Self-correcting Q-Learning

Rong Zhu[1]*, **Mattia Rigotti** [2]
[1] ISTBI, Fudan University
[2] IBM Research AI
rongzhu56@gmail.com, mr2666@columbia.edu

Abstract

The Q-learning algorithm is known to be affected by the *maximization bias*, i.e. the systematic overestimation of action values, an important issue that has recently received renewed attention. Double Q-learning has been proposed as an efficient algorithm to mitigate this bias. However, this comes at the price of an *underestimation* of action values, in addition to increased memory requirements and a slower convergence. In this paper, we introduce a new way to address the maximization bias in the form of a "self-correcting algorithm" for approximating the maximum of an expected value. Our method balances the overestimation of the single estimator used in conventional Q-learning and the underestimation of the double estimator used in Double Q-learning. Applying this strategy to Q-learning results in *Self-correcting Q-learning*. We show theoretically that this new algorithm enjoys the same convergence guarantees as Q-learning while being more accurate. Empirically, it performs better than Double Q-learning in domains with rewards of high variance, and it even attains faster convergence than Q-learning in domains with rewards of zero or low variance. These advantages transfer to a Deep Q Network implementation that we call *Self-correcting DQN* and which outperforms regular DQN and Double DQN on several tasks in the Atari 2600 domain.

1 Introduction

The goal of Reinforcement Learning (RL) is to learn to map situations to actions so as to maximize a cumulative future reward signal (Sutton and Barto 2018). Q-learning proposed by Watkins (1989) is one of the most popular algorithms for solving this problem, and does so by estimating the optimal action value function. The convergence of Q-learning has been proven theoretically for discounted Markov Decision Processes (MDPs), and for undiscounted MDPs with the condition that all policies lead to a zero-cost absorbing state (Watkins and Dayan 1992; Tsitsiklis 1994; Jaakkola, Jordan, and Singh 1994). Q-learning has been widely successfully deployed on numerous practical RL problems in fields including control, robotics (Kober, Bagnell, and Peters 2013) and human-level game play (Mnih et al. 2015).

*Part of the work was done while at Columbia University

However, Q-learning is known to incur a *maximization bias*, i.e., the overestimation of the maximum expected action value, which can result in poor performance in MDPs with stochastic rewards. This issue was first pointed out by Thrun and Schwartz (1993), and further investigated by van Hasselt (2010) that proposed Double Q-learning as a way of mitigating the problem by using the so-called double estimator, consisting in two separately updated action value functions. By decoupling action selection and reward estimation with these two estimators, Double Q-learning addresses the overestimation problem, but at the cost of introducing a systematic underestimation of action values. In addition, when rewards have zero or low variances, Double Q-learning displays slower convergence than Q-learning due to its alternation between updating two action value functions.

The main contribution of this paper is the introduction of a new *self-correcting estimator* to estimate the maximum expected value. Our method is based on the observation that in Q-learning successive updates of the Q-function at time steps n and $n-1$ are correlated estimators of the optimal action value that can be combined into a new "self-correcting" estimator, once the resulting combination is maximized over actions. Crucially, using one value function (at different time steps) allows us to avoid having to update two action value functions. First, we will show how to combine correlated estimators to obtain a self-correcting estimator that is guaranteed to balance the overestimation of the single estimator and the underestimation of the double estimator. We then show that, if appropriately tuned, this strategy can completely remove the maximization bias, which is particularly pernicious in domains with high reward variances. Moreover, it can also attain faster convergence speed than Q-learning in domains where rewards are deterministic or with low variability. Importantly, Self-correcting Q-learning does not add any additional computational or memory costs compared to Q-learning. Finally, we propose a Deep Q Network version of Self-correcting Q-learning that we successfully test on the Atari 2600 domain.

Related work. Beside Double Q-learning, other methods have been proposed to reduce the maximization bias, e.g., removing the asymptotic bias of the max-operator under a Gaussian assumption (Lee, Defourny, and Powell 2013), estimating the maximum expected value by Gaussian approx-

imation (D'Eramo, Nuara, and Restelli 2016), averaging Q-values estimates (Anschel, Baram, and Shimkin 2017), and softmax Bellman operator (Song, Parr, and Carin 2019), clipping values in an actor-critic setting (Dorka, Boedecker, and Burgard 2019). Fitted Q-iteration (Ernst, Geurts, and Wehenkel 2005), Speedy Q-learning (Azar et al. 2011), and Delayed Q-learning (Strehl et al. 2006) are related variations of Q-learning with faster convergence rate. In its basic form, our approach is a generalization of regular Q-learning. But importantly it can be applied to any other variant of Q-learning to reduce its maximization bias. Finally, DeepMellow (Kim et al. 2019) establishes that using a target network in DQN (Mnih et al. 2015) also reduces maximization bias, and proposes a soft maximization operator as an alternative.

Paper organization. In Section 2, we review MDPs and Q-learning. In Section 3, we consider the problem of estimating the maximum expected value of a set of random variables, and review the maximization bias, the single estimator and the double estimator. Then we propose the self-correcting estimator, and show that this estimator can avoid both the overestimation of the single estimator and the underestimation of the double estimator. In Section 4 we apply the self-correcting estimator to Q-learning and propose Self-correcting Q-learning which converges to the optimal solution in the limit. In Section 5 we implement a Deep Neural Network version of Self-correcting Q-learning. In Section 6 we show the results of several experiments empirically examining these algorithms. Section 7 concludes the paper with consideration on future directions.

