Regularized Policy Gradients: Direct Variance Reduction in Policy Gradient Estimation

Tingting Zhao TINGTING@TUST.EDU.CN
Tianjin University of Science and Technology, China

Gang Niu GANG@MS.K.U-TOKYO.AC.JP
The University of Tokyo, Japan

Ning Xie NINGXIE@TONGJI.EDU.CN
Tongji University, China

Jucheng Yang JCYANG@TUST.EDU.CN
Tianjin University of Science and Technology, China

Masashi Sugiyama SUGI@K.U-TOKYO.AC.JP
The University of Tokyo, Japan

Abstract

Policy gradient algorithms are widely used in reinforcement learning problems with continuous action spaces, which update the policy parameters along the steepest direction of the expected return. However, large variance of policy gradient estimation often causes instability of policy update. In this paper, we propose to suppress the variance of gradient estimation by directly employing the variance of policy gradients as a regularizer. Through experiments, we demonstrate that the proposed variance-regularization technique combined with parameter-based exploration and baseline subtraction provides more reliable policy updates than non-regularized counterparts.

Keywords: Reinforcement learning; Policy gradients; Variance-regularization.

1. Introduction

Reinforcement learning (RL), which studies how an agent ought to act in an unknown environment so as to maximize the cumulative rewards (Sutton and Barto, 1998), is a powerful machine learning paradigm for sequential decision making. Two popular branches of RL research are: *policy iteration* and *policy search*.

Policy iteration iteratively goes through the value function estimation based on the current policy and policy updates based on the estimated value function (Kaelbling et al., 1996). It was demonstrated to work well in many real-world problems, particularly in those with *discrete* states and actions (Tesauro, 1994; Abe et al., 2010). Policy iteration can also handle continuous states via function approximation (Lagoudakis and Parr, 2003). However, dealing with continuous actions is not straightforward due to the difficulty of finding a maximizer of the value function with respect to continuous actions.

Policy search can overcome the above limitation by directly learning the policy parameter without using the value function. Consequently, it has been widely applied to more complex problems with continuous action spaces (Ng and Jordan, 2000; Abbeel et al., 2007; Xie et al., 2012). Among policy search methods, the *policy gradient* (PG) method (Williams, 1992; Sehnke et al., 2010) demonstrated remarkable successes in robotics (Peters and Schaal, 2006; Deisenroth et al., 2013; Sugimoto et al., 2014).

Nevertheless, the PG method still has a weakness that estimation of policy gradients can be unreliable in practice. In order to reduce the variance of gradient estimation, useful techniques have been proposed, including, the natural gradient (Kakade, 2002; Peters and Schaal, 2008), parameter-based exploration (Sehnke et al., 2010; Miyamae et al., 2010), and the optimal baseline (Weaver and Tao, 2001; Greensmith et al., 2004; Zhao et al., 2012). While all of these methods were shown to stabilize the policy update to some extent, none of them directly take the variance of gradient estimates into account in the objective. Thus, further stabilization is necessary to efficiently solve challenging RL problems.

In this paper, we explore a more explicit way for further variance reduction, by directly employing the variance of policy gradients as a regularizer. Our idea is motivated by risk-sensitive RL (Mihatsch and Neuneier, 2002), which includes an additional risk term, i.e., the variance of the return, in the objective; for risk-sensitive RL, see also return density estimation (Morimura et al., 2010), the risk-sensitive PG method (Tamar et al., 2012), the actor-critic method (Prashanth and Ghavamzadeh, 2013), and the temporal difference method (Tamar et al., 2013). However, our goal is not to consider the risk, but to improve the stability of PG algorithms so that policy updates can be performed more reliably. Thus, we design a new framework for PG algorithms by directly incorporating the variance of policy gradients in the objective function. The proposed variance-regularized framework can naturally increase the expected return, but also reduce the variance of gradient estimates.

In practice, we combine our variance-regularization technique with parameter-based exploration (Sehnke et al., 2010; Miyamae et al., 2010) and the optimal baseline (Weaver and Tao, 2001; Greensmith et al., 2004) for further variance reduction. More specifically, we implement a state-of-the-art PG method, policy gradient with parameter based exploration (PGPE) with optimal baseline subtraction (Zhao et al., 2012), in our proposed variance-regularized framework. The effectiveness of our proposed approach is demonstrated through experiments using benchmark and real-world problems.

2. Framework and Formulation

In this section, we introduce the framework of policy gradients and review various stabilization techniques.

2.1. Policy Gradient Algorithm

Let us consider a *Markov decision process* (MDP) specified by

$$(\mathcal{S}, \mathcal{A}, P_T, P_I, r, \gamma),$$

where

- $\mathcal{S}$ is a set of (possibly continuous) states,
- $\mathcal{A}$ is a set of (possibly continuous) actions,

- $P_T(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t)$ is the transition probability density from current state $\boldsymbol{s}_t$ to next state $\boldsymbol{s}_{t+1}$ when action a_t is taken,
- $P_I(\boldsymbol{s})$ is the probability density of initial states $\boldsymbol{s}_1$,
- $r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})$ is an immediate reward for transition from current state $\boldsymbol{s}_t$ to next state $\boldsymbol{s}_{t+1}$ by taking action a_t,
- $0 < \gamma \leq 1$ is the discount factor for future rewards.

Let $\pi(a|\boldsymbol{s}, \boldsymbol{\theta})$ be a stochastic policy of an agent parameterized by $\boldsymbol{\theta}$, which is the conditional probability density of taking action $a \in \mathcal{A}$ at state $s \in \mathcal{S}$. Let $h := [\boldsymbol{s}_1, a_1, \ldots, \boldsymbol{s}_T, a_T]$ be a trajectory, which is a sequence of states and actions with finite length T. The discounted cumulative rewards along h, called the *return*, is given by $R(h) := \sum_{t=1}^{T} \gamma^{t-1} r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})$. The expected return for policy parameter $\boldsymbol{\theta}$ is defined by $J(\boldsymbol{\theta}) := \mathbb{E}_{\boldsymbol{\theta}}[R(h)]$, where $\mathbb{E}_{\boldsymbol{\theta}}$ denotes the expectation over h with respect to $p(h|\boldsymbol{\theta}) = p(\boldsymbol{s}_1) \prod_{t=1}^{T} p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t) \pi(a_t|\boldsymbol{s}_t, \boldsymbol{\theta})$. The goal of RL is to find optimal policy parameter $\boldsymbol{\theta}$ that maximizes the expected return $J(\boldsymbol{\theta})$:

$$\boldsymbol{\theta}^* := \arg\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta}). \tag{1}$$

In the policy gradient method (Williams, 1992), the policy parameters are learned via the gradient ascent: $\boldsymbol{\theta} \longleftarrow \boldsymbol{\theta} + \varepsilon \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$, where $\varepsilon > 0$ is the learning rate, and the gradient of the expected return with respect to the policy parameter is given by

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\theta}}[R(h) \nabla_{\boldsymbol{\theta}} \log p(h|\boldsymbol{\theta})]. \tag{2}$$

However, this algorithm is often unreliable due to the large variance of the policy gradient estimator particularly when the trajectory length T is large, which is caused by the stochasticity of policies (Zhao et al., 2012; Deisenroth et al., 2013).

