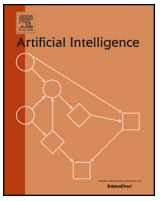
Artificial Intelligence 336 (2024) 104204

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint



Addressing maximization bias in reinforcement learning with two-sample testing

Martin Waltz [a,*], Ostap Okhrin [a,b]

[a] Technische Universität Dresden, Chair of Econometrics and Statistics, esp. in the Transport Sector, Dresden, 01062, Germany
[b] Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI), Dresden/Leipzig, Germany

ARTICLE INFO

Keywords:
Maximum expected value
Two-sample testing
Reinforcement learning
Q-learning
Estimation bias

ABSTRACT

Value-based reinforcement-learning algorithms have shown strong results in games, robotics, and other real-world applications. Overestimation bias is a known threat to those algorithms and can sometimes lead to dramatic performance decreases or even complete algorithmic failure. We frame the bias problem statistically and consider it an instance of estimating the maximum expected value (MEV) of a set of random variables. We propose the T-Estimator (TE) based on two-sample testing for the mean, that flexibly interpolates between over- and underestimation by adjusting the significance level of the underlying hypothesis tests. We also introduce a generalization, termed K-Estimator (KE), that obeys the same bias and variance bounds as the TE and relies on a nearly arbitrary kernel function. We introduce modifications of Q-Learning and the Bootstrapped Deep Q-Network (BDQN) using the TE and the KE, and prove convergence in the tabular setting. Furthermore, we propose an adaptive variant of the TE-based BDQN that dynamically adjusts the significance level to minimize the absolute estimation bias. All proposed estimators and algorithms are thoroughly tested and validated on diverse tasks and environments, illustrating the bias control and performance potential of the TE and KE.

1. Introduction

Estimating the maximum expected value (MEV) of a set of random variables is a long-standing statistical problem, including early contributions of Blumenthal and Cohen [11], Dudewicz [18], and Dhariyal et al. [16]. These works show that for various underlying distributions, for example, Gaussian and Binomial distributions, an unbiased estimator does not exist. The problem has recently attained increased attention since it also arises in reinforcement learning (RL), where an agent interacts with an environment while optimizing for a policy - a mapping from states to actions - that maximizes a numerical reward signal [57]. Many of the frequently used RL algorithms define a policy-dependent action-value, also called Q-value, for each state-action pair. This value represents the expected sum of discounted rewards when executing the given action in the given state and following a specific policy afterward. In particular, the update rule of the Q-Learning algorithm [68] is based on adjusting the Q-estimate for a given state-action pair towards the observed reward and the maximum of the estimated Q-values of the next state. However, this use of the Maximum Estimator (ME) of the MEV leads to overestimations of action-values, which are transmitted throughout the update routine [61]. These can damage the learning performance or even lead to failure of the algorithm [58], especially when function approximation is used [63].

* Corresponding author.
E-mail address: martin.waltz@tu-dresden.de (M. Waltz).

https://doi.org/10.1016/j.artint.2024.104204
Received 29 September 2022; Received in revised form 8 August 2024; Accepted 10 August 2024

Available online 16 August 2024

Van Hasselt [61] proposed the Double Estimator (DE), which splits the data into independent sets, thereby separating the selection and evaluation of a maximizing value. The corresponding Double Q-Learning is a popular choice among practitioners [70,26]. Although the DE introduces underestimation bias, Double Q-Learning offers improved robustness and strong performances, especially in highly stochastic environments. Another crucial contribution is the work by D'Eramo et al. [20], in which the Weighted Estimator (WE) alongside Weighted Q-Learning is proposed. From a bias perspective, the estimator builds a compromise between the overestimating ME and the underestimating DE. However, the WE does not offer additional flexibility in selecting the level of bias and is computationally demanding since it requires numerical integration or Monte Carlo approximation. Lan et al. [37] presented MaxMin Q-Learning, which builds on learning multiple approximators for the same action-value. Whether the algorithm over- or underestimates Q-values depends on the discrete choice of the number of approximators. We refer to the corresponding estimator of the MEV as the MaxMin Estimator (MME). Notably, the DE, WE, and the MME also led to modifications of the Deep Q-Networks (DQN, [44]). The DQN expanded Q-Learning to the deep neural network (DNN) setting and paved the path for the striking success of RL in recent years [52,65].

D'Eramo et al. [20], Lan et al. [37], among others, have shown that both over- and underestimation of Q-values might not always be harmful, depending on, e.g., the stochasticity of the environment, the difference of the action-values, the size of the action space, or the time horizon. We argue that a competitive estimator of the MEV should thus be able to interpolate between over- and underestimation via an interpretable hyperparameter, enabling it to deal with a diverse set of environments. Furthermore, the estimator should obey a variance bound similar to competitors like the ME and DE to avoid trading a smaller absolute bias for a significantly increased variance bound. In addition, for practical application, the estimator should be fast and stable to compute. Fulfilling these criteria, we propose an estimator based on two-sample testing for the mean, named T-Estimator (TE). The idea is to get a statistically significant statement of whether one mean is truly larger than others. Consequently, the hyperparameter is the level of significance α. The ME is shown to be a special case of the TE with $\alpha = 0.5$. Building on the two-sample test statistic, we further consider a generalization termed K-Estimator (KE), which is characterized by a suitable kernel function and can smooth the discontinuities around testing decisions of the TE. We theoretically and empirically analyze the TE and KE regarding their biases and variances, for which general bounds are derived. Using these newly defined estimators, we propose RL algorithms for both the tabular case and with DNNs as function approximators. Since the two-sample testing procedure incorporates variance estimates of the involved variables, we employ an online variance update routine [19] in the tabular scenario, and the framework of the Boostrapped DQN (BDQN, [47]) in the DNN setting. Furthermore, we prove the convergence of the algorithm for the tabular case.

The empirical evidence that over- and underestimation of action-values is not necessarily detrimental to learning performance might be explained by the connection of Q-estimates to the exploration procedure of algorithms [23]. However, Fox [22] and Liang et al. [41] argue that these topics should be addressed separately by focusing firstly on unbiased value-estimation and secondly on improved exploration schemes. We acknowledge this perspective by additionally proposing an adaptive tuning mechanism for the significance level α of the TE in the DNN setting with the objective of minimizing the absolute estimation bias. The approach complements recent proposals of Dorka et al. [17] and Wang et al. [67]. The dynamic adjustment of α is realized by running partial greedy episodes and comparing n-step returns [57] with the action-value estimates for the visited state-action pairs. Furthermore, through learning α, we avoid the computationally demanding tuning process of this environment-specific hyperparameter. Finally, we demonstrate the performance potential of all the newly proposed estimators and algorithms by extensively testing them in various tasks and environments, with and without function approximation.

The paper is organized as follows: Section 2 formalizes the problem of estimating the MEV. Section 3 details the proposed estimators, and Section 4 analyzes them with and without fulfillment of the underlying independence assumptions. Section 5 introduces the RL setup and presents the new temporal-difference algorithms, while Section 6 details the measurement of estimation bias and introduces the adaptive update mechanism of α. The experiments are shown and thoroughly discussed in Section 7, with the code being available at: https://github.com/MarWaltz/TUD_RL. Section 8 provides further literature on the state-of-the-art, and Section 9 concludes this article.

2. Estimating the maximum expected value

2.1. Problem definition

Let us consider $M \geq 2$ independent real-valued random variables $X_1, \dots, X_M$ with finite expectations $\mu_1 = \mathrm{E}(X_1), \dots, \mu_M = \mathrm{E}(X_M)$ and finite variances $\sigma_1^2 = \mathrm{Var}(X_1), \dots, \sigma_M^2 = \mathrm{Var}(X_M)$. The corresponding probability density functions (pdfs) and cumulative distribution functions (cdfs) are denoted $f_{X_1}, \dots, f_{X_M}$ and $F_{X_1}, \dots, F_{X_M}$, respectively. The quantity of interest is the *maximum expected value*: $\mu_* = \max_i \mu_i$. Estimation is performed based on samples $S = (S_1, \dots, S_M)$, where S_i is a set containing samples from X_i for $i = 1, \dots, M$, without knowing moments or imposing distributional assumptions. For simplicity, we refer to the set of samples S_i simply as sample S_i. In addition, we assume to have at least two different elements inside each sample S_i to ensure a non-zero sample variance. Moreover, the realizations in a sample S_i are assumed to be independent and identically distributed. Consequently, we assume that there are no *in-sample* dependencies inside a sample S_i and no *cross-sample dependencies* between samples S_i and S_j, for $i, j = 1, \dots, M$ and $i \neq j$. The unbiased sample mean of S_i is denoted $\hat{\mu}_i(S_i)$, while an estimator of the MEV is referred to as $\hat{\mu}_*(S)$. Throughout the paper, we use abbreviated notations for conciseness, such as $\hat{\mu}_i = \hat{\mu}_i(S_i)$, $\hat{\mu}_* = \hat{\mu}_*(S)$, and similar expressions. Primary evaluation criteria of an estimator are its bias $\mathrm{Bias}(\hat{\mu}_*) = \mathrm{E}(\hat{\mu}_*) - \mu_*$, and variance $\mathrm{Var}(\hat{\mu}_*) = \mathrm{E}\left\{[\hat{\mu}_* - \mathrm{E}(\hat{\mu}_*)]^2\right\}$. These can be aggregated to the mean squared error $\mathrm{MSE}(\hat{\mu}_*) = \mathrm{E}\left[(\hat{\mu}_* - \mu_*)^2\right] = \mathrm{Bias}(\hat{\mu}_*)^2 + \mathrm{Var}(\hat{\mu}_*)$.

2.2. Maximum Estimator

The ME $\hat{\mu}_*^{\mathrm{ME}}$ is the classic approach and takes the maximum of unbiased mean estimates:

$$\hat{\mu}_*^{\mathrm{ME}} = \max_i \hat{\mu}_i.$$

Denoting the pdf of $\hat{\mu}_i$ as $\hat{f}_i$ and the corresponding cdf as $\hat{F}_i$, it holds:

$$\mathrm{E}\left(\hat{\mu}_*^{\mathrm{ME}}\right) = \sum_{i=1}^{M} \int_{-\infty}^{\infty} x \hat{f}_i(x) \prod_{\substack{j=1 \\ j\neq i}}^{M} \hat{F}_j(x) dx. \tag{1}$$

The ME is positively biased: $\mathrm{E}\left(\hat{\mu}_*^{\mathrm{ME}}\right) \geq \mu_*$; see Van Hasselt [61]. This bias occurs because x inside the integral positively correlates with the monotonically increasing product $\prod_{\substack{j=1 \\ j\neq i}}^{M} \hat{F}_j(x)$. Following Aven [3], a general upper bound for the bias can be given:

$$0 \leq \mathrm{Bias}\left(\hat{\mu}_*^{\mathrm{ME}}\right) \leq \sqrt{\frac{M-1}{M} \sum_{i=1}^{M} \mathrm{Var}\left(\hat{\mu}_i\right)}. \tag{2}$$

The bias is particularly large when $\mu_1 \approx \ldots \approx \mu_M$. Furthermore, it can be shown that the variance of $\hat{\mu}_*^{\mathrm{ME}}$ is bounded from above: $\mathrm{Var}\left(\hat{\mu}_*^{\mathrm{ME}}\right) \leq \sum_{i=1}^{M} \frac{\sigma_i^2}{|S_i|}$, where $|S_i|$ is the size of S_i; see Van Hasselt [62].

2.3. Double Estimator

Van Hasselt [61] introduced the DE, which is thoroughly analyzed in Van Hasselt [62]. The key idea is to separate the selection of the maximizing random variable and the evaluation of its sample mean, which is performed simultaneously in the ME. The DE splits S randomly into disjoint subsets $S^A = (S_1^A, \ldots, S_M^A)$ and $S^B = (S_1^B, \ldots, S_M^B)$, guaranteeing that means based on the subsets are still unbiased. Afterwards, one selects an index which maximizes the sample mean in S^A: $a^* \in \{i \mid \hat{\mu}_i(S_i^A) = \max_j \hat{\mu}_j(S_j^A)\}$. The DE is defined by evaluating a^* on S^B: $\hat{\mu}_*^{\mathrm{DE}}(S) = \hat{\mu}_{a^*}(S_{a^*}^B)$. Similarly, one can perform the same procedure with S^A and S^B switched to get a second DE estimate. Averaging both DE estimates yields the 2-fold Cross-Validation estimator (CVE) $\hat{\mu}_*^{\mathrm{CVE}}$, which has a reduced variance in comparison to a single DE estimate. The expectations of the DE and the CVE are equal since both DE estimates (for S^A and S^B) have identical expectations:

$$\begin{aligned}
\mathrm{E}\left(\hat{\mu}_*^{\mathrm{CVE}}\right) = \mathrm{E}\left(\hat{\mu}_*^{\mathrm{DE}}\right) &= \sum_{i=1}^{M} \mathrm{E}\left[\hat{\mu}_i(S_i^B)\right] \mathbb{P}(i = a^*) \\
&= \sum_{i=1}^{M} \mathrm{E}\left[\hat{\mu}_i(S_i^B)\right] \int_{-\infty}^{\infty} \hat{f}_i^A(x) \prod_{\substack{j=1 \\ j\neq i}}^{M} \hat{F}_j^A(x) dx,
\end{aligned} \tag{3}$$

where $\hat{f}_i^A$ and $\hat{F}_i^A$ are the cdf and pdf of $\hat{\mu}_i(S_i^A)$, respectively. Van Hasselt [61] showed that the DE is prone to underestimation: $\mathrm{E}\left(\hat{\mu}_*^{\mathrm{DE}}\right) \leq \mu_*$, because it might attribute non-zero selection probability to non-maximum variables. Furthermore, Van Hasselt [62] conjectures the following lower bound for the bias:

$$-\frac{1}{2}\left(\sqrt{\sum_{i=1}^{M} \frac{\sigma_i^2}{|S_i^A|}} + \sqrt{\sum_{i=1}^{M} \frac{\sigma_i^2}{|S_i^B|}}\right) < \mathrm{Bias}(\hat{\mu}_*^{\mathrm{DE}}) \leq 0,$$

while the variance of the CVE is shown to be bounded as the ME: $\mathrm{Var}\left(\hat{\mu}_*^{\mathrm{CVE}}\right) \leq \sum_{i=1}^{M} \frac{\sigma_i^2}{|S_i|}$. Worth mentioning is that the variance of the CVE is not necessarily half the variance of the DE, as there is non-zero covariance between the two DE estimates, see the example in Appendix A. Throughout the experiments, we follow D'Eramo et al. [15] and use the CVE instead of the DE whenever possible.

2.4. Weighted Estimator

D'Eramo et al. [20] introduced the Weighted Estimator (WE) for the MEV, which is a weighted mean of all sample averages. Each weight corresponds to the probability of $\hat{\mu}_i$ being larger than all other means:

$$\hat{\mu}_*^{\mathrm{WE}} = \sum_{i=1}^{M} w_i \hat{\mu}_i = \sum_{i=1}^{M} \mathbb{P}\left(\hat{\mu}_i = \max_j \hat{\mu}_j\right) \hat{\mu}_i.$$

Since the probabilities depend on the unknown mean distributions $\hat{f}_i$, the authors propose a Gaussian approximation based on the central limit theorem:

$$\hat{\mu}_*^{\mathrm{WE}} = \sum_{i=1}^{M} \hat{\mu}_i \int_{-\infty}^{\infty} \tilde{f}_i(x) \prod_{\substack{j=1 \\ j\neq i}}^{M} \tilde{F}_j(x) dx, \tag{4}$$

where $\tilde{f}_i$ and $\tilde{F}_i$ are the Gaussian pdf and cdf, respectively, with mean $\hat{\mu}_i$ and variance $\frac{\hat{\sigma}_i^2}{|S_i|}$. The unbiased estimate of σ_i^2 is denoted $\hat{\sigma}_i^2$ and $|S_i|$ refers to the sample size. Crucially, the bias of the WE is bounded by the ME and DE:

$$\mathrm{Bias}(\hat{\mu}_*^{\mathrm{DE}}) \leq \mathrm{Bias}(\hat{\mu}_*^{\mathrm{WE}}) \leq \mathrm{Bias}(\hat{\mu}_*^{\mathrm{ME}}),$$

while it exhibits the same variance bound: $\mathrm{Var}\left(\hat{\mu}_*^{\mathrm{WE}}\right) \leq \sum_{i=1}^{M} \frac{\sigma_i^2}{|S_i|}$; see D'Eramo et al. [20]. Thus, the bias of the WE might be positive or negative, depending on the distribution of the random variables. A drawback of this estimator lies in increased computation time since calculating the integrals in (4) is a demanding process. Tackling this issue, D'Eramo et al. [15] propose to use Monte Carlo approximations instead, and we follow this approach when computing the WE in the experiments. The Monte Carlo sample sizes in those cases are set to 100.

2.5. MaxMin Estimator

Lan et al. [37] proposed MaxMin Q-Learning with the corresponding MaxMin Estimator (MME) for the MEV. Similar to the DE, the MME splits each of the M samples in $S = (S_1, \dots, S_M)$ into N disjoint, equally-sized subsamples, which we denote as S_i^j for $i = 1, \dots, M$ and $j = 1, \dots, N$. The estimator is then constructed as follows:

$$\hat{\mu}_*^{\mathrm{MME}} = \max_i \min_j \hat{\mu}(S_i^j), \tag{5}$$

where $\hat{\mu}\left(S_i^j\right)$ is the sample mean of S_i^j. The underlying rationale of (5) is that the underestimation introduced via the minimum operator mitigates the overestimation due to the max operator. Lan et al. [37] provide analytical results for the expectation and variance of the MME in the particular case that the sample means are uniformly distributed. Following D'Eramo et al. [15], a general expression for the expectation of the MME is:

$$\mathrm{E}\left(\hat{\mu}_*^{\mathrm{MME}}\right) = \sum_{i=1}^{M} \sum_{j=1}^{N} \mathbb{P}\left[i = \arg\max_i \min_j \hat{\mu}(S_i^j)\right] \mu_i.$$

The MME can control the estimation bias via the number of subsamples N. Consequently, this approach necessitates learning N distinct approximators for the same quantity, which in turn limits the tuning capabilities of the method due to the discrete nature of N. Moreover, as emphasized by Lan et al. [37], a critical subtlety is that the splitting procedure drastically reduces the available sample size for each of the $M \cdot N$ estimators.

3. Two-sample testing-based estimators

3.1. T-Estimator

To create a flexible estimator which:

(a) is able to interpolate between over- and underestimation,
(b) obeys a variance bound similar to the ME,
(c) has a continuously tunable and interpretable hyperparameter,
(d) is fast and easy to compute,

we propose a procedure based on one-sided two-sample testing for the mean. Generally, for two random variables X_1, X_2, we consider the hypothesis $H_0 : \mu_1 \geq \mu_2$. The test statistic is constructed as follows [66]:

$$T = \frac{\hat{\mu}_1 - \hat{\mu}_2}{\sqrt{\frac{\hat{\sigma}_1^2}{|S_1|} + \frac{\hat{\sigma}_2^2}{|S_2|}}}, \tag{6}$$

where $\hat{\sigma}_i^2 = \frac{1}{|S_i|-1} \sum_{j=1}^{|S_i|} \left(S_{i,j} - \hat{\mu}_i\right)^2$ is the unbiased estimator of the variance σ_i^2 for $i = 1, 2$. We denote with $S_{i,j}$ the j-th observation of sample S_i, while $|S_i|$ is the size of S_i. If the realization of T is smaller than the respective α-quantile, $z_\alpha = \Phi^{-1}(\alpha)$, where Φ is the standard Gaussian cdf, hypothesis H_0 is rejected. The use of the normal distribution as the asymptotic distribution of the test

statistics for H_0 can be justified via the Lindeberg-Lévy central limit theorem, since it holds $\sqrt{|S_i|}\frac{\hat{\mu}_i-\mu_i}{\sigma_i} \xrightarrow[|S_i|\to\infty]{d} \mathcal{N}(0,1)$ for $i=1,2$, and using Slutsky's theorem, as $\hat{\sigma}_i$ converges almost surely to σ_i [51].

Based on this test, the following procedure for estimating the MEV is proposed: First, we consider the complete set of indices $\mathcal{L}=\{1,\dots,M\}$ and select an index that corresponds to a variable with the value of the ME: $i^* \in \{i \mid \hat{\mu}_i = \max_j \hat{\mu}_j\}$. Second, we test for all $i \in \mathcal{L}$ the H_0: $\mu_i \geq \mu_{i^*}$. If H_0 is rejected for some i', we assert $\mu_{i'} < \mu_{i^*}$ and remove variable index i' from the index set: $\mathcal{L} \leftarrow \mathcal{L}\backslash\{i'\}$. Third, we average the remaining $\{\hat{\mu}_i \mid i \in \mathcal{L}\}$. Compactly written:

$$\hat{\mu}_*^{\mathrm{TE}}(\alpha) = \left[\sum_{i=1}^{M} \mathcal{I}\left(T_i \geq z_\alpha\right)\right]^{-1} \sum_{i=1}^{M} \mathcal{I}\left(T_i \geq z_\alpha\right)\hat{\mu}_i, \quad \text{where} \quad T_i = \frac{\hat{\mu}_i - \hat{\mu}_*^{\mathrm{ME}}}{\sqrt{\frac{\hat{\sigma}_i^2}{|S_i|}+\frac{\hat{\sigma}_{i^*}^2}{|S_{i^*}|}}}, \tag{7}$$

and $\mathcal{I}(\cdot)$ is the indicator function. We refer to (7) as *T-Estimator* (TE). In simple words: TE averages the means of all variables, which are statistically not smaller than the one of the ME. Consequently, the selection is a binary decision of rejection or non-rejection of the underlying hypothesis. A key aspect of the TE is the consideration of the values of the sample means together with their uncertainties, expressed by variances. Asymptotically, when $|S_i| \to \infty$ for $i=1,\dots,M$, the TE follows a normal distribution since it is an average of asymptotically normally distributed variables.

The hyperparameter is the significance level α, which is an interpretable quantity for practitioners and researchers, and is naturally restricted to $\alpha \in (0, 0.5]$. One can directly determine the extreme case on the upper domain limit: $\hat{\mu}_*^{\mathrm{TE}}(\alpha=0.5) = \hat{\mu}_*^{\mathrm{ME}}$. This follows from the fact that $z_{0.5}=0$, meaning that the test statistics, which is either zero or negative, has to be zero to fulfill the condition of the indicator function. Therefore, only the means reaching the maximum value will be averaged. Consequently, the ME is a special case of the TE, being prone to overestimation bias with the bounds given in Section 2.2. Intuitively, by reducing α, we reduce the bias since we tend to non-reject H_0 for smaller sample means. If one would consider a significance level of zero, the TE would collapse into the Average Estimator (AE): $\hat{\mu}_*^{\mathrm{AVG}} = M^{-1}\sum_{i=1}^{M}\hat{\mu}_i$. Imagaw and Kaneko [30] provide a similar definition in a multi-armed bandit context. The AE has low variance: $\mathrm{Var}(\hat{\mu}_*^{\mathrm{AVG}}) = M^{-2}\sum_{i=1}^{M}\frac{\sigma_i^2}{|S_i|}$, but severe negative bias: $\mathrm{Bias}(\hat{\mu}_*^{\mathrm{AVG}}) = -M^{-1}\sum_{i=1}^{M}(\max_j \mu_j - \mu_i)$. However, we do not include $\alpha=0$ in our definition domain for the TE since 1) it is statistically not reasonable to consider such hypothesis tests and 2) the uncertainties of the sample means, quantified through variances, are not present anymore.

The following lemma contains the bias bounds of the TE.

Lemma 1. *For $\alpha \in (0, 0.5]$, it holds:*

$$\min_i \mu_i - \max_i \mu_i - \sqrt{\frac{M-1}{M}\sum_{i=1}^{M}\mathrm{Var}\left(\hat{\mu}_i\right)} \leq \mathrm{Bias}\left[\hat{\mu}_*^{\mathrm{TE}}(\alpha)\right] \leq \mathrm{Bias}(\hat{\mu}_*^{\mathrm{ME}}).$$

Further, if $\mathrm{Var}\left(\hat{\mu}_i\right) = V$ for $i=1,\dots,M$ and some $V>0$, then $\mathrm{Bias}\left[\hat{\mu}_*^{\mathrm{TE}}(\alpha)\right]$ is a monotonically increasing function of α.

