Average-Reward Reinforcement Learning with Trust Region Methods

Xiaoteng Ma[1] , **Xiaohang Tang**[2] , **Li Xia**[3*] , **Jun Yang**[1*] and **Qianchuan Zhao**[1]
[1]Department of Automation, Tsinghua University
[2]Department of Statistical Science, University College London
[3]Business School, Sun Yat-sen University
ma-xt17@mails.tsinghua.edu.cn, xiaohang.tang.20@ucl.ac.uk, xiali5@mail.sysu.edu.cn,
yangjun603@mail.tsinghua.edu.cn, zhaoqc@tsinghua.edu.cn,

Abstract

Most of reinforcement learning algorithms optimize the discounted criterion which is beneficial to accelerate the convergence and reduce the variance of estimates. Although the discounted criterion is appropriate for certain tasks such as financial related problems, many engineering problems treat future rewards equally and prefer a long-run average criterion. In this paper, we study the reinforcement learning problem with the long-run average criterion. Firstly, we develop a unified trust region theory with discounted and average criteria. With the average criterion, a novel performance bound within the trust region is derived with the Perturbation Analysis (PA) theory. Secondly, we propose a practical algorithm named Average Policy Optimization (APO), which improves the value estimation with a novel technique named Average Value Constraint. To the best of our knowledge, our work is the first one to study the trust region approach with the average criterion and it complements the framework of reinforcement learning beyond the discounted criterion. Finally, experiments are conducted in the continuous control environment MuJoCo. In most tasks, APO performs better than the discounted PPO, which demonstrates the effectiveness of our approach.

1 Introduction

The deep reinforcement learning (DRL) achieves prominent progress in many fields [Silver *et al.*, 2016; Vinyals *et al.*, 2019; Senior *et al.*, 2020]. Most of the modern RL algorithms aim at maximizing $\mathbb{E}\left[\sum_t^\infty \gamma^t r_t\right]$, where γ denotes a *discount factor* strictly less than 1. The discount factor undermines the contribution of future rewards on the current value and ignores the long-run gains [Sutton and Barto, 2018], which means that the reward received after k steps in the future is only worth γ^{k-1} of it to the current state. The existence of discount factor simplifies the theoretical analysis by constructing the contraction mapping and offers a faster and more stable online learning by reducing estimation variances [Kakade, 2001; Marbach and Tsitsiklis, 2003]. However, the discounted criterion is not aligned with the natural metric in many real scenarios, and introduces unnecessary bias of estimates despite lower variance [Thomas, 2014; Schulman *et al.*, 2016]. In empirical implementations, the discount factor serves as a hyperparameter and the algorithm performance is very sensitive to it [Duan *et al.*, 2016]. Thus, the decision maker should make a compromise between the foresight of the decision and the learning stability.

Different from the discounted criterion, the average criterion directly focuses on the long-run average performance. In the everlasting real-world problems, such as manufacturing, power systems and traffic controls, the average criterion is best suited [Mahadevan, 1996; Dewanto *et al.*, 2020]. Optimizing the average criterion completely removes the discrepancy between the training performance and the evaluating metric due to discounting. Therefore, the average criterion is well explored in the classical MDP literature [Howard, 1960; Puterman, 1994; Cao, 2007]. However, for the data-driven implementation, the average reward RL algorithms is much less investigated compared with the discounted criterion. It is relatively hard to study the average RL algorithms, since many properties are lost when setting $\gamma = 1$ in the discounted framework. An obvious evidence is that the Bellman operator is no longer a contraction mapping in the average setting [Bertsekas and Tsitsiklis, 1995].

One of the key difficulties in the average reward RL is to guarantee the values estimated from the sample paths are bounded without discounting. Many previous works extended the standard Q-learning to the average setting, such as R-learning [Schwartz, 1993], RVI-learning [Puterman, 1994; Bertsekas *et al.*, 1995], and CSV-learning [Yang *et al.*, 2016]. In the methods above, a relative value (estimated average reward, reference state value or a constant from prior knowledge) is subtracted from the target value, while they may still cause divergence and lead to unstable training [Dewanto *et al.*, 2020]. Compared with the value-iteration based ones, the policy-iteration based methods, such as policy gradient [Sutton *et al.*, 2000] and Actor-Critic [Konda and Tsitsiklis, 2000], are perhaps more suitable for average reward algorithms due to more stable training and potential for large scale problems. However, they still suffer the instability problem of value estimation in the average setting.

In this paper, we solve the average reward RL problem by

[*]Corresponding author: Li Xia, Jun Yang

introducing the trust region approach for the average criterion. Based on *Perturbation Analysis* (PA) theory, we overcome the analysis difficulty and develop a unified trust region theory with discounted and average criteria. We provide a novel performance bound which tightens the previous one [Schulman *et al.*, 2015; Achiam *et al.*, 2017] when $\gamma \to 1$. The bound provides a direct evidence that average criterion is preferred than the discounted one for ergodic MDPs. From the view of implementation, we provide a practical algorithm called Average Policy Optimization (APO). We address the difficulty of estimating value with the average criterion and provide a new method named Average Value Constraint to solve the value drifting problem. To the best of our knowledge, APO is the first average DRL algorithm with neutral networks as approximators. Finally, our experiments on the continuous control benchmark MuJoCo show that APO can beat the discounted PPO in the metric of average reward, which further confirms the superiority of our approach.

