ON THE CONVERGENCE OF THE MONTE CARLO EXPLORING STARTS ALGORITHM FOR REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

A simple and natural algorithm for reinforcement learning (RL) is Monte Carlo Exploring Starts (MCES), where the Q-function is estimated by averaging the Monte Carlo returns, and the policy is improved by choosing actions that maximize the current estimate of the Q-function. Exploration is performed by "exploring starts", that is, each episode begins with a randomly chosen state and action, and then follows the current policy to the terminal state. In the classic book on RL by Sutton & Barto (2018), it is stated that establishing convergence for the MCES algorithm is one of the most important remaining open theoretical problems in RL. However, the convergence question for MCES turns out to be quite nuanced. Bertsekas & Tsitsiklis (1996) provide a counter-example showing that the MCES algorithm does not necessarily converge. Tsitsiklis (2002) further shows that if the original MCES algorithm is *modified* so that the Q-function estimates are updated at the same rate for all state-action pairs, and the discount factor is strictly less than one, then the MCES algorithm converges. In this paper we make headway with the original and more efficient MCES algorithm given in Sutton & Barto (1998), establishing almost sure convergence for Optimal Policy Feed-Forward MDPs, which are MDPs whose states are not revisited within any episode when using an optimal policy. Such MDPs include a large class of environments such as all deterministic environments and all episodic environments with a timestep or any monotonically changing values as part of the state. Different from the previous proofs using stochastic approximations, we introduce a novel inductive approach, which is very simple and only makes use of the strong law of large numbers.

1 INTRODUCTION

Perhaps the most famous algorithm in tabular reinforcement learning is the so-called Q-learning algorithm. Under very general conditions, it is well known that the Q-learning converges to the optimal Q-function with probability one (Tsitsiklis, 1994; Jaakkola et al., 1994). Importantly, in order to guarantee convergence for Q-learning, it is only required that every state-action pair be visited infinitely often. Furthermore, as discussed in the related work, Q-learning converges for the infinite-horizon discounted problem as well as for the non-discounted terminal-state problem (also known as the stochastic shortest-path problem).

The Q-learning algorithm is inspired by dynamic programming and uses back-ups to update the estimates of the optimal Q-function. An alternative methodological approach, which does not use back-ups, is to use the Monte Carlo episodic returns to estimate the values of the Q-function. In order for such an algorithm to succeed at finding an optimal policy, the algorithm must include some form of exploration. A simple form of exploration is "exploring starts," where at the beginning of each episode, a random state-action pair is chosen. In the classic book on reinforcement learning by Sutton & Barto (2018), the authors describe such an algorithm, namely, Monte Carlo Exploring Starts (MCES). In the MCES algorithm, after a (random-length) episode, the Q-function estimate is updated with the Monte Carlo return for each state-action pair along the episode, and the policy is improved in the usual fashion by setting it to the argmax of the current Q-function estimate. Exploration is performed by exploring starts, where the initial state-action pairs may be chosen with any distribution.

We briefly remark here that AlphaZero is a Monte Carlo algorithm in that it runs episodes to completion and uses the returns from those episodes for the targets in the loss function (Silver et al., 2018). AlphaZero additionally uses function approximators (convolutional networks) and planning (Monte Carlo Tree Search), and is thus much more complicated than MCES. But AlphaZero is nonetheless fundamentally a Monte Carlo algorithm rather than a Q-learning-based algorithm. We mention AlphaZero here in order to emphasize that Monte Carlo algorithms are indeed used in practice, and it is therefore important to gain a deep understanding of their underlying theoretical properties.

Since Q-learning converges under very general conditions, a natural question is: does MCES converge under equally general conditions? In the 1996 book, Bertsekas & Tsitsiklis (1996) provide a counter-example showing that the MCES algorithm does not necessarily converge. See also Liu (2020) for numerical results in this direction. Thus, we see that the MCES convergence problem is fundamentally trickier than the Q-learning convergence problem. Instead of establishing a very general result as in Q-learning, we can at best establish convergence for a broad class of special-cases.

Sutton and Barto write at the end of Section 5.3: "In our opinion, this is one of the most fundamental open theoretical questions in reinforcement learning". This paper is focused on this fundamental question. Although other questions, such as rates of convergence and regret bounds, are also important, in this paper our goal is to address the fundamental question of convergence.

Tsitsiklis (2002) made significant progress with the MCES convergence problem, showing that almost sure convergence is guaranteed if the following three conditions hold: (i) the discount factor is strictly less than one; (ii) the MCES algorithm is modified so that after an episode, the Q-function estimate is updated with the Monte Carlo return *only for the initial state-action pair of the episode*; and (iii) the algorithm is further modified so that the initial state-action pair in an episode is chosen with a uniform distribution. As in the proof of Q-learning, Tsitsiklis's proof is based on stochastic approximations. The conditions (ii) and (iii) combined ensure that the Q function estimates are updated at the same average rate for all state-action pairs, and both conditions appear to be crucial for establishing convergence in the proof in Tsitsiklis (2002). However, these two conditions have the following drawbacks:

- Perhaps most importantly, condition (ii) results in a substantially less efficient algorithm, since only one Q-function value is updated per episode. The original Sutton and Barto version is more efficient since after each episode, many Q-values are typically updated rather than just one. (We also note as an aside that AlphaZero will also collect and use Monte Carlo return for all states along the episode, not just for the first state in the episode, as discussed on page 2 of Silver et al. (2017))
- Similar to the idea of importance sampling, one may want to use a non-uniform distribution for the starting state-action pairs to accelerate convergence.
- In some cases, we may not have access to a simulator to generate uniform exploring starts. Instead, we may run episodes by interacting directly with the real environment. Such natural interactions may lead to starting from every state, but not uniformly. An example would be playing blackjack at a casino rather than training with a simulator.

In this paper we provide new convergence results and a new proof methodology for MCES. Unlike the result in Tsitsiklis (2002), the results reported here do not modify the original MCES algorithm and do not require any of the conditions $(i) - (iii)$. Hence, our results do not have the three drawbacks listed above, and also allow for no discounting (as in the stochastic shortest path problem). However, our proofs require restrictions on the dynamics of the underlying MDP. Specifically, we require that under the optimal policy, a state is never revisited. This class of MDPs includes stochastic feed-forward environments such as Blackjack (Sutton & Barto, 2018) and also all deterministic MDPs, such as gridworlds (Sutton & Barto, 2018), Go and Chess (when played against a fixed opponent policy), and the MuJoCo environments (Todorov et al., 2012) (Episodic MuJoCo tasks fall into the category of OPFF MDPs because the MuJoCo simulation is deterministic). Moreover, if the trajectory horizon is instead fixed and deterministic, we show that the original MCES algorithm *always* converges (to a time-dependent) optimal policy, without any conditions on the dynamics, initial state-action distribution or the discount factor.

Importantly, we also provide a new proof methodology. Our proof is very simple, making use of only the Strong Law of Large Numbers (SLLN) and a simple inductive argument. The proof does

not use stochastic approximations, contraction mappings, or martingales, and can be done in an undergraduate course in machine learning. We believe that this new proof methodology provides new insights for episodic RL problems.

In addition to the theoretical results, we present numerical experiments that show the original MCES can be much more efficient than the modified MCES, further highlighting the importance of improving our understanding on the convergence properties of the original MCES algorithm.

2 Related Work

Some authors refer to an MDP with a finite horizon H as an episodic MDP. For finite horizon MDPs, the optimal Q-function and optimal policy are in general non-stationary and depend on time. Here, following Sutton & Barto (2018), we instead reserve the term *episodic MDPs* for MDPs that terminate when the terminal state is reached, and thus the episode length is not fixed at H and may have a random length. Moreover, for such terminal-state episodic MDPs, under very general conditions, the optimal Q-function and policy are stationary and do not depend on time (as in infinite-horizon discounted MDPs). When the dynamics are known and the discount factor equals 1, the episodic optimization problem considered here is equivalent to the stochastic shortest path problem (SSPP) (see Bertsekas & Tsitsiklis (1991) and references therein; also see Chapter 2 of Bertsekas (2012)). Under very general conditions, value iteration converges to the optimal value function, from which an optimal stationary policy can be constructed.

