Reinforcement Learning for Control with Multiple Frequencies

Jongmin Lee[1], **Byung-Jun Lee**[1], **Kee-Eung Kim**[1,2]
[1] School of Computing, KAIST, Republic of Korea
[2] Graduate School of AI, KAIST, Republic of Korea
{jmlee,bjlee}@ai.kaist.ac.kr, kekim@kaist.ac.kr

Abstract

Many real-world sequential decision problems involve multiple action variables whose control frequencies are different, such that actions take their effects at different periods. While these problems can be formulated with the notion of multiple action persistences in factored-action MDP (FA-MDP), it is non-trivial to solve them efficiently since an action-persistent policy constructed from a stationary policy can be arbitrarily suboptimal, rendering solution methods for the standard FA-MDPs hardly applicable. In this paper, we formalize the problem of multiple control frequencies in RL and provide its efficient solution method. Our proposed method, Action-Persistent Policy Iteration (AP-PI), provides a theoretical guarantee on the convergence to an optimal solution while incurring only a factor of $|\mathcal{A}|$ increase in time complexity during policy improvement step, compared to the standard policy iteration for FA-MDPs. Extending this result, we present Action-Persistent Actor-Critic (AP-AC), a scalable RL algorithm for high-dimensional control tasks. In the experiments, we demonstrate that AP-AC significantly outperforms the baselines on several continuous control tasks and a traffic control simulation, which highlights the effectiveness of our method that directly optimizes the periodic non-stationary policy for tasks with multiple control frequencies.

1 Introduction

In recent years, reinforcement learning (RL) [23] has shown great promise in various domains, such as complex games [14, 21, 22] and high-dimensional continuous control [11, 19]. These problems have been mostly formulated as *discrete-time* Markov decision processes (MDPs) [17], assuming all decision variables are simultaneously determined at every time step. However, many real-world sequential decision-making problems involve multiple decision variables whose control frequencies are different by requirement. For example, when managing a financial portfolio of various assets, the frequency of rebalancing may need to be different for each asset, e.g. weekly for stock and monthly for real estate. Similarly, robotic systems typically consist of a number of controllers operating at different frequencies due to their system specification.

Different control frequencies can be formulated with the notion of different *action persistence* in the discrete-time factored-action MDP (FA-MDP), where the base time interval is determined by the reciprocal of the least common multiple of the control frequencies. However, while algorithms for *single* action persistence has been proposed in order to improve the empirical performance of online [9] or offline [13] RL agents, to the best of our knowledge, addressing *multiple* action persistences in RL has been mostly unexplored due to its difficulty involved in the non-stationarity nature of the optimal policy.

In this paper, we formalize the problem of multiple action persistences in FA-MDPs. We first show that any persistent policy induced by a stationary policy can be arbitrarily bad via a simple example.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

Then, we introduce efficient methods for FA-MDPs that directly optimize a periodic non-stationary policy while circumventing the exponential growth of time complexity with respect to the periodicity of action persistence. We first present a tabular planning algorithm, Action-Persistent Policy Iteration (AP-PI), which provides the theoretical guarantee on the convergence to an optimal solution while incurring only a factor of $|\mathcal{A}|$ time complexity increase in the policy improvement step compared to the policy iteration for standard FA-MDPs. We then present Action-Persistent Actor-Critic (AP-AC), a scalable learning algorithm for high-dimensional tasks via practical approximations to AP-PI, with a neural network architecture designed to facilitate the direct optimization of a periodic non-stationary policy. In the experiments, we demonstrate that AP-AC significantly outperforms a number of baselines based on SAC, from the results on modified Mujoco continuous control benchmarks [3, 26] and the SUMO traffic control simulation [8], which highlights the effectiveness of our method that directly optimizes the periodic non-stationary policy for tasks with multiple control frequencies.

2 Preliminaries

We assume the environment modeled as discrete-time factored-action MDP (FA-MDP) $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$ where $\mathcal{S}$ is the set of states s, $\mathcal{A}$ is the set of vector-represented actions $a = (a^1, \ldots, a^m)$, $P(s'|s,a) = \Pr(s_{t+1} = s'|s_t = s, a_t = a)$ is the transition probability, $R(s,a) \in \mathbb{R}$ is the immediate reward for taking action a in state s, and $\gamma \in [0,1)$ is the discount factor. A policy $\pi = (\pi_t)_{t \geq 0} \in \Pi$ is a sequence of functions where $\pi_t : \mathcal{H} \to \Delta(\mathcal{A})$ is a mapping from history $h_t = (s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t)$ to a probability distribution over $\mathcal{A}$, $\pi_t(a_t|h_t) = \Pr(a_t|h_t)$. We call π_t Markovian if π_t depends only on the last state s_t and call it stationary if π_t does not depend on t. The policy π_t is called deterministic if it maps from history to some action with probability 1 and can be denoted as $\pi_t : \mathcal{H} \to \mathcal{A}$. For simplicity, we will only consider the fully factorized policy $\pi_t(a_t^1, \ldots, a_t^m|h_t) = \prod_{k=1}^m \pi_t^k(a_t^k|h_t)$, which comprises the set Π of all fully factorized policies. The action-value function Q_t^π of policy π is defined as $Q_t^\pi(s,a) = \mathbb{E}_\pi \left[\sum_{\tau=t}^\infty \gamma^{\tau - t} R(s_\tau, a_\tau) | s_t = s, a_t = a\right]$.

We consider the sequential decision problem where each action variable a^k has its own control frequency. The notion of control frequency can be formulated in terms of *action persistence* with FA-MDP $\mathcal{M}$ by considering how frequently a^k should be decided in $\mathcal{M}$. Specifically, we let c^k be the action persistence of k-th action variable a^k, i.e. a^k is decided every c^k time step in $\mathcal{M}$. The overall action persistence of the decision problem is then described as a vector $c = (c^1, \ldots, c^m) \in \mathbb{N}^m$. Finally, we define the c-persistent policy π as follows:

Definition 1. (c-persistent policy) *Let* $\pi = (\pi_t)_{t \geq 0} \in \Pi$ *be a policy. Given the action persistence vector* $c \in \mathbb{N}^m$*, the* c*-persistent policy* $\bar{\pi}_c = (\bar{\pi}_{c,t})_{t \geq 0}$ *induced by* π *is a non-stationary policy where*

$$\forall t,\ \bar{\pi}_{c,t}(a|h_t) = \prod_{k=1}^m \bar{\pi}_{c,t}^k(a^k|h_t) \text{ s.t. } \bar{\pi}_{c,t}^k(a^k|h_t) = \begin{cases} \pi_t^k(a^k|h_t) & \text{if } t \bmod c^k = 0 \\ \delta_{a^k_{t-(t \bmod c^k)}}(a^k) & \text{otherwise} \end{cases} \tag{1}$$

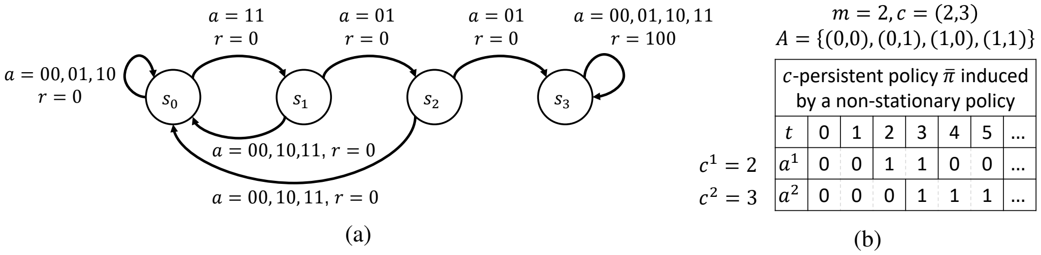
where $\delta_x(y) = 1$ *if* $x = y$ *and* 0 *otherwise. Additionally, we define the set of* c*-persistent policies* $\Pi_c = \{(\bar{\pi}_{c,t})_{t \geq 0} : \pi \in \Pi\}$.

Our goal is to find the c-persistent policy π_c^* that maximizes expected cumulative rewards:

$$\bar{\pi}_c^* = \operatorname*{arg\,max}_{\bar{\pi} \in \Pi_c} \mathbb{E}_{\bar{\pi}} \left[\sum_{t=0}^\infty \gamma^t R(s_t, a_t) \right] \tag{2}$$

Remark. When $c = (1, \ldots, 1)$, we have $\Pi_c = \Pi$. Thus, Eq. (2) is reduced to the standard objective function of FA-MDP, which is known to always have a deterministic and Markovian stationary policy as an optimal solution [17]. Also, the c-persistent policy of Definition 1 is different from the k-persistent policy [13] in that our definition considers *multiple* action persistences and is *not* limited by Markovian policy π while [13] considers *single* action persistence and a non-stationary policy induced only by a *Markovian* policy.

The agent with c-persistent policy $\bar{\pi}_c$ induced by π interacts with the environment as follows: At time step $t = 0$, all action variables are selected according to $\bar{\pi}_{c,0} = \pi_0$, i.e. $(a_0^1, \ldots, a_0^m) \sim \prod_{k=1}^m \pi_0^k(\cdot|h_0)$. Then, each action variable a^k is kept *persistent* for the subsequent $c^k - 1$ time steps. At time step $t = c^k$, the action variable a^k is set by $\bar{\pi}_{c,t}^k(\cdot|h_t) = \pi_t^k(\cdot|h_t)$, and continue into the next



(a)

$m = 2, c = (2,3)$
$A = \{(0,0), (0,1), (1,0), (1,1)\}$

	c-persistent policy $\bar{\pi}$ induced by a non-stationary policy							
	t	0	1	2	3	4	5	...
$c^1 = 2$	a^1	0	0	1	1	0	0	...
$c^2 = 3$	a^2	0	0	0	1	1	1	...