2 Preliminaries

2.1 Average Reward MDPs

Consider an infinite-horizon ergodic Markov decision process (MDP), defined by the tuple $\langle \mathcal{S}, \mathcal{A}, P, r, d_0 \rangle$, where $\mathcal{S}$ denotes a finite set of states, $\mathcal{A}$ denotes a finite set of actions, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ denotes the transition probability, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ denotes the reward function, $d_0 : \mathcal{S} \to [0,1]$ denotes the distribution of the initial state s_0. Let $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$ denote a stochastic policy, τ denote a trajectory $(s_0, a_0, s_1, \cdots)$, and $\tau \sim \pi$ denote that the distribution of trajectories depends on π: $s_0 \sim d_0, a_t \sim \pi(\cdot \mid s_t), s_{t+1} \sim P(\cdot \mid s_t, a_t)$.

With the assumption of ergodicity of MDPs, we aim at maximizing the long-run average performance under π:

$$\eta_\pi := \lim_{T\to\infty} \frac{1}{T} \mathbb{E}_{\tau\sim\pi} \left[\sum_{t=0}^{T-1} r(s_t, a_t) \right]. \tag{1}$$

We are interested in the *steady-state distribution* d_π, which is defined as follows:

$$d_\pi(s) := \lim_{T\to\infty} \frac{1}{T} \mathbb{E}_{\tau\sim\pi} \left[\sum_{t=0}^{T-1} P(s_t = s) \right]. \tag{2}$$

It is remarkable that as the effect of the current action in the infinite horizon is considered, the long-run performance is independent of the initial state distribution. Let $V_\pi(s)$ denote the *state-value function*, $Q_\pi(s,a)$ denote the *action-value function*, and $A_\pi(s,a)$ denote the *advantage function*. The value functions are defined by subtracting the long-run average reward η_π from the current reward to make the functions bounded [Howard, 1960]:

$$\begin{aligned} V_\pi(s) &:= \mathbb{E}_{\tau\sim\pi} \left[\sum_{t=0}^{\infty} (r(s_t, a_t) - \eta_\pi) \mid s_0 = s \right], \\ Q_\pi(s,a) &:= \mathbb{E}_{\tau\sim\pi} \left[\sum_{t=0}^{\infty} (r(s_t, a_t) - \eta_\pi) \mid s_0 = s, a_0 = a \right], \\ A_\pi(s,a) &:= Q_\pi(s,a) - V_\pi(s). \end{aligned} \tag{3}$$

2.2 Discounted Reward MDPs

With the discounted reward criterion, we focus on the recent rewards by introducing the discount factor $\gamma \in (0,1)$. We optimize the policy to maximize its normalized expected discounted return:

$$\eta_{\pi,\gamma} := (1-\gamma) \mathbb{E}_{\tau\sim\pi} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 \sim d_0 \right]. \tag{4}$$

The discounted performance $\eta_{\pi,\gamma}$ can be viewed as the average reward under *discounted steady-state distribution* $d_{\pi,\gamma}$:

$$d_{\pi,\gamma}(s) := (1-\gamma) \mathbb{E}_{\tau\sim\pi} \left[\sum_{t=0}^{\infty} \gamma^t P(s_t = s) \mid s_0 \sim d_0 \right]. \tag{5}$$

Different from the average criterion, the policy performance with the discounted criterion is dependent on d_0. The discount factor makes a smooth interpolation between d_0 and d_π. To see that, we have $\lim_{\gamma\to 0} d_{\pi,\gamma} = d_0$ (the bandit case) and $\lim_{\gamma\to 1} d_{\pi,\gamma} = d_\pi$ (the long-run average reward case).

In the modern DRL literature, the value functions are defined as following:

$$\begin{aligned} V_{\pi,\gamma}(s) &:= \mathbb{E}_{\tau\sim\pi} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s \right], \\ Q_{\pi,\gamma}(s,a) &:= \mathbb{E}_{\tau\sim\pi} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a \right], \\ A_{\pi,\gamma}(s,a) &:= Q_{\pi,\gamma}(s,a) - V_{\pi,\gamma}(s). \end{aligned} \tag{6}$$

As the value function is a core concept in RL, people may expect to optimize the long-run performance by letting $\gamma \to 1$. Unfortunately, many important properties are lost when $\gamma \to 1$, e.g., value functions will be unbounded which devastate the algorithm performance. Another example is the trust region theory for policy optimization, which we will discuss next.