Convergence theory for RL algorithms has a long history. For the infinite-horizon discounted criterion, by showing that Q-learning is a form of stochastic approximations, Tsitsiklis (1994) and Jaakkola et al. (1994) showed that Q-learning converges almost surely to the optimal Q-function under very general conditions. There are also convergence results for Q-learning applied to episodic MDPs as defined in this paper with discount factor equal to 1. Tsitsiklis [8, Theorems 2 and 4(c)] proved that if the sequence of Q-learning iterates is bounded, then Q-learning converges to the optimal Q values almost surely. Yu & Bertsekas (2013) prove that the sequence of Q-learning iterates is bounded for episodic MDPs with or without non-negativity assumptions, fully establishing the convergence of Q-learning for terminal-state episodic RL problems.

This paper is primarily concerned with the convergence of the MCES algorithm. In the Introduction we reviewed the important work of Sutton & Barto (1998), Bertsekas & Tsitsiklis (1996), and Tsitsiklis (2002). Importantly, unlike Q-learning, the MCES algorithm is not guaranteed to converge for all types of MDPs. Indeed, in Section 5.4 of Bertsekas & Tsitsiklis (1996), Example 5.12 shows that MCES is not guaranteed to converge for a continuing task MDP. However, if the algorithm is modified, as described in the Introduction, then convergence is guaranteed (Tsitsiklis, 2002). Recently, Chen (2018) extended the convergence result in Tsitsiklis (2002) to the undiscounted case, under the assumption that all policies are proper, that is, regardless of the initial state, all policies will lead to a terminal state in finite time with probability one. More recently, Liu (2020) relaxed the all policies being proper condition. As in Tsitsiklis (2002), both Chen (2018) and Liu (2020) assume conditions $(ii) - (iii)$ stated in the introduction, and their proofs employ the stochastic approximations methodology in Tsitsiklis (1994). The results we develop here are complementary to the results in Tsitsiklis (2002), Chen (2018), and Liu (2020), in that they do not require the strong algorithmic assumptions $(ii) - (iii)$ described in the Introduction, and they use an entirely different proof methodology.

In this work we focus on the question of convergence of the MCES problem. We briefly mention, there is also a large body of (mostly orthogonal) work on rates of convergence and regret analysis for Q-learning (e.g. see Jin et al. (2018)) and also for Monte Carlo approaches (e.g., see Kocsis & Szepesvári (2006) Azar et al. (2017)). To the best of our knowledge, these regret bounds assume finite-horizon MDPs (for which the optimal policy is time-dependent) rather than the terminal-state episodic MDPs considered here.

3 MDP Formulation

Following the notation of Sutton & Barto (2018), a finite Markov decision process is defined by a finite state space $\mathcal{S}$ and a finite action space $\mathcal{A}$, reward function $r(s, a)$ mapping $\mathcal{S} \times \mathcal{A}$ to the reals,

and a dynamics function $p(\cdot|s,a)$, which for every $s \in \mathcal{S}$ and $a \in \mathcal{A}$ gives a probability distribution over the state space $\mathcal{S}$.

A (deterministic and stationary) **policy** π is a mapping from the state space $\mathcal{S}$ to the action space $\mathcal{A}$. We denote $\pi(s)$ for the action selected under policy π when in state s. Denote S_t^π for the state at time t under policy π. Given any policy π, the state evolution becomes a well-defined Markov chain with transition probabilities

$$P(S_{t+1}^\pi = s' | S_t^\pi = s) = p(s'|s, \pi(s))$$

3.1 Return for Random-Length Episodes

As indicated in Chapters 4 and 5 of Sutton & Barto (2018), for RL algorithms based on Monte Carlo methods, we assume that the task is episodic, that is "experience is divided into episodes, and all episodes eventually terminate no matter what actions are selected." Examples of episodic tasks include "plays of a game, trips through a maze, or any sort of repeated interaction". Chapter 4 of Sutton & Barto (2018) further states: "Each episode ends in a special state called the *terminal state*, followed by a reset to a standard starting state or to a sample from a standard distribution of starting states".

The "Cliff Walking" example in Sutton & Barto (2018) is an example of an "episodic MDP". Here the terminal state is the union of the goal state and the cliff state. Although the terminal state will not be reached by all policies due to possible cycling, it will clearly be reached by the optimal policy. Another example of an episodic MDP, discussed at length in Sutton and Barto, is "Blackjack". Here we can create a terminal state which is entered whenever the player sticks or goes bust. For Blackjack, the terminal state will be reached by all policies. Let $\tilde{s}$ denote the terminal state. (If there are multiple terminal states, without loss in generality they can be lumped into one state.)

When using policy π to generate an episode, let

$$T^\pi = \min\{\ t\ :\ S_t^\pi = \tilde{s}\} \tag{1}$$

be the time when the episode ends. The expected total reward when starting in state s is

$$v_\pi(s) = E[\sum_{t=0}^{T^\pi} r(S_t^\pi, \pi(S_t^\pi)) | S_0^\pi = s] \tag{2}$$

Maximizing (2) corresponds to the stochastic shortest path problem (as defined in Bertsekas & Tsitsiklis (1991); Bertsekas (2012)), for which there exists an optimal policy that is both stationary and deterministic (for example, see Proposition 2.2 of Bertsekas (2012)).

At the end of this paper we will also consider the finite horizon problem of maximizing:

$$v_\pi^H(s) = E[\sum_{t=0}^{H} r(S_t^\pi, \pi(S_t^\pi, t)) | S_0^\pi = s] \tag{3}$$

where H is a fixed and given horizon. For this criterion, it is well-known that optimal policy $\pi(s) = (\pi(s,1), \ldots, \pi(s,H))$ is non-stationary. For the finite-horizon problem, it is not required that the MDP have a terminal state.

3.2 Classes of MDPs

We will prove convergence results for important classes of MDPs. For any MDP, define the MDP graph as follows: the nodes of the graph are the MDP states; there is a directed link from state s to s' if there exists an action a such that $p(s'|s,a) > 0$.

We say an environment is **Stochastic Feed-Forward (SFF)** if a state cannot be revisited within any episode. More precisely, the MDP is SFF if its MDP graph has no cycles. Note that transitions are permitted to be stochastic. SFF environments occur naturally in practice. For example, the Blackjack environment, studied in detail in Sutton & Barto (2018), is SFF. We say an environment is **Optimal Policy Feed-Forward (OPFF)** if under any optimal policy a state is never re-visited. More precisely, construct a sub-graph of the MDP graph as follows: each state s is a node in the

graph, and there is a directed edge from node s to s' if $p(s'|s, a^*) > 0$ for some optimal action a^*. The MDP is OPFF if this sub-graph is acyclic.

An environment is said to be a **deterministic environment** if for any state s and chosen action a, the reward and subsequent state s' are given by two (unknown) deterministic functions $r = r(s, a)$ and $s' = g(s, a)$. Many natural environments are deterministic. For example, in Sutton & Barto (2018), environments Tic-Tac-Toe, Gridworld, Golf, Windy Gridworld, and Cliff Walking are all deterministic. Moreover, many natural environments with continuous state and action spaces are deterministic, such as the Mujoco robotic locomotion environments (Todorov et al., 2012).

It is easily seen that all SFF MDPs are OPFF, and all deterministic MDPs for which the optimal policy terminates w.p.1 are OPFF.

4 Monte Carlo with Exploring Starts

The MCES algorithm is given in Algorithm 1. The MCES algorithm is a very natural and simple algorithm, which uses only Monte Carlo returns and no backups for updating the Q-function and training the policy. A natural question is: does it converge to the optimal policy? As mentioned in the Introduction, unlike for Q-learning, it does not converge for general MDPs. We instead establish convergence for important classes of MDPs.

Algorithm 1 is consistent with the MCES algorithm in Sutton & Barto (2018) and has the following important features:

- $Q(S_t, A_t)$ is updated for every S_t, A_t pair along the episode, and not just for S_0, A_0 as required in (Tsitsiklis, 2002).
- The initial state and action in an episode can be chosen arbitrarily (but infinitely often), and does not have to be chosen according to a uniform distribution as required in (Tsitsiklis, 2002).
- For simplicity, we present the algorithm with no discounting (i.e., $\gamma = 1$). However, the subsequent proofs go through for any discount factor $\gamma \leq 1$. (The proof in Tsitsiklis (2002) requires $\gamma < 1$.)

We emphasize that although the results in Tsitsiklis (2002) require restrictive assumptions on the algorithm, the results in Tsitsiklis (2002) hold for general MDPs. Our results do not make restrictive algorithmic assumptions, but only hold for a sub-class of MDPs.

