Aggressive Q-Learning with Ensembles: Achieving Both High Sample Efficiency and High Asymptotic Performance Supplementary Material

A Hyperparameters and implementation details

Table 1 gives a list of hyperparameters used in the experiments. Most of AQE's hyperparameters are made the same as in the REDQ paper to ensure fairness and consistency in comparisons, except that AQE has 2-head critic networks. As compared with AQE and REDQ, TQC uses a larger critic network with 3 layers of 512 units per layer. In table 2, we report the dropped atoms d for TQC and the number of Q values we keep in the ensemble to calculate the target in AQE. In algorithm **??**, we provide detailed pseudo-code.

Table 1: Hyperparameter values.

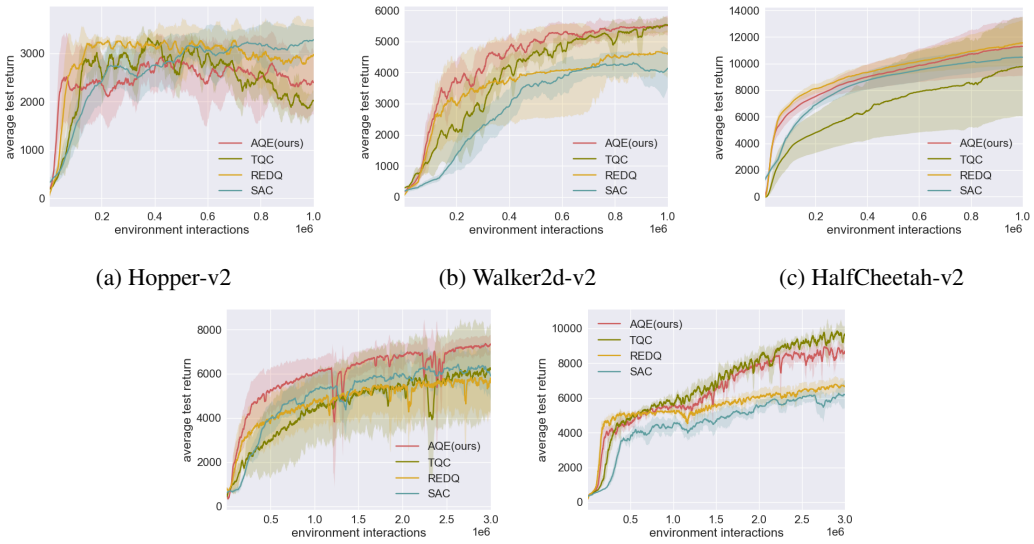
Hyperparameters	AQE	SAC	REDQ	TQC
optimizer	Adam			
learning rate	$3 \cdot 10^{-4}$			
discount(γ)	0.99			
target smoothing coefficient(ρ)	0.005			
replay buffer size	$1 \cdot 10^{6}$			
number of critics N	10	2	10	5
number of hidden layers in critic networks	2	2	2	3
size of hidden layers in critic networks	256	256	256	512
number of heads in critic networks h	2	1	1	25
number of hidden layers in policy network	2			
size of hidden layers in policy network	256			
mini-batch size	256			
nonlinearity	ReLU			
UTD ratio G	5	1	5	1

Table 2: Environment-dependent hyper-parameters for TQC and AQE.

Environment	Dropped atoms per critic	Kept Q values out of $N \cdot h$ values
Hopper	5	10
HalfCheetah	0	20
Walker	2	16
Ant	2	16
Humanoid	2	16

B Additional Results for AQE, TQC, REDQ and SAC in MuJoCo Benchamrk with Fixed Hyper-parameters

We present the experiment on the five MuJoCo environments with the same hyperparameter values across environments for TQC (drop two atoms per network) and AQE (K = 16) in Figure 1.



(a) Hopper-v2 (b) Walker2d-v2 (c) HalfCheetah-v2

(d) Ant-v2 (e) Humanoid-v2

Figure 1: Performance for AQE and TQC using same hyper-parameters across the five environments. AQE uses $K = 16$ and TQC uses atoms = 2 per critic.

Table 3: Early-stage performance comparison of SAC, TQC, REDQ and AQE when AQE and TQC using the same hyperparameters across the environments. On average, AQE performs 2.71 times better than SAC, 1.59 times better than TQC and 1.02 times better than REDQ.

