DoMo-AC: Doubly Multi-step Off-policy Actor-Critic Algorithm

Yunhao Tang* [1] **Tadashi Kozuno*** [2] **Mark Rowland** [1] **Anna Harutyunyan** [1]
Rémi Munos [1] **Bernardo Ávila Pires** [1] **Michal Valko** [1]

Abstract

Multi-step learning applies lookahead over multiple time steps and has proved valuable in policy evaluation settings. However, in the optimal control case, the impact of multi-step learning has been relatively limited despite a number of prior efforts. Fundamentally, this might be because multi-step policy improvements require operations that cannot be approximated by stochastic samples, hence hindering the widespread adoption of such methods in practice. To address such limitations, we introduce doubly multi-step off-policy VI (DoMo-VI), a novel oracle algorithm that combines multi-step policy improvements and policy evaluations. DoMo-VI enjoys guaranteed convergence speed-up to the optimal policy and is applicable in general off-policy learning settings. We then propose doubly multi-step off-policy actor-critic (DoMo-AC), a practical instantiation of the DoMo-VI algorithm. DoMo-AC introduces a bias-variance trade-off that ensures improved policy gradient estimates. When combined with the IMPALA architecture, DoMo-AC has showed improvements over the baseline algorithm on Atari-57 game benchmarks.

1. Introduction

Off-policy learning plays a central role in modern reinforcement learning (RL), where the algorithm learns from off-policy data such as exploratory actions, expert demonstrations and previous experiences. Off-policy learning consists of two critical components: *off-policy evaluation*, where the aim is to approximate the value function of a target policy; and *off-policy control*, where the aim is to approximate the optimal value function. Designing good evaluation and control algorithms are crucial to high-performing RL systems.

In the meantime, multi-step learning has provided a robust and consistent improvement to policy evaluation. Unlike one-step bootstrapping methods such as TD(0), multi-step learning bootstraps from predictions across multiple time steps along the trajectory, usually allowing for a much faster propagation of reward information across time. Empirically, this often helps the algorithm converge faster to the target value. In off-policy learning, notable examples include the Retrace and V-trace algorithms (Munos et al., 2016; Espeholt et al., 2018), which reduce to the celebrated TD(λ) algorithm in the on-policy case (Sutton and Barto, 1998).

In the control case, the most common approach is to find an improved policy by being greedy with respect to the current value function (Sutton and Barto, 1998). The greedy improvement effectively looks ahead for a single time step, and intuitively should also benefit from multi-step learning as TD(0). On the theory front, prior work has extended the one-step greedy improvement to the multi-step case (Efroni et al., 2018; Tomar et al., 2020). However, a fundamental challenge is that since multi-step control consists of solving a optimal control problem in the inner loop (Efroni et al., 2018), it is not straightforward to combine such an approach with sample-based learning and incremental learning. As a result, this hinders the widespread adoption of multi-step learning, as it cannot be directly applied to policy improvement and optimal control. In this work, we aim to address the key question: how to make multi-step off-policy learning practical and theoretically sound for the control case? To this end, we make a few theoretical and practical contributions.

Doubly multi-step off-policy value iteration (DoMo-VI). We introduce DoMo-VI, a multi-step learning algorithm consisting of multi-step policy evaluation and multi-step improvement (hence the name *doubly*, Section 3). DoMo-VI is compatible with using off-policy data, provably converges to the optimal value function with accelerated convergence rate, thanks to the application of multi-step learning to both the policy evaluation and improvement steps. To our knowledge, this is the first set of theoretical results on how multi-step control speeds up convergence in the off-policy setting.

[1]Google DeepMind [2]Omron Sinic X. Correspondence to: Yunhao Tang <robintyh@deepmind.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

Doubly multi-step off-policy actor-critic (DoMo-AC). We introduce the DoMo-AC algorithm as a practical instantiation of DoMo-VI (Section 4). The algorithm is designed to allow for a bias-variance trade-off in constructing policy gradient estimates from off-policy data. When implemented with the distributed learning architecture IMPALA, (Espeholt et al., 2018), DoMo-AC achieves stable performance improvements over baseline methods. This provides evidence on multi-step control in large-scale settings.

2. Background

Consider a Markov decision process (MDP) represented as the tuple $(\mathcal{X}, \mathcal{A}, P_R, P, \gamma)$ where $\mathcal{X}$ is a finite state space, $\mathcal{A}$ the finite action space, $P_R : \mathcal{X} \times \mathcal{A} \to \mathcal{P}(\mathbb{R})$ the reward kernel, $P : \mathcal{X} \times \mathcal{A} \to \mathcal{P}(\mathcal{X})$ the transition kernel and $\gamma \in [0, 1)$ the discount factor. For policy evaluation, the aim is to compute a value function $V^\pi(x) \coloneqq \mathbb{E}_\pi \left[\sum_{t=0}^\infty \gamma^t R_t \mid X_0 = x\right]$ for a target policy π; for optimal control, the aim is to find the optimal policy $\pi^* = \arg\max_{\pi \in \Pi} V^\pi$ from the set of all Markovian policies Π (Puterman, 1990).

Notation. For careful readers, we provide a more precise definition of $\arg\max_{\pi\in\Pi} V^\pi$. Since $\mathcal{X}$ is finite, V^π can be regarded as a $|\mathcal{X}|$-dimensional vector. We equip $\mathbb{R}^{|\mathcal{X}|}$ with the partial ordering induced by the non-negative orthant $[0, \infty)^{|\mathcal{X}|}$ as in Boyd et al. (2004). This ensures the maximization is well defined.

2.1. Off-policy evaluation

In off-policy evaluation, the aim is to compute approximations to a target value function V^π given off-policy data generated under a behavior policy $\mu : \mathcal{X} \to \mathcal{P}(\mathcal{A})$, which generally differs from the target policy π. As a standard assumption, we require the behavior policy μ to have full support over the action space: $\forall (x, a) \in \mathcal{X} \times \mathcal{A}$, $\mu(a|x) > 0$.

One general approach to off-policy evaluation is importance sampling (IS) (Precup, 2000; Precup et al., 2001). Define step-wise IS ratio $\rho_t \coloneqq \pi(A_t|X_t)/\mu(A_t|X_t)$ and the trace coefficient $c_t = \min(\bar{c}, \rho_t)$ with threshold $\bar{c} \geq 0$. Let $c_{0:t} \coloneqq c_0 ... c_t$ be the product of traces. The V-trace operator is defined as

$$\mathcal{R}_{\bar{c}}^{\pi,\mu} V(x) \coloneqq V(x) + \mathbb{E}_\mu \left[\sum_{t=0}^{\infty} \gamma^t c_{0:t-1} \rho_t \delta_t\right], \quad (1)$$

with TD error $\delta_t \coloneqq R_t + \gamma V(X_{t+1}) - V(X_t)$. The operator $\mathcal{R}_{\bar{c}}^{\pi,\mu}$ is η-contractive with some $\eta \in [0, \gamma]$ and has V^π as the unique fixed point. The threshold $\bar{c}$ determines the effective lookahead horizon for the operator. At one extreme $\bar{c} = 0$, V-trace looks ahead for a single time step and reduces to the Bellman operator $\mathcal{T}^\pi V(x) \coloneqq \mathbb{E}_\pi \left[R_0 + \gamma V(X_1) | \ X_0 = x\right]$, for which $\eta = \gamma$, and the contraction is slow. At another extreme $\bar{c} = \infty$, V-trace looks ahead until the end of the trajectory and reduces to the IS evaluation in expectation $\mathcal{R}_{\bar{c}}^{\pi,\mu} V(x) = V^\pi(x)$. In this case, the contraction is fast $\eta = 0$ but stochastic approximations to the V-trace target can have high variance. In practice, it is common to apply $\bar{c} = 1$ to achieve a better contraction-variance trade-off (Espeholt et al., 2018; Munos et al., 2016).

2.2. Optimal control by value iteration

Value iteration (VI) is one primary approach for finding the optimal policy π^*. VI is a recursion on the policy and value function pair $(\pi_{i+1}, V_i)_{i=0}^\infty$, which include a policy improvement step and a policy evaluation step (Puterman, 1990):

$$\pi_{i+1}(\cdot|x) = \arg\max_{\pi\in\Pi} \mathcal{T}^\pi V_i(x), \quad \text{(policy improvement)}$$
$$V_{i+1} = \mathcal{T}^{\pi_{i+1}} V_i. \quad \text{(policy evaluation)}$$

In the policy improvement step, π_{i+1} extracts the greedy policy at state x based on the one-step lookahead objective $\arg\max_a \mathbb{E}\left[R_0 + \gamma V(X_1) \mid X_0 = x, A_0 = a\right]$. In the policy evaluation step, $V_{i+1} = \mathcal{T}^{\pi_{i+1}} V_i \approx V^{\pi_{i+1}}$ approximates the value function of the improved policy π_{i+1}.

A potential drawback of VI is that it carries out only *shallow* policy improvement and policy evaluation. The policy improvement step looks ahead for a single time step $R_0 + \gamma V(X_1)$, which may result in slow improvement (Efroni et al., 2018; Tomar et al., 2020). For policy evaluation, one single application of the Bellman operator $\mathcal{T}^{\pi_{i+1}}$ might not be accurate enough due to slow contraction of the operator.

3. Doubly multi-step off-policy VI (DoMo-VI)

To alleviate the shallow policy improvement and evaluation of VI, we propose the following DoMo-VI recursions

$$\pi_{i+1}(\cdot|x) = \arg\max_{\pi\in\Pi} \mathcal{R}_{\bar{c}}^{\pi,\mu} V_i(x),$$
$$V_{i+1} = \mathcal{R}_{\bar{c}}^{\pi_{i+1},\mu} V_i. \quad (2)$$

By setting $\bar{c} = 0$ such that V-trace reduces to the one-step Bellman operator, DoMo-VI reduces to VI. When $\bar{c} > 0$, the improvement objective $\mathcal{R}_{\bar{c}}^{\pi,\mu} V_i$ effectively looks ahead multiple steps starting from x, resulting in a stronger improvement when the maximization problem can be solved exactly. Indeed, at the extreme when $\bar{c} = \infty$, the improvement objective becomes the value function $\mathcal{R}_{\bar{c}}^{\pi,\mu} V_i = \arg\max_{\pi\in\Pi} V^\pi$ and the improvement step returns the optimal policy π^*.

