluation report is an average of ten episode rewards. The results are reported over ten random seeds of the Gym [20] simulator, network initialization, and code dependencies.

6.2 Discussion

6.2.1 Q-value Comparisons

Actual and estimated Q-value comparisons for our approach versus TD3 [7], WD3 [16], and TADD [17] over six OpenAI Gym [20] continuous control tasks are reported in Figs. 1, 2, and 3, respectively. In addition, we provide the Q-value estimation results for our approach versus the competing methods in a single plot in Fig. 4. SWTD3 obtains more accurate Q-value estimates than TD3 [7] and the baseline algorithms in all of the environments tested. Our empirical findings indicate several cases. First, we observe in the baseline Q-value estimations that the underestimation increases since the variance of the received reward signals grows throughout the learning, reflecting Remark 4. Second, although our method obtains fairly accurate Q-value estimates and is not affected by an increasing reward variance,

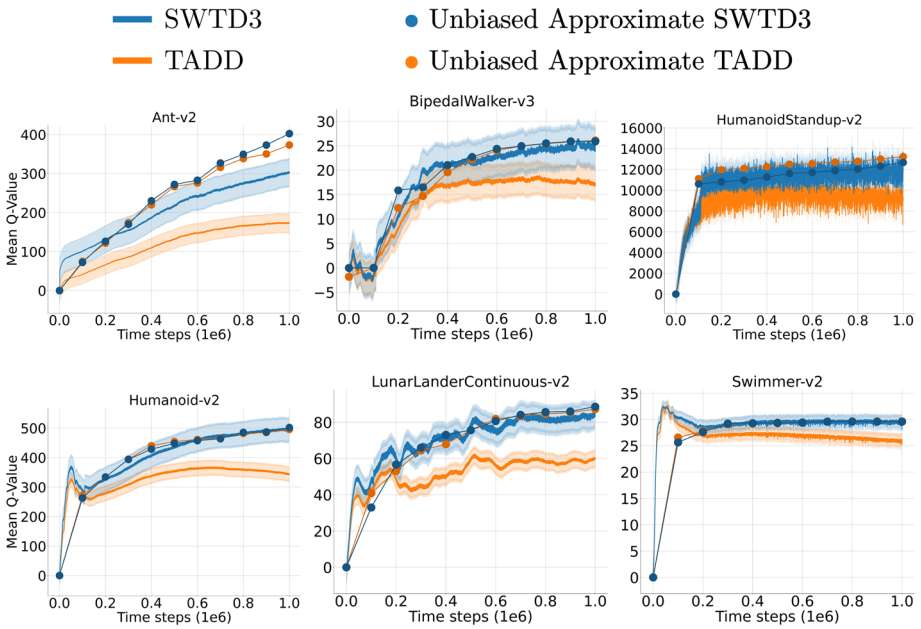

Fig. 3 Measuring estimation bias produced TADD versus SWTD3 while learning on MuJoCo and Box2D environments over 1 million time steps. Estimated and unbiased approximate Q-values are computed through Monte Carlo simulation for 1000 samples

the Q-values are overestimated in the initial steps. This is due to the large β values sampled at the beginning of the learning. However, as we discussed, such overestimated Q-values are tolerated by the agent, and the estimations reduce to a negligible margin of error, verifying our claim in Remark 4.

Furthermore, as stated previously, we fine-tune the β value for the environments that are not reported in [16, 17]. Our fine-tuning results show that the corresponding β values in these environments are the same for WD3 [16] and TADD [17] since the expected function approximation error is also the same, as highlighted in Remark 5. As a result, the mean estimation errors in these environments are practically the same for WD3 [16] and TADD [17], particularly, BipedalWalker, HumanoidStandup, Humanoid, and LunarLanderContinuous. For the environments that are reported in [16, 17], WD3 [16] obtains more accurate Q-value estimates than TADD [17] since $\beta = 0.95$ used in the TADD algorithm [17], which corresponds to a significant underestimation error due to the large contribution of the negative reward variance, as shown explicitly in (33). Our method attains substantially more accurate Q-value estimates than the competing approaches. It overcomes the effects induced by the increasing variance of the received reinforcement learning signals through sampling from an estimation error interval, the lower bound of which is constantly decreased, verifying Remark 7.

