Adaptive Distributional Double Q-learning

Maximilian Birr
Institute of Mathematics
University of Mannheim
birr.maximilian@web.de

Mihail Bîrsan
Institute of Mathematics
University of Mannheim
mihailbirsan26@gmail.com

Leif Döring
Institute of Mathematics
University of Mannheim
doering@uni-mannheim.de

Abstract

Bias problems in the estimation of maxima of random variables are a well-known obstacle that drastically slows down Q-learning algorithms. We propose to use additional insight gained from distributional reinforcement learning to deal with the overestimation in a locally adaptive way. This helps to combine the strengths and weaknesses of the different Q-learning variants in a unified framework. Our framework ADDQ is simple to implement, existing RL algorithms can be improved with a few lines of additional code. We provide experimental results in tabular, Atari, and MuJoCo environments for discrete and continuous control problems, comparisons with state-of-the-art methods, and a proof of convergence.

1 Introduction

Watkins' Q-learning is one of the most popular learning algorithms as it has a simple update structure that is straight forward to implement. In each round the agent observes a new reward signal and updates the currently estimated state-action function by combining the new reward signal with the best currently estimated action in the next step. In contrast to some other algorithms the algorithm combines simplicity with mathematical tractability, in essence it's Banach's fixed point iteration for Bellman's optimality operator with estimation errors (see e.g. Bertsekas and Tsitsiklis, 1996). Unfortunately, the update rule involves a maximum and the estimation of expectations of maxima of random variables suffers from overestimation bias. In the context of Q-learning we refer to the seminal papers Thrun and Schwartz, 1993 and van Hasselt, 2010. Even though convergence for tabular cases is proved the convergence can often be seen only after millions of iterations.

Motivated by statistical approaches to the estimation of the expectation of maxima of random variables the concept of double Q-learning was introduced in van Hasselt, 2010. Instead of only using a set of random variables one uses two independent sets. One is used to detect the maximal index, the other set to evaluate the random variable corresponding to the maximal index. In the context of Q-learning this translates to keeping two copies of the Q-matrix that are alternated to either detect the best action and to evaluate the corresponding Q-value. Double Q-learning reduces the overestimation (with function approximation see Figure 2 in Fujimoto et al., 2018) and sometimes even underestimates. A bit of care is needed, double Q-learning is not always superior to Q-learning (or other variants) the additional negative bias can be unfavorable. This can for instance be seen in a simple two-state example (compare Example 6.7 in Sutton and Barto, 2018 and also Lan et al., 2020 for the underestimation effect in the same example) but also in the simple grid world example in van Hasselt, 2010 if the reward distributions are chosen less random as in simulations presented in the paper. In simple words: Q-learning prefers regions with large reward variance, but those must not

17th European Workshop on Reinforcement Learning (EWRL 2024).

be desirable if rewards have small expectations. Thus, depending on parameters in many models either Q-learning or double Q-learning performs weak or strong. Even though overestimation has mostly negative effects it can also have positive effects, for instance to trigger exploration or to help distinguish good and bad actions. For Q-learning with function approximation it is known for Atari games that deep double Q-learning often outperforms Q-learning but sometimes fails badly. The present article is motivated by the desire to combine the strengths of both algorithms.

A number of techniques have been introduced to deal with overestimation bias. Some with even more bias than double Q-learning (e.g. clipped Q-learning from Fujimoto et al., 2018) others with less additional negative bias. Weighted double Q-learning Zhang et al., 2017 used a weighted combination of Q- and double Q-learning estimators. Ensemble Q-learning and averaged Q-learning Anschel et al., 2017 take averages of multiple action values, an approach that also reduces variance. A more recent idea is to use more than only two copies and combine those to a single estimator, see Lan et al., 2020. The number of copies can then be used to find the right amount of over- and underestimation to optimise performance for a given model and to reduce variance. The approach has the drawback that the number of copies must be optimised as an additional hyper-parameter and cannot be adapted during a running training process. Bias-corrected Q-Learning Lee et al., 2013 subtracts a bias-term in order to turn Q-learning into an unbiased stochastic approximation algorithm.

We use distributional reinforcement learning (DRL) to provide a simple framework of algorithms that (i) uses distributional properties to combine for every state-action pair adaptively Q- and double Q-learning updates during training to use their respective advantages, (ii) is easy to implement in existing algorithms, (iii) can be proved to converge in the simplest tabular setting.

To develop a locally (on state-action pair level) adjustable algorithm our approach is to ignore the basic statistical idea behind double Q-learning and purely see such algorithms as stochastic approximation algorithms with additional bias (a bit similar to SARSA compared to Q-learning). We modify the bias such that the bias is adapted to the current estimation situation. It is empirically known that high stochasticity favours double Q while low stochasticity favors Q-learning (see also our Section A). Thus, the variance of estimated Q-values seems a good idea for the interpolation weights. Unfortunately, standard Q-learning only takes track of expectations. This is where DRL enters the algorithm. In DRL one estimates the entire distributions of discounted total rewards. We use DRL to compute the variance (more precisely, the left-truncated variance) and for actions with relatively low (resp. high) variance favor the Q-learning update (resp. double Q-learning update). The figure below gives two tabular examples (details in Appendix A) that highlight the idea of our article. A gridworld (with fake goal and region of high stochasticity) and an extension of Sutton's example. These environments have high and low stochasticity regions, making them potentially complicated for both Q- and double Q-learning. The environments look artificial but similar properties can be

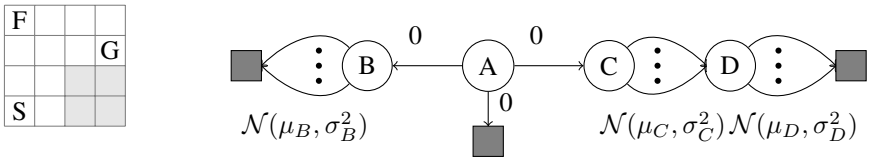


Figure 1: Variants of grid world and an extension of the Sutton/Barto example.

expected in realistic environments. The simulations below (for the Sutton/Barto example) show that adaptive distributional double Q-learning introduced below is more robust, it adapts to the stochasticity by stopping the excitement of Q-learning in high variance regions.

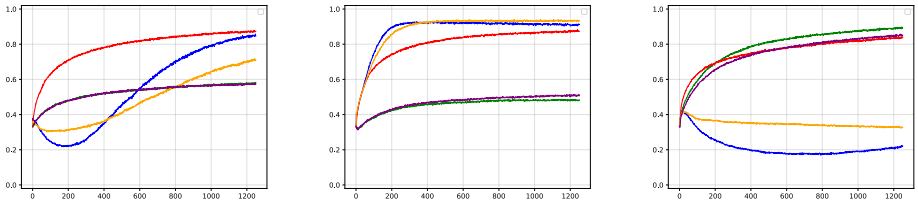


Figure 2: Three mean/variance combinations in the extended Sutton/Barto example. Proportion of optimal action in A for Q- (blue), distributional Q- (yellow), double Q- (green), distributional double Q- (purple), adaptive distributional double Q-learning (red, us). Details are given in Appendix A.

2 Problem setting and heuristics

Let us fix a Markov decision process $(\mathcal{S}, \mathcal{A}, \mathcal{R}, p)$, where $\mathcal{S}$ is the state-space, $\mathcal{A}$ the space of allowed actions, $\mathcal{R}$ is the reward space, and p a transition kernel describing the distribution of the reward r and the new state s' when action a is played in state s. Given a time-stationary policy π, a Markov kernel on $\mathcal{S} \times \mathcal{A}$, there is a Markov reward process (S_t, A_t, R_t) with transitions

$$\mathbb{P}^\pi(R_t = r, S_{t+1} = s', A_{t+1} = a' | S_t = s, A_t = a) = \pi(a' : s') p(r, s' : s, a).$$

The goal of the agent in reinforcement learning is to find a policy that maximises the expected discounted reward $V^\pi(s) = \mathbb{E}^\pi[\sum_{t=0}^\infty \gamma^t R_t | S_0 = s]$ for all starting states s. The discounting factor $\gamma \in (0, 1)$ is fixed. In the discrete setting $\mathcal{S}$ and $\mathcal{A}$ finite it is well-known that optimal stationary policies exist and can be found as greedy policy obtained by the unique solution matrix Q_* to $T^*Q = Q$. The non-linear operator $(T^*Q)(s, a) = r(s, a) + \sum_{s' \in \mathcal{S}} p(s' : s, a) \max_{a' \in \mathcal{A}} Q(s, a)$ is called Bellman's optimality operator. Bellman's optimality operator is a max-norm contraction on the $\mathcal{S} \times \mathcal{A}$ matrices. Using Banach's fixed point theorem the solution can in principle be found by iteratively applying T^* to some initial matrix Q_0. The drawback of this approach is the need to know the operator T^* explicitly, thus, having explicit knowledge on the transitions p. Using standard stochastic approximation algorithms the fixed point Q_* can be approximated by the recursive scheme

$$Q_{n+1}(s, a) = Q_n(s, a) + \alpha_n(s, a)(r + \gamma \max_{a'} Q_n(s', a') - Q_n(s, a)). \tag{1}$$

Here s', r is a one-step sample obtained from $p(\cdot : s, a)$ and the step-sizes are assumed to satisfy the Robbins-Monro conditions. The exploration can be on-policy (using Q_n) or off-policy, the only requirement is infinite visit of all state-action pairs. The algorithm was proved to converge in the tabular setting, see for instance Tsitsiklis, 1994. For some results with function approximation see for instance Melo and Ribeiro, 2007. The problem of Q-learning is the tendency for overestimation, the values $Q_n(s, a)$ will typically be larger than $Q_*(s, a)$. Here is why. In case some $Q_n(s, a)$ is overestimated by a surprisingly large random sample r the maximum $\max_{a'} Q_n(s, a')$ will be overestimated as well and thus spread the overestimation to neighboring state-action pairs. This overestimation will prevail for quite some time, only the discount factor helps to get Q-values down.

Most tricks to avoid overestimation add a bias to the righthand side of (1). This may not be obvious from the heuristic ideas, but can be seen in the convergence proofs (e.g. the sketch of proof in van Hasselt, 2010). Since this is also behind the convergence proof for SARSA (see Singh et al., 2000) we refer to the trick as SARSA trick. Here is the double Q update with SARSA trick:

$$\begin{aligned} Q^A_{t+1}(s, a) &= \overbrace{(1 - \alpha) Q^A_t(s, a) + \alpha\big(r + \gamma Q^A_t(s', a^*)\big)}^{Q\text{-learning}} + \overbrace{\gamma\left(Q^B_t(s', a^*) - Q^A_t(s', a^*)\right)}^{b^A_t}\Big) \\ Q^B_{t+1}(s, a), &= \underbrace{(1 - \alpha) Q^B_t(s, a) + \alpha\big(r + \gamma Q^B_t(s', b^*)\big)}_{Q\text{-learning}} + \underbrace{\gamma\left(Q^A_t(s', b^*) - Q^B_t(s', b^*)\right)}_{b^B_t}\Big), \end{aligned} \tag{2}$$

where $a^* = \arg\max Q^A(s, a)$ and $b^* = \arg\max Q^B(s, a)$. For clipped Q-learning the bias terms are even more negative, the possibility of positive bias is clipped: $b^A_t = \gamma \min\{Q^B_t(s', a^*) - Q^A_t(s', a^*), 0\}$ and similarly for b^B. In our interpretation double Q-learning is nothing but Q-learning with an additional bias term that is typically negative (one compares the matrix Q^B with Q^A

but at a maximal entry of Q^A) and clipped Q-learning subtracts even more. In essence there is no deeper reason behind the particular bias formulas, they are chosen such that the update rule simplifies and is easy to implement. Here is the little trick we add on top. We suggest to study bias terms of the form

$$\underbrace{C^A_{t+1}(s,a)}_{\text{new}}\,\gamma\left(Q^B_t(s',a^*)-Q^A_t(s',a^*)\right)\quad\text{and}\quad\underbrace{C^B_{t+1}(s,a)}_{\text{new}}\,\gamma\left(Q^A_t(s',b^*)-Q^B_t(s',b^*)\right)$$

with some state-action dependent constants $C\in[0,1]$. Choosing C closer to 0 or 1 allows the algorithm to emphasise the update rule of Q- or double Q-learning. Subtracting larger bias emphasises the tendency of double Q-learning to underestimate. We then use DRL to chose C adaptively in a way that they favour Q-learning when Q-learning is favorable and favor double Q-learning when double Q-learning is favorable. In Section 3 we present the basic tabular version that is extended in Section 4 to the distributional DQN- and in Section 5 to the distributional actor-critic framework.

3 Tabular adaptive distributional double Q-learning

For a concise treatment of DRL we refer to the book Bellemare et al., 2023 and only recall the notation needed to formulate our algorithms. Given a Markov decision model and a stationary policy π, Rowland et al., 2018 define the return distribution function as

$$\eta^\pi(s,a)(B):=\mathbb{P}^\pi\Big(\sum_{t=0}^\infty\gamma^tR_t\in B\Big|S_0=s,A_0=a\Big)$$

for $B\in\mathcal{B}(\mathbb{R})$. There have been plenty of theoretical articles on DRL [Bellemare et al., 2017; Dabney et al., 2018; Rowland et al., 2018; Lyle et al., 2019; Bellemare et al., 2023; Rowland et al., 2023; Rowland et al., 2023] establishing distributional Bellman operators, contractivity, convergence proofs of dynamic programming and temporal difference algorithms and projection operators. It was shown [Bellemare et al., 2017; Bellemare et al., 2023] that the return distribution function is the unique solution to $\eta^\pi=\mathcal{T}^\pi\eta^\pi$, where $\mathcal{T}^\pi:\mathcal{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}}\to\mathcal{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}}$ is the distributional Bellman operator defined as $(\mathcal{T}^\pi\eta)(s,a)=\sum_{r,s',a'\in\mathcal{R}\times\mathcal{S}\times\mathcal{A}}b_{r,\gamma}\#\eta(s',a')p(s',r;s,a)\pi(a';s')$ with bootstrap function $b_{r,\gamma}(z)=r+\gamma z$ and push-forward of measures $f\#\nu(B):=\nu(f^{-1}(B))$. In order to work algorithmically with DRL parametrisations $\mathcal{F}$ of measures need to be used. Distributional Q-learning proceeds similarly to classical expectation Q-learning. For a tuple (s,a,r,s') compute a one-sample approximation of the distributional Bellman optimality operator and update the old estimate of the distribution η behind the expectation $Q(s,a)$. To ensure that the procedure stays in the parametrised family $\mathcal{F}$ it is crucial that projections of measures into $\mathcal{F}$ can be computed explicitly. There are two simple parametrisations that have been used successfully. Categorical (fixing a number of atoms with variable weights at fixed locations) and quantile (fixing a number of atoms with fixed weights but variable locations). For the categorical algorithm suppose a set of m evenly spaced locations $\theta_1<\cdots<\theta_m$ is fixed and the categorical measures are defined by

$$\mathcal{F}_{C,m}=\Big\{\sum_{i=1}^m p_i\delta_{\theta_i}\,\Big|\,p_i\geq0,\sum_{i=1}^m p_i=1\Big\}.$$

As argued in Rowland et al. 2018 in the case of $\mathcal{F}_{C,m}$ the projection works as follows. The mass is distributed to the nearest atoms with mass proportional to the distance. For a Dirac measure δ_y

$$\Pi_C(\delta_y)=\begin{cases}\delta_{\theta_1} & :y\leq\theta_1\\ \frac{\theta_{i+1}-y}{\theta_{i+1}-\theta_i}\delta_{\theta_i}+\frac{y-\theta_i}{\theta_{i+1}-\theta_i}\delta_{\theta_{i+1}} & :y\in(\theta_i,\theta_{i+1}]\\ \delta_{\theta_m} & :y>\delta_m\end{cases}$$

and for a mixture linearly extended by $\Pi_C(\sum_{k=1}^m p_k\delta_{y_k})=\sum_{k=1}^m p_k\Pi_C(\delta_{y_k})$.

Let us now turn towards the algorithms, first in the categorical setup followed by the quantile setup. Our crucial ingredient to the algorithm is the mixing with β which depends crucially on the choice that can depend on time and state-action pair. For the moment we keep β arbitrary, a concrete choice is fixed at the end of this section. Extending arguments from the literature, notably the convergence proof for categorical Q-learning of Rowland et al., 2018 and the SARSA trick of Singh et al., 2000 used in van Hasselt, 2010 to sketch a proof of convergence of double Q-learning, we prove almost sure convergence of Algorithm 1.