2.2. Parameter-Based Exploration

To mitigate the large variance problem, an alternative method called *policy gradients with parameter based exploration* (PGPE) was proposed (Sehnke et al., 2010). The basic idea of PGPE is to use a deterministic policy and introduce stochasticity by drawing parameters from a prior distribution. More specifically, parameters are sampled from the prior distribution at the start of each trajectory, and thereafter the controller is deterministic. Thanks to this per-trajectory formulation, the variance of the gradient estimator in PGPE does not increase with respect to trajectory length T (Zhao et al., 2012).

PGPE uses a deterministic policy with typically a linear architecture: $a = \boldsymbol{\theta}^\top \boldsymbol{\phi}(\boldsymbol{s})$, where $\boldsymbol{\phi}(\boldsymbol{s})$ is an ℓ-dimensional basis function vector, and $^\top$ denotes the transpose. The policy parameter $\boldsymbol{\theta}$ is drawn from a prior distribution $p(\boldsymbol{\theta}|\boldsymbol{\rho})$ with hyper-parameter $\boldsymbol{\rho}$.

The expected return in the PGPE formulation is defined in terms of expectations over both h and $\boldsymbol{\theta}$ as a function of hyper-parameter $\boldsymbol{\rho}$: $\mathcal{J}(\boldsymbol{\rho}) := \mathbb{E}_{\boldsymbol{\rho}}[R(h)]$, where $\mathbb{E}_{\boldsymbol{\rho}}$ denotes the expectation with respect to $p(h, \boldsymbol{\theta}|\boldsymbol{\rho}) = p(h|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\rho})$. In PGPE, the hyper-parameter $\boldsymbol{\rho}$ is optimized so as to maximize $\mathcal{J}(\boldsymbol{\rho})$, i.e., the optimal hyper-parameter $\boldsymbol{\rho}^*$ is given by

$\boldsymbol{\rho}^* := \arg\max_{\boldsymbol{\rho}} \mathcal{J}(\boldsymbol{\rho})$. The gradient method is used to find $\boldsymbol{\rho}^*$: $\boldsymbol{\rho} \longleftarrow \boldsymbol{\rho} + \varepsilon \nabla_{\boldsymbol{\rho}} \mathcal{J}(\boldsymbol{\rho})$, where the gradient of the expected return with respect to $\boldsymbol{\rho}$ is given by

$$\nabla_{\boldsymbol{\rho}} \mathcal{J}(\boldsymbol{\rho}) = \mathbb{E}_{\boldsymbol{\rho}}[R(h) \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})]. \tag{3}$$

In practice, the expectation included above is estimated from data samples collected at the current iteration: $D = \{(\boldsymbol{\theta}_n, h_n)\}_{n=1}^N$, where each trajectory sample h_n is drawn independently from $p(h|\boldsymbol{\theta}_n)$ and parameter $\boldsymbol{\theta}_n$ is drawn from $p(\boldsymbol{\theta}_n|\boldsymbol{\rho})$. Then the gradient of the expected return can be estimated by using collected samples D as

$$\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}(\boldsymbol{\rho}) = \frac{1}{N} \sum_{n=1}^{N} \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}_n|\boldsymbol{\rho}) R(h_n). \tag{4}$$

The Gaussian distribution with hyper-parameter $\boldsymbol{\rho} = (\boldsymbol{\eta}, \boldsymbol{\tau})$ is often used as a policy prior (Sehnke et al., 2010), where $\boldsymbol{\eta}$ is the mean vector and $\boldsymbol{\tau}$ is the vector consisting of the standard deviation in each element:

$$p(\theta_i|\rho_i) = \frac{1}{\tau_i \sqrt{2\pi}} \exp\left(-\frac{(\theta_i - \eta_i)^2}{2\tau_i^2} \right). \tag{5}$$

Then the derivatives of $\log p(\boldsymbol{\theta}|\boldsymbol{\rho})$ with respect to η_i and τ_i are given as follows:

$$\begin{aligned} \nabla_{\eta_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}) &= \frac{\theta_i - \eta_i}{\tau_i^2}, \\ \nabla_{\tau_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}) &= \frac{(\theta_i - \eta_i)^2 - \tau_i^2}{\tau_i^3}. \end{aligned}$$

2.3. Baseline Subtraction

The gradient estimation can be further stabilized by subtracting a *baseline* b (Zhao et al., 2012):

$$\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}^b(\boldsymbol{\rho}) = \frac{1}{N} \sum_{n=1}^{N} (R(h_n) - b) \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}_n|\boldsymbol{\rho}).$$

The optimal baseline is given by

$$b^* = \arg\min_b \mathbf{Var}_{\boldsymbol{\rho}}[\nabla_{\boldsymbol{\rho}} \widehat{\mathcal{J}}^b(\boldsymbol{\rho})], \tag{6}$$

where $\mathbf{Var}_{\boldsymbol{\rho}}$ denotes the trace of the covariance matrix, i.e., for $\boldsymbol{A} = (A_1, \ldots, A_l)^\top$,

$$\begin{aligned} \mathbf{Var}_{\boldsymbol{\rho}}[\boldsymbol{A}] &= \mathrm{tr}(\mathbb{E}_{\boldsymbol{\rho}}[(\boldsymbol{A} - \mathbb{E}_{\boldsymbol{\rho}}[\boldsymbol{A}])(\boldsymbol{A} - \mathbb{E}_{\boldsymbol{\rho}}[\boldsymbol{A}])^T]) \\ &= \sum_{i=1}^{l} \mathbb{E}_{\boldsymbol{\rho}}[(A_i - \mathbb{E}_{\boldsymbol{\rho}}[A_i])^2]. \end{aligned} \tag{7}$$

Solving Eq. (6) for b gives the optimal baseline for PGPE:

$$b^* = \frac{\mathbb{E}_{\boldsymbol{\rho}}[R(h) \|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2]}{\mathbb{E}_{\boldsymbol{\rho}}[\|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|^2]},$$

which is approximated from collected samples D as

$$\widehat{b} = \frac{\frac{1}{N} \sum_{n=1}^{N} R(h_n) \|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}_n|\boldsymbol{\rho})\|^2}{\frac{1}{N} \sum_{n=1}^{N} \|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}_n|\boldsymbol{\rho})\|^2}. \tag{8}$$

3. Variance-Regularization for Policy Gradients

PGPE with baseline subtraction was demonstrated to provide the state-of-the-art performance (Zhao et al., 2012). However, it can still be unstable in challenging RL problems. In this section, we propose a more explicit method to reduce the variance of the gradient estimator.

