Adaptive Order Q-learning

Tao Tan[1] , **Hong Xie**[2*] , **Defu Lian**[2]
[1]College of Computer Science, Chongqing University
[2]University of Science and Technology of China
tantao2020@foxmail.com, xiehong2018@foxmail.com, liandefu@ustc.edu.cn

Abstract

This paper revisits the estimation bias control problem of Q-learning, motivated by the fact that the estimation bias is not always evil, i.e., some environments benefit from overestimation bias or underestimation bias, while others suffer from these biases. Different from previous coarse-grained bias control methods, this paper proposes a fine-grained bias control algorithm called Order Q-learning. It uses the order statistic of multiple independent Q-tables to control bias and flexibly meet the personalized bias needs of different environments, i.e., the bias can vary from underestimation bias to overestimation bias as one selects a higher order Q-value. We derive the expected estimation bias and its lower bound and upper bound. They reveal that the expected estimation bias is inversely proportional to the number of Q-tables and proportional to the index of order statistic function. To show the versatility of Order Q-learning, we design an adaptive parameter adjustment strategy, leading to AdaOrder (Adaptive Order) Q-learning. It adaptively selects the number of Q-tables and the index of order statistic function via the number of visits to state-action pair and the average Q-value. We extend Order Q-learning and AdaOrder Q-learning to the large scale setting with function approximation, leading to Order DQN and AdaOrder DQN, respectively. Finally, we consider two experiment settings: deep reinforcement learning experiments show that our method outperforms several SOTA baselines drastically; tabular MDP experiments reveal fundamental insights into why our method can achieve superior performance. Our supplementary file can be found in https://1drv.ms/f/s!Atddp1iaDmL2gjv31CaGquw5WwYI.

1 Introduction

Q-learning [Watkins, 1989] is one of the most fundamental algorithms in reinforcement learning [Chu *et al.*, 2022; Zhao *et al.*, 2023; Zhang *et al.*, 2023], and it has been successfully applied to various real-world problems [Yang *et al.*, 2020; Upadhyay, 2021; Mock and Muknahallipatna, 2023]. However, Q-learning has an overestimation bias problem [Thrun and Schwartz, 1993; Van Hasselt, 2013], and it can be exacerbated in deep reinforcement learning with nonlinear neural network approximation [Wu *et al.*, 2022; Kondrup *et al.*, 2023; Cetin and Celiktutan, 2023]. This overestimation bias originates from that in the updating rule of Q-learning, the maximum Q-value is an essential term and is obtained by maximizing the stochastic estimation of Q-value. Note that these stochastic estimations are caused by stochastic and unknown rewards and state transitions.

A number of algorithms were proposed to mitigate this overestimation bias, but may lead to an underestimation bias [Hasselt, 2010; Peer *et al.*, 2021; Chen *et al.*, 2021]. The reason is that they are controlling estimation bias in a coarse-grained manner. It is shown in [Lan *et al.*, 2020] and illustrated in Section 3.3 that different environments may have personalized bias needs for underestimation or overestimation. Unfortunately, the coarse-grained property has a fundamental limitation in satisfying such personalized bias needs.

To address the above coarse-grained limitation, we propose a fine-grained bias control algorithm called Order Q-learning. It uses the order statistic of multiple independent Q-tables to control bias and flexibly adapt to different environments. More specifically, the number of Q-tables controls the boundary of bias, i.e., the larger the number of Q-tables, the larger the upper bound, and the smaller the lower bound; the index of order statistic function controls the specific size of bias in the above bounds, i.e., the bias varies from the lower bound to the upper bound as one selects a larger index. Two control parameters of Order Q-learning can provide a fine-grained bias to meet the personalized bias needs of different environments. To address the challenge in Order Q-learning, i.e., the suitable control parameters are unknown in advance, we propose an adaptive parameter adjustment algorithm called AdaOrder Q-learning. Figure 2 shows the power of Order Q-learning and AdaOrder Q-learning. In summary, our contributions are as follows:

1) We propose Order Q-learning, which uses the order statistic of multiple independent Q-tables to attain fine-grained bias control, and aims to meet the personalized bias needs of different environments

*Corresponding Author

2) We design an adaptive parameter adjustment strategy to Order Q-learning, leading to AdaOrder Q-learning. We extend Order Q-learning and AdaOrder Q-learning to Order DQN and AdaOrder DQN, respectively.

3) We evaluate the efficiency of our method in two experiment settings, i.e., tabular MDP and deep reinforcement learning. Extensive experiment results show that our method outperforms SOTA baselines drastically in different settings. More specifically, AdaOrder DQN improves the average score per episode over SOTA baselines by at least 19.49% in the Pixelcopte environments.

2 Related Work

Remove overestimation bias. Double Q-learning [Hasselt, 2010] estimates the maximum Q-value via cross-validation [Van Hasselt, 2013], but it has a risk of an underestimation bias. EBQL [Peer *et al.*, 2021] is a natural extension of Double Q-learning to ensembles. It splits the sample data into at least two disjoint sets: one set estimates the optimal action that attains maximum Q-value; the other sets estimate the mean of Q-values with previously estimated optimal action. However, with a finite number of disjoint sets, the estimation bias of maximum Q-value is always negative. Different from Maxmin Q-learning [Lan *et al.*, 2020], REDQ [Chen *et al.*, 2021] minimizes over random Q-tables, i.e., the default choice for the number of randomly Q-tables is two. The Q-value of REDQ has a lower variance, but it still maintains a negative bias throughout most rounds of learning.

Control estimation bias. Weighted Double Q-learning [Zhang *et al.*, 2017] is a weighted combination of Q-learning and Double Q-learning, which controls the estimation bias through the weight parameter. Like Weighted Double Q-learning, Balanced Q-learning [Karimpanal *et al.*, 2023] weights the optimistic bias and pessimistic bias via the balancing factor. SCQ [Zhu and Rigotti, 2021] uses the current step Q-value and the last step Q-value to build a new self-correcting estimator. This estimator balances the overestimation bias and the underestimation bias via the dependence hyperparameter. Softmax Q-learning [Song *et al.*, 2019] finds that the estimation bias is proportional to the hyperparameter of softmax operation. Ensemble Q-learning [Anschel *et al.*, 2017] averages multiple independent Q-tables to reduce the variance of Q-value. The larger the number of Q-tables, the smaller the variance, and the smaller the estimation bias. This estimation bias is inversely proportional to the number of Q-tables. However, with a finite number of Q-tables, the estimation bias is always positive. Maxmin Q-learning [Lan *et al.*, 2020] controls the estimation bias through minimum operation on multiple independent Q-tables, and this estimation bias is inversely proportional to the number of Q-tables. To adaptively select the number of Q-tables for Maxmin Q-learning [Lan *et al.*, 2020] in practice, AdaEQ [Wang *et al.*, 2021] adjusts the ensemble size based on the approximation Q-value error, and this adjustment method relies on the discounted MC return [Jones and Qin, 2022]. AEQ [Gong *et al.*, 2023] uses the uncertainty of Q-values and the familiarity of sampling trajectories to control the estimation bias naturally, and can adaptively change for specific state-action pairs.

