Taylor Expansions of Discount Factors

Yunhao Tang [1] **Mark Rowland** [2] **Rémi Munos** [3] **Michal Valko** [3]

Abstract

In practical reinforcement learning (RL), the discount factor used for estimating value functions often differs from that used for defining the evaluation objective. In this work, we study the effect that this discrepancy of discount factors has during learning, and discover a family of objectives that interpolate value functions of two distinct discount factors. Our analysis suggests new ways for estimating value functions and performing policy optimization updates, which demonstrate empirical performance gains. This framework also leads to new insights on commonly-used deep RL heuristic modifications to policy optimization algorithms.

1. Introduction

One of the most popular models for reinforcement learning (RL) is the Markov decision process (MDP) with exponential discounting over an infinite horizon (Sutton and Barto, 2018; Puterman, 2014), with discounted objectives of the following form

$$V_\gamma^\pi(x) = \mathbb{E}_\pi\left[\sum_{t=0}^\infty \gamma^t r_t \,\middle|\, x_0 = x\right].$$

Discounted models enjoy favorable theoretical properties, and are also the foundation of many practical RL algorithms that enjoy empirical success (e.g. see (Mnih et al., 2015; Schulman et al., 2015a; Lillicrap et al., 2015; Schulman et al., 2017)). However, in most applications of RL, the objective of interest is the expected *undiscounted cumulative return*,

$$\mathbb{E}_\pi\left[\sum_{t=0}^T r_t \,\middle|\, x_0 = x\right], \tag{1}$$

where $T < \infty$ is a (possibly random) evaluation horizon, which usually also denotes the end of the trajectory. For example, T could be the first time the MDP gets into a terminal state (e.g., a robot falls); when the MDP does not have a natural terminal state, T could be enforced as a deterministic horizon. This creates a technical gap between algorithmic developments and implementations: it is tempting to design algorithms that optimize $V_\gamma^\pi(x)$, however, further heuristics are often needed to get strong practical performance. This issue manifests itself with the policy gradient (PG) theorem (Sutton et al., 2000). Let π_θ be a parameterized policy. The policy gradient (PG) $\nabla_\theta V_\gamma^{\pi_\theta}(x)$ is computed as

$$\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^\infty \gamma^t Q_\gamma^{\pi_\theta}(x_t, a_t)\nabla_\theta \log \pi_\theta(a_t|x_t) \,\middle|\, x_0 = x\right]. \tag{2}$$

However, the practical implementation of PG updates usually omits the discount factors (see for example the high-quality open source packages (Dhariwal et al., 2017; Achiam and OpenAI, 2018), leading to an approximate gradient of the form

$$\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^T Q_\gamma^{\pi_\theta}(x_t, a_t)\nabla_\theta \log \pi_\theta(a_t|x_t) \,\middle|\, x_0 = x\right]. \tag{3}$$

Most prior work on PG algorithms rely on this heuristic update to work properly in deep RL applications. The intuitive argument for dropping the factor γ^t is that Eqn (2) optimizes $V_\gamma^{\pi_\theta}(x)$, which is very myopic compared to the objective in Eqn (1). Consequently, the exponential discount γ^t is too aggressive for weighting updates with large t. As a concrete example, in many MuJoCo control tasks (Brockman et al., 2016), the most commonly used discount factor is $\gamma = 0.99$. This leads to an effective horizon of $\frac{1}{1-\gamma} = 100$, which is much smaller than the evaluation horizon $T = 1000$. This technical gap between theory and practice has been alluded to previously (by e.g., O'Donoghue et al., 2016) and is explicitly discussed by Nota and Thomas (2019).

To bypass this gap, a straightforward solution would be to naïvely increase the discount factor $\gamma \geq 1 - \frac{1}{T}$ and apply the PG in Eqn (2). In the example above, this implies using $\gamma \geq 0.999$. Unfortunately, this rarely works well in practice, as we will also see in experiments. The failure might be due to the higher variance of the estimation (Schulman et al., 2015b) or the collapse of the action gaps (Lehnert et al., 2018; Laroche and van Seijen, 2018), which is aggravated when combined with function approximations.

[1]Columbia University, New York, USA [2]DeepMind, London, UK [3]DeepMind, Paris, France. Correspondence to: yt2541@columbia.edu <Yunhao>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

Nevertheless, as a theoretical framework, it is insightful to emulate the undiscounted objective in Eqn (1) using the (un)discounted objective $V^\pi_{\gamma'}(x)$ with $\gamma' \geq 1 - \frac{1}{T}$. To build intuitions about this approximation, note that when the time step is small $t \ll T$, the multiplicative factor $(\gamma')^t \approx 1$ and the cumulative rewards are almost undiscounted; even when $t = T$, we have $(\gamma')^t \geq (1 - \frac{1}{T})^T \approx \frac{1}{e} \gg 0$. Overall, this is a much more accurate approximation than $V^\pi_\gamma(x)$. This naturally prompts us to answer the following general question: *How do we evaluate and optimize $V^\pi_{\gamma'}(x)$ with estimates built for $V^\pi_\gamma(x)$ where $0 < \gamma < \gamma' \leq 1$?*

Main idea. We study the relation between $V^\pi_\gamma(x)$ and $V^\pi_{\gamma'}(x)$ via Taylor expansions. In Section 3, we identify a family of interpolating objectives between the more myopic objective $V^\pi_\gamma(x)$ and the true objective of interest $V^\pi_{\gamma'}(x)$. In Section 4, we start with insights on why the heuristic in Eqn (3) might be useful in practice. Then, we apply Taylor expansions directly to the heuristic updates, to arrive at a family of interpolating updates. In Section 5, we build on theoretical insights to derive improvements to established deep RL algorithms. We show their performance gains in Section 7.

2. Background

Consider the setup of a MDP. At any discrete time $t \geq 0$, the agent is in state $x_t \in \mathcal{X}$, takes an action $a_t \in \mathcal{A}$, receives an instant reward $r_t = r(x_t, a_t) \in [0, R_{\max}]$ and transitions to a next state $x_{t+1} \sim p(\cdot|x_t, a_t)$. For simplicity, we assume $r(x, a)$ to be deterministic. Let policy $\pi : \mathcal{X} \to \mathcal{P}(\mathcal{A})$ be a mapping from states to distributions over actions. Let $\gamma \in [0, 1)$ be a discount factor, define the Q-function $Q^\pi_\gamma(x, a) := \mathbb{E}_\pi \left[\sum_{t=0}^\infty \gamma^t r_t \mid x_0 = x, a_0 = a\right]$ and value function $V^\pi_\gamma(x) := \mathbb{E}_\pi \left[\sum_{t=0}^\infty \gamma^t r_t \mid x_0 = x\right]$. We also define the advantage function $A^\pi_\gamma(x, a) := Q^\pi_\gamma(x, a) - V^\pi_\gamma(x)$. Here, $\mathbb{E}_\pi[\cdot]$ denotes that the trajectories $(x_t, a_t, r_t)_{t=0}^\infty$ are generated under policy π. Throughout the paper, we use subscripts γ to emphasize that RL quantities implicitly depend on discount factors.