2 Markov Decision Problems and Q-learning

We will start recalling the application of Q-learning to solve MDPs. Let $Q_n(s,a)$ denote the estimated value of action a in state s at time step n. An MDP is defined such that the next state s' is determined by a fixed state transition distribution $P : S \times A \times S \to [0,1]$, where $P_{ss'}(a)$ gives the probability of ending up in state s' after performing action a in s, and satisfies $\sum_{s'} P_{ss'}(a) = 1$. The reward r is drawn from a reward distribution with $\mathrm{E}(r|s,a,s') = R_{ss'}(a)$ for a given reward function $R_{ss'}(a)$. The optimal value function $Q^*(s,a)$ is the solution to the so-called *Bellman equations*: $Q^*(s,a) = \sum_{s'} P_{ss'}(a)\left[R_{ss'}(a) + \gamma \max_a Q^*(s',a)\right] \forall s,a$, where $\gamma \in [0,1)$ is the discount factor. As a solution of an MDPs, (Watkins 1989) proposed Q-learning, which consists in the following recursive update:

$$Q_{n+1}(s,a) = Q_n(s,a) + \alpha(s,a)\left[r + \gamma \max_a Q_n(s',a) - Q_n(s,a)\right]. \quad (1)$$

Notice that in this expression the $\max$ operator is used to estimate the value of the next state. Recently, (van Hasselt 2010) showed that this use of the maximum value as an approximation for the maximum expected value introduces a *maximization bias*, which results in Q-learning overestimating action values.

3 Maximum Expected Value Estimation

The single estimator. We begin by looking at estimating the maximum expected value. Consider a set of M random variables $\{Q(a_i)\}_{i=1}^M$. We are interested in estimating their maximum expected value, $\max_i \mathrm{E}[Q(a_i)]$. Clearly however, it is infeasible to know $\max_i \mathrm{E}[Q(a_i)]$ exactly in absence of any assumption on their underlying distributions. One natural way to approximate the maximum expected value is through the maximum $\max_i\{Q(a_i)\}$, which is called the *single estimator*. As noticed, Q-learning uses this method to approximate the value of the next state by maximizing over the estimated action values. Although $Q(a_i)$ is an unbiased estimator of $\mathrm{E}[Q(a_i)]$, from Jensen's inequality we get that $\mathrm{E}[\max_i\{Q(a_i)\}] \geq \max_i \mathrm{E}[Q(a_i)]$, meaning that the single estimator is positively biased. This is the so-called *maximization bias*, which interestingly has also been investigated in fields outside of RL, such as economics (Van den Steen 2004), decision making (Smith and Winkler 2006), and auctions (Thaler 1988).

The double estimator. The paper (van Hasselt 2010) proposed to address the maximization bias by introducing the *double estimator*. Assume that there are two independent, unbiased sets of estimators of $\{\mathrm{E}[Q(a_i)]\}_{i=1}^M$: $\{Q^A(a_i)\}_{i=1}^M$ and $\{Q^B(a_i)\}_{i=1}^M$. Let a_D^* be the action that maximizes $\{Q^A(a_i)\}_{i=1}^M$, that is, $Q^A(a_D^*) = \max_i\{Q^A(a_i)\}$. The double estimator uses $Q^B(a_D^*)$ as an estimator for $\max_i \mathrm{E}[Q(a_i)]$. This estimator is unbiased in the sense that $\mathrm{E}[Q^B(a_D^*)] = \mathrm{E}[Q(a_D^*)]$ due to the independence of $\{Q^A(a_i)\}_{i=1}^M$ and $\{Q^B(a_i)\}_{i=1}^M$. However, this estimator has a tendency towards underestimation, i.e., $\mathrm{E}[Q^B(a_D^*)] \leq \max_i \mathrm{E}[Q^B(a_i)]$ (see details in van Hasselt 2010).

The self-correcting estimator. Let us now consider two independent and unbiased sets of estimators of $\{\mathrm{E}[Q(a_i)]\}_{i=1}^M$, given by $\{Q^{B_1}(a_i)\}_{i=1}^M$ and $\{Q^{B_\tau}(a_i)\}_{i=1}^M$. From them, we construct another unbiased set of estimators $\{Q^{B_0}(a_i)\}_{i=1}^M$ by defining

$$Q^{B_0}(a_i) = \tau Q^{B_1}(a_i) + (1-\tau)Q^{B_\tau}(a_i), \quad (2)$$

where $\tau \in [0,1)$ denotes the degree of dependence between $Q^{B_0}(a_i)$ and $Q^{B_1}(a_i)$. Eqn. (2) clearly establishes that $Q^{B_0}(a_i)$ and $Q^{B_1}(a_i)$ are non-negatively correlated, unbiased estimators of $\mathrm{E}[Q(a_i)]$. Let σ_1^2 and σ_τ^2 be the variances of $Q^{B_1}(a_i)$ and $Q^{B_\tau}(a_i)$, respectively. The Pearson correlation coefficient between $Q^{B_0}(a_i)$ and $Q^{B_1}(a_i)$ is $\rho = \tau\sigma_1/\sqrt{\tau^2\sigma_1^2 + (1-\tau)^2\sigma_\tau^2}$. When $\tau \to 1$, $Q^{B_0}(a_i)$ is completely correlated with $Q^{B_1}(a_i)$. While as τ becomes smaller, the correlation is weaker.

Denoting $\beta = 1/(1-\tau)$,we rewrite Eqn. (2) as

$$Q^{B_\tau}(a_i) = Q^{B_1}(a_i) - \beta[Q^{B_1}(a_i) - Q^{B_0}(a_i)]. \quad (3)$$

Let a_τ^* indicate the action maximizing $\{Q^{B_\tau}(a_i)\}_{i=1}^M$, i.e. $a_\tau^* = \arg\max_{a_i} Q^{B_\tau}(a_i)$. We call $Q^{B_0}(a_\tau^*)$ *self-correcting estimator* of $\mathrm{E}[Q(a_i)]$ because in a Q-learning setting the roles of $Q^{B_0}(a_i)$ and $Q^{B_1}(a_i)$ are going to be taken up by sequential terms of Q-learning updates (see next section).