Amount of data	SAC	TQC	REDQ	AQE	AQE/SAC	AQE/TQC	AQE/REDQ
Hopper at 100K	1456	1719	2747	2294	1.58	1.33	0.84
Walker2d at 100K	501	1215	1810	2150	4.29	1.77	1.19
HalfCheetah at 100K	3055	3594	6876	6325	2.10	1.76	0.92
Ant at 250K	2107	2344	3279	4153	1.97	1.77	1.27
Humanoid at 250K	1094	3038	4535	3973	3.63	1.31	0.88
Average at early stage	-	-	-	-	2.71	1.59	1.02

Table 4: Late-stage performance comparison of SAC, TQC, REDQ and AQE when AQE and TQC using the same hyperparameters across the environments. On average, AQE performs 16% better than SAC, 9% better than TQC and 11% times better than REDQ.

Amount of data	SAC	TQC	REDQ	AQE	AQE/SAC	AQE/TQC	AQE/REDQ
Hopper at 1M	3282	2024	2954	2404	0.73	1.19	0.81
Walker2d at 1M	4134	5532	4637	5517	1.33	1.00	1.19
HalfCheetah at 1M	10475	9792	11562	11293	1.08	1.15	0.98
Ant at 3M	5903	6186	5785	7345	1.24	1.19	1.27
Humanoid at 3M	6177	9593	6649	8680	1.41	0.91	1.31
Average at late stage	-	-	-	-	1.16	1.09	1.11

C Additional Results for AQE, TQC, REDQ and SAC in DeepMind Control Suite Benchmark

Figure **??** presents the performance of AQE, TQC, REDQ and SAC for 9 DeepMind Control Suite (DMC) environments. We can see that AQE continues to outperform TQC except for Humanoid run environment, where TQC performs better than AQE in the final stage training. AQE and REDQ have comparable results in some of the DMC environments, however, AQE usually outperforms REDQ in the more challenging environments, such as Hopper-hop, Humanoid-run and Quadruped-run. We report detailed early-stage and late-stage performance comparisons of all algorithms in Table 5 and Table 6. On average, in the early stage of training, AQE performs 13.71 times better than SAC, 7.59 times better than TQC and 1.02 times better than REDQ. In the late-stage training, on average, AQE performs 1.37 times better than SAC, 1.08 times better than TQC and 1.03 times better than REDQ.

Table 5: Early-stage performance comparison of SAC, TQC, REDQ and AQE when AQE and TQC using the same hyperparameters across the DMC environments. On average, AQE performs 13.71 times better than SAC, 7.59 times better than TQC and 1.02 times better than REDQ.

Amount of data	SAC	TQC	REDQ	AQE	AQE/SAC	AQE/TQC	AQE/REDQ
Cheetah-run at 100K	205	235	317	339	1.65	1.44	1.07
Fish-swim at 100K	121	149	234	230	1.90	1.54	0.98
Hopper-hop at 100K	2	11	50	64	32	5.81	1.28
Quadruped-walk at 100K	116	172	452	341	2.94	1.98	0.75
Quadruped-run at 100K	114	111	294	284	2.49	2.56	0.97
Walker-run at 100K	305	372	468	457	1.50	1.23	0.98
Humanoid-stand at 100K	5	5	37	52	10.4	10.4	1.41
Humanoid-walk at 100K	1	1	57	40	40	40	0.70
Humanoid-run at 250K	2	18	59	61	30.5	3.39	1.03
Average at early stage	-	-	-	-	13.71	7.59	1.02

Table 6: Late-stage performance comparison of SAC, TQC, REDQ and AQE when AQE and TQC using the same hyperparameters across the DMC environments. On average, AQE performs 1.37 times better than SAC, 1.08 times better than TQC and 1.03 times better than REDQ.