Our basic idea is to regularize the objective function as

$$\Phi(\boldsymbol{\rho}) = \mathcal{J}(\boldsymbol{\rho}) - \lambda V(\boldsymbol{\rho}), \tag{9}$$

where $\lambda \geq 0$ is the regularization parameter and $V(\boldsymbol{\rho})$ is the variance of the gradient of the return with respect to $p(h, \boldsymbol{\theta}|\boldsymbol{\rho})$:

$$V(\boldsymbol{\rho}) = \mathbf{Var}_{\boldsymbol{\rho}}[R(h)\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})].$$

Then the gradient update rule is given as

$$\boldsymbol{\rho} \longleftarrow \boldsymbol{\rho} + \varepsilon \nabla_{\boldsymbol{\rho}} \Phi(\boldsymbol{\rho}), \tag{10}$$

where the gradient is given by

$$\nabla_{\boldsymbol{\rho}} \Phi(\boldsymbol{\rho}) = \nabla_{\boldsymbol{\rho}} \mathcal{J}(\boldsymbol{\rho}) - \lambda \nabla_{\boldsymbol{\rho}} V(\boldsymbol{\rho}).$$

According to Eq. (7), the variance of the gradient of the return can be expressed as

$$\begin{aligned} V(\boldsymbol{\rho}) &= \mathbf{Var}_{p(h,\boldsymbol{\theta}|\boldsymbol{\rho})}[R(h)\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})] \\ &= \sum_{i=1}^{l} \mathbb{E}_{\boldsymbol{\rho}}[(R(h)\nabla_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2] - \sum_{i=1}^{l} (\mathbb{E}_{\boldsymbol{\rho}}[R(h)\nabla_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})])^2 \\ &= \sum_{i=1}^{l} \mathbb{E}_{\boldsymbol{\rho}}[(R(h)\nabla_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2] - \sum_{i=1}^{l} (\nabla_{\rho_i} \mathcal{J}(\boldsymbol{\rho}))^2. \end{aligned}$$

The gradient of $V(\boldsymbol{\rho})$ with respect to $\boldsymbol{\rho}$ is the vector given by

$$\nabla_{\boldsymbol{\rho}} V(\boldsymbol{\rho}) = (\nabla_{\rho_1} V(\boldsymbol{\rho}), \ldots, \nabla_{\rho_l} V(\boldsymbol{\rho}))^\top.$$

Below, we derive the ith element of the gradient of the variance regularization term:

$$\begin{aligned} &\nabla_{\rho_i} V(\boldsymbol{\rho}) \\ =&\nabla_{\rho_i} \{\mathbb{E}_{\boldsymbol{\rho}}[(R(h)\nabla_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2] - (\nabla_{\rho_i} \mathcal{J}(\boldsymbol{\rho}))^2\} \\ =&\nabla_{\rho_i} \iint (R(h))^2 (\nabla_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2 p(h, \theta_i|\rho_i) \mathrm{d}h \mathrm{d}\theta_i - \nabla_{\rho_i} (\nabla_{\rho_i} \mathcal{J}(\boldsymbol{\rho}))^2 \\ =&\iint (R(h))^2 [2\nabla_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}) \nabla^2_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}) p(h, \theta_i|\rho_i) + (\nabla_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^3 p(h, \theta_i|\rho_i)] \mathrm{d}h \mathrm{d}\theta_i \\ &- 2\nabla_{\rho_i} \mathcal{J}(\boldsymbol{\rho}) \nabla^2_{\rho_i} \mathcal{J}(\boldsymbol{\rho}) \\ =&\mathbb{E}_{\rho_i}[(R(h))^2 ((\nabla_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^3 + 2\nabla_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}) \nabla^2_{\rho_i} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}))] - 2\nabla_{\rho_i} \mathcal{J}(\boldsymbol{\rho}) \nabla^2_{\rho_i} \mathcal{J}(\boldsymbol{\rho}), \end{aligned}$$

where $\nabla_{\rho_i}\mathcal{J}(\boldsymbol{\rho})$ is the gradient of the expected return with respect to ρ_i given by

$$\nabla_{\rho_i}\mathcal{J}(\boldsymbol{\rho}) = \mathbb{E}_{\boldsymbol{\rho}}[\nabla_{\rho_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho})R(h)], \tag{11}$$

$\nabla^2_{\rho_i}\mathcal{J}(\boldsymbol{\rho})$ is the second-order derivative of expected return with respect to ρ_i given by

$$\nabla^2_{\rho_i}\mathcal{J}(\boldsymbol{\rho}) = \mathbb{E}_{\boldsymbol{\rho}}[R(h)((\nabla_{\rho_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho}))^2 + \nabla^2_{\rho_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho}))], \tag{12}$$

and $\nabla^2_{\rho_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho})$ is the second-order derivative of $\log p(\boldsymbol{\theta}|\boldsymbol{\rho})$ with respect to ρ_i.

Finally, the expectations are approximated by sample averages as

$$\nabla_{\rho_i}\widehat{\Phi}(\boldsymbol{\rho}) = \nabla_{\rho_i}\widehat{\mathcal{J}}(\boldsymbol{\rho}) - \lambda\nabla_{\rho_i}\widehat{V}(\boldsymbol{\rho}),$$

where

$$\begin{aligned}
\nabla_{\rho_i}\widehat{V}(\boldsymbol{\rho}) &= \frac{1}{N}\sum_{n=1}^{N}[(R(h_n))^2((\nabla_{\rho_i}\log p(\boldsymbol{\theta}_n|\boldsymbol{\rho}))^3 + 2\nabla_{\rho_i}\log p(\boldsymbol{\theta}_n|\boldsymbol{\rho})\nabla^2_{\rho_i}\log p(\boldsymbol{\theta}_n|\boldsymbol{\rho}))] \\
&\quad - 2\nabla_{\rho_i}\widehat{\mathcal{J}}(\boldsymbol{\rho})\nabla^2_{\rho_i}\widehat{\mathcal{J}}(\boldsymbol{\rho}), \\
\nabla_{\rho_i}\widehat{\mathcal{J}}(\boldsymbol{\rho}) &= \frac{1}{N}\sum_{n=1}^{N}R(h_n)\nabla_{\rho_i}\log p(\boldsymbol{\theta}_n|\boldsymbol{\rho}), \\
\nabla^2_{\rho_i}\widehat{\mathcal{J}}(\boldsymbol{\rho}) &= \frac{1}{N}\sum_{n=1}^{N}R(h_n)[(\nabla_{\rho_i}\log p(\boldsymbol{\theta}_n|\boldsymbol{\rho}))^2 + \nabla^2_{\rho_i}\log p(\boldsymbol{\theta}_n|\boldsymbol{\rho})].
\end{aligned}$$

Note that, for the Gaussian prior expressed in Eq. (5), the second-order derivatives of $\log p(\boldsymbol{\theta}|\boldsymbol{\rho})$ with respect to η_i and τ_i are given by

$$\begin{aligned}
\nabla^2_{\eta_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho}) &= -\frac{1}{\tau_i^2}, \\
\nabla^2_{\tau_i}\log p(\boldsymbol{\theta}|\boldsymbol{\rho}) &= \frac{\tau_i^2 - 3(\theta_i - \eta_i)^2}{\tau_i^4}.
\end{aligned}$$

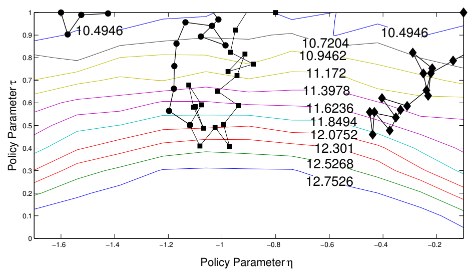
In practice, we may further subtract the baseline $\widehat{b}$ from $R(h_n)$ in the above gradient estimator, which will be experimentally demonstrated to achieve the best performance in the next section.