Algorithm 1 MCES

Initialize: $\pi(s) \in \mathcal{A}$, $Q(s, a) \in \mathbb{R}$, for all $s \in \mathcal{S}, a \in \mathcal{A}$, arbitrarily;
$Returns(s, a) \leftarrow$ empty list, for all $s \in \mathcal{S}, a \in \mathcal{A}$.
while True **do**
  Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}$ s.t. all pairs are chosen infinitely often.
  Generate an episode following π: $S_0, A_0, S_1, A_1, \ldots, S_{T-1}, A_{T-1}, S_T$.
  $G \leftarrow 0$
  for $t = T-1, T-2, \ldots, 0$ **do**
    $G \leftarrow G + r(S_t, A_t)$
    Append G to $Returns(S_t, A_t)$
    $Q(S_t, A_t) \leftarrow average(Returns(S_t, A_t))$
    $\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

We begin with the following lemma, whose proof can be found in Appendix A.

Lemma 1. *Let $X_1, X_2, \ldots$ be a sequence of random variables. Let T be a random variable taking values in the positive integers, and suppose $P(T < \infty) = 1$. Suppose that for every positive integer n, $X_n, X_{n+1}, \ldots$ are i.i.d. with finite mean x^* and finite variance when conditioned on $T = n$. Then $P(\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} X_i = x^*) = 1$.*

Say that the MCES algorithm begins iteration u at the uth time that an episode runs, and ends the iteration after Q and π have been updated (basically an iteration in the while loop in Algorithm 1).

Denote by $Q_u(s,a)$ and $\pi_u(s)$ for the values of $Q(s,a)$ and $\pi(s)$ at the end of the uth iteration. For simplicity, assume that the MDP has a unique optimal policy π^*, and denote $q^*(s,a)$ for the corresponding action-value function. Note that the proof extends easily to the case of multiple optimal policies.

We now state and prove the convergence result for SFF MDPs. We provide the full proof here in order to highlight how simple it is. Afterwards we will state the more general result for OPFF MDPs, for which the proof is a little more complicated,

Theorem 1. *Suppose the MDP is SFF. Then $Q_u(s,a)$ converges to $q^*(s,a)$ and $\pi_u(s)$ converges to $\pi^*(s)$ for all $s \in \mathcal{S}$ and all $a \in \mathcal{A}$ w.p.1.*

Proof. Because the MDP is SFF, its MDP graph is a Directed Acyclic Graph (DAG). So we can re-order the N states such that from state s_k and selecting any action, we can only transition to a state in $\{s_{k+1}, \ldots, s_N\}$. Note state s_N is the terminal state, and note that from state s_{N-1} all actions lead to s_N.

The proof is by backward induction. The result is trivially true for $s = s_N$. Suppose it is true for all states in $\{s_{k+1}, \ldots, s_N\}$ and all actions $a \in \mathcal{A}$. We now show it is true for s_k and all actions $a \in \mathcal{A}$.

We first establish $Q_u(s_k,a)$ converges to $q^*(s_k,a)$ for all $a \in \mathcal{A}$ w.p.1. Let T be the iteration u when $\pi_u(s)$ has converged to $\pi_*(s)$ for all $s \in \{s_{k+1}, \ldots, s_N\}$. By the inductive assumption, $P(T < \infty) = 1$. Let $a \in \mathcal{A}$ be any action. Now consider any episode after time T in which we visit state s_k and choose action a. Because the MDP is SFF, the next state will be in $\{s_{k+1}, \ldots, s_N\}$, and because of the inductive assumption, the subsequent actions in the episode will follow the optimal policy π^* until the episode terminates. By the definition of $q^*(s_k,a)$, the expected return for such an episode is equal to $q^*(s_k,a)$. Let G_n denote the return for (s_k,a) for the nth episode in which (s_k,a) appears. Note that after time T, these returns are i.i.d. with mean $q^*(s_k,a)$. Therefore, by Lemma 1,

$$\lim_{u\to\infty} Q_u(s_k,a) = \lim_{N\to\infty} \frac{1}{N}\sum_{n=1}^{N} G_n = q^*(s_k,a) \ \text{ w.p.1} \tag{4}$$

It remains to show that $\pi_u(s_k)$ converges to $\pi^*(s_k)$ w.p.1. Define $a^* = \pi^*(s_k)$. Since π^* is the unique optimal policy, we have:

$$q^*(s_k,a^*) \geq q^*(s_k,a) + \epsilon' \tag{5}$$

for some $\epsilon' > 0$ for all $a \neq a^*$.

Let Ω be the underlying sample space, and let Λ be the set of all $\omega \in \Omega$ such that $Q_u(s_k,a)(\omega)$ converges to $q^*(s_k,a)$ for all $a \in \mathcal{A}$. By (4), $P(\Lambda) = 1$. Thus for any $\omega \in \Lambda$ and any $\epsilon > 0$, there exists a u' (depending on ω) such that $u \geq u'$ implies

$$|q^*(s_k,a) - Q_u(s_k,a)(\omega)| \leq \epsilon \ \text{ for all } a \in \mathcal{A} \tag{6}$$

Let ϵ be any number satisfying $0 < \epsilon < \epsilon'/2$, let $\omega \in \Lambda$, and u' be such that (6) is satisfied for all $u \geq u'$. It follows from (5) and (6) that for any $u \geq u'$ we have

$$\begin{aligned} Q_u(s_k,a^*)(\omega) &\geq q^*(s_k,a^*) - \epsilon && (7)\\ &\geq q^*(s_k,a) + \epsilon' - \epsilon && (8)\\ &\geq Q_u(s_k,a)(\omega) + \epsilon' - 2\epsilon && (9)\\ &> Q_u(s_k,a)(\omega) && (10) \end{aligned}$$

for all $a \neq a^*$. Let u be any iteration after u' such that state s_k is visited in the corresponding episode. From the MCES algorithm, $\pi_u(s_k)(\omega) = \arg\max_a Q_u(s_k,a)(\omega)$. Thus the above inequality implies $\pi_u(s_k)(\omega) = a^*$; furthermore, $\pi_u(s_k)(\omega)$ will be unchanged in any subsequent iteration. Thus, for every $\omega \in \Lambda$, $\pi_u(s_k)(\omega)$ converges to a^*. Since $P(\Lambda) = 1$, it follows $\pi_u(s_k)$ converges to a^* w.p.1., completing the proof.

□

We make the following additional observations: (1) It is not necessary to assume the MDP has a unique optimal policy. The theorem statement and proof go through with minor modification without this assumption. (2) Also we can allow for random reward, with distribution depending on the current state and action. The proof goes through with minor changes.

5 OPFF MDPs

For the case of OPFF, we first need to modify the algorithm slightly to address the issue that an episode might never reach the terminal state for some policies (for example, due to cycling). To this end, let M be some upper bound on the number of states in our MDP. We assume that the algorithm designer has access to such an upper bound (which may be very loose). We modify the algorithm so that if an episode does not terminate within M steps, the episode will be forced to terminate and the returns will simply not be used. We also initialize all Q values to be $-\infty$, so that the policy will always prefer an action that has led to at least one valid return over actions that never yield a valid return. The modified algorithm is given in Algorithm 2. Finally, as in Sutton & Barto (2018), we use first-visit returns for calculating the average return. (This mechanism is not needed for SFF MDPs since under all policies states are never revisited.)

Algorithm 2 First-visit MCES for OPFF MDPs

Initialize: $\pi(s) \in \mathcal{A}$, $Q(s,a) = -\infty$, for all $s \in \mathcal{S}, a \in \mathcal{A}$, arbitrarily;
$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}, a \in \mathcal{A}$.
while True **do**
  Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}$ s.t. all pairs are chosen infinitely often.
  Generate an episode following π: $S_0, A_0, S_1, A_1, \ldots, S_{T-1}, A_{T-1}, S_T$.
  if the episode does not end in less than M time steps **then**
    terminate the episode at time step M
  else
    $G \leftarrow 0$
    for $t = T-1, T-2, \ldots, 0$ **do**
      $G \leftarrow G + r(S_t, A_t)$
      if S_t, A_t does not appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$ **then**
        Append G to $Returns(S_t, A_t)$
        $Q(S_t, A_t) \leftarrow average(Returns(S_t, A_t))$
        $\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

Theorem 2. *Suppose the MDP is OPFF. Then $Q_u(s,a)$ converges to $q^*(s,a)$ and $\pi_u(s)$ converges to $\pi^*(s)$ for all $s \in \mathcal{S}$ and all $a \in \mathcal{A}$ w.p.1.*

The proof can be found in the appendix. As with SFF MDPs, it is not necessary to assume that the MDP has a unique optimal policy; furthermore, the reward can be random (with distribution depending on current state and action). Note for OPFF MDPs, our proof has to take a more sophisticated approach since we now can transition to arbitrary state by taking any non-optimal action.