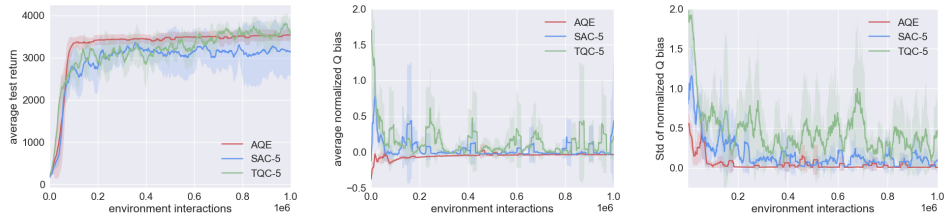
Amount of data	SAC	TQC	REDQ	AQE	AQE/SAC	AQE/TQC	AQE/REDQ
Cheetah-run at 1M	734	829	844	856	1.17	1.03	1.01
Fish-swim at 1M	639	722	753	747	1.17	1.03	0.99
Hopper-hop at 1M	293	256	279	294	1.00	1.15	1.05
Quadruped-walk at 1M	871	948	949	948	1.09	1.00	1.00
Quadruped-run at 1M	676	893	904	928	1.37	1.04	1.03
Walker-run at 1M	660	780	826	808	1.22	1.04	0.98
Humanoid-stand at 1M	323	429	547	546	1.69	1.27	1.00
Humanoid-walk at 1M	325	427	596	576	1.77	1.35	0.97
Humanoid-run at 4.5M	146	324	216	271	1.86	0.84	1.25
Average at late stage	-	-	-	-	1.37	1.08	1.03

Table 7: Sample efficiency comparison of SAC, TQC, REDQ and AQE. The numbers show the amount of data collected when the specified performance level is reached (roughly corresponding to 90% of SAC's final performance). The last three columns show how many times AQE is more sample efficient than SAC, TQC and REDQ in reaching that performance level.

Performance	SAC	TQC	REDQ	AQE	AQE/SAC	AQE/TQC	AQE/REDQ
Cheetah-run at 700	746K	440K	506K	350K	2.13	1.26	1.45
Fish-swim at 600	794K	494K	317K	417K	1.90	1.18	0.76
Hopper-hop at 250	580K	856K	451K	371K	1.56	2.31	1.22
Quadruped-walk at 800	844K	301K	302K	236K	3.58	1.28	1.28
Quadruped-run at 650	942K	521K	267K	248K	3.80	2.10	1.08
Walker-run at 600	516K	201K	156K	174K	2.97	1.16	0.90
Humanoid-stand at 250	626K	429K	279K	342K	1.83	1.25	0.82
Humanoid-walk at 300	820K	523K	279K	300K	2.73	1.74	0.93
Humanoid-run at 120	3940K	1100K	602K	603K	6.53	1.82	1.00
Average	-	-	-	-	3.00	1.57	1.05

D Additional Results for AQE, SAC-5 and TQC-5

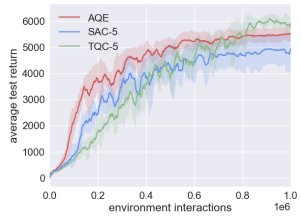
Figure 2 presents the performance of AQE, SAC-5 and TQC-5 for all the environments. SAC-5 and TQC-5 uses UTD ratio G = 5 for SAC and TQC, respectively. We can see that AQE continues to outperform both algorithms except for Humanoid, where TQC performs somewhat better than AQE in the final stage training. SAC becomes more sample efficient with $G = 5$; however, AQE still beats SAC-5 by a large margin.



