2019 IEEE 58th Conference on Decision and Control (CDC)
Palais des Congrès et des Expositions Nice Acropolis
Nice, France, December 11-13, 2019

Policy Gradient using Weak Derivatives for Reinforcement Learning

Sujay Bhatt, Alec Koppel, Vikram Krishnamurthy

***Abstract*— This paper considers policy search in continuous state-action reinforcement learning problems. Typically, one computes search directions using a classic expression for the policy gradient called the Policy Gradient Theorem, which decomposes the gradient of the value function into two factors: *the score function* and the $Q-$function. This paper presents four results: (i) an alternative policy gradient theorem using weak (measure-valued) derivatives instead of score-function is established; (ii) the stochastic gradient estimates thus derived are shown to be unbiased and to yield algorithms that converge almost surely to stationary points of the non-convex value function of the reinforcement learning problem; (iii) the sample complexity of the algorithm is derived and is shown to be $O(1/\sqrt{k})$; (iv) finally, the expected variance of the gradient estimates obtained using weak derivatives is shown to be lower than those obtained using the popular *score-function* approach. Experiments on OpenAI gym pendulum environment illustrate the superior performance of the proposed algorithm.**

I. Introduction

Reinforcement Learning (RL) is a form of implicit stochastic adaptive control where the optimal control policy is estimated without directly estimating the underlying model. This paper considers reinforcement learning for an infinite horizon discounted cost continuous state Markov decision process. In a MDP, actions affect the Markovian state dynamics and result in rewards for the agent. The objective is to find a map from the states to actions, also known as policy, that results in the accumulation of largest expected return. There are many approaches to estimate a policy: policy iteration, $Q-$learning [1], [2] (which operates in "value" space [3]), policy-gradients [4], [5] (that operate in policy space); see [6], [7].

Recently, policy-gradient algorithms have gained popularity due to their ability to address complex real-world RL problems with *continuous state-action spaces*. Given a parametrized policy space, usually designed to incorporate domain knowledge, policy-gradient algorithms update policy parameters along an estimated ascent direction of the expected return. Depending on whether the expected reward or the value function is convex or non-convex, the parameters converge to a minimum or a stationary point; for a comprehensive survey see [8], [9].

Typically, to compute the ascent direction in policy search [10], one employs the Policy Gradient Theorem [7] to write the gradient as the product of two factors: the $Q-$function[1] and the score function (a likelihood ratio). This score function approach has yielded numerous policy search techniques [11], [12], [13], [7], although the resulting gradient estimates are afflicted with high variance: the score function is a martingale and so for a Markov process its variance is $O(N)$ for N measurements. In pursuit of reducing the variance, we propose replacing the score function with the weak derivatives; see [14] for a textbook treatment.[2] Weak and measured-valued derivatives have been used for real-time reinforcement learning of constrained average cost MDPs (with finite action spaces) in [16], [17], [18], [19], [20]. These papers derive constant step size policy gradient algorithms and show analytically and via numerical examples that substantial variance reduction can be achieved compared to the score function method; moreover the optimal (randomized) policy can be tracked over time when the unknown constrained MDP parameters evolve.

In comparison to [16], [20], this paper considers off-line (decreasing step size) reinforcement learning for continuous state-continuous action infinite horizon discounted cost MDPs when the underlying system can be simulated using statistically independent trials with different policies. To estimate the $Q-$function in the policy gradient [7], we use Monte Carlo roll-outs with random path lengths akin to [21], motivated by the fact that obtaining unbiased estimates of continuous state-action $Q-$function in the infinite horizon case is otherwise challenging. The product of these terms yields a valid estimate of the overall policy gradient, as in [7].

Our main results are:

1) A decreasing step size policy gradient algorithm using Jordan decomposition for the policy gradient. We establish that the resulting policy gradient algorithm, named Policy Gradient with Jordan Decomposition (PG-JD), yields unbiased estimates of the gradient of the reward function.
2) to establish that the PG-JD algorithm converges to a stationary point of the parametrized value function almost surely under decreasing step-sizes.
3) to derive the iteration (and sample[3]) complexity as $O(1/\sqrt{k})$, where k is the time step. This shows that the convergence rate is similar to stochastic gradient method for non-convex settings.
4) to upper-bound the expected variance of the gradient estimates obtained using the PG-JD algorithm, which

[1] $Q-$function is also known as the state-action value function [7]. It gives the expected return for a choice of action in a given state.

[2] The Hahn-Jordan decomposition [15] of signed measures is a specific type of weak derivative form - this expresses the derivative of a measure as the weighted difference of orthogonal measures. For example, the gradient of a Gaussian policy [12] can be expressed as a (scaled) difference of two Rayleigh policies.

[3] Iteration complexity is a measure of the number of changes of the unknown parameter. Sample complexity includes the additional simulations required to estimate the continuous state-action $Q-$function using Monte Carlo roll-out with random path lengths.

978-1-7281-1398-2/19/$31.00 ©2019 IEEE

isshown to be lower than those generated by score function methods using Monte Carlo roll-outs with random path lengths, for common policy parametrizations.

The setup and problem formulation are discussed in Sec. II. The new policy gradient theorem using weak derivatives (Jordan decomposition) is derived in Sec. III. The algorithm to compute the stochastic gradient and the policy parameter update is given in Sec. IV. Convergence analysis of the stochastic gradient ascent algorithm and its statistical properties are derived in Sec. V. Numerical studies on OpenAI gym using the pendulum environment is discussed in Sec. VI.

II. Problem Formulation and Policy Search

The problem of reinforcement learning is considered in the framework of Markov Decision Process, which is defined as a tuple $(\mathcal{X}, \mathcal{A}, \mathcal{T}, r, \gamma)$ consisting of the *state space* $\mathcal{X} \subseteq \mathbb{R}^p$, a subset of Euclidean space with elements $x \in \mathcal{X}$; the *action space* $\mathcal{A} \subseteq \mathbb{R}^q$, a subset of Euclidean space with elements $a \in \mathcal{A}$; the *transition law* $\mathcal{T}$, a probability density function $\mathcal{T}(\cdot|a, x) \in \mathbb{P}(\mathcal{X})$ that assigns a next-state upon taking action a in state x, where $\mathbb{P}(\mathcal{X})$ denotes the set of all probability measures on $\mathcal{X}$; the *reward function* $r(x, a)$, a real valued function on the product space $\mathcal{X} \times \mathcal{A}$; the *discount* $\gamma \in (0, 1)$, a parameter that scales the importance of future rewards.