Algorithm 1 ADDQ: tabular categorical setting

Require: $\eta_t^A(s,a) = \sum_{k=1}^m p_k^A(s,a)\delta_{\theta_k}$, $\eta_t^B(s,a) = \sum_{k=1}^m p_k^B(s,a)\delta_{\theta_k}$ for each (s,a), state-action pair (s_t, a_t) to be updated

Determine $\alpha_t(s_t, a_t)$ and sample transition (s_t, a_t, R_t, S_{t+1})

Randomly choose UPDATE(A) or UPDATE(B)

if UPDATE(A) **then**

$a^* \leftarrow \arg\max_a \mathbb{E}_{Z\sim\eta_t^A(S_{t+1},a)}[Z]$

Determine $\beta_{t+1}^A(s_t, a_t) \in [0,1]$

▷ *Compute mixture* ◁

$\nu \leftarrow \beta_{t+1}^A(s_t,a_t)\eta_t^A(S_{t+1},a^*) + (1-\beta_{t+1}^A(s_t,a_t))\eta_t^B(S_{t+1},a^*)$

$\hat{\eta}_* \leftarrow b_{R_t,\gamma}\#\nu$

▷ *Project target back onto support* ◁

$\hat{\eta} \leftarrow \Pi_C(\hat{\eta}_*)$

$\eta_{t+1}^A(s_t,a_t) = (1-\alpha_t(s_t,a_t))\eta_t^A(s_t,a_t) + \alpha_t(s_t,a_t)\hat{\eta}$

$\eta_{t+1}^A(s,a) = \eta_t^A(s,a)$ for all $(s,a) \neq (s_t,a_t)$

return η_{t+1}^A

else if UPDATE(B) **then**

Proceed analogously with A and B exchanged

return η_{t+1}^B

Theorem 1. *Given some initial return distribution functions η_0^A, η_0^B supported within $[\theta_1, \theta_m]$, the induced Q-values, i.e. the expected values of the return distributions $(\eta_t^A), (\eta_t^B)$, recursively defined by Algorithm 1 converge almost surely towards Q^* if the following conditions are satisfied:*

(i) the step sizes $\alpha_t(s,a)$ almost surely fulfill the Robbins-Monro conditions.

(ii) rewards are bounded in $[R_{min}, R_{max}]$ and $[\frac{R_{min}}{1-\gamma}, \frac{R_{max}}{1-\gamma}] \subseteq [\theta_1, \theta_m]$,

(iii) the choice of updating η^A or η^B is random and independent of all previous random variables

(iv) $(\beta_t^A)_{t\in\mathbb{N}}, (\beta_t^B)_{t\in\mathbb{N}}$ only depend on the past and fulfill $\lim_{t\to\infty} |\beta_t^A - \beta_t^B| = 0$ almost surely.

If additionally the MDP has a unique optimal policy π^*, then $(\eta_t^A), (\eta_t^B)$ converge almost surely in $\bar{\ell}_2$ to some limit $\eta_C^* \in \mathcal{F}_{C,m}$ and the greedy policy with respect to η_C^* is the optimal policy.

The algorithm is a double version of categorical Q-learning with a modified adaptive update rule. As usually a state-action pair (s_t, a_t) is chosen by some exploration mechanism (essentially arbitrarily) from which the MDP dynamics are used to sample the reward R_t and the next state S_{t+1} (we use capital letter for random variables that must be sampled). As for double Q-learning it is decided randomly to update copy A or B. The position a^* (resp. b^*) only requires information on the expected action value while the distributional update uses the entire action value distributions η. For the update we use the double Q-update in a distributional sense. To compute the bias terms we compute the mixture distribution of the action value distributions behind the expectations $Q^A(S_{t+1}, a^*)$ and $Q^B(S_{t+1}, a^*)$. The mixture distribution is discounted by γ and shifted by the reward sample R_t (push-forward). This distribution is still discrete with atoms different from the fixed $\theta_1, ..., \theta_m$. This is fixed by projecting back to $\mathcal{F}_{C,m}$. Finally, see (2) again, the bias terms are mixed with the distribution behind the expectation $Q^A(s_t, a_t)$. Since both have atoms $\theta_1, ..., \theta_m$ no further projection is needed.

The categorical approach has multiple disadvantages, most notably rewards and chosen atoms must be compatible. The algorithm is included because a rigorous convergence proof can be given, the structure is easier to catch, and the implementation is a bit simpler. We now turn to the more interesting quantile setup of Dabney et al., 2018. The difference to the categorical setup is the parametric class

$$\mathcal{F}_{Q,m} = \Big\{ \sum_{i=1}^m \frac{1}{m}\delta_{\theta_i} : \theta_i \in \mathbb{R} \Big\}$$

that does not have fixed atoms (but fixed weights). The update step is a gradient step in computing the Wasserstein-projection on $\mathcal{F}_{Q,m}$ of the target distribution $\hat{\eta}$ (see algorithm below), that is a gradient

step in the quantile Huber-loss minimisation:

$$\min_{\hat{\theta}_1^A(s,a),\dots,\hat{\theta}_m^A(s,a)} \sum_{i=1}^{m} \mathbb{E}_{Z\sim\hat{\eta}}[\rho_{\tau_i}^{\kappa}(Z - \hat{\theta}_i^A(s,a))],$$

with quantile mid-points $\tau_i = \frac{2i-1}{2m}$ and

$$\rho_\tau^\kappa(u) = \begin{cases} |\tau - \mathbf{1}_{u<0}|\frac{1}{2}u^2 & : |u| \le \kappa \\ |\tau - \mathbf{1}_{u<0}|\kappa(|u| - \frac{1}{2}\kappa) & : |u| > \kappa \end{cases}.$$

Algorithm 2 gives pseudocode for the quantile variant of Algorithm 1. In addition to its already

Algorithm 2 ADDQ: tabular quantile setup

Require: $\eta_t^A(s,a) = \sum_{k=1}^m \frac{1}{m}\delta_{\theta_k^A(s,a)}$, $\eta_t^B(s,a) = \sum_{k=1}^m \frac{1}{m}\delta_{\theta_k^B(s,a)}$ for each (s,a), state-action pair (s_t,a_t) to be updated, parameter $\kappa \ge 0$

Determine $\alpha_t(s_t,a_t)$ and sample transition (s_t,a_t,R_t,S_{t+1})

Randomly choose UPDATE(A) or UPDATE(B)

if UPDATE(A) **then**

$a^* \leftarrow \arg\max_a \mathbb{E}_{Z\sim\eta_t^A(S_{t+1},a)}[Z]$

Determine $\beta_{t+1}^A(s_t,a_t) \in [0,1]$

▷ *Compute mixture* ◁

$\nu \leftarrow \beta_{t+1}^A(s_t,a_t)\eta_t^A(S_{t+1},a^*) + (1-\beta_{t+1}^A(s_t,a_t))\eta_t^B(S_{t+1},a^*)$

$\hat{\eta} \leftarrow b_{R_t,\gamma}\#\nu$ and writing $\hat{\eta} =: \sum_{j=1}^m \frac{1}{m}\delta_{\theta_j^{\text{target}}}$

▷ *Gradient step in order to minimise quantile Huber loss* ◁

$\hat{\theta}_i^A = \theta_i^A(s_t,a_t) - \alpha_t(s_t,a_t)\nabla_{\theta_i^A}\frac{1}{m}\sum_{j=1}^m \rho_{\tau_i}^\kappa(\theta_j^{\text{target}} - \theta_i^A(s_t,a_t)) \quad i = 1,\dots,m$

$\eta_{t+1}^A(s_t,a_t) = \sum_{i=1}^m \frac{1}{m}\delta_{\hat{\theta}_i^A}$

$\eta_{t+1}^A(s,a) = \eta_t^A(s,a)$ for all $(s,a) \ne (s_t,a_t)$

return η_{t+1}^A

else if UPDATE(B) **then**

Proceed analogously with A and B exchanged

return η_{t+1}^B

strong state-of-the-art performance, distributional RL enables us to choose the adaptive parameter $\beta_t^{A/B}(s,a)$ [1] based on the entire return distributions to leverage concepts such as left-truncated variance or measuring the divergence of the double estimator using the Wasserstein distance as introduced later on.

Choice of adaptive β: We will use adaptive rates $\beta_{t+1}(s,a)$ closer to 1 if $\eta_t(s,a)$, the distribution behind $Q_t(s,a)$, has small variance. In that case the update at (s,a) is closer to Q- than double Q-learning. In fact, we will be slightly more accurate. Large negative deviations do not influence the maximum, only positive deviations matter. Thus, we decided to use so-called left-truncated variances (LTV) the variance restricted to the right side of the median. LTV has already been used in DRL before, see for instance Mavrin et al. 2019. For a distribution $\nu \in \mathcal{F}_{Q,M}$ the definition is

$$\text{LTV}(\nu) = \frac{1}{\lceil \frac{m}{2} \rceil} \sum_{i=\lceil \frac{m}{2} \rceil}^{m} (\theta_{\lceil \frac{m}{2} \rceil} - \theta_i)^2,$$

the variance over the upper half of atoms. For categorical distributions $\nu =\in \mathcal{F}_{C,m}$ we define

$$\text{LTV}(\nu) = \sum_{i=M}^{m} p_i(\theta_M - \theta_i)^2,$$

where $M \in \{1,\dots,m\}$ is the smallest index fulfilling $\sum_{i=1}^M p_i \ge 0.5$. Here is the choice of adaptive β used for our tabular experiments:

[1] We use A/B to indicate that either A or B can be chosen.

1. Compute left truncated variances of $\eta_t^{A/B}(s,a)$ and denote them by $\text{ltv}_s^{A/B}(a)$.
2. Compute the averages $\text{ltv}_s(a) = \frac{1}{2}\big(\text{ltv}_s^A(a) + \text{ltv}_s^B(a)\big)$ and $\text{ltv}_s = \frac{1}{|\mathcal{A}|}\sum_{a\in\mathcal{A}} \text{ltv}_s(a)$.
3. Compute $\hat{\beta}_{t+1}(s,a) := \frac{\text{ltv}_s(a)}{\text{ltv}_s}$, values close to 1 correspond to relatively large variance.
4. Define

$$\beta_{t+1}^B(s,a) := \beta_{t+1}^A(s,a) := \begin{cases} 0.75 & : \hat{\beta}_{t+1}(s,a) < 0.75 \\ 0.5 & : \hat{\beta}_{t+1}(s,a) \in [0.75, 1.25] \\ 0.25 & : \hat{\beta}_{t+1}(s,a) > 1.25 \end{cases}. \tag{3}$$

The choice of β is rather simple minded, we tried not to over-engineer the choice and used this simple choice for all our simulations. The case study in the appendix shows that this choice of β performs well compared to more/less aggressive choices and constant mixtures of Q- and double Q-learning.

4 Adaptive double C51/QRDQN

We extend our ideas to RL with function approximation and modify quantile regression DQN (QRDQN) and C51 by using two copies with adaptive local bias control. The main contribution are LTV_C51, LTV_QRDQN, and WS_QRDQN. Implementation details are given in Appendix B.

Adaptive double C51 Algorithm: The C51 algorithm obtained its name from using a categorical representation with $m = 51$ atoms. Discounted return distributions $\eta(s,a)$ are parameterized via feedforward neural networks following the DQN architecture [Mnih et al., 2015]. The state s serves as input and the last linear layer outputs $m = 51$ logits for each action followed by a softmax.

Following Bellemare et al., 2023, Section 10.2, we write $\eta_\omega(s,a) = \sum_{i=1}^m p_i(s,a;\omega)\delta_{\theta_i}$ where ω comprises the online networks weights. The corresponding expectation is denoted by $Q_\omega(s,a) = \sum_{i=1}^m p_i(s,a;\omega)\theta_i$. Given a realized transition (s,a,r,s') from the replay buffer the target becomes

$$\bar{\eta}(s,a) = \Pi_C(b_{r,\gamma}\#\eta_{\bar{\omega}}(s',a^*)) =: \sum_{i=1}^m \bar{p}_i\delta_{\theta_i},$$

where $a^* = \arg\max_{a'} Q_{\bar{\omega}}(s',a')$. Here, $\bar{\omega}$ denotes the parameters of a separate target network which is kept constant and overwritten every e.g. 10000 steps with the parameters from the online network. The loss is calculated as cross-entropy loss $-\sum_{i=1}^m \bar{p}_i \log p_i(s,a;\omega)$ based on which the network parameters are updated using gradient descent over mini-batches. In the experiments we consider variants where we keep track of two independently initialized online networks denoted by ω^A, ω^B and a pair of respective target networks. For each gradient step we simulate a vector of random variables with the same size as the batch size with each element determining which of the two estimators is being updated based on the respective transition with the same position in the batch. Accordingly, we use twice the batch size for these methods, so that on average per gradient step, the same number of transitions is used for each estimator, compared to the single-estimator case. Given a transition (s,a,r,s') we consider different variants for the simple temporal difference targets

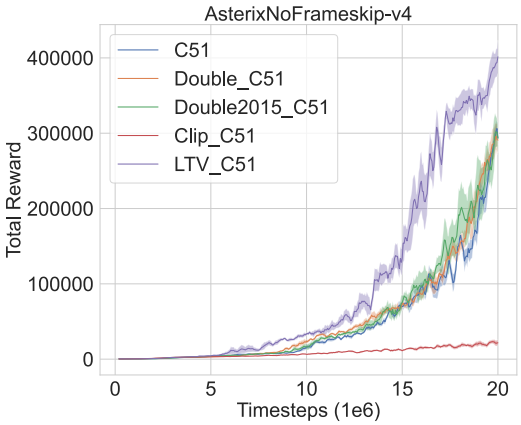


Figure 3: Ex. Asterix, more in Appendix B.1

$$\bar{\eta}(s,a) = \Pi_C(b_{r,\gamma}\#\Gamma)$$

in the following combinations. If A/B is updated then only Γ is replaced in the following way:

(i) Pure double estimator: $\Gamma^{A/B} = \eta_{\bar{\omega}^{B/A}}(s',z^*)$, where $z^* = \arg\max_{a'} Q_{\bar{\omega}^{A/B}}(s',a')$.

(ii) Double estimator based on target network, inspired by van Hasselt et al., 2015: The batch size is the same as in the single estimator case, there are no extra networks. The choice corresponds to

$$\Gamma = \eta_{\bar{\omega}}(s', z^*), \quad z^* = \arg\max_{a'} Q_{\omega}(s', a').$$

Greedy action selection is with respect to the online network, i.e. the target network acts as B, the online network as A.

(iii) Clipped estimator, inspired by Fujimoto et al., 2018[2]: Set $\Gamma^{A/B} = \eta_{\bar{\omega}^X}(s', z^*)$, where $z^* = \arg\max_{a'} Q_{\bar{\omega}^{A/B}}(s', a')$ and $X = \arg\min_{c\in\{A,B\}} Q_{\bar{\omega}^c}(s', z^*)$.

(iv) Left-truncated variance (LTV) adaptive double target: Motivated by our tabular algorithm we define the locally adaptive mixture of Q- and double Q-update:

$$\Gamma^{A/B} = \beta(s, a; \omega)\eta_{\bar{\omega}^{A/B}}(s', z^*) + (1 - \beta(s, a; \omega))\eta_{\bar{\omega}^{B/A}}(s', z^*),$$

where $z^* = \arg\max_{a'} Q_{\bar{\omega}^{A/B}}(s', a')$. The rule to determine $\beta(s, a; \omega)$ is analogue to the previous section (Equation (3)) and uses the online network's distributions $\eta_{\omega^A}, \eta_{\omega^B}$.

Experimental results in Appendix B.1 show that our LTV-based adaptive algorithm performs well throughout five standard Atari environments while performance of other algorithms varies over different environments. This relates closely to our observations in tabular control (Appendix A).