(b)

Figure 1: An illustrative example of FA-MDP with two action persistence where the initial state is s_0 and $\gamma = 0.95$. The arrows indicate transitions and rewards for each action. In this simple example, every c-persistent policy induced by any stationary policy is suboptimal.

time step. In other words, the agent decides the value for a^k only at the time steps t that are multiples of c^k, i.e. $t \bmod c^k = 0$. Figure 1b illustrates an example of c-persistent policy $\bar{\pi}_c$. For the remainder of this paper, we will omit the subscript c in $\bar{\pi}_c$ for notational brevity if there is no confusion. All the proofs of theorems are available in the Appendix.

3 Action-Persistence in FA-MDPs

Finding the optimal policy via Eq. (2) is non-trivial since any c-persistent policy naively constructed from a *stationary* policy can be suboptimal, unlike in standard FA-MDPs where there always exists a stationary optimal policy. To see this, consider the FA-MDP depicted in Figure 1, where there are two action variables with action persistences 2 and 3, respectively. In this example task, in order to obtain a positive reward, the agent should take an action $a = (1, 1)$ at state s_0 to go to the rightmost state. However, when we use this to form a stationary deterministic policy with $\pi(s_0) = (1, 1)$ and construct a c-persistent policy in a naive manner, we see that the policy can never reach s_3 due to the inherent action persistence $c = (2, 3)$: The action $(1, 1)$ taken at s_0 when $t = 0$ will persist at the next time step $t = 1$ in s_1, making the agent go back to s_0. Then, the agent will select an action $(1, 1)$ again by $\pi(s_0)$, and this will be repeated forever. As a consequence, the agent visits only s_0 and s_1, and thus cannot reach the rightmost state. In contrast, the *non-stationary* deterministic policy $\bar{\pi}$ described in Figure 1b reaches s_3. Careful readers may notice that a c-persistent policy "projected" from some stationary but stochastic policy can eventually reach s_3, but its expected return is clearly less than the non-stationary deterministic policy in Figure 1b, thus suboptimal.

Therefore, obtaining a c-persistent policy by ignoring the action persistence requirement and solving the corresponding standard FA-MDP would not work. However, one can observe that the action persistence scheme is repeated periodically at every $L \triangleq \mathrm{LCM}(c^1, \ldots, c^m)$ time steps. From this observation, a naive approach to solving Eq. (2) would be redefining the action space to have L-step actions as elements. After redefining the transition and reward function corresponding to these actions, standard solution methods for FA-MDP such as dynamic programming can be applied. Still, this approach not only has exponential time complexity with respect to L due to the increase in the size of action space, i.e. $|\mathcal{A}|^L$, but also can be suboptimal unless the underlying transition dynamics is nearly deterministic due to the *open-loop* decision-making nature of L-step actions [27]. A more principled approach is to consider an L-Markovian policy that memorizes which action was taken during the last L steps, but its straightforward conversion to the standard MDP via state augmentation still suffers from the exponential time complexity with respect to L.

3.1 Policy evaluation for c-persistent policy: c-persistent Bellman operators

As discussed in the previous section, augmenting state or action space for storing L-step information results in exponential complexity with respect to L. Instead, we take a more direct approach that optimizes the c-persistent policy via composition of Bellman operators within the space of L-periodic, non-stationary and deterministic policies Π_L:

$$\Pi_L = \{\pi \in \Pi : \forall t, \pi_t = \pi_{t+L} \text{ and } \pi_t : \mathcal{A} \times \mathcal{S} \to \mathcal{A}\} \tag{3}$$

We will later prove that there always exists an optimal policy for Eq. (2), which is induced by $\pi \in \Pi_L$. The policy in Π_L will be denoted as $\pi = (\pi_0, \ldots, \pi_{L-1})$ in the remainder of the paper.

As the first step of the derivation of our algorithm, we define function $\Gamma^c_{t,a}(a')$:

$$\Gamma^c_{t,a}(a') = (\bar{a}^1, \dots \bar{a}^m) \text{ where } \bar{a}^k \triangleq \begin{cases} a^k & \text{if } t \bmod c^k \neq 0 \\ a'^k & \text{if } t \bmod c^k = 0 \end{cases} \tag{4}$$

which projects action a' into a feasible action at time step t if the action taken at $t-1$ is assumed to be a. This is done by extracting dimensions of "effectable" action variables at time step t from a' and extracting dimensions of "uneffectable" variables at time step t from a.

For the L-periodic non-stationary deterministic policy $\pi = (\pi_0, \dots, \pi_{L-1}) \in \Pi_L$, we first define the one-step c-persistent Bellman operator $\bar{\mathcal{T}}_t^\pi$ induced by π. Specifically, for $t \in \{0, \dots, L-1\}$,

$$(\bar{\mathcal{T}}_t^\pi Q)(s,a) \triangleq R(s,a) + \gamma \mathbb{E}_{\substack{s' \sim P(s'|s,a) \\ a' = \pi_{t+1}(a,s')}} \left[Q(s', \Gamma^c_{t+1,a}(a')) \right] \tag{5}$$

Then, we define an L-step c-persistent Bellman operator $\bar{H}_t^\pi$ by making the composition of L one-step c-persistent Bellman operators:

$$\begin{aligned} (\bar{H}_0^\pi Q)(s,a) &\triangleq (\bar{\mathcal{T}}_0^\pi \bar{\mathcal{T}}_1^\pi \cdots \bar{\mathcal{T}}_{L-2}^\pi \bar{\mathcal{T}}_{L-1}^\pi Q)(s,a) \\ (\bar{H}_1^\pi Q)(s,a) &\triangleq (\bar{\mathcal{T}}_1^\pi \bar{\mathcal{T}}_2^\pi \cdots \bar{\mathcal{T}}_{L-1}^\pi \bar{\mathcal{T}}_0^\pi Q)(s,a) \\ &\vdots \\ (\bar{H}_{L-1}^\pi Q)(s,a) &\triangleq (\bar{\mathcal{T}}_{L-1}^\pi \bar{\mathcal{T}}_0^\pi \cdots \bar{\mathcal{T}}_{L-3}^\pi \bar{\mathcal{T}}_{L-2}^\pi Q)(s,a) \end{aligned} \tag{6}$$

The following theorem and corollary state that each L-step c-persistent Bellman operator $\bar{H}_t^\pi$ is a contraction mapping, and each of the fixed points $Q_0^{\bar{\pi}}, \dots, Q_{L-1}^{\bar{\pi}}$ has a recursive relationship with another by one-step c-persistent Bellman operators $\bar{\mathcal{T}}_0^\pi, \dots, \bar{\mathcal{T}}_{L-1}^\pi$.

Theorem 1. *For all $t \in \{0, \dots, L-1\}$, the L-step c-persistent Bellman operators $\bar{H}_t^\pi$ is γ^L-contraction with respect to infinity norm, thus $\bar{H}_t^\pi Q_t^{\bar{\pi}} = Q_t^{\bar{\pi}}$ has the unique fixed point solution. In other words, for any $Q_t^0 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, define $Q_t^{n+1} = \bar{H}_t^\pi Q_t^n$. Then, the sequence Q_t^n converges to t-th c-persistent value function of $\bar{\pi}$ as $n \to \infty$.*

Corollary 1. *$Q_t^{\bar{\pi}} = \bar{\mathcal{T}}_t^\pi Q_{(t+1) \bmod L}^{\bar{\pi}}$ holds for all $t \in \{0, \dots, L-1\}$, thus c-persistent value functions can be obtained by repeatedly applying 1-step c-persistent backup in a L-cyclic manner.*

Note that the c-persistent value function of the policy π obtained by $\bar{H}_t^\pi$, has the following form:

$$Q_t^{\bar{\pi}}(s,a) = \mathbb{E}_{\substack{\forall \tau,\ s_{\tau+1} \sim P(\cdot|s_\tau, a_\tau) \\ \bar{a}_{\tau+1} = \Gamma^c_{\tau+1, \bar{a}_\tau}(\pi_{\tau+1}(\bar{a}_\tau, s_{\tau+1}))}} \left[\sum_{\tau=t}^{\infty} \gamma^{\tau-t} R(s_\tau, \bar{a}_\tau) \,\middle|\, s_t = s, \bar{a}_t = a \right] \tag{7}$$

which is obtained by unfolding the L-step c-persistent Bellman recursion from $\bar{H}_t^\pi$. Here, one can easily show that every action taken at every time step t, which is projected by $\Gamma^c_{t,\bar{a}}(\cdot)$, abides by c-persistence, by mathematical induction. As a result, $Q_t^{\bar{\pi}}(s,a)$ has the intended interpretable meaning, i.e. the expected sum of rewards that can be obtained when following the c-persistent policy $\bar{\pi}$ which is induced by π, except for the initial action a, starting from the state s at time step t.

Remark. The time complexity of applying the one-step c-persistent Bellman backup $\bar{\mathcal{T}}_t^\pi$ of Eq. (5) for a deterministic policy π is $O(|\mathcal{S}|^2|\mathcal{A}|)$ for each t, which is identical to the time complexity of the non-persistent standard Bellman backup.

Now, we have a complete policy evaluation operator for c-persistent policy induced by L-periodic non-stationary deterministic policy π.

3.2 Policy improvement for c-persistent policy

The remaining step for full policy iteration is policy improvement using $Q_t^{\bar{\pi}}(s,a)$.

Theorem 2. *Given a L-periodic, non-stationary, and deterministic policy $\pi = (\pi_0, \dots, \pi_{L-1}) \in \Pi_L$, let $Q_t^{\bar{\pi}}$ be the c-persistent value of $\bar{\pi}$ denoted in Eq. (7). If we update the new policy $\pi^{\text{new}} = (\pi_0^{\text{new}}, \dots, \pi_{L-1}^{\text{new}}) \in \Pi_L$ by*

$$\forall t, a, s',\ \pi_t^{\text{new}}(a, s') = \arg\max_{a'} Q_t^{\bar{\pi}}(s', \Gamma^c_{t,a}(a')) \tag{8}$$

then $Q_t^{\bar{\pi}^{\text{new}}}(s,a) \geq Q_t^{\bar{\pi}}(s,a)$ holds for all t, s, a.

