SUPERVISED Q-LEARNING CAN BE A STRONG BASELINE FOR CONTINUOUS CONTROL

Hao Sun[*]
University of Cambridge

Ziping Xu
University of Michigan

Meng Fang
University of Liverpool

Bolei Zhou
UCLA

ABSTRACT

Policy gradient (PG) algorithms have been widely used in reinforcement learning (RL). However, PG algorithms rely on exploiting the value function being learned with the first-order update locally, which results in limited sample efficiency. In this work, we propose an alternative method called Zeroth-Order Supervised Policy Improvement (ZOSPI). ZOSPI exploits the estimated value function Q globally while preserving the local exploitation of the PG methods based on zeroth-order policy optimization. This learning paradigm follows Q-learning but overcomes the difficulty of efficiently operating argmax in continuous action space. It finds max-valued action within a small number of samples. The policy learning of ZOSPI has two steps: First, it samples actions and evaluates those actions with a learned value estimator, and then it learns to perform the action with the highest value through supervised learning. We further demonstrate such a supervised learning framework can learn multi-modal policies. Experiments show that ZOSPI achieves competitive results on the continuous control benchmarks with a remarkable sample efficiency.[1]

1 INTRODUCTION

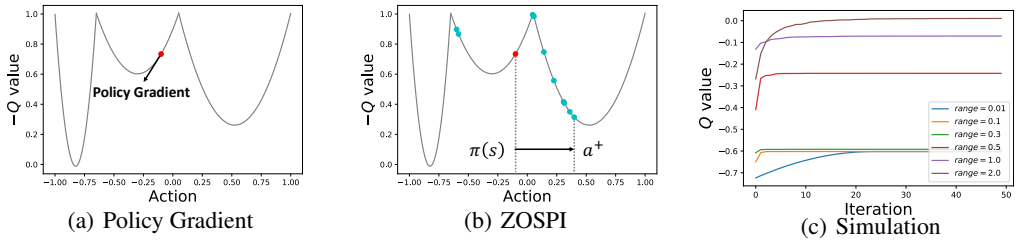
Model-free Reinforcement Learning achieves great successes in many challenging tasks Mnih et al. (2015); Vinyals et al. (2019); Pachocki et al., however one hurdle for its applicability to real-world control problems is the low sample efficiency. To improve the sample efficiency, off-policy methods Degris et al. (2012); Gu et al. (2016); Wang et al. (2016); Lillicrap et al. (2015); Fujimoto et al. (2018a) reuse the experiences generated by previous policies to optimize the current policy, therefore can obtain a much higher sample efficiency than the on-policy methods Schulman et al. (2015; 2017). Alternatively, SAC Haarnoja et al. (2018) improves sample efficiency by conducting more active exploration with maximum entropy regularizer Haarnoja et al. (2017) to the off-policy actor critic Degris et al. (2012); Zhang et al. (2019a).OAC Ciosek et al. (2019) further improves SAC by combining it with the Upper Confidence Bound heuristics Brafman and Tennenholtz (2002) to incentivize more informed exploration. Nonetheless, all of those previous methods rely on Gaussian-parameterized policies and conducts local exploration strategies that simply add noises on the action space. Consequentially, those methods can converge to sub-optimal solutions, as has been shown in (Tessler et al., 2019).

In this work, we explore an alternative approach to the conventional policy gradient paradigm for continuous control. We introduce a zeroth-order method that can be optimized through supervised learning. The proposed method carries out non-local exploration through global value-function exploitation to achieve higher sample efficiency for continuous control tasks. To better exploit the learned value function Q, we propose to search the action space globally for a better target action, which is in contrast to previous policy gradient methods that only utilize the local information of the learned value functions (e.g. the Jacobian matrix Silver et al. (2014)). The key insight is to improve sample efficiency at expense of additional computation, which is acceptable as samples are much more expensive than computations in real-world applications such as Healthcare and Robotics.

The idea behind our work is related to the value-based policy gradient methods Lillicrap et al. (2015); Fujimoto et al. (2018a), where the policy gradient optimization step takes the role of finding a

[*]hs789@cam.ac.uk

[1]Code is made open-sourced at https://anonymous.4open.science/r/Submission6417-285A



(a) Policy Gradient (b) ZOSPI (c) Simulation

Figure 1: (a) Landscape of Q value for a 1-dim continuous control task. Policy gradient methods optimize the policy according to the local information of Q. (b) For the same task, ZOSPI directly updates the predicted actions to the sampled action with the largest Q value. (c) Simulation results, where in each optimization iteration 10 actions are uniformly sampled under different ranges. The reported results are averaged over 100 random seeds. It can be seen that a larger random sample range improves the chance of finding global optima. Similar phenomenon also exist in practice as shown in Appendix A.

well-performing action given a learned state-action value function. In previous work, the policy gradient step tackles the curse of dimensionality for deep Q-learning since it is intractable to directly search for the maximal value in the continuous action space Mnih et al. (2015). Differently, we circumvent searching for the action with the maximal value in the continuous action space by finding the max-valued action within a small set of sampled actions.

A Motivating Example To perform a global exploitation of the learned value function Q, we propose to apply a zeroth-order optimization scheme and update the target policy through supervised learning, which is inspired by works of evolution strategies Salimans et al. (2017); Conti et al. (2018); Mania et al. (2018) that adopt zeroth-order optimizations in the parameter space. Figure 1 shows a motivating example that demonstrates the difference between the zeroth-order and first-order optimizations. Different from the standard policy gradient, combining the zeroth-order optimization with supervised learning forms a new way of policy update. Such an update avoids the local improvement of policy gradient: when policy gradient is applied, the target policy uses policy gradient to adjust its predictions according to the deterministic policy gradient theorem Silver et al. (2014), but such adjustments can only lead to local improvements and may induce sub-optimal policies due to the non-convexity of the policy function Tessler et al. (2019); on the contrary, our integrated approach of zeroth-order optimization and supervised learning can greatly improve the non-convex policy optimization and more likely escape the potential local minimum.