4. Experimental Results

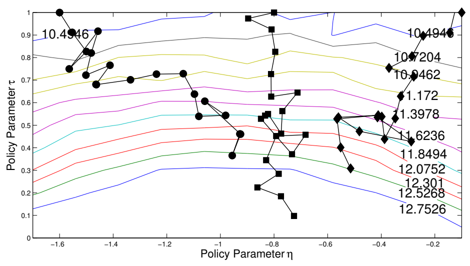
In this section, we experimentally evaluate the usefulness of our proposed method.

4.1. Illustration

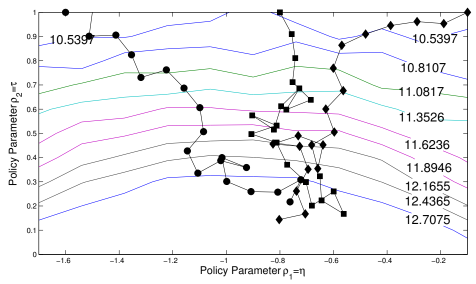
First, we evaluate our proposed variance-regularized PGPE method through an illustrative experiment. The state space is one-dimensional and continuous, and the initial state is randomly chosen from the standard normal distribution. The action space is also one-dimensional and continuous. The state transition function is defined as $s_{t+1} = s_t + a_t + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 0.3^2)$ is the noise and $\mathcal{N}(\mu, \sigma^2)$ denotes the Gaussian distribution with mean μ and variance σ^2. The reward function is defined as $r = \exp(-s^2/2 - a^2/2) + 1$. The following 4 methods are compared:



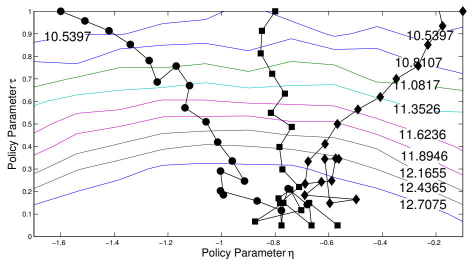
(a) PGPE



(b) R-PGPE



(c) $\mathrm{PGPE}_{\mathrm{OB}}$



(d) R-$\mathrm{PGPE}_{\mathrm{OB}}$

Figure 1: Trajectories of parameter updates over 20 iterations.

- **PGPE:** The plain PGPE method.
- **$\mathbf{PGPE_{OB}}$:** PGPE with the optimal baseline.
- **R-PGPE:** PGPE with our proposed variance regularization.
- **R-$\mathbf{PGPE_{OB}}$:** $\mathrm{PGPE}_{\mathrm{OB}}$ with the proposed variance regularization.

We use the linear policy $a = \theta s$ for all methods. The initial prior mean η is chosen randomly following the standard normal distribution, and the initial prior deviation τ is set at 1. The discount factor is set at $\gamma = 0.99$, and the trajectory length T is set at 10. The regularization weight λ in the variance-regularized methods (R-PGPE and R-$\mathrm{PGPE}_{\mathrm{OB}}$) is initially set at $\lambda_0 = 10^{-5}$. Then λ is increased by factor of 10 if policy search is making progress, and otherwise λ is decreased by factor of 10. The range of λ is kept in $[10^{-2}, 10^{-8}]$ during the policy learning process. Note that higher regularization causes the next optimization to stay closer to the current parameters. Thus, by adaptively adjusting λ, we can control the optimization close to the regions where the policy is better, or deviate the regions where the policy does not make progress. The learning rate for PGPE and $\mathrm{PGPE}_{\mathrm{OB}}$ is set at $\varepsilon = 1/\|\nabla_\rho \hat{\mathcal{J}}(\boldsymbol{\rho})\|$, and the learning rate for R-PGPE and R-$\mathrm{PGPE}_{\mathrm{OB}}$ is set at $\varepsilon = 1/\|\nabla_\rho \hat{\Phi}(\boldsymbol{\rho})\|$.

Parameter Trajectories First, we investigate how policy parameters change over 20 iterations. Here we observe three different parameter update trajectories by setting three different starting points, where three different initial mean parameters are set at: $\eta = -1.6$, $\eta = -0.8$, and $\eta = -0.1$, respectively, and the initial deviation parameter is set at $\tau = 1$. We

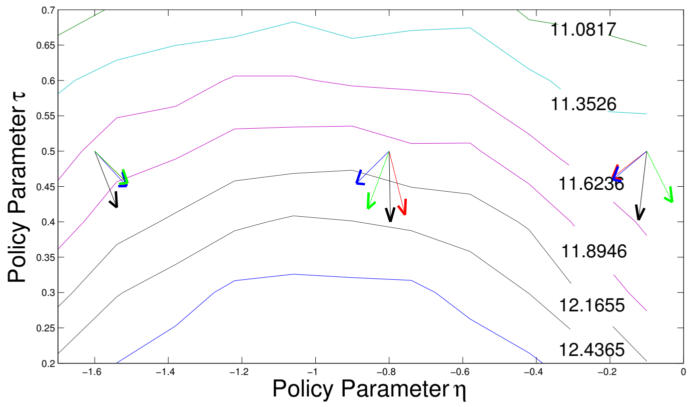


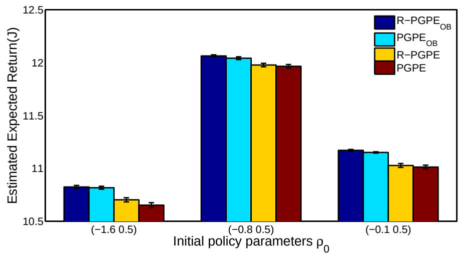
Figure 2: Illustration of one step policy parameter updates by 4 compared methods: the green arrows denote PGPE, the black arrows denote R-PGPE, the blue arrows denote PGPE$_{\mathrm{OB}}$, and the red arrows denote R-PGPE$_{\mathrm{OB}}$.

collect $N = 10$ trajectory samples at each policy update iteration for this demonstration. Fig. 1 shows the contour of the expected return, where the maximum return is located at the middle bottom.

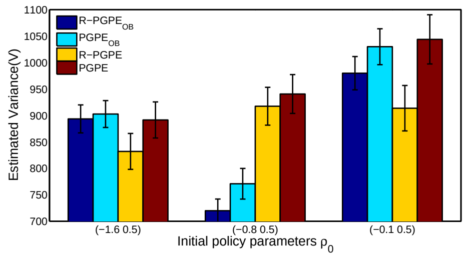
From Fig. 1(*a*), we can see that plain PGPE cannot update the parameters to the large return areas within 20 iterations, R-PGPE at least leads one trajectory to the middle bottom as shown in Fig. 1(*b*), which indicates the usefulness of our proposed variance-regularization. On the other hand, Fig. 1(*c*) shows that PGPE$_{\mathrm{OB}}$ can find fairly reliable update directions, but with some detours. Fig. 1(*d*) shows that R-PGPE$_{\mathrm{OB}}$ gives stable and reliable parameter update directions, where three trajectories converge to the middle bottom within 20 iterations. This result illustrates the rapid convergence and reliable parameter updates by our proposed variance-regularization combined with the optimal baseline.