The above bias control methods are coarse-grained, and can not meet the personalized needs of different environments. Different from coarse-grained bias control, this paper proposes a fine-grained bias control algorithm to adapt to different environments.

3 Background & Model

3.1 Basics of an MDP

We consider an infinite-horizon MDP [Chen *et al.*, 2022], where each decision step is indexed by $t \in \mathbb{N}$. Let $\mathcal{S}$ and $\mathcal{A}$ denote the state space and the action space, respectively. Let $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \longrightarrow [0, 1]$ denote the state transition probability function. Let $R : \mathcal{S} \times \mathcal{A} \longrightarrow \mathbb{R}$ denote the reward function. Let $s \mapsto \pi(s)$ denote the policy, where $s \in \mathcal{S}$ and $\pi(s) \in \mathcal{A}$. Let $Q_\pi(s, a)$ denote the expected cumulative discounted reward with discounting factor $\gamma \in (0, 1)$ as:
$Q_\pi(s,a) = \mathbb{E}[R(S_t,A_t) + \gamma R(S_{t+1},A_{t+1}) + \cdots | S_t = s, A_t = a]$,
where S_κ is generated from $P(S_\kappa|S_{\kappa-1}, A_{\kappa-1})$ and $A_\kappa = \pi(S_\kappa)$, $\forall \kappa \geq t + 1$. The learning objective is to find the optimal Q-value and the optimal policy.

3.2 Overestimation Bias & Underestimation Bias

Q-learning [Watkins, 1989] maintains one Q-table denoted by $Q_t(s, a)$, and uses it to estimate the optimal Q-value. It starts with the initialization of Q-table and gets $Q_0(s, a)$. Then in each step t, from state s, the agent takes action a with ε-greedy policy [Ghasemipour *et al.*, 2021], i.e., $\varepsilon \in (0, 1)$, gets reward $R(s, a)$ and next state s'. Q-learning uses the interaction data $\{s, a, R(s, a), s'\}$ to update the Q-table as:

$$\begin{cases} Q_{t+1}(s, a) = Q_t(s, a) + \alpha[Y(s, a) - Q_t(s, a)], \\ Y(s, a) = R(s, a) + \gamma \max\limits_{a' \in \mathcal{A}} Q_t(s', a'), \end{cases} \tag{1}$$

where α is the learning rate, $Y(s, a)$ is the target Q-value of $Q_{t+1}(s, a)$. Following Bellman expectation equation [Bouten *et al.*, 2005], the expectation of target Q-value should be as $\mathbb{E}[R(s, a)] + \gamma \max_{a' \in \mathcal{A}} \mathbb{E}[Q_t(s', a')]$. However, according to Jensen's inequality [Williams *et al.*, 2017], we have:

$$\begin{aligned} \mathbb{E}[Y(s, a)] &= \mathbb{E}\left[R(s, a) + \gamma \max_{a' \in \mathcal{A}} Q_t(s', a')\right] \\ &\geq \mathbb{E}[R(s, a)] + \gamma \max_{a' \in \mathcal{A}} \mathbb{E}[Q_t(s', a')]. \end{aligned}$$

It illustrates the overestimation bias of Q-learning.

Double Q-learning [Hasselt, 2010] maintains two independent Q-tables denoted by $Q^A_t(s, a)$ and $Q^B_t(s, a)$. The updating rule, such as $Q^A_t(s, a)$, is as:

$$\begin{cases} Q^A_{t+1}(s, a) = Q^A_t(s, a) + \alpha[Y^A(s, a) - Q^A_t(s, a)], \\ Y^A(s, a) = R(s, a) + \gamma Q^B_t(s', \arg\max\limits_{a' \in \mathcal{A}} Q^A_t(s', a')). \end{cases} \tag{2}$$

Note that $Q^A_t(s, a)$ and $Q^B_t(s, a)$ are updated with the same probability. Following previous work [Van Hasselt, 2013], we have:

$$\begin{aligned} \mathbb{E}\left[Y^A(s, a)\right] &= \mathbb{E}\left[R(s, a) + \gamma Q^B_t(s', \arg\max_{a' \in \mathcal{A}} Q^A_t(s', a'))\right] \\ &\leq \mathbb{E}[R(s, a)] + \gamma \max_{a' \in \mathcal{A}} \mathbb{E}\left[Q^A_t(s', a')\right]. \end{aligned}$$

It shows the underestimation bias of Double Q-learning.

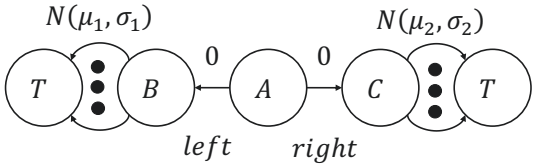


Figure 1: Diagram of a simple MDP.

3.3 Effect of Estimation Bias

Following Maxmin Q-learning [Lan *et al.*, 2020], the effect of estimation bias is not always evil and depends on the environment, i.e., some environments benefit from overestimation bias or underestimation bias, while others suffer from these biases. Further, we use an example to discuss it from a more intuitive and comprehensive perspective.

Example 1. *Consider a simple MDP depicted in Figure 1, which is adapted from Figure 6.5 in [Sutton and Barto, 2018]. This simple episodic MDP task has three non-terminal states, i.e., state A, state B, and state C, and always starts with state A. State A has two actions, i.e., left and right: the right action transitions to the state C with a reward of zero; the left action transitions to the state B with a reward of zero. State B and state C include ten identical actions, and each action transitions to the terminal state with a reward of the distribution $\mathcal{N}(\mu_1, \sigma_1)$ and $\mathcal{N}(\mu_2, \sigma_2)$, respectively. We set $\mu_1 < \mu_2$, $\sigma_1 \geq 0$, and $\sigma_2 \geq 0$. Thus, the optimal action at state A is right. We discuss the effect of estimation bias as follows:*

1) *When $\sigma_1 = 0$ and $\sigma_2 > 0$, the estimation bias occurs in the high reward area. Overestimation bias can attract the agent to choose the right action at state A. Underestimation bias may make the agent choose the left action at state A. Thus, the overestimation bias is helpful, and the underestimation bias is hurtful.*
2) *When $\sigma_1 > 0$ and $\sigma_2 = 0$, the estimation bias occurs in the low reward area. Overestimation bias may induce the agent to choose the left action at state A. Underestimation bias can guide the agent to choose the right action at state A. Thus, the overestimation bias is hurtful, and the underestimation bias is helpful.*
3) *When $\sigma_1 > 0$ and $\sigma_2 > 0$, the estimation bias occurs both in the high reward area and the low reward area. Both overestimation bias and underestimation bias have a risk of making the agent choose the left action at state A. Thus, both overestimation bias and underestimation bias may be hurtful.*

4 Order Q-learning & Order DQN

4.1 Order Q-learning

Example 1 illustrates that the correct approach should be to control the estimation bias according to the environment, such as increasing the estimation bias in case one, reducing the estimation bias in case two, and removing the estimation bias in case three. To meet the personalized bias needs of different environments, we propose a fine-grained bias control algorithm called Order Q-learning.