A *stochastic Markov policy* $\boldsymbol{\mu} = \{\mu_k\}$ is defined as a sequence of transition probabilities from $\mathcal{X}$ to $\mathcal{A}$ such that $\mu_k(D(x)|x) = 1$ for each $x \in \mathcal{X}$ and $k = 0, 1, \cdots$. Here D maps each $x \in \mathcal{X}$ to the set of all available actions $D(x)$. Let Σ denote the class of stochastic Markov policies.

For an initial state x_0 and a stochastic Markov policy $\boldsymbol{\mu} \in \Sigma$, define the expected reward function

$$J(x_0, \boldsymbol{\mu}) = \lim_{N\to\infty} \mathbb{E}^{x_0}_{\boldsymbol{\mu}} \Big\{ \sum_{k=0}^{N} \gamma^k r(x_k, a_k) \ \Big| a_k \sim \mu_k(\cdot|x_k) \Big\} \tag{1}$$

For an initial state x_0 and a Markov policy $\boldsymbol{\mu} \in \Sigma$, using Ionescu Tulcea theorem [22], [23], define $\mathbb{P}^{x_0}_{\boldsymbol{\mu}}$ as

$$\mathbb{P}^{x_0}_{\boldsymbol{\mu}}(dx_0 da_0 \cdots dx_k da_k \cdots) = \mu_0(dx_0) \prod_{k=1}^{\infty} \mu_k(da_k|x_k) \times \mathcal{T}(dx_k|x_k, a_k). \tag{2}$$

Here $\mu_0 \in \mathbb{P}(\mathcal{X})$ is an atomic measure with $\mu_0(x_0) = 1$. The expectation $\mathbb{E}^{x_0}_{\boldsymbol{\mu}}$ in (1) is with respect to $\mathbb{P}^{x_0}_{\boldsymbol{\mu}}$ in (2). Our goal is to find the policy $\boldsymbol{\mu}$ that maximizes the long-term reward accumulation, or value:

$$\boldsymbol{\mu}^* = \arg\sup_{\boldsymbol{\mu}\in\Sigma} \lim_{N\to\infty} \mathbb{E}^{x_0}_{\boldsymbol{\mu}} \Big\{ \sum_{k=0}^{N} \gamma^k r(x_k, a_k) \Big| a_k \sim \mu_k(\cdot|x_k) \Big\}. \tag{3}$$

For the infinite horizon problem (3), it is sufficient [24], [25], [23], [26] to restrict the class Σ of policies to the class $\Sigma_s \subset \Sigma$ of stationary stochastic Markov policies. A *stationary stochastic Markov policy* $\boldsymbol{\mu}(= \{\mu\}) \in \Sigma_s$ is defined as the transition probability from $\mathcal{X}$ to $\mathcal{A}$ such that $\mu(D(x)|x) = 1$ for each $x \in \mathcal{X}$. In order to solve (3) we resort to direct policy search over the space of continuous stationary policies. It is convenient to *parametrize* the stationary policy $\mu(\cdot|\cdot)$ as $\mu_\theta(\cdot|\cdot)$ for $\theta \in \Theta \subseteq \mathbb{R}^d$, for $d \in \mathbb{N}$, and search over the space of θ. For example, consider Gaussian policy $\mu_\theta(\cdot|x) = \mathcal{N}(\theta'\phi(x), \sigma^2)$. Here the function $\phi(\cdot)$ is commonly referred to as the feature map and σ denotes the standard deviation. With a slight abuse of notation, the problem (3) can be reformulated in terms of the finding a parameter vector θ to satisfy:

$$\theta^* = \arg\max_{\theta\in\mathbb{R}^d} J(\theta), \tag{4}$$

$$J(\theta) = \lim_{N\to\infty} \mathbb{E}^{x_0}_{\mu_\theta} \Big\{ \sum_{k=0}^{N} \gamma^k r(x_k, a_k) \ \Big| a_k \sim \mu_\theta(\cdot|x_k) \Big\}.$$

Here $\mathbb{E}^{x_0}_{\mu_\theta}$ is the expectation with respect to the measure induced by the probability measure as in (2) with the policy $\boldsymbol{\mu}_\theta = \{\mu_\theta\}$ and initial state x_0.

III. Policy Gradient Theorem via Hahn-Jordan

The foundation of any valid policy search technique is a valid ascent direction on the value function with respect to the policy parameters. Classically, one may derive that the policy gradient decomposes into two factors: the action-value (Q) function and the score function [4]. Here we establish that one may obviate the need for the log trick that gives rise to the score function through measure-valued differentiation by employing the Jordan decomposition of signed measures [15]. To begin doing so, define the $Q-$function as

$$Q_{\mu_\theta}(x, a) = \mathbb{E}_{\mu_\theta} \Big\{ \sum_{k=0}^{\infty} \gamma^k r(x_k, a_k) \Big| x_0 = x, a_0 = a \Big\}. \tag{5}$$

The weak derivative of the signed measure $\nabla\mu_\theta(\cdot|x)$ using Jordan decomposition [4] is given as

$$\nabla\mu_\theta(\cdot|x) = g(\theta, x) \Big\{ \mu^{\oplus}_\theta(\cdot|x) - \mu^{\ominus}_\theta(\cdot|x) \Big\} \tag{6}$$

Here the decomposed positive and negative component measures $\mu^{\oplus}_\theta(\cdot|x)$ and $\mu^{\ominus}_\theta(\cdot|x)$ are orthogonal in L^2 (see Example 1 below). The ergodic measure associated with the transition kernel $\mathcal{T}(\cdot|x_0, a_0)$ and policy μ_θ is $\pi_{\mu_\theta}(x) = (1-\gamma)\sum_{k=0}^{\infty} \gamma^k \cdot \mathcal{T}(x_k = x|x_0, \mu_\theta)$. The induced measures on $\mathcal{X} \times \mathcal{A}$ by $\mu^{\oplus}_\theta$ and $\mu^{\ominus}_\theta$ are defined as $\mu^{\oplus}_\theta(x, a) \triangleq \mu^{\oplus}_\theta(a|x) \cdot \pi_{\mu_\theta}(x)$ and $\mu^{\ominus}_\theta(x, a) \triangleq \mu^{\ominus}_\theta(a|x) \cdot \pi_{\mu_\theta}(x)$. Using this measure (weak) derivative representation of the policy, we can write the gradient of the value function with respect to policy parameters θ in an unusual way which is given in the following theorem.