2.1. Linear programs for reinforcement learning

Henceforth, we assume all vectors to be column vectors. The value functions V^π_γ satisfy the Bellman equations $V^\pi_\gamma(x) = \mathbb{E}_\pi \left[r(x, a) + \gamma V^\pi_\gamma(x') \mid x_0 = x\right]$ (Bellman, 1957). Such equations can be encoded into a linear program (LP) (De Farias and Van Roy, 2003; Puterman, 2014). Let $V \in \mathbb{R}^\mathcal{X}$ be the primal variables, consider the following LP,

$$\max \ \delta_x^T V, \ \ V = r^\pi + \gamma P^\pi V, \tag{4}$$

where $r^\pi \in \mathbb{R}^\mathcal{X}$ is the state-dependent reward $r^\pi(x') := \sum_{a'} \pi(a'|x') r(x', a')$ and $P^\pi \in \mathbb{R}^{\mathcal{X} \times \mathcal{X}}$ is the transition matrix under π. Here, $\delta_x \in \mathbb{R}^\mathcal{X}$ encodes the one-hot distribution (Dirac) at x. Similar results hold for considering the LP objective $v^T V$ with a general distribution $v \in \mathcal{P}(\mathcal{X})$. It then follows that the optimal solution to the above LP is $V^* = V^\pi_\gamma$. Now, consider the dual LP to Eqn (4), let $d \in \mathbb{R}^\mathcal{X}$ be the dual variables,

$$\min \ (1 - \gamma)^{-1} (r^\pi)^T d, \ \ d = (1 - \gamma)\delta_x + \gamma (P^\pi)^T d. \tag{5}$$

The optimal solution to the dual program has a natural probabilistic interpretation. It is the discounted visitation distribution $d^\pi_{x,\gamma}$ under policy π with starting state x as $d^\pi_{x,\gamma}(x') := (1 - \gamma) \sum_{t \geq 0} \gamma^t P_\pi(x_t = x' | x_0 = x)$ where $P_\pi(x_t = x' | x_0 = x)$ is a probability measure induced by the policy π and the MDP transition kernel. By strong duality, the value function can be equivalently written as

$$V^\pi_\gamma(x) = \frac{1}{1 - \gamma} \mathbb{E}_{x' \sim d^\pi_{x,\gamma}, a' \sim \pi(\cdot|x')} \left[r(x', a')\right]. \tag{6}$$

3. Taylor Expansions of Value Functions

Below, we show how to estimate $V^\pi_{\gamma'}(x)$ with approximations constructed from value functions $V^\pi_\gamma(x)$ for $\gamma < \gamma'$. Unless otherwise stated, we always assume $\gamma' < 1$ for a more convenient mathematical treatment of the problem.

3.1. Taylor expansions of discount factors

We start with some notations: we abuse the notation of value functions $V^\pi_\gamma \in \mathbb{R}^\mathcal{X}$ to both refer to the scalar function as well as a vector. The Bellman equation for the value-function is expressed in the matrix form (Puterman, 2014)

$$V^\pi_{\gamma'} = r^\pi + \gamma' P^\pi V^\pi_{\gamma'}. \tag{7}$$

Inverting the equation,

$$V^\pi_{\gamma'} = (I - \gamma' P^\pi)^{-1} r^\pi. \tag{8}$$

Now, we present the main result of Taylor expansions.

Proposition 3.1. The following holds for all $K \geq 0$,

$$V^\pi_{\gamma'} = \sum_{k=0}^K \left((\gamma' - \gamma)(I - \gamma P^\pi)^{-1} P^\pi\right)^k V^\pi_\gamma + \underbrace{\left((\gamma' - \gamma)(I - \gamma P^\pi)^{-1} P^\pi\right)^{K+1} V^\pi_{\gamma'}}_{\text{residual}}. \tag{9}$$

When $\gamma < \gamma' < 1$, the residual norm converges to 0, which implies

$$V^\pi_{\gamma'} = \sum_{k=0}^\infty \left((\gamma' - \gamma)(I - \gamma P^\pi)^{-1} P^\pi\right)^k V^\pi_\gamma. \tag{10}$$

We provide a proof sketch here: Note that $\gamma' P^\pi = (\gamma' - \gamma)P^\pi + \gamma P^\pi$ and apply the Woodbury matrix identity to obtain $(I - \gamma' P^\pi)^{-1} = (I - \gamma P^\pi)^{-1} + (\gamma' - \gamma)(I - \gamma P^\pi)^{-1} P^\pi (I - \gamma' P^\pi)^{-1}$. We can then recursively expand Eqn (8) K times to arrive at Eqn (9). In particular, by expanding the equation once, we see that $(I - \gamma' P^\pi)^{-1}$ is equivalent to the following,

$$(I - \gamma P^\pi)^{-1} + (\gamma' - \gamma)(I - \gamma P^\pi)^{-1} P^\pi (I - \gamma P^\pi)^{-1} + (\gamma' - \gamma)^2 \left((I - \gamma P^\pi)^{-1} P^\pi\right)^2 \underbrace{(I - \gamma' P^\pi)^{-1}}_{\text{can be expanded further}},$$

where the last term can be expanded further by plugging in the Woodbury matrix identity. See the complete proof in Appendix A.

Extensions to $\gamma' = 1$. The above result can extend to the case $\gamma' = 1$. We make two assumptions: **A.1** The Markov chain induced by π is absorbing and T is the absorption time; **A.2** $r^\pi(x) = 0$ for absorbing states x. Under these assumptions, we can interpret such absorbing states as the terminal states. As a result, $V^\pi_{\gamma'=1}(x) = \mathbb{E}_\pi \left[\sum_{t=0}^T r_t \,\middle|\, x_0 = x\right]$ is well-defined and Proposition 3.1 still holds; see Appendix A for the complete proof.

In practice, it is infeasible to sum up all infinite number of terms in the Taylor expansion. It is then of interest to consider the K^{th}-order expansion of $V^\pi_{\gamma'}$, which truncates the infinite series. Specifically, we define the K^{th}-order expansion as

$$V^\pi_{K,\gamma,\gamma'} \coloneqq \sum_{k=0}^{K} ((\gamma' - \gamma)(I - \gamma P^\pi)^{-1} P^\pi)^k V^\pi_\gamma \,. \quad (11)$$

As K increases, the K^{th} order expansion becomes increasingly close to the infinite series, which evaluates to $V^\pi_{\gamma'}(x)$. This is formalized next.

Proposition 3.2. The following bound holds for all $K \geq 0$,

$$\left|V^\pi_{\gamma'}(x) - V^\pi_{K,\gamma,\gamma'}(x)\right| \leq \left(\frac{\gamma' - \gamma}{1 - \gamma}\right)^{K+1} \frac{R_{\max}}{1 - \gamma'}. \quad (12)$$

3.2. Sample-based approximations of Taylor expansions

We now describe how to estimate $V^\pi_{K,\gamma,\gamma'}(x)$ via samples. First, we build some intuition on the behavior of expansions at different orders K by considering a few special cases.

Zeroth-order expansion. By setting $K = 0$, we see that

$$V^\pi_{0,\gamma,\gamma'} = V^\pi_\gamma. \quad (13)$$

The zeroth order expansion approximates the value function $V^\pi_{\gamma'}(x)$ of the discount factor γ' with that $V^\pi_\gamma(x)$ of a lower discount factor $\gamma < \gamma'$. This is a very straightforward approximation to use in that no sampling at all is required, but it may not be accurate.