Remark. The time complexity of policy improvement step defined by Eq. (8) is $O(|\mathcal{S}||\mathcal{A}|^2)$ for each t, which has $|\mathcal{A}|$ times worse time complexity compared to the standard non-persistent policy improvement whose complexity is $O(|\mathcal{S}||\mathcal{A}|)$. Note also that the new policy π^{new} is not necessarily c-persistent, i.e. $\pi^{\mathrm{new}} \notin \Pi_c$ is possible, but the performance of its inducing c-persistent policy is always improved.

Finally, Theorems 1 and 2 lead us to a full algorithm, action-persistent policy iteration (AP-PI). AP-PI iterates between c-persistent policy evaluation by Eq. (6) and the c-persistent policy improvement of Eq. (8), and it is guaranteed to converge to the optimal c-persistent policy $\bar{\pi}^* \in \Pi_c$. The pseudo-code of AP-PI can be found in Appendix D.

Theorem 3. *Starting from any $\bar{\pi}^0 \in \Pi_c$ induced by L-periodic non-stationary deterministic policy $\pi^0 \in \Pi_L$, the sequence of value functions $Q^{\bar{\pi}^n}$ and the improved policies $\bar{\pi}^{n+1}$ induced by π^{n+1} converge to the optimal value function and the optimal c-persistent policy $\bar{\pi}^*$, i.e. $Q_t^{\bar{\pi}^*}(s,a) = \lim_{n\to\infty} Q_{t \bmod L}^{\bar{\pi}^n}(s,a) \geq Q_t^{\bar{\pi}}(s,a)$ for any $\bar{\pi} \in \Pi_c$, $t \in \mathbb{N}_0$, $s \in \mathcal{S}$, and $a \in \mathcal{A}$.*

Corollary 2. *There always exists a c-persistent optimal policy $\bar{\pi}_c^*$, which is induced by a L-periodic, non-stationary, and deterministic policy $\pi \in \Pi_L$.*

The policy $\bar{\pi}^* = (\bar{\pi}_0^*, \ldots, \bar{\pi}_{L-1}^*)$ obtained by AP-PI is executed as follows. First, $\bar{a}$ is initialized randomly. Then, at every step t, $a_t = \Gamma_{t,\bar{a}}^c(\bar{\pi}_{t \bmod L}^*(\bar{a}, s_t))$ is executed, and $\bar{a}$ is updated by $\bar{a} \leftarrow a_t$.

To the best of our knowledge, AP-PI is the first algorithm that addresses *multiple* action persistences, extending the *single* action persistence model that has been recently analyzed in [13]. AP-PI can be readily made scalable using the actor-critic architecture, to cope with large action spaces such as continuous actions, which we describe in the next section. This is a non-trivial extension of Persistent Fitted Q-iteration (PFQI) [13] which only applies to finite action spaces with single action persistence.

4 Action-Persistent Actor-Critic

In this section, we present Action-Persistent Actor-Critic (AP-AC), an off-policy RL algorithm that can be applied to high-dimensional tasks via practical approximation to AP-PI. AP-AC extends Soft Actor-Critic (SAC) [4] to perform iterative optimization of the parametric models of an L-periodic non-stationary policy (i.e. actor), and its c-persistent action-value function (i.e. critic). We assume that the action persistence vector $c = (c_1, \ldots, c_m)$ is given as a part of the environment specification.

As discussed in Section 3, the optimal c-persistent policy $\bar{\pi}$ can be induced by an L-periodic non-stationary policy $\pi = (\pi_0, \ldots, \pi_{L-1})$, where $\pi_t : \mathcal{A} \times \mathcal{S} \to \Delta(\mathcal{A})$ for all t. The corresponding optimal value function is also represented by the L-periodic action-value function $Q^{\bar{\pi}} = (Q_0^{\bar{\pi}}, \ldots, Q_{L-1}^{\bar{\pi}})$ with $Q_t^{\bar{\pi}} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ for all t. We exploit this structure of the optimal solution in the neural network architecture. Specifically, the parameterized actor network $\pi_\phi(\bar{a}, s)$ and the critic network $Q_\theta(s,a)$ are designed to have L heads, whose t-th head represents π_t and $Q_t^{\bar{\pi}}$ respectively, thus sharing the parameters of the lower layers among different t. The t-th head of the critic recursively references the $((t+1) \bmod L)$-th head for the target value, reflecting the result of Corollary 1.

The c-persistent value function is trained to minimize the squared temporal difference error:

$$J_Q(\theta) = \frac{1}{L} \sum_{t=0}^{L-1} \mathbb{E}_{\substack{(s,a,r,s') \sim \mathcal{D} \\ a' \sim \pi_{\phi,(t+1) \bmod L}(\cdot|a,s')}} \left[\Big(Q_{\theta,t}(s,a) - y_t(a,r,s',a') \Big)^2 \right] \tag{9}$$

$$\text{s.t. } y_t(a,r,s',a') = r + \gamma Q_{\bar{\theta},(t+1) \bmod L}\big(s', \Gamma_{t+1,a}^c(a')\big) - \alpha \log \pi_{\phi,(t+1) \bmod L}(a'|a,s'),$$

where $\mathcal{D}$ denotes the replay buffer, $\bar{\theta}$ is the parameters of the target network, and $\Gamma_{t,a}^c(a')$ is the action projection function defined in Eq. (4). This objective function is obtained from Eq. (5) with an (optional) entropy regularization term $\alpha \log \pi_{\phi,t}(a'|a,s')$, following the SAC formulation. Note that every term in Eq. (9) is agnostic to the actual time step when (s,a,r,s') was collected, which is due to the way we calculate y_t using $Q_{(t+1) \bmod L}$ and Γ. Thus every (s,a,r,s') sample in $\mathcal{D}$ can be used to train $Q_{\theta,t}$ regardless of t.

The policy parameters are then optimized by maximizing:

$$J_\pi(\phi) = \frac{1}{L} \sum_{t=0}^{L-1} \mathbb{E}_{\substack{(s,a,r,s') \sim \mathcal{D} \\ a' \sim \pi_{\phi,t}(\cdot|a,s')}} \left[Q_{\theta,t}(s', \Gamma_{t,a}^c(a')) - \alpha \log \pi_{\phi,t}(a'|a,s') \right] \tag{10}$$

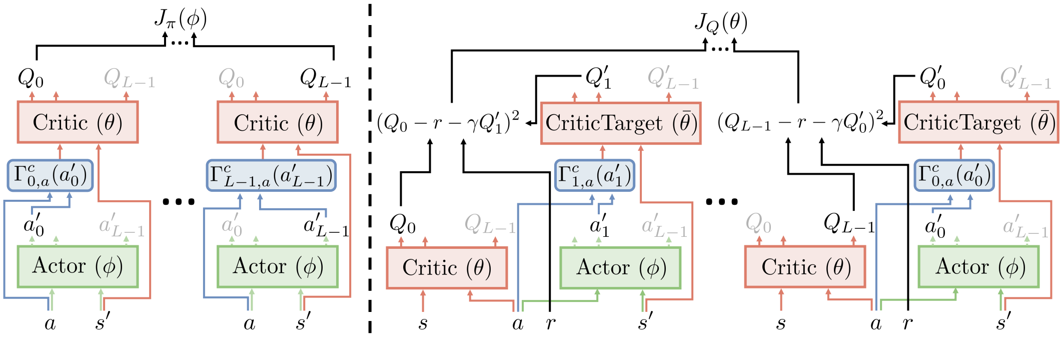


Figure 2: Overview of the network architectures and computational graphs for AP-AC training, given the (s, a, r, s') sample. The left figure corresponds to actor training of, Eq. (10), and the right figure to critic training of Eq. (9).

where the (optional) $\alpha \log \pi_{\phi,t}(a'|a, s')$ term comes from SAC formulation. In essence, maximizing $J_\pi(\phi)$ with respect to ϕ corresponds to c-persistent policy improvement by implementing Eq. (8) approximately. As with the case with critic, every term in Eq. (10) is agnostic to the actual time step t of when (s, a, r, s') was collected, thus every sample in $\mathcal{D}$ can be used to train $\pi_{\phi,t}$ for all t. The overall network architecture and the computational graph for training AP-AC are visualized in Figure 2. In order to obtain lower-variance gradient estimate $\hat{\nabla}_\phi J_\pi(\phi)$, we adopt the exact reparameterization [7] for continuous action tasks and the relaxed reparameterization with Gumbel-softmax [5, 12] for discrete action tasks. The rest of the design choices follows that of SAC such as the clipped double Q trick and soft target update. The pseudo-code for AP-AC can be found in Appendix E.

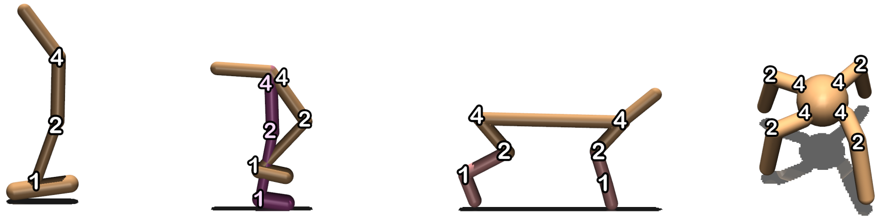
5 Related Works

Action Repetition in RL Recent deep RL algorithms have adopted action repetition to improve learning efficiency by reducing control granularity. Static action repetition, which repeats the same action over a fixed k time step, has been widely adopted in both on-policy [15] and off-policy [14] RL. Dynamic action repetition [9, 20] has also been explored to further improve the learning efficiency of online RL agents by adaptively changing the time scale of repeating actions per state. Recently, the notion of a single action-persistence has been formalized by introducing persistent Bellman operators, and its corresponding offline RL algorithm has been proposed along with a heuristic method for finding good persistence for empirical performance [13]. In contrast to the existing works that consider a *single* action-persistence, we deal with arbitrarily *multiple* action-persistence where each decision variable has its own persistence, and our goal is to provide an efficient solution method for the given action persistence c rather than finding a proper c to speed up learning.