6 Finite-Horizon MDPs

In this section, we extend our results to finite-horizon MDPs. In this case, we will be able to establish convergence for *all* MDPs (i.e., not just for OPFF MDPs).

A finite-horizon MDP is defined by a finite horizon set $\mathcal{H} = \{0, 1, \ldots, H\}$, a finite state space $\mathcal{S}$, a finite action space $\mathcal{A}$, a reward function $r(s,t,a)$ mapping $\mathcal{S} \times \mathcal{H} \times \mathcal{A}$ to the reals, and a dynamics function $p(\cdot|s,t,a)$, which for every $s \in \mathcal{S}$, $a \in \mathcal{A}$ and time step t, gives a probability distribution over the state space $\mathcal{S}$. The horizon H is the fixed time at which the episode terminates. Note in this setting, the optimal policy $\pi^*(s,t)$ will also be time-dependent, even if $p(\cdot|s,t,a)$ and $r(s,t,a)$ do not depend on t.

The MCES algorithm for this setting is given in Algorithm 3. Note that in this version of the algorithm, during exploring starts we need to choose an $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}$, and an $h \in [0, H]$,.

Corollary 1. *Suppose we are using the finite-horizon optimization criterion. Then $Q_u(s,t,a)$ converges to $q^*(s,t,a)$ and $\pi_u(s,t)$ converges to $\pi^*(s,t)$ for all $s \in \mathcal{S}$, $t \in \mathcal{H}$ and all $a \in \mathcal{A}$ w.p.1.*

Proof. A finite-horizon MDP is equivalent to a standard MDP where the time step is treated as part of the state. Since the number of time steps monotonically increases, this MDP is SFF. Thus the convergence of MCES for the finite-horizon MDP setting follows directly from Theorem 1. □

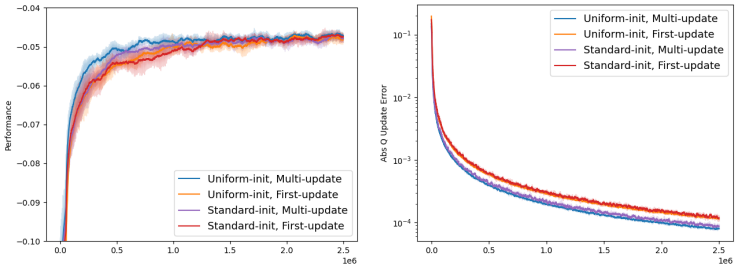
Algorithm 3 MCES for Finite-Horizon MDPs

Initialize: $\pi(s,t) \in \mathcal{A}$, $Q(s,t,a) \in \mathbb{R}$, for all $s \in \mathcal{S}, t \in [0,H], a \in \mathcal{A}$, arbitrarily;
$Returns(s,t,a) \leftarrow$ empty list, for all $s \in \mathcal{S}, t \in [0,H], a \in \mathcal{A}$.
while True **do**
  Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}$, $h \in [0,H]$, s.t. all triples are chosen infinitely often.
  Generate an episode following π: $S_h, A_h, S_{h+1}, A_{h+1}, \ldots, S_{H-1}, A_{H-1}, S_H$.
  $G \leftarrow 0$
  for $t = H-1, H-2, \ldots, h$ **do**
    $G \leftarrow G + r(S_t, t, A_t)$
    Append G to $Returns(S_t, t, A_t)$
    $Q(S_t, t, A_t) \leftarrow average(Returns(S_t, t, A_t))$
    $\pi(S_t, t) \leftarrow \arg\max_a Q(S_t, t, a)$

7 Experimental results

In addition to the theoretical results, we also provide experimental results to compare the convergence rate of the original MCES algorithm and the modified MCES algorithm in Tsitsiklis (2002), where the Q-function estimate is updated with Monte Carlo return only for the initial state-action pair of each episode. We will call the original MCES the "multi-update" variant, and the modified MCES the "first-update" variant. We consider two classical environments: blackjack and stochastic cliffwalking. We will briefly discuss the settings and summarize the results. A more detailed discussion is provided in the appendix.

For blackjack, we use the same problem setting as discussed in Sutton & Barto (1998). Blackjack is an episodic task where for each episode, the player is given 2 initial cards, and can request more cards (hits) or stops (sticks). The goal is to obtain cards whose numerical values sum to a number as great as possible without exceeding 21. Along with the two MCES variants, we also compare two initialization schemes: standard-init, where we initialize by first drawing two random cards from a deck of 52 cards; and uniform-init, where the initial state for an episode is uniformly sampled from the state space. We now compare their rate of convergence in terms of performance and the absolute Q update error, averaged across state-action pairs visited in each episode. Figure 1 shows uniform initialization converges faster than standard initialization, and the multi-update variant is faster than the first-update variant for both initialization schemes.

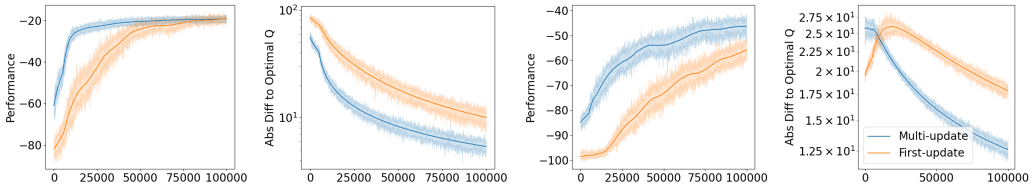


(a) Performance (b) Average absolute Q update error

Figure 1: (a) Performance and (b) average of absolute Q update error (in log scale) for blackjack. Multi-update variant has better performance and lower Q error.

In cliff walking, an agent starts from the bottom left corner of a gridworld, and tries to move to the bottom right corner, without falling into a cliff (an area covering the bottom row of the gridworld). The agent gets a -1 reward for each step it takes, and a -100 for falling into the cliff. We consider two variants: SFF cliff walking and OPFF cliff walking tasks. For the SFF variant, the current time step of the episode is part of the state, so the MDP is a SFF MDP, and the agent tries to learn a time-dependent optimal policy. In this setting we add more stochasticity by having wind move the agent towards one of four directions with some probability. For the OPFF variant, the time step is not part of the state, and the wind only affect the agent when the agent moves to the right, and can only blow the agent one step upwards or downwards, making an OPFF MDP. We now compare the

convergence rate of the two MCES variants on these two cliff walking environments. We measure the convergence rate in terms of the performance and the average L1 distance between the current Q estimate and the optimal Q values. The results are summarized in Figure 2, the training curves are averaged across a number of different gridworld sizes and wind probabilities. Results show that multi-update MCES consistently learns faster than first-update MCES.



(a) OPFF Cliff Walking, Performance

(b) OPFF Cliff Walking, average L1 distance to optimal Q values

(c) SFF Cliff Walking, Performance

(d) SFF Cliff Walking, average L1 distance to optimal Q values

Figure 2: Performance and the average L1 distance to optimal Q values, on OPFF and SFF cliff walking, the curves are averaged over different gridworld sizes and wind probability settings.

Our experimental results show that the multi-update MCES converges much faster than the first-update MCES. These results further emphasize the importance of gaining a better understanding of the convergence properties of the original multi-update MCES variant, which is much more efficient.

8 Conclusion

Theorem 2 of this paper shows that as long as the episodic MDP is OPFF, then the MCES algorithm converges to the optimal policy. As discussed in Section 3.2, many environments of practical interest are OPFF. Our proof does not require that the Q-values be updated at the same rate for all state-action pairs, thereby allowing more flexibility in the algorithm design. Our proof methodology is also novel, and can potentially be applied to other classes of RL problems. Moreover, our methodology also allows us to establish convergence results for finite-horizon MDPs as a simple corollary of Theorem 2. Combining the results of Bertsekas & Tsitsiklis (1996), Tsitsiklis (2002), Chen (2018), Liu (2020), and the results here gives Figure 3, which summarizes what is now known about convergence of the MCES algorithm.