Contributions of this paper can be summarized as follows:

1. We introduce a simple yet effective policy optimization method called Zeroth-Order Supervised Policy Improvement (ZOSPI), as an alternative to the policy gradient approaches for continuous control. The policy from ZOSPI exploits global information of the learned value function Q and updates itself through sample-based supervised learning.
2. Practically, we point out the flexibility of the proposed supervised learning paradigm and show ZOSPI is capable of working with multi-modal policy class. And introduce a hybrid method that leverages both zeroth-order and first-order gradients for policy learning.
3. Empirically, we demonstrate the improved performance of ZOSPI over policy gradient methods on the continuous control benchmarks, in terms of both higher sample efficiency and asymptotic performance.

2 Related Work

Policy Gradient Methods The policy gradient methods solve an MDP by directly optimizing the policy to maximize the cumulative reward Williams (1992); Sutton and Barto (1998). While the prominent on-policy policy gradient methods like TRPO Schulman et al. (2015) and PPO Schulman et al. (2017) improve the learning stability of vanilla policy gradient Williams (1992) via trust region

updates, the off-policy methods such as Degris et al. (2012); Lillicrap et al. (2015); Wang et al. (2016) employ an experience replay mechanism to achieve higher sample efficiency. The work of TD3 Fujimoto et al. (2018a) further addresses the function approximation error and boosts the stability of DDPG with several improvements. Another line of works is the combination of policy gradient methods and the max-entropy regularizer, which leads to better exploration and stable asymptotic performances Haarnoja et al. (2017; 2018). All of these approaches adopt function approximators Sutton et al. (2000) for state or state-action value estimation and take directionally-uninformed Gaussian as the policy parameterization, which lead to a local exploration behavior Ciosek et al. (2019); Tessler et al. (2019).

RL by Supervised Learning Iterative supervised learning and self-imitate learning are becoming an alternative approach for model-free RL. Instead of applying policy gradient for policy improvement, methods based on supervised methods update policies by minimizing the mean square error between target actions and current actions predicted by a policy network Sun et al. (2019), or alternatively by maximizing the likelihood for a stochastic policy class Ghosh et al. (2019). While those previous works focus on the Goal-Conditioned tasks in RL, in this work we aim to tackle more general RL tasks. Some other works use supervised learning to optimize the policy towards manually selected policies to achieve better training stability under both offline Wang et al. (2018) and online settings Zhang et al. (2019b); Abdolmaleki et al. (2018); Song et al. (2019). Differently, in our work the policy learning is based on a much simpler formulation than the previous attempts Abdolmaleki et al. (2018); Lim et al. (2018); Simmons-Edler et al. (2019), and it does not rely on any expert data to achieve competitive performance.

Zeroth-Order Methods Zeroth-order optimization methods, also called gradient-free methods, are widely used when it is difficult to compute the gradients. They approximate the local gradient with random samples around the current estimate. The works in Wang et al. (2017); Golovin et al. (2019) show that a local zeroth-order optimization method has a convergence rate that depends logarithmically on the ambient dimension of the problem under some sparsity assumptions. It can also efficiently escape saddle points in non-convex optimizations Vlatakis-Gkaragkounis et al. (2019); Bai et al. (2020). In RL, many studies have verified an improved sample efficiency of zeroth-order optimization Usunier et al. (2016); Mania et al. (2018); Salimans et al. (2017). In this work we provide a novel way of combining the local sampling and the global sampling to ensure that our algorithm can approximate the gradient descent locally and also find a better global region.

3 Preliminaries

3.1 Markov Decision Processes

We consider the deterministic Markov Decision Process (MDP) with continuous state and action spaces in the discounted infinite-horizon setting. Such MDPs can be denoted as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma)$, where the state space $\mathcal{S}$ and the action space $\mathcal{A}$ are continuous, and the unknown state transition probability representing the transition dynamics is denoted by $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$. $r : \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$ is the reward function and $\gamma \in [0, 1]$ is the discount factor. An MDP $\mathcal{M}$ and a learning algorithm operating on $\mathcal{M}$ with an arbitrary initial state $s_0 \in \mathcal{S}$ constitute a stochastic process described sequentially by the state s_t visited at time step t, the action a_t chosen by the algorithm at step t, the reward $r_t = r(s_t, a_t)$ and the next state $s_{t+1} = \mathcal{T}(s_t, a_t)$ for any $t = 0, \ldots, T$. Let $H_t = \{s_0, a_0, r_0, \ldots, s_t, a_t, r_t\}$ be the trajectory up to time t. Our algorithm finds the policy that maximizes the discounted cumulative rewards $\sum_{t=0}^{T} \gamma^t r_t$. Our work follows the general Actor-Critic framework Konda and Tsitsiklis (2000); Peters and Schaal (2008); Degris et al. (2012); Wang et al. (2016), which learns in an unknown environment using a value network denoted by $Q_{w_t} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ for estimating Q values and a policy network for learning the behavior policy $\pi_{\theta_t} : \mathcal{S} \mapsto \mathcal{A}$. Here w_t and θ_t are the parameters of these two networks at step t, respectively.