Temporal Abstraction in RL The notion of action persistence is also naturally related to temporally abstract actions [16, 25] and semi-MDP framework [2]. Specifically, persisting actions with multiple frequencies can be seen as a particular instance of a semi-Markov *option* as follows: initiation set is the set of all states $\mathcal{I} = \mathcal{S}$, an internal policy is c-persistent $\pi \in \Pi_c$, and the termination condition is defined as $\beta(h_t) = \mathbb{1}_{\{t \bmod L = 0\}}$. Then, our off-policy learning scheme that exploits every transition sample to update every timestep's actor and critic in Eq. (9-10) can also be understood as an intra-option learning [24] method in the constructed semi-Markov option framework. Still, the cardinality of the set of possible options has an exponential growth with respect to L, thus obtaining an optimal policy over the set of options will be computationally inefficient compared to AP-PI that enjoys a linear complexity with respect to L.

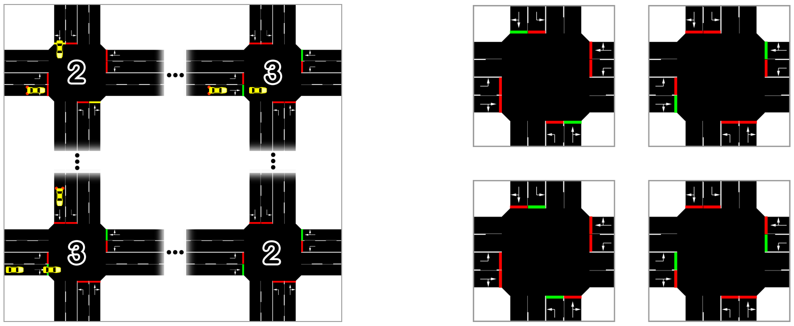
6 Experiments

We conducted a set of experiments in order to evaluate the effectiveness of AP-AC on high-dimensional tasks with different control frequencies. To the best of our knowledge, this work is the first to address multiple control frequencies in RL. Since there are no existing RL methods



(a) Hopper (b) Walker2d (c) HalfCheetah (d) Ant

Figure 3: Description of Mujoco continuous control tasks with multiple control frequencies. The number on each joint indicates the degree of persistence of each action. In other words, the action persistence vector for each domain is given as: $c = (4, 2, 1)$ for Hopper, $c = (4, 2, 1, 4, 2, 1)$ for Walker2d and HalfCheetah, and $c = (4, 2, 4, 2, 4, 2, 4, 2)$ for Ant.



(a) Domain description (b) Actions available for each junction.

Figure 4: Description of traffic light control problem. (a) We consider a 2x2 traffic simulation consisting of two rows and two columns of roads, where the traffic lights for the top-left and the bottom-right junctions have action persistence 2 while other junctions have action persistence 3, i.e. $c = (2, 3, 3, 2)$. (b) For each junction, there are 2^2 actions (i.e. the traffic light) where the green light allows the traffic to proceed and the red light prohibits any traffic from proceeding.

designed for multiple control frequencies, we take the variants of SAC as baselines for performance comparison, which are listed as follows: (1) **SAC**: this agent is trained on the standard *non-persistent* environment, while being evaluated on the environment where the action-persistence is enforced. This is intended to show the suboptimality of simply projecting a stationary policy to an action-persistent policy. (2) **SAC in AP-Env**: this agent is trained and evaluated on the action-persistent version of the environment, using the standard RL algorithm. This is to demonstrate the suboptimality of a stationary Markovian policy. (3) **SAC-L**: this agent takes a current observation, past L actions, and the one-hot indicator of the current time step ($t \bmod L$), which are sufficient for the optimal decision-making for the corresponding state augmentation approach discussed in Section 3. Still, this does not exploit the structure of the c-persistent optimal solution such as periodically recurrent policy/value representation and can take redundant information which is not fully compact. As a consequence, it is expected to show relatively weak performance. (4) **SAC-L-compact**: this agent takes a current observation, the last action which was *actually taken*, and the one-hot indicator of the current time step ($t \bmod L$), which is a compact representation of SAC-L. Still, this is unable to exploit every transition sample to update every timestep's actor and critic, while AP-AC is capable of doing it in Eq. (9-10). Therefore, it is expected to be less sample-efficient than AP-AC.

We conducted experiments on both continuous and discrete tasks, which will be detailed in the following section. The experimental setups including hyperparameters can be found in Appendix G.

6.1 Task description

Mujoco tasks (continuous action space) In many real-world situations, complex robotic systems consist of a number of controllers whose operating control frequencies vary due to the system

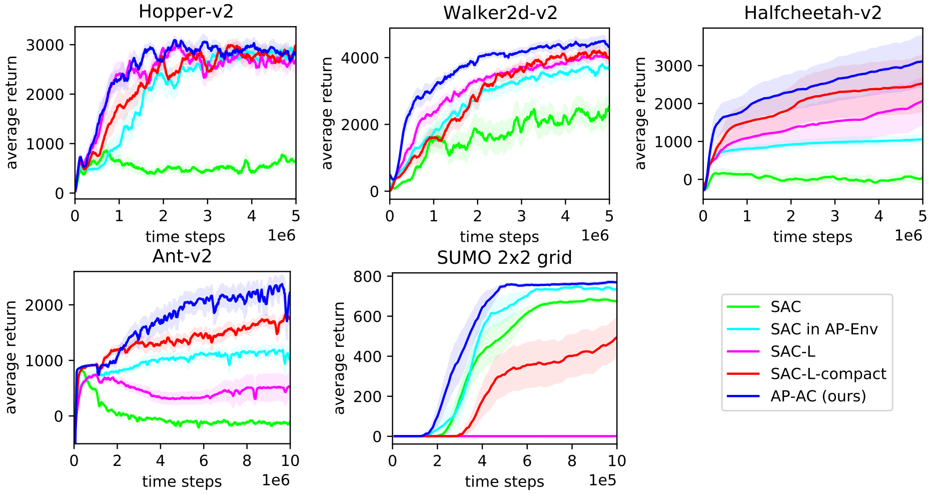


Figure 5: The results of experiments on the continuous control benchmarks and the traffic light control problems, which are averaged over 10 trials. The shaded area represents the standard error.

specification. In order to simulate this setting, we first conduct experiments on four OpenAI Gym continuous control tasks based on the Mujoco physics simulator [3, 26], where the controllable joints are modified to have different action persistence. Figure 3 depicts the detailed experimental setup for different action persistence for each task. For Hopper and Walker2d, action persistence for the thigh(s), the leg(s), and the foot(feet) are set to 4, 2, and 1 respectively. For HalfCheetah, action persistence for the thighs, the shins, and the feet are set to 4, 2, and 1 respectively. Finally, for Ant, the persistence for the hips and ankles are set to 4 and 2. We represent the policy π_t as the Gaussian with diagonal covariance matrix and $\tanh$-squashing function to bound the output in range $[-1, 1]$ for each dimension [4].

Traffic light control (discrete action space) We also tested AP-AC on a traffic control task, a realistic discrete-action sequential decision scenario with action persistence: in the traffic system, the control frequency of each traffic light can be different, for example depending on the number lanes and the speed limit. We use SUMO (Simulation of Urban MObility) [8] as the traffic simulator and SUMO-RL [1] for the environment interface. The specific instance we use is the implementation of 2X2GRID in SUMO-RL, which is depicted in Figure 4. The goal is to manipulate traffic lights located at each junction to improve the overall traffic flow, where the vehicles are generated randomly with a probability of 0.1 for every second at the end of the road.

The observation for each junction consists of the following four types of values: (1) the current traffic light status, represented by (4D one-hot), (2) the elapsed time from the current traffic light status, normalized within $[0, 1]$ (1D), (3) the density of all vehicles for each lane (8D), and (4) the density of stopped vehicles for each lane (8D). Therefore, the overall dimension of the observation space is $4 \times 21 = 81$. The action space is described in Figure 4b. The reward in the range [0, 1] is defined to be $\min_{i \in \{1,2,3,4\}} 1/(\text{waiting time of junction } i)$, with the goal of improving traffic flow of the junction of heaviest traffic. The length of episodes is 1000. We use a factorized (relaxed) categorical distribution to represent the policy with discrete action space, i.e. $\pi_{\phi,t}(a|\cdot) = \prod_{k=1}^{4} \mathrm{Cat}(a^k | p_\phi^k(\cdot))$ where $p_\phi(\cdot)$ denotes the probability vector with size 4. Though the cardinality of the entire joint action space is $|A| = 4^4 = 256$, the input and output dimensions to represent actions in the actor/critic networks are $4 \times 4 = 16$ (i.e. four one-hot vectors with size 4) since we are assuming fully factorized policies.

6.2 Results

We performed deterministic evaluation for each algorithm every 10K time steps, i.e. the peformance of the mean policy for continuous control and the greedy policy with respect to the categorical probabilities for the traffic control. The results are presented in Figure 5.

Since **SAC** (colored in green) is optimized for the non-persistent environment, its naive projection to the c-persistent policy suffers from severe performance degradation. In contrast, **SAC in AP-Env** (colored in cyan) interacts with the c-persistent environment directly while optimizing a stationary Markovian policy. Still, as discussed in Section 3, stationary Markovian policies can be suboptimal in general, which resulted in performing worse than AP-AC. **SAC-L** (colored in magenta) takes the past L-step actions and indicator of the current time step ($t \bmod L$), which is sufficient information for optimal c-persistent decision-making. Nonetheless, it does not exploit the structure of optimal c-persistent solution and can take redundant information since not all the past L-step actions are required for optimal decision-making, resulting in inefficient learning. This can be observed from the results that as the action dimension increases (Hopper (3) $\rightarrow$ Walker/Halfcheetah (6) $\rightarrow$ Ant (8) $\rightarrow$ SUMO2X2GRID (16)), the performance of SAC-L gets relatively worse. **SAC-L-compact** (colored in red) takes the last action actually taken, and indicator of the current time step ($t \bmod L$), which is also sufficient as well as compact information for optimal c-persistent decision-making, showing better performance than SAC-L in high-dimensional action tasks. Still, it is unable to exploit every transition sample to update every timestep's actor and critic, which leads to learning inefficiency compared to AP-AC. Finally, **AP-AC** significantly outperforms all of the baseline algorithms in all benchmark domains except for Hopper where AP-AC and baselines are on par. The experimental results highlight the effectiveness of our method that directly optimizes a periodic non-stationary policy for the tasks with multiple control frequencies.