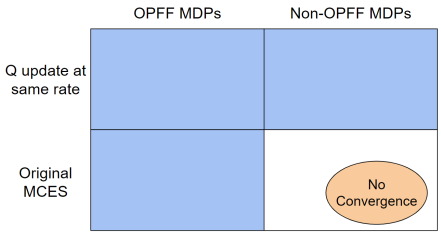


Figure 3: MCES has been studied for two classes of algorithms: Q-values updated at the same rate Tsitsiklis (2002); Chen (2018); Liu (2020) and the original, more flexible algorithm, which does not require the conditions (ii) and (iii) stated in the Introduction. We partition the episodic MDP space into two classes: OPFF MDPs and non-OPFF MDPs. As shown in the figure, this leads to four algorithmic/MDP regions. Convergence is now established for the three blue shaded regions, for all discount factors $\gamma \leq 1$. In the other region, it is known that at least for some non-OPFF MDPs, convergence does not occur (shown as the orange oval region).

The results in this paper along with other previous works (Tsitsiklis, 2002; Chen, 2018; Liu, 2020) make significant progress in establishing the convergence of the MCES algorithm. Many cases of practical interest are covered by the conditions in these papers. It still remains an open problem whether there exist conditions on MDPs that are weaker than the OPFF, and can still guarantee the convergence of the original MCES algorithm.

REFERENCES

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *International Conference on Machine Learning*, pp. 263–272. PMLR, 2017.

Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 2. Athena scientific Belmont, MA, 4 edition, 2012.

Dimitri P Bertsekas and John N Tsitsiklis. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16(3):580–595, 1991.

Dimitri P Bertsekas and John N Tsitsiklis. *Neuro-dynamic programming*, volume 5. Athena Scientific Belmont, MA, 1996.

Yuanlong Chen. On the convergence of optimistic policy iteration for stochastic shortest path problem. *arXiv preprint arXiv:1808.08763*, 2018.

Tommi Jaakkola, Michael I Jordan, and Satinder P Singh. Convergence of stochastic iterative dynamic programming algorithms. In *Advances in neural information processing systems*, pp. 703–710, 1994.

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? *arXiv preprint arXiv:1807.03765*, 2018.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pp. 282–293. Springer, 2006.

Jun Liu. On the convergence of reinforcement learning with monte carlo exploring starts. *arXiv preprint arXiv:2007.10916*, 2020.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*, volume 2. MIT press Cambridge, 1998.

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

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

John N Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine learning*, 16 (3):185–202, 1994.

John N Tsitsiklis. On the convergence of optimistic policy iteration. *Journal of Machine Learning Research*, 3(Jul):59–72, 2002.

Huizhen Yu and Dimitri P Bertsekas. On boundedness of q-learning iterates for stochastic shortest path problems. *Mathematics of Operations Research*, 38(2):209–227, 2013.

A PROOF OF LEMMA 1

Proof. We have

$$
\begin{aligned}
&P(\lim_{N\to\infty}\frac{1}{N}\sum_{i=1}^{N}X_i = x^*)\\
&=\sum_{n=1}^{\infty}P(\lim_{N\to\infty}\frac{1}{N}\sum_{i=1}^{N}X_i = x^*|T=n)P(T=n)\\
&=\sum_{n=1}^{\infty}P(\lim_{N\to\infty}(\frac{1}{N}\sum_{i=1}^{n-1}X_i+\frac{1}{N}\sum_{i=n}^{N}X_i) = x^*|T=n)P(T=n)\\
&=\sum_{n=1}^{\infty}P(\lim_{N\to\infty}\frac{1}{N}\sum_{i=n}^{N}X_i = x^*|T=n)P(T=n) = 1
\end{aligned}
$$

The first equality follows from $P(T<\infty)=1$. The last equality follows from the conditional i.i.d. assumption and the Strong Law of Large Numbers. Note we can apply the Strong Law of Large Numbers here because as stated in Lemma 1, we have the assumption that $X_n, X_{n+1}, \dots$ are i.i.d. with finite mean and finite variance.

□

B PROOF OF THEOREM 2

Proof. Because the MDP is OPFF, its optimal policy MDP graph is a DAG, so we can re-order the states such that from state s_k and selecting the optimal action a_k^*, we can only transition to a state in $\{s_{k+1},\dots,s_N\}$. Note that state s_N is the terminal state, and note that from state s_{N-1} the optimal action only leads to s_N.

We first show that for all $k=1,\dots,N$, $Q_u(s_k,a_k^*)$ converges to $q^*(s_k,a_k^*)$ and that $\pi_u(s_k)$ converges to $\pi^*(s_k)$, w.p.1. Once we establish this convergence result for all $k=1,\dots,N$, we will then complete the proof by establishing convergence of $Q_u(s_k,a)$ for arbitrary actions a. Note that the organization of this proof is slightly more complicated than that of Theorem 1 due to the weaker OPFF assumption.

The result is trivially true for $s=s_N$. Suppose now $Q_u(s_j,a_j^*)$ converges to $q^*(s_j,a_j^*)$ and that $\pi_u(s_j)$ converges to $\pi^*(s_j)$, w.p.1 for all $j=k+1,\dots,N$. We now show $Q_u(s_k,a_k^*)$ converges to $q^*(s_k,a_k^*)$ and that $\pi_u(s_k)$ converges to $\pi^*(s_k)$, w.p.1. Denote $a^*=a_k^*$.

Let T be the iteration u when $\pi_u(s)$ has converged to $\pi_u^*(s)$ for all $s\in\{s_{k+1},\dots,s_N\}$. By the inductive assumption, $P(T<\infty)=1$. Now consider any episode after time T that for some timestep in this episode, we arrive in state s_k and chooses the optimal action a^*. Because the MDP is OPFF, the next state will be in $\{s_{k+1},\dots,s_N\}$, and because of the inductive assumption the subsequent actions in the episode will follow the optimal policy π^* until the episode terminates within a finite number of steps. By the definition of $q^*(s_k,a^*)$, the expected return for (s_k,a^*) in this episode is equal to $q^*(s_k,a^*)$. Let G_n denote the return for nth episode in which we visit s_k and chooses optimal action a^*. Note that after time T, these returns are i.i.d. with mean $q^*(s_k,a^*)$. Therefore, by Lemma 1,

$$
\lim_{u\to\infty}Q_u(s_k,a^*) = \lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N}G_n = q^*(s_k,a^*)\ \text{ w.p.1} \tag{11}
$$

Next we show that $\pi_u(s_k)$ converges to $\pi^*(s_k)=a^*$ w.p.1. Since π^* is the unique optimal policy, we have:

$$
q^*(s_k,a^*) \geq q^*(s_k,a)+\epsilon' \tag{12}
$$

for some $\epsilon'>0$ for all $a\neq a^*$.

The proof at this stage is different from the proof at the corresponding stage in the proof of Theorem 1 since we can now only use (11) for $a=a^*$.

Consider state s_k and an arbitrary action $a \in \mathcal{A}$. From the MCES algorithm, we have $Q_u(s_k, a) = \frac{1}{L_u}\sum_{l=1}^{L_u} G_l$, where L_u is the total number of first-visit returns used to compute $Q_u(s_k, a)$ up through the uth iteration, and G_l is the return value for the lth such iteration. Let Π denote the (finite) set of all deterministic policies, L_u^π denote the number of first-visit returns used to compute $Q_u(s_k, a)$ up through the uth iteration when using policy π, and G_l^π, $1 \leq l \leq L_u^\pi$, denote the lth return value when policy π is used. We have

$$Q_u(s_k, a) = \frac{1}{L_u}\sum_{l=1}^{L_u} G_l \tag{13}$$

$$= \sum_{\pi \in \Pi} \frac{L_u^\pi}{L_u}\left(\frac{1}{L_u^\pi}\sum_{l=1}^{L_u^\pi} G_l^\pi\right) \tag{14}$$

By the law of large numbers, we know that for any policy π such that $L_u^\pi \to \infty$ we have w.p.1

$$\lim_{u\to\infty} \frac{1}{L_u^\pi}\sum_{l=1}^{L_u^\pi} G_l^\pi = q^\pi(s_k, a) \leq q^*(s_k, a) \tag{15}$$

where $q^\pi(s_k, a)$ is the action-value function for policy π. The inequality in (15) follows from the definition of $q^*(s, a)$. It follows from (14) and (15) that w.p.1

$$\limsup_{u\to\infty} Q_u(s_k, a) \leq q^*(s_k, a) \text{ for all } a \tag{16}$$

Note that in the OPFF setting, in the special case that no valid return has been obtained for (s_k, a), then $Q_u(s_k, a) = -\infty \leq q^*(s_k, a)$, so the above inequality still holds.