Proof. The upper bound is straightforward since the TE is a weighted average of sample means, while the ME is the extreme case of weighting the maximum sample mean with one. Regarding the lower bound, we use that per construction:

$$\hat{\mu}_*^{\mathrm{TE}}(\alpha) \geq \min_i \hat{\mu}_i. \tag{8}$$

To see this, we first note that the numerator of the test statistics T_i in (7) is always zero for the ME, leading to a value of one for the corresponding indicator functions for all $\alpha \in (0, 0.5]$. However, since the test statistics for index i positively correlates with $\hat{\mu}_i$ and $\hat{\sigma}_i^2$, extreme variance scenarios are possible in which indices with much smaller means, including the one for the minimum mean, are non-rejected. Considering M might be very large, $\hat{\mu}_*^{\mathrm{TE}}(\alpha)$ can thus be arbitrarily close to $\min_i \hat{\mu}_i$, yielding (8). Building expectations, we have:

$$\mathrm{E}\left[\hat{\mu}_*^{\mathrm{TE}}(\alpha)\right] \geq \mathrm{E}\left(\min_i \hat{\mu}_i\right) \geq \min_i \mu_i - \sqrt{\frac{M-1}{M}\sum_{i=1}^{M}\mathrm{Var}\left(\hat{\mu}_i\right)},$$

where the last inequality uses the bound for the minimum sample average of Aven [3]. The bias follows immediately. The second part regarding the monotonicity in the case of equal variances of the mean estimates follows from the definition in (7) since $z_\alpha \leq z_{\alpha'}$ for $\alpha \leq \alpha'$ and $\alpha, \alpha' \in (0, 0.5]$. □

Lemma 1 shows that the TE can achieve negative and positive bias depending on the level of significance α. However, we can not make a general statement about the sign of the bias for a particular α since this depends on the number and the distribution of the underlying random variables.

Furthermore, the TE is consistent for the MEV since, with increasing sample size, each sample mean approaches its population mean, the ME approaches the true MEV, and the variances of the means tend to zero. Consequently, all tests except for the true MEV variable will reject the H_0, and only the MEV will be left.

Table 1
Exemplary kernel functions. The parameter λ is the standard deviation of the Gaussian cdf, ν is the degree of freedom of the t-distribution, and $\Gamma(x) = \int_0^\infty t^{x-1}\exp(-t)dt$ denotes the gamma function. We abbreviate the standard Gaussian kernel $\Phi_{\lambda=1}$ with Φ.

Kernel	$\kappa(T)$ for $T \leq 0$
cdf: Gaussian Φ_λ	$\int_{-\infty}^{T} \frac{1}{\sqrt{2\pi\lambda^2}} \exp\left[-\frac{1}{2}\left(\frac{t}{\lambda}\right)^2\right] dt$
cdf: t-distribution t_ν	$\int_{-\infty}^{T} \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\left(1+\frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}} dt$
Epanechnikov	$\frac{3}{4}(1-T^2)\mathcal{I}(\lvert T\rvert \leq 1)$
Softmax	$\exp(T)$
Triangle	$(1-\lvert T\rvert)\mathcal{I}(\lvert T\rvert \leq 1)$

The TE involves conducting multiple hypothesis tests, which can potentially lead to an increase in type I error. In such cases, common approaches for correction include the Bonferroni correction [1], the method proposed by Holm [29], or, for large values of M, the procedure developed by Benjamini and Hochberg [9]. However, we have chosen not to incorporate these techniques into the TE for two reasons. Firstly, we consider the significance level α as a freely adjustable hyperparameter. For instance, if we were to employ the Bonferroni correction, which divides the significance level by the number of tests, we could simply choose a different α to achieve a similar effect. Secondly, in the upcoming section, we will introduce a generalization of the TE that generally circumvents the necessity of binary testing decisions.

Regarding variance, the TE shares the common overly pessimistic variance bound, while the proof relies on the TE being a weighted average of means and is similar to D'Eramo et al. [20]:

Lemma 2. *For $\alpha \in (0, 0.5]$, it holds:* $\text{Var}\left[\hat{\mu}^{\text{TE}}(\alpha)\right] \leq \sum_{i=1}^{M} \frac{\sigma_i^2}{|S_i|}$.

3.2. K-Estimator

By examining the structure of (7), we can perceive the TE as a re-weighting scheme for sample means that is driven by statistical considerations. The pivotal aspect lies in the determination of weights through indicator functions. Notably, the derivatives of these indicator functions with respect to α are either zero or non-existent, resulting in either an exclusion or an inclusion of specific mean. Avoiding this behavior, we propose to apply the standard Gaussian cdf Φ directly to the test statistics and use the resulting values as a smoothed weighting:

$$\hat{\mu}_*^{\Phi} = \left[\sum_{i=1}^{M} \Phi\left(T_i\right)\right]^{-1} \sum_{i=1}^{M} \Phi\left(T_i\right)\hat{\mu}_i, \quad \text{where} \quad T_i = \frac{\hat{\mu}_i - \hat{\mu}_*^{\text{ME}}}{\sqrt{\frac{\hat{\sigma}_i^2}{|S_i|} + \frac{\hat{\sigma}_{i^*}^2}{|S_{i^*}|}}}. \tag{9}$$

In fact, it can be generalized even further by considering a weighting kernel $\kappa(\cdot)$:

$$\hat{\mu}_*^{\text{KE}} = \left[\sum_{i=1}^{M} \kappa\left(T_i\right)\right]^{-1} \sum_{i=1}^{M} \kappa\left(T_i\right)\hat{\mu}_i, \tag{10}$$

with $\kappa : (-\infty; 0] \to [0; \infty)$. We require that $\kappa(\cdot)$ is monotonically increasing to build a reasonable kernel since $T_i \leq 0$, $\forall i = 1, \ldots, M$, and that $\lim_{T_i \to -\infty} \kappa(T_i) = 0$ for consistency. Similar kernel functions are considered in Mammen [43] for isotonic regressions. We refer to (10) as the *K-Estimator* (KE). Crucially, the hyperparameter of the KE is not a fixed scalar anymore, but the specification of $\kappa(\cdot)$. For example, the TE is a special case of the KE obtained by setting $\kappa(T_i; \alpha) = \mathcal{I}(T_i \geq z_\alpha)$. We emphasize that re-weighting schemes of sample means are widely known in the form of softmax operators in the RL literature [2,57]. Although structurally similar, the KE uses the test statistics T_i to determine the weight of mean $\hat{\mu}_i$. The weights in conventional softmax approaches are determined only by the mean itself, thereby neglecting the uncertainty of the mean estimates.

Further options for $\kappa(\cdot)$ are listed in Table 1. Additionally, more flexible parametrized specifications are available. Consider, for example, the cdf of the beta distribution $\mathcal{B}_{\mathfrak{a},\mathfrak{b}}$ with two shape parameters $\mathfrak{a}$, $\mathfrak{b}$. Although the latter is naturally defined on $[0, 1]$, one could simply scale and shift it to, e.g., $[-5, 0]$ to generate a more valid specification. This particular case is used in the example of Section 4.1. Apart from that, we primarily apply the standard Gaussian cdf throughout the paper.

Bias and variance of the KE depend on the chosen kernel specification, but the bounds of the TE are still valid as long as the kernel function fulfills the requirements stated above.

Corollary 1. *For the KE, it holds:*

$$\min_i \mu_i - \max_i \mu_i - \sqrt{\frac{M-1}{M}\sum_{i=1}^{M} \operatorname{Var}\left(\hat{\mu}_i\right)} \leq \operatorname{Bias}\left(\hat{\mu}_*^{\mathrm{KE}}\right) \leq \operatorname{Bias}(\hat{\mu}_*^{\mathrm{ME}}).$$

Proof. The KE cannot exceed the ME, thus the upper bound holds. For the lower bound, we note the same relationship as for the TE:

$$\hat{\mu}_*^{\mathrm{KE}} \geq \min_i \hat{\mu}_i.$$

The sample mean corresponding to the ME is per construction weighted with $\kappa(0)$ (before normalization). Simultaneously, for extreme variance scenarios, it is possible that the weight of smaller means tends to $\kappa(0)$ as well, and, for sufficiently large M, the $\hat{\mu}_*^{\mathrm{KE}}$ thus might be arbitrarily close to $\min_i \hat{\mu}_i$. The remaining steps are identical to the proof of Lemma 1. □

Corollary 2. *For the KE, it holds:* $\operatorname{Var}\left(\hat{\mu}^{\mathrm{KE}}\right) \leq \sum_{i=1}^{M} \frac{\sigma_i^2}{|S_i|}$.

The proof is again similar to D'Eramo et al. [20]. Moreover, in Appendix B, we derive a general expression for the expectation of the KE for arbitrary M in the case of known variances.

Finally, we highlight three main differences between the KE (including the TE as a special case) and the WE of Section 2.4 since both are constructed as a weighted sum of sample means. First, the weights of the WE are probabilities, while the weights of the KE do not have a probabilistic interpretation. Second, while the KE allows for a multitude of specifications, the WE is not tunable and thus cannot be adjusted to a given scenario in a practical problem. Third, the computation of the WE requires integration or Monte Carlo approximation schemes, while the KE's computation is extremely fast.

4. On the role of dependencies

The previous exposition and the analytical properties of the estimators were predicated on the assumption of acquiring independent samples from independent random variables. However, in the realm of reinforcement learning, where we intend to apply these estimators, both of these independence assumptions often do not hold true. Specifically, the recursive nature of the algorithms and the bootstrapping of target values, which will be detailed in Section 5, can lead to the emergence of *in-sample dependencies*. For instance, the observations S_1 of a random variable X_1 might display auto-correlations. Moreover, given that different states can share successor states, *cross-sample dependencies* can manifest, implying correlations between the samples S_1 and S_2 of random variables X_1 and X_2, respectively.

Deriving general analytical bounds for the bias of the estimators under the presence of these two distinct types of dependencies is a challenging task. Therefore, we will undertake an empirical investigation of bias and variance of the considered estimators by introducing in-sample and cross-sample dependencies into two Gaussian random variables. At first, however, we will analyze the example if all independence assumptions are fulfilled.

4.1. Independence: a Gaussian example

We consider a similar setup to D'Eramo et al. [20] with $M = 2$ Gaussian random variables $X_1 \sim \mathcal{N}(\mu_1, \sigma^2)$, $X_2 \sim \mathcal{N}(\mu_2, \sigma^2)$, where $\sigma^2 = 100$ is the common known variance, and we have $|S_1| = |S_2| = 100$ observations of each variable. To stress again: We assume to have independent samples from independent random variables. We fix $\mu_2 = 0$ and compute bias, variance, and MSE for different $\mu_1 \in [0, 5]$. For completeness, we report the analytic forms for the expectation and variance of the estimators in this case in Appendix A. For the TE, we select significance levels $\alpha \in \{0.05, 0.10, 0.15\}$ and for the KE, we analyze the standard Gaussian kernel Φ, the Epanechnikov kernel, and the shifted and scaled $\mathcal{B}_{\mathfrak{a},\mathfrak{b}}$ cdf kernel with $\mathfrak{a} = 2$, $\mathfrak{b} = 0.5$ as described above. Results are displayed in Figs. 1 and 2.

Regarding the TE, we see how the bias decreases with a smaller significance level. In general, the estimator operates between the biases of ME and DE, although the bias of the DE is not necessarily a lower bound (see $\alpha = 0.05$). In the mean equality case, $\mu_1 = \mu_2$, the TE can avoid the significant overestimation of the ME while having only slightly increased variance. Consequently, the TE outperforms the conventional competitors for all considered significance levels under the MSE criterion for $\mu_1 = \mu_2$. However, if the difference of the true expectations of the random variables is large, all estimators become unbiased. In this scenario, the ME is the best choice due to its low variance. Regarding the KE in Fig. 2, we see that the standard Gaussian and Epanechnikov kernels can achieve a desirable balance between under- and overestimation while maintaining a smaller variance than the considered TE. The chosen specification of the beta distribution appears sub-optimal for this particular problem. In Appendix C, we optimize for a better fitting parametrization to further illustrate the flexibility of the KE.

Overall, we have seen through this investigation that both the TE and the KE can achieve flexible trade-offs between bias and variance in the estimation of the MEV. Considering typical levels of significance like 0.05, 0.10, or 0.15 in the TE builds a robust estimator, further enhanced by accurately specifying a suited KE.

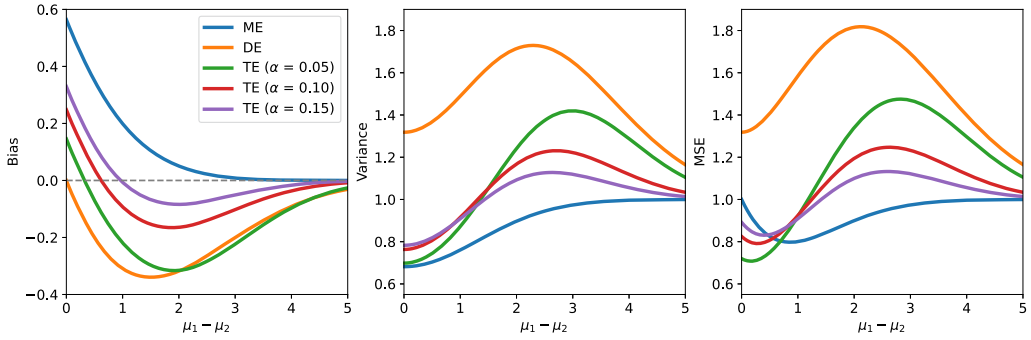


Fig. 1. Comparison of the ME, DE, and TE with level of significance in parentheses.

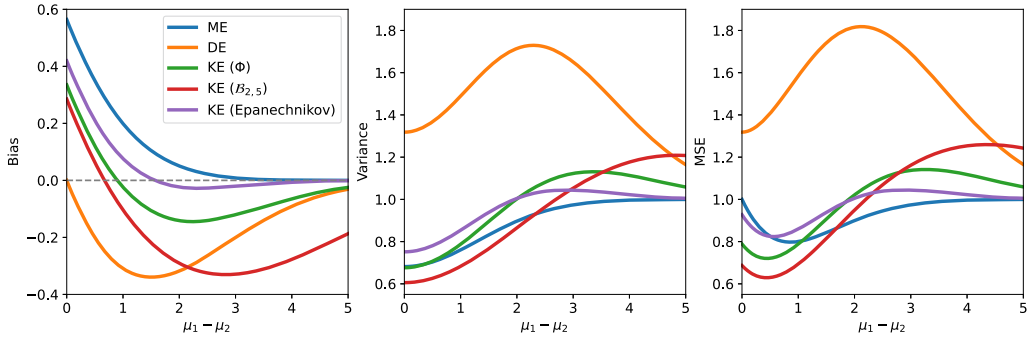


Fig. 2. Comparison of the ME, DE, and KE with kernel in parentheses.

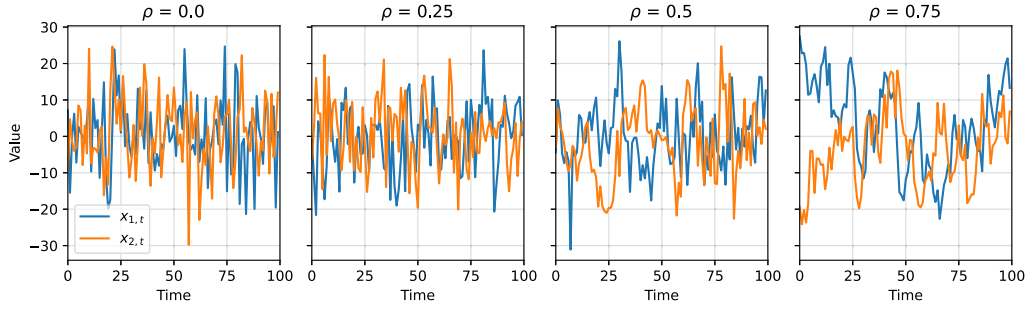


Fig. 3. Realizations of the processes in (11) for $\mu_1 = \mu_2 = 0$, $\sigma^2 = 100$, and varying ρ. While $\rho = 0.0$ represents independent random noise, the in-sample dependencies are visible for larger ρ through the prolonged increasing or decreasing patterns of the time series.

4.2. In-sample dependencies

In this section, we introduce in-sample dependencies into our two Gaussian random variables. Specifically, we will examine the following auto-correlated process, motivated from the literature on time series econometrics [59]:

$$X_{i,t} = (1-\rho)\mu_i + \rho X_{i,t-1} + \varepsilon_{i,t}, \tag{11}$$
$$\varepsilon_{i,t} \sim \mathcal{N}\left[0, (1-\rho^2)\sigma^2\right],$$

where $X_{i,0} = \mu_i + \varepsilon_{i,0}$. We set $\varepsilon_{i,0} \sim \mathcal{N}\left(0, \sigma^2\right)$, for $i = 1, 2$, and consider time steps $t = 1, \ldots, T$. Throughout the experiments, we set $T = 100$. This formulation ensures that the unconditional first two moments remain unchanged: $\mathrm{E}[X_{i,t}] = \mu_i$ and $\mathrm{Var}[X_{i,t}] = \sigma^2$. Similar to Section 4.1, there are still no cross-sample dependencies between the two processes. In particular, we have $\mathrm{Cov}(X_{1,t}, X_{2,t}) = 0$ for all t. However, there is now auto-correlation within each process, $\mathrm{Cov}(X_{i,t-1}, X_{i,t}) \neq 0$ for $i = 1, 2$, which can be modulated by the parameter $\rho \in [0, 1)$. Fig. 3 illustrates exemplary realizations of (11) for $\rho \in \{0.0, 0.25, 0.5, 0.75\}$.

We compute the ME, DE, TE ($\alpha \in \{0.05, 0.10, 0.20\}$), and the KE (standard Gaussian kernel) after simulating from (11) for different $\mu_1 \in [0, 5]$, while $\mu_2 = 0$ and $\sigma^2 = 100$ are fixed. The simulation and estimation procedure is repeated 10 000 times for each μ_1, and the results are used to estimate the bias and the variance of the respective estimator of the MEV. Fig. 4 displays the results.

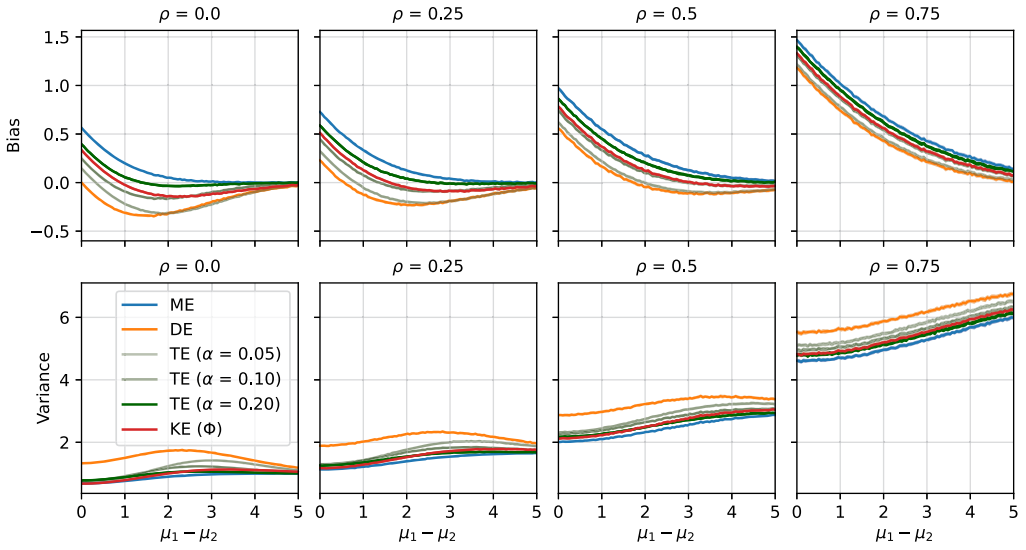


Fig. 4. Bias and variance of the ME, DE, TE, and KE when in-sample dependencies are present. The latter are introduced via autocorrelations, which are modulated by the parameter $\rho \in [0, 1)$. The larger ρ, the larger the in-sample dependence. Three different levels of significance (α) are displayed in green for the TE, where the green gets darker with increasing α. Point-wise 95% confidence intervals are included based on repeating the experiment in 30 independent runs. However, the intervals are very narrow. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

It is evident that the scenario with $\rho = 0.0$ aligns with the analytical outcomes depicted in Figs. 1 and 2, thus providing a validation of the derivations in Appendix A through simulation. However, all estimators exhibit increasing bias and variance when the in-sample dependencies increase. This observation can be explained by the fact that introducing autocorrelation has a similar effect as reducing the sample size, which can be seen as follows: If $\rho = 0$, the variance of the mean estimate of the process (11) is $\text{Var}\left(T^{-1}\sum_{t=1}^{T} X_{i,t}\right) = T^{-2}\sum_{t=1}^{T}\text{Var}\left(X_{i,t}\right) = T^{-1}\sigma^2$, where the first step holds since the $X_{i,t}$ are independent. Now assume $\rho > 0$. This leads to: $\text{Var}\left(T^{-1}\sum_{t=1}^{T} X_{i,t}\right) = T^{-2}\left[\sum_{t=1}^{T}\text{Var}\left(X_{i,t}\right) + \sum_{t=1}^{T}\sum_{\substack{\tilde{t}=1 \\ \tilde{t}\neq t}}^{T}\text{Cov}\left(X_{i,t}, X_{i,\tilde{t}}\right)\right] > T^{-1}\sigma^2$. Due to the positive covariance terms, the variance of the mean estimates is increased. Consequently, the estimates of the MEV are less accurate and have an increased variance. Intuitively, each new realization of the process (11) contains less new information than the $\rho = 0$ case due to the dependence on the past.

However, we observe that the relative ranking of the estimators in terms of both bias and variance remains unchanged. In general, the ME consistently exhibits the highest bias, while the bias of the TE decreases monotonically with smaller α.

4.3. Cross-sample dependencies

Regarding cross-sample dependencies, we model the two random variables with a bivariate Gaussian distribution, instead of considering two independent univariate Gaussians. In particular, we set:

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim \mathcal{N}\left\{\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \sigma^2 & \rho_G\sigma^2 \\ \rho_G\sigma^2 & \sigma^2 \end{pmatrix}\right\}, \tag{12}$$

where $\rho_G \in [0, 1]$ is the correlation parameter of the distribution. The unconditional first two moments are identical to Sections 4.1 and 4.2, no in-sample dependencies are contained, and the simulation conditions are set to mimic the situation of the previous case. However, the cross-sample dependence can now be increased by increasing ρ_G. Fig. 5 shows different realizations of (12), while Fig. 6 displays the results.

Analyzing the ME and DE, we see that the increased correlation between the samples is generally beneficial in reducing the absolute value of the bias and the variance of the estimators. However, a different behavior is observed in the TE and KE, which are more sensitive to cross-sample dependencies. In particular, a positive correlation leads to an underestimation of the MEV and an increased variance, although the effects of cross-sample dependencies on bias and variance are generally not nearly as pronounced as the ones of in-sample dependencies.

This behavior can be elucidated by examining the construction of the estimators. The underlying test statistics, as defined in (6), assumes independence between the random variables, implying $\text{Var}(\hat{\mu}_1 - \hat{\mu}_2) = \text{Var}(\hat{\mu}_1) + \text{Var}(\hat{\mu}_2)$. However, this assumption no longer holds true in scenarios with $\rho_G > 0$. More precisely, in these scenarios, (6) overestimates $\text{Var}(\hat{\mu}_1 - \hat{\mu}_2)$, leading to the consideration of smaller means and, consequently, the observed underestimation of the TE and KE. To account for this circumstance, one could

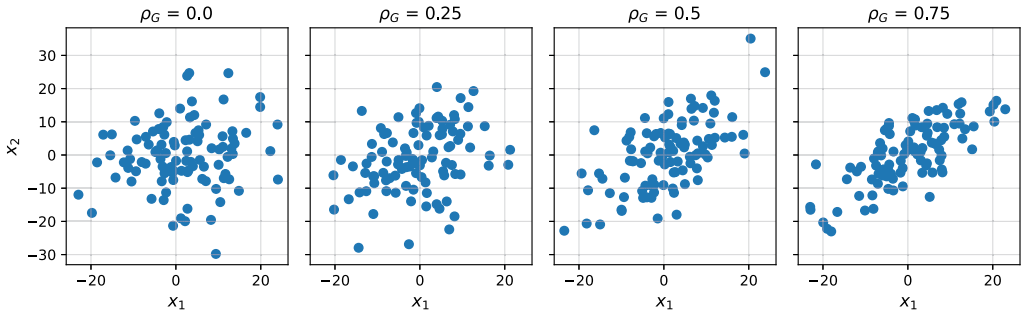


Fig. 5. Realizations of the bivariate Gaussian distribution defined in (12) for $\mu_1 = \mu_2 = 0$, $\sigma^2 = 100$, and varying ρ. Setting $\rho_G = 0$ means that the two samples are independent, while the cross-sample correlation gets stronger with larger ρ_G.