Algorithm 1 Order Q-learning

Parameter: M, m
Initialize: $Q_0^1(s,a), Q_0^2(s,a), \cdots, Q_0^M(s,a)$
Get the starting state s
for $t = 0, 1, 2, \cdots$ **do**:
  Choose action a at state s by ε-greedy policy
  Take action a, get reward $R(s,a)$ and next state s'
  Compute $Q_t^{m:M}(s', a')$ as Equation (3), $\forall a' \in \mathcal{A}$
  Randomly select one Q-table $i \leq M$
  Update $Q_t^i(s,a)$ as Equation (4)
  $s \longleftarrow s'$

We first maintain $M \in \mathbb{N}^+$ independent Q-tables denoted by $Q_t^1(s,a), Q_t^2(s,a), \cdots, Q_t^M(s,a), \forall (s,a) \in \mathcal{S} \times \mathcal{A}$. Then we reorder them from smallest to largest, and define order Q-value as:

$$Q_t^{m:M}(s,a) = \underset{m:M}{order} \left(Q_t^1(s,a), \cdots, Q_t^M(s,a) \right), \tag{3}$$

where $\underset{m:M}{order}$ is the order statistic function and $m \in [1, M]$. Note that this order statistic function is used to extract the m^{th} largest Q-value. We can write it more explicitly as:

$$Q_t^{1:M}(s,a) = \min_i \left(Q_t^1(s,a), Q_t^2(s,a), \cdots, Q_t^M(s,a) \right),$$
$$\cdots$$
$$Q_t^{M:M}(s,a) = \max_i \left(Q_t^1(s,a), Q_t^2(s,a), \cdots, Q_t^M(s,a) \right).$$

Based on the above order Q-value, Order Q-learning uses ε-greedy policy with order Q-value to interact with the environment and gets $\{s, a, R(s,a), s'\}$. Following previous works [Hasselt, 2010; Lan *et al.*, 2020; Peer *et al.*, 2021], we update each Q-table with the same probability, such as $Q_t^i(s,a)$, as:

$$\begin{cases} Q_{t+1}^i(s,a) = Q_t^i(s,a) + \alpha[Y^i(s,a) - Q_t^i(s,a)], \\ Y^i(s,a) = R(s,a) + \gamma \max\limits_{a' \in \mathcal{A}} Q_t^{m:M}(s', a'), \end{cases} \tag{4}$$

where $i \in [1, M]$ is the index of Q-table. Algorithm 1 outlines our Order Q-learning. Note that Order Q-learning turns to Q-learning [Watkins, 1989] when $M = 1$, and turns to Maxmin Q-learning [Lan *et al.*, 2020] when $m = 1$.

To analyze the estimation bias of Order Q-learning, following previous works [Thrun and Schwartz, 1993; Lan *et al.*, 2020], we first make a common assumption as follows:

Assumption 1. *Each independent Q-table $Q_t^i(s,a)$ has a uniform random error $e_t^i(s,a)$, we have:*

$$Q_t^i(s,a) = \mathbb{E}\left[Q_t^i(s,a)\right] + e_t^i(s,a), \tag{5}$$

where $e_t^i(s,a)$ obeys the uniform distribution $U(-\tau, \tau)$ for some $\tau > 0$. Due to all Q-tables being independent and updated with the same probability, we have:

$$\mathbb{E}\left[Q_t^1(s,a)\right] = \mathbb{E}\left[Q_t^2(s,a)\right] = \cdots = \mathbb{E}\left[Q_t^M(s,a)\right].$$

Equation (4) illustrates the updated rule of our Order Q-learning, where $Y^i(s,a)$ is the target Q-value of $Q^i_{t+1}(s,a)$. Thus, we define the estimation bias of Order Q-learning as:

$$\begin{aligned} Z &= Y^i(s,a) - \left(\mathbb{E}\left[R(s,a)\right] + \gamma \max_{a' \in \mathcal{A}} \mathbb{E}\left[Q^i_t(s',a')\right]\right) \\ &= \left(R(s,a) - \mathbb{E}\left[R(s,a)\right]\right) \\ &\quad + \gamma \left(\max_{a' \in \mathcal{A}} Q^{m:M}_t(s',a') - \max_{a' \in \mathcal{A}} \mathbb{E}\left[Q^i_t(s',a')\right]\right). \end{aligned}$$

Theorem 1. *Based on Assumption 1, the expected estimation bias of Order Q-learning is as:*

$$\mathbb{E}\left[Z\right] = \gamma\left(\tau - \int_{-\tau}^{\tau} \left[\sum_{i=m}^{M} \mathrm{C}^i_M \left[\frac{1}{2} + \frac{x}{2\tau}\right]^i \left[\frac{1}{2} - \frac{x}{2\tau}\right]^{M-i}\right]^{|\mathcal{A}|} dx\right),$$

where $\mathrm{C}^i_M \left[\frac{1}{2} + \frac{x}{2\tau}\right]^i \left[\frac{1}{2} - \frac{x}{2\tau}\right]^{M-i}$ *is the* i^{th} *term probability of binomial distribution* $B(M,p)$*,* $p = \frac{1}{2} + \frac{x}{2\tau}$ *and* $x \in [-\tau, \tau]$*. For compactness, we write* p_i *instead of* $\mathrm{C}^i_M \left[\frac{1}{2} + \frac{x}{2\tau}\right]^i \left[\frac{1}{2} - \frac{x}{2\tau}\right]^{M-i}$*, and rewrite* $\mathbb{E}\left[Z\right]$ *as:*

$$\mathbb{E}\left[Z\right] = \gamma\left(\tau - \int_{-\tau}^{\tau} \left[\sum_{i=m}^{M} p_i\right]^{|\mathcal{A}|} dx\right). \tag{6}$$

According to the binomial distribution $B(M,p)$, we have $\sum_{i=1}^{M} p_i = 1$ and $p_i \geq 0$. Thus, the larger the m, the smaller the $\sum_{i=m}^{M} p_i$, the larger the $\mathbb{E}\left[Z\right]$. Similarly, the larger the M, the larger the $\sum_{i=m}^{M} p_i$, the smaller the $\mathbb{E}\left[Z\right]$. According to the above analysis, we find that the expected estimation bias of Order Q-learning is inversely proportional to M and proportional to m.