Let Ω be the underlying sample space, and let $\Lambda \subset \Omega$ be the set over which (16) holds. Note that $P(\Lambda) = 1$. Thus for any $\omega \in \Lambda$ and any $\epsilon > 0$, there exists a u' such that $u \geq u'$ implies

$$Q_u(s_k, a)(\omega) \leq q^*(s_k, a) + \epsilon \text{ for all } a \in \mathcal{A} \tag{17}$$

Let ϵ be any number satisfying $0 < \epsilon < \epsilon'/2$, let $\omega \in \Lambda$, and u' be such that (17) is satisfied for all $u \geq u'$. It follows from (11), (12) and (17) that for any $u \geq u'$ we have

$$Q_u(s_k, a^*)(\omega) \geq q^*(s_k, a^*) - \epsilon \tag{18}$$

$$\geq q^*(s_k, a) + \epsilon' - \epsilon \tag{19}$$

$$\geq Q_u(s_k, a)(\omega) + \epsilon' - 2\epsilon \tag{20}$$

$$> Q_u(s_k, a)(\omega) \tag{21}$$

for all $a \neq a^*$. Let u be any iteration after u' such that the episode includes state s_k. From the MCES algorithm, $\pi_u(s_k)(\omega) = \arg\max_a Q_u(s_k, a)(\omega)$. Thus the above inequality implies $\pi_u(s_k)(\omega) = a^*$; furthermore, $\pi_u(s_k)(\omega)$ will be unchanged in any subsequent iteration. Thus, for every $\omega \in \Lambda$, $\pi_u(s_k)(\omega)$ converges to a^*. Since $P(\Lambda) = 1$, it follows $\pi_u(s_k)$ converges to a^* w.p.1.

We have now shown that for all $k = 1, \ldots, N$, $Q_u(s_k, a_k^*)$ converges to $q^*(s_k, a_k^*)$ and that $\pi_u(s_k)$ converges to $\pi^*(s_k)$, w.p.1. It remains to show convergence of $Q_u(s, a)$ to $q^*(s, a)$ for arbitrary state s and action a. Let u' be such that $u \geq u'$, $\pi_u(s) = \pi^*(s)$ for all $s \in \mathcal{S}$. Consider an episode after iteration u' in which we visit state s and take action a. After taking action a, the policy follows the optimal policy π^*. Thus the expected return for (s, a) in this episode is $q^*(s, a)$. We can thus once again apply Lemma 1 to show $Q_u(s, a)$ converges to $q^*(s, a)$ w.p.1, thereby completing the proof. □

C ADDITIONAL EXPERIMENTAL DETAILS

C.1 BLACKJACK ENVIRONMENT

Here we give a more in-depth discussion on our blackjack experiments.

We use the same problem setting as discussed in Sutton & Barto (1998). We consider only a single player against a dealer. The goal of the blackjack game is to request a number of cards so that the sum of the numerical values of your cards is as great as possible without exceeding 21. All face cards will be counted as 10 and an ace can be counted as either 1 or 11. If the player holds an ace that can be counted as 11 without causing the total value to exceed 21, then the ace is said to be usable.

For each round of blackjack, both the player and the dealer will get two cards respectively at the beginning. One of the dealer's cards is face up and another is face down. For each time step in the episode, the player has two possible actions: either to request an additional card (hits), or stops (sticks), if the player's sum exceeds 21 (goes bust), then the player loses. After the player sticks, The dealer will hit or stick according to a fixed policy: he sticks on any sum of 17 or greater and hits otherwise, if the dealer goes bust then the player wins. In the end, if both sides do not go bust, their sums are compared, and the player wins if the player's sum is greater, and loses if the dealer's sum is greater, and the game is a draw if both sides have the same sum. When the episode terminates, the reward is 1 if the player wins, -1 if the player loses, and 0 if the game is a draw. Note that in an episode, the player cannot revisit a previously visited state (if the player has a usable ace, the player can move into a state with a unusable ace, but not vice-versa). Therefore blackjack is a stochastic feed-forward (SFF) environment.

For blackjack, the player's state consists of three components: the player's current sum, the card that the dealer is showing, and whether the player has a usable ace. There are some states where the optimal policy is trivial. For example, when the player has a very small sum, the player can always hit and the sum value will increase for sure without the danger of going bust. If we ignore such special states, then we have a total of 200 states that are interesting to the player (it is 200 states considering the player's sum (12-21), dealer's showing card (ace-10) and whether the player has a usable ace (Sutton & Barto, 1998)). For the uniform initialization scheme, the initial state of the player is uniformly sampled from these 200 states.

For the standard initialization, we instead simply follow the rules of blackjack. We randomly draw cards for the player and the dealer, each of these cards can be one of the 13 cards (ace-10, J, Q and K) with equal probability. (The cards are dealt from an infinite deck, i.e., with replacement.)

When we compare the convergence rate of the multi-update MCES and the first-update MCES, we use performance and the average absolute Q update error. The Q update error for a state-action pair is simply the difference between the Q values before and after a Q update. For each episode, the average absolute Q update error is averaged over all state-action pairs that are updated in this episode. In all figures, the solid curve shows the mean value over 5 seeds, and the shaded area indicates the confidence interval.

In Figure 1, we see that the first-update MCES variant (which only updates the Q value for the initial state-action pair in each episode) works in blackjack even when we sample according to the standard rule of blackjack and not uniformly. This might be due to fact that for blackjack, although the initial state distribution under standard initialization (where the player gets 2 random initial cards from the dealer) is not the same as uniform initialization, it still sufficiently covers the state space, so the agent is able to update its value estimates for all state-action pairs and gradually learn a good policy. However, this is not the case for cliffwalking, where if we initialize an episode according to the standard rule, then the initial state for the agent is always at the the bottom left corner of the gridworld, making it impossible to update the value function for other states. In many practical problems, we might not have access to an initial state distribution that sufficiently covers the entire state space, so it is very important to gain a better understanding of the convergence properties of the multi-update MCES variant.

C.2 STOCHASTIC CLIFF WALKING ENVIRONMENT

Here we give a more in-depth discussion on our cliff walking experiments.

As illustrated in Figure 4, in the cliff walking environment, the agent starts from the bottom left corner of a gridworld (marked with the letter S), and tries to move to the goal state at the bottom right corner (marked with the letter G), without falling into the cliff area (shaded area at the bottom row of the gridworld).

In our setting, the agent can move in four directions (right, up, left, down), and gets a -1 reward for each step it takes, a -100 for falling into the cliff, and a 0 for reaching the goal state. If the agent reaches the goal state or falls off the cliff, the episode will immediately terminate. If the agent moves out of the boundary of the gridworld, the agent will be "bounced" back and will not be able to go outside the boundaries.

Here we consider two variants of the cliffwalking environment: SFF cliff walking and OPFF cliff walking. For the SFF variant, the current time step of the episode is part of the state. Since the time step can only increase, the MDP is now a SFF MDP, and the agent tries to learn a time-dependent optimal policy. In this setting we add stochasticity by having wind move the agent one step towards one of the four directions with some probability. For example, when the wind probability is 50%, there is 12.5% chance that the agent will take an additional random step for each direction due to the wind. In Figure 4 this is illustrated with the arrows near the position P_2. If the agent is in the SFF cliff walking environment and is currently at position P_2, then if the agent moves one step upward (indicated by the solid arrow), there is some chance that the agent will also take an additional step towards one of the four directions, with equal probability (indicated by dashed arrows). Note that the agent's state space now includes the x, y positions of the agent, as well as the current time step.

For the OPFF variant, the time step is not part of the state. To make sure the environment is OPFF, we only allow the wind to affect the agent when it moves to the right, and the wind can only blow the agent one step upwards, or downwards. This is shown in Figure 4 by the arrows near the position P_1, if the agent is in the OPFF cliff walking environment and is current in position P_1, then if the agent moves one step to the right, the wind has some probability of making it take an additional step upwards or downwards, with equal probability. In this setting, if the agent takes any action other than moving one step right, then it will not be affected. Under such a setting, if the agent is following an optimal policy, then in an episode it will never revisit a previously visited state, thus the MDP is OPFF.

Special care is required when running MCES in the OPFF environment, since it is possible the agent can run into infinite loops (e.g. keep bumping into the boundary), depending on the initial policy. One simple method to tackle this issue is to have an artificial time limit, and if during one episode the agent reaches this time limit, the episode is terminated and the agent receives a large, negative reward. This allows deterministic agents to train effectively in these OPFF environments.