2.3 Trust Region Method

The trust region methods [Schulman *et al.*, 2015; Schulman *et al.*, 2017; Achiam *et al.*, 2017] are a series of DRL algorithms updating policies with the following approximate policy iteration:

$$\begin{aligned} \pi_{k+1} = &\arg\max_\pi \mathbb{E}_{s\sim d_{\pi_k,\gamma}, a\sim\pi} \left[A_{\pi_k}(s,a) \right] \\ &\text{s.t. } \bar{D}(\pi \parallel \pi_k) \le \delta, \end{aligned} \tag{7}$$

where $\bar{D}(\pi \parallel \pi_k) = \mathbb{E}_{s\sim d_{\pi_k,\gamma}}[D(\pi \parallel \pi_k)[s]]$ with D denoting some distance measure and $\delta > 0$ is the step size. The policy set $\left\{\pi \mid \bar{D}(\pi \parallel \pi_k) \le \delta\right\}$ is called *trust region*, in which it is safe to evaluate policies with the samples collected with π_k. In theoretical analysis, D is often chosen as the *total variation divergence* defined by $D_{\text{TV}}(\pi' \parallel \pi)[s] = \frac{1}{2} \sum_{a\in\mathcal{A}} |\pi'(a \mid s) - \pi(a \mid s)|$. In practice, *Kullback-Leibler divergence* is preferred, which is defined as $D_{\text{KL}}(\pi' \parallel \pi)[s] = \sum_{a\in\mathcal{A}} \pi'(a \mid s) \log \frac{\pi'(a|s)}{\pi(a|s)}$. Since $D_{\text{TV}}(p \parallel q) \le \sqrt{D_{\text{KL}}(p \parallel q)/2}$, the theoretical analysis is aligned with the practical implementation.

The primary motivation of the trust region method is that optimizing the surrogate objective in the trust region guarantees monotonic performance improvements. When the distance is bounded by D_{TV}, the solution of the problem in (7) has the lower performance bound [Achiam *et al.*, 2017]

$$\eta_{\pi_{k+1},\gamma} - \eta_{\pi_k,\gamma} \geq -\frac{2\gamma\epsilon_\gamma}{1-\gamma}\mathbb{E}_{d_{\pi_k,\gamma}}[D_{\text{TV}}(\pi_{k+1} \parallel \pi_k)[s]],$$

where $\epsilon_\gamma = \max_s |\mathbb{E}_{a\sim\pi_{k+1}}[A_{\pi_k,\gamma}(s,a)]|$.

The advantage of the trust region method majorly comes from two aspects. In each iteration, the samples collected with π_k are reused in evaluating any π in trust region, which significantly improves the data efficiency compared with the traditional policy gradient methods. The other hand, the performance lower bound avoids collapse after bad updates when neural networks are used as the policy approximator [Duan *et al.*, 2016].

However, existing analysis of the lower performance bound for the trust region method is not applicable when $\gamma \to 1$, which is easy to be verified by $\lim_{\gamma\to 1}\gamma/(1-\gamma) = \infty$. As the larger discounted factor and average reward setting are important in real problems, more delicate theoretical analysis is needed to understand the behavior of trust region method when $\gamma \to 1$ even $\gamma = 1$.

3 Unified Trust Region Theory with Discounted and Average Criteria

In this section, we propose a unified trust region theory based on the PA theory that extends the existing analysis of performance improvement bound from discounted criterion to average case. Moreover, we provide a novel policy performance bound for police updating. With the help of this bound, performance monotonicity of policy improvement is guaranteed when $\gamma \to 1$. Typically, our result shows monotonic policy improvement is also guaranteed with the average reward criterion, simply by setting $\gamma = 1$.

3.1 General Formulation

We redefine the discounted value functions in (6) as follows:

$$\begin{aligned}
&V_{\pi,\gamma}(s) := \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t\left(r(s_t,a_t)-\eta_\pi\right) \mid s_0 = s\right],\\
&Q_{\pi,\gamma}(s,a) := \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t\left(r(s_t,a_t)-\eta_\pi\right) \mid s_0 = s, a_0 = a\right],\\
&A_{\pi,\gamma}(s,a) := Q_{\pi,\gamma}(s,a) - V_{\pi,\gamma}(s). \qquad (8)
\end{aligned}$$

Although this formulaic approach may be not familiar to DRL researchers, it is commonly used in the MDP literature [Cao, 2007]. The main benefit of this value definition is that it bridges the gap between the discounted and average criteria. With the new definition of value functions, we obtain that $\lim_{\gamma\to 1}V_{\pi,\gamma} = V_\pi$ immediately.

Next, we present the *performance difference formula* for any policies π, π'.

Lemma 1. *For any policies π, π', the following equation holds:*

$$\eta_{\pi',\gamma} - \eta_{\pi,\gamma} = \mathbb{E}_{s\sim d_{\pi',\gamma},a\sim\pi'}\left[A_{\pi,\gamma}(s,a)\right]. \qquad (9)$$

This relationship has been shown in many previous studies of trust region method [Kakade and Langford, 2002; Schulman *et al.*, 2015; Achiam *et al.*, 2017]. Actually, (9) can be viewed as a rewriting of the performance difference formula in a sample path form. The performance difference formula is the key result of PA theory [Cao, 2007] and it quantifies the change of the performance of Markov systems corresponding to the change of policies. For the readers unfamiliar with PA, we attach a brief introduction in Appendix A.1.

Powered by the performance difference formula, we derive the average performance difference formula at once.

