Controlling Overestimation Bias with Truncated Mixture of Continuous Distributional Quantile Critics

Arsenii Kuznetsov [1] **Pavel Shvechikov** [1 2] **Alexander Grishin** [1 3] **Dmitry Vetrov** [1 3]

Abstract

The overestimation bias is one of the major impediments to accurate off-policy learning. This paper investigates a novel way to alleviate the overestimation bias in a continuous control setting. Our method—Truncated Quantile Critics, TQC,—blends three ideas: distributional representation of a critic, truncation of critics prediction, and ensembling of multiple critics. Distributional representation and truncation allow for arbitrary granular overestimation control, while ensembling provides additional score improvements. TQC outperforms the current state of the art on all environments from the continuous control benchmark suite, demonstrating 25% improvement on the most challenging Humanoid environment.

1. Introduction

Sample efficient off-policy reinforcement learning demands accurate approximation of the Q-function. Quality of approximation is key for stability and performance, since it is the cornerstone for temporal difference target computation, and action selection in value-based methods (Mnih et al., 2013), or policy optimization in continuous actor-critic settings (Haarnoja et al., 2018a; Fujimoto et al., 2018b).

In continuous domains, policy optimization relies on gradients of the Q-function approximation, sensing and exploiting erroneous positive biases. Recently, Fujimoto et al. (2018b) significantly improved the performance of a continuous policy by introducing a novel way to alleviate the overestimation bias (Thrun & Schwartz, 1993). We continue this line of research and propose an alternative highly competitive method for controlling overestimation bias.

[1]Samsung AI center, Moscow, Russia [2]National Research University Higher School of Economics, Moscow, Russia [3]Samsung-HSE Laboratory, National Research University Higher School of Economics, Moscow, Russia. Correspondence to: Arsenii Kuznetsov <brickerino@gmail.com>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

Thrun & Schwartz (1993) elucidate the overestimation as a consequence of Jensen's inequality: the maximum of the Q-function over actions is not greater than the expected maximum of noisy (approximate) Q-function. Specifically, for any action-dependent random noise $U(a)$ such that $\forall a \;\; \mathbb{E}_U\left[\, U(a) \,\right] = 0$,

$$\begin{aligned} \max_a Q(s,a) &= \max_a \mathbb{E}_U\left[\, Q(s,a) + U(a) \,\right] \\ &\leq \mathbb{E}_U\left[\, \max_a \{Q(s,a) + U(a)\} \,\right]. \end{aligned} \quad (1)$$

In practice, the noise $U(a)$ may arise for various reasons and from various sources, such as spontaneous errors in function approximation, Q-function invalidation due to ongoing policy optimization, stochasticity of environment, etc. Off-policy algorithms grounded in temporal difference learning are especially sensitive to approximation errors since errors are propagated backward through episodic time and accumulate over the learning process.

The de facto standard for alleviating overestimations in discrete control is the double estimator (Van Hasselt, 2010; 2013). However, Fujimoto et al. (2018b) argue that for continuous control this estimator may still overestimate in highly variable state-action space regions, and propose to promote underestimation by taking the minimum over two separate approximators. These approximators constitute naturally an ensemble, the size of which controls the intensity of underestimation: more approximators correspond to more severe underestimation (Lan et al., 2020). We argue, that this approach, while very successful in practice, has a few shortcomings:

- The overestimation control is coarse: it is impossible to take the minimum over a fractional number of approximators (see Section 4.1).
- The aggregation with $\min$ is wasteful: it ignores all estimates except the minimal one, diminishing the power of the ensemble of approximators.

We address these shortcomings with a novel method called Truncated Quantile Critics (TQC). In the design of TQC, we draw on three ideas: distributional representation of a critic, truncation of approximated distribution, and ensembling.

Distributional representations The distributional perspective (Bellemare et al., 2017) advocates the modeling of the *distribution* of the random return, instead of the more common modeling of the Q-function, the expectation of the return. In our work, we adapt QR-DQN (Dabney et al., 2018b) for continuous control and approximate the quantiles of the return distribution conditioned on the state and action. Distributional perspective allows for learning the intrinsic randomness of the environment and policy, also called aleatoric uncertainty. We are not aware of any prior work employing aleatoric uncertainty for overestimation bias control. We argue that the granularity of distributional representation is especially useful for precise overestimation control.

Truncation To control the overestimation, we propose to truncate the right tail of the return distribution approximation by dropping several of the topmost atoms. By varying the number of dropped atoms, we can balance between over- and underestimation. In a sense, the truncation operator is parsimonious: we drop only a small number of atoms (typically, around 8% of the total number of atoms). Additionally, truncation does not require multiple separate approximators: our method surpasses the current state of the art (which uses multiple approximators) on some benchmarks even using only a single one (Figure 1).

Ensembling The core operation of our method—truncation of return distribution—does not impose any restrictions on the number of required approximators. This effectively decouples overestimation control from ensembling, which, in turn, provides for additional performance improvement (Figure 1).

Our method improves the performance on all environments in the OpenAI gym (Brockman et al., 2016) benchmark suite powered by MuJoCo (Todorov et al., 2012), with up to 30% improvement on some of the environments. For the most challenging Humanoid environment this improvement translates into twice the running speed of the previous SOTA. The price to pay is the computational overhead carried by distributional representations and ensembling (Section 5.3).

This work makes the following contributions to the field of continuous control:

1. We design a practical method for the fine-grained control over the overestimation bias, called Truncated Quantile Critics (Section 3). For the first time, we (1) incorporate aleatoric uncertainty into the overestimation bias control, (2) decouple overestimation control and multiplicity of approximators, (3) ensemble distributional approximators in a novel way.

2. We advance the state of the art on the standard continuous control benchmark suite (Section 4) and perform extensive ablation study (Section 5).

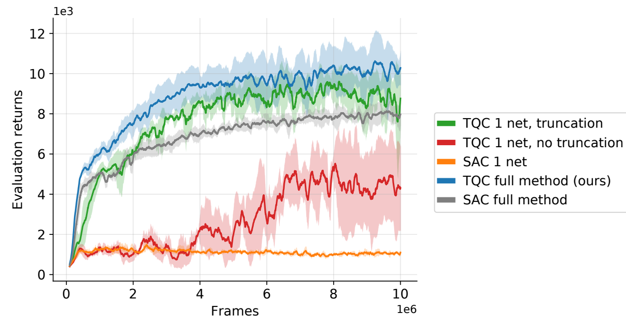


Figure 1. Evaluation on the Humanoid environment. Results are averaged over 4 seeds, ± std is shaded.