Evaluation of Updated Parameters Next, we illustrate the expected return $\mathcal{J}(\boldsymbol{\rho})$ and the variance of the gradient of return $V(\boldsymbol{\rho})$ in the objective function with the updated policy parameters by variance-regularized methods and non-regularized counterparts. In the experiment, we fix the initial policy parameters $\boldsymbol{\rho}_0$ at three different points: $(-1.6, 0.5)$, $(-0.8, 0.5)$, and $(-0.1, 0.5)$, then update the policy parameters with one step by the compared 4 methods. We collect $N = 10$ trajectory samples at each iteration to estimate the gradients, then update parameters. The updated parameters by 4 compared methods are shown in Fig. 2.

The expected return and the variance of the gradient of the return with updated parameters $\boldsymbol{\rho}_1$, i.e., $\mathcal{J}(\boldsymbol{\rho}_1)$ and $V(\boldsymbol{\rho}_1)$ are calculated by 100 newly collected trajectory samples. The numerical results of $\mathcal{J}(\boldsymbol{\rho}_1)$ and $V(\boldsymbol{\rho}_1)$ shown in Fig. 3 are obtained over 100 runs, where the error bars denote standard errors. The results in Fig. 3(*a*) show that the expected return with parameters updated by our proposed variance-regularized methods is larger than non-regularized methods, i.e., $\boldsymbol{\rho}_1$ learned by R-PGPE$_{\mathrm{OB}}$ provides larger $\mathcal{J}(\boldsymbol{\rho}_1)$ than PGPE$_{\mathrm{OB}}$, and also $\mathcal{J}(\boldsymbol{\rho}_1)$ obtained by R-PGPE is larger than plain PGPE. On the other



(a) Estimated expected return $\mathcal{J}(\boldsymbol{\rho})$



(b) Estimated variance of the gradient of return $V(\boldsymbol{\rho})$

Figure 3: Average estimated expected returns and estimated variance of policy gradients of the updated policy parameters over 100 runs. Error bars denote standard errors.

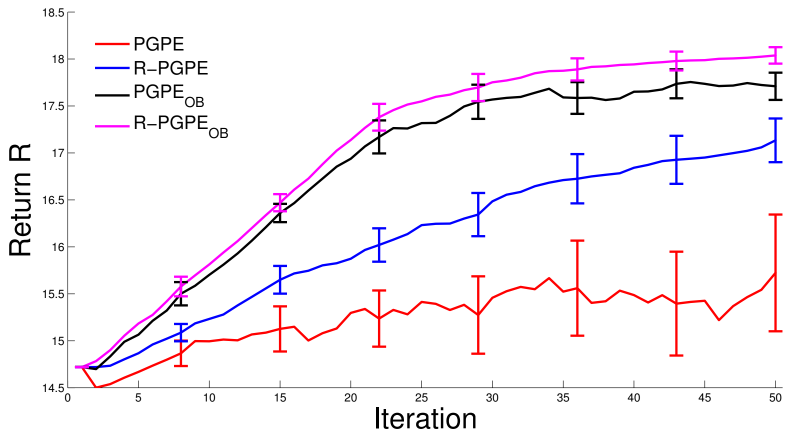


Figure 4: Average expected returns over 20 runs for the illustrative data. Error bars denote standard errors.

hand, results in Fig. 3(b) show that the variance of the gradient of return $V(\boldsymbol{\rho}_1)$ with parameters learned by variance-regularized methods is smaller than non-regularized methods. Overall, this experiment demonstrates that our proposed variance-regularized methods can lead the learned parameters to the regions with large returns but small variance of gradients, which is consistent with our motivation.

Performance of Learned Policies The performance of each method is measured by the average return over 20 runs. In each run, policy parameters are updated for 50 iterations. At each iteration, we collect $N = 2$ trajectory samples to estimate the gradient of the objective function. The expected return at each run is computed over 100 newly-drawn test episodic samples (which are not used for policy learning).

The results are summarized in Fig. 4, showing that our proposed variance-regularized methods outperform the plain counterparts, i.e., R-PGPE outperforms the plain PGPE and R-PGPE$_{\mathrm{OB}}$ outperforms PGPE$_{\mathrm{OB}}$. PGPE$_{\mathrm{OB}}$ works better than PGPE, which demonstrates

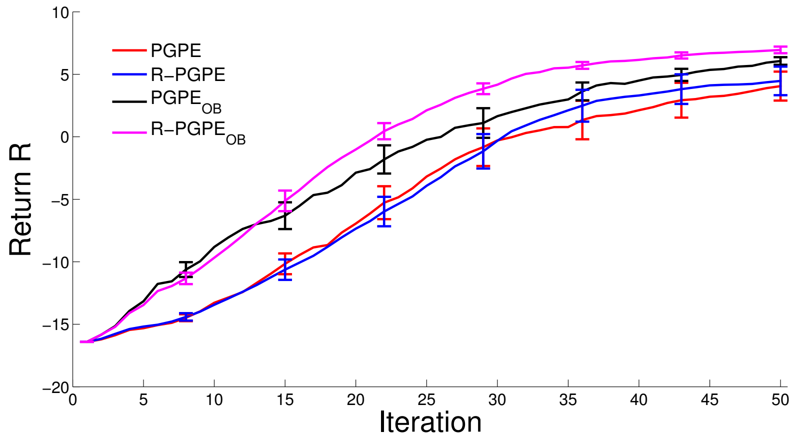


Figure 5: Average expected returns over 20 runs for the mountain car task. Error bars denote standard errors.

the advantage of the baseline subtraction technique. This well agrees with the results shown in Zhao et al. (2012). The combination of the baseline subtraction technique with our proposed variance-regularization achieves the best performance.

4.2. Mountain Car

Next, we evaluate our proposed method on the standard *mountain car* task, whose goal is to guide the car to the top of the right-hill. The landscape of the two hills is described by $\sin(3x)$.