Theorem 2. *Based on Assumption 1, the boundary of expected estimation bias for Order Q-learning is as:*

$$\begin{cases} \mathbb{E}[Z] \geq \gamma\left(\tau - 2\tau \dfrac{|\mathcal{A}|\left(|\mathcal{A}|-1\right)\cdots 1}{\left(|\mathcal{A}|+\frac{1}{M}\right)\left(|\mathcal{A}|-1+\frac{1}{M}\right)\cdots\left(1+\frac{1}{M}\right)}\right), \\ \mathbb{E}[Z] \leq \gamma\left(\tau - 2\tau \dfrac{1}{M|\mathcal{A}|+1}\right). \end{cases} \tag{7}$$

Equation (7) illustrates that the boundary depends on M, i.e., the larger the M, the larger the upper bound, the smaller the lower bound. When $M = 1$, the lower bound equals the upper bound, and the expected estimation bias is $\gamma\tau\frac{|\mathcal{A}|-1}{|\mathcal{A}|+1}$. When $M \to +\infty$, the lower bound and the upper bound are $-\gamma\tau$ and $\gamma\tau$, respectively.

Combinate Theorem 1 and Theorem 2, Order Q-learning can provide a fine-grained bias via (M,m) to adapt to different environments, where M controls the boundary of bias and m controls the specific size of bias in the above bounds. More specifically, when overestimation bias is helpful, Order Q-learning can increase the estimation bias by increasing m and decreasing M; when underestimation bias is helpful, Order Q-learning can decrease the estimation bias by decreasing m and increasing M; when both overestimation bias and underestimation bias are hurtful, Order Q-learning can remove the estimation bias by adjusting (m,M). *We prove Theorem 1 and Theorem 2 in our supplementary file.*

Algorithm 2 Order DQN

Parameter: M, m
Initialize: $\theta_0 = \{\theta^1_0, \theta^2_0, \cdots, \theta^M_0\}$, $\theta^-_0 = \theta_0$
Get the starting state s
for $t = 0, 1, 2, \cdots$ **do**:
  Choose action a at state s by ε-greedy policy
  Take action a, get reward r and next state s'
  Store (s, a, r, s') in $\mathcal{D}$
  Randomly sample a set of $(s_\mathcal{B}, a_\mathcal{B}, r_\mathcal{B}, s'_\mathcal{B})$ from $\mathcal{D}$
  for $(s_j, a_j, r_j, s'_j) \in (s_\mathcal{B}, a_\mathcal{B}, r_\mathcal{B}, s'_\mathcal{B})$ **do**:
    $Y_j = r_j + \gamma \max\limits_{a'_j \in \mathcal{A}} Q^{m:M}(s'_j, a'_j; \theta^-_t)$
  Randomly select one Q-function $i \leq M$
  $\theta^i_{t+1} \approx \arg\min_{\theta^i_t} \mathbb{E}_{j \in [1,2,\ldots,|\mathcal{B}|]}\left[Y_j - Q(s_j, a_j; \theta^i_t)\right]^2$
  Every V steps reset $\theta^-_{t+1} \longleftarrow \theta_{t+1}$
  $s \longleftarrow s'$

4.2 Order DQN

Following previous deep reinforcement learning algorithms, such as DQN [Mnih *et al.*, 2015], DDQN [Van Hasselt *et al.*, 2016], Averaged DQN [Anschel *et al.*, 2017], and Maxmin DQN [Lan *et al.*, 2020], we represent the Q-function by a neural network for the high-dimensional environment. Based on experience replay and target network techniques [Mnih *et al.*, 2015], we extend Order Q-learning to Order DQN as Algorithm 2, where θ is the network weight, θ^- is the target network weight, $|\mathcal{D}|$ is the buffer size, $|\mathcal{B}|$ is the batch size, and $V \in \mathbb{N}^+$ is the updated steps for target network weight.

5 AdaOrder Q-learning & AdaOrder DQN

To address the challenge in Order Q-learning, i.e., the suitable parameters (M,m) are unknown in advance, we propose AdaOrder Q-learning and extend it to AdaOrder DQN. In particular, AdaOrder Q-learning is just one of the adaptive adjustment algorithms for Order Q-learning, used to fine-grained control bias close to zero.

5.1 AdaOrder Q-learning

When both overestimation bias and underestimation bias are hurtful, such as the case three in Example 1, Order Q-learning requires multiple experiments to find the suitable parameters (m,M) for removing the estimation bias, and it is time-consuming. Thus, we need to design an adaptive parameter adjustment strategy.

Theorem 2 illustrates that the range of bias is proportional to M. According to temporal difference [Tesauro and others, 1995], the larger the number of visits to each state-action pair, the more accurate the estimated Q-value. Thus, the range of bias should decrease as the number of visits to each state-action pair increases. According to the above analysis, the number of Q-tables for Order Q-learning should decrease with the increase of the number of visits to (s,a) as:

$$M(s,a) = \left\lceil \frac{C}{\left[n(s,a)+1\right]^\beta} \right\rceil, \tag{8}$$

Algorithm 3 AdaOrder Q-learning

Parameter: C, β
Initialize: $Q_0^1(s,a), Q_0^2(s,a), \cdots, Q_0^C(s,a)$
Set $n(s,a)$ as zero, $\forall(s,a) \in (\mathcal{S}, \mathcal{A})$
Get the starting state s
for $t = 0, 1, 2, \cdots$ **do**:
  Choose action a at state s by ε-greedy policy
  Take action a, get reward $R(s,a)$ and next state s'
  $n(s,a) = n(s,a) + 1$
  Compute $M(s', a')$ as Equation (8), $\forall a' \in \mathcal{A}$
  Compute $m(s', a')$ as Equation (9), $\forall a' \in \mathcal{A}$
  Compute $Q_t^{m(s',a'):M(s',a')}(s',a')$ as Equation (3)
  Randomly select one Q-table $i \leq C$
  $Y^i(s,a) = R(s,a) + \gamma \max_{a' \in \mathcal{A}} Q_t^{m(s',a'):M(s',a')}(s',a')$
  $Q_{t+1}^i(s,a) = Q_t^i(s,a) + \alpha[Y^i(s,a) - Q_t^i(s,a)]$
  $s \longleftarrow s'$

where $C \in \mathbb{N}^+$ is the initial number of independent Q-tables, $n(s,a) \in \mathbb{N}$ is the number of visits to (s,a), and $\beta \in (0,1)$ controls the decreasing speed of the number of Q-tables.