3.2 Mixture Density Networks

The optimal policies in many environments tend to have multi-modal property. Mixture Density Networks (MDN), previously proposed in Bishop (1994) to solve the stochastic prediction problem, is a potential multi-modal policy function to be introduced in this work. One challenge here is that

Algorithm 1 Policy Update with Zeroth-Order and First-Order Optimization

1: **Require**
2: Objective function Q_s, domain $\mathcal{A}$, current policy network π_θ, perturbation network π_ϕ current point $a_0 = \pi_\theta(s)$, number of global samples n_1, number of local samples n_2
3: **Global sampling**
4: Sample n_1 points uniformly in the entire space by

$$a_i \sim \mathcal{U}_\mathcal{A}, \text{ for } i = 1, \ldots, n_1,$$

where $\mathcal{U}_\mathcal{A}$ is the uniform distribution over $\mathcal{A}$.
5: **Local sampling**
6: Sample n_2 points locally from a Gaussian centered at a_0 with a covariance matrix $\sigma^2 I$:

$$a_{i+n_1} \sim \mathcal{N}(a_0, \sigma^2 I), i = 1, \ldots, n_2.$$

7: Set $a^+ = \arg\max_{a \in \{a_0, \ldots, a_{n_1+n_2}\}} Q_s(a)$.
8: Update policy π_θ according to Eq.(2)
9: **Local perturbation**
10: Update perturbation network according to Eq. (4).

previous policy gradient methods are not compatible with multi-modal policies based on MDN. While in normal regression tasks, the learning objective is to maximize the log-likelihood of observed data given the current parameters of a predictive model, MDN uses the mixture of Gaussian with multiple parameters as $\mathcal{D} = \{X_i, Y_i\}_\mathbb{N}$. Then the likelihood is given by

$$\mathcal{L} = \sum_{i=1}^{i=\mathbb{N}} \mathcal{L}(Y_i|X_i, \theta) = \sum_{i=1}^{i=\mathbb{N}} \sum_{k=1}^{K} w_k(X_i, \theta)\phi(Y_i|\mu_k(X_i, \theta), \sigma_k(X_i, \theta)), \tag{1}$$

where θ denotes the parameters of neural networks with three branches of outputs $\{w_i, \mu_i, \sigma_i\}_K$, ϕ is the probability density function of normal distribution and $\sum_{i=1}^{K} w_i = 1$.

4 Zeroth-Order Supervised Continuous Control

Q-learning in the tabular setting finds the best discrete action given the current state, which can be difficult in the continuous action space due to the non-convexity of Q. A policy network is thus trained to approximate the optimal action. However, gradient-based algorithm may be trapped by local minima or saddle point depending on the initialization. In this section, we introduce a hybrid method with a global policy and a perturbation policy. The global policy predicts a coarse action by supervised learning on uniformly sampled actions and the perturbation network iterates based on the coarse prediction. In the following section, we give a motivating example to demonstrate the benefits of the hybrid framework.

Figure 1 shows a motivating example to demonstrate the benefits of applying zeroth-order optimization to policy updates. Consider we have a learned Q function with multiple local optima. Here we assume the conventional estimation of Q function is sufficient for a global exploitation Fujimoto et al. (2018a); Haarnoja et al. (2018) and we will discuss an improved estimation method in the Appendix. Our deterministic policy selects a certain action at this state, denoted as the red dot in Figure 1(a). In deterministic policy gradient methods Silver et al. (2014); Lillicrap et al. (2015), the policy gradient is conducted according to the chain rule to update the policy parameter θ with regard to Q-value by timing up the Jacobian matrix $\nabla_\theta \pi_\theta(s)$ and the derivative of Q, *i.e.*, $\nabla_a Q(s, a)$. Consequently, the policy gradient can only guarantee to find a local minima, and similar local improvement behaviors are also observed in stochastic policy gradient methods like PPO and SAC Schulman et al. (2017); Haarnoja et al. (2018); Tessler et al. (2019); Ciosek et al. (2019). Instead, if we can sample sufficient *random* actions in a broader range of the action space, denoted as blue dots in Figure 1(b), and then evaluate their values respectively through the learned Q estimator, it is possible to find a better initialization, from which the policy gradient can more likely find the global minima. Figure 1(c) shows the simulation result using different sample ranges for the sample-based optimization starting from the red point. It is clear that a larger sample range improves the chance of finding the global

optima. Utilizing such a global exploitation on the learned value function is the key insight of this work.

4.1 Zeroth-Order Supervised Policy Improvement

Based on the above motivation, we propose a framework called ZOSPI (Zeroth-Order Supervised Policy Improvement). ZOSPI consists of two policy networks: the global policy network that gives us a coarse estimate on the optimal action and a perturbation network that iterates based on the output of the global policy network using policy gradient.

Supervised learning for global policy network. We denote the global policy network by π_θ. At any step t, we sample a set of actions uniformly over the entire action space as well as a set of actions sampled from Gaussian distribution centered at current prediction π_{θ_t}. We denote by a_t^+, the action that gives the highest Q value with respect to current state s_t. Then we apply the supervised policy improvement that minimizes the L_2 distance between a_t^+ and $\pi_{\theta_t}(s_t)$, which gives the descent direction:

$$\nabla_\theta \frac{1}{2}(a_t^+ - \pi_{\theta_t}(s_t))^2 = (a_t^+ - \pi_{\theta_t}(s_t))\nabla_\theta \pi_{\theta_t}(s_t). \tag{2}$$

The global samples can help finding the regions that are better in the whole space. The local samples from Gaussian distribution accelerate later-stage training when the prediction is accurate enough and most global samples are not as good as the current prediction. The implementation detail is shown in Algorithm 1.