7 Discussion and Conclusion

In this work, we formalized the notion of *multiple* action persistences in RL, which generalizes the result of [13] that deals with single action persistence. We introduced AP-PI, an efficient tabular planning algorithm for c-persistent policy for FA-MDP, and showed a formal analysis on its optimal convergence guarantee while it has only a marginal increase in the time complexity compared to the standard policy iteration. We then presented AP-AC, an off-policy deep reinforcement learning algorithm that scales, which directly exploits the structure of the optimal solution from the formal analysis on AP-PI. We empirically demonstrated that AP-AC significantly outperforms a number of strong baselines, both on continuous and discrete problems with action persistence. Extending the results of this work to multi-agent or hierarchical RL would be an interesting direction for future work.

Broader Impact

In recent years, reinforcement learning (RL) has shown remarkable successes in various areas, where most of their results are based on the assumption that all decision variables are simultaneously determined at every discrete time step. However, many real-world sequential decision-making problems involve multiple decision variables whose control frequencies are different by the domain requirement. In this situation, standard RL algorithms without considering the control frequency requirement may suffer from severe performance degradation as discussed in Section 3. This paper provides a theoretical and algorithmic foundation of how to address multiple control frequencies in RL, which enables RL to be applied to more complex and diverse real-world problems that involve decision variables with different frequencies. Therefore, this work would be beneficial for those who want to apply RL to various tasks that inherently have multiple control frequencies. As we provide a general-purpose methodology, we believe this work has little to do with a particular system failure or a particular data bias. On the other hand, this work could contribute to accelerating industrial adoption of RL, which has the potential to adversely affect employment due to automation.

Acknowledgments

This work was supported by the National Research Foundation (NRF) of Korea (NRF-2019R1A2C1087634 and NRF-2019M3F2A1072238), the Ministry of Science and Information communication Technology (MSIT) of Korea (IITP No. 2020-0-00940, IITP No. 2019-0-00075, IITP No. 2017-0-01779 XAI), and POSCO.

References

[1] Lucas N. Alegre. Sumo-rl. `https://github.com/LucasAlegre/sumo-rl`, 2019.

[2] Steven J. Bradtke and Michael O. Duff. Reinforcement learning methods for continuous-time markov decision problems. In *Proceedings of the 7th International Conference on Neural Information Processing Systems*, page 393–400, 1994.

[3] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[4] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the International Conference on Machine Learning*, pages 1856–1865, 2018.

[5] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017.

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[7] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations (ICLR)*, 2014.

[8] Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker. Recent development and applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, 5(3&4):128–138, December 2012.

[9] Aravind Lakshminarayanan, Sahil Sharma, and Balaraman Ravindran. Dynamic action repetition for deep reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2017.

[10] Boris Lesner and Bruno Scherrer. Non-stationary approximate modified policy iteration. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1567–1575, 2015.

[11] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[12] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *ICLR*, 2017.

[13] Alberto Maria Metelli, Flavio Mazzolini, Lorenzo Bisi, Luca Sabbioni, and Marcello Restelli. Control frequency adaptation via action persistence in batch reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, 2020.

[14] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[15] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016.

[16] Doina Precup. *Temporal Abstraction in Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 2000.

[17] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

[18] Bruno Scherrer and Boris Lesner. On the use of non-stationary policies for stationary infinite-horizon markov decision processes. In *Advances in Neural Information Processing Systems 25*, pages 1826–1834, 2012.

[19] John Schulman, Sergey Levine, Philipp Moritz, Michael Jordan, and Pieter Abbeel. Trust region policy optimization. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, page 1889–1897, 2015.

[20] Sahil Sharma, Aravind S. Lakshminarayanan, and Balaraman Ravindran. Learning to repeat: Fine grained action repetition for deep reinforcement learning. In *ICLR*, 2017.

[21] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, pages 484–489, 2016.

[22] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550:354–359, 2017.

[23] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[24] Richard S. Sutton, Doina Precup, and Satinder P. Singh. Intra-option learning about temporally abstract actions. In *ICML*, pages 556–564, 1998.

[25] Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.*, 112(1–2):181–211, August 1999.

[26] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

[27] Ari Weinstein and Michael Littman. Bandit-based planning and learning in continuous-action markov decision processes. In *Proceedings of the 22nd International Conference on Automated Planning and Scheduling*, 2012.

Reinforcement Learning for Control with Multiple Frequencies - Supplementary Material

(Jongmin Lee, Byung-Jun Lee, Kee-Eung Kim)

Appendix A Proof of Theorem 1

Lemma 1. *For any $\pi \in \Pi_L$, $t \in \{0, \dots, L-1\}$, and $k \in \mathbb{N}$, the composition of k one-step c-persistent Bellman operators $(\bar{\mathcal{T}}_t^{\pi} \cdots \bar{\mathcal{T}}_{(t+k-1) \text{ mod } L}^{\pi})$ satisfies:*

$$(\bar{\mathcal{T}}_t^{\pi} \cdots \bar{\mathcal{T}}_{(t+k-1) \text{ mod } L}^{\pi} Q)(s,a) = \mathbb{E}_{\forall \tau, s_{\tau+1} \sim P(\cdot|s_\tau, \bar{a}_\tau)} \left[\sum_{\tau=t}^{t+k-1} \gamma^{\tau-t} R(s_\tau, \bar{a}_\tau) + \gamma^k Q(s_{t+k}, \bar{a}_{t+k}) \,\middle|\, s_t = s,\ \bar{a}_t = a \right]$$
$$\textit{where } \bar{a}_{\tau+1} = \Gamma^c_{\tau+1, \bar{a}_\tau}(\pi_{(\tau+1) \text{ mod } L}(\bar{a}_\tau, s_{\tau+1}))\ \textit{for } \tau = t, \dots, t+k-1$$

Proof. We give a proof based on induction. For $k = 1$,

$$(\bar{\mathcal{T}}_t^{\pi} Q)(s,a) = \mathbb{E}_{s_{t+1} \sim P(\cdot|s_t, \bar{a}_t)} \left[R(s_t, \bar{a}_t) + \gamma Q(s_{t+1}, \bar{a}_{t+1}) \mid s_t = s, \bar{a}_t = a \right]$$
$$\text{where } \bar{a}_{t+1} = \Gamma^c_{t+1, \bar{a}_t}(\pi_{(t+1) \text{ mod } L}(\bar{a}_t, s_{t+1}))$$

holds by the definition of one-step c-persistent Bellman operator $\bar{\mathcal{T}}_t^{\pi}$ (Eq. (5)). Now, assume the induction hypothesis for k. Then,

$$\begin{aligned}
&\Big(\bar{\mathcal{T}}_t^{\pi} \cdots \bar{\mathcal{T}}_{(t+k-1) \text{ mod } L} (\bar{\mathcal{T}}_{(t+k) \text{ mod } L}^{\pi} Q)\Big)(s,a) \\
=&\mathbb{E}_{\forall \tau, s_{\tau+1} \sim P(\cdot|s_\tau, \bar{a}_\tau)} \left[\sum_{\tau=t}^{t+k-1} \gamma^{\tau-t} R(s_\tau, \bar{a}_\tau) + \gamma^k (\bar{\mathcal{T}}_{(t+k) \text{ mod } L}^{\pi} Q)(s_{t+k}, \bar{a}_{t+k}) \,\middle|\, s_t = s,\ \bar{a}_t = a \right] \\
&\text{where } \bar{a}_{\tau+1} = \Gamma^c_{\tau+1, \bar{a}_\tau}(\pi_{(\tau+1) \text{ mod } L}(\bar{a}_\tau, s_{\tau+1})) \text{ for } \tau = t, \dots, t+k-1 \\
&\text{(by the induction hypothesis)} \\
=&\mathbb{E}_{\forall \tau, s_{\tau+1} \sim P(\cdot|s_\tau, \bar{a}_\tau)} \Bigg[\sum_{\tau=t}^{t+k-1} \gamma^{\tau-t} R(s_\tau, \bar{a}_\tau) + \gamma^k \Big(R(s_{t+k}, \bar{a}_{t+k}) \\
&\qquad\qquad + \gamma Q\big(s_{t+k+1}, \Gamma^c_{t+k+1, \bar{a}_{t+k}}(\pi_{(t+k+1) \text{ mod } L}(\bar{a}_{t+k}, s_{t+k+1}))\big)\Big) \,\Big|\, s_t = s,\ \bar{a}_t = a \Bigg] \\
&\text{where } \bar{a}_{\tau+1} = \Gamma^c_{\tau+1, \bar{a}_\tau}(\pi_{(\tau+1) \text{ mod } L}(\bar{a}_\tau, s_{\tau+1})) \text{ for } \tau = t, \dots, t+k-1 \\
=&\mathbb{E}_{\forall \tau, s_{\tau+1} \sim P(\cdot|s_\tau, \bar{a}_\tau)} \left[\sum_{\tau=t}^{t+k} \gamma^{\tau-t} R(s_\tau, \bar{a}_\tau) + \gamma^{k+1} Q(s_{t+k+1}, \bar{a}_{t+k+1}) \,\middle|\, s_t = s,\ \bar{a}_t = a \right] \\
&\text{where } \bar{a}_{\tau+1} = \Gamma^c_{\tau+1, \bar{a}_\tau}(\pi_{(\tau+1) \text{ mod } L}(\bar{a}_\tau, s_{\tau+1})) \text{ for } \tau = t, \dots, t+k
\end{aligned}$$

thus the given statement holds for $k+1$, which concludes the proof. □

Theorem 1. *For all $t \in \{0, \dots, L-1\}$, the L-step c-persistent Bellman operator $\bar{H}_t^{\pi}$ is γ^L-contraction with respect to infinity norm, thus $\bar{H}_t^{\pi} Q_t^{\bar{\pi}} = Q_t^{\bar{\pi}}$ has the unique fixed point solution. In other words, for any $Q_t^0 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, define $Q_t^{n+1} = \bar{H}_t^{\pi} Q_t^n$. Then, the sequence Q_t^n converges to t-th c-persistent value function of $\bar{\pi}$ as $n \to \infty$.*