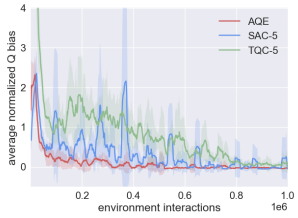
(a) Performance, Hopper

(b) Average normalized bias

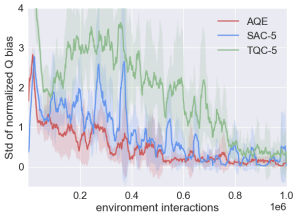
(c) Std of normalized bias



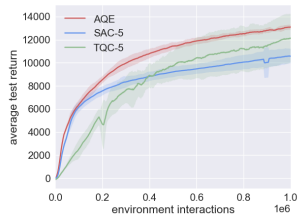
(d) Performance, Walker



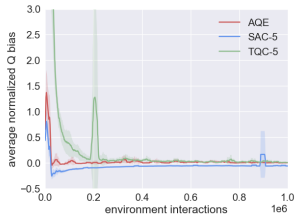
(e) Average normalized bias



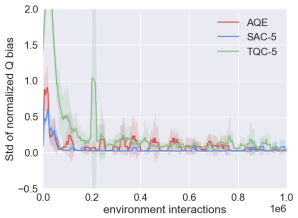
(f) Std of normalized bias



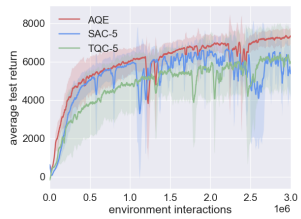
(g) Performance, HalfCheetah



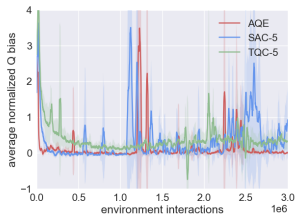
(h) Average normalized bias



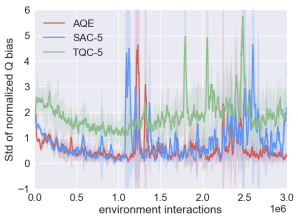
(i) Std of normalized bias



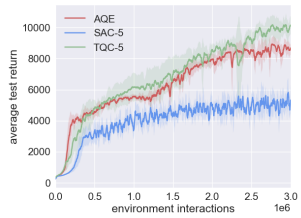
(j) Performance, Ant



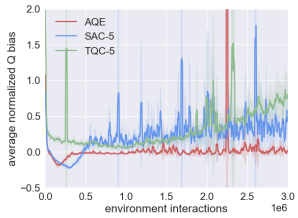
(k) Average normalized bias



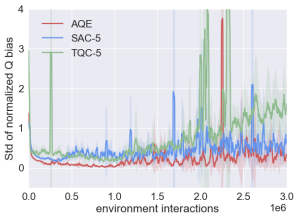
(l) Std of normalized bias



(m) Performance, Humanoid



(n) Average normalized bias

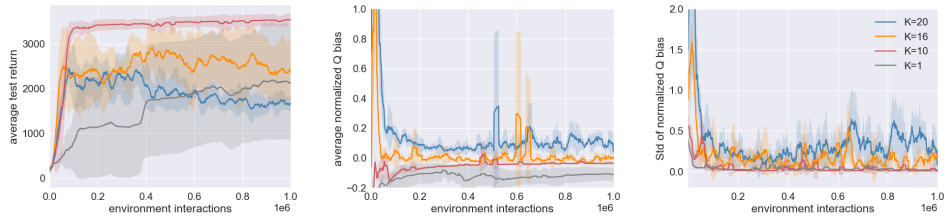


(o) Std of normalized bias

Figure 2: Performance, average and std of normalized Q bias for AQE, SAC-5 and TQC-5. All of the algorithms in this experiment use UTD = 5.

E Additional Results for parameter K

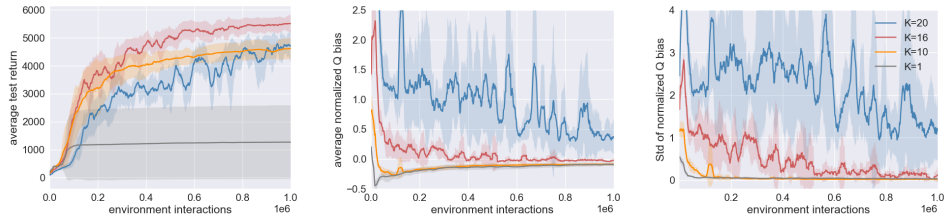
Due to lack of space, Figure **??** only compares different AQE keep numbers K for Ant. Figure 3 shows the performance, average estimation bias and standard deviation for all five environments. Consistent with the theoretical result in Theorem 1, by decreasing K, we lower the average bias.