Theorem 1 shows that the specific bias of Order Q-learning is proportional to m, i.e., the bias varies from the lower bound to the upper bound as one selects a larger m. According to Ensemble Q-learning [Anschel *et al.*, 2017], it uses the average operation to estimate the maximum expected Q-value. As a result, the estimation bias of Ensemble Q-learning is inversely proportional to the number of Q-tables, and it is always positive. We take Ensemble Q-learning as the baseline and select the m^{th} largest Q-value, which is closest and smaller than the average Q-value. This way can help the estimation bias of Order Q-learning to be close to zero. The index of order statistic function for each state-action pair is as:

$$m(s,a) = \begin{cases} 1 & \text{if } \quad Q_t^{1:M(s,a)}(s,a) \geq \frac{\sum_{i=1}^C Q_t^i(s,a)}{C}, \\ m & \text{if } \begin{cases} Q_t^{m:M(s,a)}(s,a) < \frac{\sum_{i=1}^C Q_t^i(s,a)}{C}, \\ Q_t^{m+1:M(s,a)}(s,a) \geq \frac{\sum_{i=1}^C Q_t^i(s,a)}{C}, \end{cases} \\ M(s,a) & \text{if } \quad Q_t^{M(s,a):M(s,a)}(s,a) \leq \frac{\sum_{i=1}^C Q_t^i(s,a)}{C}. \end{cases} \tag{9}$$

Note that for $Q_t^{m:M(s,a)}(s,a)$, we randomly select $M(s,a)$ Q-tables from C Q-tables as the input to Equation (3).

Incorporating the above adaptive parameter adjustment strategy into Order Q-learning, we obtain Adaptive Order Q-learning as Algorithm 3, and call it AdaOrder Q-learning.

5.2 AdaOrder DQN

During Equation (8), $n(s,a)$ can not be calculated in the high-dimensional or continuous state environment. Due to $M(s,a)$ is inversely proportional to $n(s,a)$ and $n(s,a)$ is proportional to the training step t, we have that the number of Q-functions should be inversely proportional to t. Thus, we adjust the number of Q-functions as:

$$M(t) = \left\lceil C \left(1 - \frac{t}{T+1}\right) \right\rceil, \tag{10}$$

Algorithm 4 AdaOrder DQN

Parameter: C
Initialize: $\theta_0 = \{\theta_0^1, \theta_0^2, \cdots, \theta_0^C\}$, $\theta_0^- = \theta_0$
Get the starting state s
for $t = 0, 1, 2, \cdots$ **do**:
  Compute $M(t)$ as Equation (10)
  Choose action a at state s by ε-greedy policy
  Take action a, get reward r and next state s'
  Store (s, a, r, s') in $\mathcal{D}$
  Randomly sample a set of $(s_\mathcal{B}, a_\mathcal{B}, r_\mathcal{B}, s'_\mathcal{B})$ from $\mathcal{D}$
  for $(s_j, a_j, r_j, s'_j) \in (s_\mathcal{B}, a_\mathcal{B}, r_\mathcal{B}, s'_\mathcal{B})$ **do**
    Compute $m(s'_j, a'_j)$ as Equation (9)
    $Y_j = r_j + \gamma \max_{a'_j \in \mathcal{A}} Q^{m(s'_j,a'_j):M(t)}(s'_j, a'_j; \theta_t^-)$
  Randomly select one Q-function $i \leq C$
  $\theta_{t+1}^i \approx \arg\min_{\theta_t^i} \mathbb{E}_{j \in [1,2,\ldots,|\mathcal{B}|]} \left[Y_j - Q(s_j, a_j; \theta_t^i)\right]^2$
  Every V steps reset $\theta_{t+1}^- \longleftarrow \theta_{t+1}$
  $s \longleftarrow s'$

where $T \in \mathbb{N}^+$ is the total training steps. Like Order DQN in Section 4.2, we extend AdaOrder Q-learning to AdaOrder DQN as Algorithm 4.

6 Experiments

We start with tabular MDP experiments, to reveal fundamental insights into why Order Q-learning and AdaOrder Q-learning can achieve superior performance. We then evaluate the impact of Order DQN and AdaOrder DQN in deep reinforcement learning settings. The code of all experiments can be found in link[1].

6.1 Tabular MDP Experiments

We introduce three tabular MDP environments: (1) Multi-armed bandit is adapted from [Mannor *et al.*, 2007], which considers the single-state with ten action and the reward of each action obeys the distribution $\mathcal{N}(0,1)$; (2) A simple MDP environment is depicted in Figure 1, where $\mu_1 = -0.1$, $\sigma_1 = 1.0$, $\mu_2 = 0.1$, $\sigma_2 = 1.0$; (3) Gridworld [Zhang *et al.*, 2017] has four actions, i.e., up, down, left, and right for each state. The start and goal states are in the southwest and northeast positions of grid, respectively. The agent will return to the start state after it does any action at the goal state. If the agent selects an action and makes it out of grid, i.e., the grid size is 3×3, it will not move and stay in the previous state. The agent gets a stochastic reward of $r_1 = -12$ or $r_2 = 10$ with the same probability for any action at the non-goal state, and gets a stochastic reward of $r_3 = -40$ or $r_4 = 50$ with the same probability for any action at the goal state. Following [Hasselt, 2010; Zhu and Rigotti, 2021; Pentaliotis and Wiering, 2021], we set $\gamma = 0.95$, $\alpha = \frac{1}{n(s,a)^{0.8}}$, and $\varepsilon = \frac{1}{n(s)^{0.5}}$ for the Multi-armed bandit and Gridworld environments; and set $\gamma = 1.0$, $\alpha = 0.1$, and $\varepsilon = 0.1$ for the MDP environment.