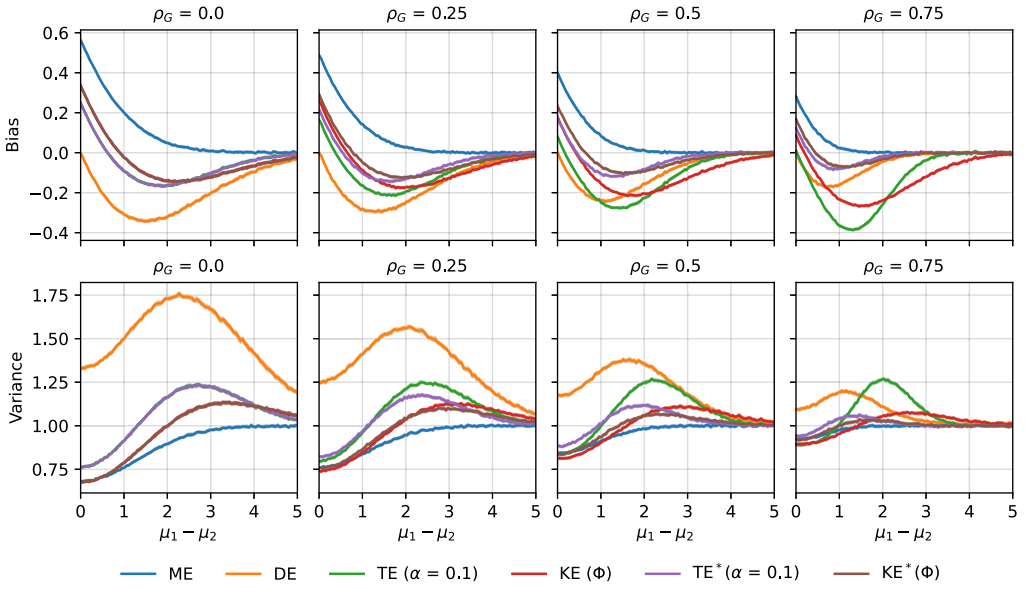


Fig. 6. Bias and variance of the ME, DE, TE, and KE when cross-sample dependencies in the form of the correlation parameter $\rho_G \in [0, 1]$ of a bivariate Gaussian are introduced. The larger ρ_G, the larger the cross-sample dependence. We include versions of the TE and KE which contain a covariance correction during the computation of the test statistics. These are denoted TE* and KE*, respectively. For the case $\rho_G = 0.0$, the curves of the TE* and KE* strongly overlap with those of the TE and KE, respectively. Point-wise 95% confidence intervals are included based on repeating the experiment in 30 independent runs. However, the intervals are very narrow.

implement a covariance correction and replace the test statistics of TE and KE as given in (6) with:

$$T^* = \frac{\hat{\mu}_1 - \hat{\mu}_2}{\sqrt{\frac{\hat{\sigma}_1^2}{|S_1|} + \frac{\hat{\sigma}_2^2}{|S_2|} - 2 \cdot \widehat{\mathrm{Cov}}(S_1, S_2)}}, \tag{13}$$

using the notation of Section 2 and with $\widehat{\mathrm{Cov}}(S_1, S_2)$ being the covariance estimate from samples S_1 and S_2. We include the resulting covariance-corrected TE and KE in Fig. 6, denoted as TE* and KE*. Due to the correction, the dependence sensitivity is alleviated, and the expected desirable behavior materializes.

4.4. Discussion

Both in-sample and cross-sample dependencies can impact the properties of the estimators of the MEV, although the effects of in-sample dependencies on bias and variance appear to be much stronger. Importantly, we observed that the *relative ordering* of the estimators to each other remains the same for these crucial in-sample dependencies, implying that the choice of the estimator of the MEV is still critical in the presence of temporal dependencies. Similar to the works of Van Hasselt [61], D'Eramo et al. [20], and Zhu and Rigotti [72], we will transfer the estimators of the MEV to the domain of reinforcement learning in the next section.

5. Application to reinforcement learning

5.1. Tabular version

Reinforcement learning describes a collection of learning techniques for sequential decision processes, in which an agent aims to maximize its reward while interacting with an environment; see Sutton and Barto [57] and Bertsekas [10]. The problem is modeled as a Markov Decision Process (MDP, [49]), consisting of a state space $\mathcal{S}$, a finite action space $\mathcal{A}$, a state transition probability distribution $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$, a bounded reward distribution $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathcal{P}_{\mathbb{R}}$, where $\mathcal{P}_{\mathbb{R}}$ is the set of probability distributions over $\mathbb{R}$, and a discount factor $\gamma \in [0,1]$. If $\gamma = 1$, we assume there is a zero-reward absorbing state and that the probability of reaching this state converges to one as time tends to infinity; see Lan et al. [37]. At each time step t, the agent takes an action $a_t \in \mathcal{A}$ based on state information $s_t \in \mathcal{S}$, receives a reward $r_t \sim \mathcal{R}(s_t, a_t)$, and transitions with probability $\mathcal{P}(s_{t+1} \mid s_t, a_t)$ to a new state $s_{t+1} \in \mathcal{S}$. Objective is to optimize for a policy $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$, a mapping from states to distributions over actions, that maximizes the expected return $\mathrm{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$. Value-based methods, which are very common RL approaches, define action-values $Q^\pi(s,a) = \mathrm{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a\right]$ for a certain policy. Thus, $Q^\pi(s,a)$ is the expected return when starting in state s, executing a, and following policy π afterwards. There exists an optimal deterministic stationary policy $\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s,a)$, that is connected with optimal action-values $Q^*(s,a) = \max_\pi Q^\pi(s,a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$ if the state space is finite or countably infinite [49, Theorem 6.2.10]. To optimize for $Q^*(s,a)$, one uses a recursive relationship known as Bellman [8] optimality equation:

$$Q^*(s,a) = \mathcal{R}(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}(s' \mid s,a) \max_{a' \in \mathcal{A}} Q^*(s',a'), \tag{14}$$

where s' is a successor state after performing action a in state s. Since $Q^*(s',a')$ is the expected return from executing a' in s' and following the optimal policy afterward, the problem immediately appears as an instance of the estimation of the MEV of a set of random variables, namely the stochastic returns. Consequently, the methodology of Section 2 applies. Q-Learning, from Watkins and Dayan [68], translates (14) into a sample-based algorithm by using the ME:

$$\hat{Q}^*_{t+1}(s_t,a_t) \leftarrow \hat{Q}^*_t(s_t,a_t) + \tau_t(s_t,a_t)\left[y_t^Q - \hat{Q}^*_t(s_t,a_t)\right],$$

with target $y_t^Q = r_t + \gamma \max_{a \in \mathcal{A}} \hat{Q}^*_t(s_{t+1},a)$ and learning rate $\tau_t(s_t,a_t)$. An estimate of $Q^*(s_t,a_t)$ at time t is denoted $\hat{Q}^*_t(s_t,a_t)$. The algorithm is known to converge to the optimal action-values if the conditions of Robbins and Monro [50] on the learning rate are fulfilled, and each state-action pair is visited infinitely often [60]. However, especially in early stages of the training when the Q-estimates are imprecise, the algorithm tends to transmit overestimated values. Overcoming this issue, Van Hasselt [61] uses the DE and stores two separate Q-tables with estimates $\hat{Q}^*_{A,t}$ and $\hat{Q}^*_{B,t}$, leading to the target: $y_t^{DQ} = r_t + \gamma \hat{Q}^*_{B,t}\left[s_{t+1}, \arg\max_{a \in \mathcal{A}} \hat{Q}^*_{A,t}(s_{t+1},a)\right]$. To apply the TE and KE in a Q-Learning setup, we propose to replace the target with $y_t^{KQ} = r_t + \gamma\, \mathrm{KE}_{\mathrm{a}}\, \hat{Q}^*_t(s_{t+1},a)$, where:

$$\begin{aligned}
\mathrm{KE}_{\mathrm{a}}\, \hat{Q}^*_t(s_{t+1},a) &= \left\{ \sum_{a \in \mathcal{A}} \kappa\left[T_{\hat{Q}^*_t}(s_{t+1},a)\right] \right\}^{-1} \sum_{a \in \mathcal{A}} \kappa\left[T_{\hat{Q}^*_t}(s_{t+1},a)\right] \hat{Q}^*_t(s_{t+1},a), \\
T_{\hat{Q}^*_t}(s_{t+1},a) &= \frac{\hat{Q}^*_t(s_{t+1},a) - \max_{a' \in \mathcal{A}} \hat{Q}^*_t(s_{t+1},a')}{\sqrt{\widehat{\mathrm{Var}}_t\left[\hat{Q}^*_t(s_{t+1},a)\right] + \widehat{\mathrm{Var}}_t\left[\hat{Q}^*_t(s_{t+1},a^*)\right]}},
\end{aligned} \tag{15}$$

for a maximizing action $a^* \in \{a \in \mathcal{A} \mid \hat{Q}^*_t(s_{t+1},a) = \max_{a' \in \mathcal{A}} \hat{Q}^*_t(s_{t+1},a')\}$. The variance estimate of $\hat{Q}^*_t(s_{t+1},a)$ for some action $a \in \mathcal{A}$ at time t is denoted $\widehat{\mathrm{Var}}_t\left[\hat{Q}^*_t(s_{t+1},a)\right]$ and will be generated following the proposal of D'Eramo et al. [19]. First, the process variance of the underlying return of the visited state-action pair (s_t,a_t) is estimated via an exponentially-weighted online update:

$$\widehat{\sigma^2}_{\mathrm{pro},t+1}(s_t,a_t) \leftarrow \left[1 - \tau_t(s_t,a_t)\right]\left\{\widehat{\sigma^2}_{\mathrm{pro},t}(s_t,a_t) + \tau_t(s_t,a_t)\left[y_t^{KQ} - \hat{Q}^*_t(s_t,a_t)\right]^2\right\}. \tag{16}$$

Second, to get the variance estimate in (15) for some $a \in \mathcal{A}$, a normalization by Kish [34] effective sample size $n_{\mathrm{eff},t}(s_{t+1},a)$ is performed:

$$\widehat{\mathrm{Var}}_t\left[\hat{Q}^*_t(s_{t+1},a)\right] = \frac{\widehat{\sigma^2}_{\mathrm{pro},t}(s_{t+1},a)}{n_{\mathrm{eff},t}(s_{t+1},a)}.$$

The effective sample size weights each sample depending on the learning rate and is computed via $n_{\mathrm{eff},t}(s_{t+1},a) = \frac{\left[\omega_t(s_{t+1},a)\right]^2}{\omega_t^2(s_{t+1},a)}$, in which numerator and denominator are incrementally updated for each visited state-action pair (s_t,a_t):

$$\begin{aligned}
\omega_{t+1}(s_t,a_t) &\leftarrow [1 - \tau_t(s_t,a_t)]\omega_t(s_t,a_t) + \tau_t(s_t,a_t), \\
\omega^2_{t+1}(s_t,a_t) &\leftarrow [1 - \tau_t(s_t,a_t)]^2 \omega^2_t(s_t,a_t) + [\tau_t(s_t,a_t)]^2.
\end{aligned} \tag{17}$$

With this approach, we can introduce TE-Q-Learning (TE-Q) and KE-Q-Learning (KE-Q), respectively, being summarized in Algorithm 1. The following theorem states the convergence of the algorithm to the optimal action-values with probability 1.

Theorem 1. *Let the following regularity conditions be fulfilled:*

1. *The MDP is finite.*
2. $\gamma \in [0, 1)$.
3. *The learning rates satisfy* $\tau_t(s, a) \in [0, 1]$, $\sum_t \tau_t(s, a) = \infty$, $\sum_t \tau_t^2(s, a) < \infty$ *all with probability 1 for all* $s \in \mathcal{S}, a \in \mathcal{A}$.
4. *The reward function is bounded.*
5. *Each state-action pair is visited infinitely often.*

Then the following holds for the random sequence of action-value estimates $\hat{Q}_t^*$ *generated by TE/KE-Q-Learning:*

$$\mathbb{P}\left[\lim_{t\to\infty} \hat{Q}_t^*(s, a) = Q^*(s, a)\right] = 1 \quad \forall s \in \mathcal{S}, a \in \mathcal{A}.$$

Proof. Please refer to Appendix D. □

We emphasize that the proof scheme can be similarly applied to Weighted Q-Learning [20], for which a proof of convergence has been missing so far.

As a note, we mention the possibility of incorporating a covariance correction, as shown in (13), within the definition of the target in (15). Implementing such an approach would entail updating the covariances between the return estimates for all actions of each state and can be possibly constructed analogous to (16). However, assume that only two actions, A and B, are available in some state. To estimate the return covariance between these actions, one needs to execute A, compute its target, and do the same for B. Although a parallelized solution is straightforward in a simulation environment, for the sake of simplicity and taking into account the insights from Section 4.1 that suggest cross-sample dependencies are not as critical, we opt to retain the test statistics as defined in (15).

Algorithm 1: TE-Q-Learning/KE-Q-Learning.

initialize $\forall s \in \mathcal{S}, a \in \mathcal{A}$: $\hat{Q}^*(s, a) = 0, \widehat{\sigma^2}_{\text{pro}}(s, a) > 0, \omega(s, a) \in (0, 1], \omega^2(s, a) \in (0, 1]$
repeat
  Initialize s
  repeat
    Choose action a from state s with policy derived from $\hat{Q}^*$ (e.g., ϵ-greedy)
    Take action a, observe reward r and next state s'
    Update effective sample size:
      $\omega(s, a) \leftarrow (1 - \tau)\omega(s, a) + \tau$
      $\omega^2(s, a) \leftarrow (1 - \tau)^2\omega^2(s, a) + \tau^2$
      $n_{\text{eff}}(s, a) \leftarrow \frac{\omega(s,a)^2}{\omega^2(s,a)}$
    Calculate target y_{KQ}
    Update process variance:
      $\widehat{\sigma^2}_{\text{pro}}(s, a) \leftarrow (1 - \tau)\left\{\widehat{\sigma^2}_{\text{pro}}(s, a) + \tau\left[y^{KQ} - \hat{Q}^*(s, a)\right]^2\right\}$
    Update Q-estimate:
      $\hat{Q}^*(s, a) \leftarrow \hat{Q}^*(s, a) + \tau\left[y^{KQ} - \hat{Q}^*(s, a)\right]$
    $s \leftarrow s'$
  until s *is terminal*
until

5.2. Deep version

Real-world control tasks frequently entail continuous state spaces, which necessitates to employ strategies such as state aggregation [53] to make tabular algorithms applicable. In recent research, there has been a growing emphasis on leveraging DNNs as function approximators to parameterize the function $\hat{Q}_t^*(s_t, a_t; \theta_t)$, with θ_t being the parameter set of the neural network at time t. The resulting DQN [44] and their extensions [63,14,27] have shown breakthrough performances on various challenging tasks [6,4], thereby revolutionizing the capabilities of modern artificial intelligence.

The optimization procedure of these algorithms is still based on the Bellman optimality equation (14), but uses gradient descent to update θ_t:

$$\theta_{t+1} \leftarrow \theta_t + \tau_t\left[y_t^{DQN} - \hat{Q}_t^*(s_t, a_t; \theta_t)\right]\nabla_{\theta_t}\hat{Q}_t^*(s_t, a_t; \theta_t),$$

where $y_t^{DQN} = r_t + \gamma \max_{a\in\mathcal{A}} \hat{Q}_t^*(s_{t+1}, a; \theta_t^-)$. The set θ_t^- refers to the parameters of the target network, which is a time-delayed copy of the main network with parameter θ_t. Instead of updating fully online, DQN samples minibatches of past experiences from a replay

buffer D to stabilize training. Van Hasselt et al. [63] proposed the Double DQN (DDQN) and uses the DE to compute the target: $y_t^{DDQN} = r_t + \gamma \hat{Q}_t^*[s_{t+1}, \arg\max_{a\in\mathcal{A}} \hat{Q}_t^*(s_{t+1}, a; \theta_t); \theta_t^-]$. The action selection is performed via the main network, while the evaluation uses the target network like the regular DQN.

To translate the TE and KE to the function approximation case, we require a variance estimate of the Q-estimates. We follow D'Eramo et al. [19] and use the framework of the BDQN [47] to accomplish this task. Generally, the bootstrap is a method for computing measures of accuracy for statistical estimates [21]. The method trains different regressors of the target function based on bootstrap samples generated by sampling with replacement from the original dataset. The BDQN transfers this idea to the DQN algorithm by maintaining $K \in \mathbb{N}$ differently initialized Q-networks $\hat{Q}_{k,t}^*(s_t, a_t; \theta_{k,t})$ with parameter $\theta_{k,t}$ at time t and $k = 1, \dots, K$, each equipped with its own target network $\hat{Q}_{k,t}^*(s_t, a_t; \theta_{k,t}^-)$. This DQN modification was proposed to improve over the usual ϵ-greedy exploration strategy. At the beginning of each episode, one $\hat{Q}_{k,t}^*$ is selected randomly, and the agent acts greedily with relation to this $\hat{Q}_{k,t}^*$. At test time, the majority vote of the function approximators is used.

Generally, the BDQN can be implemented using K different networks or maintaining a common network body and specifying K different heads. We pursue the latter approach. Diversification across the heads is achieved by different random initialization of the parameters and random generation of binary masks $m_k^i \in \{0, 1\}$, where i refers to the i-th transition tuple of the replay buffer, and $k = 1, \dots, K$. The masks indicate which head should be trained on which sample. More precisely, if some tuple i is sampled from the buffer, the k-th head of the ensemble will only be trained if m_k^i, the k-th mask associated with tuple i, takes value 1. All masks are typically sampled from the same distribution, and a Bernoulli distribution with parameter p is a possible choice.

Crucially, through the K heads, we can directly use the sample variance of the Q-estimates and compute the target $y_{k,t}^{KDQN}$ for the k-th network $\hat{Q}_{k,t}^*$:

$$y_{k,t}^{KDQN} = r_t + \gamma \left\{ \sum_{a\in\mathcal{A}} \kappa \left[T_{\hat{Q}_{k,t}^*}(s_{t+1}, a) \right] \right\}^{-1} \sum_{a\in\mathcal{A}} \kappa \left[T_{\hat{Q}_{k,t}^*}(s_{t+1}, a) \right] \hat{Q}_{k,t}^*(s_{t+1}, a; \theta_{k,t}^-), \tag{18}$$

where

$$T_{\hat{Q}_{k,t}^*}(s_{t+1}, a) = \frac{\hat{Q}_{k,t}^*(s_{t+1}, a; \theta_{k,t}^-) - \max_{a'\in\mathcal{A}} \hat{Q}_{k,t}^*(s_{t+1}, a'; \theta_{k,t}^-)}{\sqrt{\widehat{\mathrm{Var}}_t \left[\hat{Q}_{k,t}^*(s_{t+1}, a; \theta_{k,t}^-) \right] + \widehat{\mathrm{Var}}_t \left[\hat{Q}_{k,t}^*(s_{t+1}, a^*; \theta_{k,t}^-) \right]}}, \tag{19}$$

for an action $a^* \in \{a \in \mathcal{A} \mid \hat{Q}_{k,t}^*(s_{t+1}, a; \theta_{k,t}^-) = \max_{a'\in\mathcal{A}} \hat{Q}_{k,t}^*(s_{t+1}, a'; \theta_{k,t}^-)\}$. The resulting gradient $g_{k,t}^i$ for the i-th tuple from the replay buffer is:

$$g_{k,t}^i = m_k^i \left[y_{k,t}^{KDQN,i} - \hat{Q}_{k,t}^*(s_i, a_i; \theta_{k,t}) \right] \nabla_{\theta_{k,t}} \hat{Q}_{k,t}^*(s_i, a_i; \theta_{k,t}). \tag{20}$$

The full procedure is termed TE-BDQN and KE-BDQN, respectively, being detailed in Algorithm 2. Similar to the tabular case, there is the possibility to include a covariance correction in the denominator of (19), thereby combating possible cross-sample dependencies. Although the implementation is straightforward due to the bootstrap ensemble, we propose a more holistic adaptive bias reduction scheme in the next section, which goes beyond mitigating the consequences of cross-sample dependencies.

Algorithm 2: TE-BDQN/KE-BDQN.

initialize Action-value estimate networks with K outputs $\{\hat{Q}_k^*\}_{k=1}^K$, masking distribution M, empty replay buffer D
repeat
  Initialize s
  Pick a value function to act: $k \sim \text{Uniform}\{1, \dots, K\}$
  repeat
    Choose action a from state s with greedy policy derived from $\hat{Q}_k^*$
    Take action a, observe reward r and next state s'
    Sample bootstrap masks $m = (m_1, \dots, m_K)$
    Add (s, a, r, s', m) to replay buffer D
    Sample random minibatch of transitions $\{(s_i, a_i, s_i', r_i, m^i)\}_{i=1}^B$ from D
    Perform gradient descent step based on (20)
    Every C steps reset $\theta_k^- = \theta_k$ for $k = 1, \dots, K$
    $s \leftarrow s'$
  until *s is terminal*
until

6. Adaptive absolute bias minimization

While the TE can interpolate between under- and overestimation by selecting a smaller or larger α, it is a priori not known which α is adequate for an unknown environment. The method of choice for practitioners in such situations is a grid search to select a suitable value empirically. Next to the selection issue, a fixed parameter might not be sufficient for controlling the estimation bias, and, e.g., an ascending strategy would be favorable. Leveraging these considerations, we propose an adaptive modification of TE-BDQN to adjust α under the objective of minimizing the absolute estimation bias during training.

6.1. Bias estimation

Before introducing the adaptive mechanism, we first outline how to estimate the bias of given action-value estimates at a certain point in training. Following Chen et al. [12], we consider a current policy π, which is connected with true action-values $Q^\pi(s,a)$ for $s \in \mathcal{S}, a \in \mathcal{A}$. The aggregated bias of estimates $\hat{Q}^\pi(s,a)$ of $Q^\pi(s,a)$ for all $s \in \mathcal{S}, a \in \mathcal{A}$ is defined as:

$$\text{Bias}(\hat{Q}^\pi, \pi) = \text{E}_{s\sim\rho^\pi, a\sim\pi}[\hat{Q}^\pi(s,a) - Q^\pi(s,a)],$$

where ρ^π is the state-visitation distribution of π. Chen et al. [12] proposed to repeatedly run analysis episodes from random initial states while following π. The observed Monte Carlo return of an encountered state-action pair serves as an unbiased estimate of its true Q-value. Averaging over all encountered state-action pairs yields the estimate of the estimation bias:

$$\widehat{\text{Bias}}(\hat{Q}^\pi, \pi) = \frac{1}{|\mathcal{T}|} \sum_{(s,a,R)\in\mathcal{T}} [\hat{Q}^\pi(s,a) - R], \tag{21}$$

where $\mathcal{T}$ is a set of encountered (s,a,R)-tuples, where s is the state, a the executed action, R the (later) observed Monte Carlo return, and $|\mathcal{T}|$ is the cardinality of $\mathcal{T}$. Chen et al. [12] applied this procedure to the Soft-Actor Critic algorithm [25], which uses an actor dictating the policy π and a critic providing the estimates $\hat{Q}^\pi$. In standard Q-Learning, no explicit actor is providing the policy, and the algorithm directly approximates the optimal action-values $Q^*(s,a)$, leading to estimates $\hat{Q}^*(s,a)$ [57]. To still generate insights into the action-value estimation accuracy of the algorithm, Van Hasselt et al. [63] compare the $\hat{Q}^*(s,a)$ generated *during* training with Monte Carlo returns of the final greedy policy *after* training. Although this approach is certainly instructive, it does not enable an assessment without having a converged baseline.

We instead propose to use (21) with the policy being defined as $\pi(s) = \arg\max_a \hat{Q}^*(s,a)$ for all $s \in \mathcal{S}$, being briefly summarized in Algorithm 3. Crucially, the algorithm can be run already *during* training, say always after a fixed number of steps, allowing to get an intuition about the development of the value estimation bias over the training time. The method is justified since Q-Learning uses a greedy target policy in its update. Thus, the algorithm *evaluates* the greedy policy with respect to its own action-value estimates, and we can assess whether the Q-estimates are too optimistic or pessimistic. Crucially, we emphasize that the greedy bias estimation roll-outs of Algorithm 3 are independent of the learning episodes, which rely on an explorative policy. Transferred to the BDQN and its modifications, the procedure can be similarly applied by assessing each head separately since they constitute different estimates $\hat{Q}^*_k$ for $k = 1, \dots, K$. Thus, we run Algorithm 3 for each head and average the output to receive an aggregated bias estimate for the bootstrap-ensemble.