6.2.2 Evaluation

Table 2 reports the evaluation results in terms of the average of the last ten evaluation rewards over ten random seeds. Additionally, Fig. 5 depicts the corresponding learning curves. From our experimental results, we observe that our method either matches or outperforms the

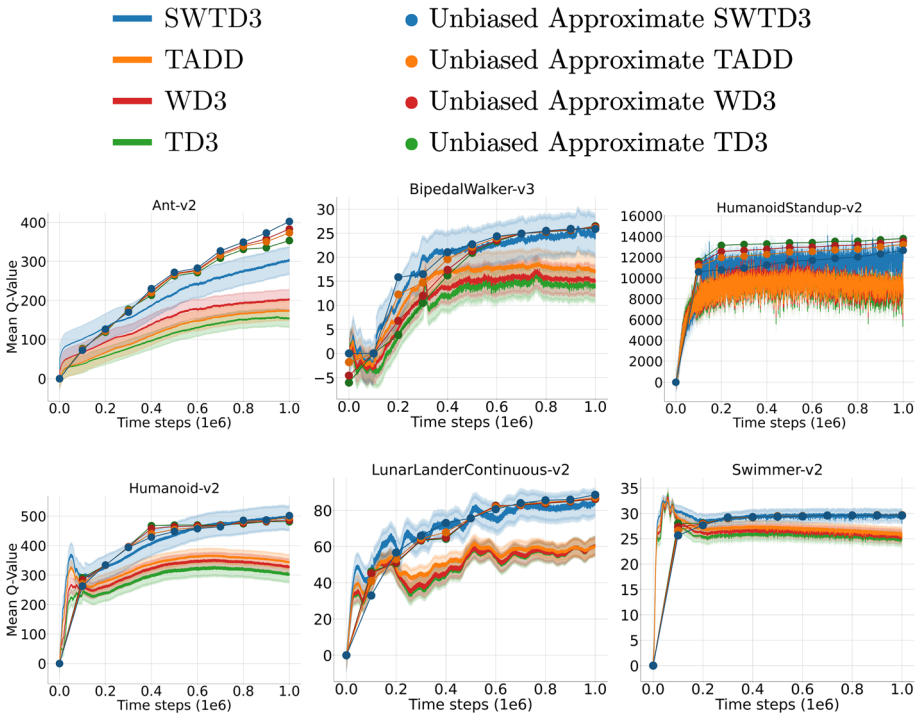

Fig. 4 Measuring estimation bias of WD3, TADD, and fine-tuned TD3 versus SWTD3 while learning on MuJoCo and Box2D environments over 1 million time steps. Estimated and unbiased approximate Q-values are computed through Monte Carlo simulation for 1000 samples

performance of TD3 [7] and baseline algorithms in terms of the learning speed and highest evaluation return. In the environments such as BipedalWalker, Humanoid, and LunarLanderContinuous, where our algorithm and competing approaches converge to the approximately same highest evaluation returns, Fig. 5 demonstrates that SWTD3 obtains a faster convergence by largely shrinking the underestimation bias and overcoming the increasing reward variance. Moreover, we do not observe a significant performance difference in trivial environments, e.g., InvertedDoublePendulum, InvertedPendulum, and Reacher, as they do not require complex solutions [39].

We observe that TCD3 [14], WD3 [16], and TADD [17] exhibit a better performance than TD3 [7]. However, in the environments reported by [17], where $\beta = 0.95$, the performance of TADD [17] is very similar to TD3 [7] as $\beta = 1.0$ corresponds to the same expected error in TD3 [7]. Furthermore, from our discussion in Remarks 5 and 6, and theoretical analysis in (30), (32), and (33), we infer that TCD3 [14], WD3 [16], and TADD [17] yield approximately the same performance for $\beta = 0.5$, which is depicted in the BipedalWalker environment. In addition, when the β value of WD3 [16] is smaller than that of TADD [17], it outperforms TADD [17] since a small fixed β value often corresponds to a decreased underestimation error. It exhibits the same performance in contrast when the β values are the same. Overall, these results are consistent with our Q-value comparisons and reflect the theoretical insights made in this study.