(a) Performance, Hopper

(b) Average normalized bias

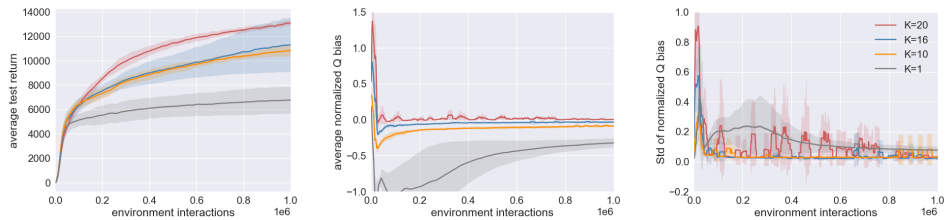
(c) Std of normalized bias



(d) Performance, Walker

(e) Average normalized bias

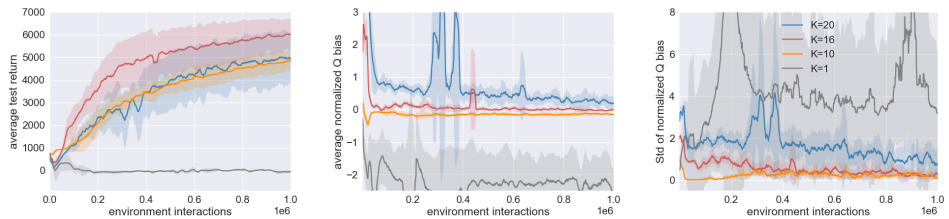
(f) Std of normalized bias



(g) Performance, HalfCheetah

(h) Average normalized bias

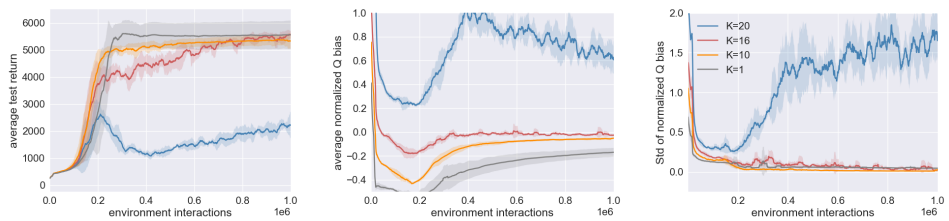
(i) Std of normalized bias



(j) Performance, Ant

(k) Average normalized bias

(l) Std of normalized bias



(m) Performance, Humanoid

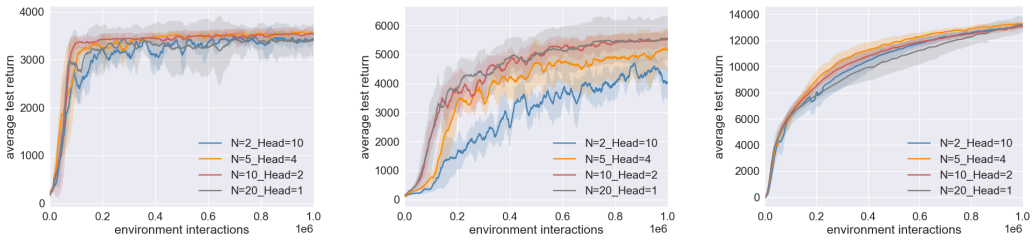
(n) Average normalized bias

(o) Std of normalized bias

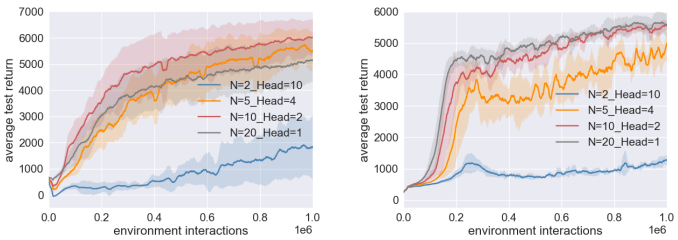
Figure 3: Performance, average and std of normalized Q bias for AQE with different values of K.

F Additional Results for Multi-head Architecture

Due to lack of space, Figure **??** only compares the different size of the ensemble N and the number of heads h for Ant. Figure 4 shows the results for all five environments. We can see that the combination of $N = 10, h = 2$ and $N = 20, h = 1$ have comparable performance. However, $N = 10$ and $h = 2$ is faster in terms of computation time.



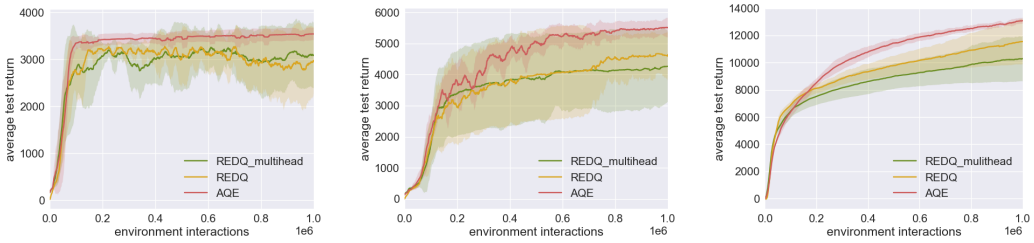
(a) Hopper-v2 (b) Walker2d-v2 (c) HalfCheetah-v2