[4]
Result 1. *[15]* [Hahn Decomposition] Let μ be a finite signed measure on the measurable space $(\Omega, \mathcal{F})$. There exists a disjoint partition of the set Ω into Ω^+ and Ω^- such that $\Omega = \Omega^+ \cup \Omega^-$, $\mu(A) \geq 0, \forall A \subset \Omega^+$, and $\mu(B) \leq 0, \forall B \subset \Omega^-$.

Result 2. *[15]* [Jordan Decomposition] Every finite signed measure μ has a unique decomposition into a difference $\mu = \mu^+ - \mu^-$ of two finite non-negative measures μ^+ and μ^- such that for any Hahn decomposition (Ω^+, Ω^-) of μ, we have for $A \in \mathcal{F}$ that $\mu^+(A) = 0$ if $A \subset \Omega^-$ and $\mu^-(A) = 0$ if $A \subset \Omega^+$.

Theorem 1. *(Jordan Decomposition for Policy Gradients) The policy gradient using Jordan decomposition takes the form*

$$\nabla J(\theta) = \frac{1}{1-\gamma}\Big[\mathbb{E}_{(x,a)\sim\mu_\theta^{\oplus}(\cdot,\cdot)}\Big\{g(\theta,x)\cdot Q_{\mu_\theta}(x,a)\Big\} - \mathbb{E}_{(x,a)\sim\mu_\theta^{\ominus}(\cdot,\cdot)}\Big\{g(\theta,x)\cdot Q_{\mu_\theta}(x,a)\Big\}\Big]. \quad (7)$$

where $g(\theta,x)$ is a normalizing constant to ensure $\mu^{\oplus}$ and $\mu^{\ominus}$ are valid measures.

Discussion: The proof can be found in [27]. Theorem 1 is the policy gradient theorem using weak derivatives, specifically Jordan decomposition. In Theorem 1, note that the Q functions in the expectations are the same, indicating that the model is unaffected by the measure decomposition; only the induced measures are different. The expression for the gradient in (7) contains a difference of two expectations. Unlike, the method of score functions, the expectation obviates the need for a *score function* term. Intuitively, this allows us to avoid computing the logarithm of the policy which may amplify useless parts of the state-action space and cause variance to needlessly be increased, and instead yield a sharp "perceptron-like" behavior. In subsequent sections, we indeed establish that this representation may reduce variance but this reduction intrinsically depends on the policy parameterization. Note that $g(\theta,x)$ for a given parameter θ and state x, is a constant, which makes the stochastic gradient easier to compute in Algorithm 2. Before continuing, we present a representative example; see [14] for several examples.

Example 1. Consider a gaussian policy $\mu_\theta(\cdot|x) = \mathcal{N}(\theta'\phi(x),\sigma^2)$, where the mean of the gaussian distribution is modulated by the optimization parameter. The Jordan decomposition of the gaussian policy can be derived as follows:

$$\mu_\theta(\cdot|x) = \mathcal{N}(\theta'\phi(x),\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\Big(\frac{(a-\theta'\phi(x))^2}{2\sigma^2}\Big). \quad (8)$$

$$\begin{aligned}\nabla\mu_\theta(\cdot|x) &= \frac{1}{\sqrt{2\pi\sigma^2}}\exp\Big(\frac{(a-\theta'\phi(x))^2}{2\sigma^2}\Big)\\ &\quad\times\frac{1}{\sigma^2}(a-\theta'\phi(x))\cdot\phi(x).\\ &:= g(\theta,x)\Big\{\mu_\theta^{\oplus}(\cdot|x)-\mu_\theta^{\ominus}(\cdot|x)\Big\}, \quad (9)\end{aligned}$$

Here we may glean the normalizing constant $g(\theta,x)=\frac{\phi(x)}{\sqrt{2\pi\sigma^2}}$ and the positive and negative component measures are

$$\mu_\theta^{\oplus}(\cdot|x) = \frac{1}{\sigma^2}(a-\theta'\phi(x))\cdot\exp\Big(\frac{(a-\theta'\phi(x))^2}{2\sigma^2}\Big), \quad (10)$$

$$\mu_\theta^{\ominus}(\cdot|x) = \frac{1}{\sigma^2}(\theta'\phi(x)-a)\cdot\exp\Big(\frac{(a-\theta'\phi(x))^2}{2\sigma^2}\Big). \quad (11)$$

Observe that $\mu_\theta^{\oplus}(\cdot|x)$ and $\mu_\theta^{\ominus}(\cdot|x)$ define the Rayleigh[5] policy. They are orthogonal in the sense that $\mu_\theta^{\oplus}(\cdot|x)$ is defined on [6] $\chi(a>\theta'\phi(x))$ and $\mu_\theta^{\ominus}(\cdot|x)$ is defined over $\chi(a<\theta'\phi(x))$.

[5]The probability density function corresponding to Rayleigh distribution is: $f(x)=\frac{x}{\sigma^2}\cdot\exp\Big(\frac{x^2}{2\sigma^2}\Big)$, $x\geq 0$.

[6]$\chi(\cdot)$ denotes the indicator function.

IV. Policy Search via Jordan Decomposition

In order to develop a policy search method based on Theorem 1, we need samples of both factors inside the expectation in (7) which are unbiased. We first focus on the later factor, the $Q-$function.