Algorithm 3: Bias estimation for Q-Learning like algorithms.

Input Estimates $\hat{Q}^*(s,a)$ for all $s \in \mathcal{S}, a \in \mathcal{A}$
Set $\mathcal{T} = \emptyset$ and $\pi(s) = \arg\max_a \hat{Q}^*(s,a)$ for all $s \in \mathcal{S}$
for *number of episodes* **do**
  Randomly initialize s
  Play episode following π and append encountered state-action-return tuples to $\mathcal{T}$
end
return $\frac{1}{|\mathcal{T}|} \sum_{(s,a,R)\in\mathcal{T}} [\hat{Q}^*(s,a) - R]$

6.2. Adaptive TE-BDQN

Algorithm 3 will be used to assess the algorithms in the experiments of Section 7. Furthermore, the approach serves as a basis for dynamically adjusting the α of the TE-BDQN. Intuitively, since larger α leads to larger Q-estimates, α is reduced if the Q-estimates are too high. Vice versa, we increase α if the Q-estimates are too small. Precisely, we perform the following update:

$$\alpha_{t+1} \leftarrow \alpha_t + \frac{\tau_{\text{Ada}}}{K} \sum_{k=1}^{K} \sum_{\tilde{t}=1}^{T_{\text{Ada}}} \left[R_k(s_{\tilde{t},k}, a_{\tilde{t},k}) - \hat{Q}^*_{k,t}(s_{\tilde{t},k}, a_{\tilde{t},k}; \theta_k) \right], \tag{22}$$

with step size τ_{Ada} and roll-out length T_{Ada}. Importantly, we use n-steps returns [57]:

$$R_k(s_{\tilde{t},k}, a_{\tilde{t},k}) = r_{\tilde{t},k} + \gamma r_{\tilde{t}+1,k} + \gamma^2 r_{\tilde{t}+2,k} + \ldots + \gamma^{T_{\text{Ada}}-\tilde{t}} r_{T_{\text{Ada}},k} + \gamma^{T_{\text{Ada}}-\tilde{t}+1} \max_a \hat{Q}^*_{k,t}(s_{T_{\text{Ada}}+1,k}, a; \theta_k),$$

where $\tilde{t} = 1, \ldots, T_{\text{Ada}}$. Consistent with the notation above, we use the index t to refer to step size of the overall experiment, and $\tilde{t}$ to the step size inside a Monte Carlo roll-out. We denote state and action at time $\tilde{t}$ under head k as $s_{\tilde{t},k}$ and $a_{\tilde{t},k}$, respectively. The resulting immediate reward is $r_{\tilde{t},k}$ and the initial states $s_{1,k}$ for $k = 1, \ldots, K$ are randomly sampled from the replay buffer. As motivated in Section 6.1, the actions $a_{\tilde{t},k}$ under head k are selected by acting greedily with relation to $\hat{Q}^*_{k,t}$. Although n-step returns are generally not unbiased estimates of the expected return of a policy like a complete Monte Carlo roll-out, we found them empirically much more practicable since they do not require running full episodes while still allowing us to judge the accuracy of the current value estimates. Consequently, this approach can also be applied to non-episodic problems.

We update α immediately after the target networks, avoiding instabilities in the learning process. A similar proposal to (22) in an episodic context with continuous action spaces based on full Monte Carlo roll-outs has recently been made by Dorka et al. [17]. Note that it is possible to maintain a separate α for each bootstrap head, enabling a tailored parametrization for the bias of each approximator. However, we only consider one parameter for the whole ensemble for simplicity in the following. The resulting algorithm is called Ada-TE-BDQN and is shown in Appendix E.

7. Experiments

We analyze the proposed estimators of the MEV on a statistically motivated real-world example before considering two tabular environments that serve as a proof-of-concept for TE/KE-Q-Learning. The experiments with function approximation are carried out in the MinAtar environments of Young and Tian [69], which allow for a thorough algorithmic comparison.

7.1. Internet ads

We consider the internet ad problem previously studied by Van Hasselt [62], D'Eramo et al. [15], and Jiang et al. [32]. There are M different ads, and each has the same return per click. Consequently, the click rate is the only quantity of interest and is modeled as a Bernoulli variable (click or no click). The true expectations $\mu_1, \ldots, \mu_M$ of these M variables equal the respective click probability and are equally spaced in a specific interval μ_{int}. We consider N customers and assume that the ads are presented equally often to have N/M samples per ad. Thus, the sample size for each of the M Bernoulli variables is $N/M \gg 1$, and we refer to this ratio as the number of impressions. In addition, we assume N/M is integer-valued. The objective is to estimate the maximum true mean accurately, thus finding the best ad based on the given samples. We compare the TE ($\alpha = 0.1$) and the KE (standard Gaussian kernel) with the ME, DE, and WE based on bias, variance, and MSE. Six configurations of the problem are considered by varying the number of customers N, the number of ads M, or the upper limit of the sampling interval μ_{int}, while the lower limit is always fixed at 0.02. Fig. 7 displays the results.

Both TE and KE yield lower MSE than their competitors in most scenarios. We emphasize that $\alpha = 0.1$ was not cherry-picked for this problem, and the TE's performance could thus be further increased by tailoring α for each experiment. In general, all estimators' MSEs decrease with an increasing number of ads, while they are more accurate with a higher number of customers N, constituting reasonable observations. The DE often yields large variances, while the ME provides biased estimates. Despite producing a higher MSE than TE and KE in most cases, the WE outperforms the conventional competitors ME and DE.

7.2. Maximization bias example

We consider the example in Figure 6.5 of Sutton and Barto [57]. A simple MDP with two non-terminal states A and B is given, which is visualized in Fig. 8. The agent starts in A. If it goes *right* from A, the episode ends, and zero reward is received. If action *left* is selected in A, the agent deterministically gets to state B and receives zero reward. There are eight actions to choose from B, but all lead to a terminal state and yield a reward sampled from $\mathcal{N}(-0.1, 1)$. The parameters are $\gamma = 1$, $\epsilon = 0.1$, and learning rate $\tau = 0.1$. Since this is an undiscounted task, the expected return starting with action *left* is -0.1, and the agent should always prefer going *right*. However, due to the random selection of the ϵ-greedy strategy, the *left* action will always be picked at least 5% in expectation.

Fig. 9 depicts the results. The upper-left part displays the percentage of selecting action *left* in A; the upper-right plot contains the same percentage after 500 training episodes. The lower-right graph shows the estimate of Q^*(A, left) over training. Finally, the lower-left plot displays the number of non-rejected hypotheses of TE-Q-Learning when updating $\hat{Q}^*$(A, left) according to (15), which is equal to the number of Q-estimates of the follow-up state B that are averaged during the target computation. Emphasizing the relationship to (7), we refer to this quantity as the number of averaged means. Q-Learning is the same algorithm as TE-Q-Learning with $\alpha = 0.5$ and only included for comparison.

Q-Learning initially overestimates the value of the *left* action in state A and selects it nearly 10% of the cases after 500 episodes, which is twice as optimal considering that $\epsilon = 0.1$. Double Q-Learning performs better and achieves a final rate of roughly 6%. On the other hand, TE-Q-Learning can modulate the overestimation bias through its significance level α and reaches a near-optimal selection percentage for $\alpha \leq 0.10$. Interestingly, TE-Q ($\alpha = 0.4$) performs worse after 500 episodes than Q-Learning. Although the initial overestimation is not as large as for $\alpha = 0.5$, the effect is more persistent, and more interactions are needed to reduce the estimate. For additional insights, we display the number of means averaged by the TE. TE-Q-Learning with $\alpha = 0.5$ naturally considers only the maximum sample mean, which is non-unique in the first several episodes since all action-value estimates are initialized with zero. The lower the significance level of TE-Q gets, the more means are averaged until nearly all sample means are considered for

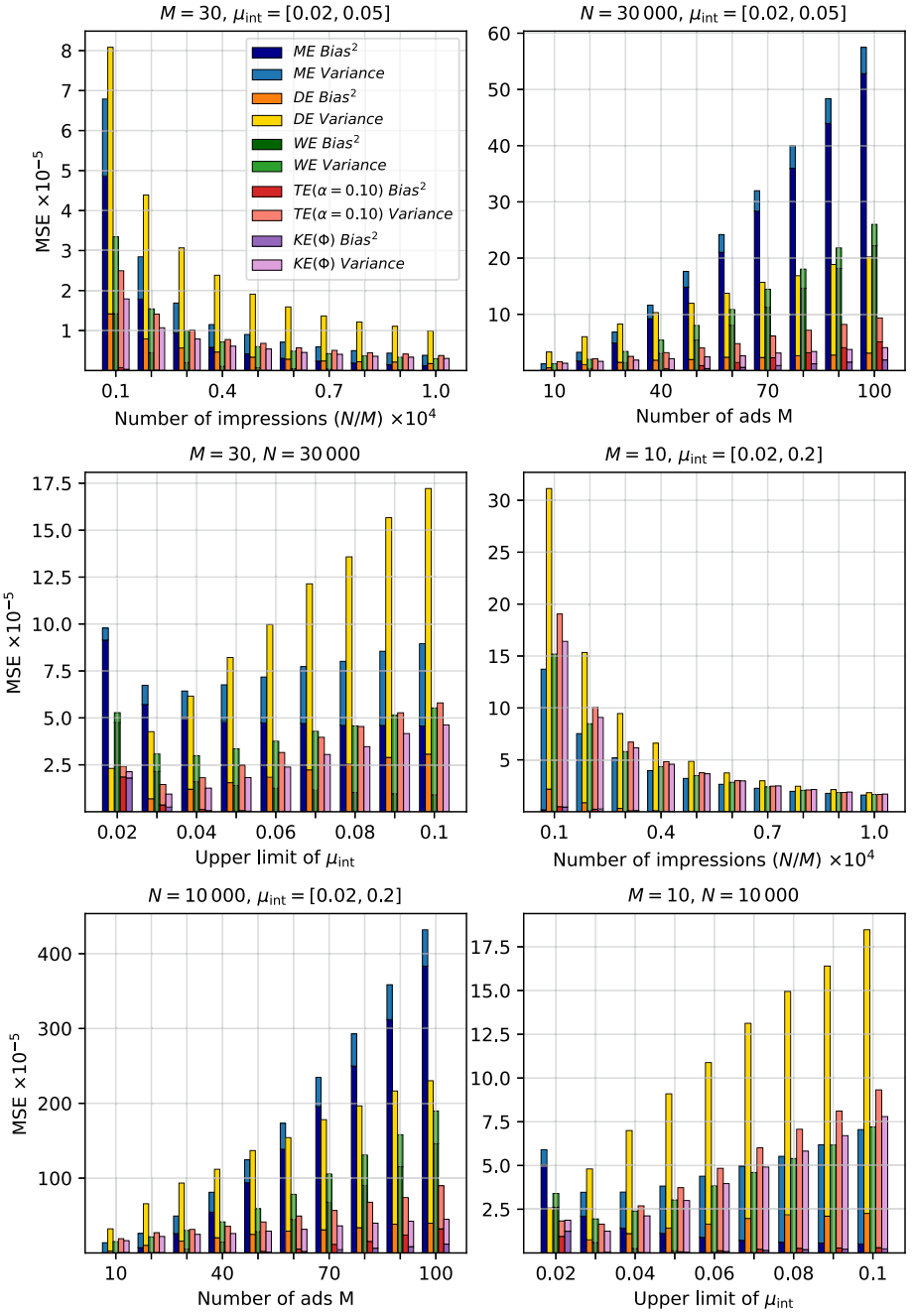


Fig. 7. Comparison of ME, DE, WE, TE, and KE on the internet ad problem. Results are averaged over 10 000 runs.

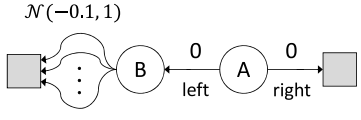


Fig. 8. The environment of the Maximization Bias Example; compare Figure 6.5 of Sutton and Barto [57].

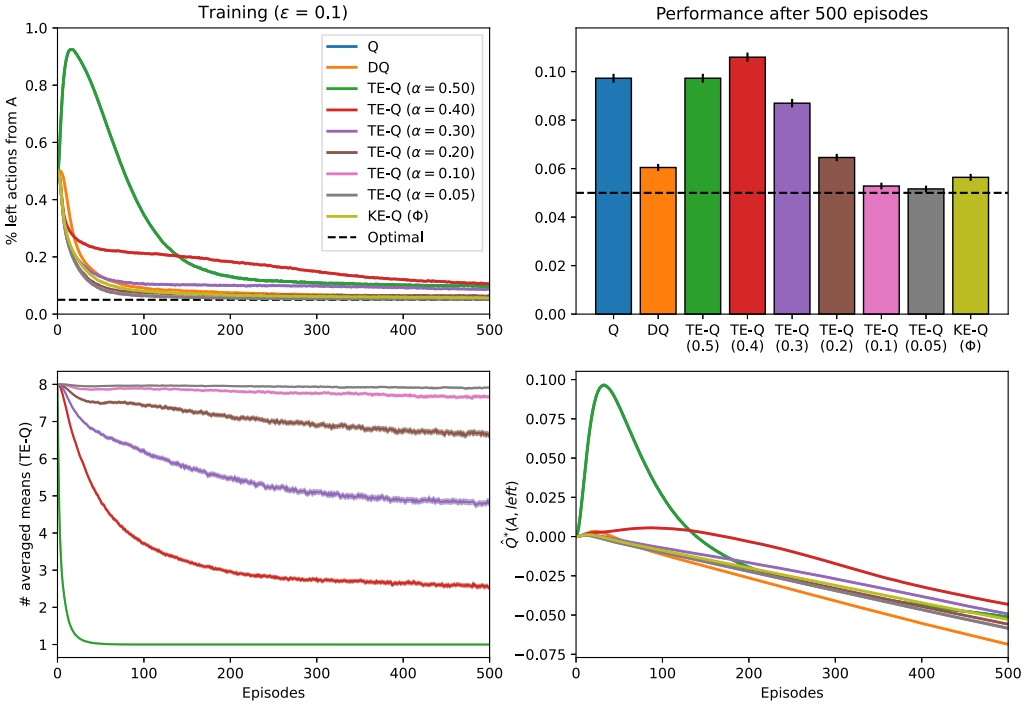


Fig. 9. Maximization Bias Example from Sutton and Barto [57] with parameters $\gamma = 1$, $\epsilon = 0.1$, and $\tau = 0.1$. Q and DQ refer to Q-Learning and Double Q-Learning, respectively. Results are averaged over 100 000 runs and 95% confidence intervals are included. The action-value estimate in the lower-right graph for DQ is generated by averaging over both Q-tables.

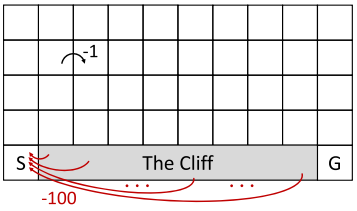


Fig. 10. The Cliff Walking environment; compare Example 6.6 of Sutton and Barto [57].

$\alpha \leq 0.1$. Furthermore, KE-Q-Learning with the standard Gaussian kernel performs reasonably well and achieves a final selection rate below Double Q-Learning.

7.3. Cliff Walking

We examine the Cliff Walking task from Example 6.6 in Sutton and Barto [57], which is an undiscounted, episodic task with start and goal states. Our environment is a grid of width 10 and height 5, being depicted in Fig. 10. Start state S is the lower-left grid point; goal state G is the lower-right grid point. All transitions are rewarded with -1, except those which lead to the grid points directly between S and G. Those are referred to as *The Cliff*, yield reward -100, and send the agent back to S. Actions are the four movement directions up, down, right, and left. Performance is measured via the return during an episode.

Fig. 11 follows the setup of Zhu and Rigotti [72] and contains results for constant $\epsilon = 0.1$ and annealing $\epsilon(s) = 1/\sqrt{n(s)}$, where $n(s)$ is the number of times state s has been visited. The learning rate is $\tau(s,a) = 0.1(100+1)/[100+n(s,a)]$, with $n(s,a)$ being the number of updates for the state-action pair. Next to Q- and Double Q-Learning, we consider Weighted Q-Learning (WQ, [20]) and Self-correcting Q-Learning (SCQ, [72]) with $\beta \in \{2,3\}$, following the recommendation of the authors. Crucially, the SCQ is a recent modification of Q-Learning that leverages a self-correcting mechanism. The idea is to combine subsequent, correlated action-value estimates into a single self-correcting estimator without increasing the computational burden of the procedure. Due to its solid theoretical foundation and competitive empirical results, we include the SCQ as a benchmark algorithm. Its deep version will also serve as a baseline for the experiments with function approximation in Section 7.4.

In the Cliff Walking task, we run each algorithm for 3000 episodes and average the results over 500 independent runs. Additionally, we display the maximum action-value estimate of the start state S over training. For comparison, since at least eleven steps are necessary to walk across our map, it holds for the optimal policy: $\max_{a'} Q^*(S,a') = -11$.

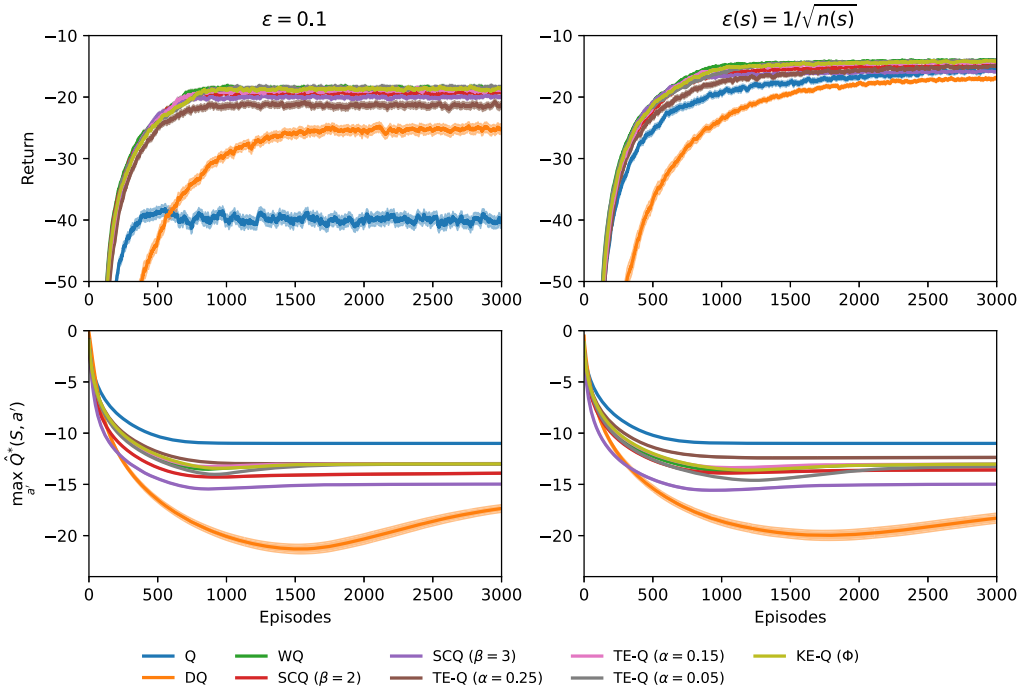


Fig. 11. Cliff Walking example from Sutton and Barto [57] with parameters $\gamma = 1$, $\tau(s,a) = 0.1(100+1)/[100+n(s,a)]$, and two different ϵ-greedy strategies. Results are averaged over 500 runs, exponentially smoothed for visualization purposes, and 95% confidence intervals are included. The maximum action-value estimate of the start state for DQ is computed by averaging this quantity over both Q-tables.

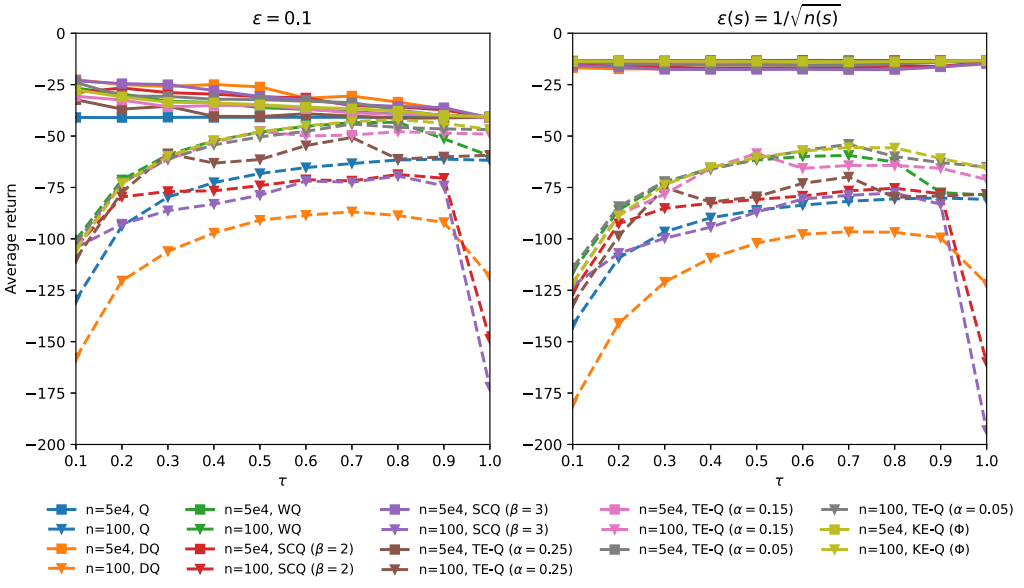


Fig. 12. Cliff Walking example adapted from Van Seijen et al. [64]. The algorithms' interim (dotted lines) and asymptotic (solid lines) return averages are analyzed for different learning rates. The number of episodes is n.

We see the strong performance of the newly proposed algorithms for both exploration strategies. Similar to the example in Section 7.2, especially TE-Q with $\alpha = 0.05$ and KE-Q are appropriate for this task and achieve the highest returns together with WQ. Furthermore, the higher action-value estimates for Q-Learning are apparent, while Double Q-Learning leads to severe underestimation. Finally, the returns are higher for all algorithms with $\epsilon(s) = 1/\sqrt{n(s)}$ than with a constant exploration rate, which is reasonable since action selection yields a higher probability of selecting greedy in the long-term.

To investigate the interim and asymptotic behavior of the algorithms for different learning rates, we run an analysis similar to Van Seijen et al. [64], and depict the results in Fig. 12. Precisely, we consider learning rates in $\{0.1, 0.2, \dots, 0.9, 1.0\}$ and employ the

two exploration strategies $\epsilon = 0.1$ and $\epsilon(s) = 1/\sqrt{n(s)}$ again. We analyze the average return over the first 100 episodes and average the results over 5000 runs for the interim performance. For the asymptotic scenario, we run each algorithm for 50 000 episodes, compute the average return of the episodes, and average the results over 5 runs.

The TE-Q and KE-Q algorithms offer the most robust interim progress across learning rates for both exploration strategies, while the DQ and SCQ expose a severe performance drop when $\tau = 1$. This might be because both algorithms rely on two different Q-tables, and complete replacement of the entries yields instabilities in this case. Regarding the asymptotic analysis, the SCQ and DQ algorithms improve on Q-Learning and are marginally above WQ, TE-Q, and KE-Q for $\epsilon = 0.1$, while the return differences are close to zero for the annealing exploration strategy due to long-term greedy action selection.

7.4. MinAtar

We select the MinAtar [69] environments to test the proposed Deep RL algorithms. MinAtar is a testbed incorporating several Atari games from the Arcade Learning Environment [7], which is considered a challenging benchmark for modern AI algorithms for sequential decision making. MinAtar is based on a reduced state-representation, incorporates sticky actions [42], and is designed to enable thorough algorithmic comparisons due to reduced computation times. Following Young and Tian [69], the network structure consists of a convolutional and a fully-connected layer. The remaining hyperparameters match Young and Tian [69], except that we use the Adam [33] optimizer, which led to much more stable results than using RMSprop [28] during our experiments. Appendix F contains the full list of specifications.