To facilitate reproducibility, we carefully document the experimental setup, perform exhaustive ablation, average experimental results over a large number of seeds, publish raw data of seed runs, and release the code for Tensorflow[1] and PyTorch[2].

2. Background

2.1. Notation

We consider a Markov decision process, MDP, defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma)$, with continuous state and action spaces $\mathcal{S}$ and $\mathcal{A}$, unknown state transition density $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, \infty)$, random variable reward function R, and discount factor $\gamma \in [0, 1)$. A policy π maps each state $s \in \mathcal{S}$ to a distribution over $\mathcal{A}$. We write $\mathcal{H}(\pi(s_t))$ to denote the entropy of the policy conditioned on the state s_t.

We write $\dim \mathcal{X}$ for the dimensionality of the space $\mathcal{X}$. Unless explicitly stated otherwise, the $\mathbb{E}_{\mathcal{D},\pi}[\cdot]$ signifies the expectation over the (s_t, a_t, r_t, s_{t+1}) from experience replay $\mathcal{D}$, and a_{t+1} from $\pi(\cdot|s_{t+1})$. We use the overlined notation to denote the parameters of target networks, i.e., $\overline{\psi}$ denotes the exponential moving average of parameters ψ.

2.2. Soft Actor Critic

The Soft Actor Critic (SAC) (Haarnoja et al., 2018a) is an off-policy actor-critic algorithm based on the maximum entropy framework. The objective encourages policy stochasticity by augmenting the reward with the entropy.

The policy parameters ϕ can be learned by minimizing the

$$J_\pi(\phi) = \mathbb{E}_{\mathcal{D},\pi}\left[\mathrm{D_{KL}}\left(\pi_\phi(\cdot|s_t) \middle\| \frac{\exp\left(\frac{1}{\alpha} Q_\psi(s_t, \cdot)\right)}{C_\psi(s_t)}\right)\right], \tag{2}$$

where Q_ψ is the soft Q-function and $C_\psi(s_t)$ is the normal-

[1] https://github.com/bayesgroup/tqc

[2] https://github.com/bayesgroup/tqc_pytorch

izing constant.

The soft Q-function parameters θ can be learned by minimizing the soft Bellman residual

$$J_Q(\psi) = \mathbb{E}_{\mathcal{D},\pi}\left[\frac{1}{2}\left(Q_\psi(s_t,a_t) - y(s_t,a_t)\right)^2\right], \quad (3)$$

where $y(s_t, a_t)$ denotes the temporal difference target

$$r(s_t,a_t) + \gamma\left[Q_{\overline{\psi}}(s_{t+1},a_{t+1}) - \alpha\log\pi_\phi(a_{t+1}|s_{t+1})\right], \quad (4)$$

and α is the entropy temperature coefficient. Haarnoja et al. (2018b) proposed to dynamically adjust the α by taking a gradient step with respect to the loss

$$J(\alpha) = \mathbb{E}_{\mathcal{D},\pi_\phi}\left[\log\alpha\cdot(-\log\pi_\phi(a_t|s_t) - \mathcal{H}_T)\right], \quad (5)$$

each time the π_ϕ changes. This decreases the α, if the stochastic estimate of policy entropy, $-\log\pi_\phi(a_t|s_t)$, is higher than $\mathcal{H}_T$, and increases α otherwise. The target entropy usually is set heuristically to $\mathcal{H}_T = -\dim\mathcal{A}$.

Haarnoja et al. (2018b) takes the minimum over two Q-function approximators to compute the target in equation 4 and policy objective in equation 2.

2.3. Distributional Reinforcement Learning with Quantile Regression

Distributional reinforcement learning focuses on approximating the return random variable $Z^\pi(s,a) := \sum_{t=0}^{\infty}\gamma^t R(s_t,a_t)$ where $s_0 = s$, $a_0 = a$ and $s_{t+1} \sim \mathcal{P}(\cdot|s_t,a_t)$, $a_t \sim \pi(\cdot|s_t)$, as opposed to approximating the expectation of the return, also known as the Q-function, $Q^\pi(s,a) := \mathbb{E}\left[Z^\pi(s,a)\right]$.

QR-DQN (Dabney et al., 2018b) approximates the distribution $Z^\pi(s,a)$ with $Z_\psi(s,a) := \frac{1}{M}\sum_{m=1}^{M}\delta(\theta_\psi^m(s,a))$, a mixture of atoms—Dirac delta functions at locations $\theta_\psi^1(s,a),\ldots,\theta_\psi^M(s,a)$ given by a parametric model $\theta_\psi : \mathcal{S}\times\mathcal{A}\to\mathbb{R}^M$.

Parameters ψ are optimized by minimizing the averaged over the replay 1-Wasserstein distance between Z_ψ and the temporal difference target distribution $\mathcal{T}_\pi Z_{\overline{\psi}}$, where $\mathcal{T}_\pi$ is the distributional Bellman operator (Bellemare et al., 2017):

$$\mathcal{T}_\pi Z(s,a) \stackrel{D}{:=} R(s,a) + \gamma Z(s',a'), \\ s' \sim \mathcal{P}(\cdot|s,a), a' \sim \pi(\cdot|s'). \quad (6)$$

As Dabney et al. (2018b) show, this minimization can be performed by learning quantile locations for fractions $\tau_m = \frac{2m-1}{2M}, m \in [1..M]$ via quantile regression. The quantile regression loss, defined for a quantile fraction $\tau\in[0,1]$, is

$$\mathcal{L}_{\mathrm{QR}}^{\tau}(\theta) := \mathbb{E}_{\tilde{Z}\sim Z}\left[\rho_\tau(\tilde{Z}-\theta)\right], \text{ where} \\ \rho_\tau(u) = u\left(\tau - \mathbb{I}(u<0)\right), \forall u\in\mathbb{R}. \quad (7)$$

To improve gradients for small u authors propose to use the Huber quantile loss (asymmetric Huber loss):

$$\rho_\tau^H(u) = |\tau - \mathbb{I}(u<0)|\mathcal{L}_H^1(u), \quad (8)$$

where $\mathcal{L}_H^1(u)$ is a Huber loss with parameter 1.

3. Truncated Quantile Critics, TQC

We start with an informal explanation of TQC and motivate our design choices. Next, we outline the formal procedure at the core of TQC, specify the loss functions and present an algorithm for practical implementation.