First-order expansion. When $K = 1$, we consider the increments of the expansions,

$$V^\pi_{1,\gamma,\gamma'} - V^\pi_{0,\gamma,\gamma'} = (\gamma' - \gamma)(I - \gamma P^\pi)^{-1} P^\pi V^\pi_\gamma. \quad (14)$$

To understand the first order expansion, recall that in the definition of value function $V^\pi_\gamma = (I - \gamma P^\pi)^{-1} r^\pi$, immediate rewards r^π are *accumulated* via the matrix $(I - \gamma P^\pi)^{-1}$. In general, for any $X, Y \in \mathbb{R}^{\mathcal{X}}$, we can interpret $X = (I - \gamma P^\pi))^{-1} Y$ as accumulating Y as rewards to compute X as value functions. By analogy, we can interpret the RHS of Eqn (14) as the value function assuming $(\gamma' - \gamma)P^\pi V^\pi_\gamma$ as immediate rewards. In other words, the first order expansion bootstraps the zeroth order expansion V^π_γ to form a more accurate approximation. Combined with the zeroth order expansion, we can also conveniently write the difference of first- and zeroth-order expansions as an expectation $V^\pi_{1,\gamma,\gamma'}(x) - V_{0,\gamma,\gamma'}(x) = (\gamma' - \gamma)\mathbb{E}_\pi \left[\sum_{t=1}^\infty \gamma^{t-1} V^\pi_\gamma(x_t) \,\middle|\, x_0 = x\right]$. Let $\tau \sim \text{Geometric}(1 - \gamma)$ be a random time such that $P(\tau = t) = (1 - \gamma)\gamma^t, \forall t \in \mathbb{Z}_{\geq 1}$. The difference can also be expressed via this random time

$$V^\pi_{1,\gamma,\gamma'}(x) - V_{0,\gamma,\gamma'}(x) = \frac{\gamma' - \gamma}{1 - \gamma} \mathbb{E}_{\pi,\tau} \left[V^\pi_\gamma(x_\tau)\right].$$

Note that from this expression, we obtain a simple unbiased estimate for $V^\pi_{1,\gamma,\gamma'}(x) - V_{0,\gamma,\gamma'}(x)$, using a sampled trajectory and a random time step τ.

General K^{th}-order expansion. We now present results for general K. Consider the incremental term,

$$V^\pi_{K,\gamma,\gamma'} - V^\pi_{K-1,\gamma,\gamma'} = (\gamma' - \gamma)^K \left((I - \gamma P^\pi)^{-1} P^\pi\right)^K V^\pi_\gamma. \quad (15)$$

Note that the aggregate matrix $\left((I - \gamma P^\pi)^{-1} P^\pi\right)^K$ suggests a recursive procedure to bootstrap from lower order expansions to construct higher order expansions. To see why, we can rewrite the right-hand side of Eqn (15) as

$$(\gamma' - \gamma)(I - \gamma P^\pi)^{-1} P^\pi \left(V^\pi_{K-1,\gamma,\gamma'} - V^\pi_{K-2,\gamma,\gamma'}\right).$$

Indeed, we can interpret the difference $V^\pi_{K,\gamma,\gamma'} - V^\pi_{K-1,\gamma,\gamma'}$ as the value function under the immediate reward $(\gamma' - \gamma)P^\pi \left(V^\pi_{K-1,\gamma,\gamma'} - V^\pi_{K-2,\gamma,\gamma'}\right)$. This generalizes the bootstrap procedure of the first order expansion as a special case where we naturally assume $V^\pi_{-1,\gamma,\gamma'} = 0$. Given K

i.i.d. random times $\tau_i \sim \text{Geometric}(1-\gamma)$, we can write $V^\pi_{K,\gamma,\gamma'}(x) - V^\pi_{K-1,\gamma,\gamma'}(x)$ as the expectation

$$\left(\frac{\gamma'-\gamma}{1-\gamma}\right)^K \mathbb{E}_{\tau_i, 1\le i\le K}\left[V^\pi_\gamma\left(x_{\tau_1+\cdots+\tau_K}\right)\right].$$

Based on the above expression, Algorithm 1 provides a subroutine that generates unbiased estimates of $V^\pi_{K,\gamma,\gamma'}(x)$ by sub-sampling an infinite trajectory $(x_t, a_t, r_t)_{t=0}^\infty$ with the random times.

Practical implementations. While the above and Algorithm 1 show how to compute one-sample estimates, in practice, we might want to average multiple samples along a single trajectory for variance reduction. See Appendix F for further details on the practical estimates.

Algorithm 1 Estimating the K^{th} order expansion

Require: A trajectory $(x_t, a_t, r_t)_{t=0}^\infty \sim \pi$ and discount factors $\gamma < \gamma' < 1$

1. Compute an unbiased estimate $\widehat{V}^\pi_\gamma(x_t)$ for states along the trajectory, e.g., $\widehat{V}^\pi_\gamma(x_t) = \sum_{t'\ge t} \gamma^{t'-t} r_{t'}$.
2. Sample K random time $\{\tau_i\}_{1\le i\le K}$, all i.i.d. geometrically distributed $\tau_i \sim \text{Geometric}(1-\gamma)$.
3. Return the unbiased estimate $\sum_{k=0}^K \left(\frac{\gamma'-\gamma}{1-\gamma}\right)^k \widehat{V}^\pi_\gamma(x_{t_k})$ where $t_k = \sum_{i=1}^k \tau_i$.

Interpretation of expansions in the dual space. Recall that $V^\pi_{\gamma'} = (I-\gamma' P^\pi)^{-1} r^\pi = I(I-\gamma' P^\pi)^{-1} r^\pi$ where the identity matrix $I = [\delta_0, \delta_1, ...\delta_{\mathcal{X}}]$ concatenates Dirac delta vectors $\delta_x, \forall x \in \mathcal{X}$. Since r^π is a constant vector, Taylor expansions essentially construct approximations to the matrix $(I-\gamma' P^\pi)^{-1}$. By grouping the matrix with the reward vector (or the density matrix), we arrive at the primal expansion (or the dual expansion),

$$I \underbrace{(I-\gamma' P^\pi)^{-1} r^\pi}_{\text{primal expansions of } V^\pi_{\gamma'}(x)} = \underbrace{I(I-\gamma' P^\pi)^{-1}}_{\text{dual expansions of } d^\pi_{x,\gamma'}} r^\pi$$

The derivations above focus on the primal expansion view. We show a parallel theory of dual expansion in Appendix B. The equivalence of primal-dual view of Taylor expansions suggests connections with seemingly disparate lines of prior work: Janner et al. (2020) propose a density model for visitation distribution of different γ in the context of model-based RL. They show that predictions of large discount factors could be bootstrapped from predictions of small discount factors. This corresponds exactly to the dual space expansions, which is equivalent to the primal space expansions.

Extensions to Q-functions. In Appendix C, we show that it is possible to build approximations to $Q^\pi_{\gamma'}$ using Q^π_γ as building blocks. The theoretical guarantees and estimation procedures are similar to the case of value functions.