Ultimately, some methods exhibit a worse performance than what is outlined in the original articles. This is due to the stochasticity of the environment dynamics, that is, used dependen-

Table 2 Average of last 10 evaluation returns over 10 trials

Environment	SWTD3	TADD	WD3	TCD3	TD3
Ant-v2	**5216.25 ± 156.85**	3679.45 ± 279.48	4181.72 ± 69.7	4753.41 ± 294.3	3151.4 ± 259.28
BipedalWalker-v3	**309.42 ± 1.02**	297.95 ± 20.51	300.34 ± 8.93	307.27 ± 1.82	296.16 ± 32.75
HalfCheetah-v2	**10990.44 ± 95.26**	8651.46 ± 72.87	10170.53 ± 109.76	9961.2 ± 80.39	7574.84 ± 48.86
Hopper-v2	**3655.96 ± 7.03**	3138.8 ± 284.49	3142.01 ± 321.87	3187.83 ± 17.03	3275.41 ± 906.72
HumanoidStandup-v2	**149164.28 ± 10951.44**	120007.62 ± 7053.61	136939.36 ± 867.69	108915.83 ± 2534.29	93588.07 ± 10085.69
Humanoid-v2	**5269.39 ± 154.65**	4391.13 ± 470.96	4984.03 ± 122.46	4863.22 ± 266.93	5211.33 ± 129.84
InvertedDoublePendulum-v2	**9357.99 ± 0.71**	9350.29 ± 1.5	9353.49 ± 2.97	9334.67 ± 10.95	9350.29 ± 1.5
InvertedPendulum-v2	**1000.0 ± 0.0**	**1000.0 ± 0.0**	**1000.0 ± 0.0**	**1000.0 ± 0.0**	**1000.0 ± 0.0**
LunarLanderContinuous-v2	**277.11 ± 7.59**	276.88 ± 4.58	268.1 ± 7.78	272.77 ± 5.21	275.85 ± 8.21
Reacher-v2	**−3.53 ± 0.06**	−5.23 ± 0.03	−4.86 ± 0.05	−4.27 ± 0.03	−4.06 ± 0.01
Swimmer-v2	**145.54 ± 5.41**	144.52 ± 4.81	137.64 ± 1.8	127.69 ± 2.24	104.5 ± 1.46
Walker2d-v2	**4517.13 ± 287.1**	3525.91 ± 94.07	4117.14 ± 67.13	3886.36 ± 221.19	3492.81 ± 34.7

Boldface represents the maximum in each task. ± denotes the single standard deviation over trials. The WD3 and TADD algorithms use the beta values given in Table 1