3.1. Overview

To achieve granularity in controlling the overestimation, we ”decompose” return into atoms of distributional representation. By varying the number of atoms, we can control the precision of the return distribution approximation.

To control the overestimation, we propose to truncate the approximation of the return distribution: we drop atoms with the largest locations and estimate the Q-value by averaging the locations of the remaining atoms. By varying the total number of atoms and the number of dropped ones, we can flexibly balance between under- and overestimation. The truncation naturally accounts for the inflated overestimation due to the high return variance: the higher the variance, the lower the Q-value estimate after truncation.

To improve the Q-value estimation, we ensemble multiple distributional approximators in the following way. First, we form a mixture of distributions predicted by N approximators. Second, we truncate this mixture by removing atoms with the largest locations and estimate the Q-value by averaging the locations of the remaining atoms. The order of operations—the truncation of the mixture vs. the mixture of truncated distributions—may matter. The truncation of a mixture removes the largest outliers from the pool of all predictions. Such a truncation may be useful in a hypothetical case of one of the critics goes crazy and overestimates much more than the others. In this case, the truncation of a mixture removes the atoms predicted by this inadequate critic. In contrast, the mixture of truncated distributions truncates all critics evenly.

Our method is different from previous approaches (Zhang & Yao, 2019; Dabney et al., 2018a) that distorted the critic's distribution at the policy optimization stage only. We use nontruncated critics' predictions for policy optimization. And truncate target return distribution at the value learning stage. Intuitively, this prevents errors from propagating to other states via TD learning updates and eases policy optimization.

Next, we present TQC formally and summarize the proce-

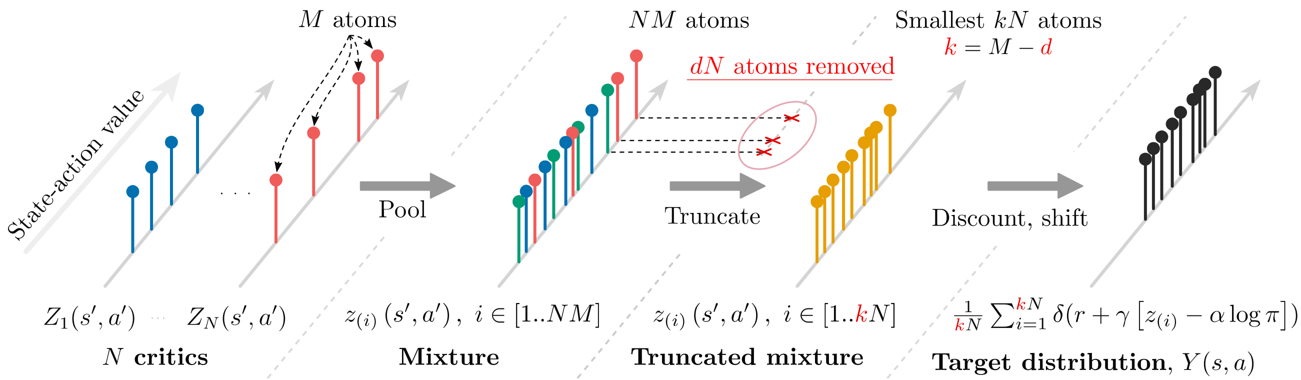


Figure 2. Step-by-step construction of the temporal difference target distribution $Y(s,a)$. First, we compute approximations of the return distribution conditioned on s' and a' by evaluating N separate target critics. Second, we make a mixture out of the N distributions from the previous step. Third, we truncate the right tail of this mixture to obtain atoms $z_{(i)}(s',a')$ from equation 11. Fourthly, we add entropy term, discount and add reward as in soft Bellman equation.

dure in Algorithm 1.

3.2. Computation of the target distribution

We propose to train N approximations $Z_{\psi_1}, \dots Z_{\psi_N}$ of the policy conditioned return distribution Z^π. Each Z_{ψ_n} maps each (s,a) to a probability distribution

$$Z_{\psi_n}(s,a) := \frac{1}{M}\sum_{m=1}^{M} \delta\left(\theta^m_{\psi_n}(s,a)\right), \tag{9}$$

supported on atoms $\theta^1_{\psi_n}(s,a), \dots, \theta^M_{\psi_n}(s,a)$.

We train approximations $Z_{\psi_1}, \dots Z_{\psi_N}$ on the temporal difference target distribution $Y(s,a)$. We construct it as follows. We pool atoms of distributions $Z_{\psi_1}(s',a'), \dots, Z_{\psi_N}(s',a')$ into a set

$$\mathcal{Z}(s',a') := \{\theta^m_{\psi_n}(s',a') \mid n \in [1..N], m \in [1..M]\} \tag{10}$$

and denote elements of $\mathcal{Z}(s',a')$ sorted in ascending order by $z_{(i)}(s',a')$, with $i \in [1..MN]$.

The kN smallest elements of $\mathcal{Z}(s',a')$ define atoms

$$y_i(s,a) := r(s,a) + \gamma[z_{(i)}(s',a') - \alpha \log \pi_\phi(a'|s')] \tag{11}$$

of the target distribution

$$Y(s,a) := \frac{1}{kN}\sum_{i=1}^{kN} \delta\left(y_i(s,a)\right). \tag{12}$$

In practice, we always populate $\mathcal{Z}(s',a')$ with atoms predicted by target networks $Z_{\overline{\psi}_1}(s',a'), \dots, Z_{\overline{\psi}_N}(s',a')$, which are more stable.

3.3. Loss functions

We minimize the 1-Wasserstein distance between each of $Z_{\psi_n}(s,a), n \in [1..N]$ and the temporal difference target distribution $Y(s,a)$. Equivalently (Dabney et al., 2018b), to minimize this distance we can approximate the quantiles of the target distribution, i.e., learn the locations for quantile fractions $\tau_m = \frac{2m-1}{2M}, m \in [1..M]$.

We approximate the $\tau_m, m \in [1..M]$ quantiles of $Y(s,a)$ with $\theta^1_{\psi_n}(s,a), \dots, \theta^M_{\psi_n}(s,a)$ by minimizing the loss

$$J_Z(\psi_n) = \mathbb{E}_{\mathcal{D},\pi}\left[\mathcal{L}^k(s_t, a_t; \psi_n)\right], \tag{13}$$

over the parameters ψ_n, where

$$\mathcal{L}^k(s,a;\psi_n) = \frac{1}{kNM}\sum_{m=1}^{M}\sum_{i=1}^{kN} \rho^H_{\tau_m}(y_i(s,a) - \theta^m_{\psi_n}(s,a)). \tag{14}$$

In this way, each learnable location $\theta^m_{\psi_n}(s,a)$ becomes dependent on all atoms of the truncated mixture of target distributions.