The compared algorithms are the DQN [44], DDQN [63], Self-Correcting DQN (SCDQN, [72]), MaxMin DQN [37], BDQN [47], TE-BDQN, KE-BDQN (with standard Gaussian cdf), and Ada-TE-BDQN. For the parametrization of the BDQN and its modifications, we follow Osband et al. [47] by using $K = 10$ bootstrap heads, each corresponding to one fully-connected layer, and setting $p = 1$ for the masking Bernoulli distribution. The BDQN uses the target computation of the DDQN, which we apply consequently. Furthermore, we scale the gradients of the convolutional core part for the bootstrap-based algorithms by $1/K$, which was also recommended by Osband et al. [47]. We consider $\beta \in \{2, 3, 4\}$ for the SCDQN, the number of networks $N \in \{2, 3\}$ for the MaxMin DQN, and $\alpha \in \{0.1, 0.2, 0.3, 0.4\}$ for the TE-BDQN. The bias parameter of the Ada-TE-BDQN is initialized with $\alpha = 0.25$ and we consider two step sizes $\tau_{\text{Ada}} \in \{10^{-4}, 10^{-5}\}$ with horizon $T_{\text{Ada}} = 32$.

To check the robustness of the algorithms, we analyze three different learning rates for each environment and algorithm: $\tau \in \{10^{-5}, 10^{-4.5}, 10^{-4}\}$. Every 10 000 steps during an experiment, we average the return of 10 test episodes. For the BDQN and its variants, the majority vote of the ensemble is applied. Additionally, we run for all algorithms bias estimation episodes from random initial states sampled from the replay buffer, following Algorithm 3. The number of those episodes are 10 for the DQN, DDQN, and SCDQN, while we run only 3 episodes for each head of the BDQN-based algorithms due to computation time. We repeat all experiments for ten independent runs, exponentially smooth the results for clarity, and include 95% point-wise confidence intervals over the runs.

Figs. 13 - 17 depict the results. We show the final return across learning rates in the first row of each plot, while the second and third rows contain the return and bias plots, respectively, for $\tau = 10^{-4.5}$. Lastly, the fourth row displays the dynamic behavior of the α for the Ada-TE-BDQN. Please note that the bias and return curves for $\tau \in \{10^{-5}, 10^{-4}\}$ are provided in Appendix G. The BDQN-based algorithms are generally comparable to the MaxMin DQN and outperform their competitors DQN, DDQN, and SCDQN. Especially the KE-BDQN and Ada-TE-BDQN show a robust performance across environments, although the algorithms' variances are relatively high in Seaquest. As expected, the DQN is affected by massive overestimations, while the DDQN can reduce the Q-estimates in comparison. Although the DE theoretically underestimates the MEV, the DDQN still offers a positive bias in the experiments. This observation is in line with Van Hasselt et al. [63] and might be explained by the fact that the DE, as introduced in Section 2.3, would require two independent networks. However, the DDQN, as commonly implemented, uses the main and target networks for action evaluation and selection, respectively, and these two networks are time-delayed copies of each other and thus are not independent.

As theoretically discussed in prior sections, a larger α in the TE-BDQN yields larger Q-estimates and, consequently, a larger estimation bias. The adaptive mechanism of Ada-TE-BDQN, especially for $\tau_{\text{Ada}} = 10^{-4}$, results in approximately unbiased action-value estimates. Throughout environments, the α of the Ada-TE-BDQN mostly increases during training but stays on moderate values in approximately $[0.2, 0.4]$. Large values with $\alpha \approx 0.5$ are not achieved even in later stages of training, indicating the criticality of the ME combined with function approximation.

7.5. Discussion

The experimental results confirm that we can embed statistical thinking in form of the TE/KE into value-based RL methods to control their estimation bias. In addition, the experiments support the finding of Lan et al. [37] that unbiased Q-estimation does not necessarily translate into the best return performance. For example, in SpaceInvaders, depicted in Fig. 17, the KE-BDQN is return-wise the strongest algorithm despite its severe negative bias. However, the TE-BDQN ($\alpha = 0.1$) offers an even lower bias but cannot match the return of the KE-BDQN. There seems to be a critical level - or path over time - of estimation bias for a given MDP, which yields maximum performance. Careful selection of a bias control parameter like α for the TE or the kernel function for the KE thus constitutes a crucial component in designing temporal-difference algorithms.

Analyzing the behavior of the algorithms in more detail, we see that the estimated bias *changes over time*. With random initialization of the networks and a couple of zero-return episodes, all algorithms' bias is shortly approximately zero. As soon as some non-zero rewards are observed, the different target specifications affect the update routine and result in severely different bias plots over time.

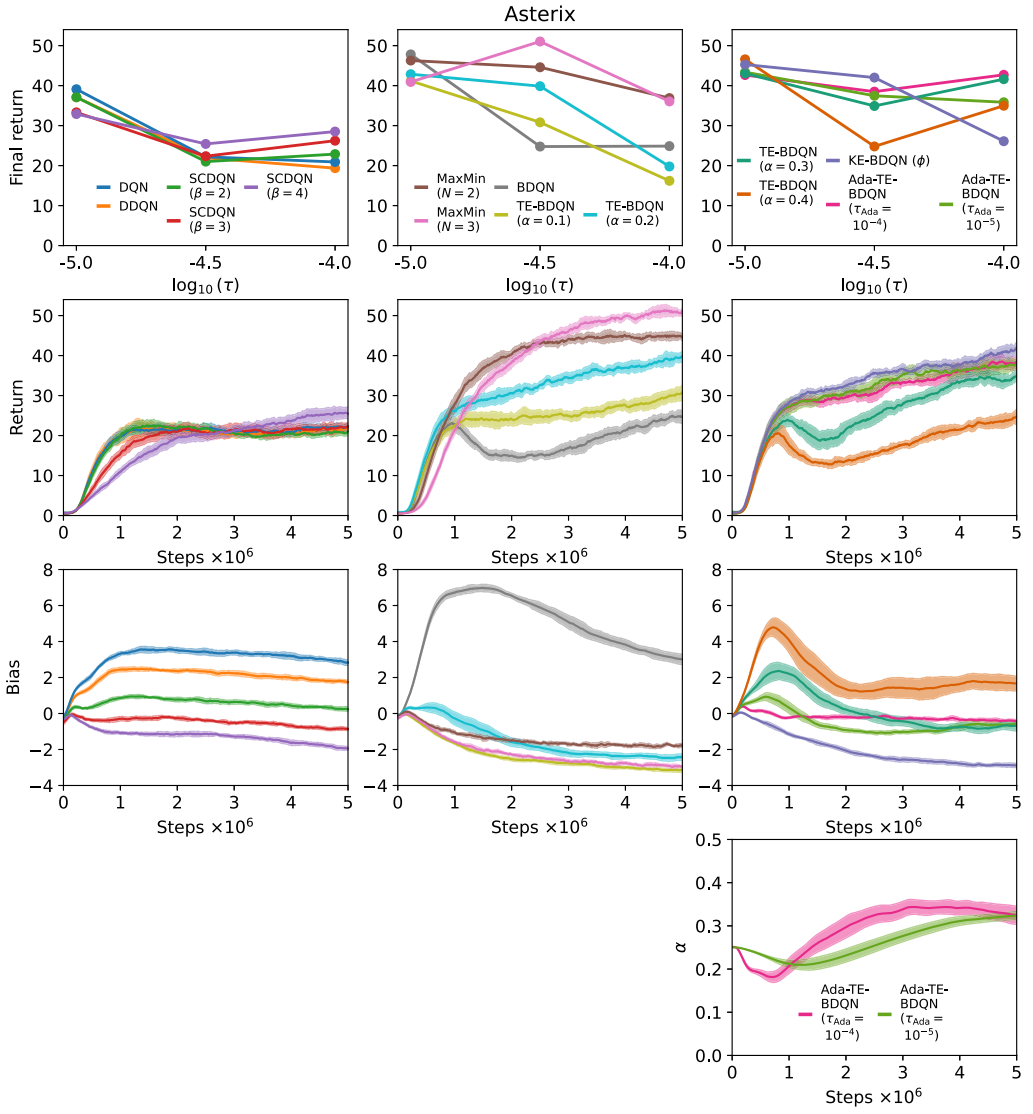


Fig. 13. Algorithm comparison on Asterix. The first row shows the final return of different learning rates. The second and third row show the return and bias over time for $\tau = 10^{-4.5}$. The dynamic behavior of the α for the Ada-TE-BDQN is displayed in row four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results.

Besides the Ada-TE-BDQN, each algorithm reveals its tendency towards over- or underestimation, although exceptions are possible. For example, the TE-BDQN ($\alpha = 0.3$) offers slight overestimations during the first three million steps in Breakout, as shown in Fig. 14, before shifting towards underestimation. Importantly, **none** of the non-adaptive algorithms shows reliable convergence to zero-bias as training proceeds, which agrees with the observations of Van Hasselt et al. [63]. Finally, we summarize the core findings of our investigation:

1. *Absolute bias minimization does not equal return maximization.* In order to maximize performance in a real application, different bias control configurations should be considered, for which the TE/KE-BDQN build a flexible framework.
2. *Approximately unbiased estimation offers a robust baseline across tasks.* Although it is not always the return-maximizing choice, using a scheme for approximately unbiased value-estimation appears more robust across tasks than fixing a particular bias control parameter. The Ada-TE-BDQN is a powerful candidate for such a scheme since it almost achieves zero bias during the MinAtar experiments.
3. *The compatibility between bias control algorithms and exploration schemes requires systematic analysis.* The impact on exploration most likely constitutes an essential factor in the occasional return-improvement of a biased procedure over an unbiased one [41]. Further study needs to generate insights on how these components interact and, crucially, whether the algorithmic approaches are compatible. Can we use the Ada-TE-BDQN with a modified exploration scheme to boost return performance? Do we maintain

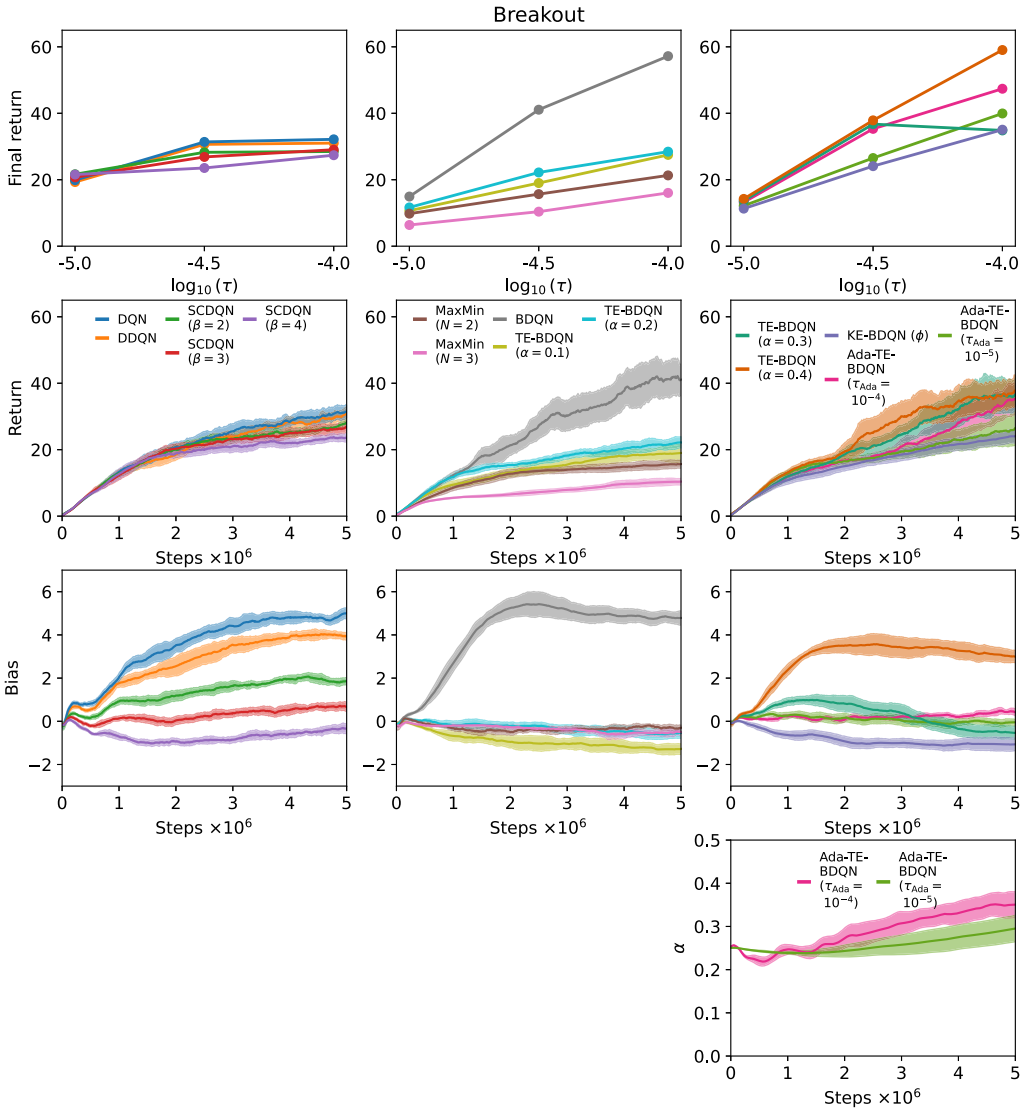


Fig. 14. Algorithm comparison on Breakout. The first row shows the final return of different learning rates. The second and third row show the return and bias over time for $\tau = 10^{-4.5}$. The dynamic behavior of the α for the Ada-TE-BDQN is displayed in row four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results.

unbiased action-value estimation in this process? Can we achieve or even improve on the return peaks of a fine-tuned bias control configuration in this manner? Exploration and action-value estimation are even in off-policy RL not necessarily orthogonal and thus constitute a crucial path for future research. We emphasize that similar considerations are prevalent in the bandit literature [38,55], whose insights might be leveraged to further analyze the interplay of exploration and action-value estimation.

8. Related works

Next to Van Hasselt [61], D'Eramo et al. [20], Lan et al. [37], and Zhu and Rigotti [72], several proposals have been made to further tackle the issue of estimation bias in temporal-difference algorithms. Zhang et al. [71] proposed a hybrid between the ME and the DE called Weighted Double Estimator. It relies on a hyperparameter on the positive real axis, for which the authors propose a heuristic based on empirical experiences. Lee et al. [39] proposed Bias-corrected Q-Learning, which incorporates a correction term depending on the reward variance. The Randomized Ensembled Double Q-Learning (REDQ, [12]) is an extension of MaxMin Q-Learning [37] and applies a minimization operator over a subset of the ensemble. The Action-Candidate based Clipped Double Estimator [32] extends the DE by creating a so-called candidate set of indices of which the maximizing one will be picked. Furthermore, the Clipped Double Estimator of Fujimoto et al. [24] and the Truncated Quantile Critic (TQC, [36]) algorithm are relevant contributions to addressing the overestimation issue in actor-critic frameworks. Finally, Lee et al. [40] pursued a re-weighting strategy of sampled transitions based

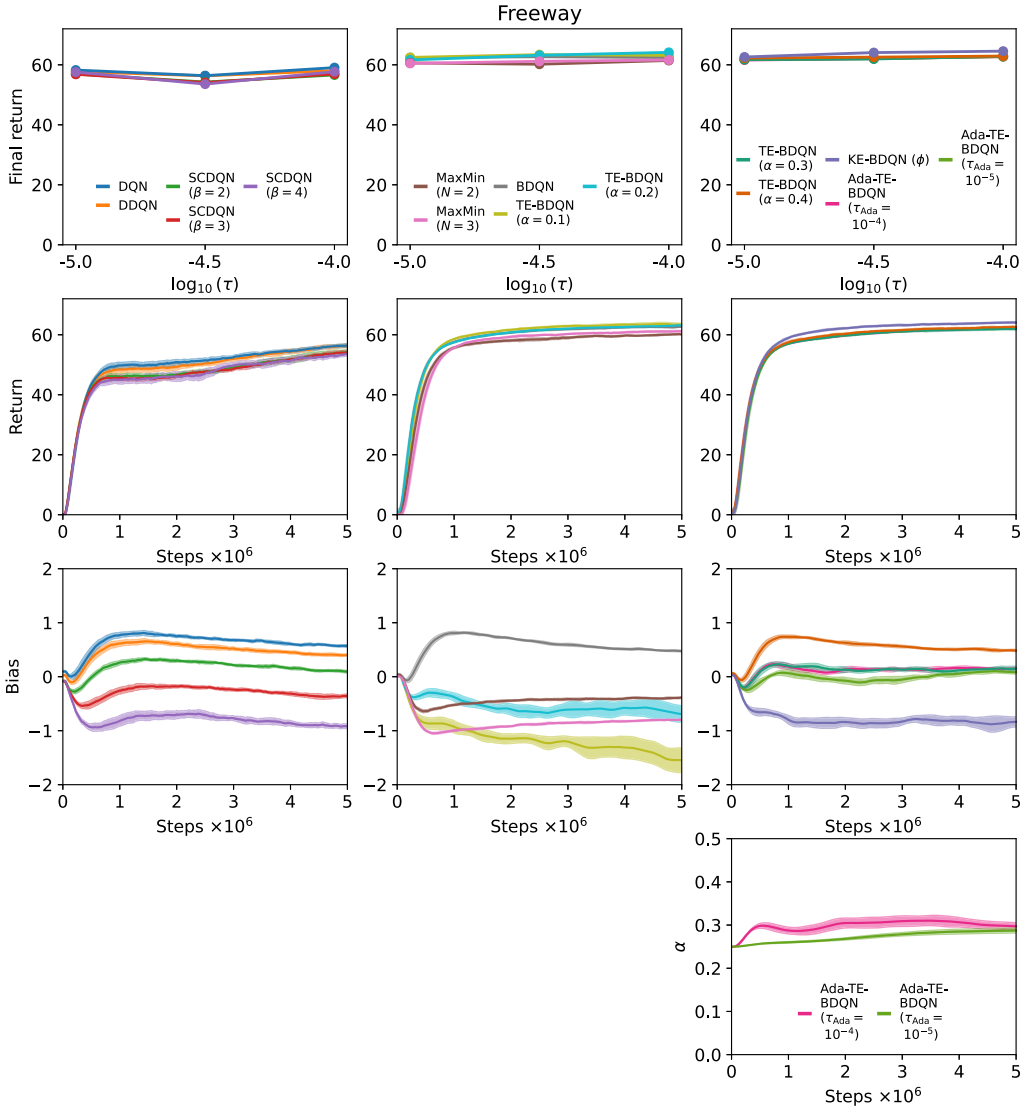


Fig. 15. Algorithm comparison on Freeway. The first row shows the final return of different learning rates. The second and third row show the return and bias over time for $\tau = 10^{-4.5}$. The dynamic behavior of the α for the Ada-TE-BDQN is displayed in row four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results.

on uncertainty estimates from an ensemble. Apart from methodological extensions, Chen et al. [13] recently reported that a lower learning rate or an adequate schedule could also avoid the massive overestimations of Q-Learning. However, lowering the learning rate can come at the expense of impractically slow learning, as seen in our Breakout experiments in Fig. 14, and constitutes thus not a practical option to address the issue of action-value estimation.

Recently, some proposals have been made to minimize the estimation bias of temporal-difference algorithms through online parameter adjustments in the spirit of the Ada-TE-BDQN. Liang et al. [41] expand the work of Fox et al. [23], Fox [22] by using an ensemble to adjust the temperature parameter in a maximum entropy framework. Kuznetsov et al. [35] and Dorka et al. [17] introduce adaptive variants of the TQC by adjusting the number of quantiles to drop based on recent near on-policy trajectories. Finally, Wang et al. [67] generalize MaxMin Q-Learning and REDQ by changing the size of the subset of the ensemble on which the minimization operator is performed. The metric driving the adjustment is the ensemble's function approximation error since it is argued that high approximation error is connected with the overestimation of action-values.

9. Conclusion

Reinforcement learning as a domain of artificial intelligence has made significant breakthroughs in a diverse set of real-world applications, particularly in the last decade. A key issue of frequently applied temporal-difference algorithms is the propagation of

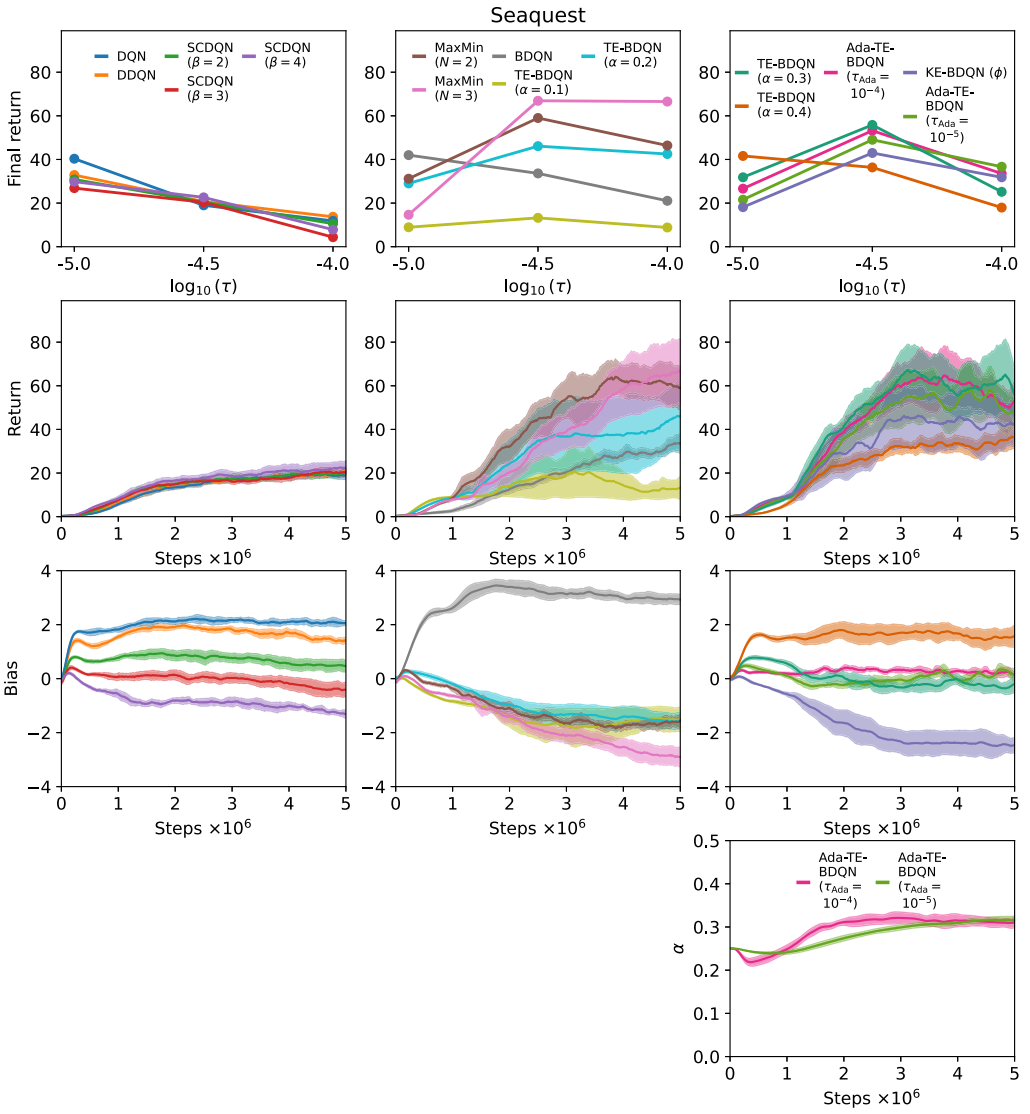


Fig. 16. Algorithm comparison on Seaquest. The first row shows the final return of different learning rates. The second and third row show the return and bias over time for $\tau = 10^{-4.5}$. The dynamic behavior of the α for the Ada-TE-BDQN is displayed in row four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results.

biased action-value estimates. We address this topic by proposing the T-Estimator and the K-Estimator for the underlying problem of estimating the maximum expected value of random variables. Both estimators are easy to compute and allow to flexibly interpolate between over- and underestimation bias, leading to promising modifications of Q-Learning and the Bootstrapped DQN algorithm. Coupled with the dynamic selection procedure of the significance level of TE, our work constitutes an important step towards unbiased estimation of action-values with function approximation.