Lemma 1 *Consider a set of M random variables $\{Q(a_i)\}_{i=1}^M$ with the expected values $\{E[Q(a_i)]\}_{i=1}^M$. Let $\{Q^{B_0}(a_i)\}_{i=1}^M$, $\{Q^{B_1}(a_i)\}_{i=1}^M$, and $\{Q^{B_\tau}(a_i)\}_{i=1}^M$ be unbiased sets of estimators satisfying the relation Eqn.* (2)*, and a_τ^* the action that maximizes $\{Q^{B_\tau}(a_i)\}_{i=1}^M$. Assume that $\{Q^{B_\tau}(a_i)\}_{i=1}^M$ are independent from $\{Q^{B_1}(a_i)\}_{i=1}^M$. Then*

$$E[Q^{B_1}(a_\tau^*)] \leq E[Q^{B_0}(a_\tau^*)] \leq E[\max_i Q^{B_\tau}(a_i)].$$

Furthermore, there exists a β such that $E[Q^{B_0}(a_\tau^*)] = \max_i E[Q(a_i)]$.

Proof The proof is provided in the Appendix. ■

Notice that, under the assumption that $\{Q^{B_\tau}(a_i)\}_i$ are independent from $\{Q^{B_1}(a_i)\}_i$, by construction $Q^{B_1}(a_\tau^*)$ is a *double estimator* of $\max_i \mathrm{E}[Q(a_i)]$. Lemma 1 then establishes that the bias of $Q^{B_0}(a_\tau^*)$ is always between the positive bias of the single estimator $\max_i Q^{B_\tau}(a_i)$ and the negative bias of the double estimator $Q^{B_1}(a_\tau^*)$. In other words, $Q^{B_0}(a_\tau^*)$ is guaranteed to balance the overestimation of the single estimator and the underestimation of the double estimator. Therefore, the self-correcting estimator can reduce the maximization bias, and even completely remove it if the parameter β is set appropriately.

Let us denote with β^* such an ideal (and in general unknown) parameter for which the self-correcting estimator is unbiased. A value $\beta^* \to 1$ indicates that no bias needs to be removed from $Q^{B_0}(a_\tau^*)$. While as β^* becomes larger, progressively more bias has to be removed. Thus, β^* can be seen as a measure of the severity of the maximization bias, weighting how much bias should be removed. As remarked, it is in practice impossible to know the ideal β^*. But these observations tell us that larger biases will have to be corrected by choosing correspondingly larger values of β.

4 Self-correcting Q-learning

In this section we apply the self-correcting estimator to Q-learning, and propose a novel method to address its maximization bias. We consider sequential updates of the action value function, $Q_n(s', a)$ and $Q_{n+1}(s', a)$, as candidates for the correlated estimators $Q^{B_0}(a_i)$ and $Q^{B_1}(a_i)$ in Lemma 1, despite the relationship between $Q_n(s', a)$ and $Q_{n+1}(s', a)$ in Q-learning being seemingly more complicated than that defined in Eq. (2). Replacing $Q^{B_0}(a_i)$ and $Q^{B_1}(a_i)$ in Eq. (3) with $Q_n(s', a)$ and $Q_{n+1}(s', a)$ gives

$$Q_{n+1}^\beta(s', a) = Q_{n+1}(s', a) - \beta[Q_{n+1}(s', a) - Q_n(s', a)].$$

Let a_τ^* be the action that maximizes $Q_{n+1}^\beta(s', a)$ over a. Following Lemma 1, $Q_n(s', a_\tau^*)$ balances the overestimation of the single estimator and the underestimation of the double estimator, and moreover there exists an optimal value of β for which it is unbiased. However, $Q_{n+1}(s', a)$ is not available at time step n. To address this issue, we construct an alternative Q-function by replacing the sequential updates $Q_n(s', a)$ and $Q_{n+1}(s', a)$ with the sequential updates $Q_{n-1}(s', a)$ and $Q_n(s', a)$ at the previous update step. Specifically, we define the following Q-function:

$$Q_n^\beta(s', a) = Q_n(s', a) - \beta[Q_n(s', a) - Q_{n-1}(s', a)], \quad (4)$$

where $\beta \geq 1$ is a constant parameter tuning the bias correction. Therefore, we propose to use $Q_n^\beta(s', a)$ for action selection according to $\hat{a}_\beta = \arg\max_a Q_n^\beta(s', a)$, and to use $Q_n(s', \hat{a}_\beta)$ to estimate the value of the next step. This results in the following *self-correcting estimator* approximating the maximum expected value: $Q_n(s', \hat{a}_\beta) \approx \max_a \mathrm{E}[Q_n(s', a)]$. Thus, we propose to replace Eqn. (1) from Q-learning with the following novel updating scheme:

$$\begin{aligned} Q_{n+1}(s, a) =& Q_n(s, a) \\ &+ \alpha_n(s, a)\left[r + \gamma Q_n\left(s', \hat{a}_\beta\right) - Q_n(s, a)\right]. \end{aligned} \quad (5)$$

We call this *Self-correcting Q-learning*, and summarize the method in Algorithm 1.

Algorithm 1: Self-correcting Q-learning.