The policy parameters ϕ can be optimized to maximize the entropy penalized estimate of the Q-value by minimizing the loss

$$J_\pi(\phi) = \mathbb{E}_{\mathcal{D},\pi}\left[\alpha \log \pi_\phi(a|s) - \frac{1}{NM}\sum_{m,n=1}^{M,N} \theta^m_{\psi_n}(s,a)\right], \tag{15}$$

where $s \sim \mathcal{D}, a \sim \pi_\phi(\cdot|s)$. We use nontruncated estimate of the Q-value for policy optimization to avoid double truncation: Z-functions approximate already truncated future distribution.

4. Experiments

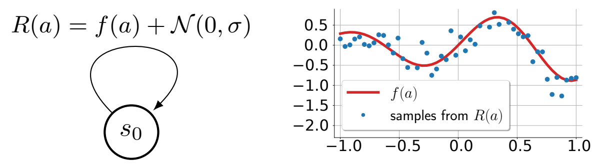
First, we compare our method with other possible ways to mitigate the overestimation bias on a simple MDP, for which we can compute the true Q-function and the optimal policy.

Second, we quantitatively compare our method with competitors on a standard continuous control benchmark – the

Algorithm 1 TQC. $\hat{\nabla}$ denotes the stochastic gradient

- Initialize policy π_ϕ, critics $Z_{\psi_n}, Z_{\overline{\psi}_n}$ for $n \in [1..N]$
- Set replay $\mathcal{D} = \varnothing$, $\mathcal{H}_T = -\dim \mathcal{A}$, $\alpha = 1$, $\beta = .005$

for each iteration **do**
  for each environment step, until done **do**
    collect transition (s_t, a_t, r_t, s_{t+1}) with policy π_ϕ
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r_t, s_{t+1})\}$
  end for
  for each gradient step **do**
    sample a batch from the replay $\mathcal{D}$
    $\alpha \leftarrow \alpha - \lambda_\alpha \hat{\nabla}_\alpha J(\alpha)$ Eq. (5)
    $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$ Eq. (15)
    $\psi_n \leftarrow \psi_n - \lambda_Z \hat{\nabla}_{\psi_n} J_Z(\psi_n)$, $n \in [1..N]$ Eq. (13)
    $\overline{\psi}_n \leftarrow \beta \psi_n + (1-\beta)\overline{\psi}_n$, $n \in [1..N]$
  end for
end for
return policy π_ϕ, critics Z_{ψ_n}, $n \in [1..N]$.



(a) Diagram (b) Reward function

Figure 3. Infinite horizon MDP with a single state and one-dimensional action space $[-1, 1]$. At each step agent receives a stochastic reward $R(a) \sim \mathcal{N}(f(a), \sigma)$ (see Appendix C for details).

Table 1. Bias correction methods. For simplicity, we omit the state s_0 from all arguments.

METHOD	CRITIC TARGET $r(a) + \gamma\langle \widehat{Q}$ OR $\widehat{Z}\rangle(a')$	POLICY OBJECTIVE
AVG	$\widehat{Q}(\cdot) = \frac{1}{N}\sum_{i=1}^{N} Q_i(\cdot)$	$\frac{1}{N}\sum_{i=1}^{N} Q_i(a)$
MIN	$\widehat{Q}(\cdot) = \min_i Q_i(\cdot)$	$\min_i Q_i(a)$
TQC	$\widehat{Z}(\cdot) = \frac{1}{kN}\sum_{i=1}^{kN} \delta\left(z_{(i)}(\cdot)\right)$	$\frac{1}{NM}\sum_{i=1}^{NM} z_{(i)}(a)$

set of MuJoCo (Todorov et al., 2012) environments implemented in OpenAI Gym (Brockman et al., 2016). The details of the experimental setup are in Appendix A.

Third, we quantify the overestimation bias of TQC in two MuJoCo environments.

We implement TQC on top of the SAC (Haarnoja et al., 2018b) with auto-tuning of the entropy temperature (Section 2.2). For all MuJoCo experiments, we use $N = 5$ critic networks with three hidden layers of 512 neurons each, $M = 25$ atoms, and the best number of dropped atoms per network $d \in [0..5]$, if not stated otherwise. The other hyperparameters are the same as in SAC (see Appendix B).

4.1. Single state MDP

In this experiment we evaluate bias correction techniques (Table 1) in a single state continuous action infinite horizon MDP (Figure 3). We train Q-networks (or Z-networks, depending on the method) with two hidden layers of size 50 from scratch on the replay buffer of size 50 for 3000 iterations, which is enough for all methods to converge. We populate the buffer by sampling a reward once for each action from a uniform action grid. At each step of temporal difference learning, we use a policy, which is greedy with respect to the objective in Table 1.

We define $\Delta(a) := \widehat{Q}^\pi(a) - Q^\pi(a)$ as a signed discrepancy between the approximate and the true Q-value. For TQC $\widehat{Q}^\pi(a) = \mathbb{E}\widehat{Z}^\pi(a) = \frac{1}{kN}\sum_{i=1}^{kN} z_{(i)}(a)$. We vary the parameters controlling the overestimation for each method and report the robust average (10% of each tail is truncated) over 100 seeds of $\mathbb{E}_{a\sim\mathcal{U}(-1,1)}\left[\Delta(a)\right]$ and $\mathbb{V}\mathrm{ar}_{a\sim\mathcal{U}(-1,1)}\left[\Delta(a)\right]$.

For AVG and MIN we vary the number of networks N, for TQC—the number of dropped quantiles per network $d = M - k$. We present the results in Figure 4 with bubbles of diameter, inversely proportional to the averaged over the seeds absolute distance between the optimal a^* and the $\arg\max$ of the policy objective.

The results (Figure 4) suggest TQC can achieve the lowest variance and the smallest bias of Q-function approximation among all the competitors. The variance and the bias correlate well with the policy performance, suggesting TQC may be useful in practice.

4.2. Comparative Evaluation

We compare our method with original implementations of state of the art algorithms: SAC[3], TrulyPPO[4], and TD3[5]. For HalfCheetah, Walker, and Ant we evaluate methods on the extended frame range: until all methods plateaus ($5 \cdot 10^6$ versus usual $3 \cdot 10^6$). For Hopper, we extended the range to $3 \cdot 10^6$ steps.