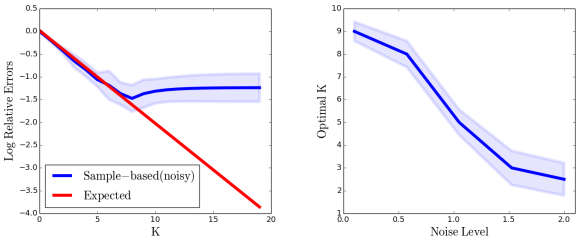
3.3. Approximation errors with finite samples

Proposition 3.2 shows that the *expected* approximation error decays as $\left|V^\pi_{K,\gamma,\gamma'}(x) - V^\pi_{\gamma'}(x)\right| = O\left(\left(\frac{\gamma'-\gamma}{1-\gamma}\right)^{K+1}\right)$ for $\gamma < \gamma' < 1$. This motivates using a high value of K when constructing the approximation. However, in practice, all constituent terms in the K^{th} order expansion are random estimates, each with a non-zero variance. This might lead the variance of the overall estimate to increase as K increases. As a result, K mediates a trade-off between bias (expected approximation error) and variance. We formalize such intuitions in Appendix E, where we theoretically analyze the trade-off using the phased TD-learning framework (Kearns and Singh, 2000).

A numerical example. To get direct intuition about the effect of K, we focus on a tabular MDP example. The MDP has $|\mathcal{X}| = 10$ states and $|\mathcal{A}| = 2$ actions. All entries of the transition table $p(y|x,a)$ are generated from a Dirichlet distribution with parameters $(\alpha, \ldots, \alpha)$ with $\alpha = 0.01$. The policy $\pi(a|x)$ is uniformly random. We take $\gamma = 0.2$ and $\gamma' = 0.8$. The agent generates $N = 10$ trajectories $(x_t, a_t, r_t)_{t=0}^T$ with a very large horizon T with a fixed starting state x_0. We assume access to base estimates $\widehat{V}^\pi_\gamma(x_t)$ and the Taylor expansion estimates $\widehat{V}^\pi_{K,\gamma,\gamma'}(x_0)$ are computed based on Algorithm 1. We estimate the relative error as $\widehat{E}_K(x_0) = \left|V^\pi_{\gamma'}(x_0) - \widehat{V}^\pi_{K,\gamma,\gamma'}(x_0)\right|$. For further experiment details, see Appendix F.

In Figure 1(a), we show how errors vary as a function of K. We study two settings: **(1)** Expected estimates (red), where $\widehat{V}^\pi_{K,\gamma,\gamma'}(x_0)$ is computed analytically through access to transition tables. In this case, similar to how the theory suggests, the error decays exponentially; **(2)** Sample-based estimates (blue) with base estimates $\widehat{V}^\pi_\gamma(x_t) = \sum_{s=0}^\infty \gamma^s r_{t+s}$. The errors decay initially with K but later start to increase a bit as K gets large. The optimal K in the middle achieves the best bias-variance trade-off. Note that in this particular example, the estimates do not pay a very big price in variance for large K. We speculate this is because increments to the estimates are proportional to $\left(\frac{\gamma'-\gamma}{1-\gamma}\right)^{K+1}$, which scales down additional variance terms quickly as K increases.

In Figure 1(b), we study how the optimal expansion order K^* depends on the noise level of base estimates. To emulate the noise, we assume access to base estimates $\widehat{V}^\pi_\gamma(x_t) = V^\pi_\gamma(x_t) + \mathcal{N}(0, \sigma^2)$ for some noise level σ. The optimal order K^* is computed as $K^* = \arg\min_k \widehat{E}_k(x_0)$. In general, we observe that when σ increases, K^* decreases. Intuitively, this implies that as the base estimates $\widehat{V}^\pi_\gamma(x)$ become noisy, we should prefer smaller value of K to control the variance. This result bears some insights for practical applications such as downstream policy optimization, where



(a) Trade-off of K (b) Optimal K

Figure 1. Comparison of Taylor expansions with different orders. The x-axis shows the order K, the y-axis shows the log relative errors of the approximations. The blue curve shows the exact computations while the red curve shows the sample based estimations. See Appendix F for more details.

we need to select an optimal K for the tasks at hand.

4. Taylor Expansions of Gradient Updates

In Section 3, we discussed how to construct approximations to $V^{\pi}_{\gamma'}(x)$. For the purpose of policy optimization, it is of direct interest to study approximations to $\nabla_\theta V^{\pi_\theta}_{\gamma'}(x)$. As stated in Section 1, a major premise of our work is that in many practical contexts, estimating discounted values under $\gamma' \approx 1$ is difficult. As a result, directly evaluating the full gradient $\nabla_\theta V^{\pi_\theta}_{\gamma'}(x)$ is challenging, because it requires estimating Q-functions $Q^{\pi_\theta}_{\gamma'}(x,a)$. Below, we start by showing how the decomposition of $\nabla_\theta V^{\pi_\theta}_{\gamma'}(x)$ motivates a particular form of gradient update, which is generally considered a deep RL heuristic. Then we construct approximations to this update based on Taylor expansions.

4.1. $V^{\pi}_{\gamma'}$ as a weighted mixture of V^{π}_{γ}

We can explicitly rewrite $V^{\pi}_{\gamma'}(x)$ as a weighted mixture of value functions $V^{\pi}_{\gamma}(x'), x' \in \mathcal{X}$. This result was alluded to in (Romoff et al., 2019) and formally shown below.

Lemma 4.1. Assume $\gamma < \gamma' < 1$. We can write $V^{\pi}_{\gamma'}(x) = (\rho^{\pi}_{x,\gamma,\gamma'})^T V^{\pi}_{\gamma}$, where the weight vector $\rho^{\pi}_{x,\gamma,\gamma'} \in \mathbb{R}^{\mathcal{X}}$ is

$$\left(I - \gamma (P^\pi)^T\right)\left(I - \gamma' (P^\pi)^T\right)^{-1} \delta_x.$$

Also we can rewrite $V^{\pi}_{\gamma'}(x)$, using an expectation, as:

$$V^{\pi}_{\gamma}(x) + \mathbb{E}_\pi\left[\sum_{t=1}^{\infty} (\gamma' - \gamma)(\gamma')^{t-1} V^{\pi}_{\gamma}(x_t) \,\middle|\, x_0 = x\right]. \tag{16}$$

When $\gamma' = 1$, $\rho^{\pi}_{x,\gamma,\gamma'}$ might be undefined. However, Eqn (16) still holds if assumptions **A.1** and **A.2** are satisfied.

4.2. Decomposing the full gradient $\nabla_\theta V^{\pi_\theta}_{\gamma'}(x)$

Lemma 4.1 highlights that $V^{\pi}_{\gamma'}(x)$ depends on π in two aspects: **(1)** the value functions $V^{\pi}_{\gamma}(x'), x' \in \mathcal{X}$; **(2)** the state-dependent distribution $\rho^{\pi}_{x,\gamma,\gamma'}(x')$. Let π_θ be a parameterized policy. For conceptual clarity, we can write $V^{\pi_\theta}_{\gamma'}(x) = F(V^{\pi_\theta}_{\gamma}, \rho^{\pi_\theta}_{x,\gamma,\gamma'})$ with a function $F : \mathbb{R}^{\mathcal{X}} \times \mathbb{R}^{\mathcal{X}} \to \mathbb{R}$. Though this function is essentially the inner product, i.e., $F(V,\rho) = V^T\rho$, notationally, it helps stress that $V^{\pi_\theta}_{\gamma'}(x)$ depends on θ through two vector arguments. Now, we can decompose $\nabla_\theta V^{\pi_\theta}_{\gamma'}(x)$.