Policy gradient for perturbation network. We introduce another perturbation network, denoted by π_{ϕ_t}, as Fujimoto et al. (2018b). Such a perturbation network is parameterized by ϕ_t that performs fine-grained control on top of the global policy network π_{θ_t}. Different from the π_{θ_t}, the perturbation network π_{ϕ_t} takes both the current state s_t and the predicted action $\pi_{\theta_t}(s_t)$ as inputs, thus the final executed action is as follows:

$$a_t = \pi_{\theta_t}(s_t) + \pi_{\phi_t}(s_t, \pi_{\theta_t}(s_t)). \tag{3}$$

The range of the outputs for the perturbation network is limited to 0.05 times the value of maximal action. Therefore, it is only able to *perturb* the action provided by the policy network π_{θ_t}.

The perturbation network π_{ϕ_t} is trained with the policy gradient:

$$\nabla_\phi J = \mathbb{E}[\nabla_{a'} Q_w(s_t, a')|_{a'=\pi_{\theta_t}(s_t)+\pi_{\phi_t}(s_t)} \nabla_\phi \pi_{\phi_t}(s_t)]. \tag{4}$$

The intuition behind such an empirical design can be drawn from the example of Figure 1: although the global sampling step as well as the zeroth-order method helps policy optimization escape sub-optimal regions of the non-convex value function, the first order method can help to optimize the decision afterwards inside the locally convex region more efficiently. Eq. (4) shows that the updates are accessed only through the gradient, $\nabla_a Q_w$.

4.2 Analyses on the Benefits of Global Sampling

In this section, we give some analyses on the benefits of global sampling in terms of the sampling efficiency. Our analyses does not consider the improvement from the local sampling set, with which the quality of supervised learning can only be even better.

Error rate of the a_t^+. The performance of the supervised policy improvement heavily depends on the goodness of a_t^+, the best action in the sampling set. By assuming the continuity of the estimated Q function, we give an upper bound on the error rate $\|a_t^+ - a^*(s)\|$, where $a^*(s)$ is the true optimal action given the current state s.

Lemma 1. Assume the estimated Q function $Q_w(s_t, \cdot)$ given any s_t is L-Lipschitz with respect to action inputs. Let the action space $\mathcal{A} \subset \mathbb{R}^d$ for some positive integer d and assume that any action $a \in \mathcal{A}$, $\|a\|_2 \leq k$ for $k > 0$. Then with a probability at least $1 - \delta$, we have

$$\|a_t^+ - a^*(s)\|_2 \leq 2k\sqrt{d}L \left(\frac{\log^2(n/\delta)}{n}\right)^{1/d}.$$

Though the dependence of n is $1/n^{1/d}$, the estimation of a_t^+ is sufficient for a coarse prediction, because our prediction prediction π_θ aggregates all the historical information, which further reduces the error and we only need π_θ in a local convex region of the global optima. This rate can be much lower when we have a fairly better prediction and the best actions are given by the local sampling set instead.

Quality of π_θ. It is hard to directly evaluate the error rate of $\pi_\theta(s_t)$ because of the online stochastic gradient descent applied on a non-stationary distribution of states and a Q-function that is continuously being updated. Thus we take a step back and consider a simpler scenario. We assume a fixed Q function and a fixed state distribution. We assume that the policy network is a linear model with $\theta \in \mathbb{R}^p$. We consider a global empirical risk minimizer, so called ERM, since it is easy to achieve for linear models. Let our dataset with T samples be $\{(s_t, a_t^+)\}_{t=1}^T$. Our ERM estimate is defined by

$$\hat{\theta} = \arg\min_{\theta \in \Theta} \sum_{t=1}^{T} (a_t^+ - \pi_\theta(s_t))^2.$$

Theorem 1. If assumptions in Lemma 1 hold, with a high probability we have

$$\mathbb{E}\|\pi_{\hat{\theta}}(s) - a^*(s)\|_2^2 = \tilde{\mathcal{O}}\left(\frac{k\sqrt{d}pL}{n^{1/d}T} + \min_\theta \mathbb{E}\|\pi_\theta(s) - a^*(s)\|_2\right).$$

where $\tilde{\mathcal{O}}$ hides all the constant and logarithmic terms.

Theorem 1 implies that though the error of a_t^* at one step can be high, the error of our policy network can be further reduced by aggregating information from multiple steps. However, when π_θ is nonlinear functions, the quality may be worse than the case in our analyses. That is why we need some local perturbation through policy gradient to further improve our policy. Missing proofs in this section are given by Appendix B. Pseudo code of ZOSPI is provided in Appendix D.

4.3 Multi-Modal Continuous Control with ZOSPI

Different from standard policy gradient methods, the policy optimization step in ZOSPI can be considered as sampling-based supervised learning. Such a design enables many extensions of the proposed learning paradigm. In this section, we introduce the combination of ZOSPI and first-order method for stabilized training and the combination of ZOSPI and MDN for multi-modal policy learning.

Learning Multi-Modal Policies with Mixture Density Networks In the context of RL, there might be multiple optimal actions for some certain states, e.g., stepping up-ward and then right-ward may lead to the same state as stepping right and then upward, therefore both choices result in identical return in Figure 2. However, normal deterministic policy gradient methods can not capture such multi-modality due to the limitation that the DPG theorem is not applicable to a stochastic policy class.

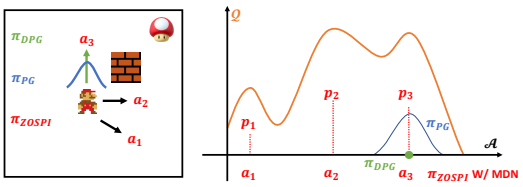


Figure 2: Illustration of how ZOSPI with MDN works when multiple optimal actions exist for a certain state: (left) Supermario is going to collect the mushroom, both a_2 and a_3 are the optimal actions for the current location. While deterministic policy gradient methods (green color) are only able to learn one of those optimal actions, ZOSPI with MDN is able to learn both. Policy gradient methods normally learn a Gaussian policy class.