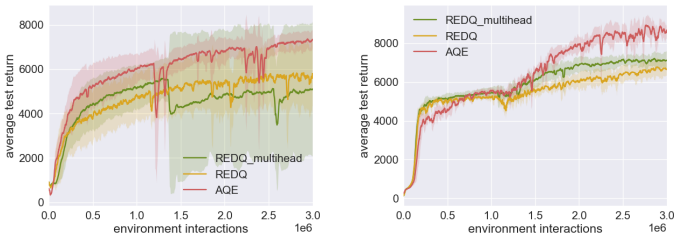
(d) Ant-v2 (e) Humanoid-v2

Figure 4: Performance for AQE with different combinations of number of Q networks and number of heads.

Will the performance of REDQ match that of AQE if we also provide REDQ a multi-head architecture? Figure 5 examines the performance of REDQ when it is endowed with the same multi-head architecture as AQE. We see that the performance of REDQ does not substantially improve.



(a) Hopper-v2 (b) Walker2d-v2 (c) HalfCheetah-v2



(d) Ant-v2 (e) Humanoid-v2

Figure 5: Performance of REDQ with N=10 and heads = 2 as compared with REDQ and AQE.

G Theoretical Results

In this section, we characterize how changing the size of the ensemble N and the keep parameter K affects the estimation bias term in the AQE algorithm. We will restrict our analysis to the tabular version of AQE shown in algorithm 1.

Our analysis will follow similar lines of reasoning as **?** and **?** which extends upon the theoretical framework introduced in **?**.

For each $a \in \mathcal{A}$, let $E_{K,N}(s,a)$ be the ensemble members in $\{1,\dots,N\}$ with the K lowest values of $Q^j(s,a)$, $j = 1,\dots,N$. In the tabular case, the target for the Q networks take the form:

$$r + \gamma \max_{a'} \left(\frac{1}{K} \sum_{j \in E_{K,N}(s',a')} Q^j(s',a') \right). \tag{1}$$

Define the *post-update estimation bias* as

$$\begin{aligned} Z_{K,N} &:= r + \gamma \max_{a'} \left(\frac{1}{K} \sum_{j \in E_{K,N}(s',a')} Q^j(s',a') \right) - \left(r + \gamma \max_{a'} Q^\pi(s',a') \right) \\ &= \gamma \left[\max_{a'} \left(\frac{1}{K} \sum_{j \in E_{K,N}(s',a')} Q^j(s',a') \right) - \max_{a'} Q^\pi(s',a') \right] \end{aligned} \tag{2}$$

Under this definition, if $\mathbb{E}[Z_{K,N}] > 0$, then the expected post-update estimation bias is positive and there is a tendency for the positive bias to accumulate during updates. Similarly, if $\mathbb{E}[Z_{K,N}] < 0$, then the expected post-update estimation bias is negative and there is a tendency for the negative bias to accumulate during updates. Ideally, we would like $\mathbb{E}[Z_{K,N}] \approx 0$

Also let

$$Q^j(s,a) = Q^\pi(s,a) + e^j(s,a) \tag{3}$$

where $e^j(s,a)$ is an independent and identically distributed error term across all j's and all a's for each fixed s. We further assume that $\mathbb{E}[e^j(s,a)] = 0$. Note that with this assumption

$$\mathbb{E}\left[\frac{1}{N} \sum_{j=1}^{N} Q^j(s,a) \right] - Q^\pi(s,a) = 0,$$

that is the pre-update estimation bias is zero. The following theorem shows how the expected estimation bias changes with N and K:

Theorem 1. *The following results hold for* $\mathbb{E}[Z_{K,N}]$*:*

1. $\mathbb{E}[Z_{N,N}] \geq 0$ *for all* $N \geq 1$.
2. $\mathbb{E}[Z_{K-1,N}] \leq \mathbb{E}[Z_{K,N}]$ *for all* $K \leq N$.
3. $\mathbb{E}[Z_{K,N+1}] \leq \mathbb{E}[Z_{K,N}]$.
4. *Suppose that* $e^j_{sa} \leq c$ *for some* $c > 0$ *for all* s, a *and* j*. Then there exists an* N *sufficiently large and* $K < N$ *such that* $\mathbb{E}[Z_{K,N}] < 0$.