One subtle technical question is whether the above maximization is well defined, i.e., whether there exists a single Markov policy π which achieves the maximum. Fortunately, this is indeed the case.

Lemma 1. (**Optimal Markov policy**) For any real-valued function V over $\mathcal{X}$, a scalar $\bar{c}$, and a behavior policy μ, there exists a Markov policy π such that $\pi = \arg\max_p \mathcal{R}_{\bar{c}}^{p,\mu} V$.

Lemma 1 implies that we can obtain a single Markov policy that maximizes the improvement objective $\mathcal{R}_{\bar{c}}^{\pi,\mu}$ simultaneously across all states x. In practice, this means it is feasible to find the optimally improved policy according to the improvement objective $\mathcal{R}_{\bar{c}}^{\pi,\mu} V_i$. Such an improvement step can be carried out by a policy optimization subroutine. In general, when computing the exact optimal solution is too expensive, the optimization subroutine can be replaced by incremental updates, such as the policy gradient algorithm. We will discuss such a practical approach in Section 4.

3.1. Convergence of DoMo-VI

We now show that DoMo-VI converges to the optimal policy π^* at an accelerated convergence rate.

Theorem 2. (**Convergence rate to optimality**) Assume that expected rewards take values in $[-\bar{R}, \bar{R}]$, and V_0 is bounded by $1/(1-\gamma)$. Then, there exist a scalar $\eta^* \in [0, \gamma]$ and a sequence of scalars $(\eta_j)_{j=1}^{\infty}$ in $[0, \gamma]$ such that DoMo-VI (Eqn (2)) generates a sequence of Markov policies $(\pi_i)_{i=1}^{\infty}$ with value functions satisfying the following guarantee:

$$\|V^{\pi_{i+1}} - V^*\|_\infty \leq \max\left\{ (\eta^*)^i, \prod_{j=1}^{i} \eta_j \right\} \frac{4\bar{R}}{(1-\gamma)^2}.$$

The above result shows that DoMo-VI generates policy sequence π_i whose performance V^{π_i} converges to the optimal performance V^*. The convergence rate depends on η^* and $(\eta_j)_{j=1}^{\infty}$. It is useful to examine the explicit form of the contraction rate (Espeholt et al., 2018). Let us consider only η^* for simplicity. It holds that

$$\begin{aligned}\eta^* &= \mathbb{E}_\mu \left[\sum_{t=1}^{\infty} \gamma^t c_{0:t-2} \left(1 - c_{t-1}\right) \right] \\ &= \gamma \left(1 - \mathbb{E}_\mu[c_0]\right) + \gamma^2 \left(\mathbb{E}_\mu[c_0] - \mathbb{E}_\mu[c_0 c_1]\right) + \ldots\end{aligned}$$

When $\bar{c} = 0$, the above result recovers the convergence rate of one-step VI, which is γ^i. When $\bar{c}$ is large and there is little truncation on the IS ratio $\pi^*(a|x)/\mu(a|x)$, the contraction rate is small $\eta^* \approx 0$ and the convergence to optimality takes place in one iteration. For intermediate values of $\bar{c}$, since $\eta^* \leq \gamma$ and $\eta_i \leq \gamma$, we expect a speed up to the convergence rate of VI.

The accelerated convergence rate comes at a cost, as much of the computational complexity is hidden under the policy improvement step $\arg\max_{\pi \in \Pi} \mathcal{R}_{\bar{c}}^{\pi,\mu} V$. Since $\bar{c}$ determines the lookahead horizon of the V-trace operator, it also determines how difficult to solve the inner loop optimization problem exactly. When $\bar{c} = \infty$ and $\eta^* = 0$, the policy improvement step effectively reduces to solving the control problem itself $\arg\max_{\pi \in \Pi} V^\pi$. In practice, $\bar{c}$ mediates a trade-off between the inner loop complexity of multi-step policy improvement and outer loop convergence rate. As we will show empirically, approximately optimizing the policy improvement objective suffices to speed up convergence (Section 6)

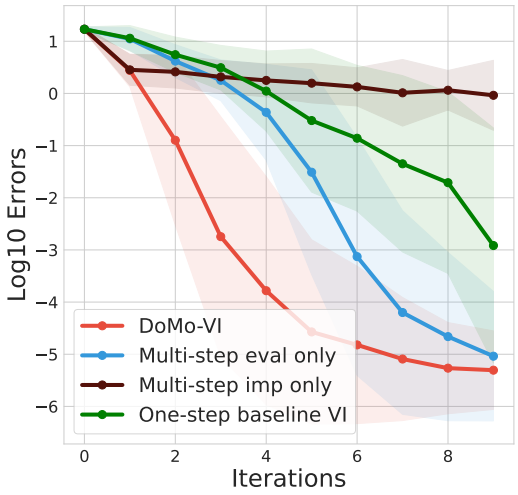

Figure 1. Comparing DoMo-VI with multi-step policy evaluation only (similar to (Espeholt et al., 2018; Munos et al., 2016)), multi-step policy optimization (similar to (Efroni et al., 2018)) and one-step baseline VI. The y-axis shows the value error $\|V^{\pi_i} - V^*\|$ on tabular MDPs. DoMo-VI combines the strengths of both multi-step policy evaluation and optimization, and achieves the fastest convergence rate among all baselines. Results are averaged across 100 runs on tabular MDPs. See Appendix B for details.

3.2. Understanding DoMo-VI

Next we discuss algorithms that interpolate VI and DoMo-VI. This helps decompose the performance improvement of DoMo-VI, and sheds light on the design choice of the algorithm. In Table 1, we make a list of algorithms that interpolate VI and DoMo-VI, as well as a number of highly related algorithms in prior literature.

Multi-step policy evaluation. Starting with VI, let us first seek to remedy shallow policy evaluation in VI. We can replace the one-step operator $\mathcal{T}^\pi$ by the V-trace operator $\mathcal{R}_{\bar{c}}^{\pi,\mu}$ for policy evaluation, resulting in the following recursion of *multi-step policy evaluation*,

$$\pi_{i+1}(\cdot|x) = \arg\max_{\pi \in \Pi} \mathcal{T}^\pi V_i(x), \quad V_{i+1} = \mathcal{R}_{\bar{c}}^{\pi_{i+1},\mu} V_i.$$

(multi-step policy evaluation)

Such a recursion bears close connections to algorithms such as Q(λ), Retrace and Peng's Q(λ) in the control case (Haru-

Table 1. A list of algorithms that can be decomposed into a policy improvement (PI) step and a policy evaluation (PE) step. The convergence rate measures how fast V^{π_i} converges to the optimal value function V^*. Concretely, if an algorithm's performance is bounded as $\|V^{\pi_i} - V^*\|_\infty \leq \eta^i C$ for some constant C. Here, $\eta \in [0, 1]$ is the convergence rate. The list of algorithms include (1) multi-step PE, which closely relates to Q(λ), Retrace and Peng's Q(λ) in the control case (Harutyunyan et al., 2016; Munos et al., 2016; Peng and Williams, 1994; Kozuno et al., 2021); (2) multi-step PI, which relates to λ-VI in the on-policy case (Efroni et al., 2018); (3) one-step baseline VI, and (4) λ-policy iteration (Efroni et al., 2018), which requires a PE oracle.

Algorithm	Policy improvement	Policy evaluation	Convergence rate
DoMo-VI	$\pi_{i+1}(\cdot\|x) = \arg\max_{\pi\in\Pi} \mathcal{R}_{\bar{c}}^{\pi,\mu} V_i(x)$	$V_{i+1} = \mathcal{R}_{\bar{c}}^{\pi_{i+1},\mu} V_i$	$\eta^* \in [0, \gamma]$
Multi-step PE only	$\pi_{i+1}(\cdot\|x) = \arg\max_{\pi\in\Pi} \mathcal{T}^{\pi} V_i(x)$	$V_{i+1} = \mathcal{R}_{\bar{c}}^{\pi_{i+1},\mu} V_i$	NA
Multi-step PI only	$\pi_{i+1}(\cdot\|x) = \arg\max_{\pi\in\Pi} \mathcal{R}_{\bar{c}}^{\pi,\mu} V_i(x)$	$V_{i+1} = \mathcal{T}^{\pi_{i+1}} V_i$	NA
Value Iteration	$\pi_{i+1}(\cdot\|x) = \arg\max_{\pi\in\Pi} \mathcal{T}^{\pi} V_i(x)$	$V_{i+1} = \mathcal{T}^{\pi_{i+1}} V_i$	γ
λ-policy iteration	$\pi_{i+1}(\cdot\|x) = \arg\max_{\pi\in\Pi} \mathcal{T}_{\lambda}^{\pi} V_i(x)$	$V_{i+1} = V^{\pi_{i+1}}$	$\frac{\gamma(1-\lambda)}{1-\gamma\lambda}$

tyunyan et al., 2016; Munos et al., 2016; Kozuno et al., 2021). The aim of such algorithm is to improve the convergence speed of the policy evaluation step. In the extreme when $\bar{c} = \infty$, the evaluation is exact $V_{i+1} = V^{\pi_{i+1}}$ and the above recursion is equivalent to policy iteration (PI), which empirically at a much faster rate than VI to the optimal policy (Puterman, 1990; Scherrer et al., 2012).

Multi-step policy improvement. Next, we can replace the one-step operator $\mathcal{T}^{\pi}$ by the V-trace operator $\mathcal{R}_{\bar{c}}^{\pi,\mu}$ for policy improvement. This leads to the following recursion of *multi-step policy improvement*,

$$\pi_{i+1}(\cdot|x) = \arg\max_{\pi\in\Pi} \mathcal{R}_{\bar{c}}^{\pi,\mu} V_i(x), \quad V_{i+1} = \mathcal{T}^{\pi_{i+1}} V_i. \tag{multi-step policy improvement}$$

In the on-policy case $\pi = \mu$ and $c_t = \lambda \in [0, 1]$, the V-trace operator is equivalent to the on-policy TD(λ) operator $\mathcal{R}_{\bar{c}}^{\pi,\mu} = \mathcal{T}_{\lambda}^{\pi}$. As a result, the above recursion recovers the multi-step greedy algorithm λ-VI proposed in (Efroni et al., 2018; Tomar et al., 2020).