A. Estimating the Action-Value

The estimation of the $Q-$function is carried out using Monte Carlo roll-outs of random path lengths, similar to [21]. Here the random length is a geometric random variable with parameter γ, the discount factor in the reinforcement learning problem. Specifically, we simulate $T\sim$ Geom $(1-\gamma)$ and then simulate state-action pairs according to the positive and negative induced policies $\pi^{\oplus}$ and $\pi^{\ominus}$. For this time horizon, we collect rewards for the two different trajectories.

More specifically, from a given starting state x_0, a (real) trajectory is simulated to update the policy parameters θ. At each epoch k of the parameter update θ_k, the simulator (modeled as $(\mathcal{S}(=\mathcal{X}),\mathcal{A},\mathcal{T},r,\gamma)$) is called two times to simulate two different (phantom[7]) trajectories. These trajectories correspond to the random Monte-Carlo roll-outs used to estimate the $Q-$functions with two different policies, the positive and negative policy measure, and hence the stochastic gradient of the expected reward function. Let T denote a geometrically distributed random variable: $T\sim$ Geom$(1-\gamma)$ where γ is the discount factor. Let the path-wise cost be defined by $\mathcal{R}_{\mu_\theta}^T=\sum_{k=0}^T r(x_k,a_k)\Big|a_k\sim\mu_\theta(\cdot|x_k)$.

Discussion: Algorithm 2 with Algorithm 1 is the stochastic gradient algorithm that is used to update the policy parameters. The simulation consists of a single simulation (real trajectory) to update the parameters and multiple phantom simulations to estimate the gradient of the expected reward function. The two phantom trajectories correspond to different polices and not different models, starting from the system's state represented by the state corresponding to the real trajectory. The stochastic gradient computation is summarized in three steps: For a fixed initial state– (i) Simulate two phantom initial actions from the measures obtained using Jordan decomposition, i.e, $\mu_{\theta_k}^{\ominus}(\cdot|s_0^{\ominus})$ and $\mu_{\theta_k}^{\oplus}(\cdot|s_0^{\oplus})$. (ii) Simulate a geometric random variable T_k, and (iii) Perform Monte Carlo roll-outs of length T_k-1 (i.e, simulate and feed actions to the simulator and collect the rewards) using the system policy derived from old parameters, i.e using $\{\mu_{\theta_k}(\cdot|s_u^{\oplus})\}_{u=1}^{u=T_k-1}\}$ and $\{\mu_{\theta_k}(\cdot|s_u^{\ominus})\}_{u=1}^{a=T_k-1}\}$.

The merit of using these random horizons for estimation of the Q function, as summarized in Algorithm 1, is that one may establish that it is an unbiased estimate in the infinite-horizon discounted case, as we summarize in the following theorem.

Theorem 2. *For a geometric r.v T, let the approximate state-action value function (Q-function) be defined by $\hat{Q}_{\mu_\theta}(x,a;T)=\mathbb{E}_{\mu_\theta}\Big\{\sum_{k=0}^T r(x_k,a_k)\Big|x_0=x,a_0=a\Big\}$. Let*

[7]Here the word "phantom" is used to refer to the actions on the simulator.

Algorithm 1 Unbiased estimation of Q_μ

Input: Trajectory length T_k, states $s_0 = s_0^\oplus, s_0^\ominus$, phantom actions $a_0^s = a_0^\oplus, a_0^\ominus$, simulator policies $\mu = \mu_{\theta_k}^\oplus, \mu_{\theta_k}^\ominus$.

Output: Unbiased Q-function estimates: $\mathcal{R}_{\hat{\mu}_\theta^\oplus}^{T_k}$ and $\mathcal{R}_{\hat{\mu}_\theta^\ominus}^{T_k}$.

Initialize $\mathcal{R}_\mu^{T_k} \leftarrow 0$.

for all $\mu = \mu_{\theta_k}^\oplus, \mu_{\theta_k}^\ominus$ and $t = 0, 1, 2, \cdots, T_k - 1$ **do**

$\mathcal{R}_\mu^{T_k} \leftarrow \mathcal{R}_\mu^{T_k} + r(s_t, a_t^s)$.

$s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t^s)$, $a_{t+1}^s \sim \mu(\cdot|s_{t+1})$.

end for

Algorithm 2 Policy Gradient with Jordan Decomposition (PG-JD)

Input: System state x_{k+1}, parameter vector θ_k, and continuous random policy μ_{θ_k}.

Output: Parameter θ_{k+1} and next system input $a_{k+1} \sim \mu_{\theta_{k+1}}$.

Step 1. Simulate $T_k \sim$ Geom$(1-\gamma)$, i.e., $P(T_k = t) = (1-\gamma)\gamma^t$.

Define the initial conditions: $s_0^\oplus, s_0^\ominus = x_{k+1}$.

Define: $\hat{\mu}_{\theta_k}^\oplus \triangleq \{\mu_{\theta_k}^\oplus(\cdot|s_0^\oplus), \{\mu_{\theta_k}(\cdot|s_a^\oplus)\}_{a=1}^{a=T_k-1}\}$ as the policy for trajectory 1.

Define: $\hat{\mu}_{\theta_k}^\ominus \triangleq \{\mu_{\theta_k}^\ominus(\cdot|s_0^\ominus), \{\mu_{\theta_k}(\cdot|s_a^\ominus)\}_{a=1}^{a=T_k-1}\}$ as the policy for trajectory 2.

Step 2. Simulate $a_0^\oplus \sim \mu_{\theta_k}^\oplus(\cdot|s_0^\oplus)$ and $a_0^\ominus \sim \mu_{\theta_k}^\ominus(\cdot|s_0^\ominus)$.

Step 3. Compute $Q_{\hat{\mu}_{\theta_k}^\oplus}(s_0^\oplus, a_0^\oplus)$ and $Q_{\hat{\mu}_{\theta_k}^\ominus}(s_0^\ominus, a_0^\ominus)$ using Algorithm 1.