Lemma 4.2. The full gradient $\nabla_\theta V^{\pi_\theta}_{\gamma'}(x)$ can be decomposed into the sum of two partial gradients as follows,

$$\begin{aligned}&\left(\partial_V F(V,\rho)\right)^T \nabla_\theta V^{\pi_\theta}_{\gamma} + \left(\partial_\rho F(V,\rho)\right)^T \nabla_\theta \rho^{\pi_\theta}_{x,\gamma,\gamma'} \\ &= \underbrace{\mathbb{E}\left[\nabla_\theta V^{\pi_\theta}_{\gamma}(x')\right]}_{\text{first partial gradient}} + \underbrace{\mathbb{E}\left[V^{\pi_\theta}_{\gamma}(x') \nabla_\theta \log \rho^{\pi_\theta}_{x,\gamma,\gamma'}(x')\right]}_{\text{second partial gradient}},\end{aligned}$$

where the above partial gradients are both evaluated at $V = V^{\pi_\theta}_{\gamma}, \rho = \rho^{\pi_\theta}_{x,\gamma,\gamma'}$ and both expectations are with respect to $x' \sim \rho^{\pi_\theta}_{x,\gamma,\gamma'}$.

We argue that the second partial gradient introduces most challenges in practical optimization. Intuitively, this is because its unbiased estimator is equivalent to a REINFORCE gradient estimator which requires estimating discounted values that accumulate $V^{\pi}_{\gamma}(x')$ as 'reward' under discount factor γ'. By the premise of our work, this estimation would be difficult. We will detail the discussions in Appendix D.

The following result characterizes the first partial gradient.

Proposition 4.3. For any $\gamma < \gamma' < 1$, the first partial gradient $(\partial_V F(V^{\pi_\theta}_{\gamma}, \rho^{\pi_\theta}_{x,\gamma,\gamma'}))^T \nabla_\theta V^{\pi_\theta}_{\gamma}$ can be expressed as

$$\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} (\gamma')^t Q^{\pi_\theta}_{\gamma}(x_t, a_t) \nabla_\theta \log \pi_\theta(a_t|x_t) \,\middle|\, x_0 = x\right]. \tag{17}$$

When $\gamma' = 1$, under assumptions **A.1** and **A.2**, the first partial gradient exists and is expressed as

$$\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} Q^{\pi_\theta}_{\gamma}(x_t, a_t) \nabla_\theta \log \pi_\theta(a_t|x_t) \,\middle|\, x_0 = x\right]. \tag{18}$$

Connections to common deep RL heuristic. Many high-quality deep RL algorithms (see, e.g. Dhariwal et al., 2017; Achiam and OpenAI, 2018) implement parameter updates which are very similar to Eqn (18). As such, Proposition 4.3 provides some insights on why implementing such a heuristic might be useful in practice: though in general Eqn (18) is not a gradient (Nota and Thomas, 2019), it is a partial gradient of $V^{\pi_\theta}_{\gamma'=1}(x)$, which is usually the objective of interest at evaluation time. Compared with the formula of vanilla

PG in Eqn (2), Eqn (18) offsets the *over-discounting* by via a uniform average over states.

However, it is worth noting that in deep RL practice, the definition of the evaluation horizon T might slightly differ from that specified in **A.1**. In such cases, Proposition 4.3 does not hold. By **A.1**, T is the absorption time that defines when the MDP enters a terminal absorbing state. In many applications, however, for MDPs without a natural terminatal state, T is usually enforced by an external time constraint which does not depend on states. In other words, an environment can terminate even when it does not enter any terminal state (see, e.g., Brockman et al., 2016 for such examples). To bypass this subtle technical gap, one idea is to incorporate time steps as part of the state $\widetilde{x} \leftarrow [x, t]$. This technique was hinted at in early work such as (Schulman et al., 2015b) and empirically studied in (Pardo et al., 2018). In this case, the random absorbing time T depends fully on the augmented states, and Proposition 4.3 holds.

4.3. Taylor expansions of partial gradients

We now consider approximations to the first partial gradients

$$\left(\partial_V F(V_\gamma^{\pi_\theta}, \rho_{x,\gamma,\gamma'}^{\pi_\theta})\right)^T \nabla_\theta V_\gamma^{\pi_\theta} = (\rho_{x,\gamma,\gamma'}^{\pi_\theta})^T \nabla_\theta V_\gamma^{\pi_\theta}.$$

Since $\nabla_\theta V_\gamma^{\pi_\theta}$ does not depend on γ', the approximation is effectively with respect to the weight vector $\rho_{x,\gamma,\gamma'}^{\pi_\theta}$. Below, we show results for the K^{th} order approximation.

Proposition 4.4. Assume $\gamma < \gamma' < 1$. For any $x \in \mathcal{X}$, define the K^{th} Taylor expansion to $\rho_{x,\gamma,\gamma'}^{\pi}$ as

$$\rho_{x,K,\gamma,\gamma'}^{\pi} = \sum_{k=0}^{K} \left((\gamma' - \gamma)\left(I - \gamma(P^\pi)^T\right)^{-1}(P^\pi)^T\right)^k \delta_x.$$

It can be shown that $V_{K,\gamma,\gamma'}^{\pi}(x) = (\rho_{x,K,\gamma,\gamma'}^{\pi})^T V_\gamma^\pi$ and $\left\|\rho_{x,K,\gamma,\gamma'}^{\pi} - \rho_{K,\gamma,\gamma'}^{\pi}\right\|_\infty = O\left(\left(\frac{\gamma'-\gamma}{1-\gamma}\right)^{K+1}\right)$.

We build some intuitions about the approximations. Note that in general we can write the partial gradient as a weighted mixture of *local gradients* $Q_t \nabla_\theta \log \pi_\theta(a_t|x_t)$ where $Q_t := Q_\gamma^{\pi_\theta}(x_t, a_t)$,

$$\mathbb{E}_\pi\left[\sum_{t=0}^{\infty} w_{K,\gamma,\gamma'}(t) Q_t \nabla_\theta \log \pi_\theta(a_t|x_t) \,\middle|\, x_0 = x\right], \quad (19)$$

for some weight function $w_{K,\gamma,\gamma'}(t) \in \mathbb{R}$. When $K \to \infty$, $\lim w_{K,\gamma,\gamma'}(t) = (\gamma')^t$ and we recover the original first partial gradient defined in Eqn (17); when $K = 0$, $w_{K,\gamma,\gamma'}(t) = \gamma^t$ recovers the vanilla PG in Eqn (2). For other values of K, we show the analytic weights $w_{K,\gamma,\gamma'}(t)$ in Appendix D. Similar to how $V_{K,\gamma,\gamma'}^{\pi}$ interpolates V_γ^π and $V_{\gamma'}^\pi$, here the K^{th} order expansion to the partial gradients interpolate the full partial gradients and vanilla PG. In practice, we might expect an intermediate value of K achieve the best bias and variance trade-off of the update.