Parameters: step size $\alpha \in (0, 1]$, discount factor $\gamma \in (0, 1]$, small $\epsilon > 0$, and $\beta \geq 1$.
Initialize $Q_0(s, a) = 0$ for all $a \in A$, and s terminal
Loop for each episode:
(1) Initialize s
(2) Loop for each step of episode:
(2.a) Choose a from s using the policy ϵ-greedy in Q
(2.b) Take action a, observe r, s', update $Q_n(s, a)$:

$$Q_n^\beta(s', a) = Q_n(s', a) - \beta[Q_n(s', a) - Q_{n-1}(s', a)]$$
$$\hat{\alpha}_\beta = \arg\max_a Q_n^\beta(s', a)$$
$$\begin{aligned} Q_{n+1}(s, a) =& Q_n(s, a) \\ &+ \alpha_n(s, a)\left[r + \gamma Q_n\left(s', \hat{\alpha}_\beta\right) - Q_n(s, a)\right] \end{aligned}$$
$$s \leftarrow s'$$

(3) until s is terminal.

Remarks. $Q_n^\beta(s', a) = Q_{n-1}(s', a)$ if $\beta = 1$. In this case, the algorithm uses Q_{n-1} from the previous time step instead of Q_n to select the action. This is different from Double Q-learning which trains two Q-functions, but is reminiscent of using a "delayed" target network (Mnih et al. 2015).

We now prove that asymptotically Self-correcting Q-learning converges to the optimal policy. Comparing Eqns. (1) and (5), we see that the difference between Self-correcting Q-learning and regular Q-learning is due to $Q_n^\beta(s', a)$ being different from $Q_n(s', a)$. As the gap between $Q_n(s', a)$ and $Q_{n-1}(s', a)$ becomes smaller, less bias is self-corrected. This suggests that the convergence of Self-correcting Q-learning for $n \to \infty$ can be proven with similar techniques as Q-learning.

We formalize this intuition in a theoretical result that follows the proof ideas used by (Tsitsiklis 1994) and (Jaakkola, Jordan, and Singh 1994) to prove the convergence of Q-learning, which are in turn built on the convergence property of stochastic dynamic approximations. Specifically, Theorem 1 below claims that the convergence of Self-correcting Q-learning holds under the same conditions as the convergence of Q-learning. The proof is in the Appendix.

Theorem 1 *If the following conditions are satisfied: (C1) The MDP is finite, that is, the state and action spaces are finite; (C2)* $\alpha_n(s,a) \in (0,1]$ *is such that* $\sum_{n=1}^{\infty} \alpha_n(s,a) = \infty$ *and* $\sum_{n=1}^{\infty}[\alpha_n(s,a)]^2 < \infty$, $\forall s, a$*; (C3) Var*$(r) < \infty$*; (C4)* $1 \leq \beta < \infty$*; then* $Q_n(s,a)$ *as updated by Self-correcting Q-learning (Algorithm 1), will converge to the optimal value* Q^* *defined by the Bellman optimality given by the* Bellman equations *with probability one.*

We conclude this section by discussing the parameter β in Self-correcting Q-learning. In estimating the maximum expected value, Lemma 1 shows that β relies on the correlation between $Q^{B_0}(a_i)$ and $Q^{B_1}(a_i)$. However, the relation between $Q_n(s',a)$ and $Q_{n-1}(s',a)$ in Q-learning is more complicated than the setting of the Lemma. Therefore, the significance of Lemma 1 lies in the fact that $Q_n(s',a) - Q_{n-1}(s',a)$ can be used as "direction" for removing the maximization bias in the objective to search policy, and that β can be tuned to approximate the premise of the Lemma and match the correlation between the estimators $Q_n(s',a)$ and $Q_{n-1}(s',a)$, corresponding to $Q^{B_0}(a_i)$ and $Q^{B_1}(a_i)$ of the self-correcting estimator.

As we will show empirically in Section 6, the correction of the maximization bias is robust to changes in values of β. As rule of thumb, we recommend setting $\beta \approx 2, 3,$ or 4, keeping in mind that as reward variability increases (which exacerbates the maximization bias), improved performance may be obtained by setting β to larger values.

5 Self-correcting Deep Q-learning

Conveniently, Self-correcting Q-learning is amenable to a Deep Neural Network implementation, similarly to how Double Q-learning can be turned into Double DQN (van Hasselt, Guez, and Silver 2016). This will allow us to apply the idea of the self-correcting estimator in high-dimensional domains, like those that have been recently solved by Deep Q Networks (DQN), the combination of Q-learning with Deep Learning techniques (Mnih et al. 2015). A testbed that has become standard for DQN is the Atari 2600 domain popularized by the ALE Environment (Bellemare et al. 2013), that we'll examine in the Experiments section.

We first quickly review Deep Q Networks (DQN). A DQN is a multi-layer neural network parameterizing an action value function $Q(s,a;\theta)$, where θ are the parameters of the network that are tuned by gradient descent on the Bellman error. An important ingredient for this learning procedure to be stable is the use proposed in (Mnih et al. 2015) of a target network, i.e. a network with parameters θ^- (using the notation of (van Hasselt, Guez, and Silver 2016)) which are a delayed version of the parameters of the online network. Our DQN implementation of Self-correcting Q-learning also makes use of such a target network.

Specifically, our proposed *Self-correcting Deep Q Network algorithm* (ScDQN) equates the current and previous estimates of the action value function $Q_n(s',a)$ and $Q_{n-1}(s',a)$ in Eqn. (4) to the target network $Q(s,a;\theta^-)$ and the online network $Q(s,a;\theta)$, respectively. In other words, we compute

$$Q^{\beta}(s',a) = Q(s',a;\theta^-) - \beta[Q(s',a;\theta^-) - Q(s',a;\theta)],$$

which, analogously to Algorithm 1, is used for action selection: $\hat{a}_\beta = \arg\max_a Q^{\beta}(s',a)$, while the target network $Q(s',a;\theta^-)$ is used for action evaluation as in regular DQN. Everything else also proceeds as in DQN and Double-DQN.