Step 4. Compute $\hat{\nabla} J(\theta_k) = \frac{g(\theta_k, x_{k+1})}{1-\gamma} \cdot \left\{ \mathcal{R}_{\hat{\mu}_{\theta_k}^\oplus}^{T_k} - \mathcal{R}_{\hat{\mu}_{\theta_k}^\ominus}^{T_k} \right\}$

Step 5. Compute $\theta_{k+1} = \theta_k + \epsilon_k \cdot \hat{\nabla} J(\theta_k)$.

T denote a geometrically distributed random variable. Then,

$$\mathbb{E}_{\mu_\theta}\left\{\mathcal{R}_{\mu_\theta}^T\right\} = \hat{Q}_{\mu_\theta}(x, a; T). \tag{12}$$

$$\mathbb{E}_T\left\{\hat{Q}_{\mu_\theta}(x, a; T)\right\} = Q_{\mu_\theta}(x, a). \tag{13}$$

The proof can be found in [27]. Now that we may obtain unbiased samples of the action-value function, we shift focus to how to compute the stochastic gradients needed for policy search based on Jordan decomposition (Theorem 1).

B. Stochastic Gradient Algorithm

With the estimation of the action-value function addressed, we now discuss how we can sample the former factor: the signed measure gradients. Specifically, Theorem 1 can be used to effectively compute the gradient given access to an oracle/simulator that may generate state-action-reward triples. It is well known that one only needs to compute estimates of the gradient that are unbiased in expectation to ensure convergence of the iterates to a stationary point [7]. This results in a modification of the gradient expression as in REINFORCE algorithm [11], [7], which is a stochastic gradient, for computing the optimal policy of the reinforcement learning problem. Let $\mathbb{E}_T$ denote the expectation with respect to the geometric distribution.

Using Theorem 2 and Fubini's Theorem [28], the gradient in (7) can be rewritten to make it implementable on a simulator:

$$\nabla J(\theta) = \frac{1}{1-\gamma}\Big[\mathbb{E}_T\Big\{\mathbb{E}_{(x,a)\sim\mu_\theta^\oplus(\cdot,\cdot)}\Big\{g(\theta, x) \cdot \hat{Q}_{\mu_\theta}(x, a; T)\Big\} - \mathbb{E}_{(x,a)\sim\mu_\theta^\ominus(\cdot,\cdot)}\Big\{g(\theta, x) \cdot \hat{Q}_{\mu_\theta}(x, a; T)\Big\}\Big\}\Big] \tag{14}$$

We have from Theorem 2 and (14),

$$\hat{\nabla} J_T(\theta) = \frac{g(\theta, x_0)}{1-\gamma}\left[\mathcal{R}_{\hat{\mu}_\theta^\oplus}^T - \mathcal{R}_{\hat{\mu}_\theta^\ominus}^T\right] \tag{15}$$

$$\hat{\nabla} J(\theta) = \frac{g(\theta, x_0)}{1-\gamma}\left[\mathcal{R}_{\hat{\mu}_\theta^\oplus}^{T_z} - \mathcal{R}_{\hat{\mu}_\theta^\ominus}^{T_z}\right] \tag{16}$$

Here the initial state simulated from the ergodic measure is $x_0 \sim \pi_{\mu_\theta}(x)$, and the policies that simulate the two trajectories are: $\hat{\mu}_\theta^\oplus \triangleq \{\mu_\theta^\oplus, \{\mu_\theta\}_l\}, l = 1, 2, \cdots$ and $\hat{\mu}_\theta^\ominus \triangleq \{\mu_\theta^\ominus, \{\mu_\theta\}_l\}, l = 1, 2, \cdots$. Here the initial actions are simulated from the decomposed measures and the parametrized policy is used for the remainder of the trajectory simulation. Here (15) is the (stochastic) gradient estimate for a random path length T and (16) is the (stochastic) gradient estimate using a realization T_z. Using the estimates (16) that are computable using Algorithm 1 to estimate the Q function with respect to the signed measures, then, we may write out an iterative stochastic gradient method to optimize θ with respect to the value function as

$$\theta_{k+1} = \theta_k + \epsilon_k \cdot \hat{\nabla} J(\theta_k)\ . \tag{17}$$

The overall policy search routine is summarized as Algorithm 2. Its convergence and variance properties are discussed in the following section.

V. Convergence, Complexity, & Variance Analysis

In this section, we discuss a few properties of the stochastic gradient ascent algorithm derived using weak derivatives, namely, convergence, the iteration complexity, sample complexity, and the variance of the resulting gradient estimates.

A. Convergence Analysis

We now analyze the convergence of the PG-JD algorithm (Algorithm 2), establishing that the stochastic gradient estimates obtained from the algorithm are unbiased estimates of the true gradient, and that the parameter sequence (17) converges almost surely to a stationary point of the value function (4). To do so, some assumptions are required which we state next.

1) Assumptions:

(i) The reward function[8] $r(x, a)$ is bounded Lipschitz, i.e,

$$|r(x, a)| \leq M(< \infty),\ \forall (x, a) \in \mathcal{X} \times \mathcal{A}.$$
$$\forall (x_1, x_2, a_1, a_2) \in \mathcal{X}^2 \times \mathcal{A}^2,$$
$$|r(x_1, a_1) - r(x_2, a_2)| \leq L_r \cdot d_{\mathcal{X}\mathcal{A}}((x_1, a_1), (x_2, a_2)).$$

[8]Let the product space $\mathcal{X} \times \mathcal{A}$ be equipped with the taxi-cab norm:

$$d_{\mathcal{X}\mathcal{A}}((x_1, a_1), (x_2, a_2)) = d_{\mathcal{X}}(x_1, x_2) + d_{\mathcal{A}}(a_1, a_2)$$
$$\forall (x_1, x_2, a_1, a_2) \in \mathcal{X}^2 \times \mathcal{A}^2,$$

where $d_{(\cdot)}$ denotes the corresponding metric on the Euclidean space.