Finally, DoMo-VI can be understood as combining the strengths of both multi-step policy evaluation and multi-step policy improvement. In a tabular setting, we make a comparison between DoMo-VI and multiple algorithmic variants discussed above (see Figure 1). Multi-step evaluation takes up most performance improvements from baseline VI, speeding up the convergence of V^{π_i} to V^*. Perhaps surprisingly, multi-step policy optimization provides an initial speed up, but ultimately falls short even compared to the baseline. DoMo-VI seems to combine the strength of both variants, leading to consistent speed-up throughout.

4. Doubly multi-step off-policy actor-critic (DoMo-AC)

Now, we present the core practical algorithm DoMo-AC. Starting with DoMo-VI in Eqn (2), note that in general it is computationally expensive to exactly solve the maximization problem that defines the policy improvement step $\arg\max_{\pi\in\Pi} \mathcal{R}_{\bar{c}}^{\pi,\mu} V_i(x)$. Instead, it is more tractable to take a single gradient step from the current policy iterate. When the policy is parameterized π_θ, the update in the parameter space at state x is

$$\theta_{i+1} = \theta_i + \beta \nabla_{\theta_i} \mathcal{R}_{\bar{c}}^{\pi_i,\mu} V_i(x), \tag{3}$$

where $\beta > 0$ is the learning rate. Note that going from θ_i to θ_{i+1}, the policy locally increases the policy improvement objective $\mathcal{R}_{\bar{c}}^{\pi_i,\mu} V_i(x)$. For general parameterization where $\theta \in \mathbb{R}^d$ is a vector in some d-dimensional Euclidean space, policies at different states share parameters. The policy update requires averaging gradient updates under a weighting distribution over state $x \sim b$. The combined recursion is hence

$$\theta_{i+1} = \theta_i + \beta \mathbb{E}_{x\sim b} \left[\nabla_{\theta_i} \mathcal{R}_{\bar{c}}^{\pi_{\theta_i},\mu} V_i(x) \right], \quad V_{i+1} = \mathcal{R}_{\bar{c}}^{\pi_{\theta_i},\mu} V_i.$$

We can interpret the above recursion as an an actor-critic algorithm, where the value function V_i serves as the critic. Intriguingly, when $\bar{c} = 0$, the policy update reduces to

$$\theta_{i+1} = \theta_i + \beta \mathbb{E} \left[(R_0 + \gamma V(x')) \nabla_{\theta_i} \log \pi_{\theta_i}(a|x) \right],$$

where the expecation is under $x \sim b, a \sim \pi_{\theta_i}(\cdot|x), x' \sim P(\cdot|x, a)$. This bears close resemblance to practical policy gradient updates adopted in high-performing policy-based deep RL agents (Wang et al., 2016; Mnih et al., 2016; Schulman et al., 2017; Espeholt et al., 2018).

To derive properties for the gradient update, we assume a smoothly differentiable parameterization of the policy.

Assumption 3. (**Smooth policy**) The policy $\pi_\theta(a|x)$ is differentiable with respect to θ and $\left\| \frac{\partial \pi_\theta(a|x)}{\partial \theta} \right\|_\infty \leq G$ for some constant $G \geq 0$ and for all $(x, a) \in \mathcal{X} \times \mathcal{A}$.

4.1. Approximation to policy gradient update

At the extreme when $\bar{c} = \infty$, $\mathcal{R}_{\bar{c}}^{\pi_{\theta_i},\mu} V_i(x) \approx V^{\pi_{\theta_i}}(x)$ and the policy update reduces to an exact policy gradient update

averaged over state distribution $x \sim b$,

$$\theta_{i+1} = \theta_i + \beta \mathbb{E}_{x\sim b}\left[\nabla_{\theta_i} V^{\pi_{\theta_i}}(x)\right].$$

Such an update is potentially desirable because it locally improves the average value function objective $\mathbb{E}_{x\sim b}\left[V^{\pi_\theta}(x)\right]$. In general when $\bar{c}$ is finite, the update may not locally improve the value function objective since $\nabla_{\theta_i}\mathcal{R}_{\bar{c}}^{\pi_{\theta_i},\mu}V_i(x)$ differs from the policy gradient direction $\nabla_{\theta_i}V^{\pi_{\theta_i}}(x)$. To clarify the effect of $\bar{c}$ on how well $\nabla_{\theta_i}\mathcal{R}_{\bar{c}}^{\pi_{\theta_i},\mu}V_i(x)$ carries out local improvement, we characterize its difference from the exact policy gradient.

Theorem 4. (**Approximating policy gradient**) Recall η to be the contraction rate of the V-trace operator $\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}$. Let θ_j be any scalar component of parameter $\theta \in \mathbb{R}^d$ and recall $V \in \mathbb{R}^{\mathcal{X}}$ to be a value function vector. Then $\nabla_{\theta_j}V^{\pi_\theta} \in \mathbb{R}^{\mathcal{X}}$ is a policy gradient vector over state for parameter θ_j. Assume $V = V^{\pi_\theta}$, then

$$\left\|\nabla_{\theta_j}\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}V - \nabla_{\theta_j}V^{\pi_\theta}\right\|_\infty \leq \eta\left\|\nabla_{\theta_j}V^{\pi_\theta}\right\|_\infty.$$

We offer some interpretations of the above result. Note that even if the value function is perfectly evaluated $V = V^{\pi_\theta}$, there is an irreducible error as characterized by the error bound. To see why, recall the exact policy gradient as

$$\nabla_\theta V^{\pi_\theta}(x) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty}\gamma^t\sum_a Q^{\pi_\theta}(X_t,a)\nabla_\theta\pi_\theta(a|X_t)\right].$$

Let $\bar{c} = 0$ and $V = V^{\pi_\theta}$, the approximate gradient is

$$\nabla_\theta\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}V(x) = \sum_a Q^{\pi_\theta}(x,a)\nabla_\theta\pi_\theta(a|x),$$

which corresponds to the term at $t = 0$ of the exact policy gradient. hence, we can indeed interpret the truncation threshold $\bar{c}$ as determining the lookahead horizon when calculating the policy gradient estimates, which become more accurate when $\bar{c}$ increases. This effect is reflected by the contraction rate η in the error bound.

Though a large value of $\bar{c}$ decreases the bias of the gradient estimate against the true policy gradient, it can also lead to high variance in the stochastic gradient estimates. We will examine such a bias-variance trade-off numerically in Section 6.

4.2. Low-variance unbiased gradient estimate

In general, it is challenging to compute the gradient update exactly. Instead, it is more computationally desirable to construct unbiased gradient estimate with stochastic samples. To this end, we recall that since the V-trace back-up target $\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}V$ can be approximated by off-policy stochastic estimates in an unbiased way, this naturally leads to an unbiased estimate to $\nabla_\theta\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}V$.

Algorithm 1 Doubly multi-step off-policy actor-critic (DoMo-AC)

Policy parameter θ_0, critic parameter ϕ_0 and target parameter ϕ_0^-.
for $i = 0, 1, 2...$ **do**
  Collect data. Collect trajectories $(X_t, A_t, R_t)_{t=0}^{T-1}$ of length T under behavior policy μ.
  Actor update. Update policy π_{θ_i} based on Eqn (6).
  Critic update. Update critic V_{ϕ_i} based on Eqn (7). and update target network.
end for
Output the final policy.

Theorem 5. (**Unbiased gradient estimate**) Assume trajectories $(X_t, A_t, R_t)_{t=0}^{\infty} \sim \mu$ reach a terminal state within $H < \infty$ steps almost surely. Let $X_0 = x$ be the initial state, the unbiased V-trace back-up target estimate is

$$\widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}V}(x) \coloneqq V(x) + \sum_{t=0}^{\infty}\gamma^t c_{0:t-1}\rho_t\delta_t. \tag{4}$$

Further, $\widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}V}(x)$ is differentiable and $\nabla_\theta\widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}V}(x)$ is an unbiased estimate to $\nabla_\theta\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}V(x)$.

Intriguingly, the naive estimate based on Eqn (4) turns out to have low variance. To see this, consider the special case when $\bar{c} = \infty$ and the trace coefficient is effectively the step-wise IS ratio $c_t = \rho_t$. In this case, the gradient estimate evaluates to

$$\nabla_\theta\widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}V}(x) = \sum_{t=0}^{\infty}\gamma^t\rho_{0:t}\widehat{A}_t\nabla_\theta\log\pi_\theta(A_t|X_t), \tag{5}$$

where $\widehat{A}_t = R_t + \gamma\widehat{V}(X_{t+1}) - V(X_t)$ is the advantage estimate. Here, the built-in variance reduction technique is the subtraction of value function $V(X_t)$ as a baseline when computing advantage estimate $\widehat{A}_t$, which is most commonly used in policy gradient estimate (Sutton et al., 2000; Weaver and Tao, 2013). Secondly, the value estimate $\widehat{V}(X_t)$ turns out to be the doubly-robust value function estimate (Jiang and Li, 2016; Thomas and Brunskill, 2016), which writes recursively as

$$\widehat{V}(X_t) = V(X_t) + \rho_t\left(R_t + \gamma\widehat{V}(X_{t+1}) - V(X_t)\right).$$

The doubly-robust estimation technique has also been known to reduce variance in off-policy learning (Jiang and Li, 2016; Thomas and Brunskill, 2016). For general values of the trace coefficient c_t, we should expect a similar variance reduction effect.

4.3. Implementation with function approximation

Finally, we spell out the algorithm with both a parameterized policy π_θ and a parameterized critic V_ϕ. Given a trajectory

$(X_t, A_t, R_t)_{t=0}^{T-1}$ of length T, sampled under the behavior policy μ, the policy is updated via the DoMo-AC gradient estimate

$$\theta_{i+1} = \theta_i + \beta \frac{1}{T} \sum_{t=0}^{T-1} \nabla_{\theta_i} \widehat{\mathcal{R}_{\bar{c}}^{\pi_{\theta_i},\mu}} V_{\phi_i}(X_t). \quad (6)$$

Meanwhile, the critic is updated using gradient descent on the least square loss function

$$\phi_{i+1} = \phi_i - \beta \frac{1}{T} \sum_{t=0}^{T-1} \nabla_{\phi_i} \left(V_{\text{target}}(X_t) - V_{\phi_i}(X_t) \right)^2, \quad (7)$$

where $V_{\text{target}}(X_t) = \widehat{\mathcal{R}_{\bar{c}}^{\pi_{\theta_{i+1}},\mu}} V_{\phi_i^-}(X_t)$ is the back-up target computed via the target network ϕ_i^-. The target network is slowly updated towards the main network $\phi_i^- = (1-\tau)\phi_i^- + \tau\phi_i$ (Lillicrap et al., 2015). In practical implementations, it is more common to carry out the above gradient updates simultaneously. See Algorithm 1 for full algorithm.