Corollary 1. *For any policies π, π', the following equation holds:*

$$\eta_{\pi'} - \eta_\pi = \mathbb{E}_{s\sim d_{\pi'},a\sim\pi'}\left[A_\pi(s,a)\right]. \qquad (10)$$

3.2 Policy Improvement Bound

While the performance difference formula accurately describes the performance difference for two arbitrary policies, it is difficult to use the formula to develop practical algorithms directly. The bottleneck is that to evaluate the performance difference, we need the samples from $d_{\pi',\gamma}$. That is self-contradictory since no trajectory of the new policy is available unless the policy is updated. Thus, instead of optimizing the difference formula directly, the trust region method optimizes the following surrogate objective:

$$L_{\pi,\gamma}(\pi') := \mathbb{E}_{s\sim d_{\pi,\gamma},a\sim\pi}\left[\frac{\pi'(a\mid s)}{\pi(a\mid s)}A_{\pi,\gamma}(s,a)\right], \qquad (11)$$

where $d_{\pi',\gamma}$ is replaced by $d_{\pi,\gamma}$ in (9) and the probability of the sampled actions is corrected by importance sampling.

It is natural to query what is the difference between the surrogate objective and the real performance difference. To answer this question, we present the bound as follows.

Proposition 1. *For any two stochastic policies π, π', the following bound holds:*

$$\begin{aligned}
&\eta_{\pi',\gamma} - \eta_{\pi,\gamma} \geq L_{\pi,\gamma}(\pi') - 2\epsilon_\gamma D_{TV}(d_{\pi',\gamma} \parallel d_{\pi,\gamma}),\\
&\eta_{\pi',\gamma} - \eta_{\pi,\gamma} \leq L_{\pi,\gamma}(\pi') + 2\epsilon_\gamma D_{TV}(d_{\pi',\gamma} \parallel d_{\pi,\gamma}), \qquad (12)
\end{aligned}$$

where $\epsilon_\gamma = \max_s |\mathbb{E}_{a\sim\pi'}[A_{\pi,\gamma}(s,a)]|$.

The result tells us that if the discounted steady-state distribution of two policies is close enough and the advantage function is bounded, we are safe to use the surrogate objective for policy updating.

Next, we build up the connection between the distance of the discounted steady-state distributions and the distance of policies.

Proposition 2. *For any two stochastic policies π, π', the following bound holds:*

$$D_{TV}(d_{\pi',\gamma} \parallel d_{\pi,\gamma}) \leq \xi_\gamma \mathbb{E}_{s\sim d_{\pi,\gamma}}[D_{TV}(\pi' \parallel \pi)[s]], \qquad (13)$$

where $\xi_\gamma = \min\left\{\frac{\gamma}{1-\gamma}, \left(\frac{\gamma(\kappa_{\pi'}-1)}{1-(1-\gamma)\kappa_{\pi'}}\right)_+\right\}$, $(\cdot)_+ = \max(\cdot, 0)$, and $\kappa_{\pi'}$ is the Kemeny's constant of the Markov chain induced by π'.

The above conclusion is the key contribution of this paper, which reveals that *the difference of discounted steady-state distribution is bounded with the distance of the policies multiplied with a constant factor* ξ_γ. This factor ξ_γ is a time constant that reflects how policy changes affect long-term behaviors of Markov chains. When γ is not very large, the long-run effect of the current decision gradually decreases, allowing decision makers to focus on a few steps after the decision and ignore the impact afterwards. However, if γ is large enough, we cannot use discounts to suppress the long-term impact of the policy anymore. Fortunately, when letting γ approach 1, we find that it is still able to bound the long-run impact by introducing *Kemeny's constant* [Kemeny and Snell, 1960], which means the average time of returning to the steady-state distribution starting from an arbitrary state.

We further explore the theoretical result of ξ_γ by observing following phenomena.

Remark 1. *For the average reward criterion, we have* $\xi = \lim_{\gamma\to 1}\xi_\gamma = \kappa_{\pi'} - 1$.

Remark 2. *The largest* ξ_γ *is not taken at* $\gamma = 1$. *In particular,*

$$\arg\max_{\gamma} \xi_\gamma = 1 - \frac{3}{2\kappa_{\pi'}+1}. \tag{14}$$

We should not intuitively believe that a smaller γ always leads to a tighter performance bound. When γ is larger than (14), the lower bound decreases as the γ increases and finally becomes $\kappa_{\pi'} - 1$ at $\gamma = 1$. It supports that when the problem focuses more on long-term performance, the average criterion is better than a large discount factor in the trust region method, since the former is able to obtain a more accurate estimation.

Combining the preceding two bounds together, we conclude the following theorem.

Theorem 1. *For any two stochastic policies* π, π', *the following bound holds:*

$$\eta_{\pi',\gamma} - \eta_{\pi,\gamma} \geq L_{\pi,\gamma}(\pi') - 2\epsilon_\gamma\xi_\gamma\mathbb{E}_{s\sim d_{\pi,\gamma}}[D_{TV}(\pi' \parallel \pi)[s]],$$
$$\eta_{\pi',\gamma} - \eta_{\pi,\gamma} \leq L_{\pi,\gamma}(\pi') + 2\epsilon_\gamma\xi_\gamma\mathbb{E}_{s\sim d_{\pi,\gamma}}[D_{TV}(\pi' \parallel \pi)[s]].$$

In particular, the bounds hold with the average criterion:

$$\eta_{\pi'} - \eta_{\pi} \geq L_{\pi}(\pi') - 2\epsilon\xi\mathbb{E}_{s\sim d_{\pi}}[D_{TV}(\pi' \parallel \pi)[s]],$$
$$\eta_{\pi'} - \eta_{\pi} \leq L_{\pi}(\pi') + 2\epsilon\xi\mathbb{E}_{s\sim d_{\pi}}[D_{TV}(\pi' \parallel \pi)[s]].$$

The result clearly shows that the performance difference is bounded by three components: *the difference of policies, the maximum change in the values, and a time constant depending on the problem.* It not only facilitates the design of practical trust region algorithms, but also inspires us to further understand the nature of general RL algorithms.