Remarks. It's worth drawing a parallel between the relation of ScDQN with Self-correcting Q-learning, and that of regular DQN with Double-DQN. First, unlike Self-correcting Q-learning which uses $Q_n(s',a) - Q_{n-1}(s',a)$ to correct the maximization bias, ScDQN uses $Q(s',a;\theta^-) - Q(s',a;\theta)$, and therefore takes advantage of the target network introduced in DQN. This is analogous to Double-DQN performing action selection through the target network, instead of using a second independent Q function like vanilla Double Q-learning. If memory requirements weren't an issue, a closer implementation to Self-correcting Q-learning would be to define $Q^{\beta}(s',a) = Q(s',a;\theta^-) - \beta[Q(s',a;\theta^-) - Q(s',a;\theta^=)]$, where $\theta^=$ denotes a set of parameters delayed by a fixed number of steps. This would be an alternative strategy worth investigating. Second, ScDQN is implemented such that with $\beta = 0$ it equals regular DQN, and with $\beta = 1$ it goes to Double-DQN. The intuition that this provides is that ScDQN with $\beta \geq 1$ removes a bias that is estimated in the direction between DQN and Double-DQN, rather then interpolating between the two.

In the Experiments section we benchmark Self-correcting Q-learning on several classical RL tasks, and ScDQN on a representative set of tasks of the Atari 2600 domain.

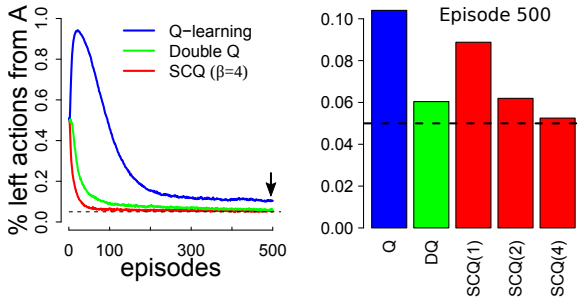

Figure 1: Maximization bias example. Left: percent of left actions taken as a function of episode. Parameter settings are $\epsilon = 0.1$, $\alpha = 0.1$, and $\gamma = 1$. Initial action value estimates are zero. Right: percent of left actions from A, averaged over the last five episodes (arrows) for Q-learning, Double Q-learning and Self-correcting Q-learning with increasing β, which decreases bias. Results averaged over 10,000 runs.

6 Experiments

We compare in simulations the performance of several algorithms: *Q-learning*, *Double Q-learning*, and our *Self-correcting Q-learning* (denoted as *SCQ* in the figures), with $\beta = 1, 2, 4$. Note that Self-correcting Q-learning can be applied to debias any variant of Q-learning, but for simplicity in the empirical studies we only apply our method to Q-learning and focus on the comparison of the resulting self-correcting algorithm with Q-learning and Double

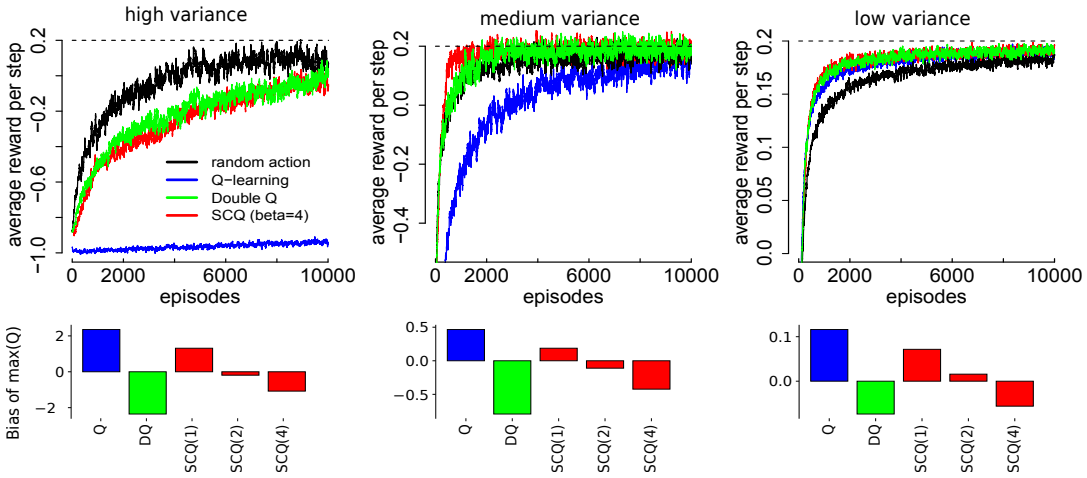

Figure 2: Grid-world task. Rewards of a non-terminating step are uniformly sampled between $(-12, 10)$ (high variance), $(-6, 4)$ (medium variance), and $(-2, 0)$ (low variance). Upper row: average reward per time step. Lower row: bias of the maximal action values of the final episode in the starting state, where SCQ(c), c=1,2,4, denotes SCQ($\beta = c$). Average rewards are accumulated over 500 rounds and averaged over 500 runs.

Q-learning. We consider three simple but representative tasks. First, the maximization bias example shows that Self-correcting Q-learning can remove more bias than Double Q-learning. Second, the grid-world task serves to establish the advantage of Self-correcting Q-learning over Double Q-learning in terms of overall performance, and shows its robustness towards high reward variability. Lastly, the cliff-walking task shows that Self-correcting Q-learning displays faster convergence than Q-learning when rewards are fixed.

Maximization Bias Example. This simple episodic MDP task has two non-terminal states A and B (see details in Figure 6.5 of (Sutton and Barto 2018)). The agent always starts in A with a choice between two actions: left and right. The right action transitions immediately to the terminal state with a reward of zero. The left action transitions to B with a reward of zero. From B, the agent has several possible actions, all of which immediately transition to the termination with a reward drawn from a Gaussian $\mathcal{N}(-0.1, 1)$. As a result, the expected reward for any trajectory starting with left is -0.1 and that for going right is 0. In this settings, the optimal policy is to choose action left 5% from A.