In this section, we further integrate ZOSPI with the Mixture Density Networks (MDNs) Bishop (1994), which was introduced for multi-modal regression. Applying MDNs to ZOSPI leads to a more flexible and interpretable multi-modal policy class Tessler et al. (2019): different from normal Dirac policy parameterization used in TD3, ZOSPI with MDN predicts a mixture of Dirac policies, $i.e.$, the policy $\pi_{\text{MDN}}(s)$ predicts K choices for action $\vec{a} = \{a_1, ..., a_K\} \in \mathcal{A}^K$, with their corresponding probability $\vec{p} = \{p_1, ..., p_K\} \in \mathbb{R}^K$, and $\sum_i^K p_i = 1$. Then the action is sampled by

$$\pi_{\text{MDN}}(s) = a_i, \quad \text{w.p. } p_i, \text{ for } i = 1, ..., K \tag{5}$$

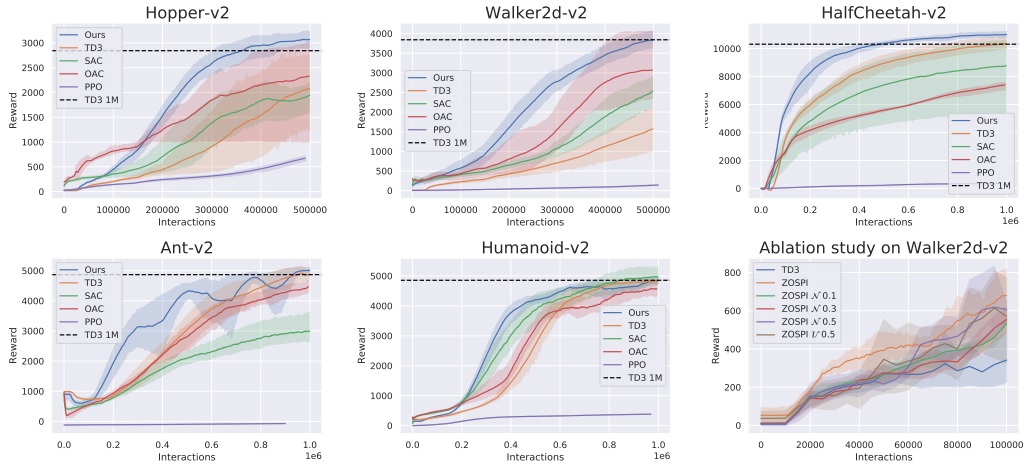


Figure 3: Experimental results on the MuJoCo locomotion tasks. The shaded region represents standard deviation. The dashed lines indicate asymptotic performance of TD3 after 1M interactions. ZOSPI is able to reach on-par performance within much less interactions. In all main experiments the results reported are collected from 10 random seeds and in ablations studies we use 5 random seeds. Curves are smoothed uniformly for visual clarity.

Thereby, instead of learning the mean value of multiple optimal actions, ZOSPI with MDN is able to learn multiple optimal actions. Different from previous stochastic multi-modal policies discussed in Haarnoja et al. (2018), the learning of multi-modal policy does not aim to fit the entire Q function distribution. Rather, ZOSPI with MDN focuses on learning the multi-modality in the best choices of actions, thus addresses the difficulty in generating multi-modal policies Haarnoja et al. (2017); Tessler et al. (2019). Figure 2 illustrates the difference between ZOSPI with MDN and the previous policy gradient methods.

5 EXPERIMENTS

In this section, we conduct experiments on five MuJoCo locomotion benchmarks to demonstrate the effectiveness of the proposed method. Specifically, we validate the following statements:

1. If we use ZOSPI with locally sampled actions, the performance of ZOSPI should be the same as its policy gradient counterpart like TD3; if we increase the sampling range, ZOSPI can better exploit the Q function thus find better solution than the policy gradient methods.
2. If we continuously increase the sampling range, it will result in an uniform sampling, and the Q function can be maximally exploited.
3. The perturbation network can help to improve the sample efficiency of the primal ZOSPI frame work purely based on zeroth-order optimization.
4. The supervised learning policy update paradigm of ZOSPI permits it to be flexibly work with multi-modal policy class.

5.1 ZOSPI ON THE MUJOCO LOCOMOTION TASKS.

We evaluate ZOSPI on the Gym locomotion tasks based on the MuJoCo engine Brockman et al. (2016); Todorov et al. (2012). The five locomotion tasks are Hopper-v2, Walker2d-v2, HalfCheetah-v2, Ant-v2, and Humanoid-v2. We compare our method with TD3 and SAC, the deterministic and the stochastic SOTA policy gradient methods. We also include PPO and OAC Ciosek et al. (2019) to better show the learning efficiency of ZOSPI. We compare different methods within 0.5M environment interactions in the two easy tasks to demonstrate the high learning efficiency of ZOSPI, and present results during 1M interactions for the other tasks. The results of TD3 and OAC are obtained by running the code released by the authors, the results of PPO are obtained through the

Table 1: Quantitative results. ZOSPI achieves comparable performance with only half the number of interactions with the environment, and achieves superior performance for 1M interactions.

