Variance-Reduced Off-Policy Memory-Efficient Policy Search

Daoming Lyu
Auburn University
`daoming.lyu@auburn.edu`

Qi Qi
University of Iowa
`qi-qi@uiowa.edu`

Mohammad Ghavamzadeh
Google Research
`mohammad.ghavamzadeh51@gmail.com`

Hengshuai Yao
Huawei Inc.
`hengshuai.yao@huawei.com`

Tianbao Yang
University of Iowa
`tianbao-yang@uiowa.edu`

Bo Liu *
Auburn University
`boliu@auburn.edu`

Abstract

Off-policy policy optimization is a challenging problem in reinforcement learning (RL). The algorithms designed for this problem often suffer from high variance in their estimators, which results in poor sample efficiency, and have issues with convergence. A few variance-reduced on-policy policy gradient algorithms have been recently proposed that use methods from stochastic optimization to reduce the variance of the gradient estimate in the REINFORCE algorithm. However, these algorithms are not designed for the off-policy setting and are memory-inefficient, since they need to collect and store a large "reference" batch of samples from time to time. To achieve variance-reduced off-policy-stable policy optimization, we propose an algorithm family that is memory-efficient, stochastically variance-reduced, and capable of learning from off-policy samples. Empirical studies validate the effectiveness of the proposed approaches.

1 Introduction

Off-policy control and policy search is a ubiquitous problem in real-world applications wherein the goal of the agent is to learn a near-optimal policy π (that is, close to the optimal policy π^*) from samples collected via a (non-optimal) behavior policy μ. Off-policy policy search is important because it can learn from previously collected data generated by non-optimal policies, such as from demonstrations (LfD) [Hester et al., 2018], from experience replay [Mnih et al., 2015], or from executing an exploratory (even randomized) behavior policy. It also enables learning multiple tasks in parallel through a single sensory interaction with the environment [Sutton et al., 2011]. However, research into efficient off-policy policy search has encountered two major challenges: off-policy stability and high variance. There is work in each direction, e.g., addressing off-policy stability [Imani et al., 2018; Zhang et al., 2019] via emphatic weighting [Sutton et al., 2016; Hallak et al., 2016], and reducing the high variance caused by "curse of horizon" [Liu et al., 2018; Xie et al., 2019], but little that addresses both challenges at the same time.

Stochastic variance reduction has recently emerged as a strong alternative to stochastic gradient descent (SGD) in finding first-order critical points in non-convex optimization. The key idea is to

---

*Correspondence to: Bo Liu<`boliu@auburn.edu`>.

replace the stochastic gradient (used by vanilla SGD techniques) with a "*semi-stochastic*" gradient for objective functions with a finite-sum structure. A semi-stochastic gradient combines the stochastic gradient in the current iterate with a snapshot of an earlier iterate, called the *reference iterate*. This line of research includes methods such as SVRG [Johnson and Zhang, 2013; Zhang et al., 2013], SAGA [Defazio et al., 2014], SARAH [Nguyen et al., 2017a], and SPIDER [Fang et al., 2018]. A common feature of these techniques is storing a "reference" sample set in memory to estimate the gradient at a "checkpoint," and then using it in updates across different training epochs. The reference set is usually very large—$O(n)$ in SVRG, for example, where n is the size of the training data. This is a significant obstacle limiting the application of these variance-reduction techniques in deep learning. There has been a recent surge in research applying these "semi-stochastic" gradient methods to policy search to help reduce variance [Papini et al., 2018; Xu et al., 2019a,b], for example. However, there are two drawbacks with these algorithms. The first is that a large "reference" sample set must be stored, which is memory costly. The second is that these algorithms lack off-policy guarantees because they adopt the REINFORCE [Williams, 1992] algorithm as the policy search subroutine, which is only on-policy stable.

In this paper, we aim to address the memory-efficiency and off-policy stability issues of existing stochastic variance-reduced policy search methods, we propose a novel *variance-reduced off-policy* policy search algorithm that is both *convergent* and *memory efficient*. To this end, we introduce novel ingredients, i.e., STOchastic Recursive Momentum (STORM) [Cutkosky and Orabona, 2019], Actor-Critic with Emphatic weightings (ACE) [Imani et al., 2018], and Generalized Off-Policy Actor-Critic (GeoffPAC) [Zhang et al., 2019]. Combining the novel components of ACE/GeoffPAC and STORM offers a number of advantages. First, *ACE and GeoffPAC are off-policy stable.* We choose ACE and GeoffPAC especially they are the only two stable off-policy policy gradient approaches to the best of our knowledge.[1] Second, *STORM is memory-efficient.* STORM is so far the only stochastic variance-reduced algorithm that need not revisit a "fixed" batch of samples. Based on these key ingredients, we propose the **V**ariance-reduced **O**ff-policy **M**emory-efficient **P**olicy **S**earch (VOMPS) algorithm and the **A**ctor-**C**ritic with **E**mphatic weighting and **STO**chastic **R**ecursive **M**omentum (ACE-STORM) algorithm, with two primary contributions: **(1)** VOMPS and ACE-STORM are both off-policy stable. Previous approaches are on-policy by nature (by adopting REINFORCE as their policy search component), and thus cannot be applied to the off-policy setting. To the best of our knowledge, VOMPS and ACE-STORM are the *first* off-policy variance-reduced policy gradient methods. **(2)** The two algorithms are memory-efficient. Unlike previous approaches that must store a large number of samples for reference-point computation, our new algorithms do not need a reference sample set and are thus memory efficient.