Fig. 1 shows that Q-learning initially learns to take the left action much more often than the right action, and asymptotes to taking it about 5% more often than optimal, a symptom of the maximization bias. Double Q-learning has a smaller bias, but still takes the left action about 1% more often than optimal. Self-correcting Q-learning performs between Double Q-learning and Q-learning when $\beta = 1$, performs similarly to Double Q-learning when $\beta = 2$, and almost completely removes the bias when $\beta = 4$, demonstrating almost complete mitigation of the maximization bias.

Grid-world task. We follow the 3×3 grid-world MDP in (van Hasselt 2010), but study different degrees of reward randomness. The starting state is in the southwest position of the grid-world and the goal state is in the northeast. Each time the agent selects an action that puts it off the grid, the agent stays in the same state. In this task each state has 4 actions, i.e. the 4 directions the agent can go. In each non-terminating step, the agent receives a random reward uniformly sampled from an interval (L, U), which we choose to modulate the degree of randomness. We consider three intervals: $(-12, 10)$ (high variability), $(-6, 4)$ (medium variability), and $(-2, 0)$ (low variability). Note that all 3 settings have the same optimal values. In the goal state any action yields $+5$ and the episode ends. The optimal policy ends one episode after five actions, so that the optimal average reward per step is $+0.2$. Exploration is encouraged with ϵ-greedy action selection with $\epsilon(s) = 1/\sqrt{n(s)}$, where $n(s)$ is the number of times state s has been visited. The learning rate is set to $\alpha(s, a) = 1/n(s, a)$, where $n(s, a)$ is the number of updates of each state-action. For Double Q-learning, both value functions are updated for each state-action.

The upper panels of Fig. 2 show the average reward per time step, while the lower panels show the deviation of the maximal action value from optimal (the bias) after 10,000 episodes. First, Self-correcting Q-learning with $\beta = 2$ gets very close to the optimal value of the best action in the starting state (the value is about 0.36). Self-correcting Q-learning with $\beta = 1$ still displays overestimation which, however, is much smaller than Q-learning. Self-correcting Q-learning with $\beta = 4$ shows a small underestimation which, however, is much smaller than that of Double Q-learning. These observations are consistent for different degrees of reward randomness. This supports the idea that Self-correcting Q-learning can balance the overestimation of Q-learning and the underestimation of Double Q-learning. Second, Self-correcting Q-learning performs as well as Double Q-learning in terms of average rewards per step. Comparing performance under high, medium, and low reward variability, we observe the following. When variability is high,

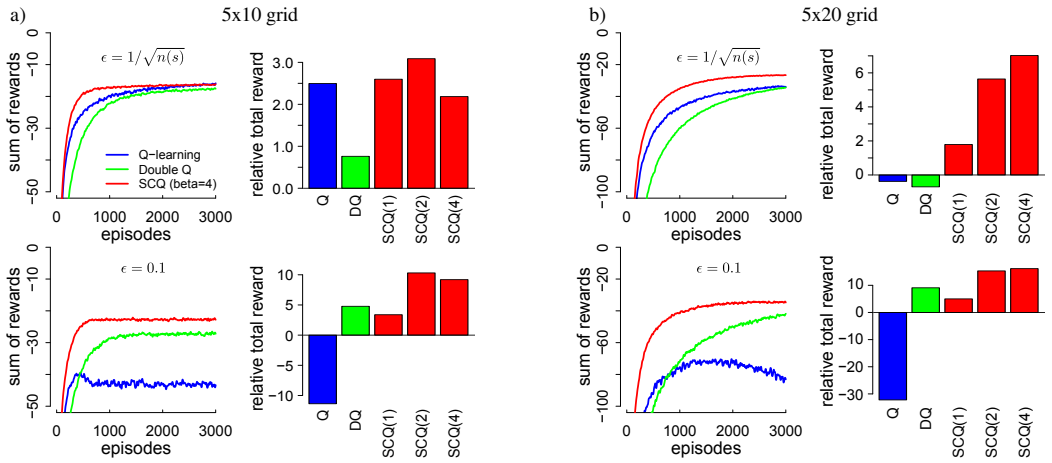

Figure 3: Cliff-walking task. Two environments are considered: a 5×10 arena (a), and a 5×20 arena (b). Left in each panel: cumulative rewards. Right in each panel: relative total reward (i.e. the difference in total reward from random action) at the final episode. SCQ(c), c=1,2,4, denotes SCQ($\beta = c$). Each panel shows two ϵ-greedy exploration strategies: $\epsilon = 1/\sqrt{n(s)}$ (upper panel), and $\epsilon = 0.1$ (lower panel), where $n(s)$ is the number of times state s has been visited. The step sizes are chosen: $\alpha_n(s,a) = 0.1(100+1)/(100+n(s,a))$, where $n(s,a)$ is the number of updates of each state-action. Data points are averaged over 500 runs, then are smoothed for clarity.

Self-correcting Q-learning performs as well as Double Q-learning, and Q-learning the worst. When variability is moderate, Self-correcting Q-learning can performs a little better than Double Q-learning, and Q-learning still performs the worst. When variability is low, that is, the effect of the maximization bias is low, the difference between all methods becomes small. Third, the performance of Self-correcting Q-learning is robust to changes in β. For moderate reward variance, $\beta = 2 - 4$ is a reasonable choice. As reward variability increases, larger β may be better.