In future research, we will analyze the discussed interplay of action-value estimation and exploration. As methodological extensions, we will investigate possibilities to extend the two-sample testing procedures into continuous action spaces to modify policy gradient methods because the latter constitute an elementary class of methods in several application domains. Furthermore, next to the considered procedure, there are alternative approaches for uncertainty quantification in the neural network scenario. For example, the regularization technique dropout [56] can be applied similar to D'Eramo et al. [15] to obtain the required variance estimate for the newly proposed algorithms, and the Bootstrapped DQN can be enhanced by adding random prior functions [46]. Finally, our work estimates the bias for given Q-estimates via Algorithm 3, leading to one scalar bias estimate for the whole state-action distribution. While we analyze how this scalar changes throughout training, we do not delve into a detailed differentiation of how this bias is distributed across the state-action space at a specific point during training. We acknowledge that assessing complex MDPs in this fashion might result in an over-simplification. In particular, this line of investigation might lead to more tailored solutions, and one could consider, for example, an individual significance level of the TE for different regions of the state-action space.

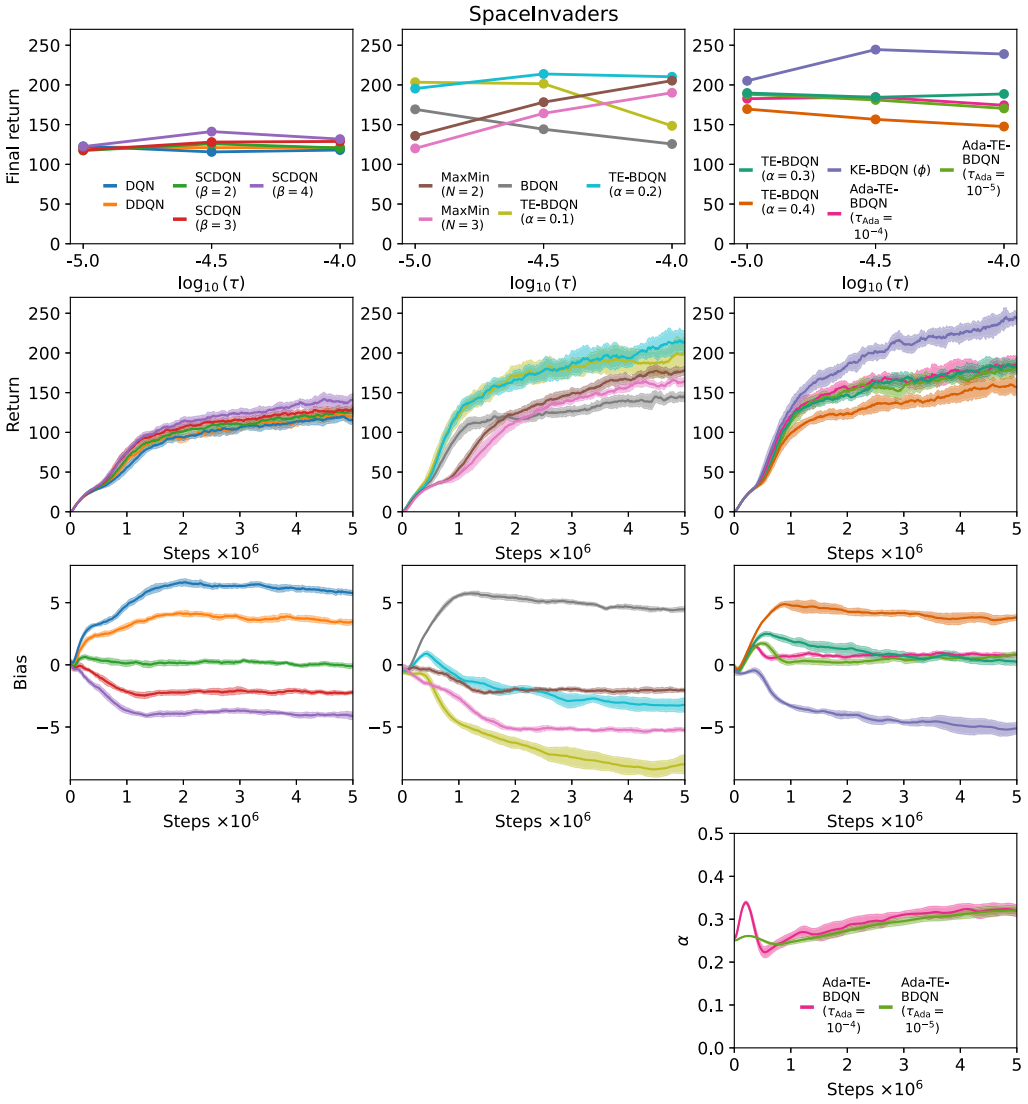


Fig. 17. Algorithm comparison on SpaceInvaders. The first row shows the final return of different learning rates. The second and third row show the return and bias over time for $\tau = 10^{-4.5}$. The dynamic behavior of the α for the Ada-TE-BDQN is displayed in row four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results.

CRediT authorship contribution statement

Martin Waltz: Conceptualization, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Ostap Okhrin:** Conceptualization, Investigation, Methodology, Project administration, Supervision, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

We would like to thank Niklas Paulig for fruitful discussions in the early stages of this work. Furthermore, we are grateful to the Center for Information Services and High Performance Computing at TU Dresden for providing its facilities for high throughput calculations. Moreover, we appreciate the valuable feedback of two anonymous reviewers, which helped to improve this paper thoroughly. Finally, we would like to thank the participants of the Conference on Reinforcement Learning and Decision Making (RLDM) 2022, the German Probability and Statistics Days (GPSD) 2023, and the Conference on Computational Statistics (COMPSTAT) 2023 for their fruitful feedback on this work.

Appendix A. Analytic forms for Section 4.1

A.1. Maximum Estimator

Consider the following setup: $M = 2$, $X_1 \sim \mathcal{N}(\mu_1, \sigma^2)$, $X_2 \sim \mathcal{N}(\mu_2, \sigma^2)$, and given sample sizes $|S_1|$, $|S_2|$, from which follows: $\hat{\mu}_1 \sim \mathcal{N}(\mu_1, \sigma^2/|S_1|)$ and $\hat{\mu}_2 \sim \mathcal{N}(\mu_2, \sigma^2/|S_2|)$. Regarding the ME, we use the expectation in (1) to compute the bias. However, we can alternatively use the following closed-form solutions [45]:

$$\begin{aligned}
\mathrm{E}\left[\max(\hat{\mu}_1, \hat{\mu}_2)\right] &= \mu_1 \Phi\left(\frac{\mu_1 - \mu_2}{\theta}\right) + \mu_2 \Phi\left(\frac{\mu_2 - \mu_1}{\theta}\right) + \theta\phi\left(\frac{\mu_1 - \mu_2}{\theta}\right), \\
\mathrm{E}\left\{\left[\max(\hat{\mu}_1, \hat{\mu}_2)\right]^2\right\} &= \left(\frac{\sigma^2}{|S_1|} + \mu_1^2\right) \Phi\left(\frac{\mu_1 - \mu_2}{\theta}\right) + \left(\frac{\sigma^2}{|S_2|} + \mu_2^2\right) \Phi\left(\frac{\mu_2 - \mu_1}{\theta}\right) \\
&\quad + (\mu_1 + \mu_2)\theta\phi\left(\frac{\mu_1 - \mu_2}{\theta}\right),
\end{aligned}$$

where ϕ is the standard Gaussian pdf and $\theta = \sqrt{\frac{\sigma^2}{|S_1|} + \frac{\sigma^2}{|S_2|}}$. The expectation of the squared ME can be used to compute the variance:

$$\mathrm{Var}\left[\max(\hat{\mu}_1, \hat{\mu}_2)\right] = \mathrm{E}\left\{\left[\max(\hat{\mu}_1, \hat{\mu}_2)\right]^2\right\} - \mathrm{E}\left[\max(\hat{\mu}_1, \hat{\mu}_2)\right]^2.$$

A.2. Double Estimator

The expectation of the DE is given in (3), which directly yields the bias. As already mentioned, what we refer to as the DE throughout the paper is actually the CVE whenever possible, thus we compute the variance of the latter for this example. For notation, we use $\hat{\mu}_i^A = \hat{\mu}_i(S_i^A)$ and $\hat{f}_i^A$, $\hat{F}_i^A$ for the pdf and cdf of $\hat{\mu}_i^A$, respectively, and similarly for S^B. We assume that the sample S is split evenly between S^A and S^B, so that the theoretical mean distribution $\hat{f}_i^A$ equals $\hat{f}_i^B$. The DE estimate when index selection is performed on subsample S^A is denoted with $\hat{\mu}_*^{DE,A}$, and when selecting based on S^B with $\hat{\mu}_*^{DE,B}$. It follows:

$$\begin{aligned}
\mathrm{Var}\left(\hat{\mu}_*^{\mathrm{CVE}}\right) &= \mathrm{Var}\left(\frac{\hat{\mu}_*^{DE,A} + \hat{\mu}_*^{DE,B}}{2}\right) \\
&= \frac{1}{4}\mathrm{Var}\left(\hat{\mu}_*^{DE,A}\right) + \frac{1}{4}\mathrm{Var}\left(\hat{\mu}_*^{DE,B}\right) + \frac{1}{2}\mathrm{Cov}\left(\hat{\mu}_*^{DE,A}, \hat{\mu}_*^{DE,B}\right) \\
&= \frac{1}{2}\mathrm{Var}\left(\hat{\mu}_*^{DE,A}\right) + \frac{1}{2}\mathrm{Cov}\left(\hat{\mu}_*^{DE,A}, \hat{\mu}_*^{DE,B}\right),
\end{aligned} \tag{A.1}$$

because $\mathrm{Var}\left(\hat{\mu}_*^{DE,A}\right) = \mathrm{Var}\left(\hat{\mu}_*^{DE,B}\right)$. Using definition:

$$\mathrm{Var}\left(\hat{\mu}_*^{DE,A}\right) = \mathrm{E}\left[\left(\hat{\mu}_*^{DE,A}\right)^2\right] - \mathrm{E}\left[\hat{\mu}_*^{DE,A}\right]^2, \tag{A.2}$$

in which:

$$\mathrm{E}\left[\left(\hat{\mu}_*^{DE,A}\right)^2\right] = \mathrm{E}\left[\left(\hat{\mu}_1^B\right)^2\right] \int_{-\infty}^{\infty} \hat{f}_1^A(x)\hat{F}_2^A(x)dx + \mathrm{E}\left[\left(\hat{\mu}_2^B\right)^2\right] \int_{-\infty}^{\infty} \hat{f}_2^A(x)\hat{F}_1^A(x)dx,$$

where we compute: $\mathrm{E}\left[\left(\hat{\mu}_1^B\right)^2\right] = \mathrm{Var}\left(\hat{\mu}_1^B\right) + \mathrm{E}\left[\hat{\mu}_1^B\right]^2$; and $\mathrm{E}\left[\left(\hat{\mu}_2^B\right)^2\right]$ analogously, so that (A.2) is complete. To compute the covariance in (A.1), we have:

$$\mathrm{Cov}\left(\hat{\mu}_*^{DE,A}, \hat{\mu}_*^{DE,B}\right) = \mathrm{E}\left[\hat{\mu}_*^{DE,A}\hat{\mu}_*^{DE,B}\right] - \mathrm{E}\left[\hat{\mu}_*^{DE,A}\right]\mathrm{E}\left[\hat{\mu}_*^{DE,B}\right],$$

with the expectation of the product being

$$\begin{aligned}\mathrm{E}\left[\hat{\mu}_*^{DE,A}\hat{\mu}_*^{DE,B}\right] &= \mathrm{E}\left\{\left[\mathcal{I}\left(\hat{\mu}_1^A > \hat{\mu}_2^A\right)\hat{\mu}_1^B + \mathcal{I}\left(\hat{\mu}_1^A \le \hat{\mu}_2^A\right)\hat{\mu}_2^B\right]\left[\mathcal{I}\left(\hat{\mu}_1^B > \hat{\mu}_2^B\right)\hat{\mu}_1^A + \mathcal{I}\left(\hat{\mu}_1^B \le \hat{\mu}_2^B\right)\hat{\mu}_2^A\right]\right\} \\ &= \mathrm{E}\left[\mathcal{I}\left(\hat{\mu}_1^A > \hat{\mu}_2^A\right)\hat{\mu}_1^A\right]\mathrm{E}\left[\mathcal{I}\left(\hat{\mu}_1^B > \hat{\mu}_2^B\right)\hat{\mu}_1^B\right] + \mathrm{E}\left[\mathcal{I}\left(\hat{\mu}_1^A > \hat{\mu}_2^A\right)\hat{\mu}_2^A\right]\mathrm{E}\left[\mathcal{I}\left(\hat{\mu}_1^B \le \hat{\mu}_2^B\right)\hat{\mu}_1^B\right] \\ &+ \mathrm{E}\left[\mathcal{I}\left(\hat{\mu}_1^A \le \hat{\mu}_2^A\right)\hat{\mu}_1^A\right]\mathrm{E}\left[\mathcal{I}\left(\hat{\mu}_1^B > \hat{\mu}_2^B\right)\hat{\mu}_2^B\right] + \mathrm{E}\left[\mathcal{I}\left(\hat{\mu}_1^A \le \hat{\mu}_2^A\right)\hat{\mu}_2^A\right]\mathrm{E}\left[\mathcal{I}\left(\hat{\mu}_1^B \le \hat{\mu}_2^B\right)\hat{\mu}_2^B\right].\end{aligned}$$

This expression is simplified using $\mathcal{I}\left(\hat{\mu}_1^A > \hat{\mu}_2^A\right) = 1 - \mathcal{I}\left(\hat{\mu}_1^A \le \hat{\mu}_2^A\right)$ to get

$$\mathrm{E}\left[\hat{\mu}_*^{DE,A}\hat{\mu}_*^{DE,B}\right] = \mu_1^2 + 2I_1(\mu_2 - \mu_1) + (I_1 - I_2)^2,$$

where

$$I_1 = \mathrm{E}\left[\mathcal{I}(\hat{\mu}_1^A \le \hat{\mu}_2^A)\hat{\mu}_1^A\right] = \mu_1 - \int_{-\infty}^{\infty} x\hat{f}_1^A(x)\hat{F}_2^A(x)dx,$$

$$I_2 = \mathrm{E}\left[\mathcal{I}(\hat{\mu}_1^A \le \hat{\mu}_2^A)\hat{\mu}_2^A\right] = \int_{-\infty}^{\infty} x\hat{f}_2^A(x)\hat{F}_1^A(x)dx.$$

A.3. T-Estimator and K-Estimator

Regarding the expectation of the KE, we have:

$$\begin{aligned}\mathrm{E}\left[\hat{\mu}_*^{\mathrm{KE}}\right] &= \mathrm{E}\left[\hat{\mu}_*^{\mathrm{KE}}\mathcal{I}(\hat{\mu}_1 > \hat{\mu}_2)\right] + \mathrm{E}\left[\hat{\mu}_*^{\mathrm{KE}}\mathcal{I}(\hat{\mu}_1 \le \hat{\mu}_2)\right] \\ &= \mathrm{E}\left[\left\{\left[\sum_{j=1}^{2}\kappa\left(\frac{\hat{\mu}_j - \hat{\mu}_1}{\theta_{1j}}\right)\right]^{-1}\sum_{j=1}^{2}\kappa\left(\frac{\hat{\mu}_j - \hat{\mu}_1}{\theta_{1j}}\right)\hat{\mu}_j\right\}\mathcal{I}\left(\hat{\mu}_1 > \hat{\mu}_2\right)\right] \\ &+ \mathrm{E}\left[\left\{\left[\sum_{j=1}^{2}\kappa\left(\frac{\hat{\mu}_j - \hat{\mu}_2}{\theta_{2j}}\right)\right]^{-1}\sum_{j=1}^{2}\kappa\left(\frac{\hat{\mu}_j - \hat{\mu}_2}{\theta_{2j}}\right)\hat{\mu}_j\right\}\mathcal{I}\left(\hat{\mu}_1 \le \hat{\mu}_2\right)\right] \\ &= \mathrm{E}\left\{\frac{1}{\kappa(0) + \kappa\left(\frac{\hat{\mu}_2 - \hat{\mu}_1}{\theta_{12}}\right)}\left[\kappa(0)\hat{\mu}_1 + \kappa\left(\frac{\hat{\mu}_2 - \hat{\mu}_1}{\theta_{12}}\right)\hat{\mu}_2\right]\mathcal{I}(\hat{\mu}_1 > \hat{\mu}_2)\right\} \\ &+ \mathrm{E}\left\{\frac{1}{\kappa\left(\frac{\hat{\mu}_1 - \hat{\mu}_2}{\theta_{21}}\right) + \kappa(0)}\left[\kappa\left(\frac{\hat{\mu}_1 - \hat{\mu}_2}{\theta_{21}}\right)\hat{\mu}_1 + \kappa(0)\hat{\mu}_2\right]\mathcal{I}(\hat{\mu}_1 \le \hat{\mu}_2)\right\} \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{x_1}\frac{1}{\kappa(0) + \kappa\left(\frac{x_2 - x_1}{\theta_{12}}\right)}\left[\kappa(0)x_1 + \kappa\left(\frac{x_2 - x_1}{\theta_{12}}\right)x_2\right]\hat{f}_1(x_1)\hat{f}_2(x_2)dx_2dx_1 \\ &+ \int_{-\infty}^{\infty}\int_{-\infty}^{x_2}\frac{1}{\kappa\left(\frac{x_1 - x_2}{\theta_{21}}\right) + \kappa(0)}\left[\kappa\left(\frac{x_1 - x_2}{\theta_{21}}\right)x_1 + \kappa(0)x_2\right]\hat{f}_1(x_1)\hat{f}_2(x_2)dx_1dx_2,\end{aligned}$$

where $\theta_{ij} = \sqrt{\frac{\sigma^2}{|S_i|} + \frac{\sigma^2}{|S_j|}}$ and $\hat{f}_i$ is the pdf of $\hat{\mu}_i$. For the variance, we can compute $\mathrm{E}\left\{[\hat{\mu}_*^{\mathrm{KE}}]^2\right\}$ analogously. Since the TE is a special case of the KE with $\kappa(T) = \mathcal{I}(T \ge z_\alpha)$, the above formula is also applicable for TE. We emphasize that we numerically approximate all integrals when using the formulas of Appendix A.

Appendix B. Expectation of the KE for known variances

In the following, we detail an expression for the expectation of the KE in case the variances of the underlying random variables are known. Since the KE generalizes the TE, the expression similarly holds for the TE.

Corollary 3. *The expectation of the KE is:*

$$\mathrm{E}[\hat{\mu}_*^{\mathrm{KE}}] = \sum_{i=1}^{M} \int_{-\infty}^{\infty} \int_{-\infty}^{x_i} \cdots \int_{-\infty}^{x_i} \left[\sum_{j=1}^{M} \kappa\left(\frac{x_j - x_i}{\theta_{ij}} \right) \right]^{-1} \left[\sum_{j=1}^{M} \kappa\left(\frac{x_j - x_i}{\theta_{ij}} \right) x_j \right] \left[\prod_{j=1}^{M} \hat{f}_j(x_j) \right] \left[\prod_{\substack{j=1 \\ j \neq i}}^{M} dx_j \right] dx_i,$$

where $\theta_{ij} = \sqrt{\frac{\sigma_i^2}{|S_i|} + \frac{\sigma_j^2}{|S_j|}}$, $\hat{f}_j$ *is the pdf of* $\hat{\mu}_j$*, being asymptotically normal, and* σ_i^2 *are known.*

Proof. Follows immediately from generalizing the derivation in Appendix A.3 from the case $M = 2$ to higher dimensions. □

Appendix C. Optimizing the example from Section 4.1

The chosen specification of the beta distribution in the example with two Gaussian random variables from Fig. 2 of Section 4.1 results in a rather strong underestimation for large $\mu_1 - \mu_2$. To find a better fitting parametrization, we numerically solved the optimization problem of minimizing the squared bias for the range of $\mu_1 \in [0, 5]$ over the parameters $\mathfrak{a}$, $\mathfrak{b}$ of the $\mathcal{B}_{\mathfrak{a},\mathfrak{b}}$ kernel. To enable comparability between the estimators, we run the identical optimization for the parameter λ of the Gaussian kernel Φ_λ (deviating from the unit variance specification) and the level of significance of the TE. The optimized kernel functions alongside specifications from Figs. 1 and 2 are depicted in Fig. C.1, while Fig. C.2 displays the performance of the optimized estimators.

The functions in Fig. C.1 are normalized to $[0, 1]$ by division through $\kappa(0)$ of the respective kernel. Optimizing the standard deviation of the Gaussian cdf yields $\lambda \approx 0.84$, which is close to the unit variance specification. On the other hand, the bias-optimal value for the significance level of the TE is ≈ 0.14. Regarding the $\mathcal{B}_{\mathfrak{a},\mathfrak{b}}$ specification, one needs to recall that the beta kernel is capable of approximating both the optimized TE and the optimized KE with the (non-standard) Gaussian cdf kernel. Following Figs. C.1 and C.2, the optimized TE is favorable in this scenario since the optimized beta cdf is in line with the optimized TE. We emphasize that the numerical optimization of the beta CDF yields different solutions depending on the starting values. However, all solutions result in a distribution with zero variance.

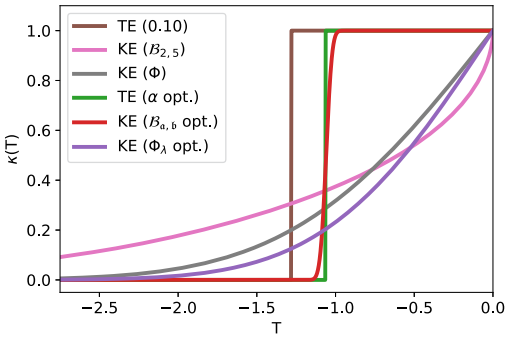


Fig. C.1. Original kernel functions and optimized specification for minimizing the squared bias in Figs. 1 and 2.

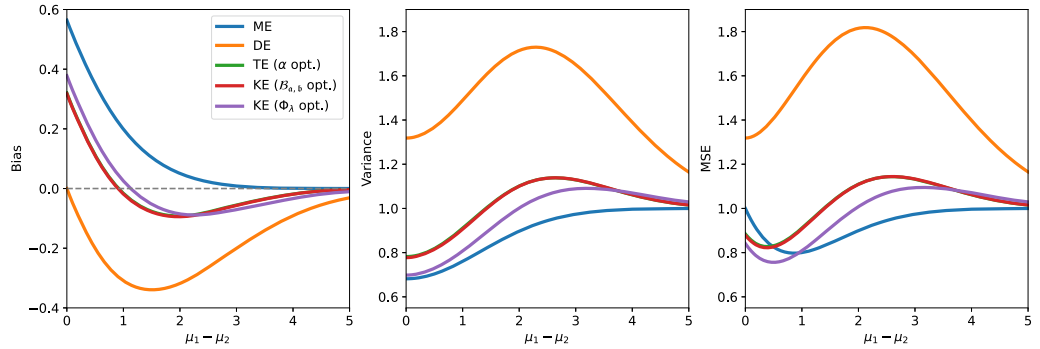


Fig. C.2. Bias, variance, and MSE for different estimators of the MEV in the case of two Gaussian random variables using the optimized TE and KE's shown in Fig. C.1. The optimized beta kernel opts for a similar solution as the optimized TE.