Proof. By Lemma 1, for any t, s, a, and $Q_1 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $Q_2 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$,

$$\begin{aligned}
\Big| \bar{H}_t^{\pi} Q_1(s,a) - \bar{H}_t^{\pi} Q_2(s,a) \Big| &= \Big| \mathbb{E}_{\forall \tau, s_{\tau+1} \sim P(\cdot|s_\tau, \bar{a}_\tau)} \left[\gamma^L Q_1(s_{t+L}, \bar{a}_{t+L}) - \gamma^L Q_2(s_{t+L}, \bar{a}_{t+L}) \mid s_t = s, \bar{a}_t = a \right] \Big| \\
&\quad \text{where } \bar{a}_{\tau+1} = \Gamma^c_{\tau+1, \bar{a}_\tau}(\pi_{(\tau+1) \text{ mod } L}(\bar{a}_\tau, s_{\tau+1})) \text{ for } \tau = t, \dots, t+L-1 \\
&\le \gamma^L \max_{s', a'} \Big| Q_1(s', a') - Q_2(s', a') \Big| \\
\therefore \Big\| \bar{H}_t^{\pi} Q_1 - \bar{H}_t^{\pi} Q_2 \Big\|_\infty &\le \gamma^L \Big\| Q_1 - Q_2 \Big\|_\infty
\end{aligned}$$

Therefore, $\bar{H}_t^{\pi}$ is γ^L-contraction with respect to infinity norm, and by Banach fixed-point theorem, $\bar{H}_t^{\pi} Q_t^{\bar{\pi}} = Q_t^{\bar{\pi}}$ has the unique fixed point solution for all t. □

A deeper discussion on the Bellman operators with a periodic non-stationary policy can be found in [10, 18] though it analyzes the error in approximate policy/value iterations, rather than considering action persistence.

Corollary 1. *$Q^{\bar{\pi}}_t = \bar{\mathcal{T}}^{\pi}_t Q^{\bar{\pi}}_{(t+1) \bmod L}$ holds for all $t \in \{0, \dots, L-1\}$, thus c-persistent value functions can be obtained by repeatedly applying 1-step c-persistent backup in a L-cyclic manner.*

Proof.

$$
\begin{aligned}
Q^{\bar{\pi}}_t &= \bar{H}^{\pi}_t Q^{\bar{\pi}}_t = \bar{H}^{\pi}_t \bar{H}^{\pi}_t Q^{\bar{\pi}}_t = \bar{H}^{\pi}_t \bar{H}^{\pi}_t \bar{H}^{\pi}_t Q^{\bar{\pi}}_t = \cdots && \text{(by Theorem 1)} \\
&= \underbrace{\bar{\mathcal{T}}^{\pi}_t \bar{\mathcal{T}}^{\pi}_{(t+1) \bmod L} \cdots \bar{\mathcal{T}}^{\pi}_{(t+L-1) \bmod L}}_{\bar{H}^{\pi}_t} \underbrace{\bar{\mathcal{T}}^{\pi}_t \bar{\mathcal{T}}^{\pi}_{(t+1) \bmod L} \cdots \bar{\mathcal{T}}^{\pi}_{(t+L-1) \bmod L}}_{\bar{H}^{\pi}_t} Q^{\bar{\pi}}_t \\
&= \bar{\mathcal{T}}^{\pi}_t \underbrace{\bar{\mathcal{T}}^{\pi}_{(t+1) \bmod L} \cdots \bar{\mathcal{T}}^{\pi}_{(t+L-1) \bmod L} \bar{\mathcal{T}}^{\pi}_t}_{\bar{H}^{\pi}_{(t+1) \bmod L}} \underbrace{\bar{\mathcal{T}}^{\pi}_{(t+1) \bmod L} \cdots \bar{\mathcal{T}}^{\pi}_{(t+L-1) \bmod L} Q^{\bar{\pi}}_t}_{\triangleq Q} \\
&= \bar{\mathcal{T}}^{\pi}_t \bar{H}^{\pi}_{(t+1) \bmod L} Q = \cdots = \bar{\mathcal{T}}^{\pi}_t \lim_{n \to \infty} (\bar{H}^{\pi}_{(t+1) \bmod L})^n Q \\
&= \bar{\mathcal{T}}^{\pi}_t Q^{\bar{\pi}}_{(t+1) \bmod L} && \text{(by Theorem 1)}
\end{aligned}
$$

□

Appendix B Proof of Theorem 2

Theorem 2. *Given a L-periodic, non-stationary, and deterministic policy $\pi = (\pi_0, \dots, \pi_{L-1}) \in \Pi_L$, let $Q^{\bar{\pi}}_t$ be the c-persistent value of $\bar{\pi}$ denoted in Eq. (7). If we update the new policy $\pi^{\text{new}} = (\pi^{\text{new}}_0, \dots, \pi^{\text{new}}_{L-1}) \in \Pi_L$ by*

$$
\forall t, a, s', \ \pi^{\text{new}}_t(a, s') = \arg\max_{a'} Q^{\bar{\pi}}_t(s', \Gamma^c_{t,a}(a')) \tag{8}
$$

then $Q^{\bar{\pi}^{\text{new}}}_t(s, a) \geq Q^{\bar{\pi}}_t(s, a)$ holds for all t, s, a.

Proof. For any t, s, a,

$$
\begin{aligned}
&Q^{\bar{\pi}}_t(s, a) \\
&= \mathbb{E}_P \left[R(s_t, \bar{a}_t) + \gamma Q^{\bar{\pi}}_{(t+1) \bmod L}(s_{t+1}, \Gamma^c_{t+1,\bar{a}_t}(\pi_{(t+1) \bmod L}(\bar{a}_t, s_{t+1}))) \mid s_t = s, \bar{a}_t = a \right] \\
&\leq \mathbb{E}_P \left[R(s_t, \bar{a}_t) + \gamma Q^{\bar{\pi}}_{(t+1) \bmod L}\big(s_{t+1}, \Gamma^c_{t+1,\bar{a}_t}(\pi^{\text{new}}_{(t+1) \bmod L}(\bar{a}_t, s_{t+1}))\big) \mid s_t = s, \bar{a}_t = a \right] \quad \text{(by Eq. (8))} \\
&= \mathbb{E}_P \left[R(s_t, \bar{a}_t) + \gamma \Big(R(s_{t+1}, \bar{a}_{t+1}) + \gamma Q^{\bar{\pi}}_{(t+2) \bmod L}(s_{t+2}, \Gamma^c_{t+2,\bar{a}_{t+1}}(\pi_{(t+2) \bmod L}(\bar{a}_{t+1}, s_{t+2}))) \Big) \mid s_t = s, \bar{a}_t = a \right] \\
&\qquad \text{where } \bar{a}_{t+1} = \Gamma^c_{t+1,\bar{a}_t}(\pi^{\text{new}}_{(t+1) \bmod L}(\bar{a}_t, s_{t+1})) \\
&= \mathbb{E}_P \left[\sum_{\tau=t}^{t+1} \gamma^{\tau-t} R(s_\tau, \bar{a}_\tau) + \gamma^2 Q^{\bar{\pi}}_{(t+2) \bmod L}\big(s_{t+2}, \Gamma^c_{t+2,\bar{a}_{t+1}}(\pi_{(t+2) \bmod L}(\bar{a}_{t+1}, s_{t+2}))\big) \mid s_t = s, \bar{a}_t = a \right] \\
&\qquad \text{where } \bar{a}_{\tau+1} = \Gamma^c_{\tau+1,\bar{a}_\tau}(\pi^{\text{new}}_{(\tau+1) \bmod L}(\bar{a}_\tau, s_{\tau+1})) \text{ for } \tau = t \\
&\leq \mathbb{E}_P \left[\sum_{\tau=t}^{t+1} \gamma^{\tau-t} R(s_\tau, \bar{a}_\tau) + \gamma^2 Q^{\bar{\pi}}_{(t+2) \bmod L}\big(s_{t+2}, \Gamma^c_{t+2,\bar{a}_{t+1}}(\pi^{\text{new}}_{(t+2) \bmod L}(\bar{a}_{t+1}, s_{t+2}))\big) \mid s_t = s, \bar{a}_t = a \right] \\
&\qquad \text{where } \bar{a}_{\tau+1} = \Gamma^c_{\tau+1,\bar{a}_\tau}(\pi^{\text{new}}_{(\tau+1) \bmod L}(\bar{a}_\tau, s_{\tau+1})) \text{ for } \tau = t \\
&= \mathbb{E}_P \left[\sum_{\tau=t}^{t+2} \gamma^{\tau-t} R(s_\tau, \bar{a}_\tau) + \gamma^3 Q^{\bar{\pi}}_{(t+3) \bmod L}\big(s_{t+3}, \Gamma^c_{t+3,\bar{a}_{t+2}}(\pi_{(t+3) \bmod L}(\bar{a}_{t+2}, s_{t+3}))\big) \mid s_t = s, \bar{a}_t = a \right] \\
&\qquad \text{where } \bar{a}_{\tau+1} = \Gamma^c_{\tau+1,\bar{a}_\tau}(\pi^{\text{new}}_{(\tau+1) \bmod L}(\bar{a}_\tau, s_{\tau+1})) \text{ for } \tau = t, \ t+1 \\
&\ \vdots \\
&\leq \mathbb{E}_P \left[\sum_{\tau=t}^{\infty} \gamma^{\tau-t} R(s_\tau, \bar{a}_\tau) \mid s_t = s, \bar{a}_t = a \right] \\
&\qquad \text{where } \bar{a}_{\tau+1} = \Gamma^c_{\tau+1,\bar{a}_\tau}(\pi^{\text{new}}_{(\tau+1) \bmod L}(\bar{a}_\tau, s_{\tau+1})) \text{ for } \tau = t, \ t+1, \ t+2, \dots \\
&= Q^{\bar{\pi}^{\text{new}}}_t(s, a)
\end{aligned}
$$

where each of inequalities holds by Eq. (8), and this concludes the proof. □

Appendix C Proof of Theorem 3

We first define the following one-step c-persistent Bellman *optimality* operator:

$$(\bar{\mathcal{T}}_t^* Q)(s,a) \triangleq \mathbb{E}_{s' \sim P(\cdot|s,a)} \Big[R(s,a) + \gamma \max_{a'} Q(s', \Gamma^c_{t+1,a}(a')) \Big] \tag{11}$$