For our method we selected the number of dropped atoms d for each environment independently, based on separate evaluation. Best value for Hopper is $d = 5$, for HalfCheetah $d = 0$ and for the rest $d = 2$.

Figure 5 shows the learning curves. In Table 2 we report the average and std of 10 seeds. Each seed performance is an av-

[3] https://github.com/rail-berkeley/softlearning
[4] https://github.com/wangyuhuix/TrulyPPO
[5] https://github.com/sfujim/TD3

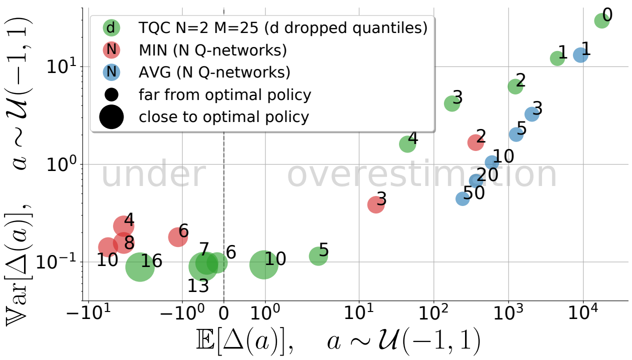


Figure 4. Robust average (10% of each tail is truncated) of bias and variance of Q-function approximation for different methods: TQC, MIN, AVG. See the text for the details about axis labels.

Table 2. Average and std of the seed returns (thousands). The best average return is bolded, and marked with $*$ if it is the best at level 0.05 according to the two-sided Welch's t-test with Bonferroni correction for multiple comparison testing.

ENV	TRULYPPO	TD3	SAC	TQC
HOP	1.98(.54)	3.31(.55)	2.86(.58)	**3.71(.16)**
HC	5.78(.62)	15.12(.59)	12.41(5.14)	**18.09(.34)**$*$
WAL	4.00(.50)	5.11(.52)	5.76(.46)	**7.03(.62)**$*$
ANT	−0.01(.00)	5.68(1.04)	6.16(.93)	**8.01(.87)**$*$
HUM	5.86(.45)	5.40(.36)	7.76(.46)	**9.54(1.18)**$*$

erage of 100 last evaluations. We evaluate the performance every 1000 frames as an average of 10 deterministic rollouts. As our results suggest, TQC performs consistently better than any of the competitors. TQC also improves upon the maximal published score on four out of five environments (Table 3).

4.3. Overestimation in MuJoCo

To quantify the overestimation, we measure distributions of the discrepancy $\Delta(s,a) = \widehat{Q}^\pi(s,a) - Q^\pi_{\mathrm{MC}}(s,a)$ between TQC's and Monte Carlo value estimates. The randomness is due to $s \sim \rho^\pi(\cdot), a \sim \pi(\cdot|s)$.

The measurements were performed in the end of the learning for each method. Figure 6 shows how overestimation decreases with an increase of the number of dropped atoms d. The best performance corresponds to the level of overestimation close to zero. See Appendix G for discussion of the details of the experiment and cautions regarding its interpretation.

5. Ablation Study

We ablate TQC on the Humanoid 3D environment, which has the highest resolution power due to its difficulty, and Walker2d—a 2D environment with the largest sizes of action

Table 3. Maximum immediate evaluation score (thousands). Maximum was taken over the learning progress and over 10 seeds (see Figure 5 for the mean plot). ARS results were taken from (Mania et al., 2018). The best return per row is bolded.

ENV	ARS-V2-T	SAC	TQC
HOP	3.909	4.232	**4.288**
HC	6.722	16.934	**18.908**
WAL	**11.389**	6.900	8.646
ANT	5.146	7.417	**9.011**
HUM	11.600	9.411	**13.163**

and observation spaces. In this section and in the Appendix we average metrics over four seeds.

5.1. Design choices evaluation

The path from SAC to TQC comprises five modifications: Q-network size increase (**B**ig), quantile Q-network introduction (**Q**uantile), target distribution truncation (**T**runcate), atom pooling (**P**ool), and ensembling. To reveal the effects behind these modifications, we build four methods – the intermediate steps on the incremental path from SAC to TQC. Each subsequent method adds the next modification from the list to the previous method or changes the order of applying modifications. For all modifications, except the final (ensembling), we use $N = 2$ networks. In all truncation operations we drop dN atoms in total, where $d = 2$.

B-SAC is SAC with an increased size of Q-networks (Big SAC): 3 layers with 512 neurons versus 2 layers of 256 neurons in SAC. Policy network size does not change.

QB-SAC is B-SAC with Quantile distributional networks (Dabney et al., 2018b). This modification changes the form of Q-networks and the loss function, quantile Huber (equation 8). We adapt the clipped double estimator (Fujimoto et al., 2018b) to quantile networks: we recover Q-values from distributions and use atoms of the argmin of Q-values to compute the target distribution $Y^{QB}(s,a) := \frac{1}{M}\sum_{m=1}^{M} \delta\left(y^m(s,a)\right)$, where $y^m(s,a)$ is

$$r(s,a) + \gamma[\theta^m_{\overline{\psi}_{j(s',a')}}(s',a') - \alpha \log \pi_\phi(a'|s')] \qquad (16)$$

and $j(s',a') := \arg\min_n \frac{1}{M}\sum_{m=1}^{M} \theta^m_{\overline{\psi}_n}(s',a')$.

TQB-SAC is QB-SAC with individual truncation instead of min: Z_{ψ_n} is trained to approximate the truncated temporal difference distribution $Y^{TQB}_n(s,a)$, which is based on the predictions of the single target network $Z_{\overline{\psi}_n}$ only. That is, Z_{ψ_n} is trained to approximate $Y^{TQB}_n(s,a) := \frac{1}{k}\sum_{m=1}^{k} \delta(y^m_n(s,a))$, where $y^m_n(s,a)$ is

$$r(s,a) + \gamma[\theta^m_{\overline{\psi}_n}(s',a') - \alpha \log \pi_\phi(a'|s')]. \qquad (17)$$

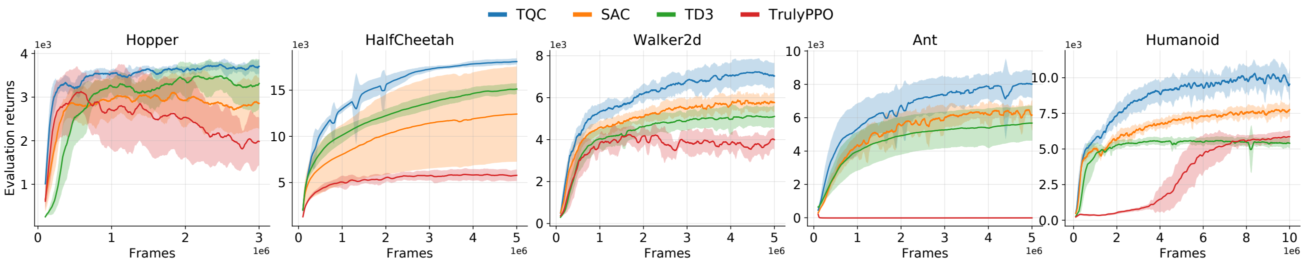


Figure 5. Average performances of methods on MuJoCo Gym Environments with ± std shaded. Smoothed with a window of 100.