Here is a roadmap to the rest of the paper. Sec. 2 of this paper follows by introducing background on stochastic variance reduction. Sec. 3 develops the algorithms and conducts sample-complexity analysis. An empirical study is conducted in Sec. 4. Then Sec. 5 contains more detailed related work and Sec. 6 concludes the paper.

2 Preliminaries

2.1 Stochastic Variance Reduction and STORM

STORM [Cutkosky and Orabona, 2019] is a state-of-the-art stochastic variance-reduction algorithm that avoids the reference sample set storage problem. The stochastic optimization problem is of the form $J(x) = \min_{x \in \mathbb{R}^d} \mathbb{E}\big[f(x, \xi)\big]$, where the function $J : \mathbb{R}^d \to \mathbb{R}$ can be thought of as the training loss of a machine learning model, and $f(x, \xi)$ represents the loss of a sample ξ for the parameter $x \in \mathbb{R}^d$. In this setting, SGD produces a sequence of iterates $x_1, \ldots, x_T$ using the recursion $x_{t+1} = x_t - \eta_t \nabla f(x_t, \xi_t)$, where $f(\cdot, \xi_1), \ldots, f(\cdot, \xi_T)$ are i.i.d. samples and $\eta_1, \ldots \eta_T \in \mathbb{R}$ is a sequence of stepsizes. STORM replaces the gradient in the SGD's update with

$$g_t = \underbrace{(1 - \alpha_t)g_{t-1} + \alpha_t \nabla f(x_t, \xi_t)}_{①} + \underbrace{(1 - \alpha_t)(\nabla f(x_t, \xi_t) - \nabla f(x_{t-1}, \xi_t))}_{②} ,$$

---

[1]Degris et al. [2012] proposed the OffPAC algorithm with certain theoretical incoherence, which was then fixed in [Imani et al., 2018].

where $\alpha_t \in [0, 1]$ is the momentum parameter, ① is the update rule of vanilla SGD with momentum, and ② is an additional term introduced to reduce variance. STORM achieves the so-far optimal convergence rate of $O(1/\epsilon^{3/2})$ to find a ϵ-stationary point—$||\nabla J(x)||^2 \leq \epsilon$. (We report the convergence rates of several variance reduction algorithms in Appendix D as well.) Thus STORM achieves variance reduction using a version of the momentum term, and does not use the estimated gradient at a checkpoint in its update. It alleviates the need to store a large reference sample set and therefore is memory-efficient.

2.2 Reinforcement Learning and Off-Policy Policy Search

In RL, the agent's interaction with the environment is often modeled as a Markov Decision Process (MDP), which is a tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action sets, the transition kernel $p(s'|s, a)$ specifies the probability of transition from state $s \in \mathcal{S}$ to state $s' \in \mathcal{S}$ by taking action $a \in \mathcal{A}$, $r(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a bounded reward function, and $0 \leq \gamma < 1$ is a discount factor. Given a (stochastic) policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, $V_\pi : \mathcal{S} \to \mathbb{R}$ is the associated state value function, $Q_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the state-action value function, and P_π the transition kernel, $P_\pi(s'|s) = \sum_a \pi(a|s)p(s'|s, a)$. In policy gradient methods, π is often approximated in a parametric form π_θ which is differentiable with respect to its parameter θ.

In the off-policy setting, an agent aims to learn a target policy π from samples generated by a behavior policy μ. We assume that the Markov chains induced by policies π and μ are ergodic, and denote by d_π and d_μ their unique stationary distributions. The stationary distribution matrices are $D_\pi := \mathrm{Diag}(d_\pi)$ and $D_\mu := \mathrm{Diag}(d_\mu)$. The standard coverage assumption for π and μ is used, $\forall(s, a), \pi(a|s) > 0$ implies $\mu(a|s) > 0$ [Sutton and Barto, 2018]. With this assumption, the non-trivial importance sampling ratio is well defined, $\rho(s, a) := \frac{\pi(a|s)}{\mu(a|s)}$. For simplicity, we use $\rho_t := \rho(s_t, a_t)$ for the importance sampling ratio at time t. *Distribution mismatch* between the stationary distributions of the behavior and the target policies is the primary challenge in off-policy learning. To correct this mismatch, Sutton et al. [2016] introduced *emphatic weighting*, where for a given state s an emphatic weight $M(s)$ is computed to offset the state-wise distribution mismatch. This technique has recently been widely used for off-policy value function estimation [Sutton et al., 2016; Hallak and Mannor, 2017] and policy optimization [Imani et al., 2018; Zhang et al., 2019].

In policy gradient literature, different objectives have been used. In the on-policy continuing task setting, the goal is often to optimize the *alternative life objective* $J_\pi = \sum_s d_\pi(s)i(s)V_\pi(s)$ [Silver, 2015], which is equivalent to optimizing the average reward objective [Puterman, 2014], when $\gamma = 1$ and interest function $i(s) = 1$, $\forall s \in \mathcal{S}$. On the other hand, in the off-policy continuing task setting where d_π is difficult to achieve due to that the samples are collected from the behavior policy [Imani et al., 2018], it is more practical to resort to the *excursion objective* [Imani et al., 2018]—that is, $J_\mu := \sum_s d_\mu(s)i(s)V_\pi(s)$ instead of J_π, where d_π (in J_π) is replaced by d_μ (in J_μ). However, the excursion objective does not correctly represent the state-wise weighting of the target policy π's performance [Gelada and Bellemare, 2019]. To address this, Zhang et al. [2019] introduced the *counterfactual objective*, $J_{\hat{\gamma}}$, to unify J_μ and J_π in the continuing RL setting:

$$J_{\hat{\gamma}} := \sum_s d_{\hat{\gamma}}(s)\hat{i}(s)V_\pi(s),$$

where $\hat{\gamma} \in [0, 1]$ is a constant, and $d_{\hat{\gamma}}$ is the stationary distribution of the Markov chain with transition matrix $P_{\hat{\gamma}} = \hat{\gamma}P_\pi + (1 - \hat{\gamma})\mathbf{1}d_\mu^\top$. $d_{\hat{\gamma}} = (1 - \hat{\gamma})(\mathbf{I} - \hat{\gamma}P_\pi^\top)^{-1}d_\mu$ ($\hat{\gamma} < 1$) and $d_{\hat{\gamma}} = d_\pi$ ($\hat{\gamma} = 1$), and $\hat{i}$ is a user-defined extrinsic interest function. In these equations, $\mathbf{I}$ and $\mathbf{1}$ are the identity matrix and all-one column vector. Zhang et al. [2019] argue that $J_{\hat{\gamma}}$ is potentially a better objective for off-policy control, for the following reasons: **1)** $J_{\hat{\gamma}}$ is more general than J_π and J_μ, since J_π and J_μ can be recovered from $J_{\hat{\gamma}}$ for $\hat{\gamma} = 1$ and $\hat{\gamma} = 0$, respectively. This is because for $\hat{\gamma} = 1$ and $\hat{\gamma} = 0$, we have $d_{\hat{\gamma}} = d_\pi$ and $d_{\hat{\gamma}} = d_\mu$ [Gelada and Bellemare, 2019]. An intermediate $\hat{\gamma}$ tweaks the stationary distribution towards that of the target policy and makes the objective closer to the original alternative life objective. **2)** $J_{\hat{\gamma}}$ is more suitable than J_μ for the off-policy setting, as it better reflects state-wise weighting of π's performance and typically leads to a better empirical performance according to the observation of [Zhang et al., 2019]. The Generalized Off-Policy Actor-Critic (GeoffPAC) algorithm is a state-of-the-art approach that optimizes $J_{\hat{\gamma}}$. A key component of the GeoffPAC algorithm is the *emphatic weight update component*, which is discussed in detail in the Appendix. As noted above, when $\hat{\gamma} = 0$, the stationary distribution $d_{\hat{\gamma}}$ reduces to d_μ, and correspondingly the objective

3

of GeoffPAC ($J_{\hat{\gamma}}$) reduces to the that of Actor-Critic with Emphatic-weighting (ACE) algorithm (J_μ) [Imani et al., 2018].

3 Algorithm Design and Analysis

3.1 VOMPS Algorithm Design

We consider off-policy policy optimization of infinite-horizon discounted MDP problems, which is identical to the problem setting of ACE and GeoffPAC. Our new algorithm the **V**ariance-reduced **O**ff-policy **M**emory-efficient **P**olicy **S**earch (VOMPS) is presented in Algorithm 1. For simplicity, the subscript of π is omitted for V_π and Q_π. We denote the state value function as $V(s; \nu)$ and the policy function as $\pi(a|s; \theta)$ with ν and θ being their parameters. A parametric approximation of the density ratio function, $C(s; \psi)$ is introduced to reweight online updates to the value function in order to avoid divergence issues in off-policy learning [Gelada and Bellemare, 2019; Zhang et al., 2019]. VOMPS is an off-policy actor-critic method that uses emphatic weighting based policy gradient for off-policy stability guarantee, and stochastic recursive momentum for memory-efficient variance reduction. Algorithm 1 is illustrated below in the order of updating the critic, the density ratio, the emphatic weights, and the actor. The hyperparameters in the algorithm are identified in the Appendix. *First*, the *critic update* is conducted using a Temporal Difference (TD) method:

$$\delta_t = r_t + \gamma V(s_{t+1}; \nu_t) - V(s_t; \nu_t), \qquad \nu_{t+1} = \nu_t + \alpha_\nu \delta_t \nabla_\nu V(s_t; \nu_t) , \qquad (1)$$

where δ_t is the TD error at t-th timestep, and α_ν is the stepsize. In fact, the critic is not limited to the TD method and can be replaced by other approaches in order to improve the value function estimation. *Second*, the *density ratio update* is performed:

$$\psi_{t+1} = \psi_t + \alpha_\psi \big(\hat{\gamma}\rho_t C(s_t; \psi_t) + (1 - \hat{\gamma}) - C(s_{t+1}; \psi_t)\big)\nabla_\psi C(s_{t+1}; \psi_t) , \qquad (2)$$

where α_ψ is the stepsize. *Third*, we conduct the *emphatic weights update* of $M_t^{(1)}, M_t^{(2)}$ that are used to correct the impact of the distribution discrepancy between π and μ. For the *counterfactual objective* of $J_{\hat{\gamma}}$, the policy gradient $\nabla J_{\hat{\gamma}}$ is computed as follows:

$$\begin{aligned}
\nabla J_{\hat{\gamma}} &= \sum_s d_{\hat{\gamma}}(s)\hat{i}(s)\nabla V(s) + \sum_s \nabla d_{\hat{\gamma}}(s)\hat{i}(s)V(s) \\
&= \sum_s d_\mu(s)C(s)\hat{i}(s)\sum_a Q(s,a)\nabla\pi(a|s) + \sum_s d_\mu(s)\nabla C(s)\hat{i}(s)V(s) \\
&= \mathbb{E}_\mu\big[M_t^{(1)}\rho_t\delta_t\nabla\log\pi(a_t|s_t) + \hat{\gamma}M_t^{(2)}V(s_t)\hat{i}(s_t)\big] .
\end{aligned}$$