Adaptive double QRDQN: We next turn towards adaptive double QRDQN. Using the same network architecture as C51, QRDQN approximates return distributions using the quantile representation. Therefore the last layer outputs the m quantile locations for each action. In the quantile setup we write $\eta_\omega(s, a) = \frac{1}{m}\sum_{i=1}^m \delta_{\theta_i(s,a;\omega)}$ with induced mean values $Q_\omega(s, a) = \frac{1}{m}\sum_{i=1}^m \theta_i(s, a; \omega)$. Given a sample transition (s, a, r, s'), the network parameters are updated via gradient descent with respect to the loss function

$$\mathcal{L}(\omega) = \frac{1}{m}\sum_{i,j=1}^{m} \rho^1_{\tau_i}(r + \gamma\theta_j(s', z^*; \bar{\omega}) - \theta_i(s, a; \omega)), \quad z^* = \arg\max_{a'} Q_{\bar{\omega}}(s', a'), \tag{4}$$

and the quantile mid-points $\tau_i = \frac{2i-1}{2m}$. For the double estimator variants the target atoms $\Gamma = \{r + \gamma\theta_j(s', z^*; \bar{\omega}) : j = 1, \ldots, m\}$, which are used to calculate the loss, are replaced as follows:

(i) Double, Clipped, LTV adaptive: analogously to the previous section.

(ii) Wasserstein reduced mixture (Wasserstein QRDQN): To show robustness of our adaptive bias control approach we introduce a second (more indirect) way for adaptive bias control. We need this alternative for continuous control in Section 5 as β from Equation (3) is not well-defined for infinitely many actions. In order to control the overestimation bias, based on the current estimates, we use the 1-Wasserstein w_1 distance of both estimates from the mixture distribution. A large discrepancy between $\eta_{\bar{\omega}^A}$ and $\eta_{\bar{\omega}^B}$ is a sign that one estimate has been overestimated. Since in DRL we have access to the full return distribution, this allows us to measure the estimator's divergence directly based on the Wasserstein distance, a metric for probability distributions. The more aligned both distributions, the smaller the reduction. We replace

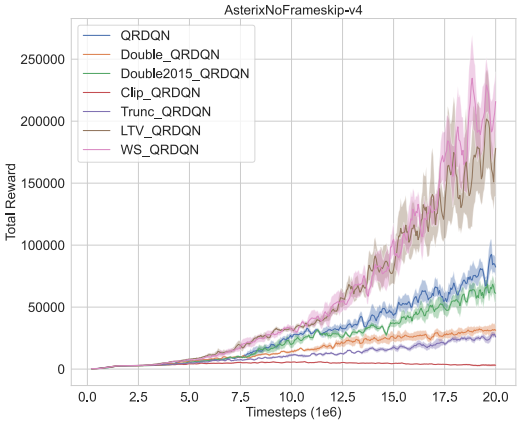


Figure 4: Ex. Asterix, more in Appendix B.2

$$\Gamma = \{r + \gamma(\theta_j(s', z^*; \bar{\omega}^X) - \beta w_1(\eta_{\bar{\omega}^A}(s', z^*), \eta_{\bar{\omega}^B}(s', z*))) : j = 1, \ldots, m,\ X = A, B\},$$

where $z^* = \arg\max_{a'}(Q_{\bar{\omega}^A}(s', a') + Q_{\bar{\omega}^B}(s', a'))$ and accordingly the $\frac{1}{m}$ in (4) is replaced by $\frac{1}{2m}$. Since $w_1(\nu, \nu') \geq |\mathbb{E}_{Z\sim\nu}[Z] - \mathbb{E}_{Z\sim\nu'}[Z]|$ factors fulfilling $\beta \in (0, \frac{1}{2})$ are reasonable. In the Atari experiments we fix $\beta = 0.1$, an experiment with varying β is provided in Appendix B.3.

[2]TD3 [Fujimoto et al., 2018] introduced clipping in an actor-critic setting, where the action is given by the actor. In our C51 adaptation we select the greedy action based on the target network that is updated.

(iii) Truncated QRDQN, inspired by Kuznetsov et al., 2020: Drop top k atoms of the mixture:

$$\Gamma^{A/B} = \{r + \gamma\hat{\theta}_j(s', z^*) : j = 1, \dots, (2m-k)\}, \quad \text{where}$$
$$\hat{\theta}_j(s', a') = \text{sort}(\{\theta_i(s, a; \bar{\omega}^X) : i = 1, \dots, m,\ X \in \{A, B\}\})_j$$

and $z^* = \arg\max_{a'} Q_{\bar{\omega}^{A/B}}(s', a')$ and accordingly the $\frac{1}{m}$ in (4) is replaced by $\frac{1}{2m-k}$.

Experimental results in Appendix B.2 show that both our algorithms work similarly well and outperform the other examples with comparable algorithm architecture. This suggests that not the exact form of bias control matters, but mostly the local adaptivity. On these environments sample efficiency and final performance are improved. Let us compare the experiments with the experiments in tabular control (Appendix A). If algorithms subtract non-adaptive biases then there performance depends on the stochasticity of the environment. This can be seen for double, even more for clipped, but also for truncated mixture. Truncated mixture is actually adaptive but shifts distributions downward even if both copies are identical. The way we control bias represses bias control if copies are equal.

5 Adaptive double QR-SAC

TD3 [Fujimoto et al., 2018] proposed to take the minimum over two independently initialized critics (clipping) in the temporal difference target which is also applied in SAC [Haarnoja et al., 2017; Haarnoja et al., 2018; Haarnoja et al., 2018]. REDQ [Chen et al., 2021] minimizes over a randomly selected set of critics (in-target minimization) and Realistic Actor-Critic [Li et al., 2023] uses a 'punished Bellman update' in combination with universal value function approximation (UVFA) in order to mitigate under-/overestimation bias. TQC [Kuznetsov et al., 2020] is a state of the art algorithm that successfully enhances SAC by distributional RL, as the critic is a quantile network trained with the quantile regression loss. A crucial aspect of TQC is its precise overestimation control, achieved by truncating the topmost atoms of a mixture of multiple quantile networks in the TD target. This method significantly increased performance on the MuJoCo benchmark.

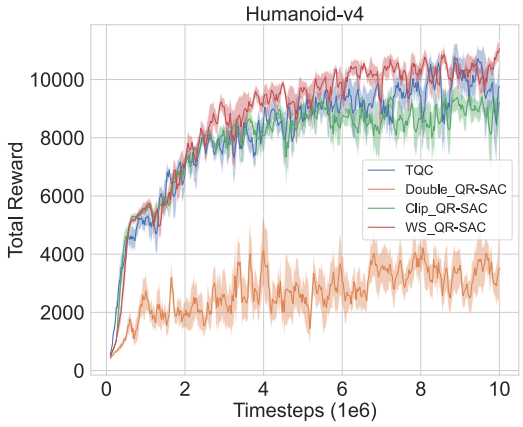


Figure 5: Ex. Humanoid, more in Appendix B.3

Wasserstein Reduced QR-SAC: We build on TQC's framework combining quantile networks with SAC, but replace the TQC target with our Wasserstein reduced mixture target. Additionally, unlike TQC, which updates every critic with every transition drawn from the replay buffer, we randomly select one of the two critics for each transition. This approach prevents a too quick alignment given the identical targets, ensuring that the differences between the two networks arise from more than just their initializations. In particular, given a transition (s, a, r, s'), the target atoms, based on which we calculate the quantile Huber loss, are

$$\bar{\theta}_j = r + \gamma\big(\frac{1}{2}(\theta_j(s', a'; \bar{\omega}^A) + \theta_j(s', a'; \bar{\omega}^B)) - \beta w_1(\eta_{\bar{\omega}^A}(s', a'), \eta_{\bar{\omega}^B}(s', a')) - \alpha E\big)$$

for $j = 1, \dots, m$, where $a' \sim \pi(\cdot; s', \phi)$ is an action given by the current actor network and the deduction of $E = \log(\pi(a'; s', \phi))$ comes from the maximum entropy regularisation applied in SAC.

Experiments on MuJoCo environments, Figure 14 of the appendix, show a comparison of TQC[3]. in comparison with a clipped (resp. double) target and our Wasserstein reduced mixture target. As expected double can perform well/badly. Adding bias control by clipping improves QR-SAC almost to the level of TQC, our Wasserstein target algorithm (red) even a bit more. It should be mentioned

[3]In the original paper TQC uses $N = 5$ critics. For an equal comparison, we use 2 critics for every algorithm.

that we did not over-engineer the choice of β, we used $\beta = 0.3$ for all environments besides Hopper ($\beta = 0.5$). This is reasonable, for Hopper also TQC switches to drop 5 instead of 2 atoms per critic.

6 Summary and limitations

In this article we introduce ADDQ, a simple way to profit at the same time from the estimation biases of Q- and double Q-learning. This is achieved by subtracting state-action dependent biases that take into account variances of current estimates. Even though no detailed tuning was performed our modifications greatly improve the base algorithms C51, QRDQN, and SAC with quantile networks. There are a number of further developments that should be considered in future work. Our simple choices of β works surprisingly well, more adaptivity (in particular in the actor-critic setting) should further improve the algorithms.

7 Acknowledgement

The authors acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant INST 35/1597-1 FUGG.

References

Anschel, Oron, Nir Baram, and Nahum Shimkin (June 2017). "Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 176–185. URL: https://proceedings.mlr.press/v70/anschel17a.html.

Bellemare, Marc G., Will Dabney, and Rémi Munos (2017). "A Distributional Perspective on Reinforcement Learning". In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML'17. Sydney, NSW, Australia: JMLR.org, pp. 449–458.

Bellemare, Marc G., Will Dabney, and Mark Rowland (2023). *Distributional Reinforcement Learning*. http://www.distributional-rl.org. MIT Press.

Bellemare, Marc G., Yavar Naddaf, et al. (May 2013). "The arcade learning environment: an evaluation platform for general agents". In: *J. Artif. Int. Res.* 47.1, pp. 253–279. ISSN: 1076-9757.

Bertsekas, Dimitri P. and John N. Tsitsiklis (1996). *Neuro-dynamic programming*. Vol. 3. Optimization and neural computation series. Athena Scientific, pp. I–XIII, 1–491. ISBN: 1886529108.

Castro, Pablo Samuel et al. (2018). "Dopamine: A Research Framework for Deep Reinforcement Learning". In: URL: http://arxiv.org/abs/1812.06110.

Chen, Xinyue et al. (2021). "Randomized ensembled double q-learning: Learning fast without a model". In: *arXiv preprint arXiv:2101.05982*.

Dabney, Will et al. (2018). "Distributional reinforcement learning with quantile regression". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI'18/IAAI'18/EAAI'18. New Orleans, Louisiana, USA: AAAI Press. ISBN: 978-1-57735-800-8.

Fujimoto, Scott, Herke van Hoof, and David Meger (2018). "Addressing function approximation error in actor-critic methods". In: *arXiv preprint arXiv:1802.09477*.

Haarnoja, Tuomas, Haoran Tang, et al. (2017). "Reinforcement learning with deep energy-based policies". In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML'17. Sydney, NSW, Australia: JMLR.org, pp. 1352–1361.

Haarnoja, Tuomas, Aurick Zhou, Pieter Abbeel, et al. (2018). "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor". In: *International conference on machine learning*. PMLR, pp. 1861–1870.

Haarnoja, Tuomas, Aurick Zhou, Kristian Hartikainen, et al. (2018). "Soft actor-critic algorithms and applications". In: *arXiv preprint arXiv:1812.05905*.

Kuznetsov, Arsenii et al. (2020). "Controlling overestimation bias with truncated mixture of continuous distributional quantile critics". In: *Proceedings of the 37th International Conference on Machine Learning*. ICML'20. JMLR.org.

Lan, Qingfeng et al. (2020). "Maxmin Q-learning: Controlling the Estimation Bias of Q-learning". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. URL: https://openreview.net/forum?id=Bkg0u3Etwr.

Lee, Donghun, Boris Defourny, and Warren B. Powell (2013). "Bias-corrected Q-learning to control max-operator bias in Q-learning". In: *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pp. 93–99. DOI: 10.1109/ADPRL.2013.6614994.

Li, Sicen et al. (2023). "Realistic Actor-Critic: A framework for balance between value overestimation and underestimation". In: *Frontiers in Neurorobotics* 16. ISSN: 1662-5218. DOI: 10.3389/fnbot.2022.1081242. URL: https://www.frontiersin.org/articles/10.3389/fnbot.2022.1081242.

Lyle, Clare, Marc G. Bellemare, and Pablo Samuel Castro (July 2019). "A Comparative Analysis of Expected and Distributional Reinforcement Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01, pp. 4504–4511. DOI: 10.1609/aaai.v33i01.33014504. URL: https://ojs.aaai.org/index.php/AAAI/article/view/4365.

Mavrin, Borislav et al. (2019). "Distributional Reinforcement Learning for Efficient Exploration". In: *Proceedings of the 36th International Conference on Machine Learning*. Vol. 97. ICML'19. ,Long Beach, California, USA: JMLR.org, pp. 4424–4434.

Melo, Francisco S. and M.I. Ribeiro (2007). "Q-learning with linear function approximation". In: *Proc. Int. Conf. Computational Learning Theory*, pp. 308–322.

Mnih, Volodymyr et al. (2015). "Human-level control through deep reinforcement learning". In: *Nature* 518.7540, pp. 529–533. DOI: 10.1038/nature14236. URL: https://doi.org/10.1038/nature14236.

Quan, John and Georg Ostrovski (2020). *DQN Zoo: Reference implementations of DQN-based agents*. Version 1.2.0. URL: http://github.com/deepmind/dqn_zoo.

Raffin, Antonin (2020). *RL Baselines3 Zoo*. https://github.com/DLR-RM/rl-baselines3-zoo.

Raffin, Antonin et al. (2021). "Stable-Baselines3: Reliable Reinforcement Learning Implementations". In: *Journal of Machine Learning Research* 22.268, pp. 1–8. URL: http://jmlr.org/papers/v22/20-1364.html.

Rowland, Mark, Marc G. Bellemare, et al. (Sept. 2018). "An Analysis of Categorical Distributional Reinforcement Learning". In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Ed. by Amos Storkey and Fernando Perez-Cruz. Vol. 84. Proceedings of Machine Learning Research. PMLR, pp. 29–37. URL: https://proceedings.mlr.press/v84/rowland18a.html.

Rowland, Mark, Rémi Munos, et al. (2023). "An analysis of quantile temporal-difference learning". In: *arXiv preprint arXiv:2301.04462*.

Rowland, Mark, Yunhao Tang, et al. (2023). "The statistical benefits of quantile temporal-difference learning for value estimation". In: *Proceedings of the 40th International Conference on Machine Learning*. ICML'23. , Honolulu, Hawaii, USA, JMLR.org.

Singh, Satinder, Tommi Jaakkola, Michael Littman, et al. (2000). "Convergence Results for Single-Step On-Policy Reinforcement-Learning Algorithms". In: *Machine Learning* 38.3, pp. 287–308. DOI: 10.1023/A:1007678930559.

Singh, Satinder, Tommi Jaakkola, Michael L. Littman, et al. (2000). "Convergence Results for Single-Step On-Policy Reinforcement-Learning Algorithms". In: *Machine Learning* 38.3, pp. 287–308. DOI: 10.1023/A:1007678930559. URL: https://doi.org/10.1023/A:1007678930559.

Sutton, Richard S. and Andrew G. Barto (2018). *Reinforcement Learning: An Introduction*. Second. The MIT Press. URL: http://incompleteideas.net/book/the-book-2nd.html.

Thrun, Sebastian and Anton Schwartz (1993). "Issues in Using Function Approximation for Reinforcement Learning". In: *Proceedings of the 1993 Connectionist Models Summer School*. Ed. by Michael Mozer et al. Lawrence Erlbaum, pp. 255–263. URL: http://www.ri.cmu.edu/pub_files/pub1/thrun_sebastian_1993_1/thrun_sebastian_1993_1.pdf.

Todorov, Emanuel, Tom Erez, and Yuval Tassa (2012). "MuJoCo: A physics engine for model-based control". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. DOI: 10.1109/IROS.2012.6386109.

Towers, Mark et al. (Mar. 2023). *Gymnasium*. DOI: 10.5281/zenodo.8127026. URL: https://zenodo.org/record/8127025 (visited on 07/08/2023).

Tsitsiklis, John N. (1994). "Asynchronous Stochastic Approximation and Q-Learning". In: *Machine Learning* 16.3, pp. 185–202. DOI: 10.1023/A:1022689125041. URL: https://doi.org/10.1023/A:1022689125041.

van Hasselt, Hado (2010). "Double Q-learning". In: *Advances in Neural Information Processing Systems*. Ed. by J. Lafferty et al. Vol. 23. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2010/file/091d584fced301b442654dd8c23b3fc9-Paper.pdf.