The state space $\mathcal{S}$ is two-dimensional and continuous, which consists of the horizontal position $x[m] \in [-1.2, 0.5]$ and the velocity $\dot{x}[m/s] \in [-1.5, 1.5]$, i.e., $\boldsymbol{s} = (x, \dot{x})^\top$. This is non-linearly transformed to a feature space via a basis function vector $\boldsymbol{\phi}(\boldsymbol{s})$. We use 12 Gaussian kernels with mean $\boldsymbol{c}$ and unit standard deviation as the basis functions, $\boldsymbol{\phi}(\boldsymbol{s}) = \exp\left(-\frac{\|\boldsymbol{s}-\boldsymbol{c}\|^2}{2}\right)$, where the kernel centers $\boldsymbol{c}$ are distributed over the following grid points: $\{-1.2, -0.35, 0.5\} \times \{-1.5, -0.5, 0.5, 1.5\}$. The action space $\mathcal{A}$ is one-dimensional and continuous, which corresponds to the force applied to the car. Note that the force of the car is not strong enough to climb up the slope to directly reach the goal. The dynamics of the car are given by $x_{t+1} = x_t + \dot{x}_{t+1}\Delta t$ and $\dot{x}_{t+1} = \dot{x}_t + \left(-9.8w\cos(3x_t) + \frac{a_t}{w} - k\dot{x}_t\right)\Delta t$, where a_t is the action taken at time t. We set the problem parameters as follows: The mass of the car $w = 0.2$[kg], the friction coefficient $k = 0.3$, and the simulation time step $\Delta t = 0.1$[s]. The reward $+1$ is given if the car achieves the goal, i.e., $x_{t+1} \geq 0.45$; otherwise, the reward -1 is given. The initial state of the car is set at the bottom of the mountain with velocity $\dot{x} = 0$. The discount factor is set at $\gamma = 0.95$.

We again compare the same 4 methods: PGPE, PGPE$_{\text{OB}}$, R-PGPE, and R-PGPE$_{\text{OB}}$. The regularization parameter λ is initially set at 10^{-6}, and it is updated in the same way as the previous experiment. Each element of the initial prior mean $\boldsymbol{\eta}$ is chosen randomly following the standard normal distribution, and each element of the initial prior deviation

$\boldsymbol{\tau}$ is set at 1. The learning rate for PGPE and PGPE$_{\mathrm{OB}}$ is set at $\varepsilon = 1/\|\nabla_\rho \hat{\mathcal{J}}(\boldsymbol{\rho})\|$, and for R-PGPE and R-PGPE$_{\mathrm{OB}}$ is set at $\varepsilon = 1/\|\nabla_\rho \hat{\Phi}(\boldsymbol{\rho})\|$.

We investigate average expected returns over 20 trials, where the expected return at each trial is computed over 100 newly-drawn test episodic samples. In each trial, the policy parameters are updated for 50 iterations, and the agent collects $N = 10$ episodic samples with trajectory length $T = 40$ at each iteration.

The experimental results are plotted in Fig. 5, showing that the variance-regularized methods consistently outperform the non-regularized counterparts, i.e., R-PGPE outperforms PGPE and R-PGPE$_{\mathrm{OB}}$ outperforms PGPE$_{\mathrm{OB}}$. The combination of baseline subtraction and variance-regularization, R-PGPE$_{\mathrm{OB}}$, performs the best, demonstrating that our proposed variance-regularized idea is promising.

4.3. Sumi-e Stroke based Rendering System

Finally, we apply our proposed variance-regularized PGPE method to the problem of stroke-based rendering (Hertzmann, 2003) for non-photorealistic rendering in computer graphics. The aim is to learn the optimal policy to control the brush agent to cover the given shapes as smoothly as possible. This task involves high-dimensional and complex decision making, so it is much more challenging than standard RL benchmark tasks such as the mountain car.

We essentially follow the PG formulation for this stroke based rendering system proposed in Xie et al. (2012), which is briefly explained below. The state space is expressed by six features $\boldsymbol{s} = (\omega, \phi, d, \kappa_1, \kappa_2, l)^\top$, where $\omega \in (-\pi, \pi]$ denotes the angle of the velocity vector, $\phi \in (-\pi, \pi]$ means the heading direction of the footprint relative to the medial axis of the drawing shape, $d \in [-2, 2]$ is the ratio of the offset distance from the center of the footprint to the nearest point on the medial axis over the radius of the footprint, κ_1 and $\kappa_2 \in (-1, 1)$ provide the relative curvatures of the nearest current point and the next point on the medial axis, and $l \in \{0, 1\}$ is the binary signal of the reverse drawing or not. The action space is defined as a 3-dimensional vector, including lifting motion, dragging motion, and pushing motion. Therefore, we deal with multi-dimensional action spaces in this task. The reward measures the quality of the agent's movement after taking an action, i.e., the smoother the brush stroke is, the higher the reward is. If the brush is blocked by a boundary or the brush is going backward to a region that has already been covered by previous footprints, the reward is 0; otherwise, the reward is given by

$$r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) = \frac{1 + |\kappa_1(t)| + |\kappa_2(t)|}{E^{(t)}_{\mathrm{location}} + E^{(t)}_{\mathrm{posture}}},$$

where the curvature $|\kappa_1(t)| + |\kappa_2(t)|$ affects the difficulty of the current shape; $E^{(t)}_{\mathrm{location}}$ indicates whether the agent moves out the region or moves backward from the correct direction; $E^{(t)}_{\mathrm{posture}}$ measures whether the agent moves smoothly.

The advantage of the optimal baseline over plain PGPE has been investigated throughly in the previous research by Zhao et al. (2012), thus, we compare two best performed methods on this real-world problem: PGPE$_{\mathrm{OB}}$ and R-PGPE$_{\mathrm{OB}}$. The regularization parameter λ is initially set at 10^{-6}, and it is updated in the same way as the previous experiment. Each element of the initial prior mean $\boldsymbol{\eta}$ is set at 0, and each element of the initial prior deviation

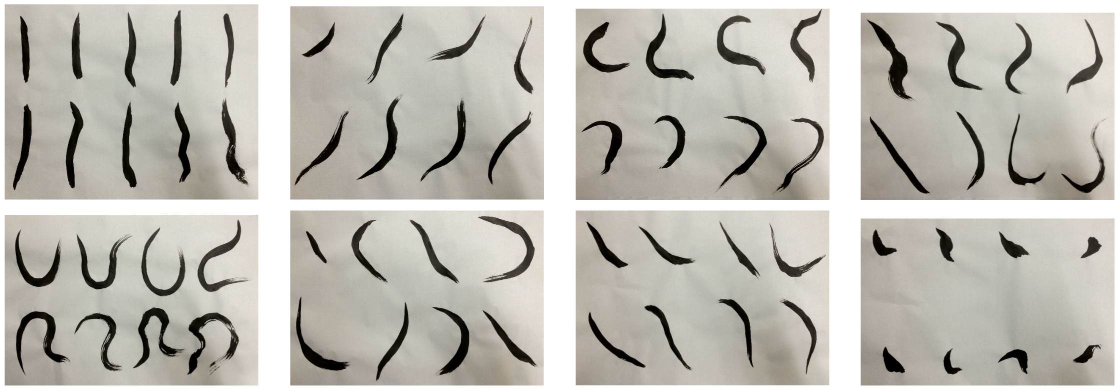


Figure 6: Samples of real strokes in our training data set.