C.2.1 CLIFF WALKING EXPERIMENTAL RESULTS

We use performance and the average L1 distance to the optimal Q value as metrics for the convergence rates of the multi-update and first-update MCES variants. Here the optimal Q values are computed using value iteration. The full experimental results are shown in Figures 5, 6, 7 and 8. We present results on gridworld sizes of $8 \times 6, 12 \times 9, 16 \times 12$ and wind probability = $0.1, 0.3, 0.5$.

In Fig 5 and Fig 6, we consider the OPFF cliff walking environment, where time is not included in the state space and the wind only affects the action to the right.

In Fig 7 and Fig 8, we consider the SFF cliff walking environment, where the current time step is included as part of the state and the wind can affect the agent in all four directions.

Results in both cliff walking environments, and across all tested gridworld size and wind probability consistently show that multi-update MCES learns faster than first-update MCES. These results show there is indeed a significant performance difference between these two variants of the MCES algorithm. We also observe a general trend that the performance gap tends to increase when the state space becomes larger (a larger gridworld) and when we have more stochasticity in the environment (a higher wind probability). In Figure 6 and 8, each row shows performance for a gridworld size and each column shows a wind probability, if we go from the top left figure (figure (a)) to the bottom

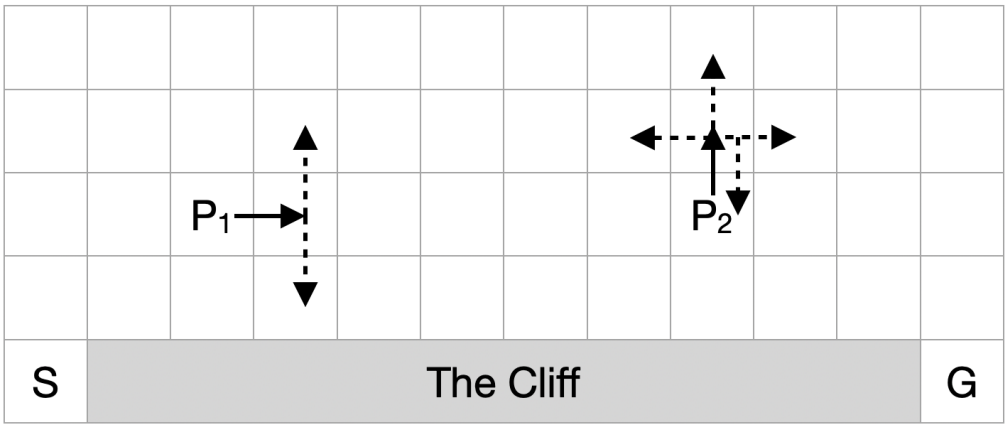
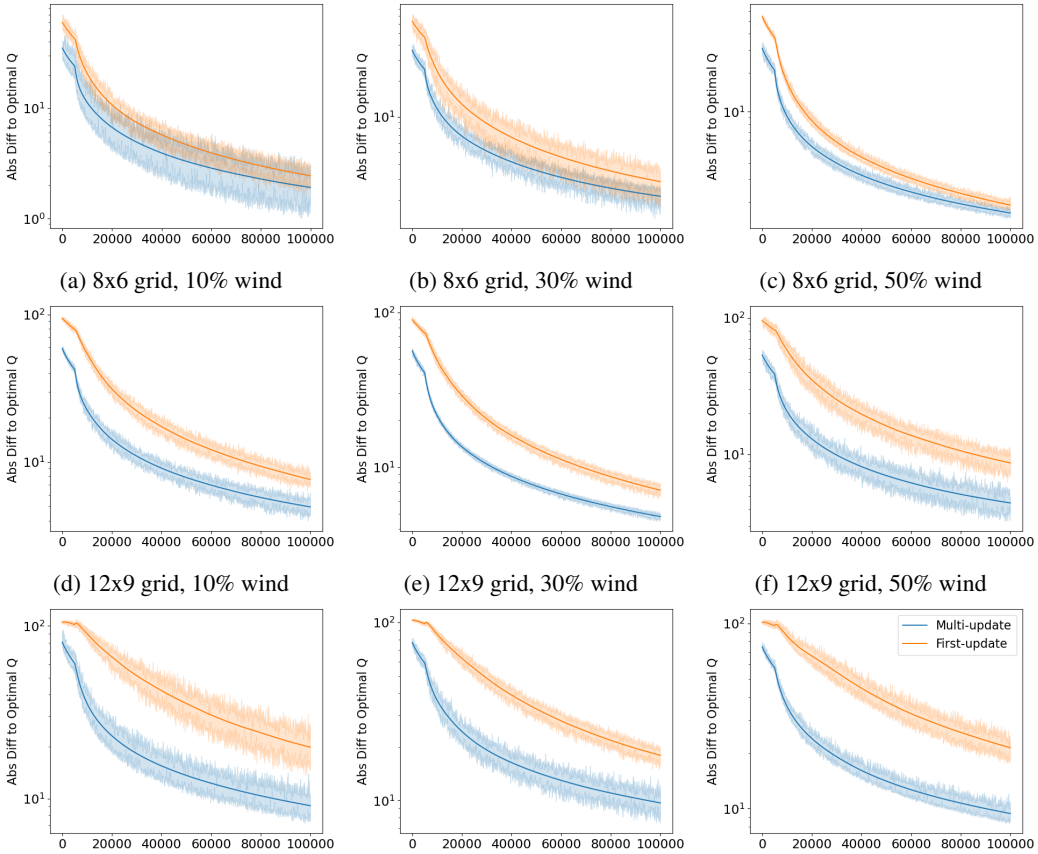


Figure 4: Cliff Walking Environment Illustration: The grid world contains of a start state, a goal state and a cliff area between them. In the OPFF cliff walking setting with wind probability $\mathbb{P}_w$, when the agent is at P_1 and moves one step to the right, it will take an additional step upwards or downwards, each with probability $\frac{\mathbb{P}_w}{2}$ and not take this additional step with probability $1 - \mathbb{P}_w$. In the SFF cliff walking environment with wind probability $\mathbb{P}_w$, when the agent is at P_2 and takes one step up, it will take an additional step towards one of the four directions, each with probability $\frac{\mathbb{P}_w}{4}$, and take no additional step with probability $1 - \mathbb{P}_w$.

right figure (figure (i)), we see that the performance gap tends to become bigger. This result shows that taking more updates to the value estimate can be important especially in complex tasks with more randomness.

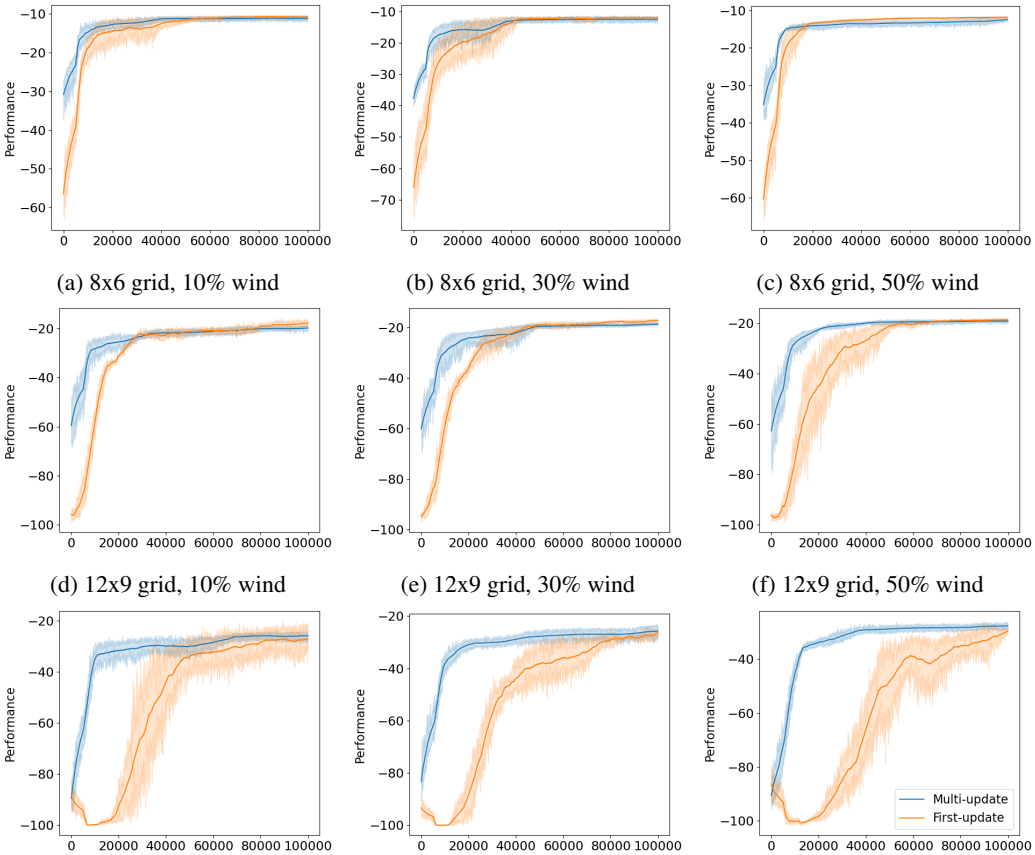


(a) 8x6 grid, 10% wind (b) 8x6 grid, 30% wind (c) 8x6 grid, 50% wind

(d) 12x9 grid, 10% wind (e) 12x9 grid, 30% wind (f) 12x9 grid, 50% wind

(g) 16x12 grid, 10% wind (h) 16x12 grid, 30% wind (i) 16x12 grid, 50% wind

Figure 5: OPFF Cliff Walking: average L1 distance to optimal Q values, with different grid world sizes and wind probabilities. Y-axis is in log scale.