Cliff-walking Task. Fig. 3 shows the results on the cliff-walking task Example 6.6 in (Sutton and Barto 2018), a standard undiscounted episodic task with start and goal states, and four movement actions: up, down, right, and left. Reward is -1 on all transitions except those into the "Cliff" region (bottom row except for the start and goal states). If the agent steps into this region, she gets a reward of -100 and is instantly sent back to the start. We vary the environment size by considering a 5×10 and a 5×20 grid.

We measure performance as cumulative reward during episodes, and report average values for 500 runs in Fig. 3. We investigate two ϵ-greedy exploration strategies: $\epsilon = 1/\sqrt{n(s)}$ (annealed) and $\epsilon = 0.1$ (fixed). With rewards of zero variance, Double Q-learning shows no advantage over Q-learning, while Self-correcting Q-learning learns the values of the optimal policy and performs better, with an even larger advantage as the state space increases. This is consistent for both exploration strategies. Conversely, Double Q-learning is much worse than Q-learning when exploration is annealed. This experiments indicate that Double Q-learning may work badly when rewards have zero or low variances. This might be due to the fact that Double Q-learning successively updates two Q-functions in a stochastic way. We also compare the performance of Self-correcting Q-learning under various β values. Self-correcting Q-learning performs stably over different β values, and works well for β between 2 and 4. Finally, larger β results in better performance for environments with larger number of states.

DQN Experiments in the Atari 2600 domain. We study the Self-correcting Deep Q Network algorithm (ScDQN), i.e. the Neural Network version of Self-correcting Q-learning, in five representative tasks of the Atari 2600 domain: *VideoPinball*, *Atlantis*, *DemonAttack*, *Breakout* and *Assault*. These games were chosen because they are the five Atari 2600 games for which Double DQN performes the best compared to human players (van Hasselt, Guez, and Silver 2016). We compare the performance of ScDQN against Double DQN (van Hasselt, Guez, and Silver 2016). We trained the same architecture presented in (Mnih et al. 2015) as implemented in Vel (0.4 candidate version, (Tworek 2018)). Each experiment is run 6 times with different random seeds (as in e.g. (van Hasselt, Guez, and Silver 2016)), and we report average performance and variability across 6 independently trained networks. In all experiments the network explores through ϵ-greedy action selection. The parameter ϵ starts off a 1.0 and is linearly decreased to 0.1 over 1M simulation steps, while β is kept constant throughout.

We observed that our algorithm ScDQN has a faster and more stable convergence to a high reward solution than DQN and double DQN in the tested tasks, as we show in Fig. 4 the representative example of the *VideoPinball* task. Interestingly, ScDQN also tends to display lower value estimates than Double DQN. We hypothesize that this might mainly be due to Double DQN being able to mitigate the underestimation problem of vanilla Double Q-learning thanks to the target network. Fig. 5 shows the final evaluation of DQN net-

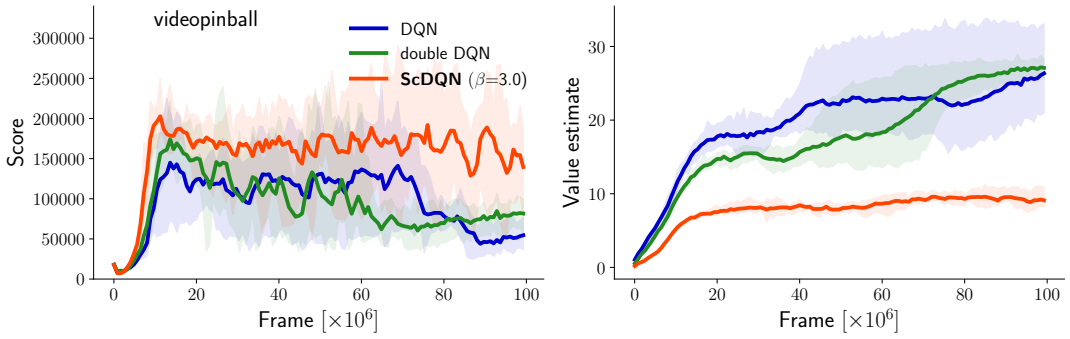

Figure 4: Atari 2600 videopinball game. Results are obtained by running DQN, Double DQN, and ScDQN with 6 different random seeds and hyper-parameters from (Mnih et al. 2015). Lines show average over runs and the shaded areas indicate the min and max values of average performance in intervals of 80000 frames over the 6 runs. Left plot: ScDQN quickly reaches a higher score than DQN and Double DQN. DQN and Double DQN catch up, but their scores are unstable and drop when they excessively overestimate values measured in terms of the estimated value of the selected action (right plot).