(ii) The transition law[9] $\mathcal{T}(\cdot|x,a)$ is Lipschitz, i.e,

$$\forall (x_1,x_2,a_1,a_2) \in \mathcal{X}^2 \times \mathcal{A}^2,$$
$$\mathcal{K}\Big(\mathcal{T}(\cdot|x_1,a_1),\mathcal{T}(\cdot|x_2,a_2)\Big) \leq L_{\mathcal{T}} \cdot d_{\mathcal{X}\mathcal{A}}((x_1,a_1),(x_2,a_2))$$

(iii) For $\theta \in \mathbb{R}^d$, the transition law $\mathcal{T}(\cdot|x,\mu_\theta)$ is $\psi-$irreducible, positive Harris recurrent, and geometrically ergodic.

(iv) The continuous policy $\mu_\theta(a|x)$ is Lipschitz, i.e,

$$\forall (x_1,x_2) \in \mathcal{X}^2, \theta \in \Theta,$$
$$\mathcal{K}\Big(\mu_\theta(\cdot|x_1),\mu_\theta(\cdot|x_2)\Big) \leq L_\theta \cdot d_{\mathcal{X}}(x_1,x_2).$$

(v) $\sum_k \epsilon_k = \infty$ and $\sum_k \epsilon_k^2 < \infty$.

(vi) The stochastic gradient

$$\mathbb{E}\Big\{\|\hat{\nabla} J(\theta)\|^2\Big\} \leq m + n\|\nabla J(\theta)\|^2$$

for all $\theta \in \Theta$, and $n, m > 0$.

Assumptions (i) - (iii) are model assumptions, whereas Assumptions (iv) - (vi) impose restricts about how the algorithm behaves. Assumption (i) is standard, and tied to learnability of the problem. Assumption (ii) is a continuity assumption on the transition law that is easily satisfied by most physical systems. Assumption (iii) makes sure that for every policy μ_θ, there exists a unique invariant (stationary) measure and the Markov chain reaches stationarity geometrically fast; see [30]. All the results hold without the transition law being geometrically ergodic. Assuming geometric ergodicity makes simulating from the ergodic measure (in Algorithm 2, Sec.IV) more meaningful. Regarding the algorithmic conditions: Assumptions (iv)-(v) are standard in stochastic gradient methods; see [31]. Assumption (vi) says that the stochastic gradient is always bounded by the true gradient, which can grow unbounded with θ. This assumption makes sure that the martingale noise of the stochastic gradient is bounded by the true gradient; see [31].

Proposition 1. *Under Assumption (i), the expected cost $J(\theta)$ in the reinforcement learning problem (4) is a bounded real-valued function, i.e,*

$$|J(\theta)| \leq \frac{M}{1-\gamma} \ \forall \ \theta \in \Theta. \tag{18}$$

The following result makes sure that the stochastic gradient estimates so obtained are representative of the true gradient.

Theorem 3. *The stochastic gradient obtained in (16) is an unbiased estimate of the true gradient $\nabla J(\theta)$, i.e,*

$$\mathbb{E}\Big\{\hat{\nabla} J(\theta)\Big\} = \nabla J(\theta). \tag{19}$$

Discussion: The proof can be found in [27]. Theorem 3 says that the estimates of the stochastic gradient are unbiased in expectation. This is required to ensure the almost sure convergence of the iterates to a stationary point [7].

Theorem 4. *Consider the sequence of policy parameters generated by Algorithm 2. Under Assumptions (i) - (vi), the sequence of iterates $\{\theta_k\}$ satisfies*

$$\theta_k \to \theta^*, \ \textit{where } \nabla J(\theta^*) = 0, \ \textit{almost surely}. \tag{20}$$

Discussion: The proof can be found in [27]. The expected cost function $J(\theta)$, under model assumptions, is continuous and $L-$ Lipschitz; see [Chapter 7] [32] and [29]. Theorem 4 says that the sequence of iterates $\{\theta_k\}$ converges to θ^* with probability one, and since $J(\theta)$ is a continuous function, $J(\theta_k)$ converges to $J(\theta^*)$ with probability one. The gradient (which can be unbounded) at iterates $\{\theta_k\}$ is such that $\nabla J(\theta^*) = 0$ with probability one.

B. Sample Complexity

In this section, we consider the convergence rate analysis of the PG-JD algorithm. We choose the stepsize to be $\epsilon_k = k^{-b}$ for some parameter $b \in (0,1)$. Since the optimization of $J(\theta)$ is generally non-convex, we consider the convergence rate in terms of a metric of non-stationarity, i.e., the norm of the gradient $\|\nabla J(\theta)\|^2$. The following theorem considers a step-size that diminishes more slowly than Assumption (v), which yields a $O(1/\sqrt{k})$ rate for the decrement of the expected gradient norm square $\mathbb{E}\|\nabla J(\theta_k)\|^2$.

Theorem 5. *Let $\Big\{\theta_k\Big\}_{k\geq 0}$ be the sequence of parameters of the policy μ_{θ_k} generated by Algorithm 2. Let the stepsize be $\epsilon_k = k^{-b}$ for $b \in (0,1)$ and $\Delta = \min\Big\{\varepsilon, \eta\Big\}$ for some $\varepsilon, \eta > 0$. Let*

$$K_\Delta = \min\Big\{k : \inf_{0\leq d\leq k} \mathbb{E}[\|\nabla J(\theta_d)\|^2] \leq \Delta\Big\} \tag{21}$$

denote the number of iteration steps for the norm of the expected cost to come within the error neighborhood. Then, under Assumptions (i) - (iv), (vi)

$$K_\Delta = O(\Delta^{-1/p}), \ \textit{where } p = \min\Big\{1-b, b\Big\}, \tag{22}$$

where optimizing the complexity bound over b, we have $b = 1/2$. Therefore, $K_\Delta = O(\Delta^{-2})$.

Discussion: See [27] for proof. Theorem 5 characterizes the iteration complexity, which is a measure of the number of iteration steps of the algorithm are required to settle down on a stationary point of the value function. The iteration complexity is $O(1/\sqrt{k})$ showing that the convergence rate is similar to the stochastic gradient methods for non-convex settings. We emphasize that despite the re-invention of such proofs in machine learning, in fact much more general analysis has been developed in the literature. For example Proposition 5, pg 294 in [33] gives general L^q error bounds for $q/geq2$ in the presence of Markovian noise (whereas our setting is i.i.d. noise). Our intention is to identify these rates in the context and language of modern RL.