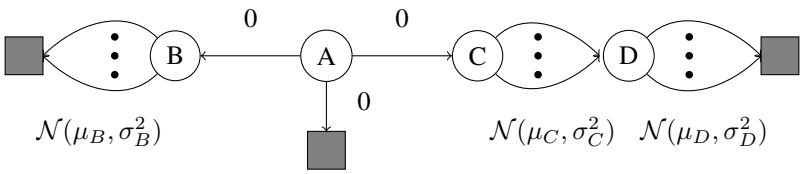
van Hasselt, Hado, Arthur Guez, and David Silver (2015). "Deep Reinforcement Learning with Double Q-learning". In: cite arxiv:1509.06461Comment: AAAI 2016. URL: http://arxiv.org/abs/1509.06461.

Zhang, Zongzhang, Zhiyuan Pan, and Mykel J. Kochenderfer (2017). "Weighted Double Q-learning". In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 3455–3461. DOI: 10.24963/ijcai.2017/483. URL: https://doi.org/10.24963/ijcai.2017/483.

A Tabular reinforcement learning experiments

A.1 Extended Sutton/Barto example: Experiments

In this section we have a closer look at the extended Sutton/Barto example from the introduction.



The example is inspired by Example 6.7 in Sutton and Barto, 2018 that has been studied further in Lan et al., 2020. Our extended example is more complex as it includes more states and more effects that are supposed to mimic (in a very simplified setting) effects from more realistic environments. There are four non-terminating states (A, B, C, D) and three terminating states, the boxes. In non-terminating states A, C, and D several actions can be taken that all lead to the same next state. Rewards for all transitions from A are 0, all other rewards are Gaussian, with equal laws for all actions (arrows) leaving the same state. We denote by N_B, N_C and N_D the number of possible actions in the respective states. The goal is to learn the optimal action (left/right/down) in state A.

In what follows we provide a simulation study of our tabular algorithms from Section 4. We start with six plots for the quantile version of adaptive distributional double Q-learning and provide at the end of this section the same plots for the categorical setting.

The figure below shows a comparison of Q-, double Q-, and our (quantile) adaptive tabular algorithm. The choices of parameters are chosen such that they highlight difficulties for Q- and/or double Q-learning. We plot optimal decisions in A over 10,000 runs. We discount with $\gamma = 0.9$ and perform ϵ-greedy exploration with $\epsilon = 0.1$

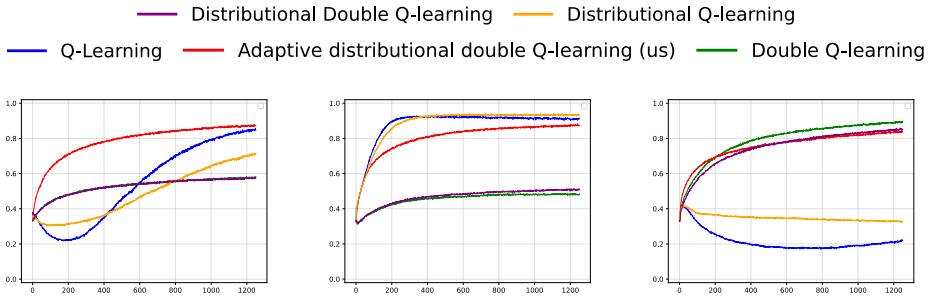


Figure 6: Three extreme scenarios.

The choices of parameters used in the three simulations are as follows:

	N_B, μ_B, σ_B	N_C, μ_C, σ_C	N_D, μ_D, σ_D
left	300, -0.3, 1	5, 0.15, 0.5	5, 0.15, 0.5
middle	300, -0.3, 1	50, 0.1, 1	50, 0.1, 1
right	300, -0.75, 10	5, 1.25, 1.25	5, 1.25, 1.25

The parameter choices are extreme in order to present clear visual statements. All plots clearly show the danger of Q-learning. The agent tends to prefer regions (here states) with high variance as the maximum in the update forces to follow large reward samples. If coincidentally these states are also good (i.e. have large expectation) Q-learning profits. Otherwise Q-learning fails. Double Q-learning is somewhat opposite. The agent prefers uniformity over actions which is favored by small variances. If such regions have large expected rewards this behavior is desirable, otherwise not. In the regime on the right the left state has relatively high variance (compared to the alternatives), but small expectation. This makes the overestimation of Q-learning harmful, while double Q rightfully prefers the state with smaller variance (and larger expectation). In the middle, the variances are equal, but having two stochastic states on the right side, leads Q-learning to overestimate the (coincidentally) optimal state. The effects are mixed on the left, favouring an interpolation between the two. What can be seen is that the adaptive algorithm performs well in all situations. In the cases where one of Q- or double Q

is much better, our algorithm adapts β so as to be similar to this algorithm. However, this is not all. In the case when choosing over- and underestimation everywhere is harmful, our state-action wise adaptive choice can be superior as can be seen in the left plot. To gain a better feeling for properties of adapted distributional double Q-learning we provide simulations with different atoms and different β. The three sets of parameters (left, middle, right) from above are kept fixed.

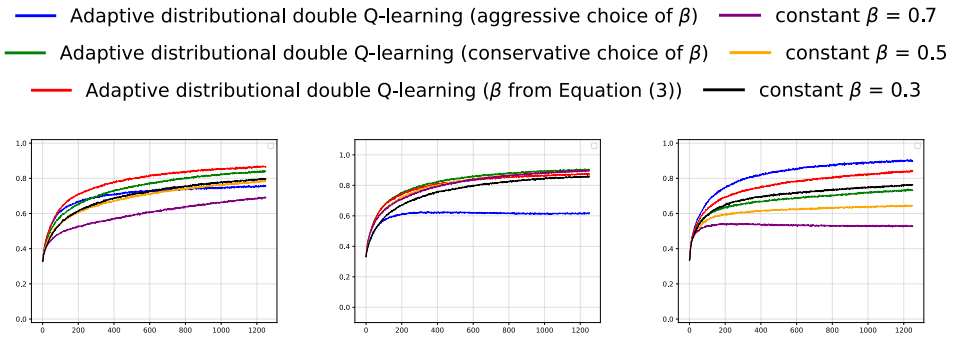


Figure 7: Different choices of adaptive and constant β.

The choice of adaptive β is crucial for our approach. While constant β corresponds to a fixed mixture of Q- and double Q-learning the main idea of this article is to chose β that locally depend on state-action pairs. Adaptive β can be 'aggressive' or 'conservative', depending on how strongly they prefer the Q- or double Q-learning update. In the figure above we study the following choices:

- For aggressive β we choose β to be 0 or 1 if $\hat{\beta}_t(s,a)$ is outside the interval [0.99,1.01], yielding more extreme switching between the two variants.
- For conservative β, we choose β to be 0.4 or 0.6 if $\hat{\beta}_t(s,a)$ is outside the interval [0.6,1.4], yielding a smoother switching.

It turns out that the choice of β from Equation (3) performs quite robust over different type of environments.

Finally, we provide the same simulations for our categorical variant of adaptive distributional double Q-learning. We again compare Q- and double Q- with (categorical) adaptive distributional double Q-learning and then only (categorical) adaptive distributional double Q-learning for the same choices of β as above. The simulations confirm the observations for the quantile algorithm. The algorithm is quite robust over different environments and choices of β:

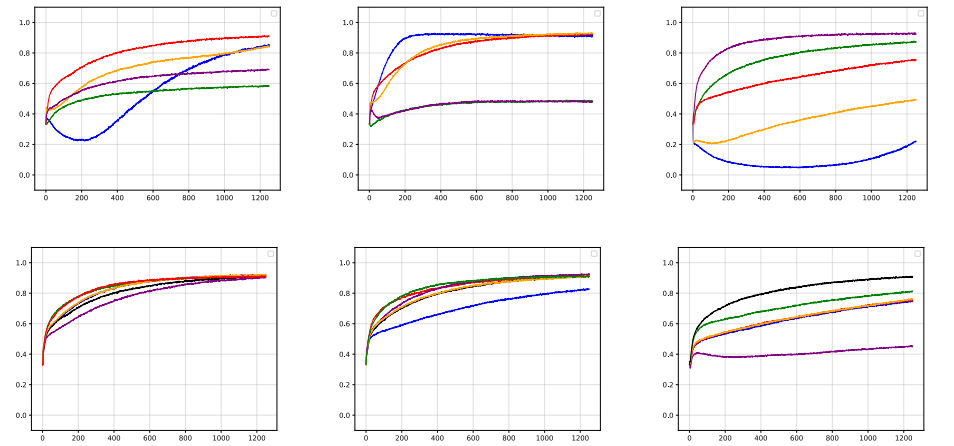


Figure 8: Analogous plots (same colours) for the categorical variant of our algorithm

A.2 Gridworld Experiments

In this section we provide a second tabular example. The example is quite different from the extended Sutton/Barto example but shares common features. The example is motivated by the example in the appendix of van Hasselt, 2010. In van Hasselt, 2010 it was shown that double Q-learning outperforms Q-learning on gridworld where ordinary states get Bernoulli rewards either $+10$ or -12. With less variance or in the typical situation of rewards -1 the opposite is the case and Q-learning outperforms double Q-learning. Our example shows that one can also create simple gridworld examples on which both Q- and double Q-learning do not perform well.

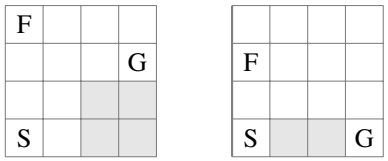


Figure 9: Gridworld 1 and Gridworld 2

We created two variants of gridworld to highlight the influence of (local) stochasticity on Q-learning algorithms. The agent starts in state S and each episode ends in the terminal states F and G. The parameters are chosen so as to make G preferable to F, F is a fake goal. Similar to the example from van Hasselt, 2010 the goal is to maximise the average reward per step, as each non-terminating step has negative mean value. The light gray area represents a more stochastic region of the grid world. The rewards in F and G are deterministic, denoted by R_F and R_G. The other states yield Bernoulli rewards, either e_1 or e_2 for the normal states and s_1 or s_2 for the stochastic states. Double Q underestimates the stochastic region and is more tempted to chose the terminal state F. In contrast, a highly variable stochastic region may cause Q-learning to spend more time in it (Q-learning believes the gray region is better than it really is) and thus find the state G later. Depending on the choice of parameters these effects may dominate or be irrelevant.

As in the previous section, different choices of parameters can cause Q-/double Q-learning or both to struggle. Three such scenarios are summarised in the plots below, comparing Q-, double Q-, and our adaptive tabular algorithms. Once again, extreme choices of parameters are chosen to create examples that highlight the difficulties for Q- and double Q-learning and the strengths of our adaptive algorithm.

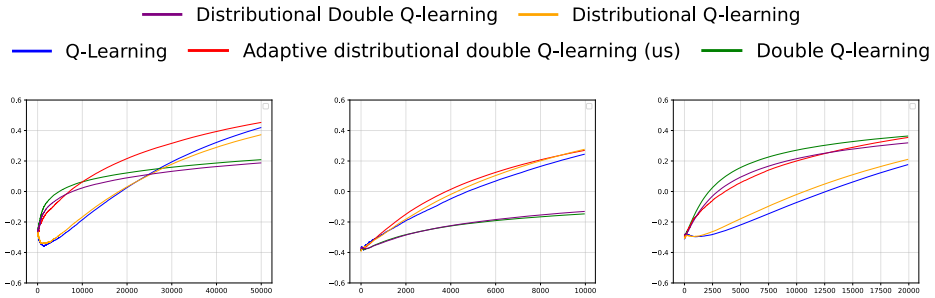


Figure 10: Three extreme scenarios

The plots have been created for Gridworld 1 and two parameter configurations of Gridworld 2:

	Gridworld	R_G	R_F	small randomness	high randomness
left	1	8.5	1.5	$e_1 = -0.5, e_2 = -0.5$	$s_1 = -11, s_2 = 10$
middle	2	6	-0.5	$e_1 = -2, e_2 = 1$	$s_1 = -9, s_2 = 8$
right	2	8.5	3	$e_1 = -2, e_2 = 1$	$s_1 = -10, s_2 = 9$

What can be seen is that the adaptive algorithm performs well in all situations, adapting to the strengths of Q- and double Q, but also works when both algorithms struggle. This is because the adaptive algorithm locally adapts to over- and underestimation according to the variance.

The final plots show the effect of different choices of β from the previous section. It turns out that the choice from Equation (3) is robust for the three scenarios.

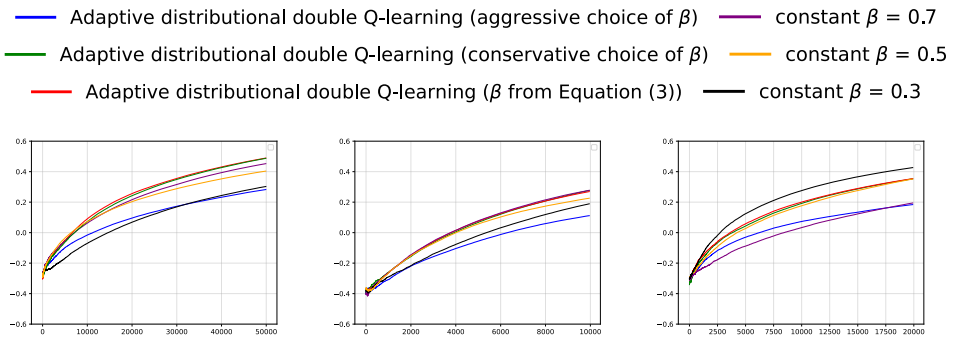


Figure 11: Different choices of our algorithm.

B Deep reinforcement learning experiments

To ensure fair comparison we modified the algorithms C51 [Bellemare et al., 2017] and QRDQN [Dabney et al., 2018] within the Stable-Baselines3 framework [Raffin et al., 2021]. The C51 implementation has been added to this framework by adapting from the Dopamine framework [Castro et al., 2018] and the DQN Zoo [Quan and Ostrovski, 2020]. We run Atari environments from the Arcade Learning Environment [Bellemare et al., 2013] and MuJoCo [Todorov et al., 2012] environments both using the Gymnasium API [Towers et al., 2023]. We run the experiments via the RL Baselines3 Zoo [Raffin, 2020] training framework.

The code to the implementations of this section has been uploaded as a zip file and will be provided on GitHub for the final version.

The experiments were executed on a HPC cluster with NVIDIA Tesla V100 and NVIDIA A100 GPUs. The replay buffer on Atari environments takes around 57GB of memory and less than 7 GB of memory for MuJoCo environments.

For the experiments the training has been interrupted every 50000 (Atari) / 25000 (MuJoCo) steps and 10 evaluation episodes without exploration have been performed. The plots below show the mean total reward (sum of all rewards) averaged over 3 seeds with standard errors of seeds as the shaded regions. To improve visibility a rolling window of size 4 is applied. Atari runs took less than 48 hours for 20 million train steps and periodic evaluations and MuJoCo runs less than 36 for 10 million train steps (Humanoid) and periodic evaluations. Note that one timestep in the Atari environments corresponds to 4 frames, which are stacked together. This corresponds to repeating every action 4 times in the actual game. Therefore 20 million timesteps correspond to 80 million frames.

B.1 Adaptive double C51 Experiments

As in Bellemare et al., 2017 we use 51 atoms for all C51 variants. All other hyperparameters are identical to those given in Table B.2.