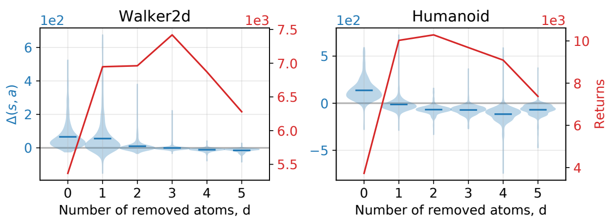


Figure 6. The dependence of the performance (red) and the distribution of value estimation error (blue) on the number of atoms removed d. A blue bar depicts the mean of the distribution. See Section 4.3 for details.

PTQB-SAC is TQB-SAC with pooling: Z_{ψ_n} approximates the mixture of (already truncated) $Y_n^{TQB}(s,a), n \in [1..N]$:

$$Y^{PTQB}(s,a) := \frac{1}{kN} \sum_{n=1}^{N} \sum_{m=1}^{k} \delta(y_n^m(s,a)). \tag{18}$$

TQC = TPQB-SAC is PTQB-SAC with pooling and truncation operations swapped. This modification drops the same number of atoms as two previous methods, but differs in *which* atoms are dropped. TQC drops dN largest *from the union of* N critics predictions. While each of PTQB-SAC and TQB-SAC (no pooling) drops d largest atoms *from each* of N critics.

Ensembling To illustrate the advantage brought by ensembling, we include the results for TQC with two and five Z-networks.

The ablation results (Figure 7) suggest the following. The increased network size does not necessarily improve SAC (though some improvement is visible on Walker2d). The quantile representation improves the score in both environments with the most notable improvement on Humanoid. Among the three modifications—individual truncation, and two orders of applying the pooling and truncation—the TQC is a winner for Humanoid and "truncate-pooling" modification seems to be better on Walker2d. Overall, truncation stabilizes results on Walker2d and reduces the seed variance on Humanoid. Finally, the ensembling consistently improves results on both environments.

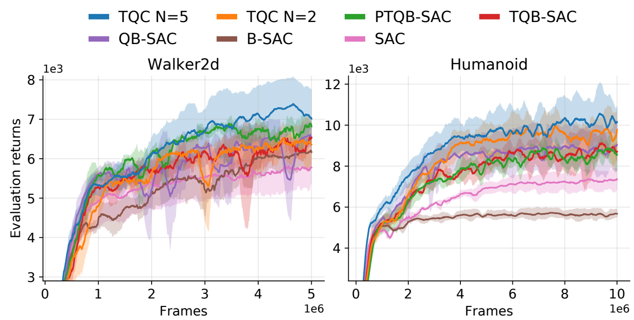


Figure 7. Design choices evaluation. $N = 2$ where isn't stated otherwise, $d = 2$ and $M = 25$ where applicable. Smoothed with a window of 200, ± std is shaded.

5.2. Details on overestimations

TD3 To better understand the interplay between TQC and SAC, we measured the performance of TQC with TD3, instead of SAC, as the base algorithm. Appendix E shows that the most of the improvement comes from distributional representations, and truncation is responsible for stability. The best performance across variations is well below the SOTA, achieved by TQC.

Variations of Clipped Estimator In Appendix F we report results on several variations of the Clipped Double Estimator (Fujimoto et al., 2018b).

1. We varied the number of networks $N \in [1..5]$ under the $\min$ to change the intensity of overestimation compensation. The results in Appendix F.1 shows that the usual $N = 2$ delivers best performance.

2. We benchmarked the convex sum of $\min$ and $\max$ over two networks (Fujimoto et al., 2018a) to achieve intermediate level of overestimation compensation between $N = 1$ and $N = 2$. Performance tends to improve with an increase of the weight in front of $\min$ (Appendix F.2). That is $N = 2$ with the usual estimator is the best.

3. We bencmarked the same convex sum, as above, but with Quantile Z-networks, (Q-SAC and QB-SAC). The

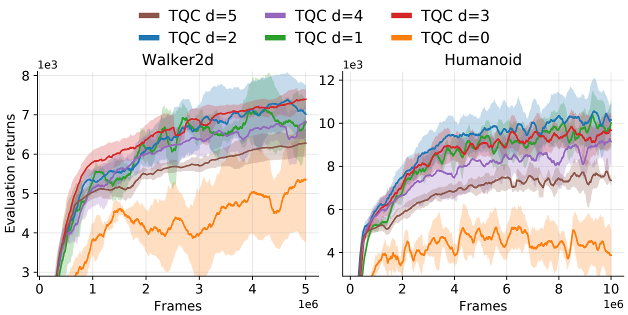


Figure 8. Varying the number of dropped atoms per critic d. $N = 5$ networks, $M = 25$ atoms. Smoothed with a window of 200, $\pm$ std is plotted.

best performance across variations is much closer to that of the TQC and tends to improve with an increase of the weight in front of min (Appendix F.2). We have found that Quantile Z-networks tend to benefit more from an increase in network size.

Appendix F.3 includes learning curves for TQC with small and large $N \in \{2, 5\}$ networks.

Overestimation in MuJoCo In Appendix G we discuss the details of the measurments in Figure 6, as well as compare them with those for the Clipped Estimator. Also we discuss the problems behind measuring the average overestimation. In Appendix H we provide the evidence for similar magnitudes of overestimation for each critic in an ensemble.