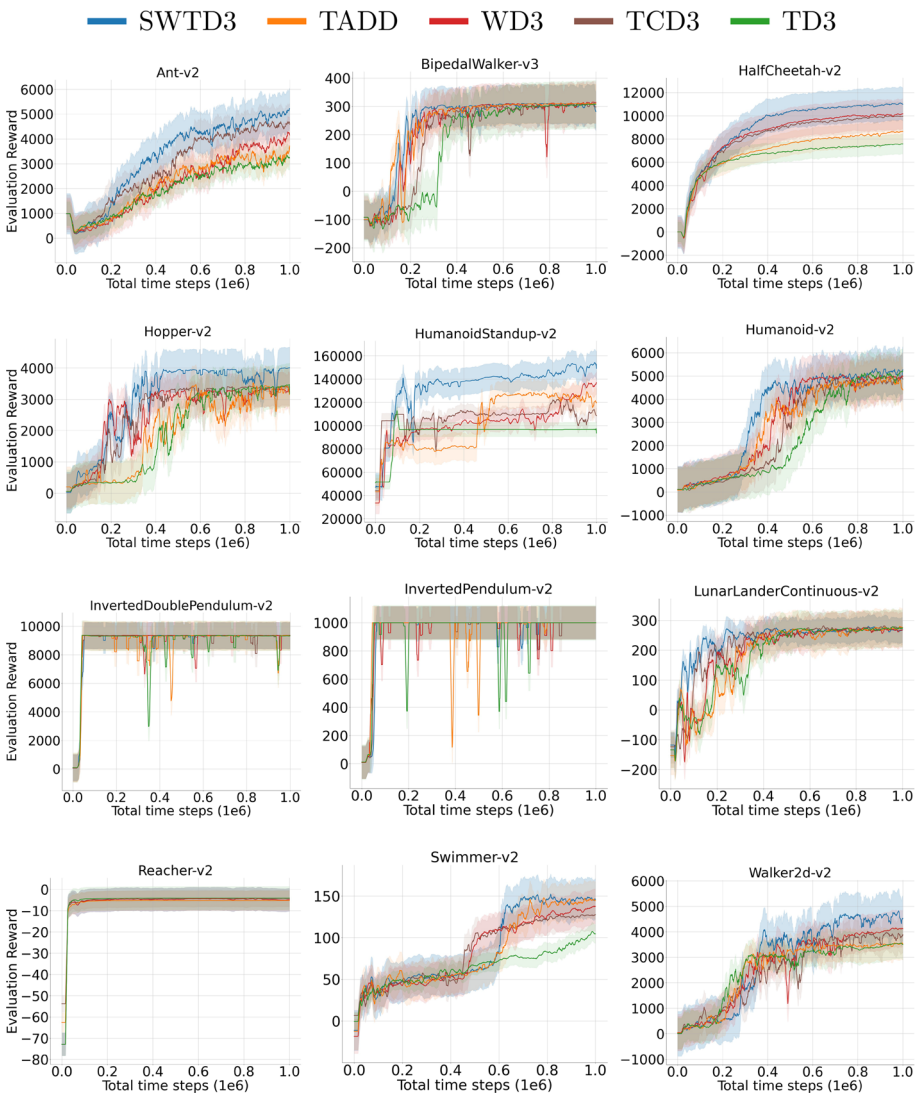

Fig. 5 Learning curves for the set of OpenAI Gym continuous control tasks. The shaded region represents half a standard deviation of the average evaluation over ten trials. Curves are smoothed uniformly with a sliding window of size 10

cies, hardware, and random seeds have a large effect on the performance of reinforcement learning algorithms [39]. Nevertheless, we use the same set of seeds for all algorithms in our experiments, and evaluation results would be consistent if we used different seeds, which suffices a fair evaluation procedure [39]. This is also valid for the resulting performances when the same β value is used for WD3 [16] and TADD [17]. The algorithmic differences alter the pseudorandom number order in the environment dynamics and cause the performances to differ slightly even under the same β value. Nonetheless, the overall performances are practically the same.

7 Conclusion

In this paper, we focus on the underestimation of the Q-values in deterministic policy gradient [18] methods. We extend our previous work on the underestimation by addressing the infeasible assumptions in the existing approaches that prevent them from adapting to off-policy actor-critic algorithms. We support our claims through Remarks and show that receiving different reward signals that vary on a large scale increases the underestimation of the action-value estimates. Then, through an extensive analysis of the estimation bias induced by the existing approaches, we introduce our novel Deep Q-learning [19] variant that forms a linear combination of two Q-value approximators with weights sampled from a shrunk estimation bias interval. Having our statistical analysis and extensive set of empirical studies combined, we demonstrate that the introduced approach notably outperforms the existing methods and improves our previous study. We also provide the exact implementation of the introduced algorithm at the GitHub repository[1] for reproducibility concerns.

In future work, a set of possible research directions could be: (i) The estimation bias interval from which the weights for the Q-value approximators are sampled can be shrunk in a non-linear manner, e.g., exponentially, instead of the linear decay presented in this work if further supported by theoretical analysis. Furthermore, a supervised learning method can also be employed to decrease the lower bound of the estimation bias interval by monitoring the learning progress of the agents. (ii) Lastly, a different, clipped probability distribution could be used instead of the uniform distribution to sample the weights of the Q-value approximators. Overall, we believe that our introduced Q-learning method is an essential step toward realizing unbiased value estimates in reinforcement learning. We also expect that the introduce approach would be used in other deterministic policy gradient-based reinforcement learning algorithms that will be introduced in the future.

Author Contributions All authors contributed to the study's conception and design. BS, FBM and DCC conducted an investigation and literature search. BS performed the statistical analysis and wrote the first draft of the manuscript. BS and SSK did the editing. SSK performed supervision. All authors read and approved the final manuscript.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data Availability Simulators used in this study are publicly available online: https://gym.openai.com/.