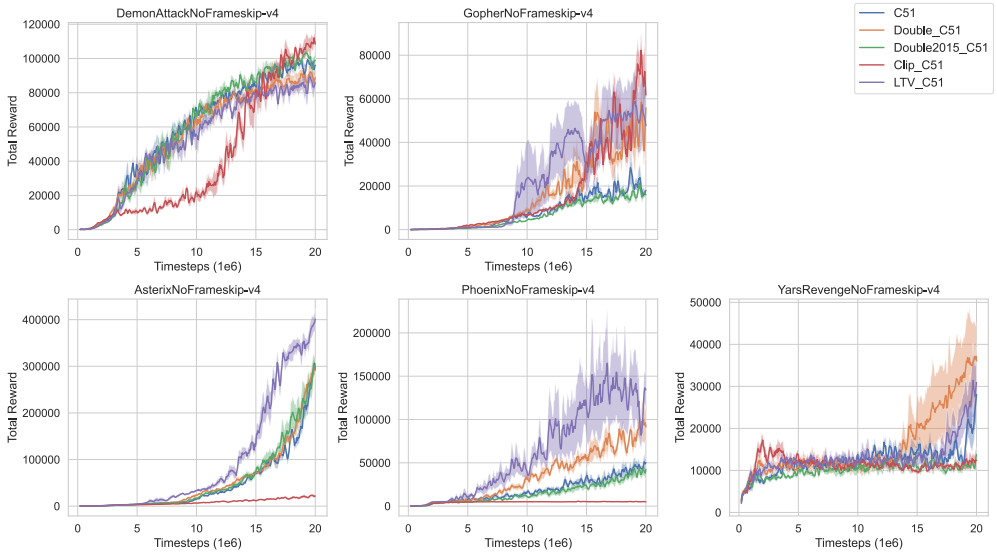


Figure 12: Experiments on Atari environments, using all categorical algorithms from Section 4

Environments	C51	Double_C51	Double2015_C51	Clip_C51	LTV_C51
DemonAttackNoFrameskip-v4	99682 +/- 224	94961 +/- 4895	99858 +/- 3540	102885 +/- 5005	85230 +/- 2276
GopherNoFrameskip-v4	18241 +/- 1063	52631 +/- 9204	15505 +/- 899	50211 +/- 8318	43346 +/- 19994
AsterixNoFrameskip-v4	263093 +/- 11803	270030 +/- 8655	281073 +/- 16837	21000 +/- 3798	407403 +/- 28542
PhoenixNoFrameskip-v4	63568 +/- 4855	99317 +/- 6721	53728 +/- 12261	4985 +/- 87	133240 +/- 14603
YarsRevengeNoFrameskip-v4	29010 +/- 8041	33757 +/- 10066	12834 +/- 1493	10942 +/- 1761	32091 +/- 7524

Table 1: Comparison of final performance on Atari environments. Values show the average over 10 evaluation epsiodes and 3 seeds with standard errors over the seeds.

B.2 Adaptive double QRDQN Experiments

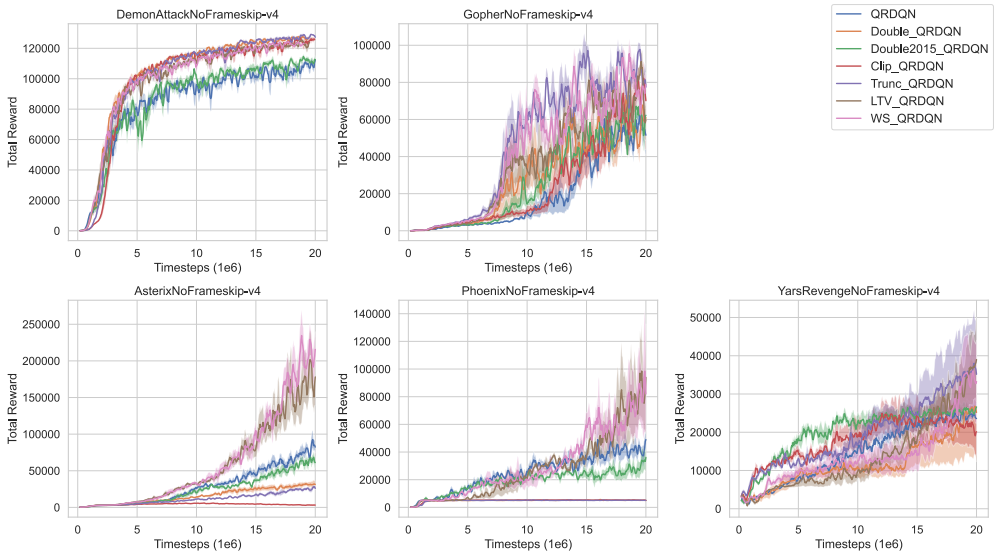


Figure 13: Experiments on Atari environments, using all quantile algorithms from Section 4

Environments	QRDQN	Double_QRDQN	Double2015_QRDQN	Clip_QRDQN	Trunc_QRDQN	LTV_QRDQN	WS_QRDQN
DemonAttackNoFrameskip-v4	115706 +/- 2053	129010 +/- 682	111057 +/- 4307	124011 +/- 1598	125708 +/- 575	125166 +/- 262	124247 +/- 1455
GopherNoFrameskip-v4	52219 +/- 5009	63463 +/- 23046	61715 +/- 9728	73197 +/- 10218	67561 +/- 6148	58726 +/- 16318	83505 +/- 4696
AsterixNoFrameskip-v4	88013 +/- 18430	31050 +/- 2286	61955 +/- 6473	2973 +/- 742	20543 +/- 1253	195613 +/- 53041	221328 +/- 29495
PhoenixNoFrameskip-v4	44527 +/- 8129	5091 +/- 121	44206 +/- 14257	5163 +/- 313	5234 +/- 85	96424 +/- 43412	63868 +/- 13198
YarsRevengeNoFrameskip-v4	24123 +/- 1497	24240 +/- 13716	25024 +/- 1507	21242 +/- 5147	33759 +/- 10867	39304 +/- 11551	34827 +/- 11271

Table 2: Comparison of final performance on Atari environments. Values show the average over 10 evaluation epsiodes and 3 seeds with standard errors over the seeds.

HYPERPARAMETER	QRDQN	Double2015_QRDQN	Double_QR-SAC	Clip_QRDQN	Trunk_QRDQN	LTV_QRDQN	WS_QRDQN
OPTIMIZER	ADAM						
LEARNING RATE	$5 \cdot 10^{-5}$						
DISCOUNT FACTOR γ	0.99						
REPLAY BUFFER SIZE	$1 \cdot 10^{6}$						
BASE ARCHITECTURE	DQN						
NUMBER OF QUANTILE NETWORKS	1				2		
MINIBATCH SIZE	32				64		
TARGET UPDATE INTERVAL	10000						
GRADIENT STEPS PER TRAIN ITERATION	1						
ENVIRONMENT STEPS PER ITERATION	1						
LEARNING STARTS	50000						
EXPLORATION FRACTION	0.025						
EXPLORATION FINAL EPSILON	0.01						
NUMBER OF ATOMS m	200						
HUBER LOSS PARAMETER κ	1						
β	-						0.1
NUMBER OF DROPPED ATOMS PER CRITIC	-				15		-

Table 3: Hyperparameters for QRDQN like algorithms for Atari environments. Hyperparameters for C51 are identical besides the number of atoms m is 51

B.3 Wasserstein Reduced QR-SAC

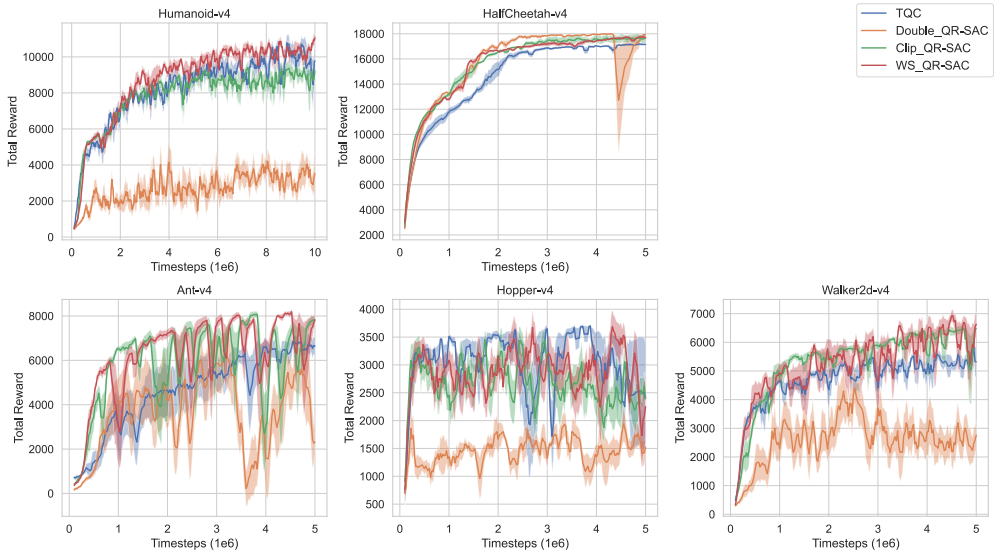


Figure 14: Experiments on MuJoCo environments, using all algorithms from Section 4

Environments	TQC	Double_QR-SAC	Clip_QR-SAC	WS_QR-SAC
Humanoid-v4	10210 +/- 263	3897 +/- 954	9362 +/- 483	11105 +/- 261
HalfCheetah-v4	17147 +/- 86	17837 +/- 60	17619 +/- 366	17730 +/- 107
Ant-v4	6295 +/- 329	3726 +/- 347	7651 +/- 187	7953 +/- 252
Hopper-v4	2185 +/- 909	1659 +/- 217	2030 +/- 14	2538 +/- 700
Walker2d-v4	5796 +/- 264	3139 +/- 187	6446 +/- 121	6838 +/- 109

Table 4: Comparison of final performance on MuJoCo environments. Values show the average over 10 evaluation epsiodes and 3 seeds with standard errors over the seeds.

HYPERPARAMETER	TQC	Double_QR-SAC	Clip_QR-SAC	WS_QR-SAC
OPTIMIZER	ADAM			
LEARNING RATE	$3 \cdot 10^{-4}$			
DISCOUNT FACTOR γ	0.99			
REPLAY BUFFER SIZE	$1 \cdot 10^{6}$			
NUMBER OF CRITICS N	2			
NUMBER OF HIDDEN LAYERS IN CRITIC NETWORKS	3			
SIZE OF HIDDEN LAYERS IN CRITIC NETWORKS	512			
NUMBER OF HIDDEN LAYERS IN POLICY NETWORK	2			
SIZE OF HIDDEN LAYERS IN POLICY NETWORK	256			
MINIBATCH SIZE	256	512		
ENTROPY TARGET $\mathcal{H}_T$	$-\dim \mathcal{A}$			
ACTIVATION FUNCTION	ReLU			
TARGET SMOOTHING COEFFICIENT	0.005			
TARGET UPDATE INTERVAL	1			
GRADIENT STEPS PER ITERATION	1			
ENVIRONMENT STEPS PER ITERATION	1			
LEARNING STARTS	10000			
NUMBER OF ATOMS m	25			
HUBER LOSS PARAMETER κ	1			

Table 5: Hyperparameters for continuous control algorithms

ENVIRONMENT	TQC: number of dropped atoms per critic	WS_QR-SAC: β
Hopper	5	0.5
HALFCHEETAH	2	0.3
WALKER2D	2	0.3
ANT	2	0.3
HUMANOID	2	0.3

Table 6: Environment dependent hyperparameters for TQC and WS$_Q R - SAC$

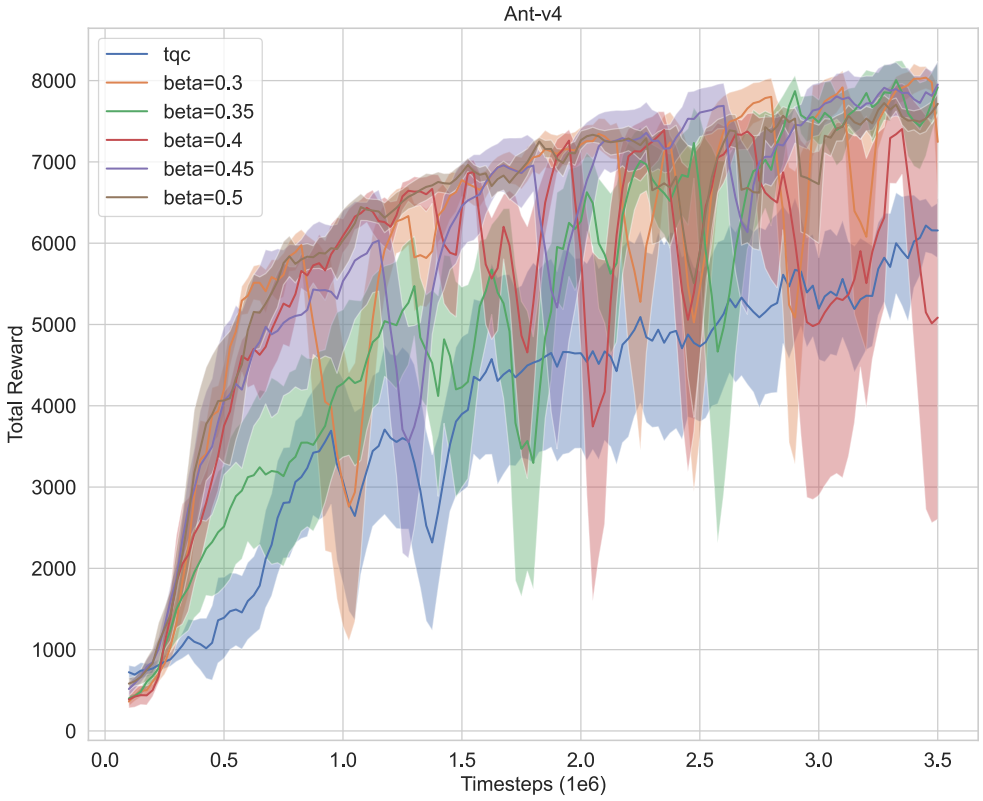


Figure 15: Experiments with Wasserstein Reduced QR-SAC on Ant with varying values β. The results show that the algorithm performs well across different reduction factors β, in particular in comparison to TQC.

C Convergence proof in the tabular categorical setup

In this section we give a convergence proof for the adaptive distributional double-Q algorithm in the simplest setting, the categorical setting. The proof is based on known arguments from the literature and requires some modifications to work in our generality. Since many papers only sketched proofs we decided to spell out all details.

Remark 2 (Notation and short recap). The Cramér distance ℓ_2 for probability distributions $\nu, \nu' \in \mathcal{P}(\mathbb{R})$ is given by

$$\ell_2(\nu, \nu') = \left(\int_{\mathbb{R}} |F_\nu(z) - F_{\nu'}(z)|^2 \, dz \right)^{1/2}.$$

Following Rowland et al., 2018; Bellemare et al., 2023 the supremum extension of a probability metric d between two return distribution functions $\eta, \eta' \in \mathcal{P}^{\mathcal{S}\times\mathcal{A}}$ is denoted as

$$\bar{d}(\eta, \eta') = \sup_{s,a \in \mathcal{S}\times\mathcal{A}} d(\eta(s,a), \eta'(s,a).$$

Then the iterates $\eta_{k+1} = \Pi_C \mathcal{T}^\pi \eta_k$ converge to the unique fixed point in $\mathcal{F}_{C,m}^{\mathcal{S}\times\mathcal{A}}$ with respect to $\bar{\ell}_2$ based on Banach's fixed point Theorem.
In particular, we highlight the contraction property

$$\bar{\ell}_2(\Pi_C \mathcal{T}^\pi \eta, \Pi_C \mathcal{T}^\pi \eta') \leq \sqrt{\gamma} \bar{\ell}_2(\eta, \eta') \tag{5}$$

[compare Rowland et al., 2018; Bellemare et al., 2023] while Π_C is a non-expansion.

Theorem 3 (Convergence of adaptive distributional Q-learning in the categorical setting)**.** *Given some initial return distribution functions η_0^A, η_0^B supported within $[\theta_1, \theta_m]$, the induced Q-values, i.e. the expected values of the return distributions $(\eta_t^A), (\eta_t^B)$, recursively defined by Algorithm 1 converge almost surely towards Q^* if the following conditions are satisfied:*

- *(i) the step sizes $\alpha_t(s,a)$ almost surely fulfill the Robbins-Monro conditions $\sum_{t=0}^\infty \alpha_t(s,a) = \infty$ and $\sum_{t=0}^\infty \alpha_t^2(s,a) < \infty$.*
- *(ii) rewards are bounded in $[R_{min}, R_{max}]$ and $[\frac{R_{min}}{1-\gamma}, \frac{R_{max}}{1-\gamma}] \subseteq [\theta_1, \theta_m]$,*
- *(iii) the choice of updating η^A or η^B is random and independent of all previous random variables*
- *(iv) the sequences $(\beta_t^A)_{t\in\mathbb{N}}, (\beta_t^B)_{t\in\mathbb{N}}$ only depend on the past and fulfill $\lim_{t\to\infty} |\beta_t^A - \beta_t^B| = 0$ almost surely.*

If additionally the MDP has a unique optimal policy π^*, then $(\eta_t^A), (\eta_t^B)$ converge almost surely in $\bar{\ell}_2$ to some limit $\eta_C^* \in \mathcal{F}_{C,m}$ and the greedy policy with respect to η_C^* is the optimal policy.