METHOD/TASK	HOPPER-V2	WALKER2D-V2	HALFCHEETAH-V2	ANT-V2	HUMANOID-V2
TD3	2843 ± 197	3842 ± 239	10314 ± 93	4868 ± 388	4855 ± 263
SAC	2158 ± 388	$\mathbf{4154 \pm 333}$	8735 ± 170	3051 ± 469	$\mathbf{5012 \pm 478}$
OAC	2983 ± 317	3075 ± 183	4497 ± 296	4497 ± 297	4624 ± 351
OURS	$\mathbf{3268 \pm 234}$	4027 ± 28	$\mathbf{10992 \pm 126}$	$\mathbf{5006 \pm 135}$	4881 ± 164
IMPROV. OVER TD3/SAC	$\uparrow 15\%/ \uparrow 51\%$	$\uparrow 5\%/ \downarrow 3\%$	$\uparrow 7\%/ \uparrow 26\%$	$\uparrow 3\%/ \uparrow 64\%$	$\uparrow 1\%/ \downarrow 3\%$

METHOD/TASK	HOPPER-V2	WALKER2D-V2	HALFCHEETAH-V2	ANT-V2	HUMANOID-V2
TD3-0.5M	2234 ± 231	1648 ± 204	8801 ± 176	3232 ± 268	3036 ± 567
SAC-0.5M	1926 ± 410	2640 ± 529	7255 ± 180	2167 ± 399	4068 ± 610
OAC-0.5M	2395 ± 166	3074 ± 182	7405 ± 115	2903 ± 322	3610 ± 376
OURS-0.5M	$\mathbf{3048 \pm 307}$	$\mathbf{3932 \pm 125}$	$\mathbf{10290 \pm 129}$	$\mathbf{4304 \pm 260}$	$\mathbf{4175 \pm 302}$
IMPROV. OVER TD3/SAC	$\uparrow 58\%/ \uparrow 36\%$	$\uparrow 139\%/ \uparrow 49\%$	$\uparrow 17\%/ \uparrow 42\%$	$\uparrow 33\%/ \uparrow 99\%$	$\uparrow 38\%/ \uparrow 3\%$

high quality open-source implementation Dhariwal et al. (2017), and the results of SAC are extracted from the training logs of Haarnoja et al. (2018).

The learning curves of those tasks are shown in Figure 3. It is worth noting that in all of the results we reported, only 50 actions are sampled and it is sufficient to learn well-performing policies. With a high sampling efficiency, ZOSPI works well in the challenging environments that have high-dimensional action spaces such as Ant-v2 and Humanoid-v2. In all the tasks the sample efficiency is consistently improved over TD3, which is the DPG counterpart of ZOSPI. While a total of 50 sampled actions should be very sparse in the high dimensional space, we attribute the success of ZOSPI to the generality of the policy network as well as the sparsity of meaningful actions. For example, even in the tasks that have high dimensional action spaces, only limited dimensions of the action are crucial.

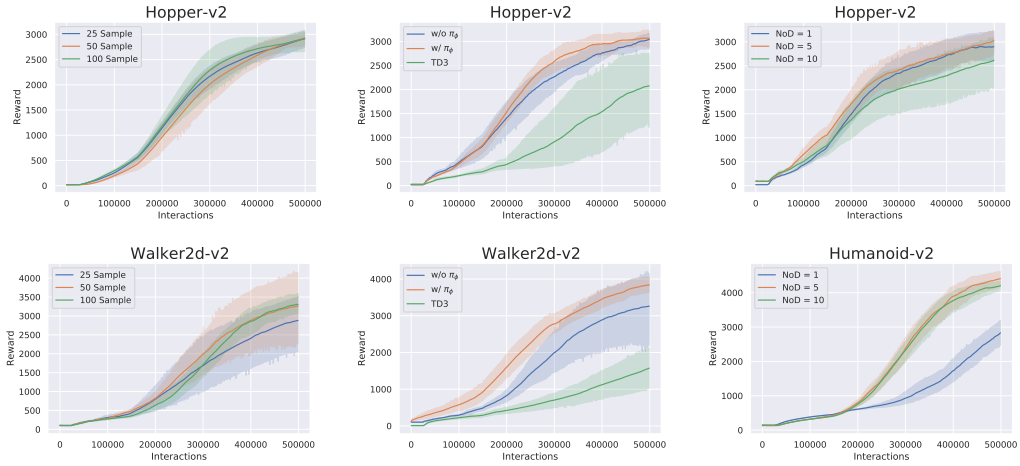
We provide quantitative comparison in Table 1. We report both the averaged score and the standard deviation of different methods after 1M step of interactions with the environment. Moreover, we provide the quantitative results of ZOSPI with 0.5M interactions during training to emphasize on its high sample efficiency. In 4 out of the 5 environments (except Humanoid), ZOSPI is able to achieve on-par performance compared to TD3 and SAC, with only half number of interactions. The last line of Table 1 reports the improvement of ZOSPI over TD3 and SAC respectively. All results are collected by experimenting with 10 random seeds.

The last plot in Figure 3 shows the ablation study on the sampling range in ZOSPI, where a sampling method based on a zero-mean Gaussian is applied and we gradually increase its variance from 0.1 to 0.5. We also evaluate the uniform sampling method with radius of 0.5, which is denoted as $\mathcal{U}$ 0.5 in the Figure. The results suggest that zeroth-order optimization with local sampling performs similarly to the policy gradient method, and increasing the sampling range can effectively improve the performance.

5.2 ABLATION STUDY

In this section, we conduct a series of experiments as the ablation study to evaluate different components of ZOSPI. Specifically, we investigate the performance of ZOSPI under different number of samples, with or without perturbation networks, and Number of Diracs (NoD) used in the policy network.

Number of Samples Used in ZOSPI The first column of Figure 4 shows the performance of ZOSPI when using different number of samples. In both environments, more samples lead to better performance, but in the easier task of Hopper, there are only tiny differences, while in the task of Walker2d, using more than 50 samples noticeably improves the performance. Trading off the improvements and the computational costs when using more samples, we choose to use 50 samples for the experiments we reported in the previous section.