5. Discussion

We provide discussions on a few lines of related work and natural extensions of our current method.

λ-policy iteration (λ-PI). Another important variant of multi-step policy improvement algorithm is λ-PI (Efroni et al., 2018), which in our notations can be expressed as

$$\pi_{i+1}(\cdot|x) = \arg\max_{\pi \in \Pi} \mathcal{T}_\lambda^\pi V_i(x), \quad V_{i+1} = V^{\pi_{i+1}},$$

where $\mathcal{T}_\lambda^\pi$ is the on-policy TD(λ) operator. This algorithm achieves a convergence rate of $\frac{\gamma(1-\lambda)}{1-\lambda\gamma}$ to the optimal value function, which significantly speeds up one-step VI when λ is close to 1. One primary bottleneck of λ-PI is that it requires a policy evaluation oracle, setting the value function estimate V_{i+1} to be the exact value function $V^{\pi_{i+1}}$. Such a critic is in general not accessible in practice. DoMo-VI removes such a limitation and replaces the oracle by a multi-step evaluation operator $V_{i+1} = \mathcal{R}_{\bar{c}}^{\pi_{i+1},\mu} V_i$, which can be practically implemented. Another major difference between DoMo-VI and λ-PI is that the latter requires on-policy data when doing policy improvement.

Off-policy corrections are important for multi-step policy improvement. DoMo-VI can be extended to evaluation operators $\mathcal{R}_{\bar{c}}^{\pi,\mu}$ beyond V-trace, such as the value function variant of Q(λ) (Harutyunyan et al., 2016), where the trace coefficient $c_t = \lambda$. This closely resembles TD(λ) with the main difference being that the data is off-policy. The tree-backup trace $c_t = \pi(A_t|X_t)$ can be understood as a special case of V-trace (Precup et al., 2001) since $c_t \leq \rho_t$. A primary bottleneck of tree-backup is that it cuts traces quickly and is not efficient when near on-policy (Munos et al., 2016). Another alternative is the value function equivalent of Peng's Q(λ) operator (Peng and Williams, 1994), which can be understood as geometrically weighted sum of n-step TD(n) operators. Unlike V-trace and Q(λ), which carry out off-policy corrections, Peng's Q(λ) does not have the target value function as the fixed point. Nevertheless, Peng's Q(λ) has displayed practical benefits over methods based on proper off-policy corrections, thanks to its significant improvement in the contraction rate (though to the biased fixed point) (Kozuno et al., 2021).

However, we can verify that when $\mathcal{R}_{\bar{c}}^{\pi,\mu}$ is the Peng's Q(λ) operator, $\arg\max_{\pi\in\Pi} \mathcal{R}_{\bar{c}}^{\pi,\mu} V(x)$ corresponds to the one-step greedy policy. This means uncorrected algorithms such as Peng's Q(λ) cannot entail multi-step policy improvement.

Multiple applications of evaluation operator. We can consider a more general form of the DoMo-AC gradient update, by differenting through multiple applications of the evaluation operator

$$\theta_{i+1} = \theta_i + \mathbb{E}_{x\sim b} \left[\nabla_{\theta_i} \left(\mathcal{R}_{\bar{c}}^{\pi_{\theta_i},\mu} \right)^m V_i(x) \right],$$

for $m \geq 1$.Increasing m has a similar effect as increasing $\bar{c}$ as both lengthen the effective lookahead horizon. Intriguingly, when we take $\mathcal{R}_{\bar{c}}^{\pi,\mu}$ to be the Q(λ) operator with $\lambda = 1$, the policy improvement objective $\left(\mathcal{R}_{\bar{c}}^{\pi_{\theta_i},\mu}\right)^m V_i(x)$ closely resembles the Taylor expansion policy optimization objective proposed in (Tang et al., 2020). A notable difference is that Tang et al. (2020) considered the special case where $V_i = V^\mu$ as the origin of the expansion, while here V_i does not have to be the value function for any specific policy.

6. Experiments

We seek to answer the following questions: (Q1) Does multi-step improvement entail faster convergence to the optimal policy in tabular settings (Theorem 2)? (Q2) Does DoMo-AC introduce a bias-variance trade-off to estimating PG (Theorem 4)? (Q3) Does DoMo-AC improve state-of-the-art policy based agents in large-scale settings?

6.1. Tabular experiments

To answer Q1, we start by empirically validating the speed-up of the convergence guarantee (predicted by Theorem 2) entailed by DoMo-VI and DoMo-AC. We mainly compare three baselines: (1) one-step baseline VI (green), which consists of one-step policy improvement and evaluation $V_{i+1} = \mathcal{T}^{\pi_{i+1}} V_i$ where π_{i+1} is one-step greedy; (2) multi-step policy evaluation (brown), which improves over VI with multi-step evaluation $V_{i+1} = \mathcal{R}_{\bar{c}}^{\pi_{i+1}} V_i$ for $\bar{c} = 1$; (3) finally, the multi-step policy improvement algorithm where

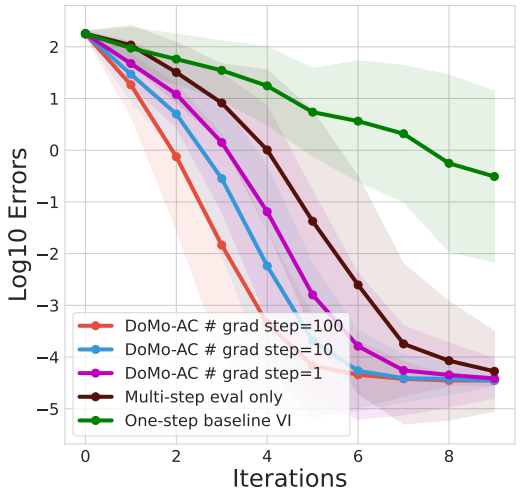

Figure 2. Evaluating the impact of approximate optimization of the policy improvement objective $\arg\max_{\pi\in\Pi} \mathcal{R}_{\bar{c}}^{\pi,\mu} V(x)$. The y-axis shows the value error $\|V^{\pi_i} - V^*\|$ on tabular MDPs. Throughout, we parameterize softmax policy and optimize the improvement objective with gradient ascent. Varying the number of gradient ascent steps, we see that as the number of steps increases, the improvement to convergence speed becomes more profound.

the policy $\pi_i = \pi_{\theta_i}$ is improved via N gradient ascents with approximate gradient $\nabla_{\theta_i} \mathcal{R}_{\bar{c}}^{\pi_i,\mu}$ across all states. Formally, for $\forall 1 \le j \le N$,

$$\theta_{i+1}^{(j+1)} = \theta_{i+1}^{(j)} + \eta \frac{1}{|\mathcal{X}|} \sum_{x=1}^{|\mathcal{X}|} \nabla_{\theta_i^{(j)}} \mathcal{R}_{\bar{c}}^{\pi_{\theta_i^{(j)}},\mu} V(x_i),$$

where we let $\theta_{i+1} = \theta_{i+1}^{(N)}$ as the final iterate of the gradient update. The value function is then updated via multi-step evaluation $V_{i+1} = \mathcal{R}_{\bar{c}}^{\pi_{i+1}} V_i$. To study the impact of the degree of optimization, we consider $N \in \{1, 10, 100\}$ (purple, blue and red). By increasing N, the policy iterate $\pi_{\theta_{i+1}}$ gets closer to the optimal policy $\arg\max_{\pi_i} \mathcal{R}_{\bar{c}}^{\pi,\mu} V_i(x)$. All results are averaged over 100 randomly generated MDPs. See Appendix B for experimental details.

Figure 2 shows the error $\|V^{\pi_i} - V^*\|_2$ as a function of iteration i. As expected, multi-step policy evaluation provides a major improvement over the VI baseline in accelerating the convergence. On top of that, as N increases, multi-step policy improvement exhibits further performance improvements. This confirms the benefits of combining multi-step evaluation and improvement in the tabular settings where exact gradient computations are available.

Stochastic gradient estimates in tabular settings. To answer Q2, note that in DoMo-AC we use the stochastic update $\nabla_\theta \widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}} V(x)$ to update the policy parameter θ. As discussed in Section 4, the choice of $\bar{c}$ mediates a trade-off between bias and variance, on the approximation of $\nabla_\theta \widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}} V(x)$ to the true policy gradient $\nabla_\theta V^{\pi_\theta}(x)$.

In Figure 5, we examine such a bias-variance trade-off numerically. On a set of randomly generated MDPs, we calculate $\nabla_\theta \widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}} V(x)$ based on a fixed number of trajectories generated under behavior policy μ. We then estimate the bias, variance and squared error of the policy gradient estimate against the ground truth $\nabla_\theta V^{\pi_\theta}(x)$. The results show that, as expected, when $\bar{c}$ increases from 0 to 10, the bias generally decreases, whereas the variance increases rapidly. This leads to an optimal middle ground (in this case $\log \bar{c} \approx 0$ and $\bar{c} \approx 1$) at which $\nabla_\theta \widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}} V(x)$ obtains the lowest squared error among this class of stochastic gradient estimates. Naturally, this trade-off will significantly impact the agent performance in large-scale settings, which we investigate next.

6.2. Deep RL experiments

To investigate the practical performance of DoMo-AC gradient update, we test different algorithmic variants with distributed actor-critic over architecture the Atari-57 games (Bellemare et al., 2013).

Our implementation is based on the IMPALA architecture (Espeholt et al., 2018), an actor-critic algorithm with distributed actors and a centralized learner. The actors collect partial trajectories with the behavior policy μ and send to the learner with target policy π_θ. Due to the latency of the actor-learner communication, the behavior policy uses a slightly stale copy of the policy parameter $\mu = \pi_{\theta_{\text{old}}}$, leading to inherent off-policyness during training $\pi_\theta \neq \mu$. By default, the learner maintains a policy network π_θ and a value network V_ϕ. Across all algorithmic variants we consider, the value networks are updated with the V-trace back-up targets (Espeholt et al., 2018) while we test different variants of actor updates. All algorithmic variants share hyper-parameters wherever possible. See Appendix for further experiment details.