Note that the algorithm and proof uses $\beta_{t+1}^{A/B}(s,a)$ with index $t+1$ when updating $\eta_t^{A/B}$. This is to show that in general the parameter is allowed to depend on S_{t_1} and the respective greedy action, i.e. it must only be $\mathcal{F}_{t+1}$ measurable. To portray this generality in the following we will only write $\beta_{t+1}^{A/B}$ without referencing a state-action pair.

The simplest way to guarantee the assumptions on the adaptive parameters β^A, β^B to be satisfied is to chose them equal.

As in van Hasselt, 2010, the proof is based on the following stochastic approximation result, which has been proven in Singh et al., 2000 based on [Bertsekas and Tsitsiklis, 1996 Proposition 4.5].

Lemma 4 (Singh et al., 2000 Lemma 1)**.** *Suppose $(\Omega, \mathcal{A}, \mathbb{P}, (\mathcal{F}_n))$ is a filtered probability space on which all appearing random variables are defined. Suppose that*

- *(i) a stochastic process $(F_n)_{n\in\mathbb{N}} \subset \mathbb{R}^d$ with the coordinates $F_{i,n}$ for $i = 1, \dots, d$ such that F_n is $\mathcal{F}_{n+1}$-measurable and for all $i = 1, \dots, d$*

$$\|\mathbb{E}[F_n|\mathcal{F}_n]\|_\infty \leq \kappa \|X_n\|_\infty + c_n \quad \text{and} \quad \mathbb{V}[F_{i,n}|\mathcal{F}_n] \leq K(1 + \kappa\|X_n\|_\infty)^2 \quad n \geq 1,$$

where $\kappa \in [0,1]$, an adapted, stochastic process $(c_n)_{n\in\mathbb{N}} \subset \mathbb{R}^+$ that converges to 0 almost surely and some constant $K > 0$.

(ii) the non-negative stochastic process $(\alpha_n)_{n\in\mathbb{N}} \subset \mathbb{R}^d$*, with the coordinates* $\alpha_{i,n} \in [0,1]$ *for* $i = 1, \dots, d$ *is adapted with*

$$\sum_{n=1}^{\infty} \alpha_{i,n} = \infty \quad \textit{and} \quad \sum_{n=1}^{\infty} \alpha_{i,n}^2 < \infty \quad \textit{a.s..}$$

Then, for any $\mathcal{F}_0$*-measurable initial condition* X_0 *the stochastic process* $(X_n)_{n\in\mathbb{N}} \subset \mathbb{R}^d$ *with coordinates* $X_{i,n}$ *for* $i = 1, \dots, d$ *that is recursively defined by*

$$X_{i,n+1} = (1 - \alpha_{i,n})X_{i,n} + \alpha_{i,n}F_{i,n}, \quad n \in \mathbb{N},$$

converges almost surely to zero.

Furthermore, we follow Rowland et al., 2018 by first showing the convergence of the mean-values to Q^* and afterwards showing convergence of the return distribution functions, under the assumption of a unique optimal policy, by coupling it with policy evaluation. The convergence of the latter is easier to prove and we will do so at the end.

Lemma 5 (Adaptive Double Categorical Temporal Difference for Policy Evaluation)**.** *Given some initial return distribution functions* η_0^A, η_0^B *supported within* $[\theta_1, \theta_m]$ *and a stationary policy* $\pi \in \Pi_S$*, the return distribution functions* $(\eta_t^A), (\eta_t^B)$ *recursively defined by Algorithm 1, but with* $a^* \sim \pi(\cdot; S_{t+1})$ *instead, converge almost surely towards the unique fixed point* $\eta_C \in \mathcal{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}}$ *of the operator* $\Pi_C \mathcal{T}^\pi$ *with respect to* $\bar{\ell}_2$*, if the following conditions are satisfied:*

(i) the step sizes $\alpha_t(s,a)$ *fulfill the Robbins-Monro conditions:*

- $\sum_{t=0}^{\infty} \alpha_t(s,a) = \infty$
- $\sum_{t=0}^{\infty} \alpha_t^2(s,a) < \infty,$

(ii) rewards are bounded in $[R_{min}, R_{max}]$ *and* $[\frac{R_{min}}{1-\gamma}, \frac{R_{max}}{1-\gamma}] \subseteq [\theta_1, \theta_m]$*,*

(iii) the choice of updating η^A *or* η^B *is random and independent of all other previous random variables*

The above result is only relevant for the proof of Theorem 3, as policy evaluation with a double estimator is not of interest. Note that convergence of categorical temporal difference for policy evaluation (in the single estimator case) has been proven in [Rowland et al., 2018 Theorem 2 mimicking Tsitsiklis, 1994 Theorem 2] and [Bellemare et al., 2023 Theorem 6.12 applying Tsitsiklis, 1994 Theorem 3 or Bertsekas and Tsitsiklis, 1996 Proposition 4.5].

Lemma 6. *Let* $(\alpha_t)_{t\in\mathbb{N}_0}$ *be a sequence fulfilling the Robbins-Monro conditions and* $(Y_t)_{t\in\mathbb{N}}$ *an* iid *sequence of Bernoulli(0.5) random variables, i.e.* $\mathbb{P}(Y_t = 1) = \mathbb{P}(Y_t = 0) = 0.5$ *for all* $t \in \mathbb{N}_0$*. Then* $(\alpha_t Y_t)_{t\in\mathbb{N}_0}$ *also fulfills the Robbins-Monro condition.*

Proof. The almost sure convergence of the summed squares is obviously fulfilled due to

$$\sum_{t=0}^{\infty} (\alpha_t Y_t)^2 \leq \sum_{t=0}^{\infty} \alpha_t^2 < \infty \quad \text{almost surely.}$$

Due to independence of each Y_t with $\{Y_n | n \in \mathbb{N}_0,\ n \neq t\}$ as well as with $\alpha = (\alpha_t)_{t=0}^{\infty}$ we will consider a two stage experiment, where we first draw the sequence $\alpha = (\alpha_t)_{t=0}^{\infty}$ and then independently of this realization sample the iid sequence $Y = (Y_t)_{t=0}^{\infty}$. Due to the independence the joint measure of α and Y is the product measure. Consider the product space $(\Omega, \mathcal{F}, \mathbb{P}) = (\Omega_\alpha \times \Omega_Y, \mathcal{F}_\alpha \otimes \mathcal{F}_Y, \mathbb{P}_\alpha^{\otimes\mathbb{N}} \otimes \mathbb{P}_Y^{\otimes\mathbb{N}})$ where $\Omega_\alpha, \Omega_Y = [0,1]^{\mathbb{N}}$, $\mathcal{F}_\alpha, \mathcal{F}_Y = \mathcal{B}([0,1])^{\otimes\mathbb{N}}$. Then, using

that $\sum_{t=0}^{\infty} \alpha_t = \infty$ $\mathbb{P}_\alpha$-almost surely, we have

$$\begin{aligned}
&\mathbb{P}\Big(\sum_{t=0}^{\infty} \alpha_t Y_t = \infty\Big) \\
=&\int_{\Omega_\alpha} \mathbb{P}_Y\Big(\sum_{t=0}^{\infty} \alpha_t Y_t = \infty\Big) d\mathbb{P}_\alpha(\alpha) \\
=&\int_{\{(\alpha_t)_{t=0}^{\infty} \in \Omega_\alpha : \sum_{t=0}^{\infty} \alpha_t = \infty\}} \mathbb{P}_Y\Big(\sum_{t=0}^{\infty} \alpha_t Y_t = \infty\Big) d\mathbb{P}_\alpha(\alpha) \\
\overset{(a)}{=}&\int_{\{(\alpha_t)_{t=0}^{\infty} \in \Omega_\alpha : \sum_{t=0}^{\infty} \alpha_t = \infty\}} 1 \, d\mathbb{P}_\alpha(\alpha) \\
=&1,
\end{aligned}$$

where (a) can be seen as follows. Consider any deterministic sequence $(b_t) \subseteq [0,1]$ fulfilling $\sum_{t=0}^{\infty} b_t = \infty$. Then

$$\infty = \sum_{t=0}^{\infty} b_t = \sum_{t=0}^{\infty} b_t Y_t + \sum_{t=0}^{\infty} b_t \mathbf{1}_{Y_t=0}.$$

Now notice that $A = \sum_{t=0}^{\infty} b_t Y_t$ and $B = \sum_{t=0}^{\infty} b_t \mathbf{1}_{Y_t=0}$ are identically distributed and since the sum of A and B is always infinity, almost surely either one of them is infinite. Given the identical distribution, we infer

$$\mathbb{P}_Y(\sum_{t=0}^{\infty} b_t Y_t = \infty) > 0.$$

But since $(b_t Y_t)$ is an independent sequence of random variables and the event that the infinite sum diverges is in the tail sigma algebra, the Kolmogorov 0-1 law yields:

$$\mathbb{P}_Y(\sum_{t=0}^{\infty} b_t Y_t = \infty) = 1.$$

□

Remark 7. As outlined in [Rowland et al., 2018, Proof of Proposition 1], denoting by $\mathcal{M}(\mathbb{R})$ the space of all finite signed measures on $(\mathbb{R}, \mathcal{B}(\mathbb{R}))$, the subspace

$$\mathcal{M}_0(\mathbb{R}) := \{\nu \in \mathcal{M}(\mathbb{R}) | \nu(\mathbb{R}) = 0, \int_{\mathbb{R}} F_\nu(x)^2 dx < \infty\},$$

"where $F_\nu(x) = \nu([-\infty, x))$ for $x \in \mathbb{R}$, is isometrically isomorphic to a subspace of the Hilbert space $L^2(\mathbb{R})$ with inner product given by

$$\langle \nu_1, \nu_2 \rangle_{\ell_2} = \int_{\mathbb{R}} F_{\nu_1}(x) F_{\nu_2}(x) dx."$$

Then the affine translation $\delta_0 + \mathcal{M}_0$ is also an Hilbert endowed with the same inner product. It contains probability measures $\nu \in \mathcal{P}(\mathbb{R})$ satisfying

$$\int_{-\infty}^{0} F_\nu(x)^2 \, dx < \infty \quad \text{and} \quad \int_{0}^{\infty} (1 - F_\nu(x))^2 \, dx < \infty.$$

To see this, consider $\mu = \nu - \delta_0$ fulfills $F_\mu(x) = F_\nu(x)$ for $x < 0$ and $F_\mu(x) = F_\nu(x) - 1$ for $x \geq 1$. Hence, $\mu \in \mathcal{M}_0$. The two conditions assure that the tails decay fast enough.
Note that the inner product induces a norm through $\|\nu\|_{\ell_2}^2 = \langle \nu, \nu \rangle$. And we have $\ell_2(\nu_1, \nu_2) = \|\nu_1 - \nu_2\|_{\ell_2}$. In the following proof, we will make use of the relationship

$$\ell_2^2(\nu_1 + \nu_2, \nu_1' + \nu_2') = \|\nu_1 - \nu_1'\|_{\ell_2}^2 + \|\nu_2 - \nu_2'\|_{\ell_2}^2 + 2\langle \nu_1 - \nu_1', \nu_2 - \nu_2' \rangle$$

holding by bilinearity of the inner product.

Proof of Thoerem 3. **Step 1: Convergence of mean values to Q^***
The proof mainly follows Rowland et al., 2018 and van Hasselt, 2010. Let the filtration be given by $\mathcal{F}_t = \sigma(\eta_0^A, \eta_0^B, s_0, a_0, \alpha_0, R_0, S_1, Y_1, \beta_1^A, \beta_1^B \ldots, s_t, a_t, \alpha_t)$, where $(Y_n)_{n\in\mathbb{N}}$ is an iid sequence of *Bernoulli(0.5)* random variables, independent of all other appearing random variables, such that A is updated when $Y_{n+1} = 1$. Denote the expected values of the return-distributions by $Q_t^A(s,a) = \mathbb{E}_{R\sim\eta_t^A(s,a)}[R]$ and overloading notation, let us further write $\mathbb{E}[\nu]$ for the expected value $\mathbb{E}_{R\sim\nu}[R]$ of a probability distribution $\nu \in \mathcal{P}(\mathbb{R})$. We will first consider how the expected values evolve. Due to the symmetry of the updates it is sufficient to show convergence of Q_t^A to Q^*. It is implied that $\alpha(s,a) = 0$ for $(s,a) \neq (s_t, a_t)$. Further, define

$$\begin{aligned}
X_t(s_t,a_t) :=& Q_t^A(s_t,a_t) - Q^*(s_t,a_t)\\
F_t(s_t,a_t) :=& \mathbf{1}_{Y_{t+1}=1}\Big(R_t + \gamma(\beta_{t+1}^A Q_t^A(S_{t+1},a^*) + (1-\beta_{t+1}^A)Q_t^B(S_{t+1},a^*)) - Q^*(s_t,a_t)\Big)\\
&+\mathbf{1}_{Y_{t+1}=0}X_t(s_t,a_t)\\
F_t(s,a) :=& 0 \text{ whenever } (s,a) \neq (s_t,a_t)
\end{aligned}$$

with $a^* = \arg\max_{a'\in\mathcal{A}_{S_{t+1}}} Q^A(S_{t+1},a')$. According to [Lyle et al., 2019 Proposition 1] projection Π_C is mean-preserving, i.e $\mathbb{E}[\Pi_C\nu] = \mathbb{E}[\nu]$ for when ν is a distribution supported within $[\theta_1,\theta_m]$. This is the case for every $\hat{\eta}_*$ as in Algorithm 1, which can be seen as following. Assume $\eta_t^A(s_t,a_t), \eta_t^B(s_t,a_t) \in \mathcal{F}_{C,m}$. Then also

$$\nu = \beta_{t+1}^A\eta_t^A(S_{t+1},a^*) + (1-\beta_{t+1}^A)\eta_t^B(S_{t+1},a^*)) \in \mathcal{F}_{C,m},$$

and suppose $\nu = \sum_{i=1}^m p_i\delta_{\theta_i}$ for some p_i. Then

$$\hat{\eta}_* := b_{R_t,\gamma}\#\nu = \sum_{i=1}^m p_i\delta_{R_t+\gamma\theta_i}.$$

But now

$$\theta_1 \leq \frac{R_{min}}{1-\gamma} \leq \frac{R_{min}}{1-\gamma} \leq \theta_m$$

(Assumption (ii)) guarantees that

$$\theta_1 \leq R_t + \gamma\theta_i \leq \theta_m \quad \forall\, i \in \{1,\ldots,m\}$$

and $\hat{\eta}_*$ is supported within $[\theta_1,\theta_m]$. Similarly for a realized transition with $(R_t,S_{t+1}) = (r_t,s_{t+1})$, we have for the expected value of the distribution

$$\begin{aligned}
&\mathbb{E}\left[b_{r_t,\gamma}\#\Big(\beta_{t+1}^A\eta_t^A(s_{t+1},a^* + (1-\beta_{t+1}^A)\eta_t^B(s_{t+1},a^*))\Big)\right]\\
=& r_t + \gamma(\beta_{t+1}^A Q_t^A(s_{t+1},a^*) + (1-\beta_{t+1}^A)Q_t^B(s_{t+1},a^*)).
\end{aligned}$$

Hence, the expected values of the return distributions η_t^A subtracted by Q^* indeed evolve as

$$X_{t+1}(s,a) = (1-\alpha_t(s,a))X_t(s,a) + \alpha_t(s,a)F_t(s,a).$$

We now proceed similarly as in van Hasselt, 2010 to show that the conditions of Lemma 4 are satisfied.
We first show that $\mathbb{V}[F_t(s,a)|\mathcal{F}_t]$ is bounded for all $(s,a) \in \mathcal{S}\times\mathcal{A}$ and therefore satisfies $\mathbb{V}[F_t(s,a)|\mathcal{F}_t] \leq K(1+\kappa\|X_t\|_\infty)$ as required. Since the rewards were assumed to be bounded there is an $\bar{R} > 0$ such that $|r|, |\theta_1|, |\theta_m| \leq \bar{R}\ \forall r \in \mathcal{R}$. Hence, we have

$$\begin{aligned}
|F_t(s_t,a_t)| \leq& |R_t + \gamma(\beta_{t+1}^A Q_t^A(S_{t+1},a^*) + (1-\beta_{t+1}^A)Q_t^B(S_{t+1},a^*)) - Q^*(s_t,a_t)|\\
&+|X_t(s_t,a_t)|\\
\leq& \bar{R} + 3\bar{R} + 2\frac{\bar{R}}{1-\gamma}.
\end{aligned}$$