Code Availability The corresponding author's GitHub repository provides the source code for the study that led to all of the presented empirical findings: https://github.com/baturaysaglam/SWTD3.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Likas A, Blekas K (1996) A reinforcement learning approach based on the fuzzy min–max neural network. Neural Process Lett 4(3):167–172. https://doi.org/10.1007/BF00426025
2. Zhao J (2020) Neural network-based optimal tracking control of continuous-time uncertain nonlinear system via reinforcement learning. Neural Process Lett 51(3):2513–2530. https://doi.org/10.1007/s11063-020-10220-z
3. Yi M, Yang P, Du M et al (2022) DMADRL: a distributed multi-agent deep reinforcement learning algorithm for cognitive offloading in dynamic MEC networks. Neural Process Lett. https://doi.org/10.1007/s11063-022-10811-y
4. Ferguson A, Bolouri H (1996) Improving reinforcement learning in stochastic ram-based neural networks. Neural Process Lett 3(1):11–15. https://doi.org/10.1007/BF00417784
5. Zheng L, Cho SY (2011) A modified memory-based reinforcement learning method for solving POMDP problems. Neural Process Lett 33(2):187–200. https://doi.org/10.1007/s11063-011-9172-2
6. Ren L, Zhang G, Mu C (2019) Optimal output feedback control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning. Neural Process Lett 50(3):2963–2989. https://doi.org/10.1007/s11063-019-10072-2
7. Fujimoto S, van Hoof H, Meger D (2018) Addressing function approximation error in actor-critic methods. In: Dy J, Krause A (eds) Proceedings of the 35th international conference on machine learning, proceedings of machine learning research, vol 80. PMLR, Stockholmsmässan, Stockholm SWEDEN, pp 1587–1596. https://proceedings.mlr.press/v80/fujimoto18a.html
8. Sutton R (1988) Learning to predict by the method of temporal differences. Mach Learn 3:9–44. https://doi.org/10.1007/BF00115009
9. Hasselt Hv, Guez A, Silver D (2016) Deep reinforcement learning with double q-learning. In: Proceedings of the thirtieth AAAI conference on artificial intelligence. AAAI Press, Phoenix, Arizona, AAAI'16, pp 2094–2100
10. Lan Q, Pan Y, Fyshe A, et al (2020) Maxmin q-learning: controlling the estimation bias of q-learning. In: International conference on learning representations. https://openreview.net/forum?id=Bkg0u3Etwr
11. Precup D, Sutton R, Dasgupta S (2001) Off-policy temporal-difference learning with function approximation. In: Proceedings of the 18th international conference on machine learning
12. Espeholt L, Soyer H, Munos R, et al (2018) IMPALA: scalable distributed deep-RL with importance weighted actor-learner architectures. In: Dy J, Krause A (eds) Proceedings of the 35th international conference on machine learning, proceedings of machine learning research, vol 80. PMLR, Stockholmsmässan, Stockholm SWEDEN, pp 1407–1416. https://proceedings.mlr.press/v80/espeholt18a.html
13. Munos R, Stepleton T, Harutyunyan A, et al (2016) Safe and efficient off-policy reinforcement learning. In: Lee D, Sugiyama M, Luxburg U, et al (eds) Advances in neural information processing systems, vol 29. Curran Associates, Inc., Centre Convencions Internacional Barcelona, Barcelona SPAIN. https://proceedings.neurips.cc/paper/2016/file/c3992e9a68c5ae12bd18488bc579b30d-Paper.pdf
14. Saglam B, Duran E, Cicek DC, et al (2021) Estimation error correction in deep reinforcement learning for deterministic actor-critic methods. In: 2021 IEEE 33rd international conference on tools with artificial intelligence (ICTAI), pp 137–144. https://doi.org/10.1109/ICTAI52525.2021.00027
15. Watkins CJCH, Dayan P (1992) Q-learning. Mach Learn 8(3):279–292. https://doi.org/10.1007/BF00992698
16. He Q, Hou X (2020) Wd3: taming the estimation bias in deep reinforcement learning. In: 2020 IEEE 32nd international conference on tools with artificial intelligence (ICTAI), pp 391–398. https://doi.org/10.1109/ICTAI50040.2020.00068
17. Wu D, Dong X, Shen J et al (2020) Reducing estimation bias via triplet-average deep deterministic policy gradient. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2019.2959129
18. Silver D, Lever G, Heess N, et al (2014) Deterministic policy gradient algorithms. In: 31st international conference on machine learning, ICML 2014 1
19. Mnih V, Kavukcuoglu K, Silver D et al (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533. https://doi.org/10.1038/nature14236
20. Brockman G, Cheung V, Pettersson L, et al (2016) Openai gym. CoRR. arXiv:abs/1606.01540.
21. Todorov E, Erez T, Tassa Y (2012) Mujoco: A physics engine for model-based control. In: 2012 IEEE/RSJ international conference on intelligent robots and systems, pp 5026–5033. https://doi.org/10.1109/IROS.2012.6386109
22. Parberry I (2013) Introduction to game physics with box 2D, 1st edn. CRC Press Inc, Boca Raton
23. Boyan JA (2002) Technical update: least-squares temporal difference learning. Mach Learn 49(2):233–246. https://doi.org/10.1023/A:1017936530646