[1] https://1drv.ms/u/s!Atddp1iaDmL2ghdcHyYXNO785moD

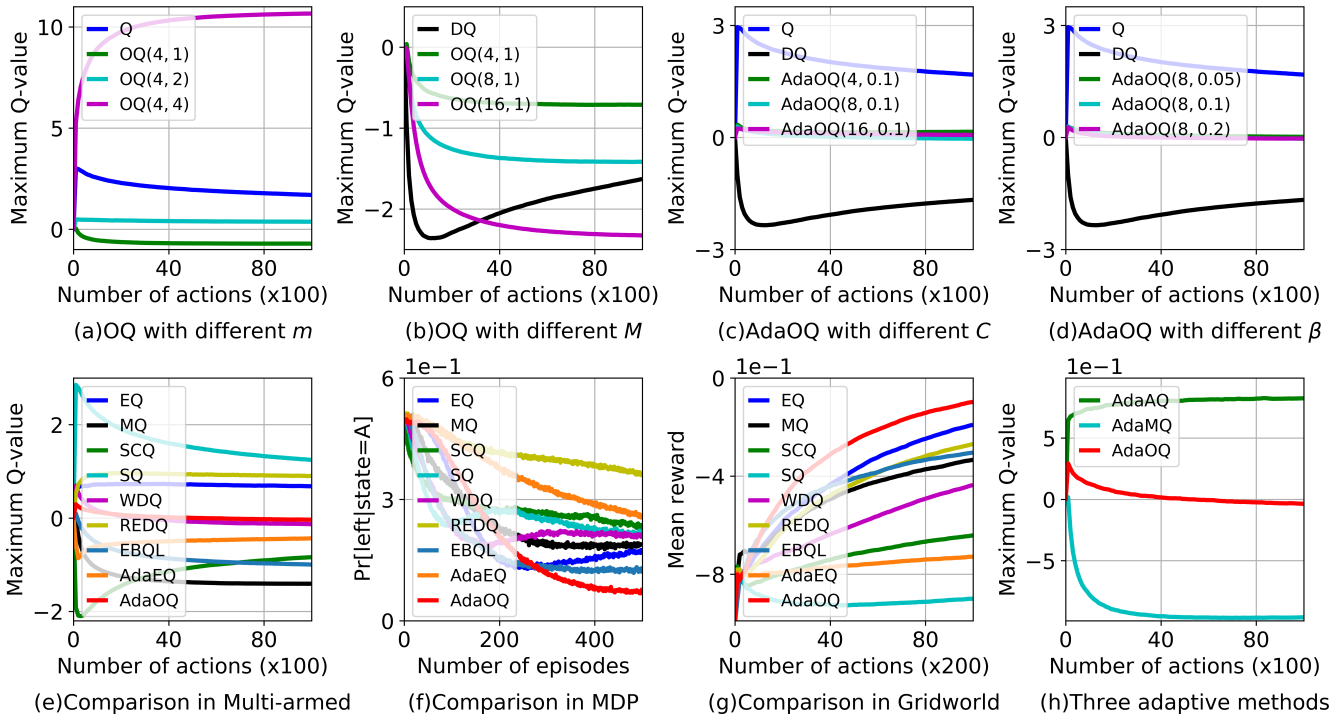


Figure 2: The performance of Order Q-learning and AdaOrder Q-learning in tabular MDP settings. For compactness, we write Q, DQ, EQ, MQ, SQ, WDQ, OQ, and AdaOQ as Q-learning, Double Q-learning, Ensemble Q-learning, Maxmin Q-learning, Softmax Q-learning, Weighted Double Q-learning, Order Q-learning and AdaOrder Q-learning, respectively. We vary $m = 1, 2, 4; M = 4, 8, 16$ for OQ, $C = 4, 8, 16; \beta = 0.05, 0.1, 0.2$ for AdaOQ, and set $(C = 8, \beta = 0.1)$ for default. For a fair comparison, we set the number of Q-tables of EQ, MQ, REDQ, EBQL, and AdaEQ as 8. The hyperparameters for SCQ, SQ, and WDQ are $2, 2$, and 0.5, respectively, which are recommended and fine-tuned. Note that all experimental results are averaged over $1,000$ times.

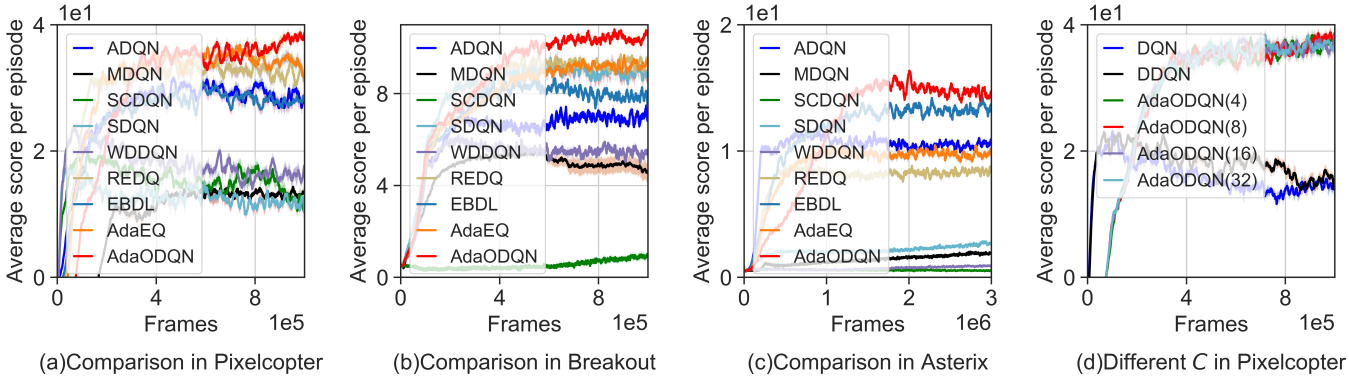
Figure 2 shows the maximum Q-value, the probability of taking action left at state A denoted by $\Pr[left|state = A]$, and the mean reward of Order Q-learning and AdaOrder Q-learning in three tabular MDP environments. *We discuss the performance of Order & AdaOrder Q-learning under different bias preference environments, in our supplementary file.*

Order Q-learning. Figure 2(a) and Figure 2(b) show the maximum Q-value of Order Q-learning with different (M, m) in the Multi-armed bandit environment. One can observe that the estimation bias of Order Q-learning is proportional to m and inversely proportional to M, and it can be higher than that of Q-learning and lower than that of Double Q-learning. This implies that Order Q-learning can provide a more fine-grained bias for meeting the personalized needs of different environments than Q-learning and Double Q-learning.

AdaOrder Q-learning. Figure 2(c) and Figure 2(d) show the maximum Q-value of AdaOrder Q-learning with different (C, β) in the Multi-armed bandit environment. One can observe that the estimation bias curves of AdaOrder Q-learning with different (C, β) overlap, and they are close to zero. This implies that AdaOrder Q-learning is not sensitive to (C, β), and can fine-grained control bias close to zero.