Next, we need to show that $\|\mathbb{E}[F_t \mid \mathcal{F}_t]\|_\infty \leq \kappa\|X_t\|_\infty + c_n$. Let us therefore decompose

$$\begin{aligned}
F_t(s_t,a_t) =& \mathbf{1}_{Y_{t+1}=1}\Big(F_t^Q(s_t,a_t) + \gamma(1-\beta_{t+1})(Q_t^B(S_{t+1},a^*) - Q_t^A(S_{t+1},a^*))\Big)\\
&+\mathbf{1}_{Y_{t+1}=0}\alpha_t(s_t,a_t)X_t(s_t,a_t)
\end{aligned}$$

with $F_t^Q(s_t, a_t) := R_t + \gamma Q_t^A(S_{t+1}, a^*) - Q^*(s_t, a_t)$. This yields

$$
\begin{aligned}
&|\mathbb{E}[\mathbf{1}_{Y_{t+1}=1} F_t^Q(s_t, a_t) + \mathbf{1}_{Y_{t+1}=0} \alpha_t(s_t, a_t) X_t(s_t, a_t) | \mathcal{F}_t]| \\
&= |\frac{1}{2}\mathbb{E}[R_t + \gamma Q_t^A(S_{t+1}, a^*)] - Q^*(s_t, a_t) + \frac{1}{2} X_t(s_t, a_t)| \\
&\leq |T^* Q^A(s_t, a_t) - T^* Q^*(s_t, a_t)| + |\frac{1}{2} X_t(s_t, a_t)| \\
&\leq \gamma \|Q_t^A - Q^*\|_\infty + \frac{1}{2}\|X_t\|_\infty \\
&= \underbrace{(\frac{1}{2}\gamma + \frac{1}{2})}_{<1} \|X_t\|_\infty,
\end{aligned}
$$

since the Bellman optimality operator is a γ-contraction. Subsequently, it only remains to show that

$$
c_t := |\mathbb{E}[\mathbf{1}_{Y_{t+1}=1} \gamma (1 - \beta_{t+1}^A)(Q_t^B(S_{t+1}, a^*) - Q_t^A(S_{t+1}, a^*) | \mathcal{F}_t]|
$$

goes to zero almost surely. This is immediate if we verify that

$$
X_t^{BA}(s, a) := Q_t^B(s, a) - Q_t^A(s, a)
$$

goes to zero almost surely for all $(s, a) \in \mathcal{S} \times \mathcal{A}$ which will be achieved by another application of Lemma 4. We infer that

$$
\begin{aligned}
&X_{n+1}^{BA}(s_n, a_n) \\
=&X_n^{BA}(s_n, a_n) + \alpha_n(s_n, a_n)\Big(\\
&\mathbf{1}_{Y_{n+1}=0}\Big(R_n + \gamma\big(\beta_{n+1}^B Q_n^B(S_{n+1}, b^*) + (1 - \beta_{n+1}^B) Q_n^A(S_{n+1}, b^*)\big) - Q_n^B(s_n, a_n)\Big) \\
&-\mathbf{1}_{Y_{n+1}=1}\Big(R_n + \gamma\big(\beta_{n+1}^A Q_n^A(S_{n+1}, a^*) + (1 - \beta_{n+1}^A) Q_n^B(S_{n+1}, a^*)\big) - Q_n^A(s_n, a_n)\Big) \\
&\Big) \\
=&(1 - \alpha_n(s_n, a_n)) X_n^{BA}(s_n, a_n) + \alpha_n(s_n, a_n)\Big(\\
&\mathbf{1}_{Y_{n+1}=0}\Big(R_n + \gamma\big(\beta_{n+1}^B Q_n^B(S_{n+1}, b^*) + (1 - \beta_{n+1}^B) Q_n^A(S_{n+1}, b^*)\big)\Big) \\
&-\mathbf{1}_{Y_{n+1}=1}\Big(R_n + \gamma\big(\beta_{n+1}^A Q_n^A(S_{n+1}, a^*) + (1 - \beta_{n+1}^A) Q_n^B(S_{n+1}, a^*)\big)\Big) \\
&+\mathbf{1}_{Y_{n+1}=1} Q_n^B(s_n, a_n) - \mathbf{1}_{Y_{n+1}=0} Q_n^A(s_n, a_n)\Big) \\
=&(1 - \alpha_n(s_n, a_n)) X_n^{BA}(s_n, a_n) + \alpha_n(s_n, a_n) \tilde{F}_n(s_n, a_n),
\end{aligned}
$$

with

$$
\begin{aligned}
\tilde{F}_n(s_n, a_n) = &\Big(\mathbf{1}_{Y_{n+1}=0}\Big(R_n + \gamma\big(\beta_{n+1}^B Q_n^B(S_{n+1}, b^*) + (1 - \beta_{n+1}^B) Q_n^A(S_{n+1}, b^*)\big)\Big) \\
&-\mathbf{1}_{Y_{n+1}=1}\Big(R_n + \gamma\big(\beta_{n+1}^A Q_n^A(S_{n+1}, a^*) + (1 - \beta_{n+1}^A) Q_n^B(S_{n+1}, a^*)\big)\Big) \\
&+\mathbf{1}_{Y_{n+1}=1} Q_n^B(s_n, a_n) - \mathbf{1}_{Y_{n+1}=0} Q_n^A(s_n, a_n)\Big).
\end{aligned}
$$

Now, using that $Q_n^B(s_n,a_n), Q_n^A(s_n,a_n), X_n^{BA}(s_n,a_n), \alpha_n(s_n,a_n)$ are $\mathcal{F}_n$-measurable and Y_{n+1} is independent of $\mathcal{F}_n$, the conditional expectation satisfies

$$\begin{aligned}
|\mathbb{E}[\tilde{F}_n(s_n,a_n) \mid \mathcal{F}_n]| =& \frac{1}{2}\gamma|\mathbb{E}[\beta_{n+1}^B Q_n^B(S_{n+1},b^*) + (1-\beta_{n+1}^B)Q_n^A(S_{n+1},b^*) \\
&-\beta_{n+1}^A Q_n^A(S_{n+1},a^*) - (1-\beta_{n+1}^A)Q_n^B(S_{n+1},a^*)|\mathcal{F}_n]| \\
&+\frac{1}{2}|Q_n^B(s_n,a_n) - Q_n^A(s_n,a_n)| \\
\leq& \frac{1}{2}\gamma\Big(\big|\mathbb{E}[\beta_{n+1}^B(Q_n^B(S_{n+1},b^*) - Q_n^A(S_{n+1},a^*))|\mathcal{F}_n]\big| \\
&+\big|\mathbb{E}[(1-\beta_{n+1}^B)(Q_n^A(S_{n+1},b^*) - Q_n^B(S_{n+1},a^*))|\mathcal{F}_n]\big| \\
&+\big|\mathbb{E}[(\beta_{n+1}^B - \beta_{n+1}^A)Q_n^A(S_{n+1},a^*)|\mathcal{F}_n]\big| \\
&+\big|\mathbb{E}[((1-\beta_{n+1}^B) - (1-\beta_{n+1}^A))Q_n^B(S_{n+1},a^*)|\mathcal{F}_n]\big|\Big) \\
&+\frac{1}{2}\|X_n\|_\infty
\end{aligned}$$

Now if $\mathbb{E}[Q_n^B(S_{n+1},b^*)|\mathcal{F}_n] \geq \mathbb{E}[Q_n^A(S_{n+1},a^*)|\mathcal{F}_n]$, by definition of a^* we have $Q_n^A(S_{n+1},a^*) = \max_{a\in\mathcal{A}_{S_{n+1}}} Q_n^A(S_{n+1},a) \geq Q_n^A(S_{n+1},b^*)$ and therefore

$$\begin{aligned}
\big|\mathbb{E}[Q_n^B(S_{n+1},b^*) - Q_n^A(S_{n+1},a^*)|\mathcal{F}_n]\big| =& \mathbb{E}[Q_n^B(S_{n+1},b^*) - Q_n^A(S_{n+1},a^*)|\mathcal{F}_n] \\
\leq& \mathbb{E}[Q_n^B(S_{n+1},b^*) - Q_n^A(S_{n+1},b^*)|\mathcal{F}_n] \leq \|X_n^{BA}\|_\infty.
\end{aligned}$$

Analogously, if $\mathbb{E}[Q_n^B(S_{n+1},b^*)|\mathcal{F}_n] < \mathbb{E}[Q_n^A(S_{n+1},a^*)|\mathcal{F}_n]$, then we have by definition of b^*

$$\begin{aligned}
\big|\mathbb{E}[Q_n^B(S_{n+1},b^*) - Q_n^A(S_{n+1},a^*)|\mathcal{F}_n]\big| =& \mathbb{E}[Q_n^A(S_{n+1},a^*) - Q_n^B(S_{n+1},b^*)|\mathcal{F}_n] \\
\leq& \mathbb{E}[Q_n^A(S_{n+1},a^*) - Q_n^B(S_{n+1},a^*)|\mathcal{F}_n] \leq \|X_n^{BA}\|_\infty.
\end{aligned}$$

Similarly, by distinguishing cases, one shows that

$$\big|\mathbb{E}[Q_n^A(S_{n+1},b^*) - Q_n^B(S_{n+1},a^*)|\mathcal{F}_n]\big| \leq \|X_n^{BA}\|_\infty + \frac{1}{2}\|X_n^{BA}\|.$$

Combining the above yields

$$\begin{aligned}
&\Big|\mathbb{E}[\tilde{F}_n(s_n,a_n) \mid \mathcal{F}_n]\Big| \leq \frac{1}{2}\gamma(\beta_{n+1}^B + (1-\beta_{n+1}^B))\|X_n^{BA}\|_\infty \\
&+\underbrace{\Big|\gamma\mathbb{E}[(\beta_{n+1}^B - \beta_{n+1}^A)\underbrace{Q_n^A(S_{n+1},a^*)}_{<\bar{R}<\infty}|\mathcal{F}_n]\Big| + \Big|\gamma\mathbb{E}[((1-\beta_{n+1}^B) - (1-\beta_{n+1}^A))\underbrace{Q_n^B(S_{n+1},a^*)}_{<\bar{R}<\infty}|\mathcal{F}_n]\Big|}_{:=\tilde{c}_n\to 0,\ \text{since } |\beta_n^A - \beta_n^B| \text{ converges to 0 for } n\to\infty \text{ due to } (iv)}.
\end{aligned}$$

Hence, we invoke Lemma 4 to obtain convergence of X_t^{BA} and thus with another application of Lemma 4, $X_t(s,a)$ converges to zero which finally implies $Q_t^A(s,a)$ (and also $Q^B(s,a)$) converges to $Q^*(s,a)$ almost surely for every $(s,a) \in \mathcal{S}\times\mathcal{A}$.

Since $\mathcal{S},\mathcal{A}$ are finite, for every $\varepsilon > 0$, there exists a random variable $N > 0$ such that for all $t > N$, we have

$$\max_{z\in\{A,B\}} \|Q_t^z - Q^*\|_\infty < \varepsilon \quad \text{almost surely.}$$

Step 2: Convergence of return distributions

Suppose the MDP has a unique optimal policy π^*. Now following Rowland et al., 2018, we take ε to be half the minimum action gap for the optimal action-value function $Q^* = Q^{\pi^*}$, i.e.

$$\varepsilon = \frac{1}{2}\min_{s\in\mathcal{S}}(Q^{\pi^*}(s,\pi^*(s)) - \max_{a\neq} Q^{\pi^*}(s,a))$$

which is greater than zero by assumption (v). Hence, denoting the action of the deterministic optimal policy in a certain state s by $\pi^*(s)$, we get

$$\max_a Q_t^A(s,a) = \max_a Q_t^B(s,a) = \pi^*(s)$$

for all $t > N$. For some initial condition $\tilde{\eta}_0 \in \mathcal{F}^{\mathcal{S}}_{C,m}$, let now $\tilde{\eta}_k$ be the iterates created by a double categorical policy evaluation algorithm for the optimal policy π^*, i.e.

$$\begin{aligned}\tilde{\eta}^A_{k+1}(s_k,a_k) =&(1-\mathbf{1}_{Y_{k+1}=1}\alpha_k(s_k,a_k))\tilde{\eta}_k(s_k,a_k)\\ &+\mathbf{1}_{Y_{k+1}=1}\alpha_k(s_k,a_k)\Pi_C\Big(b_{R_k,\gamma}\#\Big(\beta^A_{k+1}\tilde{\eta}^A_k(S_{k+1},\pi^*(S_{k+1}))\\ &+(1-\beta^A_{k+1})\tilde{\eta}^B_k(S_{k+1},\pi^*(S_{k+1}))\Big)\Big)\\ \tilde{\eta}^A_{k+1}(s,a) =&\tilde{\eta}^A_k(s,a) \text{ for } (s,a)\neq(s_k,a_k).\end{aligned}$$

and analogously for $\tilde{\eta}^B$. Note that the appearing $\mathbb{Y}_k, \alpha_k, \beta^A_k, \beta^B_k$ are chosen to be the same as in the control case above. Then $\tilde{\eta}^A, \tilde{\eta}^B$ converges almost surely to the unique fixed point η^*_C of the projected operator $\Pi_C\mathcal{T}^{\pi^*}$ with respect to $\bar{\ell}_2$ by Lemma 5. Similarly to Rowland et al., 2018, we now proceed by a coupling argument. Denote by π^A_k, π^B_k any greedy selection rule with respect to η^A_k and η^B_k and $A_k = \{\pi^A_k = \pi^B_k = \pi^* \text{ for all } n \geq k\}$. Then $A_k \subseteq A_{k+1}$ and by the above explanation we have $\mathbb{P}(A_k) \nearrow 1$. Additionally, let B be the event of probability 1 for which the (double) policy evaluation algorithm converges. Now on the event $B \cup A_k$, we have

$$\bar{\ell}^2_2(\tilde{\eta}^A_n, \eta^*_C) \to 0 \quad \text{and} \quad \bar{\ell}^2_2(\tilde{\eta}^B_n, \eta^*_C) \to 0.$$

Then by the triangle inequality it suffices to show $\bar{\ell}_2(\eta^A_n, \tilde{\eta}^A_n) \to 0$ and $\bar{\ell}_2(\eta^B_n, \tilde{\eta}^B_n) \to 0$ on this event too, since then the Theorem follows by $\mathbb{P}(B \cup A_k) \nearrow 1$.
To prove this we will again apply Lemma 4. This time with $d = 2 \cdot |\mathcal{S}||\mathcal{A}|$, where we identify

$$X_n := \begin{bmatrix}\ell^2_2(\eta^A_n, \tilde{\eta}^A_n)\\ \ell^2_2(\eta^B_n, \tilde{\eta}^B_n)\end{bmatrix} \in \mathbb{R}^{2|\mathcal{S}||\mathcal{A}|}.$$

Additionally, we expand the filtration by $\tilde{\mathcal{F}}_n = \sigma(\mathcal{F}_n, Y_{n+1})$ and define $\tilde{\alpha}^A_n(s,a) = \alpha_n(s,a)\mathbf{1}_{Y_{n+1}=1}$ and $\tilde{\alpha}^B_n(s,a) = \alpha_n(s,a)\mathbf{1}_{Y_{n+1}=0}$. By Lemma 6 these steps-size sequences still fulfill the Robbins-Monro conditions.
Then, writing

$$\begin{aligned}\nu^A =&\beta^A_{n+1}\eta^A_n(S_{n+1},\pi^*(S_{n+1})) + (1-\beta^A_{n+1})\eta^B_n(S_{n+1},\pi^*(S_{n+1}))\\ \tilde{\nu}^A =&\beta^A_{n+1}\tilde{\eta}^A_n(S_{n+1},\pi^*(S_{n+1})) + (1-\beta^A_{n+1})\tilde{\eta}^B_n(S_{n+1},\pi^*(S_{n+1}))\end{aligned}$$

for short, for $n \geq k$, on A_k we have

$$\begin{aligned}&\ell^2_2(\eta^A_{n+1}(s_n,a_n), \tilde{\eta}^A_{n+1}(s_n,a_n))\\ =&(1-\tilde{\alpha}^A_n(s_n,a_n))^2\|\eta^A_n(s_n,a_n) - \tilde{\eta}^A_n(s_n,a_n)\|^2_{\ell_2}\\ &+\tilde{\alpha}^A_n(s_n,a_n)^2\|\Pi_C(b_{R_n,\gamma}\#\nu^A) - \Pi_C(b_{R_n,\gamma}\#\tilde{\nu}^A)\|^2_{\ell_2}\\ &+(1-\tilde{\alpha}^A_n(s_n,a_n))\tilde{\alpha}^A_n(s_n,a_n)2\langle\eta^A_n(s_n,a_n) - \tilde{\eta}^A_n(s_n,a_n), \Pi_C(b_{R_n,\gamma}\#\nu^A) - \Pi_C(b_{R_n,\gamma}\#\tilde{\nu}^A)\rangle_{\ell_2}.\end{aligned}$$