Specifically, $M_t^{(1)}$ (resp. $M_t^{(2)}$) is used to adjust the weighting of $\rho_t\delta_t\nabla\log\pi(a_t|s_t)$ (resp. $\hat{\gamma}V(s_t)\hat{i}(s_t)$) caused by the discrepancy between d_π and d_μ. Details of the update law is shown in Appendix B, which is adopted from the emphatic weight update component proposed in [Zhang et al., 2019]. Let $Z_t := M_t^{(1)}\rho_t\delta_t\nabla\log\pi(a_t|s_t) + \hat{\gamma}M_t^{(2)}V(s_t)\hat{i}(s_t)$ be an estimate of the policy gradient at time t, then according to [Zhang et al., 2019], we have $\mathbb{E}_\mu[Z_t] = \nabla J_{\hat{\gamma}}$. *Fourth*, the *actor update* via policy gradient is conducted. Instead of using the vanilla actor update in [Zhang et al., 2019], we introduce the STORM [Cutkosky and Orabona, 2019] technique to reduce the variance in the gradient estimates. According to the technique used in STORM [Cutkosky and Orabona, 2019], both $Z_t(a_t, s_t; \theta_{t-1})$ and $Z_t(a_t, s_t; \theta_t)$ need to be calculated as follows:

$$Z_t(a_t, s_t; \theta_t) = M_t^{(1)}\rho_t\delta_t\nabla_\theta\log\pi(a_t|s_t; \theta_t) + \hat{\gamma}M_t^{(2)}V(s_t; \nu)\hat{i}(s_t) ,$$

$$Z_t(a_t, s_t; \theta_{t-1}) = M_t^{(1)}\rho_t\delta_t\nabla_\theta\log\pi(a_t|s_t; \theta_{t-1}) + \hat{\gamma}M_t^{(2)}V(s_t; \nu)\hat{i}(s_t) . \qquad (3)$$

The actor's update law is formulated as $\theta_{t+1} = \theta_t + \eta_t g_t$, where the two key ingredients are the *stochastic recursive momentum* update term g_t and the *adaptive stepsize* η_t. The update term g_t is computed as

$$g_t = Z_t(a_t, s_t; \theta_t) + (1 - \alpha_t)\big(g_{t-1} - Z_t(a_t, s_t; \theta_{t-1})\big) , \qquad (4)$$

and the adaptive stepsizes η_t and α_t are computed as follows, with k, w, and β inherited from STORM,

$$G_t = \|Z_t(a_t, s_t; \theta_t)\|, \quad \eta_t = k/(w + \sum_{i=1}^t G_t^2)^{1/3} , \quad \alpha_t = \beta\eta_{t-1}^2. \qquad (5)$$

It should be noted that $Z_t(a_t, s_t; \theta_t)$ is used in Eq. (4) & (5), while $Z_t(a_t, s_t; \theta_{t-1})$ is used in Eq. (4).

Algorithm 1 Variance-reduced Off-policy Memory-efficient Policy Search (VOMPS)

$V(s; \nu)$: state value function parameterized by ν
$C(s; \psi)$: density ratio estimation parameterized by ψ
$\pi(a|s; \theta)$: policy function parameterized by θ

1: **Input**: Parameters ψ, ν, θ;
2: **for** timestep $t = 0$ to T **do**
3:    Sample a transition s_t, a_t, r_t, s_{t+1} according to behavior policy μ.
4:    **Critic update** according to Eq. (1).
5:    **Density ratio update** according to Eq. (2).
6:    **Emphatic weights update**: update $M_t^{(1)}, M_t^{(2)}, Z_t(a_t, s_t; \theta_t)$ as in Figure 5 in the Appendix.
7:    Compute actor stepsize as in Eq. (5), $Z_t(a_t, s_t; \theta_{t-1})$ as in Eq. (3) and g_t as in Eq. (4).
8:    **Actor update** as $\theta_{t+1} = \theta_t + \eta_t g_t$.
9: **end for**
10: **Output I**: Parameters $\psi_{T+1}, \nu_{T+1}, \theta_{T+1}$.
11: **Output II**: Parameters $\psi_{T+1}, \nu_{T+1}, \theta_\tau$, where τ is sampled with a probability of $p(\tau = t) \propto \frac{1}{\eta_t^2}$.

3.2 Theoretical Analysis

In this section, we present a theoretical analysis of the VOMPS algorithm. To start, we first present the assumptions used in the study.