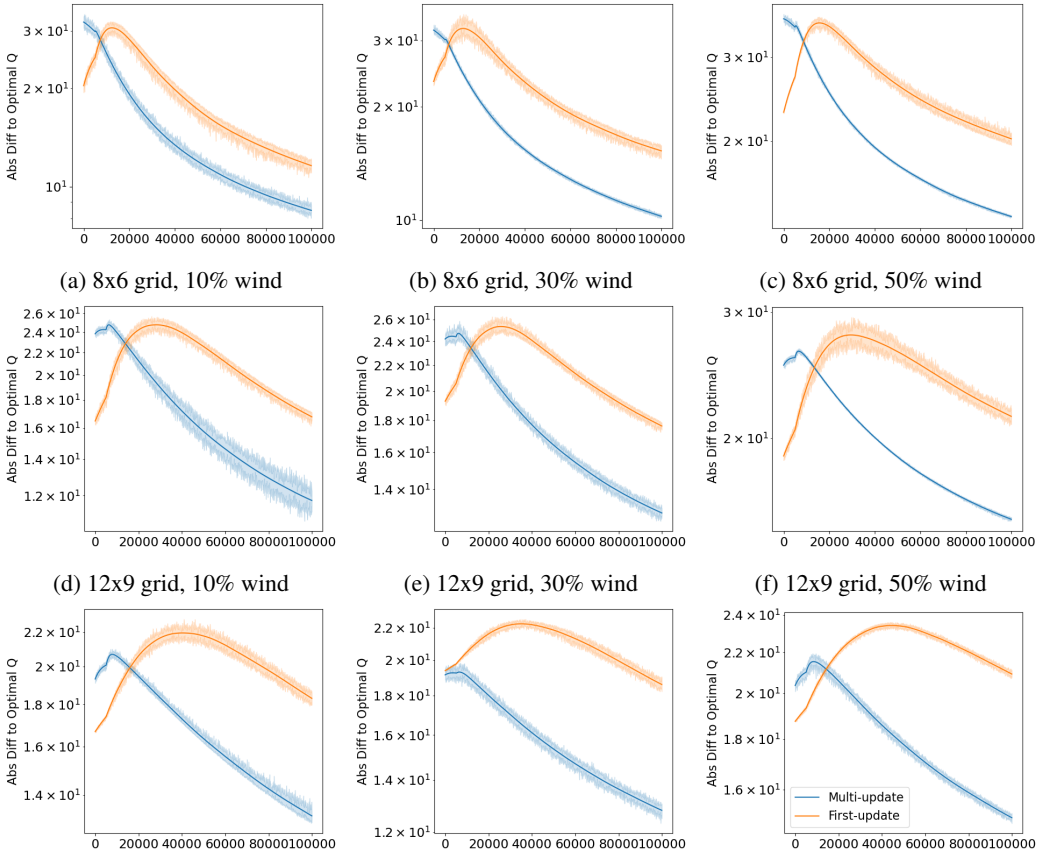


(a) 8x6 grid, 10% wind (b) 8x6 grid, 30% wind (c) 8x6 grid, 50% wind

(d) 12x9 grid, 10% wind (e) 12x9 grid, 30% wind (f) 12x9 grid, 50% wind

(g) 16x12 grid, 10% wind (h) 16x12 grid, 30% wind (i) 16x12 grid, 50% wind

Figure 6: OPFF Cliff Walking: Performance, with different grid world sizes and wind probabilities.

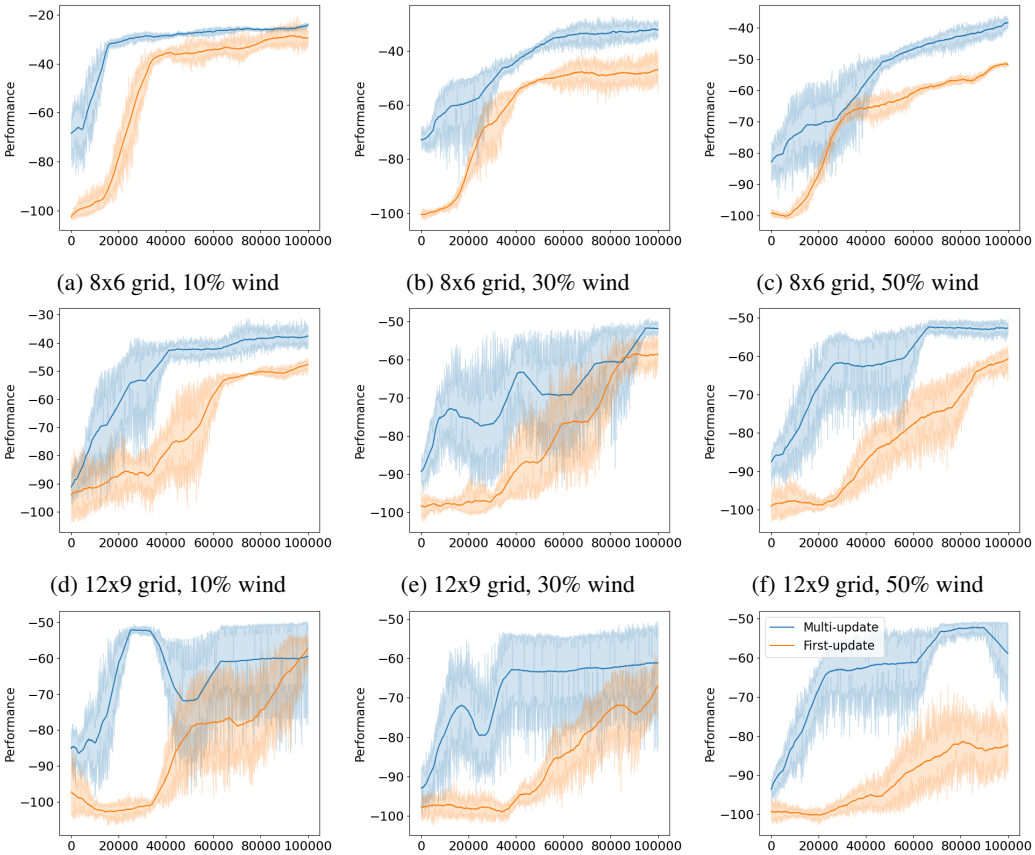


(a) 8x6 grid, 10% wind (b) 8x6 grid, 30% wind (c) 8x6 grid, 50% wind

(d) 12x9 grid, 10% wind (e) 12x9 grid, 30% wind (f) 12x9 grid, 50% wind

(g) 16x12 grid, 10% wind (h) 16x12 grid, 30% wind (i) 16x12 grid, 50% wind

Figure 7: SFF Cliff Walking: average L1 distance to optimal Q values, with different grid world sizes and wind probabilities, y axis is in log scale.



(a) 8x6 grid, 10% wind (b) 8x6 grid, 30% wind (c) 8x6 grid, 50% wind

(d) 12x9 grid, 10% wind (e) 12x9 grid, 30% wind (f) 12x9 grid, 50% wind

(g) 16x12 grid, 10% wind (h) 16x12 grid, 30% wind (i) 16x12 grid, 50% wind

Figure 8: SFF Cliff Walking: Performance, with different grid world sizes and wind probabilities.