In the end of this section, we give another perspective for why the average criterion is preferred to the discounted one in trust region method. It should be emphasized that the above analysis is based on the discounted steady-state distribution, from which it is not convenient to sample. In the practical implementation, most discounted algorithms ignore the difference in state distributions and directly sample from the steady-state distribution d_π, which leads to biased estimates for the policy gradients [Thomas, 2014]. However, with the average criterion, performance estimation is unbiased as sampling under d_π is theoretically-justified.

4 Average Policy Optimization

Based on the unified trust region theory, we extend the current discounted trust region algorithms to the average reward criterion. We address the value drifting problem raised in the average setting, and present a technique named Average Value Constraint for better value estimation. By approximating the policy and value function with neutral networks, we develop an algorithm based on the Actor-Critic framework, which is named Average Policy Optimization.

4.1 Value Estimation

In the most DRL algorithms, the value function is approximated by a value network $V_\phi(s)$ with ϕ as the parameters, which is updated by minimizing the following loss with stochastic gradient descent (SGD) method

$$\min_\phi J_V(\phi) = \frac{1}{N}\sum_{n=1}^{N}\left[\frac{1}{2}(\hat{V}(s_n) - V_\phi(s_n))^2\right], \tag{15}$$

where n indexes a batch of states sampled with π and $\hat{V}(s_n)$ is the target value for s_n.

Under the discounted criterion, $\hat{V}(s_n)$ can be calculated directly using Monte Carlo (MC) method with $\hat{V}_{\text{MC}}(s_n) = \sum_{t=0}^{\infty}\gamma^t r_{n+t}$, or using the value of next step to bootstrap (TD): $\hat{V}_\phi(s_n) = r_n + \gamma V_\phi(s_{n+1})$ [Sutton and Barto, 2018], where the latter is preferred in most cases for low-variance estimates. In the average setting, we adopt the update rule of R-learning [Schwartz, 1993] to evaluate the target value with $\hat{V}_\phi(s_n) = r_n - \hat{\eta} + V_\phi(s_{n+1})$, where $\hat{\eta}$ is the estimate of η_π. To smooth the value of $\hat{\eta}$ over batches, we update $\hat{\eta}$ by a moving average method: $\hat{\eta} \leftarrow (1-\alpha)\hat{\eta} + \alpha\frac{1}{N}\sum_{n=1}^{N} r_n$, where α is the step size.

Let ϕ' denote the updated parameters:

$$\phi' = \phi + \beta(\hat{V}_\phi(s) - V_\phi(s))\nabla V_\phi(s), \tag{16}$$

where β is the learning rate. As the value network is updated by SGD, noise in the estimate is inevitable. Suppose that there is a bias ϵ_V between the value function and the true value $V_\phi(s) = V_{\tilde{\phi}}(s) + \epsilon_V, \forall s \in \mathcal{S}$. The real updated parameters based on the $V_{\tilde{\phi}}$ is

$$\tilde{\phi}' = \phi + \beta(\hat{V}_{\tilde{\phi}}(s) - V_{\tilde{\phi}}(s))\nabla V_{\tilde{\phi}}(s). \tag{17}$$

As β is sufficiently small, the post-update value function can be well-approximated by linearizing around ϕ using Taylor's expansion:

$$V_{\phi'}(s) \approx V_\phi(s) + \beta(\hat{V}_\phi(s) - V_\phi(s))\|\nabla V_\phi(s)\|_2^2,$$
$$V_{\tilde{\phi}'}(s) \approx V_{\tilde{\phi}}(s) + \beta(\hat{V}_{\tilde{\phi}}(s) - V_{\tilde{\phi}}(s))\|\nabla V_{\tilde{\phi}}(s)\|_2^2.$$

Subtracting the first equation from the second one, we have

$$V_{\phi'}(s) - V_{\tilde{\phi}'}(s) \approx \left(1 - \beta(1-\gamma)\|\nabla_\phi V_\phi(s)\|_2^2\right)\epsilon_V. \tag{18}$$

For $\gamma < 1$, the bias between the real values and approximated values is gradually eliminated during training. However, this feedback mechanism disappears for $\gamma = 1$ as $V_{\phi'}(s) - V_{\tilde{\phi}'}(s) \approx \epsilon_V$. That means the approximated values are drifting away from the true values during training due to noises. We call it the *value drifting problem* in the average reward setting.

The analysis above explains why the value estimation in average setting is inaccurate in the data-driven mode. Next we try to correct the estimation by taking advantage of the following property.