We compare a few algorithmic variants defined by different choices of the off-policy evaluation operators $\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}$. For the multi-step variant, we choose V-trace with the trace coefficient threshold $\bar{c}$ as a tunable hyper-parameter. We find that $\bar{c}$ in between 0.3 and 0.5 works the best in practice and will report the ablation results; for the one-step variant, we use the one-step operator $\mathcal{T}^\pi$, which can be understood as the special case $\bar{c} = 0$. The baseline algorithm IMPALA (Espeholt et al., 2018) is closely related to the one-step variant, but with slightly different implementation details. We discuss such differences in Appendix B.

In Figure 3, we show the training performance curves of all algorithms. Each curve is an average over 5 runs, with each run computed as either the mean or median human normal-

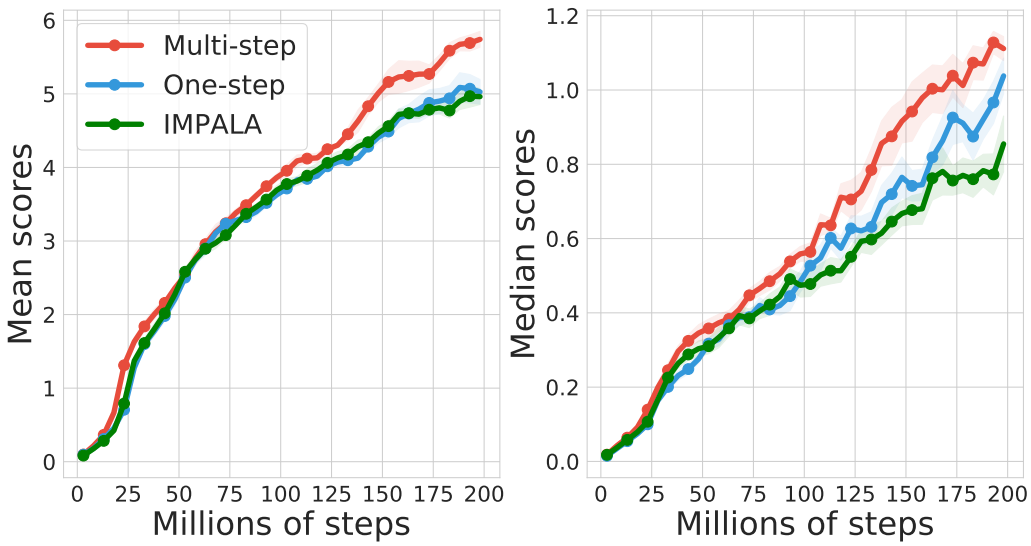

Figure 3. Comparing actor-critic algorithmic variants based on the IMPALA architecture (Espeholt et al., 2018). We compare the DoMo-AC algorithm (Algorithm 1) instantiated with V-trace operator $\mathcal{R}_{\bar{c}}^{\pi,\mu}$ with $\bar{c} = 0.5$; the one-step algorithm $\mathcal{R}_{\bar{c}}^{\pi,\mu} = \mathcal{T}^{\pi}$, which also be understood as the special case $\bar{c} = 0$; and the IMPALA baseline. We report the evaluated median and mean human normalized scores over 57 Atari games, averaged across 5 seeds. Overall, the DoMo-AC algorithm outperforms the one-step variant and the IMPALA baseline.

ized scores across 57 games. We find that the DoMo-AC implementation with V-trace $\bar{c} = 0.5$ provides statistically significant improvements over one-step trace and IMPALA, implying the potential benefits of introducing multi-step gradient estimate.

Alternative off-policy evaluation operators. Besides V-trace, other alternative trace coefficients such as tree-backup $c_t = \pi(A_t|X_t)$ (Precup et al., 2001) and Q(λ) $c_t = \lambda \in [0, 1]$ (Harutyunyan et al., 2016) all define valid off-policy evaluation operators (Munos et al., 2016). We carry out a comparison with all such alternatives in Figure 4 in Appendix B, where we show that V-trace obtains overall the best empirical performance.

Ablation on the trace coefficient threshold $\bar{c}$. We next assess how sensitive the performance is to the trace coefficient threshold $\bar{c}$. We carry out experiments with $\bar{c}$ taking values in the range $[0, 1]$ and graph the results in Figure 6 (Appendix B). Going from $\bar{c} = 0$ to $\bar{c} = 1$, we find the best performance is obtained at the range $\bar{c} = 0.3 \sim 0.5$. The fact that $\bar{c} > 0$ obtains the best performance demonstrates the practical utility of multi-step policy gradient estimate, compatible with the previous results. However, as $\bar{c}$ increases, the multi-step gradient estimate accumulates higher variance. Indeed, in the limit $\bar{c} \to \infty$, we have $c_t \to \rho_t$ and step-wise IS ratios can induce high variance to the overall estimates, which degrades the overall performance of the algorithm.

Intriguingly, here the optimal value of $\bar{c} \in [0.3, 0.5]$ is noticeably lower than the typical value of the trace threshold applied in value-based learning (e.g., Retrace and V-trace all adopt $\bar{c} = 1$ in their implementations by default (Munos et al., 2016; Espeholt et al., 2018)). We speculate this might be because policy-based algorithms are generally more susceptible to high variance than value-based algorithms, and hence enjoy better performance when the estimates are of low variance.

7. Conclusion

We have proposed DoMo-VI, an extension of the classic VI algorithm which combines multi-step policy improvement with policy evaluation. Contrast to prior work, DoMo-VI enjoys theoretical speed-up to the optimal policy and is applicable in general off-policy settings. As a practical instantiation of the oracle algorithm, we propose DoMo-AC. DoMo-AC achieves the effect of multi-step improvement by applying a policy gradient estimator with a novel bias and variance trade-off. Compared to the baseline actor-critic algorithm, DoMo-AC generally enjoys more accurate approximation to the ground truth policy gradient. Implementing DoMo-AC with the IMPALA architecture, we observe a modest improvement from the baseline over the Atari game benchmarks. Possible future directions include adaptive methods for choosing the trace coefficient $\bar{c}$, and extensions of ideas of DoMo-VI more directly to value-based agents such as DQN.

References

Mohammad Babaeizadeh, Iuri Frosio, Stephen Tyree, Jason Clemons, and Jan Kautz. Reinforcement learning through asynchronous advantage actor-critic on a gpu. *arXiv preprint arXiv:1611.06256*, 2016.

Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Yonathan Efroni, Gal Dalal, Bruno Scherrer, and Shie Mannor. Multiple-step greedy policies in approximate and online reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv preprint arXiv:1802.01561*, 2018.

Anna Harutyunyan, Marc G Bellemare, Tom Stepleton, and Rémi Munos. Q (lambda) with off-policy corrections. In *International Conference on Algorithmic Learning Theory*, pages 305–320. Springer, 2016.

Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado Van Hasselt, and David Silver. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*, 2018.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 652–661. PMLR, 2016.

Tadashi Kozuno, Yunhao Tang, Mark Rowland, Rémi Munos, Steven Kapturowski, Will Dabney, Michal Valko, and David Abel. Revisiting peng's q (lambda) for modern reinforcement learning. In *International Conference on Machine Learning*, pages 5794–5804. PMLR, 2021.

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. *arXiv preprint arXiv:1509.02971*, 2015.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

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.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1054–1062, 2016.

Arun Nair, Praveen Srinivasan, Sam Blackwell, Cagdas Alcicek, Rory Fearon, Alessandro De Maria, Vedavyas Panneershelvam, Mustafa Suleyman, Charles Beattie, Stig Petersen, et al. Massively parallel methods for deep reinforcement learning. *arXiv preprint arXiv:1507.04296*, 2015.

Jing Peng and Ronald J Williams. Incremental multi-step q-learning. In *Machine Learning Proceedings 1994*, pages 226–232. Elsevier, 1994.

Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

Doina Precup, Richard S Sutton, and Sanjoy Dasgupta. Off-policy temporal-difference learning with function approximation. In *ICML*, pages 417–424, 2001.

Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

Bruno Scherrer, Victor Gabillon, Mohammad Ghavamzadeh, and Matthieu Geist. Approximate modified policy iteration. *arXiv preprint arXiv:1205.3054*, 2012.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

Yunhao Tang, Michal Valko, and Rémi Munos. Taylor expansion policy optimization. *arXiv preprint arXiv:2003.06259*, 2020.

Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148. PMLR, 2016.

Tijmen Tieleman, Geoffrey Hinton, et al. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

Manan Tomar, Yonathan Efroni, and Mohammad Ghavamzadeh. Multi-step greedy reinforcement learning algorithms. In *International Conference on Machine Learning*, pages 9504–9513. PMLR, 2020.

Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay. *arXiv preprint arXiv:1611.01224*, 2016.

Lex Weaver and Nigel Tao. The optimal reward baseline for gradient-based reinforcement learning. *arXiv preprint arXiv:1301.2315*, 2013.

APPENDICES: DoMo-AC: Doubly Multi-step Off-policy Actor-Critic Algorithm

A. Proof of theoretical results

In this appendix, we provide missing proofs in the main paper. We begin with introducing some notations used in the proofs.