Assumption 1 (**Bounded Gradient**). *[Papini et al., 2018; Xu et al., 2019b] Let $\pi_\theta(a|s)$ be the agent's policy at state s. There exist constants $W, U > 0$ such that the log-density of the policy function satisfies: $\|\nabla_\theta \log \pi_\theta(a|s)\|_2 \leq W,\quad \left\|\nabla_\theta^2 \log \pi_\theta(a|s)\right\|_2 \leq U$, for $\forall\, a \in \mathcal{A}$ and $s \in \mathcal{S}$, and $\|\cdot\|_2$ is the ℓ_2 norm.*

Assumption 2 (**Lipschitz continuity and Bounded Variance**). *[Xu et al., 2019a,b; Cutkosky and Orabona, 2019] The estimation of policy gradient $Z(\theta)$ is bounded, Lipschitz continuous, and has a bounded variance, i.e., there exist constants L, G, σ such that $\|Z(\theta_1) - Z(\theta_2)\|_2 \leq L\|\theta_1 - \theta_2\|_2$ for $\forall\, \theta_1, \theta_2 \in \mathbb{R}^d$, and $\|Z(\theta)\|_2 \leq G, \mathbb{E}[\|Z(\theta) - \nabla J_{\hat{\gamma}}(\theta)\|_2^2] \leq \sigma^2$ for $\forall\, \theta \in \mathbb{R}^d$.*

We now present our main theoretical result, the convergence analysis of Algorithm 1.

Theorem 1. *Under the above assumptions, for any $b > 0$, let $k = \frac{b\sigma^{\frac{2}{3}}}{L}$, $\beta = 28L^2 + \sigma^2/(7Lk^3) = L^2(28 + 1/(7b^3))$, and $w = \max\left((4Lk)^3, 2\sigma^2, \left(\frac{\beta k}{4L}\right)^3\right) = \sigma^2 \max\left((4b)^3, 2, (28b + \frac{1}{7b^2})^3/64\right)$. Then, the output of Algorithm 1 satisfies*

$$\mathbb{E}\left[\|\nabla J_{\hat{\gamma}}(\hat{\theta})\|^2\right] = \mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T}\|\nabla J_{\hat{\gamma}}(\theta_t)\|^2\right] \leq \frac{40\Delta_\Phi}{T^{2/3}} + \frac{2\beta^2\sigma^2}{L^2 T^{2/3}},$$

where $\Delta_\Phi \leq \Delta_{J_{\hat{\gamma}}} + \frac{\|\epsilon_0\|^2}{32\eta_0 L^2}$, $\Delta_{J_{\hat{\gamma}}} = J_{\hat{\gamma}}(\theta^*) - J_{\hat{\gamma}}(\theta), \forall \theta \in R^d$, and $\theta^\star$ is the maximizer of $J_{\hat{\gamma}}$.

Theorem 1 indicates that VOMPS requires $O(1/\epsilon^{3/2})$ samples to find an ϵ-stationary point. As VOMPS is developed based on GeoffPAC and STORM, the proof of convergence rates of VOMPS is similar to STORM. However, the proof is not a trivial extension by merely instantiating the objective function to $J_{\hat{\gamma}}$ in the RL settings. If we just apply the original analysis of STORM, we can only achieve $O(\frac{\log(1/\epsilon)}{\epsilon^{2/3}})$. In Yuan et al. [2020], it improves the sample complexity to $O(1/\epsilon^{2/3})$ by introducing the large mini-batch $O(1/\sqrt{\epsilon})$ and using extremely small stepsize $o(\epsilon)$. Nevertheless the improved sample complexity, the introduced $O(1/\sqrt{\epsilon})$ mini-batch size will lead to memory inefficiency, and the $O(\epsilon)$ stepsize will slow down the training process. VOMPS overcomes the above two weaknesses and achieves the $O(1/\epsilon^{2/3})$ sample complexity by applying increasing weights strategy and an automatically adjusted stepsize strategy that proportions to iterations. These techniques are not used in the original STORM method. As a result, VOMPS is the first variance reduced memory-efficient off-policy method that achieves the optimal sample complexity, which matches the lower-bound provided in Arjevani et al. [2019].

Remark 1. *Theorem 1 indicates that VOMPS enjoys the same convergence rate as the state-of-the-art algorithms together with SRVR-PG [Xu et al., 2019b]. A summary of state-of-the-art convergence*

Algorithms	Objective	Sample Complexity	Off-Policy?	Required Batch
SVRPG [Papini et al., 2018]	J_π	$O(1/\epsilon^2)$	$\times$	$O(1/\epsilon)$
SVRPG [Xu et al., 2019a]	J_π	$O(1/\epsilon^{5/3})$	$\times$	$O(1/\epsilon^{2/3})$
SRVR-PG [Xu et al., 2019b]	J_π	$O(1/\epsilon^{3/2})$	$\times$	$O(1/\epsilon^{1/2})$
ACE-STORM (This paper)	J_μ	$O(1/\epsilon^{3/2})$	$\checkmark$	$\times$
VOMPS (This paper)	$J_{\hat\gamma}$	$O(1/\epsilon^{3/2})$	$\checkmark$	$\times$

Table 1: Comparison on convergence rate of different algorithms when $\|\nabla J(\theta)\|_2^2 \leq \epsilon$. The $\times$ in "Required Batch" means that no mini-batch is needed, aka, the algorithm is memory efficient.

rate is summarized in Table 1. It should be noted that unlike other algorithms that optimize J_π, the objective function of VOMPS is $J_{\hat\gamma}$. However, these two objective functions only differ in constants in the on-policy setting. Thus, it is a fair comparison of the convergence rate.

3.3 ACE-STORM Algorithm Design

As an extension, we also propose the **A**ctor-**C**ritic with **E**mphatic weighting and **STO**chastic **R**ecursive **M**omentum (ACE-STORM) algorithm, which is an integration of the ACE algorithm [Imani et al., 2018] and STORM [Cutkosky and Orabona, 2019]. Similar to discussed above, the objective of ACE-STORM is also a special case of that in Algorithm 1 by setting $\hat\gamma = 0$. The pseudo-code of the ACE-STORM Algorithm is presented in the Appendix due to space constraints.