Proposition 3. $\mathbb{E}_{s\sim d_\pi}\left[V_{\pi,\gamma}(s)\right]=0.$

It tells us that the ideal estimates of values defined by (8) should have zero mean. We rewrite the problem in (15) by adding this condition as a constraint:

$$\min_\phi J_V(\phi)=\frac{1}{N}\sum_{n=1}^{N}\left[\frac{1}{2}(\hat{V}(s_n)-V_\phi(s_n))^2\right],$$
$$\text{s.t. } \frac{1}{N}\sum_{n=1}^{N}\left[V_\phi(s_n)\right]=0. \tag{19}$$

By constructing the Lagrangian function for the above problem, we have

$$\min_\phi L_V(\phi,\nu)=\frac{1}{N}\sum_{n=1}^{N}\left[\frac{1}{2}(\hat{V}(s_n)-V_\phi(s_n))^2\right]-\frac{1}{2}\nu b^2.$$

where $b:=\frac{1}{N}\sum_{n=1}^{N}\left[V_\phi(s_n)\right]$ and ν is the Lagrangian multiplier. Taking the gradient of $L_V(\phi,\nu)$, we obtain

$$\begin{aligned}&\nabla L_V(\phi,\nu)\\&=\frac{1}{N}\sum_{n=1}^{N}\left[(\hat{V}(s_n)-V_\phi(s_n))\nabla_\phi V_\phi(s_n)\right]-\frac{\nu b}{N}\sum_{n=1}^{N}\nabla_\phi[V_\phi(s_n)]\\&=\frac{1}{N}\sum_{n=1}^{N}\left[(\hat{V}(s_n)-\nu b-V_\phi(s_n))\nabla_\phi V_\phi(s_n)\right].\end{aligned}$$

The problem in (19) is equivalent to the following problem

$$\min_\phi J_V(\phi)=\frac{1}{N}\sum_{n=1}^{N}\left[\frac{1}{2}(\tilde{V}(s_n)-V_\phi(s_n))^2\right], \tag{20}$$

where $\tilde{V}(s_n):=\hat{V}(s_n)-\nu b$. Compared with the discounted variant, we explicitly control the bias in value estimation. We name this method as Average Value Constraint.

4.2 Policy Optimization

Many algorithms [Schulman *et al.*, 2015; Schulman *et al.*, 2017; Wu *et al.*, 2017] have been proposed to approximately solve the discounted problem in (7) with a policy network. Here we choose PPO [Schulman *et al.*, 2017] as the base to develop our average reward algorithm, and point out that the policy updating is similar with other trust region methods. Instead of optimizing the original objective with the constraint, we optimize the surrogate loss as follows:

$$J_\pi(\theta)=\frac{1}{N}\sum_{n=1}^{N}\left[\min\left(\omega(\theta)\hat{A}_n,\text{clip}(\omega(\theta),1-\varepsilon,1+\varepsilon)\hat{A}_n\right)\right], \tag{21}$$

where θ is the parameters of the policy network, $\omega(\theta)=\frac{\pi_\theta(a_t|s_t)}{\pi(a_t|s_t)}$ is the importance sampling ratio and $\hat{A}_n$ is a shorthand of advantage estimation $\hat{A}(s_n,a_n)$. To better balance the variance and bias of policy gradients, we further modify the discounted GAE [Schulman *et al.*, 2016] to an average variant. Define the average TD residual as

$$\delta_t=r(s_t,a_t)-\hat{\eta}+V_\phi(s_{t+1})-V_\phi(s_t). \tag{22}$$

Then the advantage estimator is

$$\hat{A}(s_n,a_n)=\sum_{t=0}^{\infty}\lambda^t\delta_{n+t}, \tag{23}$$

where λ is the parameter to compromise between bias and variance.

Algorithm 1 Average Policy Optimization

```
Input: $\alpha, \beta, \lambda, \nu, \varepsilon, K, N, M$
 1: Initialize $\theta, \phi$ randomly.
 2: Set $\hat{\eta}=0$, $b=0$.
 3: for $k=1,2,\cdots,K$ do
 4:     Run policy $\pi_{\theta_k}$ for $N$ times to collect $\{s_n, a_n, r(s_n,a_n), s_{n+1}\}, n=1,\ldots,N$.
 5:     Update $\hat{\eta} \leftarrow (1-\alpha)\hat{\eta}+\alpha\frac{1}{N}\sum_{n=1}^{N} r(s_n,a_n)$.
 6:     Update $b \leftarrow (1-\alpha)b+\alpha\frac{1}{N}\sum_{n=1}^{N} V_\phi(s_n)$.
 7:     Compute $\delta_n$ with (22) at all timesteps.
 8:     Compute $\hat{A}(s_n,a_n)$ with (23) at all timesteps.
 9:     Update the $\theta$ with $J_\pi(\theta)$ in (21) for $M$ epochs.
10:     Update the $\phi$ with $J_V(\phi)$ in (20) for $M$ epochs.
11: end for
```

5 Experiments

We conducted a series of experiments to evaluate APO, which are used to answer two questions:

- Compared with the discounted PPO, does APO have better performance in environments suitable with average criterion?
- What are the factors that affect the performance of APO? Specifically, does APO benefit from the Average Value Constraint?