Note that the one-step c-persistent Bellman optimality operators are L-periodic with respect to t due to the L-periodic nature of the projection operator $\Gamma^c_{t,a}(a')$. Therefore, $\bar{\mathcal{T}}_t^* Q = \bar{\mathcal{T}}_{t+L}^* Q$ always holds for any t and Q. Then, similar to Eq. (6), we define an L-step c-persistent Bellman *optimality* operator $\bar{H}_t^*$ by making the composition of L one-step c-persistent Bellman optimality operators:

$$\begin{aligned}
(\bar{H}_0^* Q)(s,a) &\triangleq (\bar{\mathcal{T}}_0^* \bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-2}^* \bar{\mathcal{T}}_{L-1}^* Q)(s,a) \\
(\bar{H}_1^* Q)(s,a) &\triangleq (\bar{\mathcal{T}}_1^* \bar{\mathcal{T}}_2^* \cdots \bar{\mathcal{T}}_{L-1}^* \bar{\mathcal{T}}_0^* Q)(s,a) \\
&\vdots \\
(\bar{H}_{L-1}^* Q)(s,a) &\triangleq (\bar{\mathcal{T}}_{L-1}^* \bar{\mathcal{T}}_0^* \cdots \bar{\mathcal{T}}_{L-3}^* \bar{\mathcal{T}}_{L-2}^* Q)(s,a)
\end{aligned} \tag{12}$$

Similar to L-step c-persistent Bellman operators, we can show that L-step c-persistent Bellman *optimality* operators are contraction mapping.

Lemma 2. *For all $t \in \{0, \dots, L-1\}$, the L-step c-persistent Bellman optimality operator $\bar{H}_t^*$ is γ^L-contraction with respect to infinity norm, thus $\bar{H}_t^* Q_t^{\bar{*}} = Q_t^{\bar{*}}$ has the unique fixed point solution. In other words, for any $Q_t^0 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, define $Q_t^{n+1} = \bar{H}_t^* Q_t^n$. Then, the sequence Q_t^n converges to t-th c-persistent optimal value function as $n \to \infty$.*

Proof. Without loss of generality, it is sufficient to prove when $t = 0$.

For any $Q_1 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, $Q_2 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and $s_0 \in \mathcal{S}$, $a_0 \in \mathcal{A}$,

$$\begin{aligned}
&|(\bar{H}_0^* Q_1)(s_0,a_0) - (\bar{H}_0^* Q_2)(s_0,a_0)| \\
&= |(\bar{\mathcal{T}}_0^* \bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_1)(s_0,a_0) - (\bar{\mathcal{T}}_0^* \bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_2)(s_0,a_0)| \\
&= \Bigg| \mathbb{E}_{s_1 \sim P(\cdot|s_0,a_0)} \Big[R(s_0,a_0) + \gamma \max_{a_1} (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_1)(s_1, \Gamma^c_{1,a_0}(a_1)) \Big] \\
&\qquad - \mathbb{E}_{s_1 \sim P(\cdot|s_0,a_0)} \Big[R(s_0,a_0) + \gamma \max_{a_1} (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_2)(s_1, \Gamma^c_{1,a_0}(a_1)) \Big] \Bigg| \\
&= \gamma \Bigg| \mathbb{E}_P \Big[\max_{a_1} (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_1)(s_1, \Gamma^c_{1,a_0}(a_1)) - \max_{a_1} (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_2)(s_1, \Gamma^c_{1,a_0}(a_1)) \Big] \Bigg| \\
&\leq \gamma \Bigg| \mathbb{E}_P \Big[(\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_1)(s_1, a_1^*) - (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_2)(s_1, a_1^*) \Big] \Bigg| \\
&\qquad \text{where } a_1^* = \arg\max_a \big[(\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_1)(s_1, \Gamma^c_{1,a_0}(a)) - (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_2)(s_1, \Gamma^c_{1,a_0}(a)) \big] \\
&\leq \gamma \max_{s,a} \Bigg| (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_1)(s,a) - (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_2)(s,a) \Bigg|
\end{aligned}$$

We can continue to expand the inequality in a similar way,

$$\begin{aligned}
\forall s_0, a_0,\ |(\bar{H}_0^* Q_1)(s_0,a_0) - (\bar{H}_0^* Q_1)(s_0,a_0)| &\leq \gamma \max_{s,a} \Bigg| (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_1)(s,a) - (\bar{\mathcal{T}}_1^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_2)(s,a) \Bigg| \\
&\leq \gamma^2 \max_{s,a} \Bigg| (\bar{\mathcal{T}}_2^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_1)(s,a) - (\bar{\mathcal{T}}_2^* \cdots \bar{\mathcal{T}}_{L-1}^* Q_2)(s,a) \Bigg| \\
&\ \ \vdots \\
&\leq \gamma^L \max_{s,a} \Bigg| Q_1(s,a) - Q_2(s,a) \Bigg| \\
\therefore \|\bar{H}_0^* Q_1 - \bar{H}_0^* Q_2\|_\infty &\leq \gamma^L \|Q_1 - Q_2\|_\infty
\end{aligned}$$

Therefore, $\bar{H}_t^*$ is γ^L-contraction with respect to infinity norm, and by Banach fixed-point theorem, $\bar{H}_t^* Q_t^{\bar{*}} = Q_t^{\bar{*}}$ has the unique fixed point solution for all t. □

Therefore, the optimal c-persistent value functions (i.e. the fixed points of each $\bar{H}_0^*, \dots \bar{H}_{L-1}^*$) can be represented by L values, $(Q_0^{\bar{*}}, \dots, Q_{L-1}^{\bar{*}})$. Also, the following lemma shows that they have the largest possible value, compared to any c-persistent value functions of any history-dependent policy $\pi \in \Pi$.

Lemma 3. *For any t, let $\bar{H}^*_{t \bmod L} = \bar{\mathcal{T}}^*_{t \bmod L} \cdots \bar{\mathcal{T}}^*_{(t+L-1) \bmod L}$ be L-step c-persistent Bellman optimality operator and $Q^{\bar{*}}_{t \bmod L}$ be its fixed point. Then, for any history-dependent policy $\pi \in \Pi$, $Q^{\bar{*}}_{t \bmod L}(s,a) \geq Q^{\bar{\pi}}_t(s,a)$ holds for all t, s, a.*

Proof. For any $\pi \in \Pi$, $t \in \mathbb{N}_0$, $s \in \mathcal{S}$, $a \in \mathcal{A}$, and $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, the following inequality holds:

$$\begin{aligned}(\bar{\mathcal{T}}_t^\pi Q)(s_t, a_t) &\triangleq R(s_t, a_t) + \gamma \mathbb{E}_{\substack{s_{t+1} \sim P(\cdot|s_t, a_t) \\ a_{t+1} \sim \pi(\cdot|h_{t+1})}} \left[Q(s_{t+1}, \Gamma^c_{t+1, a_t}(a_{t+1}) \right] \\ &\leq R(s_t, a_t) + \gamma \max_{a'} \mathbb{E}_{s_{t+1} \sim P(\cdot|s_t, a_t)} \left[Q(s_{t+1}, \Gamma^c_{t+1, a_t}(a')) \right] \\ &= (\bar{\mathcal{T}}^*_{t \bmod L} Q)(s_t, a_t)\end{aligned}$$

which implies

$$(\bar{\mathcal{T}}_t^\pi \bar{\mathcal{T}}_{t+1}^\pi \cdots \bar{\mathcal{T}}_{t+L-1}^\pi Q)(s,a) \leq (\bar{\mathcal{T}}^*_{t \bmod L} \bar{\mathcal{T}}^*_{(t+1) \bmod L} \cdots \bar{\mathcal{T}}^*_{(t+L-1) \bmod L} Q)(s,a) = (\bar{H}^*_{t \bmod L} Q)(s,a)$$

Therefore, $Q_t^{\bar{\pi}}(s,a) = \lim_{n \to \infty} (\bar{\mathcal{T}}_t^\pi \bar{\mathcal{T}}_{t+1}^\pi \cdots \bar{\mathcal{T}}_{t+Ln-1}^\pi Q)(s,a) \leq \lim_{n \to \infty} ((\bar{H}^*_{t \bmod L})^n Q)(s,a) = Q^{\bar{*}}_{t \bmod L}(s,a)$ holds for any t, s, a and history-dependent policy π, which concludes the proof. □

Now, we are ready to provide the proof of Theorem 3.