5. Policy optimization with Taylor expansions

Based on theoretical insights of previous sections, we propose two algorithmic changes to baseline algorithms. Based on Section 3, we propose Taylor expansion advantage estimation; based on Section 4, we propose Taylor expansion update weighting. It is important to note that other algorithmic changes are possible, which we leave to future work.

5.1. Baseline near on-policy algorithm

We briefly introduce backgrounds for near on-policy policy optimization algorithms (Schulman et al., 2015a; Mnih et al., 2016; Schulman et al., 2017; Espeholt et al., 2018). We assume that the data are collected under a behavior policy $(x_t, a_t, r_t)_{t=0}^{\infty} \sim \mu$, which is close to the target policy π_θ. The on-policyness is ensured by constraining $D(\pi_\theta, \mu) \le \varepsilon$ for some divergence D and threshold $\varepsilon > 0$. Usually, ε is chosen to be small such that little off-policy corrections are needed for estimating value functions. With data $(x_t, a_t, r_t)_{t=0}^{\infty}$, the algorithms estimate Q-functions $\widehat{Q}_\gamma^{\pi_\theta} \approx Q_\gamma^{\pi_\theta}$. Then the estimates $\widehat{Q}_\gamma^{\pi_\theta}(x, a)$ are used as plug-in alternatives to the Q-functions in the definition of gradient updates such as Eqn (2) for sample-based updates.

5.2. Taylor expansion Q-function estimation

In Section 3, we discussed how to construct approximations to $Q_{\gamma'}^{\pi_\theta}$ using $Q_\gamma^{\pi_\theta}$ as building blocks. As the first first algorithmic change, we propose to construct the K^{th} order expansion $Q_{K,\gamma,\gamma'}^{\pi_\theta}$ as a plug-in alternative to $Q_\gamma^{\pi_\theta}$ when combined with downstream optimization. Since $Q_{K,\gamma,\gamma'}^{\pi_\theta} \approx Q_{\gamma'}^{\pi_\theta}$, we expect the optimization subroutine to account for an objective of a longer effective horizon.

In many baseline algorithms, we have access to a value function critic $V_\phi(x)$ and a subroutine which produces Q-function estimates $\widehat{Q}_\gamma^{\pi_\theta}(x, a)$ (e.g., $\widehat{Q}_\gamma^{\pi_\theta}(x_t, a_t) = \sum_{s=0}^{\infty} \gamma^s r_{t+s}$). We then construct the K^{th} order expansion $\widehat{Q}_{K,\gamma,\gamma'}^{\pi_\theta}(x, a)$ using $\widehat{Q}_\gamma^{\pi_\theta}$. This procedure is similar to Algorithm 1 and we show the full algorithm in Appendix C. See also Appendix F for further experimental details.

5.3. Taylor expansion update weighting

In Section 4, we discussed Taylor expansions approximation $\rho_{x,K,\gamma,\gamma'}^{\pi_\theta}$ to the weight vector $\rho_{x,\gamma,\gamma'}^{\pi_\theta}$. As the second algorithmic change to the baseline algorithm, we update parameters in the direction of K^{th} order approximations to the partial gradient $\theta \leftarrow \theta + \alpha \left(\rho_{x,K,\gamma,\gamma'}^{\pi_\theta}\right)^T \nabla_\theta V_\gamma^{\pi_\theta}$. Eqn (19) shows that the update effectively translates into adjusting the weight $w_t = w_{K,\gamma,\gamma'}(t)$. When combined with other

Algorithm 2 Taylor expansion Q-function estimation

Require: policy π_θ with parameter θ and α
while not converged **do**
1. Collect partial trajectories $(x_t, a_t, r_t)_{t=1}^T \sim \mu$.
2. Estimate Q-functions $\widehat{Q}_\gamma^{\pi_\theta}(x_t, a_t)$.
3. Construct K^{th} order Taylor expansion estimator $\widehat{Q}_{K,\gamma,\gamma'}^{\pi_\theta}(x_t, a_t)$ using $\widehat{Q}^{\pi_\theta}(x_t, a_t)$.
4. Update the parameter via gradient ascent $\theta \leftarrow \theta + \alpha \sum_{t=1}^T \widehat{Q}_{K,\gamma,\gamma}^{\pi_\theta}(x_t) \nabla_\theta \log \pi_\theta(a_t|x_t)$.

end while

components of the algorithm, the pseudocode is shown in Algorithm 3. Under this framework, the common deep RL heuristic could be recovered by setting $w_t = 1$.

Algorithm 3 Taylor expansion update weighting

Require: policy π_θ with parameter θ and α
while not converged **do**
1. Collect partial trajectories $(x_t, a_t, r_t)_{t=1}^T \sim \mu$.
2. Estimate Q-functions $\widehat{Q}_t = \widehat{Q}_\gamma^{\pi_\theta}(x_t, a_t)$.
3. Compute weights for each state $w_t = w_{x_0,K,\gamma,\gamma'}(t)$, and average $g_\theta = \sum_{t=1}^T w_t \widehat{Q}_t \nabla_\theta \log \pi_\theta(a_t|x_t)$.
4. Update parameters $\theta \leftarrow \theta + \alpha g_\theta$.

end while

6. Related work

Discount factors in RL. Discount factors impact RL agents in various aspects. A number of work suggest that RL problems with large discount factors are generally more difficult to solve (Jiang et al., 2016), potentially due to increased complexities of the optimal value functions or collapses of the action gaps (Lehnert et al., 2018; Laroche and van Seijen, 2018). However, optimal policies defined with small discounts can be very sub-optimal for RL objectives with a large discount factor. To entail numerical stability of using large discounts, prior work has suggested non-linear transformation of the Bellman targets for Q-learning (Pohlen et al., 2018; van Hasselt et al., 2019; Kapturowski et al., 2018; Van Seijen et al., 2019). However, when data is scarce, small discount factors might prove useful due to its implicit regularization effect (Amit et al., 2020).

As such, there is a trade-off mediated by choosing different values of discount factors. Similar trade-off effects are most well-known in the context of TD(λ), where $\lambda \in [0, 1]$ trades-off the bias and variance of the TD updates (Sutton and Barto, 2018; Kearns and Singh, 2000).

Adapting discount factors & multiple discount factors. In general, when selecting a single optimal discount factor for training is difficult, it might be desirable to adjust the discount during training. This could be achieved by human-designed (Prokhorov and Wunsch, 1997; François-Lavet et al., 2015) or blackbox adaptation (Xu et al., 2018). Alternatively, it might also be beneficial to learn with multiple discount factors at the same time, which could improve TD-learning (Sutton, 1995) or representation learning (Fedus et al., 2019). Complementary to all such work, we study the connections between value functions defined with different discounts.

Taylor expansions for RL. Recently in (Tang et al., 2020), Taylor expansions were applied to study the relationship between V_γ^π and V_γ^μ, i.e., value functions under the same discount factor but different policies $\pi \neq \mu$. This is useful in the context of off-policy learning. Our work is orthogonal and could be potentially combined with this approach.