We choose the continuous control benchmark MuJoCo [Todorov *et al.*, 2012] with the OpenAI Gym [Brockman *et al.*, 2016]. The MuJoCo tasks are designed to control the robot to finish certain tasks such as running as fast as possible without falling. It is natural to use the average criterion for some tasks in MuJoCo, such as HalfCheetch in which the robot keeps running and is rewarded by its speed. Moveover, we also evaluate APO in the other experiments with terminal states to examine the generalization ability of APO beyond the theoretical assumptions.

Both APO and PPO are implemented based on a modular RL package in PyTorch named *rlpyt* [Stooke and Abbeel, 2019]. While the performance of PPO is largely dependent on the code-optimizations [Andrychowicz *et al.*, 2020], we do not consider any tricks apart from the vanilla PPO for controlling variables to justify true impact factors. For each task, we run the algorithm with 5 random seeds for 3 million steps and do the evaluation every 2000 steps. In the evaluation, we run 10 episodes without exploration by setting the standard deviation of policy as zero. All the hyperparameter combinations we

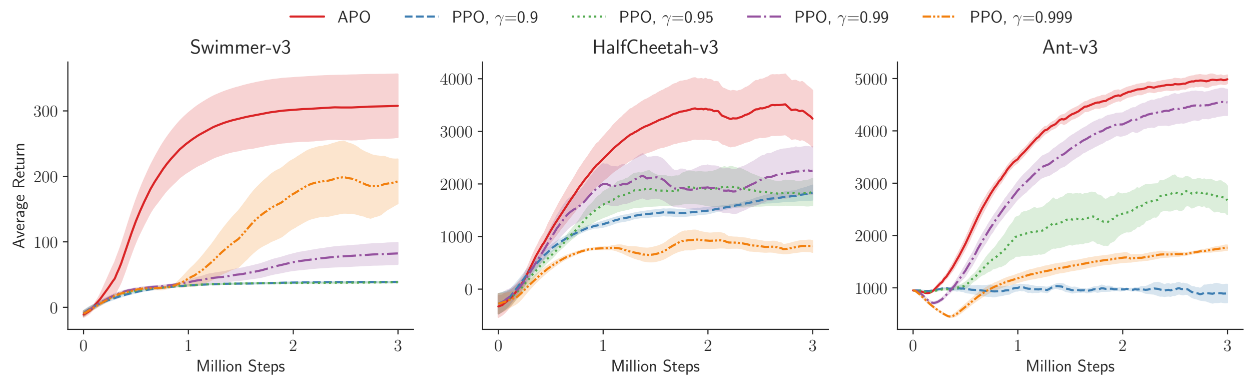

Figure 1: Compare APO and PPO with different discount factors in MuJoCo tasks. Results for more tasks can be found in Appendix C.

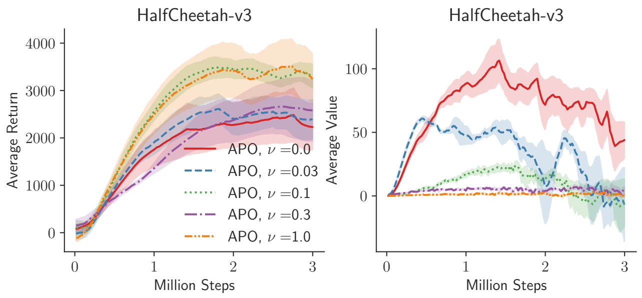

Figure 2: Evaluation for different levels ν. On the left and the right are the average return and average value curves respectively.

consider are grid searched, which are showed in Appendix B. The computing infrastructure for running experiments is a server with 2 AMD EPYC 7702 64-Core Processor CPUs and 8 Nvidia GeForce RTX 2080 Ti GPUs. The time consumed of each experiment varies from 1.2h to 2h on average according to the complexity of the task.

5.1 Comparison to Discounted PPO

We compare our APO with the standard discounted PPO with the different discounted factors, and select the combination of hyperparameters which achieves the best average performance. The results in Figure 1 show that APO surpasses all the variants of PPO with different discount factors. We observe that the performances of PPO vary drastically with the change of γ. Meanwhile, The relationship between performance and γ variation is not monotonic. For example, $\gamma = 0.99$ achieves a sound result in Ant, while the performance with $\gamma = 0.999$ is terrible. This confirms that the performance with the average criterion cannot be optimized by letting γ approach 1 in the discounted framework.

The environments shown in Figure 1 are suitable for average setting, as they have no terminal states (Swimmer and HalfCheetah) or hard to fall even with a random policy (Ant). It is natural that APO beats the discounted PPO in these tasks. The audience may be curious about the performances of APO in other tasks, such as Hopper, Walker2d and Humanoid. We also evaluate APO in them and show the results in Appendix C. As we expected, APO achieves sound results with the competent PPO but does not beat the best in Hopper and Walker. However, when we change the metric from average episode return to average reward, we find that APO is better than discounted PPO, especially in Humanoid. It shows that APO achieves higher speed but ignores the terminal reward as it focuses on the average performance under the steady-state distribution. We believe it is more suitable to model the safety requirements as constraints, which is easy to be extended from APO [Achiam *et al.*, 2017].