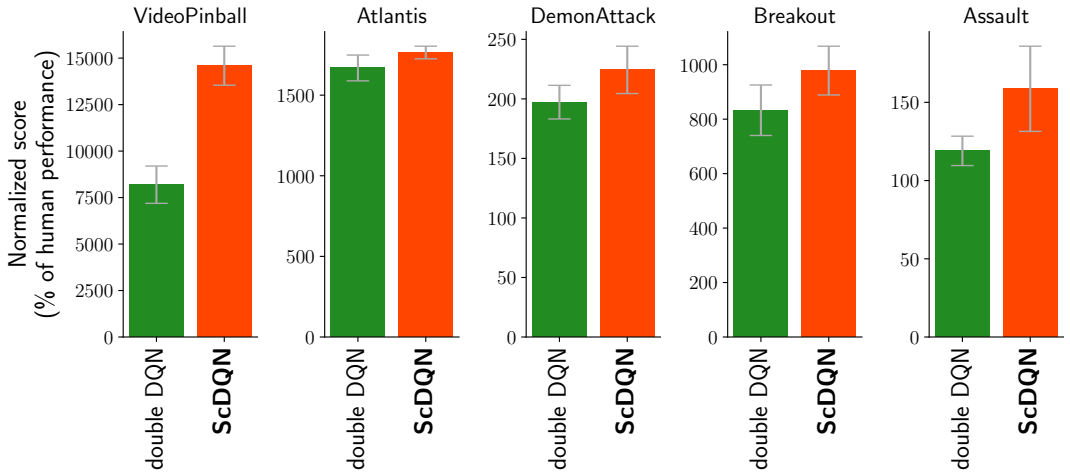

Figure 5: ScDQN rivals Double DQN on Atari 2600 games of DQN networks. Each bar indicates average score over 100 episodes and 6 random seeds (normalized to human performance as in Mnih et al. 2015), error bars are SEM over 6 random seeds. For ScDQN β is set to β=3.0 for all games. Training is done for 20M steps (see Appendix C), unless performance does not seem to stabilize (which was the case for *VideoPinball* and *Breakout*), in which case training is extended to 100M steps.

works trained with Double DQN and ScDQN, and in all of the shown tasks ScDQN is at least as good as Double DQN.

7 Conclusion and Discussion

We have presented a novel algorithm, *Self-correcting Q-learning*, to solve the maximization bias of Q-learning. This method balances the the overestimation of Q-learning and the underestimation of Double Q-learning. We demonstrated theoretical convergence guarantees for Self-correcting Q-learning, and showed that it can scales to large problems and continuous spaces just as Q-learning.

We studied and validated our method on several tasks, including a neural network implementation, ScDQN, which confirm that Self-correcting (deep) Q-learning reaches better performance than Double Q-learning, and converges faster than Q-learning when rewards variability is low.

One question left open is how to optimally set the new parameter β. Luckily, Self-correcting Q-learning does not seem sensitive to the choice of β. In our experiments, $\beta = 2 - 4$ is a good range. Empirically, for larger state spaces and reward variability, larger β tend to work better. Future investigations on the effect of β would still be welcome.

Further interesting future research directions are: (1) formal understanding of the advantage of Self-correcting Q-learning over Q-learning, as for instance in the cliff-walking task with fixed rewards. (2) Besides Q-learning, the maximization bias exists in other reinforcement learning algorithms, such as the actor-critic algorithm. (Fujimoto, van Hoof, and Meger 2018) applied the idea of Double Q-learning to the actor-critic algorithm. Our self-correcting estimator could potentially be applied in a similar way.

Acknowledgements

For this work, RZ was partially supported by the National Natural Science Foundation of China under grants 11871459 and 71532013.

References

Anschel, O.; Baram, N.; and Shimkin, N. 2017. Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning. In *Proceedings of the 34rd International Conference on Machine Learning*.

Azar, M.; R. Munos, R.; Ghavamzadeh, M.; and Kappen, H. 2011. Speedy Q-learning. In *Advances in Neural Information Processing Systems*, 2411–2419.

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research* 47: 253–279.

D'Eramo, C.; Nuara, A.; and Restelli, M. 2016. Estimating the Maximum Expected Value through Gaussian Approximation. In *Proceedings of the 33rd International Conference on Machine Learning*, 1032–1040.

Dorka, N.; Boedecker, J.; and Burgard, W. 2019. Dynamically Balanced Value Estimates for Actor-Critic Methods URL https://openreview.net/forum?id=r1xyayrtDS.

Ernst, D.; Geurts, P.; and Wehenkel, L. 2005. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research* 6(1): 503–556.

Fujimoto, S.; van Hoof, H.; and Meger, D. 2018. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceedings of the 35th International Conference on Machine Learning*.

Jaakkola, T.; Jordan, M.; and Singh, S. 1994. On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation* 6: 1185–1201.

Kim, S.; Asadi, K.; Littman, M.; and Konidaris, G. 2019. Deepmellow: removing the need for a target network in deep Q-learning. In *Proceedings of the Twenty Eighth International Joint Conference on Artificial Intelligence*.

Kober, J.; Bagnell, J.; and Peters, J. 2013. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* 32: 1238–1274.

Lee, D.; Defourny, B.; and Powell, W. 2013. Bias-Corrected Q-Learning to Control Max-Operator Bias in Q-Learning. In *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, 93–99.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.; Veness, J.; Bellemare, M.; Graves, A.; Riedmiller, M.; Fidjeland, A.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature* 518: 529–533.

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

Song, Z.; Parr, R.; and Carin, L. 2019. Revisiting the Softmax Bellman Operator: New Benefits and New Perspective. In *Proceedings of the 36rd International Conference on Machine Learning*.

Strehl, A.; Li, L.; Wiewiora, E.; Langford, J.; and Littman, M. 2006. PAC model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, 881–888.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement Learning: An Introduction*. The MIT Press, second edition edition.

Thaler, R. 1988. Anomalies: The winner's curse. *Journal of Economic Perspectives* 2(1): 191–202.

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

Tsitsiklis, J. 1994. Asynchronous stochastic approximation and Q-learning. *Machine Learning* 16: 185–202.

Tworek, J. 2018. vel (candidate-v0.4 accessed 2020-02-21). https://github.com/MillionIntegrals/vel.

Van den Steen, E. 2004. Rational overoptimism (and other biases). *American Economic Review* 94(4): 1141–1151.

van Hasselt, H. 2010. Double Q-learning. In *Advances in Neural Information Processing Systems*, 2613–2621.

van Hasselt, H.; Guez, A.; and Silver, D. 2016. Deep Reinforcement Learning with Double Q-Learning. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, 2094–2100.

Watkins, C. J. C. H. 1989. *Learning from Delayed Rewards*. Ph.D. thesis, King's College, Cambridge, England.

Watkins, C. J. C. H.; and Dayan, P. 1992. Q-learning. *Machine Learning* 8: 279–292.