Comparison with SOTA. Figure 2(e), Figure 2(f), and Figure 2(g) show the comparison results in the Multi-armed bandit, MDP, and Gridworld environments, respectively. We consider eight SOTA as Ensemble Q-learning [Anschel *et al.*, 2017], Maxmin Q-learning [Lan *et al.*, 2020], SCQ [Zhu and Rigotti, 2021], Softmax Q-learning [Song *et al.*, 2019], Weighted Double Q-learning [Zhang *et al.*, 2017], REDQ [Chen *et al.*, 2021], EBQL [Peer *et al.*, 2021], and AdaEQ [Wang *et al.*, 2021]. One can observe that the maximum Q-value curve of AdaOrder Q-learning closes to zero, the probability curve of AdaOrder Q-learning lies at the bottom, and the mean reward curve of AdaOrder Q-learning lies at the top. They imply that AdaOrder Q-learning can learn more accurate estimation bias than SOTA through fine-grained bias control, resulting in better policy and higher reward.

Different adaptive methods. Figure 2(h) shows the maximum Q-value of different adaptive methods in the Multi-armed bandit environment, where AdaAQ (Adaptive Average Q-learning) and AdaMQ (Adaptive Minimum Q-learning) are obtained by replacing the order operation of AdaOQ with the average and minimum operations, respectively. Note that we keep the same (C, β) in above three methods. One can observe that the maximum Q-value curve of AdaOrder Q-learning lies between that of AdaAQ and AdaMQ, and is close to zero. This implies that the fine-grained bias control, i.e., order operation, is more suitable for adaptively removing the estimation bias than the coarse-grained bias control, i.e., the average and minimum operations.



(a)Comparison in Pixelcopter (b)Comparison in Breakout (c)Comparison in Asterix (d)Different C in Pixelcopter

Figure 3: The average score per episode of AdaOrder DQN in deep reinforcement learning settings. For compactness, we write ADQN, MDQN, SDQN, WDDQN, and AdaODQN as Averaged DQN, Maxmin DQN, Softmax DQN, Weighted Double DQN, and AdaOrder DQN, respectively. We vary $C = 4, 8, 16, 32$ for AdaODQN, and set $C = 8$ for default. Note that we keep the same hyperparameters of comparison baselines with our tabular MDP experiments, and all experimental results are averaged over 20 repeated runs.

6.2 Deep Reinforcement Learning Experiments

To evaluate the impact of Order DQN and AdaOrder DQN, we choose three common deep reinforcement learning games from PyGame Learning Environment [Urtans and Nikitenko, 2018] and MinAtar [Young and Tian, 2019]: Pixelcopter, Breakout, and Asterix. Following [Young and Tian, 2019; Lan *et al.*, 2020], we reuse the settings of neural networks in [Lan *et al.*, 2020], and set $\gamma = 0.99$, $|\mathcal{B}| = 32$ for all environments. For the Pixelcopter environment, we set $|\mathcal{D}| = 10,000$, $V = 200$, and $\alpha = 0.001$. ε decreases linearly from 1.0 to 0.01 in $1,000$ steps, and fixes to 0.01 after $1,000$ steps. For the Breakout and Asterix environments, we set $|\mathcal{D}| = 100,000$, $V = 1,000$, and $\alpha = 0.01$. ε decreases linearly from 1.0 to 0.1 in $100,000$ steps, and fixes to 0.1 after $100,000$ steps.

Figure 3 shows the average score per episode of AdaOrder DQN in deep reinforcement learning settings, where the score is averaged over the last 100 episodes and the shaded area represents one standard error. From Figure 3(a), Figure 3(b), and Figure 3(c), one can observe that the score curves of AdaOrder DQN lie at the top. They imply that AdaOrder DQN can get a higher score than SOTA baselines. From Figure 3(d), one can observe that the score curves of AdaOrder DQN with different C overlap, and lie much higher than that of DQN and DDQN. This implies that AdaOrder DQN is not sensitive to C, and outperforms DQN and DDQN drastically. *We provide the experiment results of Order DQN with different (M, m), and discuss the performance of different adaptive methods, in our supplementary file.*

Table 1 shows the average score per episode of different algorithms, where the values are evaluated in the final round. We consider ten SOTA baselines as DQN [Mnih *et al.*, 2015], DDQN [Van Hasselt *et al.*, 2016], Averaged DQN [Anschel *et al.*, 2017], Maxmin DQN [Lan *et al.*, 2020], SCDQN [Zhu and Rigotti, 2021], Softmax DQN [Song *et al.*, 2019], WDDQN [Zhang *et al.*, 2017], REDQ [Chen *et al.*, 2021], EBQL [Peer *et al.*, 2021], and AdaEQ [Wang *et al.*, 2021]. Our AdaOrder DQN improves the average score per episode over SOTA baselines by at least $\frac{38.57-32.28}{32.28} = 19.49\%$, $\frac{10.68-9.44}{9.44} = 13.14\%$, and $\frac{14.57-13.33}{13.33} = 9.30\%$, in the Pixelcopter, Breakout, and Asterix environments, respectively.

Algorithm	Average score per episode		
	Pixelcopter	Breakout	Asterix
DQN	14.09	5.06	0.65
DDQN	13.82	5.29	0.83
Averaged DQN	28.50	6.85	10.70
Maxmin DQN	13.31	4.66	1.93
SCDQN	12.23	0.89	0.56
Softmax DQN	11.22	8.90	2.73
WDDQN	17.57	5.39	0.92
REDQ	32.28	9.41	8.41
EBQL	28.06	7.42	13.33
AdaEQ	31.77	9.44	9.91
AdaOrder DQN(Ours)	**38.57**	**10.68**	**14.57**

Table 1: The average score per episode of different algorithms.

7 Conclusion

Different from previous coarse-grained bias control manner, this paper proposes Order Q-learning, which controls the estimation bias via the order statistic of multiple independent Q-tables and can provide a fine-grained bias to flexibly meet the personalized bias needs of different environments. Based on Order Q-learning, we design an adaptive parameter adjustment strategy, leading to AdaOrder Q-learning. We extend Order Q-learning and AdaOrder Q-learning to Order DQN and AdaOrder DQN, respectively. Extensive experiment results show that our proposed method outperforms SOTA baselines drastically in different settings.

We think future work should focus on the adaptive adjustment strategies, finding more flexible adaptive adjustment strategies to adapt to highly complex or changing environments, such as non-stationary environments and reward sparse environments.

Acknowledgments

The work of Hong Xie was supported in part by National Nature Science Foundation of China (61902042), Chongqing Natural Science Foundation(cstc2020jcyi-msxmX0652).

References

[Anschel *et al.*, 2017] Oron Anschel, Nir Baram, and Nahum Shimkin. Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. In *International conference on machine learning*, pages 176–185. PMLR, 2017.