7. Experiments

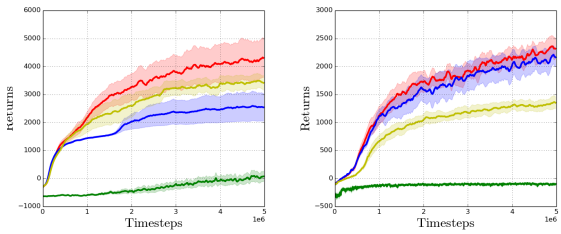
In this section, we evaluate the empirical performance of new algorithmic changes to the baseline algorithms. We focus on robotics control experiments with continuous state and action space. The tasks are available in OpenAI gym (Brockman et al., 2016), with backends such as MuJoCo (Todorov et al., 2012) and bullet physics (Coumans, 2015). We label the tasks as gym (G) and bullet (B) respectively. We always compare the undiscounted cumulative rewards evaluated under a default evaluation horizon $T = 1000$.

Hyper-parameters. Throughout the experiments, we use the same hyper-parameters across all algorithms. The learning rate is tuned for the baseline PPO, and fixed across all algorithms. See Appendix F for further details.

7.1. Taylor expansion Q-function estimation

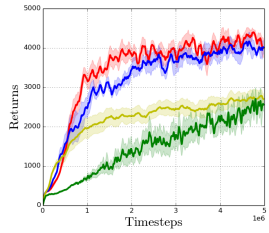
We use $\widehat{Q}_{K,\gamma,\gamma'}^{\pi_\theta}(x, a)$ with $K = 1$ as the Q-function estimator plug-in for the gradient update. When combining with PPO (Schulman et al., 2017), the resulting algorithm is named PPO(K). We compare with the baseline PPO and TRPO (Schulman et al., 2015a). In practice, we consider a mixture of advantage estimator $\widehat{Q}^{\pi_\theta}(x, a) = (1 - \eta)\widehat{Q}_\gamma^{\pi_\theta}(x, a) + \eta \widehat{Q}_{K,\gamma,\gamma'}^{\pi_\theta}(x, a)$ with $\eta \in [0, 1]$ a constant that interpolates between the PPO (i.e., $\eta = 0$) and PPO(K). Note that though η should be selected such that it balances the numerical scales of the two extremes, as a result, we usually find η to work well when it is small in absolute scale ($\eta = 0.01$ works the best).

Results. In Figure 2, we compare a few baselines: **(1)** PPO with $\gamma = 0.99$ (default); **(2)** PPO with high discount factor $\gamma = 1 - \frac{1}{T} = 0.999$; **(3)** PPO with Taylor expansion based advantage estimator, PPO(K). Throughout, we use a single hyper-parameter $\eta = 0.01$. We see that in general, PPO(K) leads to better performance (faster learning speed,

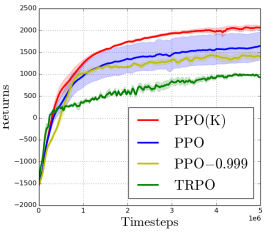


(a) HalfCheetah(G)

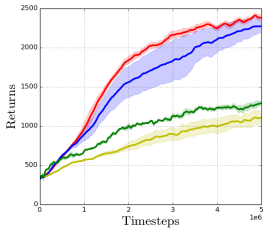
(b) Ant(G)



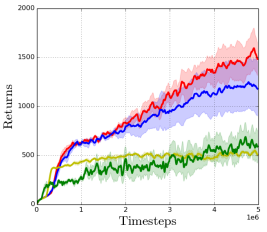
(c) Walker2d(G)



(d) HalfCheetah(B)

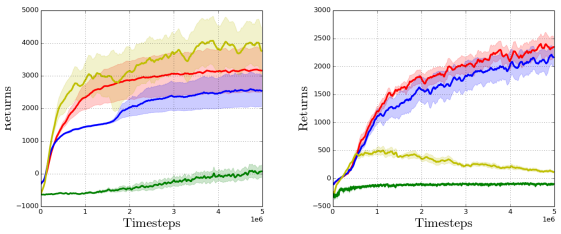


(e) Ant(B)



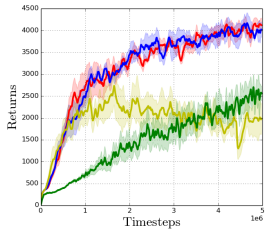
(f) Walker2d(B)

Figure 2. Comparison of Taylor expansion Q-function estimation with other baselines. Each curve shows median ± std results across 5 seeds. Taylor expansion outperforms PPO baselines with both lower and high discount factors.

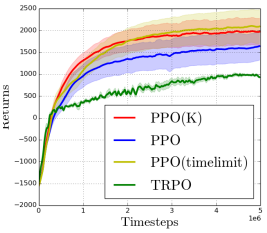


(a) HalfCheetah(G)

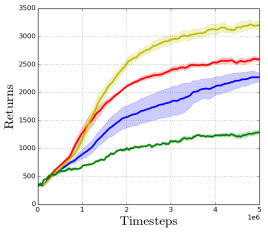
(b) Ant(G)



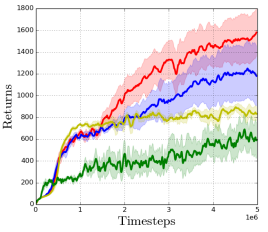
(c) Walker2d(G)



(d) HalfCheetah(B)



(e) Ant(B)



(f) Walker2d(B)

Figure 3. Comparison of Taylor expansion update weighting with other baselines. Each curve shows median ± std results across 5 seeds. Taylor expansion outperforms the default PPO baseline most stably.

better asymptotic performance or smaller variance across 5 seeds). This shows Taylor expansion Q-function estimation could lead to performance gains across tasks, given that the hyper-parameter η is carefully tuned. We provide a detailed ablation study on η in Appendix F, where we show how the overall performance across the benchmark tasks vary as η changes from small to large values.

A second observation is that simply increasing the discount factor to $\gamma = 1 - \frac{1}{T} = 0.999$ generally degrades the performance. This confirms issue with instability of directly applying high discount factors which motivates this work.

We also compare with the open source implementation of (Romoff et al., 2019) in Appendix F, where they estimate $\widehat{Q}^{\pi}_{\gamma'}$ based on recursive bootstraps of Q-function differences. Conceptually, this is similar to Taylor expansions with $K = \infty$. We show that without a careful trade-off mediated by smaller K, this algorithm does not improve performance out of the box.

7.2. Taylor expansion update weighting

As introduced in Section 5, we weigh local gradients $\widehat{Q}_t \nabla_\theta \log \pi_\theta(a_t|x_t)$ with K^{th} order expansion weights $w_{K,\gamma,\gamma'}(t)$. Here, we take $\gamma' = 1 - \frac{1}{T}$. Note that since $K = \infty$ corresponds to $\lim w_{K,\gamma,\gamma'}(t) = (\gamma')^t \approx 1$, this is very close to the commonly implemented PPO baseline. We hence expect the algorithm to work better with relatively large values of K and set $K = 100$ throughout experiments. In practice, we find the performance to be fairly robust in the choice of K. We provide further analysis and ablation study in Appendix F.