Proof Sketch. Part 1 is a result of Jensen's Inequality, and Parts 2 and 3 can be shown by analyzing how the average of the K smallest ensembles changes when adding an extra ensemble model. Given the first three parts, we only need to show that $\mathbb{E}[Z_{1,N}] < 0$ to show that there exists a K for a sufficiently large N where the expected bias is negative. See the next section for full proof. □

Theorem 1 shows that we can control the expected post-update bias $\mathbb{E}[Z_{K,N}]$ through adjusting K and N. More concretely, we can bring the bias term from above zero (i.e. over estimation) to under zero (i.e. under estimation) by decreasing K and/or increasing N.

We note also that similar to **?**, we make very few assumptions on the error term $e_{s,a}$. This is in contrary to **?** and **?**, both of whom assume that the error term is uniformly distributed.

G.1 Tabular AQE with N ensemble members and d drops

Algorithm 1 Tabular AQE

Initialize: $Q^j(s,a)$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}$, $j = 1, \ldots, N$.

1: **repeat**

2: For some state $s \in \mathcal{S}$, choose $a \in \mathcal{A}$ based on $\left\{Q^j(s,a)\right\}_{j=1}^{N}$, observe r, s'.

3: For each $a' \in \mathcal{A}$, let $E_{K,N}(s',a')$ be the ensemble members in $\{1,\ldots,N\}$ with the K lowest values of $Q^j(s',a')$, $j = 1, \ldots, N$.

4: Get target

$$y = r + \gamma \max_{a' \in \mathcal{A}} \frac{1}{K} \sum_{j \in E_{K,N}(s',a')} Q^j(s',a')$$

5: **for** $j = 1, \ldots, N$ **do**

6: Update each $Q^j(s,a)$

$$Q^j(s,a) \leftarrow Q^j(s,a) + \alpha(y - Q^j(s,a))$$

7: $s \leftarrow s'$

8: **until** end

H Proofs

We first present the following lemma:

Lemma 1 (?). *Let $X_1, X_2, \ldots$ be an infinite sequence of i.i.d. random variables with cdf $F(x)$ and let $\tau = \inf x : F(x) > 0$. Also let $Y_N = \min\{X_1, X_2, \ldots, X_N\}$. Then $Y_1, Y_2, \ldots$ converges to τ almost surely.*

Proof. See Appendix A.2 of **?** □

Theorem 1. *The following results hold for $\mathbb{E}[Z_{K,N}]$:*

1. *$\mathbb{E}[Z_{N,N}] \geq 0$ for all $N \geq 1$.*
2. *$\mathbb{E}[Z_{K-1,N}] \leq \mathbb{E}[Z_{K,N}]$ for all $K \leq N$.*
3. *$\mathbb{E}[Z_{K,N+1}] \leq \mathbb{E}[Z_{K,N}]$.*
4. *Suppose that $e^j_{sa} \leq c$ for some $c > 0$ for all s, a and j. Then there exists an N sufficiently large and $K < N$ such that $\mathbb{E}[Z_{K,N}] < 0$.*

Proof. 1. By definition,

$$\begin{aligned}\mathbb{E}[Z_{N,N}] &= \gamma\,\mathbb{E}\left[\max_{a'}\left(\frac{1}{N}\sum_{j=1}^{N} Q^j(s',a')\right) - \max_{a'} Q^\pi(s',a')\right] \\ &\geq \gamma\left[\max_{a'} E\left[\left(\frac{1}{N}\sum_{j=1}^{N} Q^j(s',a')\right)\right] - \max_{a'} Q^\pi(s',a')\right] \\ &= \gamma\left[\max_{a'} Q^\pi(s',a') - \max_{a'} Q^\pi(s',a')\right] = 0\end{aligned} \tag{4}$$

2. Let

$$\bar{Q}_{K,N}(s,a) = \frac{1}{K}\sum_{j\in E_{K,N}} Q^j(s,a). \tag{5}$$

Since for any state s, $\max_a \bar{Q}_{K+1,N}(s,a) \geq \max_a \bar{Q}_{K,N}(s,a)$,

$$\begin{aligned}\mathbb{E}[Z_{K+1,N}] &= \gamma\,\mathbb{E}\left[\max_{a'}\bar{Q}_{K+1,N}(s',a') - \max_{a'} Q^\pi(s',a')\right] \\ &\geq \gamma\,\mathbb{E}\left[\max_{a'}\bar{Q}_{K,N}(s',a') - \max_{a'} Q^\pi(s',a')\right] \\ &= \mathbb{E}[Z_{K,N}]\end{aligned} \tag{6}$$