(a) Number of Samples (b) Perturbation Network (c) Number of Gaussians

Figure 4: Ablation study. The first column (a) shows the performance of our proposed method with different number of samples; the second column (b) shows the performance difference when ZOSPI is work with or without the perturbation network; the last column (c) shows the performance difference between using different number of Diracs when combining MDNs with ZOSPI.

Perturbation Network The second column of Figure 4 shows the ablation studies on the perturbation network. While in the simpler control task of Hopper, the perturbation network helps improve the learning efficiency only a little bit, in more complex task the perturbation network is crucial for efficient learning. This result demonstrate the effectiveness of the combination of global and local information in exploiting of the learned value function. In all of our main experiments, we use a perturbation network with a perturbation range of 0.05.

The Application of MDNs Our ablation studies on the usage of MDNs are shown in the last column of Figure 4. In the easier task of Hopper, using a larger number of NoD in ZOSPI with MDN only improves a little but introduce several times of extra computation expense. However, such multi-modal policies clearly improves the performance in the complex tasks like Humanoid. In our experiments, we find larger NoD benefits the learning for the Walker2d and Humanoid tasks, and do not improve performance in the other three environments. More implementation details of ZOSPI with MDN can be found in Appendix E.

6 Conclusion

In this work, we propose the method of Zeroth-Order Supervised Policy Improvement (ZOSPI) as an alternative approach of policy gradient algorithms for continuous control. ZOSPI improves the learning efficiency of previous policy gradient learning by exploiting the learned Q functions globally and locally through sampling the action space. Different from previous policy gradient methods, the policy optimization of ZOSPI is based on supervised learning so that the learning of actor can be implemented with regression. Such a property enables potential extensions such as multi-modal policies, which can be seamlessly cooperated with ZOSPI. We evaluate ZOSPI on five locomotion tasks, where it remarkably improves the performance in terms of both sample efficiency and asymptotic performance compared to previous policy gradient methods.

REFERENCES

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

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

Jakub Pachocki, Greg Brockman, Jonathan Raiman, Susan Zhang, Henrique Pondé, Jie Tang, Filip Wolski, Christy Dennison, Rafal Jozefowicz, Przemyslaw Debiak, et al. Openai five, 2018. *URL https://blog. openai. com/openai-five*.

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

Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E Turner, and Sergey Levine. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247*, 2016.

Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay. *arXiv preprint arXiv:1611.01224*, 2016.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Scott Fujimoto, Herke Van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018a.

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

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

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1352–1361. JMLR. org, 2017.

Shangtong Zhang, Wendelin Boehmer, and Shimon Whiteson. Generalized off-policy actor-critic. *arXiv preprint arXiv:1903.11329*, 2019a.

Kamil Ciosek, Quan Vuong, Robert Loftin, and Katja Hofmann. Better exploration with optimistic actor critic. In *Advances in Neural Information Processing Systems*, pages 1785–1796, 2019.

Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002.

Chen Tessler, Guy Tennenholtz, and Shie Mannor. Distributional policy optimization: An alternative approach for continuous control. *arXiv preprint arXiv:1905.09855*, 2019.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *ICML*, 2014.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth Stanley, and Jeff Clune. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. In *Advances in Neural Information Processing Systems*, pages 5027–5038, 2018.

Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

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

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. 1998.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

Hao Sun, Zhizhong Li, Xiaotong Liu, Bolei Zhou, and Dahua Lin. Policy continuation with hindsight inverse dynamics. In *Advances in Neural Information Processing Systems*, pages 10265–10275, 2019.

Dibya Ghosh, Abhishek Gupta, Justin Fu, Ashwin Reddy, Coline Devine, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals without reinforcement learning. *arXiv preprint arXiv:1912.06088*, 2019.

Qing Wang, Jiechao Xiong, Lei Han, Han Liu, Tong Zhang, et al. Exponentially weighted imitation learning for batched historical data. In *Advances in Neural Information Processing Systems*, pages 6288–6297, 2018.

Chuheng Zhang, Yuanqi Li, and Jian Li. Policy search by target distribution learning for continuous control. *arXiv preprint arXiv:1905.11041*, 2019b.

Abbas Abdolmaleki, Jost Tobias Springenberg, Jonas Degrave, Steven Bohez, Yuval Tassa, Dan Belov, Nicolas Heess, and Martin Riedmiller. Relative entropy regularized policy iteration. *arXiv preprint arXiv:1812.02256*, 2018.

H Francis Song, Abbas Abdolmaleki, Jost Tobias Springenberg, Aidan Clark, Hubert Soyer, Jack W Rae, Seb Noury, Arun Ahuja, Siqi Liu, Dhruva Tirumala, et al. V-mpo: On-policy maximum a posteriori policy optimization for discrete and continuous control. *arXiv preprint arXiv:1909.12238*, 2019.

Sungsu Lim, Ajin Joseph, Lei Le, Yangchen Pan, and Martha White. Actor-expert: A framework for using q-learning in continuous action spaces. *arXiv preprint arXiv:1810.09103*, 2018.

Riley Simmons-Edler, Ben Eisner, Eric Mitchell, Sebastian Seung, and Daniel Lee. Q-learning for continuous actions with cross-entropy guided policies. *arXiv preprint arXiv:1903.10605*, 2019.

Yining Wang, Simon Du, Sivaraman Balakrishnan, and Aarti Singh. Stochastic zeroth-order optimization in high dimensions. *arXiv preprint arXiv:1710.10551*, 2017.

Daniel Golovin, John Karro, Greg Kochanski, Chansoo Lee, Xingyou Song, et al. Gradientless descent: High-dimensional zeroth-order optimization. *arXiv preprint arXiv:1911.06317*, 2019.

