We thank the reviewers for the detailed comments, suggestions, and a positive assessment of our work. In the final version of our paper, we shall clarify the details in Section 3 (**R2**), and make intuition in the methods section much clearer. We will correct for color schemes in all figures **(R1)**. We have also made captions of figures cleaner **(R3)**.

R2: concerns regarding Figure 3. We have added a description of the setup to the paper. Unif(s, a) is an oracle distribution where every single (s, a) tuple in the MDP appears in the buffer, exactly the same number of times. This is of course not achievable in practice, since this data is collected by the policy which might not produce a uniform distribution and enumerating all state-action pairs in a continuous state (and/or action) MDPs is not possible. This figure simply argues that for some "oracle" distributions (such as Unif(s, a)), the performance and error reduction in Q-learning can be much better than the on-policy distribution while retaining the same function approximator and other details, which provides some evidence that the on-policy distribution is not necessarily optimal in regard to error reduction with function approximation. On the other hand, without function approximation, on-policy distribution also performs well (as shown in Fig 3, "Tabular"). That said, Unif(s, a) is just one (arbitrary) example of a distribution that performs better than on-policy data. In Fig 5 (left), DisCor actually outperforms Unif(s, a) on these environments.

R2: Intuitive explanation of Theorem 4.1. We have now added a more intuitive discussion for this theorem in the paper. Intuitively, the optimal distribution assigns higher probability to state-action pairs with high Bellman error $|Q_k - \mathcal{B}^* Q_{k-1}|$, but only when the overall error $|Q_k - Q^*|$ is minimized. This amounts to minimizing Bellman error only if the resulting Q-function is going to be closer to Q^*. Our tractable approximation (in Sec. 4.2) uses an estimated error in the target values using Δ to identify if the resulting Q-function will be closer to Q^*. So, intuitively this optimal distribution up-weights state-action tuples with correct target values, agnostic of the distribution of past policies.

R2: relation to negative bias in inference on bandit data. Thank you for pointing us to this literature. We will cite these papers in the final. These prior works demonstrate how statistical sampling error can induce negative bias in policy evaluation in bandits. However, the corrective feedback problem we describe is intimately tied to the "bootstrapping" in ADP updates (not present in bandits) and how on-policy data distributions with function approximation may not be effective in correcting errors. We will discuss this issue and the connection with statistical error in the paper.

R3: Lack of corrective feedback in tabular RL. The issue of absent corrective feedback may indeed arise in tabular settings with few samples, and we will expand on this discussion in the paper. That said, DisCor is primarily focused on the case with function approximation, where this problem is particularly exacerbated, as shown in the tree MDP example (Figure 1) and also occurs in gridworld MDPs (Figure 2 and 3).

R5: High variance in meta-world. The reason for high variance on some tasks is likely because learning in different runs picked up at different times, which is probably because these tasks are especially hard to learn from (as also seen in the high variance in standard SAC runs). The new runtime with Δ is 1.3-1.4x of regular SAC.

R2 and R3: Exact source of problem with function approximation. To address this, we will add a simple computational example that illustrates that, even in a simple MDP, error can increase with standard Q-learning but decreases with DisCor. **Example:** Our example is a 5-state MDP, with the starting state s_0 and the terminal state s_T (marked in gray). Each state has two available actions, a_0 and a_1, and each action deterministically transits the agent to a state marked by arrows in Figure 1. A reward of 0.001 is received only when action a_0 is chosen at state s_3 (else reward is 0). The Q-function is a linear function over pre-defined features $\phi(s, a)$, i.e., $Q(s, a) = [w_1, w_2]^T \phi(s, a)$, where $\phi(\cdot, a_0) = [1, 1]$ and $\phi(\cdot, a_1) = [1, 1.001]$ (hence features are aliased across states). Computationally, we see that when minimizing Bellman error starting from a Q-function with weights $[w_1, w_2] = [0, \text{1e-4}]$, under the on-policy distribution of the Boltzmann policy, $\pi(a_0|\cdot) = 0.001, \pi(a_1|\cdot) = 0.999$, in the absence of sampling error (using all transitions but weighted), the error against Q^* still **increases** from 7.177e-3 to 7.179e-3 in one iteration, whereas with DisCor error **decreases** to **5.061e-4**. With uniform the error also decreases, but is larger: 4.776e-3.

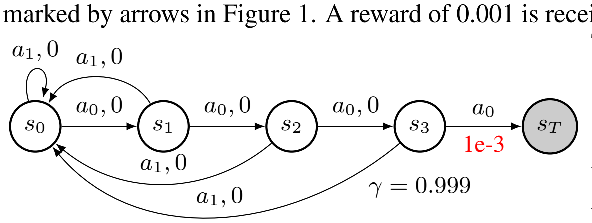

Figure 1: A simple MDP showing the effect of on-policy distribution and function approximation on learning dynamics of ADP algorithms.

<u>Intuition for the example</u>: The Q-function value error at state-action pairs that will be used as bootstrapping targets for other state-action tuples ($Q(s_0, a_1)$ is used as target for all states with action a_1) is high and the state-action pair with correct target value, (s_3, a_0), appears infrequently in the on-policy distribution, since the policy chooses the other action a_1 with high probability. Sine the function approximator couples together updates across states and actions, this infrequency of update at (s_3, a_0) and higher frequency of state-action tuples with incorrect targets will update the Q-function approximator towards increasing value error. Thus, minimizing Bellman error can lead to an increase in the error to Q^* (Also shown in **Fig. 2** on a gridworld). We can further generalize this discussion over multiple iterations of learning. This example is a computational version of the tree MDP shown in Figure 1 of the paper **[R2, R5]**.