5.3. Sensitivity to hyperparameters

Number of truncated atoms In this experiment we vary the number of atoms (per network) to drop in the range $d \in [0..5]$. The total number of atoms dropped is dN. We fix the number of atoms for each Q-network to $M = 25$. The results (Figure 8) show that (1) truncation is essential and (2) there is an optimal number of dropped atoms (i.e., $d = 2$ or $d = 3$). Note, that the dependence of the performance on d may differ across environments. Walker and Humanoid require $d \approx 2$, while HalfCheetah benefits from $d < 2$, and Hopper benefits from $d > 2$ (Appendix D.3).

Total number of atoms In this experiment we vary the total number of atoms $M \in \{10, 15, 25, 35, 50\}$ and adjust the number of dropped quantiles to keep the truncation ratio approximately constant. The results in Appendix D.2 suggest this parameter does not have much influence, except for the case of very small M, such as 10. For $M \geq 15$ learning curves are indistinguishable.

Number of Z-networks In this experiment we vary the number of Z-networks $N \in \{1, 2, 3, 5, 10\}$. The results in Appendix D.1) suggest that (1) a single network is consistently inferior to larger ensembles and (2) performance improvement saturates at approximately $N = 3$.

Table 4. Time measurements (in seconds) of a single training epoch (1000 frames), averaged over 1000 epochs, executed on the Tesla P40 GPU.

Env	SAC	B-SAC	TQC N=2	TQC N=5
Walker2d	9.5	13.9	14.1	32.4
Humanoid	10.7	16.5	17.4	36.8

The computational overhead incurred by ensembling and distributional representations slows learning down. Table 4 compares time costs between SAC and TQC modifications.

6. Related Work

Distributional perspective Since the introduction of distributional paradigm (see White (1988) and references therein) and its reincarnation for deep reinforcement learning (Bellemare et al., 2017) a great body of research emerged. Dabney et al. proposed a method to learn quantile values (or locations) for a uniform grid of fixed (Dabney et al., 2018b) or sampled (Dabney et al., 2018a) quantile fractions. Yang et al. (2019) proposed a method to learn both quantile fractions and quantile values (i.e. locations and probabilities of elements in the mixture approximating the unknown distribution). Choi et al. (2019) used a mixture of Gaussians for approximating the distribution of returns. Most of these works, as well as their influential follow-ups, such as (Hessel et al., 2018), are devoted to the discrete control setting.

The adoption of the distributional paradigm in continuous control, to the best of our knowledge, starts from D4PG (Barth-Maron et al., 2018)—a distributed distributional off-policy algorithm building on C51 (Bellemare et al., 2017). Recently, the distributional paradigm was adopted in distributed continuous control for robotic grasping (Bodnar et al., 2019). The authors proposed Q2-Opt as a collective name for two variants: based on QR-DQN (Dabney et al., 2018b), and on IQN (Dabney et al., 2018a). In contrast to D4PG and Q2-Opt, we focus on the usual, non-distributed setting and modify the target on which the critic is trained.

A number of works develop exploration methods based on the quantile form of value-function. DLTV (Mavrin et al., 2019) uses variability of the quantile distribution in the exploration bonus. QUOTA (Zhang & Yao, 2019)—the option-based modification of QR-DQN—partitions a range of quantiles into contiguous windows and trains a separate intra-option policy to maximize an average of quantiles in each of the windows. Our work proposes an alternative method for critic training, which is unrelated to the exploration problem.

Most importantly, our work differs from the research outlined above in our aim to control the *overestimation bias* by leveraging quantile representation of a critic network.

Overestimation bias The overestimation bias is a longstanding topic in several research areas. It is known as the max-operator bias in statistics (D'Eramo et al., 2017) and as the "winner's curse" in economics (Smith & Winkler, 2006; Thaler, 2012).

The statistical community studies estimators of the maximum expected value of a set of independent random variables. The simplest estimator—the maximum over sample means, Maximum Estimator (ME)—is biased positively, while for many distributions, such as Gaussian, an unbiased estimator does not exist (Ishwaei D et al., 1985). The Double Estimator (DE) (Stone, 1974; Van Hasselt, 2013) uses cross-validation to decorrelate the estimation of the argmax and of the value for that argmax. He & Guo (2019) proposed a coupled estimator as an extension of DE to the case of partially overlapping cross-validation folds. Many works have aimed at alleviating the negative bias of DE, which in absolute value can be even larger than that of ME. D'Eramo et al. (2016) assumed the Gaussian distribution for the sample mean and proposed Weighted Estimator (WE) with a bias in between of that for ME and DE. Imagaw & Kaneko (2017) improved WE by using UCB for weights computation. D'Eramo et al. (2017) assumed a certain spatial correlation and extended WE to continuous sets of random variables. The problem of overestimation has also been discussed in the context of optimal stopping and sequential testing (Kaufmann et al., 2018).

The reinforcement learning community became interested in the bias since the work of Thrun & Schwartz (1993), who attributed a systematic overestimation to the generalization error. The authors proposed multiple ways to alleviate the problem, including (1) bias compensation with additive pseudo costs and (2) underestimation (e.g., in the uncertain areas). The underestimation concept became much more prominent, while the adoption of "additive compensation" is quite limited to date (Patnaik & Anwar, 2008; Lee & Powell, 2012).

Van Hasselt (2010) proposed Double Estimation in Q-learning, which was subsequently adapted to neural networks as Double DQN Van Hasselt et al. (2015). Subsequently, (Zhang et al., 2017) and Lv et al. (2019) introduced the Weighted Estimator in the reinforcement learning community.

Another approach against overestimation and overall Q-function quality improvement is based on the idea of averaging or ensembling. Embodiments of this approach are based on dropout Anschel et al. (2017), employing previous Q-function approximations (Ghavamzadeh et al., 2011; Anschel et al., 2017), the linear combination between $\min$ and $\max$ over the pool of Q-networks (Li & Hou, 2019; Kumar et al., 2019), or the random mixture of predictions from the pool (Agarwal et al., 2019). Buckman et al. (2018) reported the reduction in overestimation originating from ensembling in model-based learning.

In *continuous* control, Fujimoto et al. (2018b) proposed the TD3 algorithm, taking the $\min$ over two approximators of Q-function to reduce the overestimation bias. Later, for *discrete* control Lan et al. (2020) developed a MaxMin Q-learning, taking the $\min$ over more than two Q-functions. We build upon the minimization idea of Fujimoto et al. (2018b) and, following Lan et al. (2020) use multiple approximators.

Our work differs in that we do not propose to control the bias by choosing *between* multiple approximators or *weighting* them. For the first time, we propose to successfully control the overestimation even for a single approximator and use ensembling only to improve the performance further.