24. Tesauro G (1992) Practical issues in temporal difference learning. Mach Learn 8(3):257–277. https://doi.org/10.1007/BF00992697
25. Zhang Z, Pan Z, Kochenderfer MJ (2017) Weighted double q-learning. In: Proceedings of the twenty-sixth international joint conference on artificial intelligence, IJCAI-17, pp 3455–3461. https://doi.org/10.24963/ijcai.2017/483
26. Schmitt S, Hessel M, Simonyan K (2020) Off-policy actor-critic with shared experience replay. In: III HD, Singh A (eds) Proceedings of the 37th international conference on machine learning, proceedings of machine learning research, vol 119. PMLR, Vienna, Austria, pp 8545–8554. https://proceedings.mlr.press/v119/schmitt20a.html
27. Petrik M, Scherrer B (2009) Biasing approximate dynamic programming with a lower discount factor. In: Koller D, Schuurmans D, Bengio Y, et al (eds) Advances in neural information processing systems, vol 21. Curran Associates, Inc., Vancouver, B.C., Canada.https://proceedings.neurips.cc/paper/2008/file/08c5433a60135c32e34f46a71175850c-Paper.pdf
28. Cicek DC, Duran E, Saglam B, et al (2021) Awd3: dynamic reduction of the estimation bias. In: 2021 IEEE 33rd international conference on tools with artificial intelligence (ICTAI), pp 775–779. https://doi.org/10.1109/ICTAI52525.2021.00123
29. Wang H, Lin S, Zhang J (2021) Adaptive ensemble q-learning: minimizing estimation bias via error feedback. In: Ranzato M, Beygelzimer A, Dauphin Y, et al (eds) Advances in neural information processing systems, vol 34. Curran Associates, Inc., pp 24,778–24,790. https://proceedings.neurips.cc/paper/2021/file/cfa45151ccad6bf11ea146ed563f2119-Paper.pdf
30. Pan L, Cai Q, Huang L (2020) Softmax deep double deterministic policy gradients. In: Larochelle H, Ranzato M, Hadsell R, et al (eds) Advances in neural information processing systems, vol 33. Curran Associates, Inc., pp 11,767–11,777. https://proceedings.neurips.cc/paper/2020/file/884d247c6f65a96a7da4d1105d584ddd-Paper.pdf
31. Lillicrap TP, Hunt JJ, Pritzel A, et al (2016) Continuous control with deep reinforcement learning. In: ICLR (Poster). arxiv:1509.02971
32. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. A Bradford Book, Cambridge
33. Bellman RE (2003) Dynamic programming. Dover Publications Inc, Mineola
34. Lin LJ (1992) Self-improving reactive agents based on reinforcement learning, planning and teaching. Mach Learn 8(3):293–321. https://doi.org/10.1007/BF00992699
35. Nadarajah S, Kotz S (2008) Exact distribution of the max/min of two gaussian random variables. IEEE Trans Very Large Scale Integr (VLSI) Syst 16(2):210–212. https://doi.org/10.1109/TVLSI.2007.912191
36. Raychaudhuri S (2008) Introduction to Monte Carlo simulation. In: 2008 Winter simulation conference, pp 91–100. https://doi.org/10.1109/WSC.2008.4736059
37. Afonja B (1972) The moments of the maximum of correlated normal and t-variates. J R Stat Soc Ser B (Methodol) 34(2):251–262
38. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: ICLR (Poster). arxiv:1412.6980
39. Henderson P, Islam R, Bachman P, et al (2018) Deep reinforcement learning that matters. In: Proceedings of the thirty-second AAAI conference on artificial intelligence and thirtieth innovative applications of artificial intelligence conference and eighth AAAI symposium on educational advances in artificial intelligence. AAAI Press, New Orleans, Louisiana, USA, AAAI'18/IAAI'18/EAAI'18