Appendix D. Proof of convergence of TE/KE-Q-learning

We sketch the proof of Theorem 1 for the discounted case by building on the works of Singh et al. [54], Van Hasselt [61], and Fujimoto et al. [24]. In particular, we need the following Lemma of Singh et al. [54]:

Lemma 3. *Consider a stochastic process* (ζ_t, Δ_t, F_t), $t \geq 0$ *where* $\zeta_t, \Delta_t, F_t : X \to \mathbb{R}$ *satisfy the equation:*

$$\Delta_{t+1}(x_t) = [1 - \zeta_t(x_t)]\Delta_t(x_t) + \zeta_t(x_t)F_t(x_t),$$

where $x_t \in X$, *with* $t = 0, 1, 2, \ldots$, *and* $\bigcup_{t=0}^{\infty} x_t = X$. *Let* P_t *be a sequence of increasing* σ*-fields such that* ζ_0 *and* Δ_0 *are* P_0*-measurable and* ζ_t, Δ_t, *and* F_{t-1} *are* P_t*-measurable,* $t = 1, 2, \ldots$. *Assume that the following hold:*

1. *The set* X *is finite.*
2. $\zeta_t(x_t) \in [0, 1]$, $\sum_t \zeta_t(x_t) = \infty$, $\sum_t [\zeta_t(x_t)]^2 < \infty$ *with probability 1 and* $\forall x \in X \setminus \{x_t\} : \zeta_t(x) = 0$.
3. $\| \mathrm{E}[F_t | P_t] \|_\infty \leq k \| \Delta_t \|_\infty + c_t$ *where* $k \in [0, 1)$ *and* c_t *converges to 0 with probability 1. Here* $\|y\|_\infty = \max_i(|y_i|)$ *for some* $y \in \mathbb{R}^n$ *denotes the maximum norm.*
4. $\mathrm{Var}[F_t(x_t) | P_t] \leq K(1 + k\|\Delta_t\|_\infty)^2$, *where* K *is some constant.*

Then Δ_t *converges to 0 with probability 1.*

Moreover, we use the following additional lemmata:

Lemma 4. *Let* τ_t, $t \geq 0$, *be a random sequence with* $\mathbb{P}(\tau_t \in [0, 1]) = 1 \ \forall t$. *Define the random process* $x_{t+1} = x_t(1 - \tau_t) + \tau_t$ *for* $t \geq 1$ *and some* $x_0 \in [\varepsilon, 1]$, *where* $0 < \varepsilon \leq 1$. *Then* $\mathbb{P}(x_t \in [\varepsilon, 1]) = 1, \forall t \geq 0$ *holds.*

Proof. We prove the lemma via induction. The statement is true per assumption for x_0. Now assume the statement holds for some $t \geq 0$: $\mathbb{P}(x_t \in [\varepsilon, 1]) = 1$. Then follows: $x_{t+1} = (1 - \tau_t)x_t + \tau_t \geq (1 - \tau_t) \cdot \varepsilon + \tau_t = \varepsilon + \tau_t(1 - \varepsilon) \geq \varepsilon$ with probability 1. Further, we have $x_{t+1} = (1 - \tau_t)x_t + \tau_t \leq (1 - \tau_t) \cdot 1 + \tau_t = 1$ with probability 1. Combining the bounds, we find that $\mathbb{P}(x_{t+1} \in [\varepsilon, 1]) = 1$, proving the induction step and the lemma. □

Lemma 5. *Let* τ_t, $t \geq 0$, *be a random sequence with* $\mathbb{P}(\tau_t \in [0, 1]) = 1 \ \forall t$, $\mathbb{P}(\sum_t \tau_t = \infty) = 1$, *and* $\mathbb{P}(\sum_t \tau_t^2 < \infty) = 1$. *Define the random process* $x_{t+1} = x_t(1 - \tau_t)^2 + \tau_t^2$ *for* $t \geq 1$ *and some* $x_0 > 0$. *Then holds* $\mathbb{P}(\lim_{t\to\infty} x_t = 0) = 1$.

Proof. We begin by unfolding the recursively defined process using simple algebra as follows; compare Jerri [31, Chapter 3]:

$$x_t = x_0 \prod_{i=0}^{t-1}(1 - \tau_i)^2 + \sum_{i=0}^{t-1} \tau_i^2 \prod_{j=i+1}^{t-1}(1 - \tau_j)^2. \tag{D.1}$$

Furthermore, we notice that $\mathbb{P}(x_t > 0) = 1$ for all $t \geq 0$ due to the conditions on x_0 and τ_t. Thus, to prove that $\mathbb{P}(\lim_{t\to\infty} x_t = 0) = 1$, we need to show that each summand in (D.1) converges to zero with probability 1. At first, we have to prove:

$$\mathbb{P}\left(\lim_{t\to\infty} \left[\prod_{i=i_0}^{t}(1 - \tau_i)^2 \right] = 0 \right) = 1, \tag{D.2}$$

for any $i_0 \in \mathbb{N}_0$, where $\mathbb{N}_0$ is the set of natural numbers including zero. To see this, we notice that with probability 1:

$$0 \leq \prod_{i=i_0}^{t}(1 - \tau_i)^2 \leq \prod_{i=i_0}^{t}(1 - \tau_i) \leq \prod_{i=i_0}^{t} \exp(-\tau_i) = \exp\left(-\sum_{i=i_0}^{t} \tau_i \right), \tag{D.3}$$

because $\exp(x) \geq 1 + x$ for all $x \in \mathbb{R}$. It follows with probability 1:

$$0 \leq \lim_{t\to\infty} \left[\prod_{i=i_0}^{t}(1 - \tau_i)^2 \right] \leq \lim_{t\to\infty} \left[\exp\left(-\sum_{i=i_0}^{t} \tau_i \right) \right] = 0, \tag{D.4}$$

which proves (D.2). Note that the last equality in (D.4) follows from the condition that $\mathbb{P}(\sum_t \tau_t = \infty) = 1$. From (D.2) and the assumption that $\mathbb{P}(\sum_t \tau_t^2 < \infty) = 1$, it immediately follows that both summands in (D.1) converge to zero with probability 1. Consequently, we deduce $\mathbb{P}(\lim_{t\to\infty} x_t = 0) = 1$. □

On this basis, we can prove Theorem 1, which we restate in the following.

Theorem 1. *Let the following regularity conditions be fulfilled:*

1. *The MDP is finite.*
2. $\gamma \in [0, 1)$.
3. *The learning rates satisfy* $\tau_t(s,a) \in [0,1]$, $\sum_t \tau_t(s,a) = \infty$, $\sum_t \tau_t^2(s,a) < \infty$ *all with probability 1 for all* $s \in \mathcal{S}, a \in \mathcal{A}$.
4. *The reward function is bounded.*
5. *Each state-action pair is visited infinitely often.*

Then the following holds for the random sequence of action-value estimates $\hat{Q}_t^*$ *generated by TE/KE-Q-Learning:*

$$\mathbb{P}\left[\lim_{t\to\infty} \hat{Q}_t^*(s,a) = Q^*(s,a)\right] = 1 \quad \forall s \in \mathcal{S}, a \in \mathcal{A}.$$

Sketch of Proof. The idea is to apply Lemma 3 with $P_t = \{\hat{Q}_0^*, s_0, a_0, \tau_0, r_1, s_1, \dots, s_t, a_t\}$, $X = \mathcal{S} \times \mathcal{A}$, $\Delta_t = \hat{Q}_t^* - Q^*$, $\zeta_t = \tau_t$, and $F_t(s_t, a_t) = r_t + \gamma\, \mathrm{KE_a}\, \hat{Q}_t^*(s_{t+1}, a) - Q^*(s_t, a_t)$. We first note that conditions 1 and 2 of Lemma 3 hold by regularity conditions 1 and 3 of KE/TE-Q-Learning, respectively. Furthermore, condition 4 of Lemma 3 is guaranteed by assuming a bounded reward function (regularity condition 4); see also the derivations in Barber [5] for an explanation of this point. Finally, condition 3 of Lemma 3 needs to be shown.

Following a similar route as Fujimoto et al. [24], we observe:

$$\begin{aligned}
\Delta_{t+1}(s_t,a_t) &= \hat{Q}_{t+1}^*(s_t,a_t) - Q^*(s_t,a_t) \\
&= \hat{Q}_t^*(s_t,a_t) + \tau_t(s_t,a_t)\left[r_t + \gamma\, \mathrm{KE_a}\, \hat{Q}_t^*(s_{t+1},a) - \hat{Q}_t^*(s_t,a_t)\right] - Q^*(s_t,a_t) + \tau_t(s_t,a_t)Q^*(s_t,a_t) - \tau_t(s_t,a_t)Q^*(s_t,a_t) \\
&= [1-\tau_t(s_t,a_t)]\left[\hat{Q}_t^*(s_t,a_t) - Q^*(s_t,a_t)\right] + \tau_t(s_t,a_t)\left[r_t + \gamma\, \mathrm{KE_a}\, \hat{Q}_t^*(s_{t+1},a) - Q^*(s_t,a_t)\right] \\
&= [1-\tau_t(s_t,a_t)]\Delta_t(s_t,a_t) + \tau_t(s_t,a_t)F_t(s_t,a_t),
\end{aligned}$$

where $F_t(s_t,a_t) = r_t + \gamma\, \mathrm{KE_a}\, \hat{Q}_t^*(s_{t+1},a) - Q^*(s_t,a_t) = F_t^Q(s_t,a_t) + c_t$ if we set $F_t^Q(s_t,a_t) = r_t + \gamma \max_a \hat{Q}_t^*(s_{t+1},a) - Q^*(s_t,a_t)$ and $c_t = \gamma\, \mathrm{KE_a}\, \hat{Q}_t^*(s_{t+1},a) - \gamma \max_a \hat{Q}_t^*(s_{t+1},a)$. We know from standard Q-Learning that $\| \mathrm{E}\left[F_t^Q | P_t\right] \|_\infty \le \gamma \|\Delta_t\|_\infty$. Hence, to show the remaining condition 3 of Lemma 3 and thus prove Theorem 1, we need to show that

$$\lim_{t\to\infty} c_t = \lim_{t\to\infty}\left[\gamma\, \mathrm{KE_a}\, \hat{Q}_t^*(s_{t+1},a) - \gamma \max_a \hat{Q}_t^*(s_{t+1},a)\right] = 0$$

with probability 1. For convenience, we recall the definition of the KE-operator:

$$\mathrm{KE_a}\, \hat{Q}_t^*(s_{t+1},a) = \left\{\sum_{a\in\mathcal{A}} \kappa\left[T_{\hat{Q}_t^*}(s_{t+1},a)\right]\right\}^{-1} \sum_{a'\in\mathcal{A}} \kappa\left[T_{\hat{Q}_t^*}(s_{t+1},a)\right] \hat{Q}_t^*(s_{t+1},a), \tag{D.5}$$

$$T_{\hat{Q}_t^*}(s_{t+1},a) = \frac{\hat{Q}_t^*(s_{t+1},a) - \max_{a'\in\mathcal{A}} \hat{Q}_t^*(s_{t+1},a')}{\sqrt{\widehat{\mathrm{Var}}_t\left[\hat{Q}_t^*(s_{t+1},a)\right] + \widehat{\mathrm{Var}}_t\left[\hat{Q}_t^*(s_{t+1},a^*)\right]}}, \tag{D.6}$$

for a maximizing action $a^* \in \{a \in \mathcal{A} \mid \hat{Q}_t^*(s_{t+1},a) = \max_{a'\in\mathcal{A}} \hat{Q}_t^*(s_{t+1},a')\}$. To ensure that $\mathrm{KE_a}\, \hat{Q}_t^*(s_{t+1},a)$ converges to $\max_a \hat{Q}_t^*(s_{t+1},a)$ with probability 1, we require that the weights of all non-maximizing actions during the summation in (D.5) converge to zero with probability 1. Since we imposed the condition on the kernel function $\kappa(\cdot)$ that $\lim_{x\to-\infty} \kappa(x) = 0$, we have:

$$\mathbb{P}\left(\lim_{t\to\infty} \kappa\left[T_{\hat{Q}_t^*}(s_{t+1},a)\right] = 0\right) = 1 \iff \mathbb{P}\left(\lim_{t\to\infty} T_{\hat{Q}_t^*}(s_{t+1},a) = -\infty\right) = 1,$$

for all $a \neq a^*$. To see whether the test statistics for these actions get arbitrarily small, we need to closely investigate (D.6). We note that the numerator of (D.6) is bounded from above by zero and bounded from below by some real number $C_1 \le 0$ with probability 1 due to assumption 4. Thus, to ensure $\mathbb{P}\left(\lim_{t\to\infty} T_{\hat{Q}_t^*}(s_{t+1},a) = -\infty\right) = 1$, we require $\mathbb{P}\left(\lim_{t\to\infty} \widehat{\mathrm{Var}}_t\left[\hat{Q}_t^*(s_{t+1},a')\right] = 0\right) = 1\ \forall a' \in \mathcal{A}$. In a classical statistical setup, where the variance of a mean estimate converges to 0 with increasing sample size, the proof would already be finished. However, since we also perform incremental updates of the variance, we need to analyze the limit behavior of:

$$\widehat{\mathrm{Var}}_t\left[\hat{Q}_t^*(s_{t+1},a)\right] = \frac{\widehat{\sigma^2}_{\mathrm{pro},t}(s_{t+1},a)}{n_{\mathrm{eff},t}(s_{t+1},a)}, \tag{D.7}$$

where $a \in \mathcal{A}$. The numerator of (D.7) is lower-bounded by zero and upper-bounded by some real number $C_2 \ge 0$ with probability 1, again due to the boundedness assumption on the rewards. Therefore, we have:

$$\mathbb{P}\left(\lim_{t\to\infty} \widehat{\mathrm{Var}}_t\left[\hat{Q}_t^*(s_{t+1},a)\right] = 0\right) = 1 \iff \mathbb{P}\left(\lim_{t\to\infty} n_{\mathrm{eff},t}(s_{t+1},a) = \infty\right) = 1.$$

The effective sample size is computed via $n_{\mathrm{eff},t}(s_{t+1},a) = \frac{[\omega_t(s_{t+1},a)]^2}{\omega_t^2(s_{t+1},a)}$, where numerator and denominator are updated for a visited state-action pair (s_t, a_t) as follows:

$$\omega_{t+1}(s_t, a_t) \leftarrow [1 - \tau_t(s_t, a_t)]\omega_t(s_t, a_t) + \tau_t(s_t, a_t),$$
$$\omega^2_{t+1}(s_t, a_t) \leftarrow [1 - \tau_t(s_t, a_t)]^2\omega^2_t(s_t, a_t) + [\tau_t(s_t, a_t)]^2. \tag{D.8}$$

If we set $\omega_0(s, a) \in [\varepsilon, 1]$ $\forall s \in \mathcal{S}, a \in \mathcal{A}$ for some $0 < \varepsilon \leq 1$, then it follows $\omega_t(s, a) \in [\varepsilon, 1]$ $\forall t$ and $\forall s \in \mathcal{S}, a \in \mathcal{A}$ with probability 1 from Lemma 4. Furthermore, from Lemma 5 it follows that $\lim_{t\to\infty} \omega^2_t(s, a) = 0$ $\forall s \in \mathcal{S}, a \in \mathcal{A}$ with probability 1 if we set, for example, $\omega^2_0(s, a) \in (0, 1]$ $\forall s \in \mathcal{S}, a \in \mathcal{A}$. Combining these findings yields $\lim_{t\to\infty} n_{\text{eff},t}(s, a) = \infty$ $\forall s \in \mathcal{S}, a \in \mathcal{A}$ with probability 1, completing the proof. □

Appendix E. Adaptive TE-BDQN algorithm

Algorithm 4: Ada-TE-BDQN.

initialize Action-value estimate networks with K outputs $\{\hat{Q}^*_k\}_{k=1}^K$, masking distribution M, empty replay buffer D
repeat
  Initialize s
  Pick a value function to act: $k \sim \text{Uniform}\{1, \dots, K\}$
  repeat
    Choose action a from state s with greedy policy derived from $\hat{Q}^*_k$
    Take action a, observe reward r and next state s'
    Sample bootstrap masks $m = (m_1, \dots, m_K)$
    Add (s, a, r, s', m) to replay buffer D
    Sample random minibatch of transitions $\{(s_i, a_i, s'_i, r_i, m^i)\}_{i=1}^B$ from D
    Perform gradient descent step based on (20)
    Every C steps:
      Reset $\theta^-_k = \theta_k$ for $k = 1, \dots, K$
      Run partial episodes to update α via:

$$\alpha \leftarrow \alpha + \frac{\tau_{\text{Ada}}}{K} \sum_{k=1}^{K} \sum_{\bar{t}=1}^{T_{\text{Ada}}} \left[R_k(s_{\bar{t},k}, a_{\bar{t},k}) - \hat{Q}^*_k(s_{\bar{t},k}, a_{\bar{t},k}; \theta_k) \right]$$

    $s \leftarrow s'$
  until *s is terminal*
until

Appendix F. Hyperparameters in MinAtar

Table F.1 details the settings for the experiments in MinAtar [69]. All algorithms were implemented using PyTorch [48] and the computation was performed on Intel(R) Xeon(R) CPUs E5-2680 v3 (12 cores) @ 2.50GHz. The source code is available at: https://github.com/MarWaltz/TUD_RL. Note that we replaced some extreme outlier seeds for the bootstrap-based algorithms in Breakout and Seaquest environments. For example, the algorithm TE-BDQN led to an astonishing peak performance with a test return of over

Table F.1
List of hyperparameters used in the MinAtar experiments. Parameters with a * are used by DQN, DDQN, SCDQN, and MaxMin DQN, while the ones with a † are relevant for BDQN, TE-BDQN, KE-BDQN, and Ada-TE-BDQN. An ‡ exclusively refers to Ada-TE-BDQN.

Hyperparameter	Value
Batch size (B)	32
Discount factor (γ)	0.99
Loss function	MSE
Min. replay buffer size	5 000
Max. replay buffer size	100 000
Optimizer	Adam
Target network update frequency (C)	1 000
Initial exploration rate* ($\epsilon_{\text{initial}}$)	1.0
Final exploration rate* (ϵ_{final})	0.1
Test exploration rate* (ϵ_{test})	0.0
Exploration steps*	100 000
Bernoulli mask probability† (p)	1.0
Number of bootstrap heads† (K)	10
Initial bias parameter‡ (α)	0.25
Time horizon‡ (T_{Ada})	32

200 in a Breakout run, while it got stuck in a rare occasion on Seaquest. Including those exceptions would paint an unrealistic picture of the actual capabilities of the algorithm. A similar phenomenon was observed for all bootstrap-based algorithms, and we argue that those rare instabilities are due to the algorithm's dependence on the initialization of the bootstrap heads.

Appendix G. Further results for MinAtar

For completeness, we present the return and bias development during training in the MinAtar environments with learning rates $\tau \in \{10^{-5}, 10^{-4}\}$ in Figs. G.1–G.5.

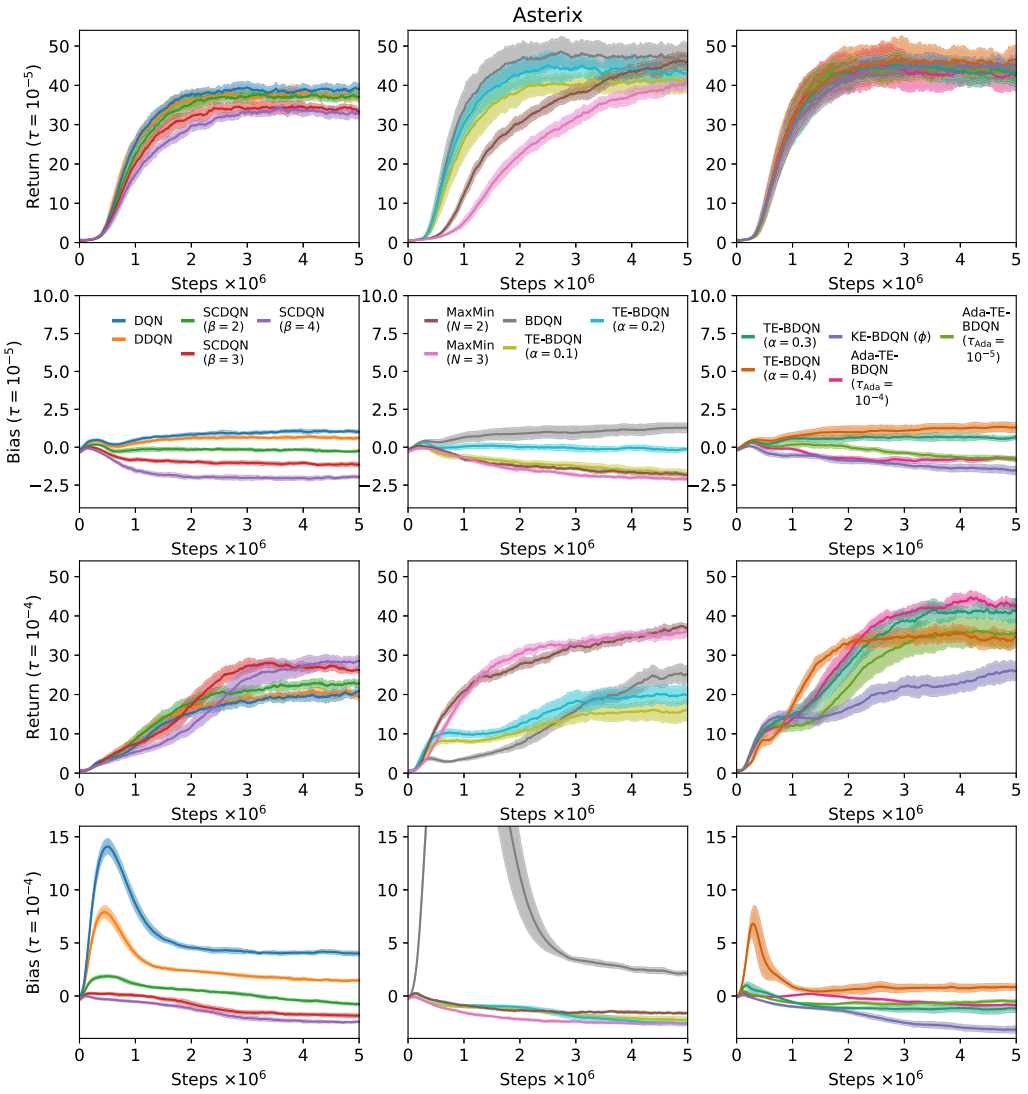


Fig. G.1. Algorithm comparison on Asterix for learning rates $\tau \in \{10^{-5}, 10^{-4}\}$. The first two rows show the return and bias over time for $\tau = 10^{-5}$, while the results for $\tau = 10^{-4}$ are displayed in rows three and four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results. The peak of the bias curve of the BDQN in row four of column two is at approximately 50, which we do not display to ensure the readability of the other curves.

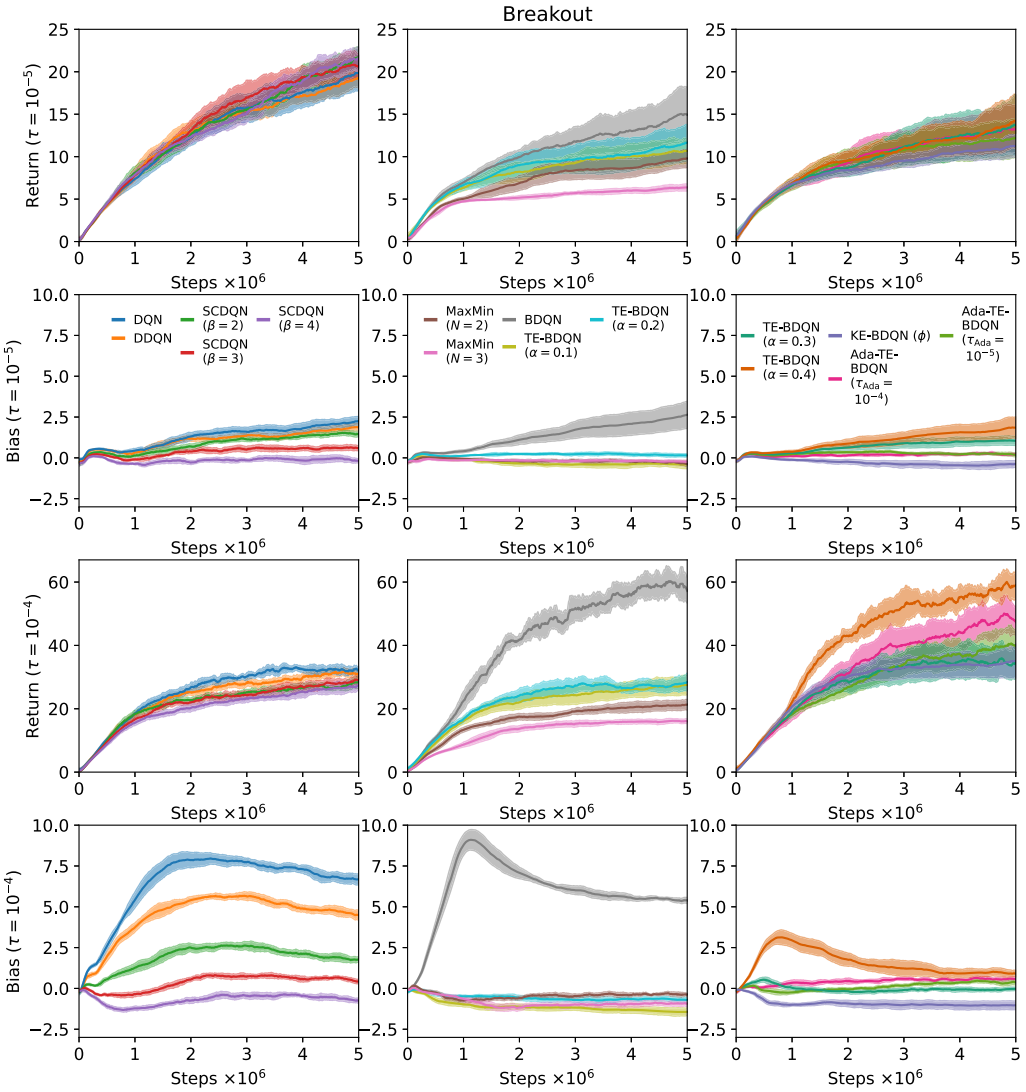


Fig. G.2. Algorithm comparison on Breakout for learning rates $\tau \in \{10^{-5}, 10^{-4}\}$. The first two rows show the return and bias over time for $\tau = 10^{-5}$, while the results for $\tau = 10^{-4}$ are displayed in rows three and four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results.