3. Comparing $\mathbb{E}[Z_{K,N}]$ and $\mathbb{E}[Z_{K,N+1}]$ is equivalent to comparing $\bar{Q}_{K,N}(s,a)$ and $\bar{Q}_{K,N+1}(s,a)$. Since $e^j(s,a)$ is i.i.d., by extension $Q^j(s,a)$ is also i.i.d. for $j = 1, 2, \cdots$. Suppose $Q^j(s,a)$ is drawn from some probability distribution F, then given $\bar{Q}_{K,N}(s,a)$, $\bar{Q}_{K,N+1}(s,a)$ can be calculated by generating an additional $Q^i(s,a)$ from F. The new sample $Q^i(s,a)$ affects the calculation of $\bar{Q}_{K,N+1}(s,a)$ under the following two cases:

   - If $Q^i(s,a) > \max_{j\in E_{K,N}} Q^j(s,a)$, then the lowest K values remain unchanged hence $\bar{Q}_{K,N}(s,a) = \bar{Q}_{K,N+1}(s,a)$.
   - Else if $Q^i(s,a) \leq \max_{j\in E_{K,N}} Q^j(s,a)$, then $\max_{j\in E_{K,N}} Q^j(s,a)$ would be removed from and $Q^i(s,a)$ would be added to the set of lowest K values, therefore $\bar{Q}_{K,N+1}(s,a) \leq \bar{Q}_{K,N}(s,a)$.

   Combining the two cases $\bar{Q}_{K,N+1}(s,a) \leq \bar{Q}_{K,N}(s,a)$, therefore $\mathbb{E}[Z_{K,N+1}] \leq \mathbb{E}[Z_{K,N}]$

4. Since $\mathbb{E}[Z_{N,N}] \geq 0$, $\mathbb{E}[Z_{K,N}] \leq \mathbb{E}[Z_{K+1,N}]$ and $\mathbb{E}[Z_{K,N+1}] \leq \mathbb{E}[Z_{K,N}]$. It is suffice to show that $\mathbb{E}[Z_{1,N}] < 0$ for some N. The rest of the proof largely follows Theorem 1 of **?**.

   Let $\tau = \inf\{x : F_a(x) > 0\}$ where $F_a(x)$ is the cdf of $Q^j(s,a)$, $j = 1, 2, \ldots$. By Lemma 1, $\bar{Q}_{1,N}(s,a) = \min_{1 \leq j \leq N} Q^j(s,a)$ converges almost surely to to τ_a for each a. Since the action space is finite, it then follows that $\max_a \bar{Q}_{1,N}(s,a)$ converges almost surely to to $\tau = \max_a \tau_a$. Due to our assumption that $e^j(s,a) \leq c$ and that $Q^\pi(s,a)$ is finite, it then follows that $\max_a \bar{Q}_{1,N}(s,a)$ is also bounded above. By Part 3 of the theorem, $\bar{Q}_{1,N}(s,a)$ is monotonoically decreasing w.r.t. N. and since $\max_a \bar{Q}_{1,N}(s,a)$ is also bounded above and converges almost surely to τ, we have

$$\begin{aligned}\mathbb{E}[Z_{1,N}] &= \gamma \left(\mathbb{E}[\max_a \min_{1 \leq j \leq N} Q^j(s,a)] - \max_a Q^\pi(s,a) \right) \\ &= \gamma \left(\mathbb{E}[\max_a Y_a^N] - \max_a Q^\pi(s,a) \right) \overset{N \to \infty}{\longrightarrow} \gamma \left(\max_a \tau_a - \max_a Q^\pi(s,a) \right) < 0 \end{aligned} \tag{7}$$

   where the last equality follows from the assumption that the error $e^j(s,a)$ is non-trivial, and hence $\tau_a < \max_a Q^\pi(s,a)$ for all a. Therefore for a sufficiently large N, there exists a $1 \leq K \leq N$ such that $\mathbb{E}_{K,N} < 0$.

□

I Computing Infrastructure

Each experiment is run on a single Nvidia 2080-Ti GPU with CentOS Linux System.