This can be rewritten in terms of Lemma 4 as

$$X^A_{n+1}(s_n,a_n) = (1-\zeta^A_n(s_n,a_n))X^A_n(s_n,a_n) + \zeta^A_n(s_n,a_n)F^A_n(s_n,a_n)$$

with $\zeta^A_n(s_n,a_n) = 2\tilde{\alpha}^A_n(s_n,a_n) - \tilde{\alpha}^A_n(s_n,a_n)^2$ and

$$\begin{aligned}F^A_n(s_n,a_n) =&\frac{1}{\zeta^A_n(s_n,a_n)}(\tilde{\alpha}^A_n(s_n,a_n)^2\|\Pi_C(b_{R_n,\gamma}\#\nu^A) - \Pi_C(b_{R_n,\gamma}\#\tilde{\nu}^A)\|^2_{\ell_2}\\ &+(1-\tilde{\alpha}^A_n(s_n,a_n))\tilde{\alpha}^A_n(s_n,a_n)2\langle\eta^A_n(s_n,a_n) - \tilde{\eta}^A_n(s_n,a_n),\\ &\Pi_C(b_{R_n,\gamma}\#\nu^A) - \Pi_C(b_{R_n,\gamma}\#\tilde{\nu}^A)\rangle_{\ell_2})\end{aligned}$$

and $F^A_n(s,a) = 0$ if $(s,a) \neq (s_n,a_n)$. It is mentioned that $\zeta^A_n(s_n,a_n) > 0$. Notice that,

$$\begin{aligned}\sum_{n=1}^{\infty}\zeta^A_n(s_n,a_n) &= \sum_{n=1}^{\infty}(2\tilde{\alpha}^A_n(s_n,a_n) - \tilde{\alpha}^A_n(s_n,a_n)^2) = \infty \quad a.s.\\ \sum_{n=1}^{\infty}\zeta^A_n(s_n,a_n)^2 &= \sum_{n=1}^{\infty}4\tilde{\alpha}^A_n(s_n,a_n)^2 - 4\tilde{\alpha}^A_n(s_n,a_n)^3 + \tilde{\alpha}^A_n(s_n,a_n)^2 < \infty \quad a.s.\end{aligned} \tag{6}$$

Finally we have

$$
\begin{aligned}
|F_n^A(s_n,a_n)| \leq & \frac{1}{\zeta_n^A(s_n,a_n)}(\tilde{\alpha}_n^A(s_n,a_n)^2\gamma\bar{\ell}_2^2(\beta_{n+1}^A\eta_n^A+(1-\beta_{n+1}^A)\eta_n^B,\beta_{n+1}^A\tilde{\eta}_n^A+(1-\beta_{n+1}^A)\tilde{\eta}_n^B) \\
& +(1-\tilde{\alpha}_n^A(s_n,a_n))\tilde{\alpha}_n^A(s_n,a_n)2\sqrt{\gamma}|\langle\eta_n^A-\tilde{\eta}_n^A, \\
& \beta_n^A\eta_n^A+(1-\beta_n^A)\eta_n^B-\beta_n^A\tilde{\eta}_n^A-(1-\beta_n^A)\tilde{\eta}_n^B\rangle_{\bar{\ell}_2}|) \\
\leq & \frac{1}{\zeta_n^A(s_n,a_n)}(\tilde{\alpha}_n^A(s_n,a_n)^2\gamma\max_{z\in\{A,B\}}\bar{\ell}_2^2(\eta_n^z,\tilde{\eta}_n^z) \\
& +(1-\tilde{\alpha}_n^A(s_n,a_n))\tilde{\alpha}_n^A(s_n,a_n)2\sqrt{\gamma}\max_{z\in\{A,B\}}\bar{\ell}_2^2(\eta_n^z,\tilde{\eta}_n^z)) \\
= & \frac{\tilde{\alpha}_n^A(s_n,a_n)^2\gamma+(1-\tilde{\alpha}_n^A(s_n,a_n))\tilde{\alpha}_n^A(s_n,a_n)2\sqrt{\gamma}}{2\tilde{\alpha}_n^A(s_n,a_n)-\tilde{\alpha}_n^A(s_n,a_n)^2}\max_{z\in\{A,B\}}\bar{\ell}_2^2(\eta_n^z,\tilde{\eta}_n^z) \\
\leq & \frac{(2\tilde{\alpha}_n^A(s_n,a_n)-\tilde{\alpha}_n^A(s_n,a_n)^2)\sqrt{\gamma}}{2\tilde{\alpha}_n^A(s_n,a_n)-\tilde{\alpha}_n^A(s_n,a_n)^2}\max_{z\in\{A,B\}}\bar{\ell}_2^2(\eta_n^z,\tilde{\eta}_n^z) \\
\leq & \sqrt{\gamma}\max_{z\in\{A,B\}}\bar{\ell}_2^2(\eta_n^z,\tilde{\eta}_n^z)=\sqrt{\gamma}\|X_n\|_\infty
\end{aligned}
$$

where we used regularity and 1/2-homogeneity of the ℓ_2 metric as described in [Bellemare et al., 2023 Section 4.6] as well as that Π_C is a non-expansion in ℓ_2 and

$$
\begin{aligned}
&|\langle u,\beta u+(1-\beta)v\rangle| = \beta\langle u,u\rangle+(1-\beta)|\langle u,v\rangle| \leq \beta\max(\|u\|^2,\|v\|^2)+(1-\beta)\|u\|\|v\| \\
&\leq \max(\|u\|^2,\|v\|^2)
\end{aligned}
$$

by the Cauchy-Schwarz inequality. Further, by the above the Variance also fulfills

$$
\begin{aligned}
\mathbb{V}[F_n^A(s_n,a_n)|\tilde{\mathcal{F}}_n] &= \mathbb{E}[F_n^A(s_n,a_n)^2|\mathcal{F}_n]-\mathbb{E}[F_n^A(s_n,a_n)|\tilde{\mathcal{F}}_n]^2 \\
&\leq 2(\sqrt{\gamma}\max_{z\in\{A,B\}}\bar{\ell}_2^2(\eta_n^z,\tilde{\eta}_n^z))^2 \\
&\leq 2\gamma\sup_{\eta,\eta\in\mathcal{F}_{C,m}^{\mathcal{S}}}\bar{\ell}_2^4(\eta,\eta')<\infty.
\end{aligned}
$$

Therefore, by Lemma 4 we obtain convergence $\bar{\ell}_2(\eta_n^A,\tilde{\eta}_n^A)\to 0$ and $\bar{\ell}_2(\eta_n^B,\tilde{\eta}_n^B)\to 0$ on A_k. As already described above, this results in

$$
\bar{\ell}_2(\eta_n^A,\eta_C^*)\to 0 \quad \text{and} \quad \bar{\ell}_2(\eta_n^B,\eta_C^*)\to 0 \quad \text{almost surely.}
$$

□

Proof of Lemma 5. Let the filtration be given by $\mathcal{F}_t = \sigma(\eta_0^A,\eta_0^B,s_0,a_0,\alpha_0,R_0,S_1,Y_1,\beta_1^A,\beta_1^B\ldots,s_t,a_t,\alpha_t,Y_{t+1})$, where $(Y_n)_{n\in\mathbb{N}}$ is an iid sequence of *Bernoulli(0.5)* random variables, independent of all other appearing random variables, such that A is updated when $Y_{n+1}=1$. To clarify, abbreviating

$$
\begin{aligned}
\nu^A &= \beta_{t+1}^A\eta_t^A(S_{t+1},A_{t+1})+(1-\beta_{t+1}^A)\eta_t^B(S_{t+1},A_{t+1}) \\
\nu^B &= \beta_{t+1}^B\eta_t^B(S_{t+1},A_{t+1})+(1-\beta_{t+1}^B)\eta_t^A(S_{t+1},A_{t+1}) \quad \text{where} \\
A_{t+1} &\sim \pi(\cdot;S_{t+1}),
\end{aligned}
$$

we are confronted with the updates

$$
\begin{aligned}
\eta_{t+1}^A(s,a) &= \eta_{t+1}^A(s,a)+\alpha_t(s,a)\mathbf{1}_{Y_{t+1}=1}(\Pi_C(b_{R_t,\gamma}\#\nu^A)-\eta_{t+1}^A(s,a)) \\
\eta_{t+1}^B(s,a) &= \eta_{t+1}^B(s,a)+\alpha_t(s,a)\mathbf{1}_{Y_{t+1}=0}(\Pi_C(b_{R_t,\gamma}\#\nu^B)-\eta_{t+1}^B(s,a)).
\end{aligned}
$$

As in the proof above, define $\tilde{\alpha}_n^A(s,a)=\alpha_n(s,a)\mathbf{1}_{Y_{n+1}=1}$ and $\tilde{\alpha}_n^B(s,a)=\alpha_n(s,a)\mathbf{1}_{Y_{n+1}=0}$. By Lemma 6 these steps-size sequences still fulfill the Robbins-Monro conditions. Also note that as in step 2 of the proof of Theorem 3, Y_{t+1} is $\mathcal{F}_t$-measurable and hence so is $\tilde{\alpha}_t^{A/B}$. In order to align this

with Lemma 4, we rewrite

$$\begin{aligned}
&X^A_{n+1}(s,a) = \ell_2^2(\eta^B_{t+1}(s,a), \eta_C(s,a)) \\
=&(1-\tilde{\alpha}^A_t(s,a))^2 \|\eta^A_t(s,a) - \eta_C(s,a)\|^2_{\ell_2} \\
+&\tilde{\alpha}^A_t(s,a)^2 \|\Pi_C(b_{R_t,\gamma}\#\nu^A) - \eta_C(s,a)\|^2_{\ell_2} \\
+&(1-\tilde{\alpha}^A_t(s,a))\tilde{\alpha}^A_t(s,a) 2\langle \eta^A_t(s,a) - \eta_C(s,a), \Pi_C(b_{R_t,\gamma}\#\nu^A) - \eta_C(s,a)\rangle_{\ell_2} \\
=&(1-\zeta^A_t(s,a))X^A_t(s,a) + \zeta^A_t(s,a) F^A_t(s,a)
\end{aligned}$$

with $\zeta^A_t(s,a) = 2\tilde{\alpha}^A_t(s,a) - \tilde{\alpha}^A_t(s,a)^2$,

$$X_t := \begin{bmatrix} \ell_2^2(\eta^A_t, \eta_C) \\ \ell_2^2(\eta^B_t, \eta_C) \end{bmatrix} \in \mathbb{R}^{2|\mathcal{S}||\mathcal{A}|}$$

and

$$\begin{aligned}
F^A_t(s,a) =& \frac{1}{\zeta^A_t(s,a)} \mathbf{1}_{\tilde{\alpha}^A_t(s,a)>0} (\tilde{\alpha}^A_t(s,a)^2 \ell_2^2(\Pi_C(b_{R_t,\gamma}\#\nu^A), \eta_C(s,a)) \\
&+(1-\tilde{\alpha}^A_t(s,a))\tilde{\alpha}^A_t(s,a) 2\langle \eta^A_t(s,a) - \eta_C(s,a), \Pi_C(b_{R_t,\gamma}\#\nu^A) - \eta_C(s,a)\rangle_{\ell_2}).
\end{aligned}$$

As in Equation (6), the sequence $\zeta^A_t(s,a)$ fulfills the Robbins-Monro condition. Additionally, note that there exists $K > 0$, such that $\ell_2^2(\Pi_C(b_{R_t,\gamma}\#\nu^A), \eta_C(s,a)) < K$ independent of s, a, t. Further, observe that

$$c_t := \max_{z\in\{A,B\}} \frac{1}{\zeta^z_t(s,a)} \mathbf{1}_{\tilde{\alpha}^z_t(s,a)>0} \tilde{\alpha}^z_t(s,a)^2 K \to 0 \text{ for } t \to \infty \text{ almost surely.}$$

We use that Π_C is mean-preserving [Lyle et al., 2019 Proposition 1] for discrete distributions supported within $[\theta_1, \theta_m]$, which is satisfied by $b_{R_t,\gamma}\#\nu^A$, due to Assumption (ii) and $\nu^A \in \mathcal{F}_m$. Together with the fact that $\Pi_C\mathcal{T}^\pi$ is a $\sqrt{\gamma}$-contraction with respect to $\bar{\ell}_2$ and the Cauchy-Schwarz inequality, we have

$$\begin{aligned}
&|\mathbb{E}[\langle \eta^A_t(s,a) - \eta_C(s,a), \Pi_C(b_{R_t,\gamma}\#\nu^A) - \eta_C(s,a)\rangle_{\ell_2} | \mathcal{F}_t]| \\
=&|\langle \eta^A_t(s,a) - \eta_C(s,a), \mathbb{E}[\Pi_C(b_{R_t,\gamma}\#\nu^A)|\mathcal{F}_t] - \eta_C(s,a)\rangle_{\ell_2}| \\
=&|\langle \eta^A_t(s,a) - \eta_C(s,a), \mathbb{E}[b_{R_t,\gamma}\#\nu^A)|\mathcal{F}_t] - \eta_C(s,a)\rangle_{\ell_2}| \\
=&|\langle \eta^A_t(s,a) - \eta_C(s,a), \Pi_C\mathcal{T}^\pi(\beta^A_{t+1}\eta^A_t + (1-\beta^A_{t+1})\eta^B_t)(s,a) - (\Pi_C\mathcal{T}^\pi\eta_C)(s,a)\rangle_{\ell_2}| \\
\leq&\sqrt{\gamma}|\langle \eta^A_t - \eta_C, (\beta^A_{t+1}\eta^A_t + (1-\beta^A_{t+1})\eta^B_t) - \eta_C\rangle_{\bar{\ell}_2}| \\
\leq&\sqrt{\gamma}(\beta^A_{t+1}\bar{\ell}_2^2(\eta^A_t, \eta_C) + (1-\beta^A_{t+1})|\langle \eta^A_t - \eta_C, \eta^B_t - \eta_C\rangle_{\bar{\ell}_2}|) \\
\leq&\sqrt{\gamma}(\beta^A_{t+1} \max_{z\in\{A,B\}} \bar{\ell}_2^2(\eta^z_t, \eta_C) + (1-\beta^A_{t+1})\|\eta^A_t - \eta_C\|_{\bar{\ell}_2}\|\eta^B_t - \eta_C\|_{\bar{\ell}_2}|) \\
\leq&\sqrt{\gamma} \max_{z\in\{A,B\}} \bar{\ell}_2^2(\eta^z_t, \eta_C) \\
=&\sqrt{\gamma}\|X_t\|_\infty.
\end{aligned}$$

In total, this yields

$$\begin{aligned}
&|\mathbb{E}[F^A_t(s,a)|\mathcal{F}_t]| \\
&\leq \frac{1}{\zeta^A_t(s,a)} \mathbf{1}_{\tilde{\alpha}^A_t(s,a)>0} \tilde{\alpha}^A_t(s,a)^2 K + \frac{1}{\zeta^A_t(s,a)} \mathbf{1}_{\tilde{\alpha}^A_t(s,a)>0} (1-\tilde{\alpha}^A_t(s,a))\tilde{\alpha}^A_t(s,a) 2\sqrt{\gamma}\|X_t\|_\infty. \\
&\leq c_t + \sqrt{\gamma}\|X_t\|_\infty.
\end{aligned}$$

Since $\bar{\ell}_2(\eta, \eta') < K$ for every $\eta, \eta' \in \mathcal{F}^{\mathcal{S}\times\mathcal{A}}_{C,m}$ some $K > 0$, the conditional variance $\mathbb{V}[F^A_t|\mathcal{F}_t]$ can be bounded uniformly in t.
Therefore, the requirements of Lemma 4 are fulfilled, and its application yields $X^A_t(s,a) = \ell_2^2(\eta^A_t(s,a), \eta_C(s,a)) \to 0$ and $X^B_t(s,a) = \ell_2^2(\eta^B_t(s,a), \eta_C(s,a)) \to 0$. Hence, also η^A_t, η^B_t converge to η_C with respect to $\bar{\ell}_2$. □