4 Experiments and Results

The experiments are conducted to investigate the following questions empirically. i) How do VOMPS&ACE-STORM compare with state-of-art off-policy policy gradient methods, such as Geoff-PAC [Zhang et al., 2019], ACE, DDPG [Lillicrap et al., 2015], and TD3 [Fujimoto et al., 2018]? ii) How do VOMPS&ACE-STORM compare with on-policy variance-reduced policy gradient methods, e.g., SVRPG [Papini et al., 2018] and SRVR-PG [Xu et al., 2019b]? iii) How is VOMPS&ACE-STORM resilient to action noise in policy gradient methods?

Since the tasks for original domains are episodic, e.g., `CartPoleContinuous-v0`, `Hopper-v2` and `HalfCheetah-v2`, we modify them as continuing tasks following [Zhang et al., 2019]— the discount factor γ is set to 0.99 for all non-terminal states and 0 for terminal states. The environment is reset to the initial state if the agent reaches the terminal state. Therefore, simulation-based cumulative rewards (Monte Carlo return) by executing the (learned) policy π is used as the performance metric, while results of episodic return are also provided in the Appendix. Detailed parameter settings are reported in the Appendix. All the curves in the results are averaged over 10 runs, where the solid curve indicates the mean and the shaded regions around the mean curve indicate standard deviation errors. To better visualize the plots, curves are smoothed by a window of size 20. Shorthands "1K" represents 10^3, and "1M" represents 10^6. In the off-policy setting, the behavior policy μ follows a fixed uniform distribution. VOMPS, ACE-STORM, GeoffPAC, and ACE have the same critic component in all experiments for a fair comparison.

4.1 Tabular off-policy policy gradient

We first compare the performance of ACE, GeoffPAC, ACE-STORM, and VOMPS on the two-circle MDP domain [Imani et al., 2018; Zhang et al., 2019] in terms of their dynamic and asymptotic solutions. In the two-circle MDP, there are a finite number of states, and an agent only decides at state A on either transitioning to state B or state C, whereas the transitions at other states will always be deterministic. The discount factor $\gamma = 0.6$ and rewards are 0 unless specified on the edge as shown in Fig. 1(a). The algorithms for this domain, GeoffPAC, ACE, VOMPS, and ACE-STORM, are implemented with a tabular version, where value function and density ratio function is computed via dynamic programming. The behavior policy μ follows a uniform distribution, and $\pi(A \rightarrow B)$, the probability from A to B under the target policy π is reported in Fig. 1(b). As shown in Fig. 1(b), VOMPS and GeoffPAC gradually choose to transition from A to B so that the agent would take the
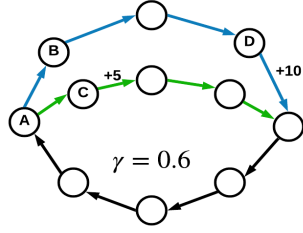
(a) Two-circle MDP
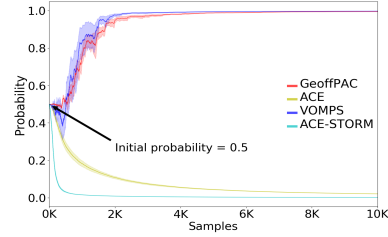
(b) The probability of transitioning from A to B

Figure 1: The two-circle MDP

route with blue color and obtain a reward of $+10$. Compared with GeoffPAC, VOMPS converges faster. Both ACE-STORM and ACE move from A to C, and ACE-STORM converges faster than ACE. Moving from A to C is an inferior solution since the agent will take the route with green color and fail to obtain a higher reward. The difference between asymptotic solutions of GeoffPAC/VOMPS and ACE/ACE-STORM is due to the difference between the objective functions $J_{\hat{\gamma}}, J_{\mu}$, and the difference in the training process is due to the STORM component integrated into VOMPS and ACE-STORM.
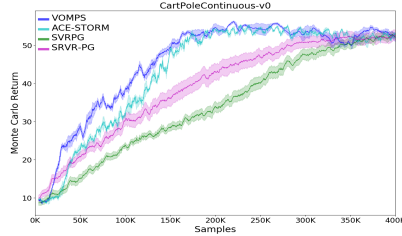
(a) Comparison with on-policy methods
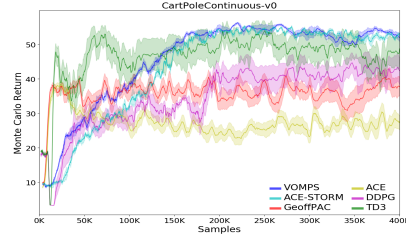
(b) Comparison with off-policy methods

Figure 2: Results on `CartPoleContinuous-v0`

4.2 Classic Control

We use `CartPoleContinuous-v0` for CartPole domain, which has a continuous action space within the range of $[-1, 1]$. A near-optimal policy can reach a Monte-Carlo return at the level of 57 within a fixed horizon of 200 timesteps. As shown in Fig. 2(a), VOMPS and ACE-STORM learn the near-optimal policy with around 200K samples, while SVRPG and SRVR-PG need more than 400K samples with larger dynamic variances. As Fig. 2(b) shows, ACE, GeoffPAC, and DDPG do not perform well in this domain. Although TD3 seems to learn faster at the beginning, it reaches an inferior solution with a mean return around 50 a higher variance than VOMPS and ACE-STORM.