5.2 Ablation Study on Average Value Constraint

We further evaluate the proposed method Average Value Constraint for the value estimation in average setting. We fix the penalty coefficient ν at different levels to see the changes of average values during training. The performances are improved by 66.14% with the Average Value Constraint. As shown in Figure 2, a larger ν produces a tighter constraint on the average value. Without the average value constraint ($\nu = 0$), the average values fluctuate dramatically during training. Since only the relative value of different states is meaningful for the decision, the bias of average value is detrimental to the policy gradient estimation. As the average value constraint is derived from the Lagrangian method, we can also optimize ν adaptively during training [Stooke *et al.*, 2020].

6 Conclusion

We pioneer the study of the trust region approach with the average-reward RL. Based on the PA theory, we first prove a lower performance bound the average criterion and tighten the previous discounted bound with a large discount factor. With the novel lower bound, the monotonic policy improvement of the average trust region method is guaranteed. Furthermore, we analyse the value estimation in the average setting and propose a new technique Average Value Constraint to stabilize the value estimation during data training. In empirical experiments, our proposed algorithm APO achieves better performance over most PPO variants with different discount factors in MuJoCo, which demonstrates the superiority of the average criterion in many engineering problems. In summary, this work fills the gap in the theory of DRL with the average criterion and provides a practical algorithm with a more appropriate criterion for many practical problems beyond the current benchmarks with the discounted criterion.

References

[Achiam *et al.*, 2017] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International Conference on Machine Learning*, pages 22–31, 2017.

[Andrychowicz *et al.*, 2020] Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphael Marinier, Léonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, et al. What matters in on-policy reinforcement learning? a large-scale empirical study. *arXiv preprint arXiv:2006.05990*, 2020.

[Bertsekas and Tsitsiklis, 1995] Dimitri P Bertsekas and John N Tsitsiklis. Neuro-dynamic programming: an overview. In *IEEE Conference on Decision and Control*, pages 560–564. IEEE, 1995.

[Bertsekas *et al.*, 1995] Dimitri P Bertsekas, Dimitri P Bertsekas, Dimitri P Bertsekas, and Dimitri P Bertsekas. *Dynamic programming and optimal control, volume II*. Athena scientific Belmont, 1995.

[Brockman *et al.*, 2016] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv preprint arXiv:1606.01540*, 2016.

[Cao, 2007] Xiren Cao. *Stochastic Learning and Optimization: A Sensitivity-Based Approach*. Springer, 2007.

[Dewanto *et al.*, 2020] Vektor Dewanto, George Dunn, Ali Eshragh, Marcus Gallagher, and Fred Roosta. Average-reward model-free reinforcement learning: a systematic review and literature mapping. *arXiv preprint arXiv:2010.08920*, 2020.

[Duan *et al.*, 2016] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

[Howard, 1960] Ronald A. Howard. Dynamic programming and markov processes. *Mathematical Gazette*, 3(358):120, 1960.

[Kakade and Langford, 2002] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, volume 2, pages 267–274, 2002.

[Kakade, 2001] Sham Kakade. Optimizing average reward using discounted rewards. In *International Conference on Computational Learning Theory*, pages 605–615. Springer, 2001.

[Kemeny and Snell, 1960] John G Kemeny and J Laurie Snell. Finite markov chains. *Van Nostrand, New Jersey*, 1960.

[Konda and Tsitsiklis, 2000] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014, 2000.

[Mahadevan, 1996] Sridhar Mahadevan. Average reward reinforcement learning: foundations, algorithms, and empirical results. *Machine Learning*, 22(1):159–195, 1996.

[Marbach and Tsitsiklis, 2003] Peter Marbach and John N Tsitsiklis. Approximate gradient methods in policy-space optimization of markov reward processes. *Discrete Event Dynamic Systems*, 13(1-2):111–148, 2003.

[Puterman, 1994] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994.

[Schulman *et al.*, 2015] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

[Schulman *et al.*, 2016] John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

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

[Schwartz, 1993] Anton Schwartz. A reinforcement learning method for maximizing undiscounted rewards. In *International Conference on Machine Learning*, pages 298–305, 1993.

[Senior *et al.*, 2020] Andrew W Senior, Richard Evans, John Jumper, James Kirkpatrick, Laurent Sifre, Tim Green, Chongli Qin, Augustin Žídek, Alexander WR Nelson, Alex Bridgland, et al. Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792):706–710, 2020.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[Stooke and Abbeel, 2019] Adam Stooke and Pieter Abbeel. rlpyt: A research code base for deep reinforcement learning in pytorch. *arXiv preprint arXiv:1909.01500*, 2019.

[Stooke *et al.*, 2020] Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by PID lagrangian methods. In *International Conference on Machine Learning*, pages 9133–9143. PMLR, 2020.

[Sutton and Barto, 2018] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

[Sutton *et al.*, 2000] 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.

[Thomas, 2014] Philip Thomas. Bias in natural actor-critic algorithms. In *International conference on machine learning*, pages 441–448, 2014.

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

[Vinyals *et al.*, 2019] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[Wu *et al.*, 2017] Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In *Advances in neural information processing systems*, pages 5279–5288, 2017.

[Yang *et al.*, 2016] Shangdong Yang, Yang Gao, Bo An, Hao Wang, and Xingguo Chen. Efficient average reward reinforcement learning using constant shifting values. In *AAAI Conference on Artificial Intelligence*, pages 2258–2264, 2016.