We denote an identity operator by I, which maps any real-valued function to itself. Its domain will be clear from contexts. For any Markov policy π, let Π^π denote an operator that maps any bounded real-valued function Q over $\mathcal{X} \times \mathcal{A}$ to a real-valued function $\Pi^\pi Q$ over $\mathcal{X}$ defined by

$$\left(\Pi^\pi Q\right)(x) = \sum_{a \in \mathcal{A}} \pi(a|x) Q(x,a) \text{ at every } x \in \mathcal{X}\,.$$

For a scalar $\bar{c} \in (0, \infty)$, and a behavior policy μ, a similar operator $\Pi_{\bar{c}}^{\pi,\mu}$ maps Q to a real-valued function $\Pi_{\bar{c}}^{\pi,\mu} Q$ over $\mathcal{X}$ defined by [1]

$$\left(\Pi_{\bar{c}}^{\pi,\mu} Q\right)(x) = \sum_{a \in \mathcal{A}} \mu(a|x) \min\left\{\bar{c}, \frac{\pi(a|x)}{\mu(a|x)}\right\} Q(x,a) \text{ at every } x \in \mathcal{X}\,.$$

Abusing notations, let P denote an operator that maps any bounded real-valued function V over $\mathcal{X}$ to a real-valued function PV over $\mathcal{X} \times \mathcal{A}$ defined by

$$\left(PV\right)(x,a) = \sum_{y \in \mathcal{X}} P(y|x,a) V(y) \text{ at every } (x,a) \in \mathcal{X} \times \mathcal{A}$$

Its conjugate with the Π^π and $\Pi_{\bar{c}}^{\pi,\mu}$ operators are denoted by $P^\pi := \Pi^\pi P$ and $P^{c\mu\wedge\pi} := \Pi_{\bar{c}}^{\pi,\mu} P$, respectively. With these operators, the V-trace operator can be rewritten as follows:

$$\mathcal{R}_{\bar{c}}^{\pi,\mu} V = V + \left(I - \gamma P^{\bar{c}\mu\wedge\pi}\right)^{-1}\left(\Pi^\pi r + \gamma P^\pi V - V\right) = \left(I - \gamma P^{\bar{c}\mu\wedge\pi}\right)^{-1}\left(\Pi^\pi r + \gamma (P^\pi - P^{\bar{c}\mu\wedge\pi}) V\right),$$

where $\left(I - \gamma P^{\bar{c}\mu\wedge\pi}\right)^{-1} := \sum_{t=0}^\infty \gamma^t \left(P^{\bar{c}\mu\wedge\pi}\right)^t$. As the notation implies, it holds that $\left(I - \gamma P^{\bar{c}\mu\wedge\pi}\right)\left(I - \gamma P^{\bar{c}\mu\wedge\pi}\right)^{-1} = \left(I - \gamma P^{\bar{c}\mu\wedge\pi}\right)^{-1}\left(I - \gamma P^{\bar{c}\mu\wedge\pi}\right) = I$.

An operator, say $\mathcal{O}$, is said to be monotonic if $\mathcal{O}f \geq \mathcal{O}g$ for any pair of functions f and g such that $f \geq g$. All operators introduce above are monotonic.

A.1. Proof of Lemma 1 (Optimal Markov Policy)

Lemma 1 states that there exists a Markov policy π such that

$$\max_{p \in \Pi}\left(\mathcal{R}_{\bar{c}}^{p,\mu} V(x)\right) = \mathcal{R}_{\bar{c}}^{\pi,\mu} V(x) \text{ for all } x \in \mathcal{X},$$

where Π is the set of all Markov policies. As p on the left hand side may depend on $x \in \mathcal{X}$, the existence of π is non-trivial.

For a fixed V and μ, let π_x be a policy such that $\pi_x := \arg\max_{p \in \Pi}\left(\mathcal{R}_{\bar{c}}^{p,\mu} V(x)\right)$. Note that it is dependent on x, and there may be multiple policies that maximize the right hand side. If it is not unique, pick up one arbitrarily. Furthermore, let π be a Markov policy such that $\pi(\cdot|x) := \pi_x(\cdot|x)$ for all $x \in \mathcal{X}$. By definition, for any Markov policy π and any state $x \in \mathcal{X}$,

$$\begin{aligned}
&\mathcal{R}_{\bar{c}}^{\pi,\mu} V(x) \\
&\leq \left(I - \gamma P^{c\mu\wedge\pi_x}\right)^{-1}\left(\Pi^{\pi_x} r + \gamma\left(P^{\pi_x} - P^{\bar{c}\mu\wedge\pi_x}\right) V\right)(x) \\
&\leq \left(\Pi^{\pi_x} r + \gamma\left(P^{\pi_x} - P^{\bar{c}\mu\wedge\pi_x}\right) V\right)(x) + \gamma P^{c\mu\wedge\pi_x}\left(I - \gamma P^{c\mu\wedge\pi_x}\right)^{-1}\left(\Pi^{\pi_x} r + \gamma\left(P^{\pi_x} - P^{\bar{c}\mu\wedge\pi_x}\right) V\right)(x) \\
&= \left(\Pi^{\pi} r + \gamma\left(P^{\pi} - P^{\bar{c}\mu\wedge\pi}\right) V\right)(x) + \gamma P^{c\mu\wedge\pi}\left(I - \gamma P^{c\mu\wedge\pi_x}\right)^{-1}\left(\Pi^{\pi_x} r + \gamma\left(P^{\pi_x} - P^{\bar{c}\mu\wedge\pi_x}\right) V\right)(x),
\end{aligned}$$

where the last line follows since $\pi(\cdot|x) = \pi_x(\cdot|x)$ by definition, and thus, $\Pi_c^{\pi_x,\mu} Q(x) = \Pi_c^{\pi,\mu} Q(x)$ for any bounded real-valued function Q over $\mathcal{X} \times \mathcal{A}$. Now, note that the second term is $\gamma P^{c\mu\wedge\pi} \mathcal{R}_{\bar{c}}^{\pi_x,\mu} V(x)$, and

$$P^{c\mu\wedge\pi} \mathcal{R}_{\bar{c}}^{\pi_x,\mu}(x) = \mathbb{E}_{y \sim P(\cdot|x,a), a \sim \mu(\cdot|x)}\left[\min\left\{\bar{c}, \frac{\pi(a|x)}{\mu(a|x)}\right\} \mathcal{R}_{\bar{c}}^{\pi_x,\mu} V(y)\right].$$

[1] Recall we assume that a behavior policy has the full support: $\mu(a|x) > 0$ for all $(x,a) \in \mathcal{X} \times \mathcal{A}$.

Therefore, applying the same argument to $\mathcal{R}_{\bar{c}}^{\pi_x,\mu}V(y)$, we can conclude that

$$\max_{\pi\in\Pi}\left(\mathcal{R}_{\bar{c}}^{\pi,\mu}V(x)\right)\leq\mathcal{R}_{\bar{c}}^{\pi,\mu}V(x).$$

A.2. Proof of Theorem 2 (Convergence Rate to Optimality)

We upper-bound $\heartsuit$ and $\spadesuit$ in the following equation:

$$V^*-V^{\pi_i}=\underbrace{V^*-\mathcal{R}_{\bar{c}}^{\pi_i,\mu}V_{i-1}}_{:=\heartsuit}+\underbrace{\mathcal{R}_{\bar{c}}^{\pi_i,\mu}V_{i-1}-V^{\pi_i}}_{:=\spadesuit}.$$

For brevity, we let $\Pi^*:=\Pi^{\pi^*}$, $P^*:=\Pi^{\pi^*}P$, $\Pi^{*,\mu}:=\Pi_{\bar{c}}^{\pi^*,\mu}$, $P^{*,\mu}:=\Pi_{\bar{c}}^{\pi^*,\mu}P$, $\mathcal{R}^{*,\mu}:=\mathcal{R}_{\bar{c}}^{\pi^*,\mu}$, $\Pi_j:=\Pi_{\bar{c}}^{\pi_j,\mu}$, $P_j:=\Pi_{\bar{c}}^{\pi_j,\mu}P$, and $\mathcal{R}_j:=\mathcal{R}_{\bar{c}}^{\pi_j,\mu}$.

Upper-bound for $\heartsuit$. By definition, $\mathcal{R}_iV_{i-1}\geq\mathcal{R}^{*,\mu}V_{i-1}$, and $V^*=\mathcal{R}^{*,\mu}V^*$. Therefore,

$$\heartsuit\leq\gamma\left(I-\gamma P^{*,\mu}\right)^{-1}\left(P^*-P^{*,\mu}\right)\left(V^*-V_{i-1}\right)=\gamma\left(I-\gamma P^{*,\mu}\right)^{-1}\left(P^*-P^{*,\mu}\right)\left(V^*-\mathcal{R}_{i-1}V_{i-2}\right).$$

By induction on i, $\heartsuit\leq\left(\Gamma^*\right)^i\left(V^*-V_0\right)$, where $\Gamma^*:=\gamma\left(I-\gamma P^{*,\mu}\right)^{-1}\left(P^*-P^{*,\mu}\right)$. As shown by Munos et al. (2016, around Eqn (12) in Appendix C), Γ^* is monotonic, and $\Gamma^*e\leq\eta^*e\leq\gamma e$, where e is a constant function over $\mathcal{X}$ outputting 1 everywhere. Thus, $\heartsuit\leq\left(\eta^*\right)^i\left\|V^*-V_0\right\|_\infty e$. As both V^* and V_0 are bounded by $1/(1-\gamma)$, $\left\|V^*-V_0\right\|_\infty\leq 2/(1-\gamma)$.

Upper-bound for $\spadesuit$. It holds that $V^{\pi_i}=\mathcal{R}_iV^{\pi_i}$. Therefore,

$$\begin{aligned}\spadesuit&=\gamma\left(I-\gamma P_i\right)^{-1}\left(P^{\pi_i}-P_i\right)\left(V_{i-1}-V^{\pi_i}\right)\\&=\gamma\left(I-\gamma P_i\right)^{-1}\left(P^{\pi_i}-P_i\right)\left(V_{i-1}-\mathcal{R}_iV_{i-1}+\spadesuit\right)\\&=\gamma\left(I-\gamma P^{\pi_i}\right)^{-1}\left(P^{\pi_i}-P_i\right)\left(V_{i-1}-\mathcal{R}_iV_{i-1}\right),\end{aligned}$$

where the last line follows since

$$\spadesuit-\gamma\left(I-\gamma P_i\right)^{-1}\left(P^{\pi_i}-P_i\right)\spadesuit=\left(I-\gamma P_i\right)^{-1}\left(I-\gamma P_i-\gamma P^{\pi_i}+\gamma P_i\right)\spadesuit=\left(I-\gamma P_i\right)^{-1}\left(I-\gamma P^{\pi_i}\right)\spadesuit.$$

By definition,

$$\begin{aligned}V_{i-1}-\mathcal{R}_iV_{i-1}&=\mathcal{R}_{i-1}V_{i-2}-\mathcal{R}_iV_{i-1}\\&\leq\mathcal{R}_{i-1}V_{i-2}-\mathcal{R}_{i-1}V_{i-1}\\&=\gamma\left(I-\gamma P_{i-1}\right)^{-1}\left(P^{\pi_{i-1}}-P_{i-1}\right)\left(V_{i-2}-V_{i-1}\right)\\&=\gamma\left(I-\gamma P_{i-1}\right)^{-1}\left(P^{\pi_{i-1}}-P_{i-1}\right)\left(V_{i-2}-\mathcal{R}_{i-1}V_{i-2}\right).\end{aligned}$$