7. Conclusion and Future Work

In this work, we propose to control the overestimation bias on the basis of aleatoric uncertainty. The method we propose comprises three essential ideas: distributional representations, truncation of a distribution, and ensembling.

Simulations reveal favorable properties of our method: low expected variance of the approximation error as well as the fine control over the under- and overestimation. The exceptional results on the standard continuous control benchmark suggest that distributional representations may be useful for controlling the overestimation bias.

Since little is known about the connection between aleatoric uncertainty and overestimation, we see its investigation as an exciting avenue for future work.

8. Acknowledgements

We would like to thank Artem Sobolev, Arsenii Ashukha, Oleg Ivanov and Dmitry Nikulin for their comments and suggestions regarding the early versions of the manuscript. We also thank the anonymous reviewers for their feedback.

This research was supported in part by the Russian Science Foundation grant no. 19-71-30020.

References

Agarwal, R., Schuurmans, D., and Norouzi, M. Striving for simplicity in off-policy deep reinforcement learning. *arXiv preprint arXiv:1907.04543*, 2019.

Anschel, O., Baram, N., and Shimkin, N. Averaged-dqn:

Variance reduction and stabilization for deep reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 176–185. JMLR. org, 2017.

Barth-Maron, G., Hoffman, M. W., Budden, D., Dabney, W., Horgan, D., Muldal, A., Heess, N., and Lillicrap, T. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

Bellemare, M. G., Dabney, W., and Munos, R. A Distributional Perspective on Reinforcement Learning. *arXiv e-prints*, art. arXiv:1707.06887, Jul 2017.

Bodnar, C., Li, A., Hausman, K., Pastor, P., and Kalakrishnan, M. Quantile qt-opt for risk-aware vision-based robotic grasping. *arXiv preprint arXiv:1910.02787*, 2019.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Buckman, J., Hafner, D., Tucker, G., Brevdo, E., and Lee, H. Sample-efficient reinforcement learning with stochastic ensemble value expansion. In *Advances in Neural Information Processing Systems*, pp. 8224–8234, 2018.

Choi, Y., Lee, K., and Oh, S. Distributional deep reinforcement learning with a mixture of gaussians. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9791–9797. IEEE, 2019.

Dabney, W., Ostrovski, G., Silver, D., and Munos, R. Implicit quantile networks for distributional reinforcement learning. *arXiv preprint arXiv:1806.06923*, 2018a.

Dabney, W., Rowland, M., Bellemare, M. G., and Munos, R. Distributional reinforcement learning with quantile regression. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018b.

D'Eramo, C., Nuara, A., Pirotta, M., and Restelli, M. Estimating the maximum expected value in continuous reinforcement learning problems. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

D'Eramo, C., Restelli, M., and Nuara, A. Estimating maximum expected value through gaussian approximation. In *International Conference on Machine Learning*, pp. 1032–1040, 2016.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. *arXiv preprint arXiv:1812.02900*, 2018a.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1587–1596, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018b. PMLR. URL http://proceedings.mlr.press/v80/fujimoto18a.html.

Ghavamzadeh, M., Kappen, H. J., Azar, M. G., and Munos, R. Speedy q-learning. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 24*, pp. 2411–2419. Curran Associates, Inc., 2011. URL http://papers.nips.cc/paper/4251-speedy-q-learning.pdf.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018a.

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

He, M. and Guo, H. Interleaved q-learning with partially coupled training process. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 449–457. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Imagaw, T. and Kaneko, T. Estimating the maximum expected value through upper confidence bound of likelihood. In *2017 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, pp. 202–207. IEEE, 2017.

Ishwaei D, B., Shabma, D., and Krishnamoorthy, K. Nonexistence of unbiased estimators of ordered parameters. *Statistics: A Journal of Theoretical and Applied Statistics*, 16(1):89–95, 1985.

Kaufmann, E., Koolen, W. M., and Garivier, A. Sequential test for the lowest mean: From thompson to murphy sampling. In *Advances in Neural Information Processing Systems*, pp. 6332–6342, 2018.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pp. 11761–11771, 2019.

Lan, Q., Pan, Y., Fyshe, A., and White, M. Maxmin q-learning: Controlling the estimation bias of q-learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Bkg0u3Etwr.

Lee, D. and Powell, W. B. An intelligent battery controller using bias-corrected q-learning. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

Li, Z. and Hou, X. Mixing update q-value for deep reinforcement learning. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–6. IEEE, 2019.

Lv, P., Wang, X., Cheng, Y., and Duan, Z. Stochastic double deep q-network. *IEEE Access*, 7:79446–79454, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2922706.

Mania, H., Guy, A., and Recht, B. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

Mavrin, B., Yao, H., Kong, L., Wu, K., and Yu, Y. Distributional reinforcement learning for efficient exploration. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4424–4434, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

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

Patnaik, K. and Anwar, S. Q learning in context of approximation spaces. *Contemporary Engineering Sciences*, 1 (1):41–49, 2008.

Smith, J. E. and Winkler, R. L. The optimizer's curse: Skepticism and postdecision surprise in decision analysis. *Management Science*, 52(3):311–322, 2006.

Stone, M. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):111–133, 1974.

Thaler, R. *The winner's curse: Paradoxes and anomalies of economic life*. Simon and Schuster, 2012.

Thrun, S. and Schwartz, A. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 Connectionist Models Summer School Hillsdale, NJ. Lawrence Erlbaum*, 1993.

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. IEEE, 2012.

Van Hasselt, H. Double q-learning. In *Advances in Neural Information Processing Systems*, pp. 2613–2621, 2010.

Van Hasselt, H. Estimating the maximum expected value: an analysis of (nested) cross validation and the maximum sample average. *arXiv preprint arXiv:1302.7175*, 2013.

Van Hasselt, H., Guez, A., and Silver, D. Deep Reinforcement Learning with Double Q-learning. *arXiv e-prints*, art. arXiv:1509.06461, Sep 2015.

White, D. Mean, variance, and probabilistic criteria in finite markov decision processes: a review. *Journal of Optimization Theory and Applications*, 56(1):1–29, 1988.

Yang, D., Zhao, L., Lin, Z., Qin, T., Bian, J., and Liu, T.-Y. Fully parameterized quantile function for distributional reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 6190–6199, 2019.

Zhang, S. and Yao, H. Quota: The quantile option architecture for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5797–5804, 2019.

Zhang, Z., Pan, Z., and Kochenderfer, M. J. Weighted double q-learning. In *IJCAI*, pp. 3455–3461, 2017.