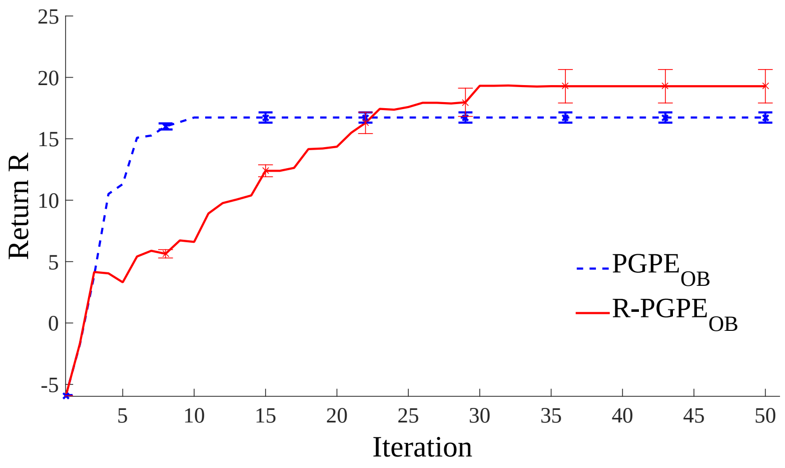
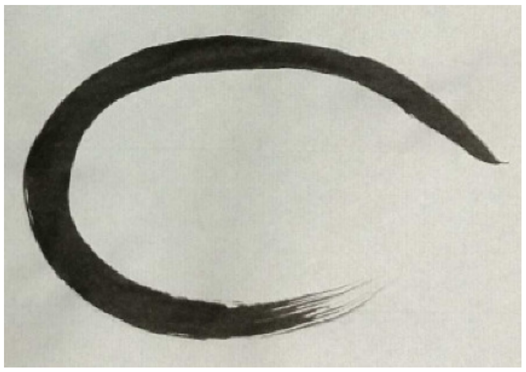


Figure 7: Average expected returns over 20 runs for stroke based rendering. Error bars show standard errors.

$\boldsymbol{\tau}$ is set at 1. The learning rate for PGPE$_{\mathrm{OB}}$ is set at $\varepsilon = 1/\|\nabla_\rho \hat{\mathcal{J}}(\boldsymbol{\rho})\|$, and for R-PGPE$_{\mathrm{OB}}$ is set at $\varepsilon = 1/\|\nabla_\rho \hat{\Phi}(\boldsymbol{\rho})\|$. The discount factor is set at $\gamma = 0.99$.

We investigate average expected returns over 20 trials, where the expected return at each trial is computed over 60 newly-drawn test episodic samples. In each trial, the policy parameters are updated for 50 iterations, and the agent collects $N = 60$ episodic samples with trajectory length $T = 30$ at each iteration. We provide a wide variety of shapes of strokes to the agent as training examples, which are illustrated in Fig. 6. For the test process, we evaluate the performance on the most difficult strokes, i.e., *oblique strokes*. Fig. 7 depicts the progress of return over policy iterations which shows that PGPE$_{\mathrm{OB}}$ is outperformed by R-PGPE$_{\mathrm{OB}}$, and R-PGPE$_{\mathrm{OB}}$ achieves more reliable policy updates. Fig. 8 shows an example of stroke drawing by the policy obtained by R-PGPE$_{\mathrm{OB}}$. This indicates that the proposed variance-regularized method produces smooth and natural brush strokes.


(a)

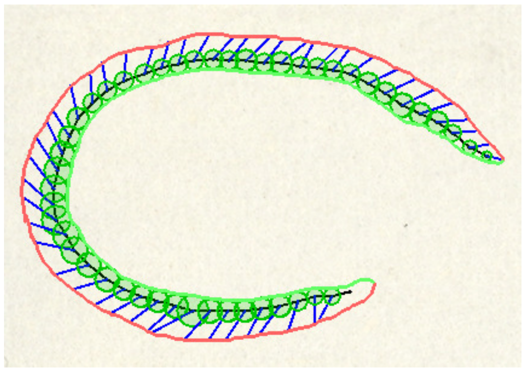

(b)

Figure 8: (a) Example of a simple stroke drawn by the policy learned by R-PGPE$_{\mathrm{OB}}$. (b) Its brush trajectory.

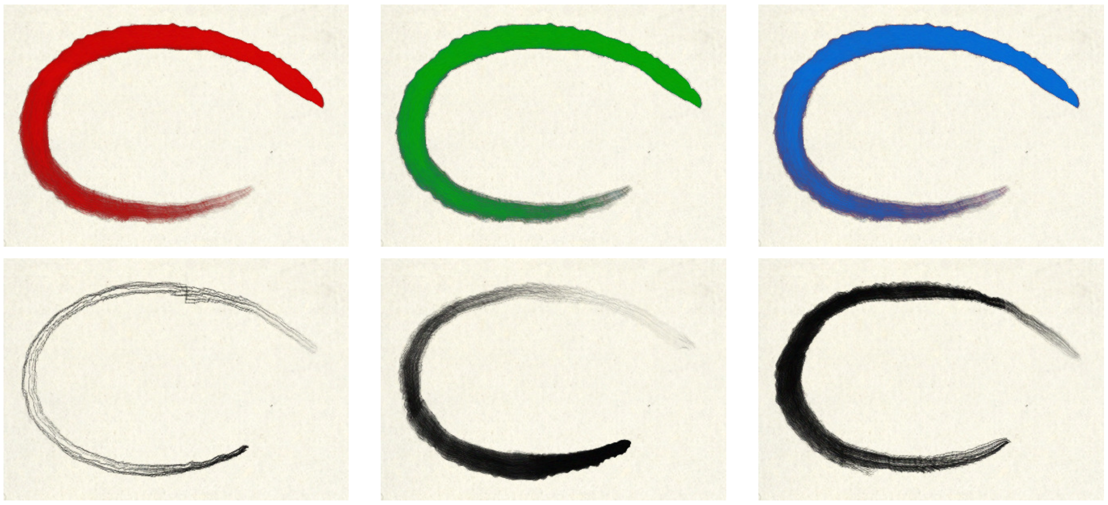


Figure 9: Rendered simple strokes in different colors and textures.

As illustrated in Fig. 9, the learned policy allows non-expert users to produce high-quality brush strokes in their own personal styles, such as their favorite color, texture, and shape. Fig. 10 shows a more complex drawing example by the policy learned by R-PGPE$_{\mathrm{OB}}$: the logo of ACML2015. Fig. 11 shows the result of converting a real photo of sun flowers into a brush painting, which further shows that the policy obtained by R-PGPE$_{\mathrm{OB}}$ can produce smooth and natural brush strokes in various unlearned shapes. This demonstrates the high generalization ability of our learned policy. Also, the resulting drawing well expresses the attractive effect of this sumi-e stroke based rendering application.