Theorem 3. *Starting from any $\bar{\pi}^0 \in \Pi_c$ induced by L-periodic non-stationary deterministic policy $\pi^0 \in \Pi_L$, the sequence of value functions $Q^{\bar{\pi}^n}$ and the improved policies $\bar{\pi}^{n+1}$ induced by π^{n+1} converge to the optimal value function and the optimal c-persistent policy $\bar{\pi}^*$, i.e. $Q_t^{\bar{\pi}^*}(s,a) = \lim_{n \to \infty} Q^{\bar{\pi}^n}_{t \bmod L}(s,a) \geq Q_t^{\bar{\pi}}(s,a)$ for any $\bar{\pi} \in \Pi_c$, $t \in \mathbb{N}_0$, $s \in \mathcal{S}$, and $a \in \mathcal{A}$.*

Proof. By Lemma 3, it is sufficient to show $\lim_{n \to \infty} Q_t^{\bar{\pi}^n} = Q_t^{\bar{*}}$ for all $t \in \{0, \dots, L-1\}$. By Theorem 2, the performance of c-persistent policy induced by π^n is monotonically improved during policy iteration, i.e. $Q_t^{\bar{\pi}^{n+1}}(s,a) \geq Q_t^{\bar{\pi}^n}(s,a)$ always holds for all t, s, a, n. Now, consider when the policy is no longer improved, i.e. $\bar{\pi}^{n+1} = \bar{\pi}^n$ and $Q_t^{\bar{\pi}^{n+1}} = Q_t^{\bar{\pi}^n}$. In this situation, for all t, s, a,

$$\begin{aligned}Q_t^{\bar{\pi}^n}(s,a) &= Q_t^{\bar{\pi}^{n+1}}(s,a) \\ &= R(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[Q^{\bar{\pi}^{n+1}}_{(t+1) \bmod L}(s', \Gamma^c_{t+1,a}(\bar{\pi}^{n+1}(a, s'))) \right] \\ &= R(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[Q^{\bar{\pi}^n}_{(t+1) \bmod L}(s', \Gamma^c_{t+1,a}(\bar{\pi}^{n+1}(a, s'))) \right] \\ &= R(s,a) + \gamma \max_{a'} \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[Q^{\bar{\pi}^n}_{(t+1) \bmod L}(s', \Gamma^c_{t+1,a}(a')) \right] \qquad \text{(by Eq. (8))}\end{aligned}$$

holds, and this implies that $Q_t^{\bar{\pi}^n}$ satisfies the c-persistent Bellman optimality equation. By Lemma 2, the c-persistent Bellman optimality equation has the unique solution, thus $Q_t^{\bar{\pi}^n} = Q_t^{\bar{*}}$. This concludes that $\bar{\pi}^n$ is the optimal c-persistent policy. □

Corollary 2. *There always exists a c-persistent optimal policy $\bar{\pi}_c^*$, which is induced by a L-periodic, non-stationary, and deterministic policy $\pi \in \Pi_L$.*

Proof. Every policy π^n encountered during action-persistent policy iteration is within Π_L. Also, by Theorem 3, $\bar{\pi}^n \in \Pi_c$ induced by $\pi^n \in \Pi_L$ eventually converges to the optimal c-persistent policy, which concludes the proof. □

Corollary 2 ensures that the optimal c-persistent policy can be always found only through $(\pi_0, \dots, \pi_{L-1}) \in \Pi_L$, where $\pi_t : \mathcal{A} \times \mathcal{S} \to \mathcal{A}$.

Appendix D Pseudo-code of Action-Persistent Policy Iteration (AP-PI)

Algorithm 1 Action-Persistent Policy Iteration (AP-PI)

```
Input: $\mathcal{M}$: FA-MDP, $c$: action persistence vector
  Randomly initialize $\pi = (\pi_0, \ldots, \pi_{L-1})$ where $\pi_t : \mathcal{A} \times \mathcal{S} \to \mathcal{A}$ for all $t = 0, \ldots, L-1$.
  Randomly initialize $Q = (Q_0, \ldots, Q_{L-1})$ where $Q_t : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ for all $t = 0, \ldots, L-1$.
  repeat
      # Policy Evaluation
      repeat
          for $t = 0, \ldots, L-1$ do
              $\forall s, a,\ Q_t(s,a) \leftarrow R(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[ Q_{(t+1) \bmod L}(s', a') \right]$
                    where $a' = \Gamma^c_{t+1,a}(\pi_{(t+1) \bmod L}(a, s'))$
          end for
      until $Q$ is converged
      # Policy Improvement
      $\forall t, a, s',\ \pi_t(a, s') \leftarrow \arg\max_{a'} Q_t(s', \Gamma^c_{t,a}(a'))$
  until $\pi$ does not change
Output: $\bar{\pi}^* = \bar{\pi}$
```

The policy $\bar{\pi}^* = (\bar{\pi}^*_0, \ldots, \bar{\pi}^*_{L-1})$ obtained by AP-PI is executed as follows. First, $\bar{a}$ is initialized randomly. Then, at every step t, $a_t = \Gamma^c_{t,\bar{a}}(\bar{\pi}^*_{t \bmod L}(\bar{a}, s_t))$ is executed, and the reward and the next state is observed: $r_t, s_{t+1} \sim p(r_t, s_{t+1}|s_t, a_t)$. Finally, $\bar{a}$ is updated by $\bar{a} \leftarrow a_t$, and this procedure continues.

Appendix E Pseudo-Code of Action-Persistent Actor-Critic

Algorithm 2 Action-Persistent Actor-Critic (AP-AC)

```
Input: $\theta_1, \theta_2, \phi$                                  ▷ Initialize parameters
  $\bar{\theta}_1 \leftarrow \theta_1$ and $\bar{\theta}_2 \leftarrow \theta_2$                ▷ Initialize target network weights
  $\bar{a} \sim \mathrm{unif}(\mathcal{A})$                                ▷ Initialize $\bar{a}$ randomly
  $\mathcal{D} \leftarrow \emptyset$                                 ▷ Initialize a replay buffer to an empty set
  for each iteration do
      for each environment step do
          $a_t \sim \pi_{\phi,t}(\cdot|\bar{a}, s_t)$                 ▷ Sample a non-persistent action from the policy
          $\bar{a} \leftarrow \Gamma_{t,\bar{a}}(a_t)$                      ▷ Project the sampled action using Eq. (4)
          $r_t, s_{t+1} \sim p(r_t, s_{t+1}|s_t, \bar{a})$          ▷ Sample reward and transition from the environment
          $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, \bar{a}, r_t, s_{t+1})\}$    ▷ Store the sampled reward and transition into replay buffer
      end for
      for each gradient step do
          $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$    ▷ Update critic weights by minimizing Eq. (9)
          $\phi \leftarrow \phi + \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$            ▷ Update policy weights by maximizing Eq. (10)
          $\bar{\theta}_i \leftarrow \tau \theta_i + (1 - \tau) \bar{\theta}_i$ for $i \in \{1, 2\}$       ▷ Update target network weights
      end for
  end for
Output: $\theta_1, \theta_2, \phi$                                 ▷ Optimized parameters
```

Appendix F Supplementary Experiments

F.1 Results on SAC

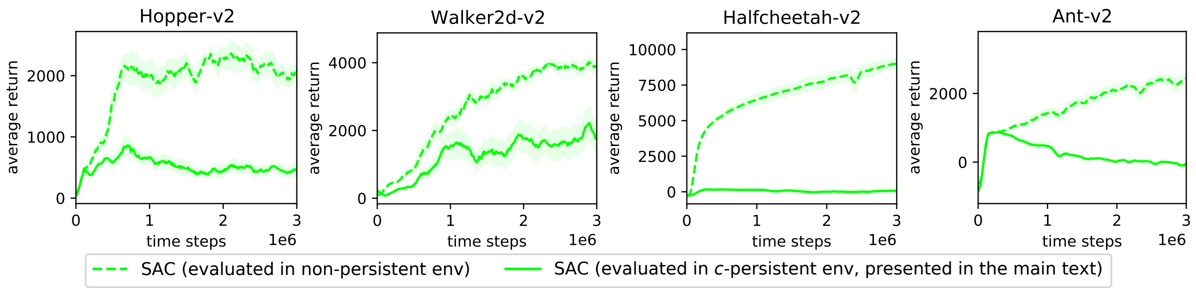


Figure 6: Results on SAC.

In Figure 5, the baseline SAC agent is trained on the standard **non-persistent** environments while being evaluated on **c-persistent** environments where the action-persistence is enforced. As shown in Figure 6[1], the performance of SAC consistently improves in the non-persistent environment that the agent is trained on, but its naïve projection into a c-persistent policy completely fails since the agent *never* considers the action-persistence during training.

F.2 Effects of varying c

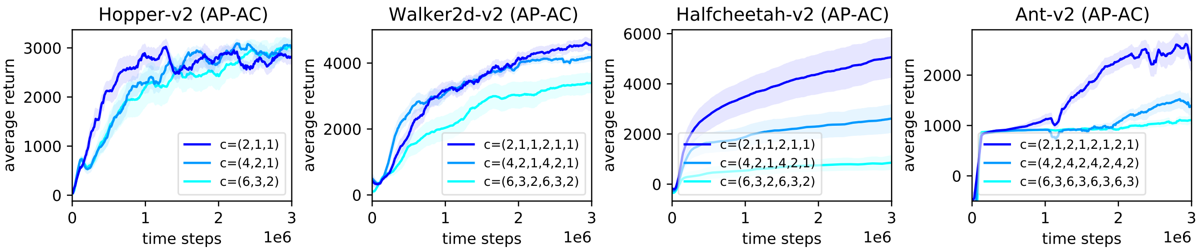


Figure 7: Effects of varying action-persistence c.

The goal of this work is to provide an efficient solution method for the *given* action-persistence c, not finding a proper c to speed up learning. Still, we conducted additional experiments to present the effects on the resulting policy of varying c. As can be seen from Figure 7, larger action persistence yields more degradation of asymptotic performance due to a limited degree of freedom of control. AP-AC consistently works well for various c's.

[1] A performance gap exists compared to those reported in the original SAC paper, due to usage of different hyperparameters such as the number of hidden units per layer, i.e. 100 (ours) / 256 (original SAC paper).

Appendix G Experimental Setup

G.1 Computing Infrastructure

All experiments were conducted on Google Cloud Platform. Specifically, we used the compute-optimized machines (c2-standard-4) that provide 4 vCPUS and 16GB memory.

G.2 Hyperparameters

Table 1: AP-AC Hyperaparameters

Parameter	Value
optimizer	Adam [6]
learning rate	$3 \cdot 10^{-4}$
discount factor γ	0.99
replay buffer size	10^6
number of hidden layers (all networks)	2
number of hidden units per layer	100
number of samples per minibatch	100
nonlinearity	ReLU
target smoothing coefficient τ	0.005
target update interval	1
gradient steps	1
(discrete only) temperature of relaxed categorical	0.1

The hyperparameters we used in the experiments are listed in Table 1. Also, for Mujoco continuous control tasks, we used automatic entropy adjustment with the entropy target $-\dim(\mathcal{A})$, and for the discrete action task (i.e. traffic light control), we used the fixed entropy coefficient $\alpha = 0.01$. We simply tried the listed hyperparameters and not tuned them further.