4.3 Mujoco Robot Simulation

Experiments are also conducted on two benchmark domains provided by OpenAI Gym, including `Hopper-v2` and `HalfCheetah-v2`. As shown in Fig. 3(a), 4(a), both GeoffPAC and VOMPS can achieve higher Monte Carlo returns than other methods and converge faster within 1M samples on `Hopper-v2`. Compared with GeoffPAC, the learning curve of VOMPS is smoother and has a smaller variance. The results on `HalfCheetah-v2` are shown in Fig. 3(b), 4(b). Fig. 3(b) indicates that VOMPS and ACE-STORM outperform SVRPG and SRVR-PG by a large margin, and Fig. 4(b) demonstrates that VOMPS and ACE-STORM achieve a similar performance of Geoff-PAC/DDPG/TD3, with obviously smaller variances. We also observe that ACE does not perform well in general, and DDPG has a very large variance in these two domains.

In addition, a 20% action noise is added to both the learning process and evaluation process in order to compare the noise resistance ability of different approaches (aka, the action is multiplied by a factor of $1 \pm 0.2\chi$, where χ is drawn from a $[0, 1]$-range uniform distribution). As shown

in Fig. 3(c), 4(c), 3(d), 4(d), compared with results under the original noise-free setting, VOMPS, ACE-STORM, SVRPG, and SRVR-PG tend to be insensitive to disturbances than other methods, which validates the effectiveness of the stochastic variance reduction component of these algorithms. In particular, VOMPS and ACE-STORM appear to be empirically the most noise-resistant in these two domains.
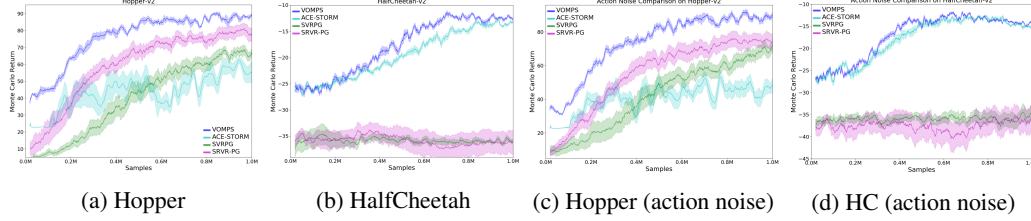

| (a) Hopper | (b) HalfCheetah | (c) Hopper (action noise) | (d) HC (action noise) |

Figure 3: Comparison with on-policy PG methods (Mujoco), "HC" is short for HalfCheetah.

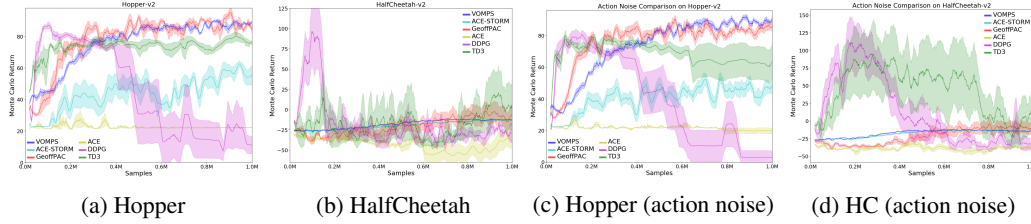| (a) Hopper | (b) HalfCheetah | (c) Hopper (action noise) | (d) HC (action noise) |

Figure 4: Comparison with off-policy PG methods (Mujoco), "HC" is short for HalfCheetah.

5 Related Work

Policy gradient methods and the corresponding actor-critic algorithm [Sutton et al., 2000; Konda, 2002] are popular policy search methods in RL, especially for continuous action setting. However, this class of policy search algorithms suffers from large variance [Peters and Schaal, 2006; Deisenroth et al., 2013]. Several approaches have been proposed to reduce variance in policy search. The first method family is to use control variate method, such as baseline removal [Sutton and Barto, 2018], to remove a baseline function in the policy gradient estimation [Weaver and Tao, 2001; Greensmith et al., 2002; Gu et al., 2017; Tucker et al., 2018]. The second method family is based on tweaking batch size, stepsize, and importance ratio used in policy search. In this research line, [Pirotta et al., 2013] proposed using an adaptive step size to offset the effect of the policy variance. Pirotta et al. [2013]; Papini et al. [2017] studied the adaptive batch size and proposed to optimize the adaptive step size and batch size jointly, and Metelli et al. [2018] investigated reducing variance via importance sampling. The third branch of methods is based on the recently developed *stochastic variance reduction* [Johnson and Zhang, 2013; Allen-Zhu and Hazan, 2016; Reddi et al., 2016b] methods as discussed above. Several variance-reduced policy gradient methods were proposed in this direction, such as SVRPG [Papini et al., 2018], SRVR-PG [Xu et al., 2019b], etc.

6 Conclusion

In this paper, we present off-policy convergent, memory-efficient, and variance-reduced policy search algorithms by leveraging emphatic-weighted policy search and stochastic recursive momentum-based variance reduction. Experimental study validates the performance of the proposed approaches compared with existing on-policy variance-reduced policy search methods and off-policy policy search methods under different settings. Future work along this direction includes integrating with baseline removal methods for further variance reduction and investigating algorithmic extensions to risk-sensitive policy search and control.

References

Allen-Zhu, Z. and Hazan, E. (2016). Variance reduction for faster non-convex optimization. In *International conference on machine learning*, pages 699–707.

Arjevani, Y., Carmon, Y., Duchi, J. C., Foster, D. J., Srebro, N., and Woodworth, B. (2019). Lower bounds for non-convex stochastic optimization. *arXiv preprint arXiv:1912.02365*.