[9] As in [29], $\mathcal{K}(\upsilon,\nu)$ denotes the Kantorovich distance between probability distributions υ and ν. It is given by:

$$\mathcal{K}(\upsilon,\nu) \triangleq \sup_f \Big\{\Big|\int f d\upsilon - \int f d\nu\Big| : \|f\|_1 \leq 1\Big\}.$$

Corollary 6. *Let γ denote the discount factor and K_Δ denote the iteration complexity. The average sample complexity M_γ^Δ using Algorithm 2 is given as:*

$$M_\gamma^\Delta = \Big(\frac{1+\gamma}{1-\gamma}\Big) K_\Delta. \tag{23}$$

Discussion: The proof can be found in [27]. Corollary 6 characterizes the sample complexity, which is a measure of the number of the expected total number of actions and states realized. Higher the discount factor γ, longer the two (random) Monte-Carlo roll-outs (trajectories) that need to simulated, and hence higher the sample complexity. Together the complexity results, Theorem 5 and Corollary 6, provide an estimate of the duration and expected number of simulations to learn a stationary solution for the reinforcement learning task considered.

C. Variance Analysis

In this section, we provide an analysis of the variance of the stochastic gradient estimates obtained using weak derivatives and score function approaches. Since the $Q-$function estimation in the computation of the gradient is performed using random Monte Carlo roll-outs as in [21], the stochastic gradient obtained is a function of the geometric random variable T that characterizes the roll-out (trajectory) length. To obtain a comparison of the different methods – weak derivatives and score function – we consider the expected variance of the gradient estimates. The proof of Theorem 7 can be found in [27]. The proof of Theorem 8 is similar and hence omitted.

Theorem 7. *The expected variance of the gradient estimates $\hat{\nabla} J$ obtained using weak derivatives is given as:*

$$\mathbb{E}\Big\{ Var^{WD}(\hat{\nabla} J_T(\theta)) \Big\} \leq \frac{2 \cdot M^2 \cdot G_{WD}}{(1-\gamma)^5}, \tag{24}$$

where $G_{WD} = \mathbb{E}_{x\sim\mu_\theta}\Big\{ \|g(\theta, x)\|^2 \Big\}$.

Theorem 8. *The expected variance of the gradient estimates $\hat{\nabla} J$, if score function is used instead of weak derivatives, is given as:*

$$\mathbb{E}\Big\{ Var^{SF}(\hat{\nabla} J_T(\theta)) \Big\} \leq \frac{M^2 \cdot G_{SF}}{(1-\gamma)^5}, \tag{25}$$

where $G_{SF} = \mathbb{E}_{(x,a)\sim\mu_\theta(a|x)}\Big\{ \|\nabla \mu_\theta(a|x)\|^2 \Big\}$.

Corollary 9. *For the Gaussian policy $\mu_\theta(\cdot|x) = \mathcal{N}(\theta' \phi(x), \sigma^2)$, we have*

$$G_{WD} = \frac{1}{2 \cdot \pi} G_{SF}. \tag{26}$$

Hence, the maximum expected variance of the gradient estimates using weak derivatives is smaller than those obtained using the score function method.

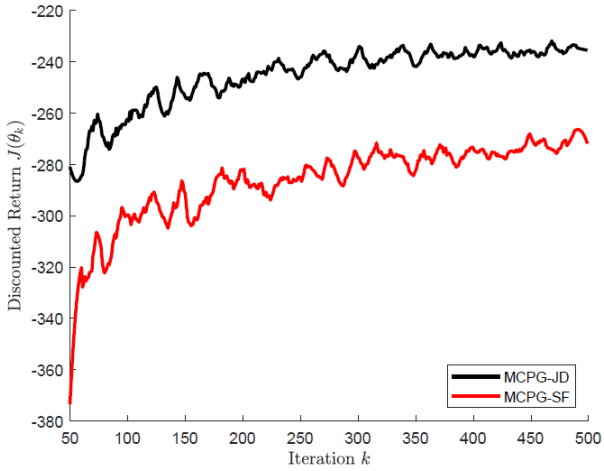

Fig. 1. The convergence of the discounted return as a function of the number of iterations of the policy gradient algorithms. Here at each iteration k, the discounted return $J(\theta) = \mathbb{E}_{\pi_\theta}\{\sum_{k=0}^{\infty} \gamma^k r(x_k, a_k)\}$ is evaluated over 50 trajectories with $\gamma = 0.97$. Observe that the discounted return is higher on average using Monte-Carlo PG-JD as opposed to PG-SF. It can be attributed to algorithm iterates converging to a "better" stationary point due to smaller variance in the gradient estimates.

VI. Numerical Studies

In this section, we present a simple experiment using PG-JD algorithm on the Pendulum environment in OpenAI gym [34]. The performance is compared with Monte Carlo Policy Gradient using Score Function (PG-SF) which is akin to REINFORCE [35] with random roll-out horizons; see Fig.1. In the simulation environment, the pendulum starts at a random position, and the goal is to swing it up so that it stays upright. The environment state is a vector of dimension three, i.e., $x_k = (\cos(\varphi_k), \sin(\varphi_k), \dot{\varphi_k})^\top$, where φ_k is the angle between the pendulum and the upright direction, and $\dot{\varphi}_k$ is the derivative of φ_k. The action a_k is a one-dimensional scalar modified using a $\tanh$-*function*, and represents the joint effort.

The received reward $r(x_k, a_k)$ is given as

$$r(x_k, a_k) := -(\varphi_k^2 + 0.1 * \dot{\varphi_k}^2 + 0.001 * a_k^2), \tag{27}$$

which lies in $[-16.2734,\ 0]$, φ_k is normalized between $[-\pi, \pi]$ and a_k lies in $[-2, 2]$. The transition dynamics are specified by Newton's Second Law of Motion. We use Gaussian policy π_θ, which is parameterized as $\pi_\theta(\cdot|x) = \mathcal{N}(\theta^T \phi(x), \sigma^2)$, where $\sigma = 1.0$ and $\phi(x)(= x)$ being the feature vector. The policy is a stationary policy (time-homogeneous) as it is well known [6] to be sufficient for infinite or random horizon discounted MDP problems. Observe that the discounted return is higher on average using PG-JD as opposed to PG-SF, which may attributable to the variance-reduced properties of the policy gradient estimates using signed measures as compared with the score function.