By induction, we deduce that $V_{i-1}-\mathcal{R}_iV_{i-1}\leq\Gamma_{i-1}\cdots\Gamma_1\left(V_0-\mathcal{R}_1V_0\right)$, where $\Gamma_j:=\gamma\left(I-\gamma P_j\right)^{-1}\left(P^{\pi_j}-P_j\right)$. As

$$\begin{aligned}V_0-\mathcal{R}_1V_0&=V_0-V^{\pi_1}+\mathcal{R}_1V^{\pi_1}-\mathcal{R}_1V_0\\&=V_0-V^{\pi_1}+\gamma\left(I-\gamma P_1\right)^{-1}\left(P^{\pi_1}-P_1\right)\left(V^{\pi_1}-V_0\right),\end{aligned}$$

we conclude that

$$\spadesuit\leq\gamma\left(I-\gamma P^{\pi_i}\right)^{-1}\left(P^{\pi_i}-P_i\right)\Gamma_{i-1}\cdots\Gamma_1\left(V_0-V^{\pi_1}+\gamma\left(I-\gamma P_1\right)^{-1}\left(P^{\pi_1}-P_0\right)\left(V^{\pi_1}-V_0\right)\right).$$

As shown by Munos et al. (2016), $P^{\pi_i}-P_i$ is monotonic, and $\left(P^{\pi_i}-P_i\right)e\leq e$, where e is a constant function over $\mathcal{X}$ outputting 1 everywhere. Furthermore, Γ_j is monotonic, and there exists some scalar κ_j such that $\Gamma_je\leq\kappa_je\leq\gamma e$. Thus,

$$\spadesuit\leq\frac{\gamma\kappa_{i-1}\cdots\kappa_1}{1-\gamma}(1+\kappa_1)\left\|V^{\pi_1}-V_0\right\|_\infty e\leq\frac{\gamma\kappa_{i-1}\cdots\kappa_1}{1-\gamma}(1+\gamma)\left\|V^{\pi_1}-V_0\right\|_\infty e.$$

Because both V^{π_1} and V_0 are bounded by $1/(1-\gamma)$, $\left\|V^{\pi_1}-V_0\right\|_\infty\leq 2/(1-\gamma)$.

Combining Together. From those bounds and noting that $\gamma(1+\gamma)/(1-\gamma)+1 \leq 2/(1-\gamma)$, we conclude the proof.

A.3. Proof of Theorem 4

For notational simplicity, let $V_1 \coloneqq \mathcal{R}_{\bar{c}}^{\pi_\theta,\mu} V$. In the below, we consider the gradient with respect to the j-th component of θ. Then $\nabla_{\theta_j} V_1(x)$ is a vector of size $\mathbb{R}^{|\mathcal{X}|}$. Now, let $R^{\pi_\theta} \in \mathbb{R}^{|\mathcal{X}|}$ be the vector of reward such that $R^{\pi_\theta}(x) \coloneqq \sum_a r(x,a)\pi_\theta(a|x)$. Plugging in the definition of the operator $\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}$ we have

$$V_1 = \left(I - \gamma P^{c\mu}\right)^{-1} R^{\pi_\theta} + \left(I - \gamma P^{c\mu}\right)^{-1} \gamma \left(P^{\pi_\theta} - P^{c\mu}\right) V.$$

Since V^{π_θ} is the fixed point of the operator, we can subtract both sides by V^{π_θ}. This produces

$$V_1 - V^{\pi_\theta} = \left(I - \gamma P^{c\mu}\right)^{-1} \gamma \left(P^{\pi_\theta} - P^{c\mu}\right) \left(V - V^{\pi_\theta}\right).$$

When the trace cofficient c is smoothly differentiable in π, and under Assumption 3, we deduce that $\left(I - \gamma P^{c\mu}\right)^{-1} \gamma \left(P^{\pi_\theta} - P^{c\mu}\right)$ is differentiable in θ_i. Let $g_1 \coloneqq \nabla_{\theta_j} V_1 \in \mathbb{R}^{|\mathcal{X}|}$ and $g \coloneqq \nabla_{\theta_j} V^{\pi_\theta} \in \mathbb{R}^{|\mathcal{X}|}$. The gradient vector g_1 satisfies the following recursive equation, with $g_0 \coloneqq \nabla_{\theta_j} V = 0$ obtained by taking derivative of both sides above w.r.t. θ_i,

$$g_1 - g = \nabla_{\theta_i} \left[\left(I - \gamma P^{c\mu}\right)^{-1} \gamma \left(P^{\pi_\theta} - P^{c\mu}\right)\right] \left(V - V^{\pi_\theta}\right) + \left(I - \gamma P^{c\mu}\right)^{-1} \gamma \left(P^{\pi_\theta} - P^{c\mu}\right) \left(g_0 - g\right).$$

When $V = V^{\pi_\theta}$, the first term vanishes and note that the matrix $\left(I - \gamma P^{c\mu}\right)^{-1} \gamma \left(P^{\pi_\theta} - P^{c\mu}\right)$ has operator norm upper bounded by $\eta \leq \gamma$ (Munos et al., 2016). We hence deduce the following inequality which concludes the proof

$$\|g_1 - g\|_\infty \leq \eta \|g_0\|_\infty.$$

A.4. Proof of Theorem 5

By construction of the V-trace operator, it is straightforward to verify that the following

$$\widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}} V(x) \coloneqq V(x) + \sum_{t=0}^{\infty} \gamma^t c_{0:t-1} \rho_t \delta_t$$

is an unbiased estimate to $\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu} V(x)$. Now, since we assume the trajectory is of finite length almost surely and since the importance sampling ratio $\rho_t \leq \max_{x,a} \frac{\pi_\theta(a|x)}{\mu(a|x)}$ is upper bounded, we can verify that we can apply the dominated convergence theorem to the limiting sequence

$$\frac{1}{\delta_j}\left(\widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta+\delta_j,\mu}} V(x) - \widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta+\delta,\mu}} V(x)\right)$$

with $\|\delta_j\|_2 \to 0$, which implies $\mathbb{E}_\mu\left[\nabla_\theta \widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}} V(x)\right] = \nabla_\theta \mathcal{R}_{\bar{c}}^{\pi_\theta,\mu} V(x)$. and hence $\nabla_\theta \widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}} V(x)$ is an unbiased gradient estimate.

B. Experiment details and additional results

We present further experiment details and results.

B.1. Tabular experiments on VI

Figure 1. We compare DoMo-VI, multi-step policy evaluation, multi-step policy optimization and one-step baseline VI. All experiments are carried out on tabular MDPs with $|\mathcal{X}| = 20$ states $|\mathcal{A}| = 5$ actions. The transition $p(\cdot|x,a)$ is generated as Dirichlet random variable with parameter $(\alpha, ...\alpha) \in \mathbb{R}^{\mathcal{X}}$ for $\alpha = 0.01$. The reward R_0 is sampled from a standard normal distribution and kept fixed. The discount factor $\gamma = 0.9$. For all multi-step variants, we set $\bar{c} = 10$.

We carry out recursions based on different algorithms and report the approximation error to the optimal value function $\|V^{\pi_i} - V^*\|_2$. All results are repeated 100 times with randomly generated MDPs. For implementing DoMo-VI and

multi-step policy optimization, we need to approximately solve the optimization problem $\arg\max_{\pi\in\Pi} \mathcal{R}_{\bar{c}}^{\pi,\mu} V(x)$.. To this end, we parameterize policy $\pi_\theta(a|x) = \text{softmax}(\theta(x,a))$ and carry out gradient ascent on the objective below until convergence.

$$L(\theta) = \frac{1}{|\mathcal{X}|} \sum_{x=1}^{|\mathcal{X}|} \mathcal{R}_{\bar{c}}^{\pi_\theta,\mu} V(x). \tag{8}$$

Figure 2. We compare DoMo-AC, multi-step policy evaluation and one-step baseline VI. All experiments are carried out using the same setup as above. Notably, DoMo-AC is an approximation to DoMo-VI in that the policy optimization stage is not necessarily carried out in full. At iteration i, let π_g be the current greedy policy with respect to V_i, we initialize a softmax policy with parameter $\theta_{i+1}^{(1)}$ such that

$$\theta_{i+1}^{(1)}(x,a) = \log\left(\pi_g(a|x) + 10^{-5}\right).$$

This is such that the softmax policy defined with $\theta_{i+1}^{(1)}$ is close to π_i. This initialization is intended such that when there is no gradient update, the performance of DoMo-AC is similar to the multi-step policy evaluation baseline (with one-step greedy). We then carry out gradient updates on the objective $L(\theta_{i+1}^{(j)})$ as defined in Eqn (8) for N steps. The final iterate $\theta_{i+1}^{(N)}$ is used for defining the policy π_{i+1} at the next iteration. All results are repeated for 100 times across randomly generated MDPs.

B.2. Deep RL experiments

All evaluation environments are the entire suite of Atari games (Bellemare et al., 2013) consisting of 57 levels. Since each level has a very different reward scale and difficulty, we report human-normalized scores for each level, calculated as $z_i = (r_i - o_i)/(h_i - o_i)$, where h_i, o_i are performances of human and a random policy on level i respectively.

For all experiments, we report summarizing statistics of the human-normalized scores across all levels. For example, at any point in training, the mean human-normalized score is the mean statistics across $z_i, 1 \le i \le 57$.

Distributed training. Distributed algorithms have led to significant performance gains on challenging domains (Nair et al., 2015; Mnih et al., 2016; Babaeizadeh et al., 2016; Barth-Maron et al., 2018; Horgan et al., 2018). Here, our focus is on recent state-of-the-art algorithms. In general, distributed agents consist of one central learner, multiple actors and optionally a replay buffer. The central learner maintains a parameter copy θ and updates parameters based on sampled data. Multiple actors each maintaining a slighted delayed parameter copy θ_{old} and interact with the environment to generate partial trajectories. Actors sync parameters from the learner periodically. In the actor-critic setting, the behavior policy is executed using the delayed copy such that $\mu = \pi_{\theta_{\text{old}}}$.