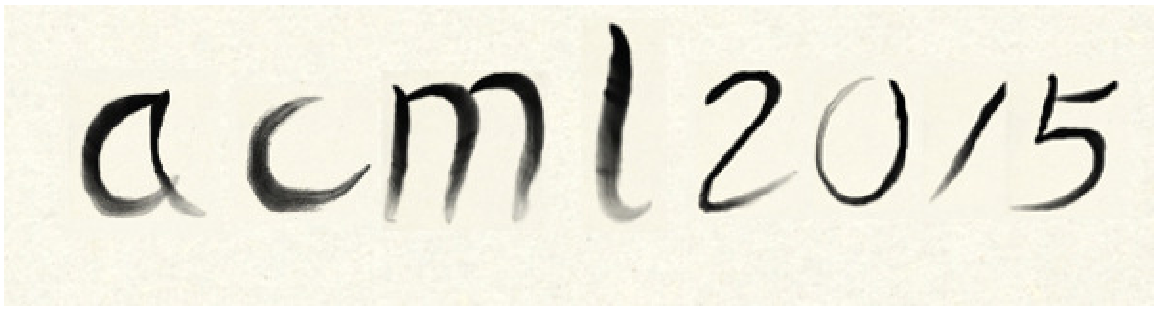


Figure 10: ACML2015 logo rendered by the policy learned by R-PGPE$_{\mathrm{OB}}$.

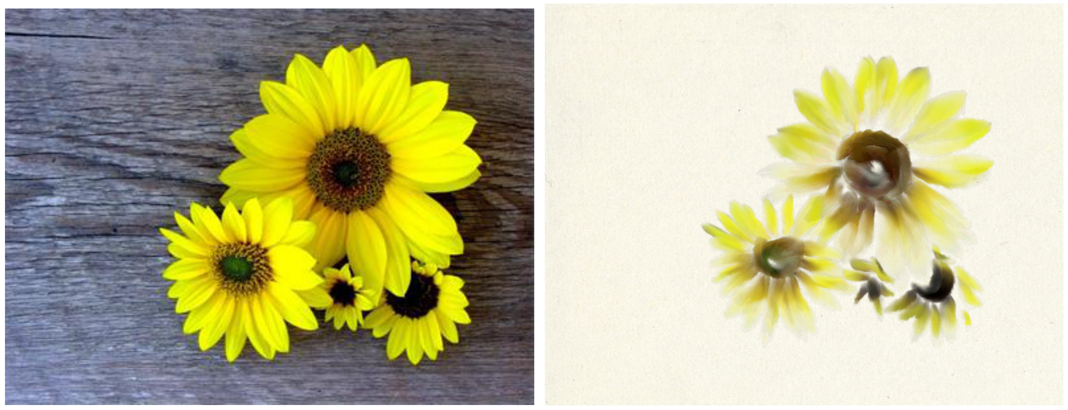


Figure 11: Sun flowers rendered by the policy learned by R-PGPE$_{\mathrm{OB}}$.

5. Conclusion and Discussion

We propose to directly regularize the variance of policy gradients to improve the stability of policy gradient algorithms. Our proposed method, which combines the variance regularization idea with parameter-based exploration and baseline subtraction, was experimentally shown to provide more reliable policy update than non-regularized counterparts. We further demonstrated the usefulness of the proposed method on a real-world digital artist system: sumi-e stroke based rendering.

Through the paper, we focused on the *on-policy* policy gradients scenario. Extending the proposed approach to the *off-policy* scenario is technically straightforward. Theoretical analysis of the proposed variance-regularized PG method, especially in terms of its convergence and sample complexity is important for our future work. Also, the comparison and relation to the other regularized PG method, such as mixing-time regularized PG method (Morimura et al., 2014), is worth to be investigated in the future work.

Acknowledgments

T. Zhao was supported by NSFC 61502339 and SRF for ROCS, SEM. G. Niu was supported by the JST CREST program and 973 Program 2014CB340505. N. Xie was supported by Tongji University Young Scholar Plan (No.2014KJ074). J. Yang was supported by NSFC 61502338. M. Sugiyama was supported by KAKENHI 23120004.

References

P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *In Advances in Neural Information Processing Systems 19*. MIT Press, 2007.

N. Abe, P. Melville, C. Pendus, C. K. Reddy, D. L. Jensen, V. P. Thomas, J. J. Bennett, G. F. Anderson, B. R. Cooley, M. Kowalczyk, M. Domick, and T. Gardinier. Optimizing debt collections using constrained reinforcement learning. In *Proceedings of the 16th ACM SGKDD Conference on Knowledge Discovery and Data Mining*, pages 75–84, 2010.

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

E. Greensmith, P. L. Bartlett, and J. Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004.

A. Hertzmann. A survey of stroke-based rendering. *IEEE Computer Graphics and Applications*, 23:70–81, 2003. doi: 10.1109/MCG.2003.1210867.

L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

S. Kakade. A natural policy gradient. In *Advances in Neural Information Processing Systems 14*, pages 1531–1538, 2002.

M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

O. Mihatsch and R. Neuneier. Risk-sensitive reinforcement learning. *Machine Learning*, 49 (2-3):267–290, 2002.

A. Miyamae, Y. Nagata, I. Ono, and S. Kobayashi. Natural policy gradient methods with parameter-based exploration for control tasks. In *Advances in Neural Information Processing Systems*, volume 2, pages 437–441, 2010.

T. Morimura, M. Sugiyama, H. Kashima, H. Hachiya, and T. Tanaka. Parametric return density estimation for reinforcement learning. In *Proceedings of the 27th International Conference on Machine Learning*, pages 368–375, 2010.

T. Morimura, T. Osogami, and T. Shirai. Mixing-time regularized policy gradient. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

A. Ng and M. Jordan. PEGASUS: A policy search method for large MDPs and POMDPs. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 406–415, 2000.

J. Peters and S. Schaal. Policy gradient methods for robotics. In *Proceedings of the IEEE/RSJ International Conferece on Inatelligent Robots and Systems*, pages 2219–2225, 2006.

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

L. A. Prashanth and M. Ghavamzadeh. Actor-critic algorithms for risk-sensitive mdps. In *Advances in Neural Information Processing Systems 26*, pages 252–260, 2013.

F. Sehnke, C. Osendorfer, T. Rückstiess, A. Graves, J. Peters, and J. Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010.

N. Sugimoto, V. Tangkaratt, T. Wensveen, T. Zhao, M. Sugiyama, and J. Morimoto. Efficient reuse of previous experiences in humanoid motor learning. In *Proceedings of IEEE-RAS International Conference on Humanoid Robots (Humanoids2014)*, pages 554–559, Madrid, Spain, Nov. 18–20 2014.

R. S. Sutton and G. A. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, MA, USA, 1998.

A. Tamar, D. Castro, and S. Mannor. Policy gradients with variance related risk criteria. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 935–942, 2012.

A. Tamar, D. Castro, and S. Mannor. Temporal difference methods for the variance of the reward to go. In *Proceedings of the 29th International Conference on Machine Learning (ICML2013)*, pages 495–503, 2013.

G. Tesauro. TD-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, 6(2):215–219, 1994.

L. Weaver and N. Tao. The optimal reward baseline for gradient-based reinforcement learning. In *Processings of the Seventeeth Conference on Uncertainty in Artificial Intelligence*, pages 538–545, 2001.

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

N. Xie, H. Hachiya, and M. Sugiyama. Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

T. Zhao, H. Hachiya, G. Niu, and M. Sugiyama. Analysis and improvement of policy gradient estimation. *Neural Networks*, 26:118–129, 2012.