Cutkosky, A. and Orabona, F. (2019). Momentum-based variance reduction in non-convex sgd. In *Advances in Neural Information Processing Systems*, pages 15210–15219.

Defazio, A., Bach, F., and Lacoste-Julien, S. (2014). SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pages 1646–1654.

Degris, T., White, M., and Sutton, R. S. (2012). Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*.

Deisenroth, M. P., Neumann, G., Peters, J., et al. (2013). A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1–2):1–142.

Fang, C., Li, C. J., Lin, Z., and Zhang, T. (2018). SPIDER: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In *Advances in Neural Information Processing Systems*, pages 689–699.

Fujimoto, S., Van Hoof, H., and Meger, D. (2018). Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*.

Gelada, C. and Bellemare, M. G. (2019). Off-policy deep reinforcement learning by bootstrapping the covariate shift. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*.

Greensmith, E., Bartlett, P. L., and Baxter, J. (2002). Variance reduction techniques for gradient estimates in reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1507–1514.

Gu, S., Lillicrap, T., Ghahramani, Z., Turner, R. E., and Levine, S. (2017). Q-prop: Sample-efficient policy gradient with an off-policy critic. In *International Conference on Learning Representations 2017*.

Hallak, A. and Mannor, S. (2017). Consistent on-line off-policy evaluation. In *Proceedings of the 34th International Conference on Machine Learning*.

Hallak, A., Tamar, A., Munos, R., and Mannor, S. (2016). Generalized emphatic temporal difference learning: Bias-variance analysis. In *Proceedins of 30th AAAI Conference on Artificial Intelligence*.

Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Horgan, D., Quan, J., Sendonaris, A., Osband, I., et al. (2018). Deep Q-learning from demonstrations. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Imani, E., Graves, E., and White, M. (2018). An off-policy policy gradient theorem using emphatic weightings. In *Advances in Neural Information Processing Systems*.

Johnson, R. and Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*.

Konda, V. R. (2002). *Actor-critic algorithms*. PhD thesis, Massachusetts Institute of Technology.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Liu, Q., Li, L., Tang, Z., and Zhou, D. (2018). Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Advances in Neural Information Processing Systems*.

Metelli, A. M., Papini, M., Faccio, F., and Restelli, M. (2018). Policy optimization via importance sampling. In *Advances in Neural Information Processing Systems*, pages 5442–5454.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takáč, M. (2017a). SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2613–2621. JMLR. org.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takáč, M. (2017b). Stochastic recursive gradient algorithm for nonconvex optimization. *arXiv preprint arXiv:1705.07261*.

Papini, M., Binaghi, D., Canonaco, G., Pirotta, M., and Restelli, M. (2018). Stochastic variance-reduced policy gradient. In *Proceedings of the 35th International Conference on Machine Learning*.

Papini, M., Pirotta, M., and Restelli, M. (2017). Adaptive batch size for safe policy gradients. In *Advances in Neural Information Processing Systems*, pages 3591–3600.

Peters, J. and Schaal, S. (2006). Policy gradient methods for robotics. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2219–2225.

Pirotta, M., Restelli, M., and Bascetta, L. (2013). Adaptive step-size for policy gradient methods. In *Advances in Neural Information Processing Systems*, pages 1394–1402.

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Reddi, S. J., Hefny, A., Sra, S., Póczos, B., and Smola, A. (2016a). Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pages 314–323.

Reddi, S. J., Sra, S., Poczos, B., and Smola, A. J. (2016b). Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In *Advances in Neural Information Processing Systems*, pages 1145–1153.

Silver, D. (2015). Reinforcement learning. *Course at University College of London. Available online at http://www0. cs. ucl. ac. uk/staff/d. silver/web/Teaching. html*.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction (2nd Edition)*. MIT press.

Sutton, R. S., Mahmood, A. R., and White, M. (2016). An emphatic approach to the problem of off-policy temporal-difference learning. *The Journal of Machine Learning Research*.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*.

Tucker, G., Bhupatiraju, S., Gu, S., Turner, R. E., Ghahramani, Z., and Levine, S. (2018). The mirage of action-dependent baselines in reinforcement learning. In *International Conference on Machine Learning*.

Weaver, L. and Tao, N. (2001). The optimal reward baseline for gradient-based reinforcement learning. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Xie, T., Ma, Y., and Wang, Y.-X. (2019). Towards optimal off-policy evaluation for reinforcement learning with marginalized importance sampling. In *Advances in Neural Information Processing Systems*, pages 9665–9675.

Xu, P., Gao, F., and Gu, Q. (2019a). An improved convergence analysis of stochastic variance-reduced policy gradient. In *Conference on Uncertainty in Artificial Intelligence*.

Xu, P., Gao, F., and Gu, Q. (2019b). Sample efficient policy gradient methods with recursive variance reduction. *arXiv preprint arXiv:1909.08610*.

Yuan, H., Lian, X., Liu, J., and Zhou, Y. (2020). Stochastic recursive momentum for policy gradient methods. *arXiv preprint arXiv:2003.04302*.

Zhang, L., Mahdavi, M., and Jin, R. (2013). Linear convergence with condition number independent access of full gradients. In *Advances in Neural Information Processing Systems*, pages 980–988.

Zhang, S., Boehmer, W., and Whiteson, S. (2019). Generalized off-policy actor-critic. In *Advances in Neural Information Processing Systems*, pages 1999–2009.