Details on the distributed architecture. The general policy-based distributed agent follows the architecture design of IMPALA (Espeholt et al., 2018), i.e. a central GPU learner and $N = 512$ distributed CPU actors. The actors keep generating data by executing their local copies of the policy μ, and sending data to the queue maintained by the learner. The parameters are periodically synchronized between the actors and the learner, as discussed above.

The architecture details are the same as those in (Espeholt et al., 2018). For completeness, we present some important details below, please refer to the original paper for other missing details. See the paper for further details.

The policy/value function networks are both trained by RMSProp optimizers (Tieleman et al., 2012) with learning rate $\alpha = 5 \cdot 10^{-4}$ and no momentum. To encourage exploration, the policy loss is augmented by an entropy regularization term with coefficient $c_e = 0.01$ and baseline loss with coefficient $c_v = 0.5$, i.e. the full loss $L = L_{\text{policy}} + c_v L_{\text{value}} + c_e L_{\text{entropy}}$. These single hyper-parameters are selected according to Appendix D of (Espeholt et al., 2018).

Actors send partial trajectories of length $T = 20$ to the learner. For robustness of the training, rewards R_t are clipped between $[-1, 1]$. We adopt frame stacking and sticky actions as commonly practiced (Mnih et al., 2013). The discount factor $\gamma = 0.99$ for calculating the baseline estimations.

V-trace value learning implementations. The targets for value learning $V_{\text{target}}(X_t)$ in Algorithm 1 are computed via V-trace. V-trace is a competitive baseline for correcting off-policy data (Espeholt et al., 2018). Given a partial trajectory

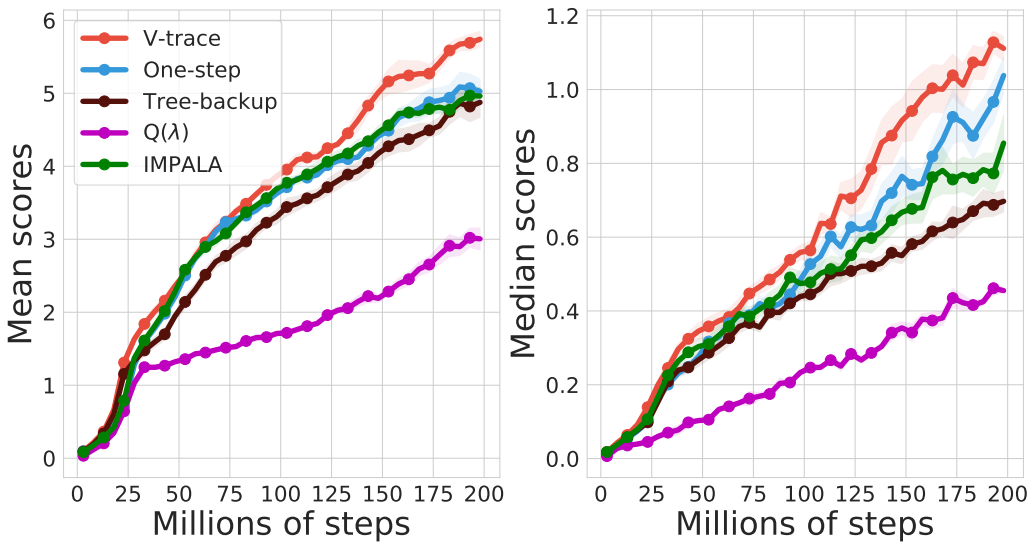

Figure 4. Full results for the Atari game suites, and comparison across various baseline operators. We show the mean and median performance of baseline algorithms across all 57 Atari games. Overall, we see that V-trace retains performance advantage compared to other alternative off-policy evaluation operators when applied under the DoMo-AC framework.

$(X_t, A_t, R_t)_{t=1}^T$, let $\tilde{\rho}_t = \min\{\bar{\rho}, \rho_t\}$ be the truncated IS ratio. Let $v(x)$ be the a certain value function baseline (e.g., we let the baseline be computed by the value network $v(x) = V_\phi(x)$). V-trace targets are calculated recursively for all $1 \leq t \leq T$ backward in time:

$$V_{\text{target}}(X_t) = v(X_t) + \tilde{\rho}_t \delta_t + \gamma c_t \left(V_{\text{target}}(X_{t+1}) - v(X_t) \right), \tag{9}$$

where $\tilde{\rho}_t = \min(\bar{\rho}, \rho_t)$ is a truncated IS ratio and $c_t = \min(\bar{c}, \rho_t)$ is the trace coefficient. When $t = T$, we initialize $V_{\text{target}}(X_t) = v(X_t)$. In practice, it is common to set $\bar{\rho} < \infty$ to avoid explosion of the IS ratio; though this introduces extra bias into the gradient estimate. The value function baseline is then trained to approximate these targets $V_\phi(x) \approx v(x)$. Following (Espeholt et al., 2018), we set $\bar{\rho} = \bar{c} = 1$.

Implementation details of DoMo-AC. We build the DoMo-AC gradient estimate on top of the V-trace recursive estimate in Eqn (9). Note that we can think of $V_{\text{target}}(X_t)$, as computed above, as a function of parameter θ as $c_t = \min(\rho_t, \bar{c})$ where $\rho_t = \pi_\theta(A_t|X_t)/\mu(A_t|X_t)$. We can understand $V_{\text{target}}(X_t)$ as effectively the estimated back-up target $\widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}} v(X_t)$ and compute the DoMo-AC gradient estimate by differentiating through $V_{\text{target}}(X_t)$ via auto-diff. In calculating the back-up targets for value learning, we use $v(X_t) = V_\phi(X_t)$; however, for estimating policy gradient, we find that the algorithm works better with $v(X_t) = V_{\text{target}}(X_t)$. We speculate that this is because policy gradient estimates would benefit from a more accurate baseline, and the V-trace estimate $V_{\text{target}}(X_t)$ provides a more accurate approximation to the true value function compared to the baseline.

Alternative evaluation operators for deep RL experiments. All operators take the same form as the V-trace operator in Eqn (1) but differ in the choice of trace coefficient c_t. We consider a few alternatives: (1) By default, the V-trace operator with Retrace trace $c_t = \min(\rho_t, \bar{c})$ with $\bar{c} = 0.5$. We will examine the sensitivity to the threshold $\bar{c}$ in ablation study; (2) The one-step trace, $c_t = 0$, which instantiates the actor-critic instantiation of the multi-step policy evaluation recursion. It turns out that such an algorithm closely resembles the original IMPALA implementation; (3) Tree back-up trace $c_t = \pi(A_t|X_t)$; (4) Q(λ) trace with $c_t = \lambda = 0.7$. Finally, we also compare with the IMPALA baseline (Espeholt et al., 2018).

C. Discussion on truncated operators

In tabular experiments, though the back-up target $\nabla_\theta \mathcal{R}_{\bar{c}}^{\pi_\theta,\mu} V(x)$ is defined with an infinite horizon, it can be computed analytically using matrix inverse and auto-diff. In large-scale experiments, gradients are computed based on sampled trajectories. Since the partial trajectories are of length T, we can understand the practical algorithm as being derived from

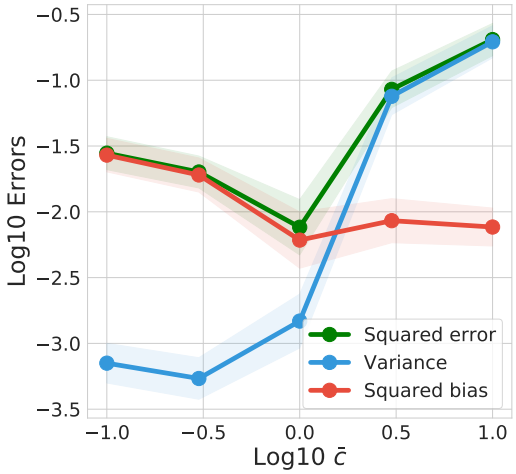

Figure 5. The bias-variance trade-off of the stochastic estimate $\nabla_\theta \widehat{\mathcal{R}_{\bar{c}}^{\pi_\theta,\mu}} V(x)$ against the true policy gradient $\nabla_\theta V^{\pi_\theta}(x)$ on a number of randomly generated MDPs. As $\bar{c}$ increases, the bias generally decreases but the variance increases. Overall, this leads to an optimal middle ground for the choice of $\bar{c}$. See Appendix B for more details on the experimental setups.

the equivalent off-policy evaluation operator takes the truncated form

$$\mathcal{R}_{T,\bar{c}}^{\pi,\mu} V(x) \coloneqq V(x) + \mathbb{E}_\mu \left[\sum_{t=0}^{T-1} \gamma^t c_{0:t-1} \rho_t \delta_t \right]. \tag{10}$$

The truncated operator enjoys similar theoretical properties as the non-truncated operator $\mathcal{R}_{\bar{c}}^{\pi,\mu}$, such as the fixed point V^π and accelerated contraction rate compared to the one-step operator.

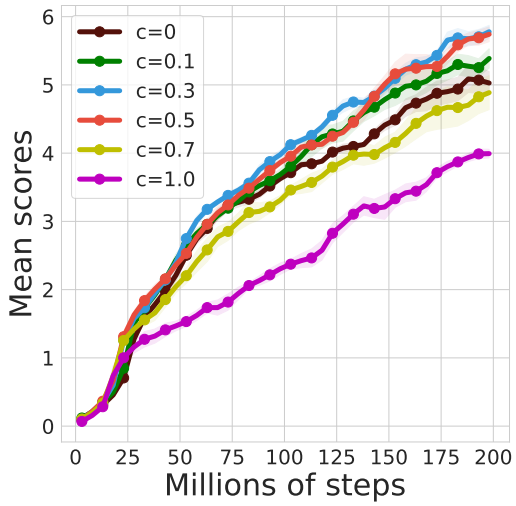

Figure 6. Ablation study on the effect of the trace coefficient threshold $\bar{c}$ for the V-trace operator in DoMo-AC algorithm. Going from $\bar{c} = 0$ to $\bar{c} = 1$, the evaluated performance throughout training first increases and then decreases. The best-performing value of $\bar{c}$ seems to be between 0.3 and 0.5, where the best bias-variance trade-off is obtained.