[Bouten *et al.*, 2005] Luc Bouten, Simon Edwards, and VP Belavkin. Bellman equations for optimal feedback control of qubit states. *Journal of Physics B: Atomic, Molecular and Optical Physics*, 38(3):151, 2005.

[Cetin and Celiktutan, 2023] Edoardo Cetin and Oya Celiktutan. Learning pessimism for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 6971–6979, 2023.

[Chen *et al.*, 2021] Xinyue Chen, Che Wang, Zijian Zhou, and Keith W. Ross. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021.

[Chen *et al.*, 2022] Liyu Chen, Rahul Jain, and Haipeng Luo. Learning infinite-horizon average-reward markov decision process with constraints. In *International Conference on Machine Learning*, pages 3246–3270. PMLR, 2022.

[Chu *et al.*, 2022] Chen Chu, Yong Li, Jinzhuo Liu, Shuyue Hu, Xuelong Li, and Zhen Wang. A formal model for multiagent q-learning dynamics on regular graphs. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 194–200. ijcai.org, 2022.

[Ghasemipour *et al.*, 2021] Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pages 3682–3691. PMLR, 2021.

[Gong *et al.*, 2023] Xiaoyu Gong, Shuai Lü, Jiayu Yu, Sheng Zhu, and Zongze Li. Adaptive estimation q-learning with uncertainty and familiarity. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 3750–3758. ijcai.org, 2023.

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

[Jones and Qin, 2022] Galin L Jones and Qian Qin. Markov chain monte carlo in practice. *Annual Review of Statistics and Its Application*, 9:557–578, 2022.

[Karimpanal *et al.*, 2023] Thommen George Karimpanal, Hung Le, Majid Abdolshah, Santu Rana, Sunil Gupta, Truyen Tran, and Svetha Venkatesh. Balanced q-learning: Combining the influence of optimistic and pessimistic targets. *Artificial Intelligence*, 325:104021, 2023.

[Kondrup *et al.*, 2023] Flemming Kondrup, Thomas Jiralerspong, Elaine Lau, Nathan de Lara, Jacob Shkrob, My Duc Tran, Doina Precup, and Sumana Basu. Towards safe mechanical ventilation treatment using deep offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15696–15702, 2023.

[Lan *et al.*, 2020] Qingfeng Lan, Yangchen Pan, Alona Fyshe, and Martha White. Maxmin q-learning: Controlling the estimation bias of q-learning. In *International Conference on Learning Representations*, 2020.

[Mannor *et al.*, 2007] Shie Mannor, Duncan Simester, Peng Sun, and John N Tsitsiklis. Bias and variance approximation in value function estimates. *Management Science*, 53(2):308–322, 2007.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[Mock and Muknahallipatna, 2023] James W Mock and Suresh S Muknahallipatna. A comparison of ppo, td3 and sac reinforcement algorithms for quadruped walking gait generation. *Journal of Intelligent Learning Systems and Applications*, 15(1):36–56, 2023.

[Peer *et al.*, 2021] Oren Peer, Chen Tessler, Nadav Merlis, and Ron Meir. Ensemble bootstrapping for q-learning. In *International Conference on Machine Learning*, pages 8454–8463. PMLR, 2021.

[Pentaliotis and Wiering, 2021] Andreas Pentaliotis and Marco A Wiering. Variation-resistant q-learning: Controlling and utilizing estimation bias in reinforcement learning for better performance. In *ICAART (2)*, pages 17–28, 2021.

[Song *et al.*, 2019] Zhao Song, Ron Parr, and Lawrence Carin. Revisiting the softmax bellman operator: New benefits and new perspective. In *International Conference on Machine Learning*, pages 5916–5925. PMLR, 2019.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. 2018.

[Tesauro and others, 1995] Gerald Tesauro et al. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68, 1995.

[Thrun and Schwartz, 1993] Sebastian Thrun and Anton Schwartz. Issues in using function approximation for reinforcement learning. In *Proceedings of the Fourth Connectionist Models Summer School*, pages 255–263. Hillsdale, NJ, 1993.

[Upadhyay, 2021] Indrima Upadhyay. *ANALYSIS OF Q-LEARNING BASED GAME PLAYING AGENTS FOR ABSTRACT BOARD GAMES WITH INCREASING STATE-*

SPACE COMPLEXITY. PhD thesis, Miami University, 2021.

[Urtans and Nikitenko, 2018] Evalds Urtans and Agris Nikitenko. Survey of deep q-network variants in pygame learning environment. In *Proceedings of the 2018 2nd International Conference on Deep Learning Technologies*, pages 27–36, 2018.

[Van Hasselt *et al.*, 2016] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

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

[Wang *et al.*, 2021] Hang Wang, Sen Lin, and Junshan Zhang. Adaptive ensemble q-learning: Minimizing estimation bias via error feedback. *Advances in Neural Information Processing Systems*, 34, 2021.

[Watkins, 1989] Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

[Williams *et al.*, 2017] Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M Rehg, Byron Boots, and Evangelos A Theodorou. Information theoretic mpc for model-based reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1714–1721. IEEE, 2017.

[Wu *et al.*, 2022] Haolin Wu, Jianwei Zhang, Zhuang Wang, Yi Lin, and Hui Li. Sub-avg: Overestimation reduction for cooperative multi-agent reinforcement learning. *Neurocomputing*, 474:94–106, 2022.

[Yang *et al.*, 2020] Yongjian Yang, Xintao Wang, Yuanbo Xu, and Qiuyang Huang. Multiagent reinforcement learning-based taxi predispatching model to balance taxi supply and demand. *Journal of Advanced Transportation*, 2020, 2020.

[Young and Tian, 2019] Kenny Young and Tian Tian. Minatar: An atari-inspired testbed for more efficient reinforcement learning experiments. *arXiv preprint arXiv:1903.03176*, 59:60, 2019.

[Zhang *et al.*, 2017] Zongzhang Zhang, Zhiyuan Pan, and Mykel J Kochenderfer. Weighted double q-learning. In *IJCAI*, pages 3455–3461, 2017.

[Zhang *et al.*, 2023] Zhe Zhang, Yukun Zou, Junjie Lai, and Qing Xu. M2DQN: A robust method for accelerating deep q-learning network. In *Proceedings of the 15th International Conference on Machine Learning and Computing, ICMLC 2023, Zhuhai, China, February 17-20, 2023*, pages 116–120. ACM, 2023.

[Zhao *et al.*, 2023] Fuqing Zhao, Qiaoyun Wang, and Ling Wang. An inverse reinforcement learning framework with the q-learning mechanism for the metaheuristic algorithm. *Knowledge-Based Systems*, 265:110368, 2023.

[Zhu and Rigotti, 2021] Rong Zhu and Mattia Rigotti. Self-correcting q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11185–11192, 2021.