Results. We compare a few baselines: **(1)** default PPO; **(2)** PPO with time limit (Pardo et al., 2018). In this case, the states are augmented with time steps $\widetilde{x} \leftarrow [x, t]$ such that the augmented states $\widetilde{x}$ are Markovian; **(3)** PPO with Taylor expansion update weighting PPO(K). In Figure 3, we see that in general, PPO(K) and PPO with time limit outperform the baseline PPO. We speculate that the performance gains

arise from the following empirical motivation: since the evaluation stops at $t = T$, local gradients close to $t = T$ should be weighed down because they do not contribute as much to the final objective. However, the default PPO ignores such an effect and weighs all updates uniformly. To tackle this issue, PPO(K) explicitly weighs down the update while and PPO with time limit augments the state space to restore stationarity. Empirically, though in some cases PPO with time limit also outperforms PPO(K), it behaves fairly unstably in other cases.

Extensions to off-policy algorithms. Above, we mainly focused on on-policy algorithms. The setup is simpler because the data are collected (near) on-policy. It is possible to extend similar results to off-policy algorithms (Mnih et al., 2015; Lillicrap et al., 2015; Fujimoto et al., 2018; Haarnoja et al., 2018). Due to the space limit, we present extended results in Appendix F, where we show how to combine similar techniques to off-policy actor-critic algorithms such as TD3 (Fujimoto et al., 2018) and SAC (Haarnoja et al., 2018) in continuous control domains.

8. Conclusion

We have proposed a family of objectives that interpolate value functions defined with two discount factors. We have shown that similar techniques are applicable to other cumulative quantities defined through discounts, such as PG updates. This framework allowed us to achieve trade-off in estimating value functions or gradient updates, and led to empirical performance gains.

We also highlighted a new direction for bridging the gap between theory and practice: the gap between a fully discounted objective (in theory) and an undiscounted objective (in practice). By building a better understanding of this gap, we shed light on seemingly opaque heuristics which are necessary to achieve good empirical performance. We expect this framework to be useful for new practical algorithms.

Acknowledgements. Yunhao thanks Tadashi Kozuno and Shipra Agrawal for discussions on the discrepancy between policy gradient theory and practices. Yunhao acknowledges the support from Google Cloud Platform for computational resources.

References

Joshua Achiam and OpenAI. Spinning Up in Deep Reinforcement Learning. *https://github.com/openai/spinningup*, 2018.

Ron Amit, Ron Meir, and Kamil Ciosek. Discount factor as a regularizer in reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, 2020.

Richard Bellman. A Markovian decision process. *Journal of mathematics and mechanics*, pages 679–684, 1957.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI gym. *arXiv*, 2016.

Erwin Coumans. Bullet physics simulation. In *ACM SIGGRAPH 2015 Courses*, page 1. 2015.

Daniela Pucci De Farias and Benjamin Van Roy. The linear programming approach to approximate dynamic programming. *Operations research*, 51(6):850–865, 2003.

Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*, 2012.

Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. OpenAI baselines, 2017.

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. In *Proceeedings of the International Conference on Machine Learning*, 2018.

William Fedus, Carles Gelada, Yoshua Bengio, Marc G Bellemare, and Hugo Larochelle. Hyperbolic discounting and learning over multiple horizons. *arXiv*, 2019.

Roy Fox, Ari Pakman, and Naftali Tishby. Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2016.

Vincent François-Lavet, Raphael Fonteneau, and Damien Ernst. How to discount deep reinforcement learning: Towards new dynamic strategies. *NIPS Deep Reinforcement Learning Workshop*, 2015.

Scott Fujimoto, Herke Van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the International Conference on Machine Learning*, 2018.

Charles Miller Grinstead and James Laurie Snell. *Introduction to probability*. American Mathematical Soc., 2012.

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

Hado Hasselt. Double Q-learning. *Advances in Neural Information Processing Systems*, 2010.

Michael Janner, Igor Mordatch, and Sergey Levine. Gamma-models: Generative temporal difference learning for infinite-horizon prediction. *Advances in Neural Information Processing Systems*, 2020.

Nan Jiang, Satinder P Singh, and Ambuj Tewari. On structural properties of MDPs that bound loss due to shallow planning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2016.

Steven Kapturowski, Georg Ostrovski, John Quan, Remi Munos, and Will Dabney. Recurrent experience replay in distributed reinforcement learning. In *Proceedings of the International Conference on Learning Representations*, 2018.

Michael J Kearns and Satinder P Singh. Bias-variance error bounds for temporal difference updates. In *Proceedings of the Conference on Learning Theory*, 2000.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Romain Laroche and Harm van Seijen. In reinforcement learning, all objective functions are not equal. 2018.

Lucas Lehnert, Romain Laroche, and Harm van Seijen. On value function representation of long horizon problems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

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 *Proceedings of the International Conference on Learning Representations*, 2015.

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.

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 *Proceedings of the International Conference on Machine Learning*, 2016.

Chris Nota and Philip S Thomas. Is the policy gradient a gradient? In *Proceedings of the International Conference on Autonomous Agenst and Multiagent Systems*, 2019.

Brendan O'Donoghue, Remi Munos, Koray Kavukcuoglu, and Volodymyr Mnih. Combining policy gradient and Q-learning. In *Proceedings of the International Conference on Learning Representations*, 2016.

Fabio Pardo, Arash Tavakoli, Vitaly Levdik, and Petar Kormushev. Time limits in reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, 2018.

Tobias Pohlen, Bilal Piot, Todd Hester, Mohammad Gheshlaghi Azar, Dan Horgan, David Budden, Gabriel Barth-Maron, Hado Van Hasselt, John Quan, Mel Večerík, Matteo Hessel, Rémi Munos, and Olivier Pietquin. Observe and look further: Achieving consistent performance on atari. *arXiv*, 2018.

Danil V Prokhorov and Donald C Wunsch. Adaptive critic designs. *IEEE transactions on Neural Networks*, 8(5): 997–1007, 1997.

Martin L Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Joshua Romoff, Peter Henderson, Ahmed Touati, Emma Brunskill, Joelle Pineau, and Yann Ollivier. Separating value functions across time-scales. *Proceedings of the International Conference on Machine Learning*, 2019.

Sheldon M Ross. *Introduction to probability models*. Academic press, 2014.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *Proceedings of the International Conference on Learning Representations*, 2015.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the International Conference on Machine Learning*, 2015a.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

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

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic

policy gradient algorithms. In *Proceedings of the International Conference on Machine Learning*, 2014.

Samarth Sinha, Jiaming Song, Animesh Garg, and Stefano Ermon. Experience replay with likelihood-free importance weights. *arXiv*, 2020.

Richard S Sutton. TD models: Modeling the world at a mixture of time scales. In *Proceedings of the International Conference on Machine Learning*. 1995.

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

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*, 2000.

Yunhao Tang, Michal Valko, and Rémi Munos. Taylor expansion policy optimization. In *Proceedings of the International Conference on Machine Learning*, 2020.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems*, 2012.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Hado van Hasselt, John Quan, Matteo Hessel, Zhongwen Xu, Diana Borsa, and André Barreto. General non-linear Bellman equations. *arXiv*, 2019.

Harm Van Seijen, Mehdi Fatemi, and Arash Tavakoli. Using a logarithmic mapping to enable lower discount factors in reinforcement learning. In *Advances in Neural Information Processing Systems*, 2019.

Zhongwen Xu, Hado P van Hasselt, and David Silver. Meta-gradient reinforcement learning. *Advances in Neural Information Processing Systems*, 2018.

Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2008.