Remark: It is noted that for common parametrizations of the mean of the Gaussian policy [12], for example like linear – $\theta^T \phi(s)$, the score function is unbounded with respect to θ with the expression being $\frac{(a-\theta^T \phi(s))}{\sigma^2} \phi(s)$. This results in convergence issues in policy gradient algorithms

for unbounded θ and unbounded state spaces. However, using Jordan decomposition, even with linear parametrization and unboundedness, the convergence of the policy gradient algorithm is ensured due to the absence of explicit function of θ.

REFERENCES

[1] C. Watkins and J. C. Hellaby, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, UK, May 1989.

[2] E. Tolstaya, A. Koppel, E. Stump, and A. Ribeiro, "Nonparametric stochastic compositional gradient descent for Q-learning in continuous markov decision problems," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 6608–6615.

[3] A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, "Policy evaluation in continuous MDPs with efficient kernelized gradient temporal difference," 2017.

[4] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in neural information processing systems*, 2000, pp. 1057–1063.

[5] K. Zhang, A. Koppel, H. Zhu, and T. Basar, "Global Convergence of Policy Gradient Methods: A Nonconvex Optimization Perspective," *SIAM Journal on control and Optimization (under review)*, 2019.

[6] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 2005, vol. 1, no. 3.

[7] R. S. Sutton, A. G. Barto *et al.*, *Reinforcement Learning: An Introduction*, 2nd ed., 2017.

[8] I. Grondman, L. Busoniu, G. A. Lopes, and R. Babuska, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1291–1307, 2012.

[9] M. P. Deisenroth, G. Neumann, J. Peters *et al.*, "A survey on policy search for robotics," *Foundations and Trends® in Robotics*, vol. 2, no. 1–2, pp. 1–142, 2013.

[10] D. Silver, "Reinforcement learning and simulation-based search," *Doctor of philosophy, University of Alberta*, 2009.

[11] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[12] K. Doya, "Reinforcement learning in continuous time and space," *Neural Computation*, vol. 12, no. 1, pp. 219–245, 2000.

[13] E. Greensmith, P. L. Bartlett, and J. Baxter, "Variance reduction techniques for gradient estimates in reinforcement learning," *Journal of Machine Learning Research*, vol. 5, no. Nov, pp. 1471–1530, 2004.

[14] G. Pflug, *Optimization of Stochastic Models: The Interface between Simulation and Optimization*. Kluwer Academic Publishers, 1996.

[15] P. Billingsley, *Probability and measure*. John Wiley & Sons, 2008.

[16] F. V. Abad and V. Krishnamurthy, "Constrained stochastic approximation algorithms for adaptive control of constrained Markov decision processes," in *42nd IEEE Confernce on Decision and Control*, 2003, pp. 2823–2828.

[17] V. Krishnamurthy, F. V. Abad, and K. Martin, "Implementation of gradient estimation to a constrained Markov decision problem," in *42nd IEEE Confernce on Decision and Control*, 2003.

[18] V. Krishnamurthy and F. J. V. Abad, "Gradient based policy optimization of constrained markov decision processes," in *Stochastic Processes, Finance and Control: A Festschrift in Honor of Robert J Elliott*. World Scientific, 2012, pp. 503–547.

[19] V. Krishnamurthy, *Partially Observed Markov Decision Processes*. Cambridge University Press, 2016.

[20] V. Krishnamurthy and F. Vazquez Abad, "Real-time reinforcement learning of constrained markov decision processes with weak derivatives," *arXiv preprint arXiv:1110.4946*, 2018.

[21] S. Paternain, "Stochastic Control Foundations of Autonomous Behavior," Ph.D. dissertation, University of Pennsylvania, 2018.

[22] J. Neveu, *Mathematical foundations of the calculus of probability*. Holden-day, 1965.

[23] O. Hernández-Lerma and J. B. Lasserre, *Discrete-time Markov control processes: basic optimality criteria*. Springer Science & Business Media, 2012, vol. 30.

[24] D. Blackwell, "Discounted dynamic programming," *The Annals of Mathematical Statistics*, vol. 36, no. 1, pp. 226–235, 1965.

[25] D. P. Bertsekas and S. E. Shreve, *Stochastic optimal control: the discrete-time case*. Academic Press Inc.[Harcourt Brace Jovanovich Publishers], New York, 1978.

[26] E. A. Feinberg, "On measurability and representation of strategic measures in Markov decision processes," *Lecture Notes-Monograph Series*, pp. 29–43, 1996.

[27] S. Bhatt, A. Koppel, and V. Krishnamurthy, "Policy Gradient using Weak Derivatives for Reinforcement Learning," *U.S. Army Research Laboratory/ Cornell University-Technical Report, 2019.*, https://koppel.netlify.com/assets/papers/2019_report_sujay_etal.pdf.

[28] V. I. Bogachev, *Measure theory*. Springer Science & Business Media, 2007, vol. 1.

[29] K. Hinderer, "Lipschitz continuity of value functions in Markovian decision processes," *Mathematical Methods of Operations Research*, vol. 62, no. 1, pp. 3–22, 2005.

[30] O. Hernández-Lerma and J. B. Lasserre, *Markov chains and invariant probabilities*. Birkhäuser, 2012, vol. 211.

[31] D. P. Bertsekas and J. N. Tsitsiklis, "Gradient convergence in gradient methods with errors," *SIAM Journal on Optimization*, vol. 10, no. 3, pp. 627–642, 2000.

[32] N. Bäuerle and U. Rieder, *Markov decision processes with applications to finance*. Springer Science & Business Media, 2011.

[33] A. Benveniste, M. Métivier, and P. Priouret, *Adaptive algorithms and stochastic approximations*. Springer Science & Business Media, 2012, vol. 22.

[34] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[35] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in neural information processing systems*, 2000, pp. 1057–1063.