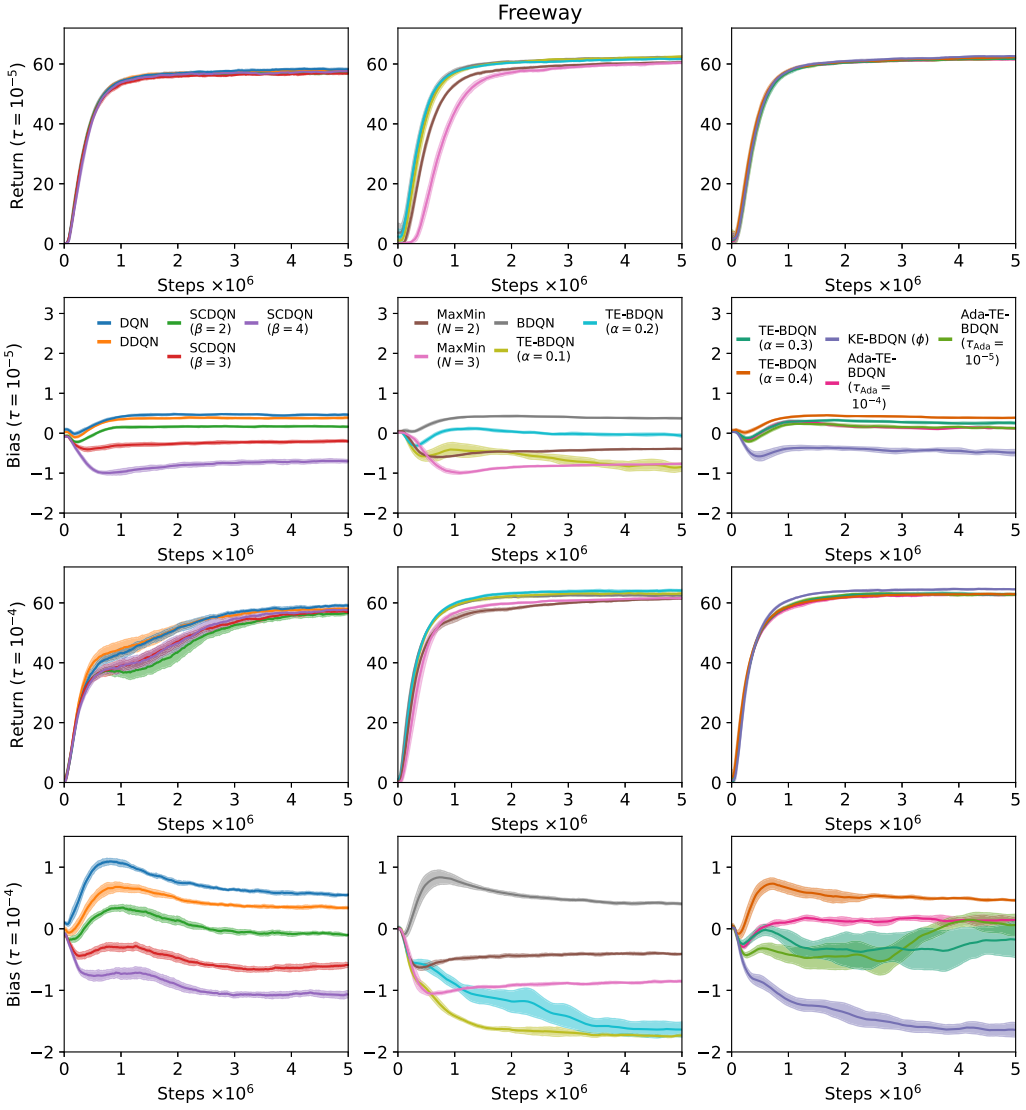


Fig. G.3. Algorithm comparison on Freeway for learning rates $\tau \in \{10^{-5}, 10^{-4}\}$. The first two rows show the return and bias over time for $\tau = 10^{-5}$, while the results for $\tau = 10^{-4}$ are displayed in rows three and four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results.

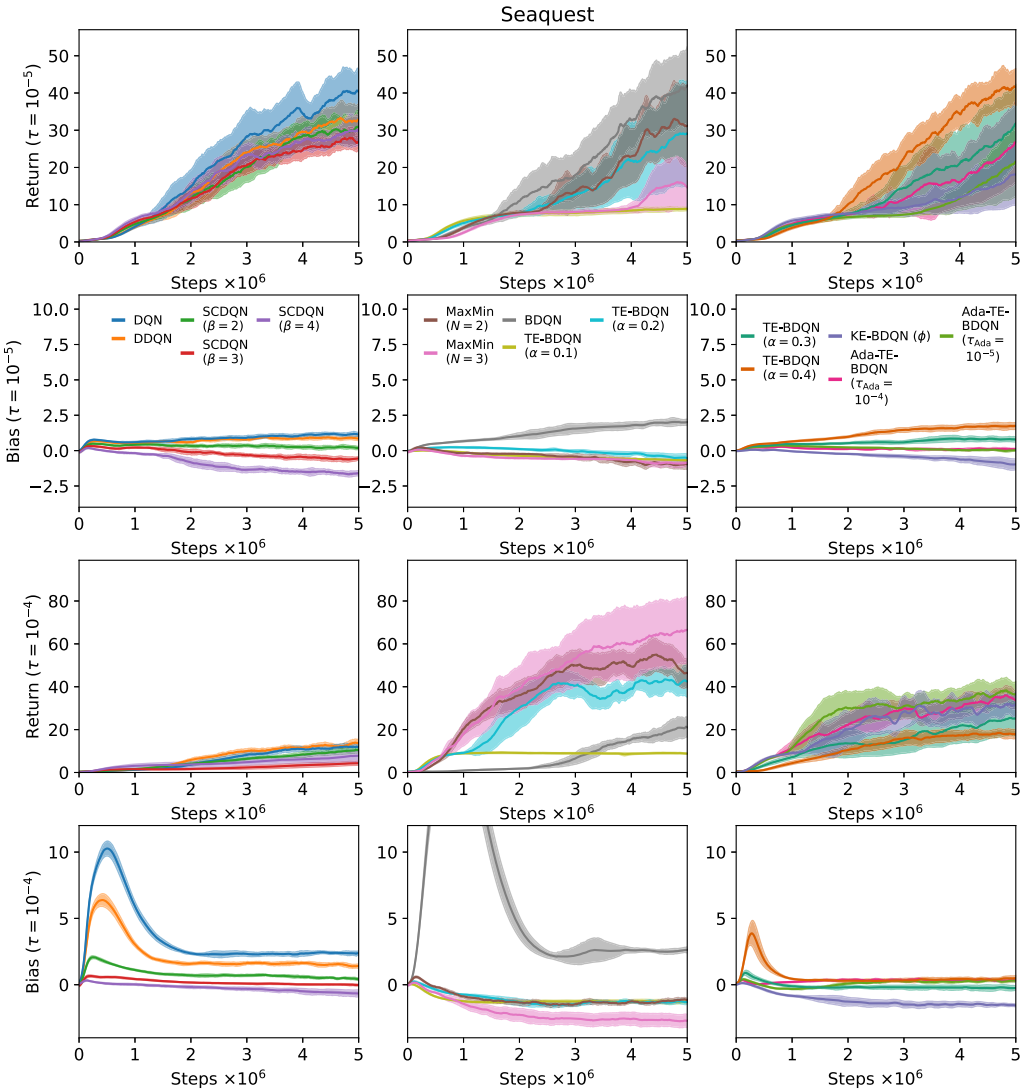


Fig. G.4. Algorithm comparison on Seaquest for learning rates $\tau \in \{10^{-5}, 10^{-4}\}$. The first two rows show the return and bias over time for $\tau = 10^{-5}$, while the results for $\tau = 10^{-4}$ are displayed in rows three and four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results. The peak of the bias curve of the BDQN in row four of column two is at approximately 20, which we do not display to ensure the readability of the other curves.

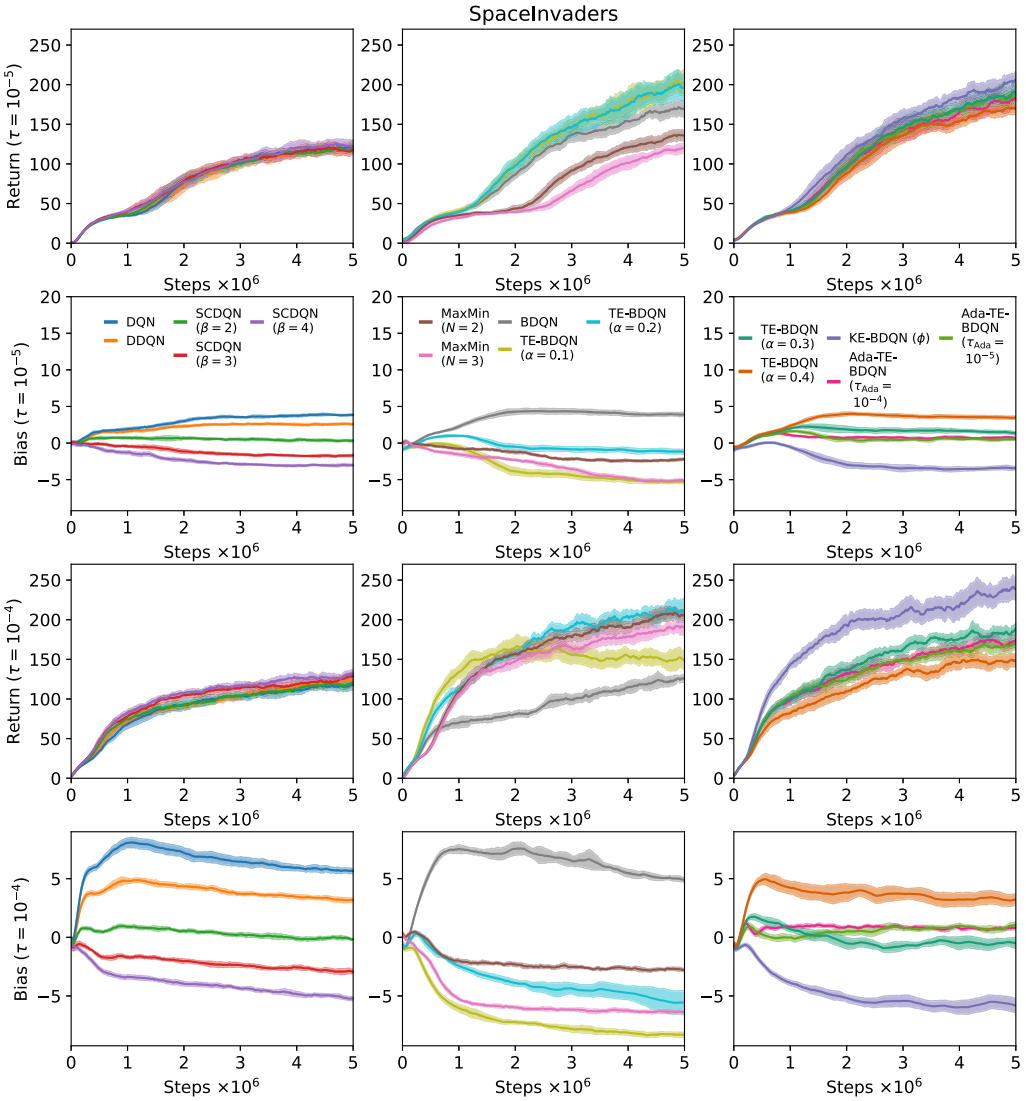


Fig. G.5. Algorithm comparison on SpaceInvaders for learning rates $\tau \in \{10^{-5}, 10^{-4}\}$. The first two rows show the return and bias over time for $\tau = 10^{-5}$, while the results for $\tau = 10^{-4}$ are displayed in rows three and four. Regarding algorithms, the left column includes the DQN, DDQN, and SCDQN; the middle column displays the MaxMin DQN, BDQN, and two TE-BDQNs; and the right column contains the remaining TE-BDQNs, the KE-BDQN, and the Ada-TE-BDQN results.

References

[1] R.A. Armstrong, When to use the Bonferroni correction, Ophthalmic Physiol. Opt. 34 (2014) 502–508.
[2] K. Asadi, M.L. Littman, An alternative softmax operator for reinforcement learning, in: International Conference on Machine Learning, PMLR, 2017, pp. 243–252.
[3] T. Aven, Upper (lower) bounds on the mean of the maximum (minimum) of a number of random variables, J. Appl. Probab. 22 (1985) 723–728.
[4] C. Barata, V. Rotemberg, N.C. Codella, P. Tschandl, C. Rinner, B.N. Akay, Z. Apalla, G. Argenziano, A. Halpern, A. Lallas, et al., A reinforcement learning model for AI-based decision support in skin cancer, Nat. Med. 29 (2023) 1941–1946.
[5] D. Barber, Smoothed Q-learning, arXiv preprint, arXiv:2303.08631, 2023.
[6] M.G. Bellemare, S. Candido, P.S. Castro, J. Gong, M.C. Machado, S. Moitra, S.S. Ponda, Z. Wang, Autonomous navigation of stratospheric balloons using reinforcement learning, Nature 588 (2020) 77–82.
[7] M.G. Bellemare, Y. Naddaf, J. Veness, M. Bowling, The arcade learning environment: an evaluation platform for general agents, J. Artif. Intell. Res. 47 (2013) 253–279.
[8] R. Bellman, The theory of dynamic programming, Bull. Am. Math. Soc. 60 (1954) 503–515.
[9] Y. Benjamini, Y. Hochberg, Controlling the false discovery rate: a practical and powerful approach to multiple testing, J. R. Stat. Soc., Ser. B, Methodol. 57 (1995) 289–300.
[10] D. Bertsekas, Reinforcement Learning and Optimal Control, Athena Scientific, Belmont, 2019.
[11] S. Blumenthal, A. Cohen, Estimation of the larger of two normal means, J. Am. Stat. Assoc. 63 (1968) 861–876.
[12] X. Chen, C. Wang, Z. Zhou, K.W. Ross, Randomized ensembled double Q-learning: learning fast without a model, in: International Conference on Learning Representations, 2021.
[13] Y. Chen, L. Schomaker, M.A. Wiering, An investigation into the effect of the learning rate on overestimation bias of connectionist Q-learning, in: International Conference on Agents and Artificial Intelligence, 2021, pp. 107–118.

[14] W. Dabney, M. Rowland, M. Bellemare, R. Munos, Distributional reinforcement learning with quantile regression, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2018, pp. 2892–2901.
[15] C. D'Eramo, A. Cini, A. Nuara, M. Pirotta, C. Alippi, J. Peters, M. Restelli, Gaussian approximation for bias reduction in Q-learning, J. Mach. Learn. Res. 22 (2021) 1–51.
[16] I. Dhariyal, D. Sharma, K. Krishnamoorthy, Non-existence of unbiased estimators of ordered parameters, Statistics 16 (1985) 89–95.
[17] N. Dorka, T. Welschehold, J. Bödecker, W. Burgard, Adaptively calibrated critic estimates for deep reinforcement learning, IEEE Robot. Autom. Lett. 8 (2022) 624–631.
[18] E.J. Dudewicz, Maximum likelihood estimators for ranked means, Z. Wahrscheinlichkeitstheor. Verw. Geb. 19 (1971) 29–42.
[19] C. D'Eramo, A. Cini, M. Restelli, Exploiting action-value uncertainty to drive exploration in reinforcement learning, in: International Joint Conference on Neural Networks, IEEE, 2019, pp. 1–8.
[20] C. D'Eramo, M. Restelli, A. Nuara, Estimating maximum expected value through Gaussian approximation, in: International Conference on Machine Learning, PMLR, 2016, pp. 1032–1040.
[21] B. Efron, R.J. Tibshirani, An Introduction to the Bootstrap, CRC Press, 1994.
[22] R. Fox, Toward provably unbiased temporal-difference value estimation, in: Optimization Foundations for Reinforcement Learning Workshop at NeurIPS, 2019.
[23] R. Fox, A. Pakman, N. Tishby, Taming the noise in reinforcement learning via soft updates, in: Conference on Uncertainty in Artificial Intelligence, 2016, pp. 202–211.
[24] S. Fujimoto, H. Hoof, D. Meger, Addressing function approximation error in actor-critic methods, in: International Conference on Machine Learning, PMLR, 2018, pp. 1587–1596.
[25] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor, in: International Conference on Machine Learning, PMLR, 2018, pp. 1861–1870.
[26] Z. He, L. Li, S. Zheng, Y. Li, H. Situ, Variational quantum compiling with double Q-learning, New J. Phys. 23 (2021) 033002.
[27] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, D. Silver, Rainbow: combining improvements in deep reinforcement learning, in: AAAI Conference on Artificial Intelligence, 2018, pp. 3215–3222.
[28] G. Hinton, N. Srivastava, K. Swersky, Neural networks for machine learning lecture 6a overview of mini-batch gradient descent, Technical Report, Department of Computer Science, University of Toronto, 2012.
[29] S. Holm, A simple sequentially rejective multiple test procedure, Scand. J. Stat. (1979) 65–70.
[30] T. Imagaw, T. Kaneko, Estimating the maximum expected value through upper confidence bound of likelihood, in: Conference on Technologies and Applications of Artificial Intelligence, IEEE, 2017, pp. 202–207.
[31] A.J. Jerri, Linear Difference Equations with Discrete Transform Methods, Springer Science Business Media, B.V., 1996.
[32] H. Jiang, J. Xie, J. Yang, Action candidate based clipped double Q-learning for discrete and continuous action tasks, in: AAAI Conference on Artificial Intelligence, 2021, pp. 7979–7986.
[33] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv preprint, arXiv:1412.6980, 2014.
[34] L. Kish, Survey Sampling, John Wiley & Sons, New York, 1965.
[35] A. Kuznetsov, A. Grishin, A. Tsypin, A. Ashukha, D. Vetrov, Automating control of overestimation bias for continuous reinforcement learning, arXiv preprint, arXiv:2110.13523, 2021.
[36] A. Kuznetsov, P. Shvechikov, A. Grishin, D. Vetrov, Controlling overestimation bias with truncated mixture of continuous distributional quantile critics, in: International Conference on Machine Learning, PMLR, 2020, pp. 5556–5566.
[37] Q. Lan, Y. Pan, A. Fyshe, M. White, Maxmin Q-learning: controlling the estimation bias of Q-learning, in: International Conference on Learning Representations, 2020.
[38] T. Lattimore, C. Szepesvári, Bandit Algorithms, Cambridge University Press, Cambridge, 2020.
[39] D. Lee, B. Defourny, W.B. Powell, Bias-corrected q-learning to control max-operator bias in q-learning, in: Symposium on Adaptive Dynamic Programming and Reinforcement Learning, IEEE, 2013, pp. 93–99.
[40] K. Lee, M. Laskin, A. Srinivas, P. Abbeel, Sunrise: a simple unified framework for ensemble learning in deep reinforcement learning, in: International Conference on Machine Learning, PMLR, 2021, pp. 6131–6141.
[41] L. Liang, Y. Xu, S.M. McAleer, D. Hu, A. Ihler, P. Abbeel, R. Fox, Temporal-difference value estimation via uncertainty-guided soft updates, in: Deep RL Workshop NeurIPS 2021, 2021.
[42] M.C. Machado, M.G. Bellemare, E. Talvitie, J. Veness, M. Hausknecht, M. Bowling, Revisiting the arcade learning environment: evaluation protocols and open problems for general agents, J. Artif. Intell. Res. 61 (2018) 523–562.
[43] E. Mammen, Estimating a smooth monotone regression function, Ann. Stat. 19 (1991) 724–740.
[44] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (2015) 529–533.
[45] S. Nadarajah, S. Kotz, Exact distribution of the max/min of two Gaussian random variables, IEEE Trans. Very Large Scale Integr. Syst. 16 (2008) 210–212.
[46] I. Osband, J. Aslanides, A. Cassirer, Randomized prior functions for deep reinforcement learning, Adv. Neural Inf. Process. Syst. 31 (2018).
[47] I. Osband, C. Blundell, A. Pritzel, B. Van Roy, Deep exploration via bootstrapped DQN, Adv. Neural Inf. Process. Syst. 29 (2016) 4026–4034.
[48] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: an imperative style, high-performance deep learning library, Adv. Neural Inf. Process. Syst. 32 (2019) 8026–8037.
[49] M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, John Wiley & Sons, 1994.
[50] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Stat. (1951) 400–407.
[51] R.J. Serfling, Approximation Theorems of Mathematical Statistics, John Wiley & Sons, 1980.
[52] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al., Mastering the game of go without human knowledge, Nature 550 (2017) 354–359.
[53] S. Singh, T. Jaakkola, M. Jordan, Reinforcement learning with soft state aggregation, Adv. Neural Inf. Process. Syst. 7 (1994) 361–368.
[54] S. Singh, T. Jaakkola, M.L. Littman, C. Szepesvári, Convergence results for single-step on-policy reinforcement-learning algorithms, Mach. Learn. 38 (2000) 287–308.
[55] A. Slivkins, Introduction to multi-armed bandits, Found. Trends Mach. Learn. 12 (2019) 1–286.
[56] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.
[57] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, The MIT Press, Cambridge, 2018.
[58] S. Thrun, A. Schwartz, Issues in using function approximation for reinforcement learning, in: Proceedings of the Fourth Connectionist Models Summer School, Hillsdale, NJ, 1993, pp. 255–263.
[59] R.S. Tsay, Analysis of Financial Time Series, John Wiley & Sons, New Jersey, 2010.
[60] J.N. Tsitsiklis, Asynchronous stochastic approximation and Q-learning, Mach. Learn. 16 (1994) 185–202.
[61] H. Van Hasselt, Double Q-learning, Adv. Neural Inf. Process. Syst. 23 (2010) 2613–2621.

[62] H. Van Hasselt, Estimating the maximum expected value: an analysis of (nested) cross validation and the maximum sample average, arXiv preprint, arXiv: 1302.7175, 2013.
[63] H. Van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double Q-learning, in: AAAI Conference on Artificial Intelligence, 2016, pp. 2094–2100.
[64] H. Van Seijen, H. Van Hasselt, S. Whiteson, M. Wiering, A theoretical and empirical analysis of expected sarsa, in: Symposium on Adaptive Dynamic Programming and Reinforcement Learning, IEEE, 2009, pp. 177–184.
[65] O. Vinyals, I. Babuschkin, W.M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D.H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al., Grandmaster level in starcraft ii using multi-agent reinforcement learning, Nature 575 (2019) 350–354.
[66] D.D. Wackerly, W. Mendenhall, R.L. Scheaffer, Mathematical Statistics with Applications, 7th ed., Thomson Brooks/Cole, Belmont, CA, 2008.
[67] H. Wang, S. Lin, J. Zhang, Adaptive ensemble Q-learning: minimizing estimation bias via error feedback, Adv. Neural Inf. Process. Syst. 34 (2021).
[68] C.J. Watkins, P. Dayan, Q-learning, Mach. Learn. 8 (1992) 279–292.
[69] K. Young, T. Tian, Minatar: an atari-inspired testbed for thorough and reproducible reinforcement learning experiments, arXiv preprint, arXiv:1903.03176, 2019.
[70] F. Yuan, J. Wu, H. Zhou, L. Liu, A double Q-learning routing in delay tolerant networks, in: International Conference on Communications, IEEE, 2019, pp. 1–6.
[71] Z. Zhang, Z. Pan, M.J. Kochenderfer, Weighted double Q-learning, in: International Joint Conference on Artificial Intelligence, 2017, pp. 3455–3461.
[72] R. Zhu, M. Rigotti, Self-correcting Q-learning, in: AAAI Conference on Artificial Intelligence, 2021, pp. 11185–11192.