Emmanouil-Vasileios Vlatakis-Gkaragkounis, Lampros Flokas, and Georgios Piliouras. Efficiently avoiding saddle points with zero order methods: No gradients required. In *Advances in Neural Information Processing Systems*, pages 10066–10077, 2019.

Qinbo Bai, Mridul Agarwal, and Vaneet Aggarwal. Escaping saddle points for zeroth-order nonconvex optimization using estimated gradient descent. In *2020 54th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6. IEEE, 2020.

Nicolas Usunier, Gabriel Synnaeve, Zeming Lin, and Soumith Chintala. Episodic exploration for deep deterministic policies for starcraft micromanagement. 2016.

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

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

Christopher M Bishop. Mixture density networks. 1994.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. *arXiv preprint arXiv:1812.02900*, 2018b.

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

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IROS*, pages 5026–5033. IEEE, 2012. ISBN 978-1-4673-1737-5. URL `http://dblp.uni-trier.de/db/conf/iros/iros2012.html#TodorovET12`.

Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. Openai baselines. `https://github.com/openai/baselines`, 2017.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 263–272. JMLR. org, 2017.

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, pages 4863–4873, 2018.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In *Advances in neural information processing systems*, pages 4026–4034, 2016.

Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 8617–8629, 2018.

Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. Striving for simplicity in off-policy deep reinforcement learning. *arXiv preprint arXiv:1907.04543*, 2019.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11761–11771, 2019.

Malte Kuss and Carl E Rasmussen. Gaussian processes in reinforcement learning. In *Advances in neural information processing systems*, pages 751–758, 2004.

Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, pages 201–208, 2005.

Malte Kuss. *Gaussian process models for robust regression, classification, and reinforcement learning*. PhD thesis, echnische Universität Darmstadt Darmstadt, Germany, 2006.

Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. In *Advances in Neural Information Processing Systems*, pages 19–27, 2011.

Robert Grande, Thomas Walsh, and Jonathan How. Sample efficient reinforcement learning with gaussian processes. In *International Conference on Machine Learning*, pages 1332–1340, 2014.

Ying Fan, Letian Chen, and Yizhou Wang. Efficient model-free reinforcement learning using gaussian process. *arXiv preprint arXiv:1812.04359*, 2018.

A VISUALIZATION OF Q-LANDSCAPE

Figure 5 shows the visualization of learned policies (actions given different states) and Q values in TD3 during training in the Pendulum-v0 environment, where the state space is 3-dim and action space is 1-dim. The red lines indicates the selected action by the current policy. The learned Q function are always **non-convex** and **locally convex**. As a consequence, in many states the TD3 is not able to find globally optimal solution and local gradient information may be misleading in finding actions with the highest Q values.

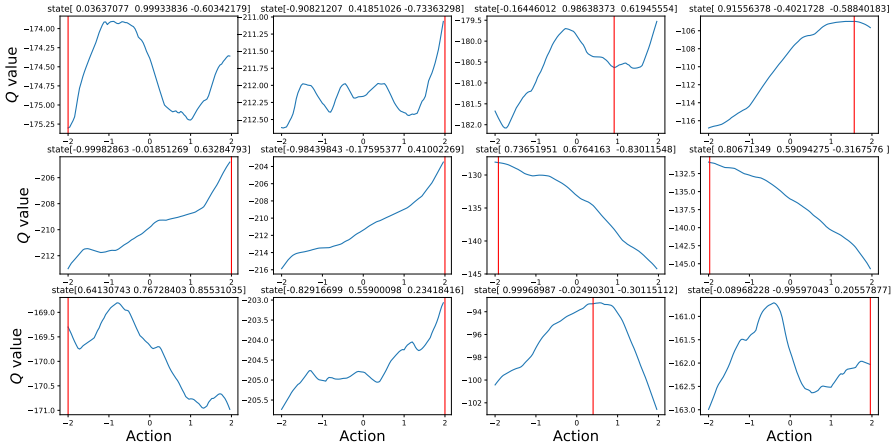


Figure 5: Landscape of learned value function of TD3 in the Pendulum-v0 environment (control task with 1-dim action space so that the state-action values can be easily visualized.)

B MISSING PROOFS IN SECTION 4.2

Proof of Lemma 1. By the properties of Euclidean metric, we have the following lemma.

Lemma 2. Let $\mathcal{A} \subset \mathbb{R}^d$ with Euclidean metric. Any $a \in \mathcal{A}$, $\|x\|_2 \leq k$. Then there exists a set $\{a_1, \ldots, a_N\}$ that is a $\frac{2k\sqrt{d}}{N^{1/d}}$-covering of N. In other words, for all $a \in \mathcal{A}$, $\exists i \in [N]$,

$$\|a - a_i\|_2 \leq \frac{2k\sqrt{d}}{N^{1/d}}.$$

Define $\mathcal{A}_i$ as the set of all the actions that is closest to a_i:

$$\mathcal{A}_i \triangleq \left\{ a \in \mathcal{A} : i = \min\{\arg\min_{j \in [N]} \|a_j - a\|_2\} \right\}.$$

Let $P(\cdot)$ be the probability measure of uniform distribution over $\mathcal{A}$ We have $P(\mathcal{A}_i) \geq 1/(2N)$. Now since we have n uniform samples from set $\mathcal{A}$. Let $N = n/\log^2(n/\delta)$.

Lemma 3 (Coupon Collector's problem)**.** It takes $O(N \log^2(N/\delta))$ rounds of random sampling to see all N distinct options with a probability at least $1 - \delta$.

Proof. Consider a general sampling problem: for any finite set $\mathcal{N}$ with $|\mathcal{N}| = N$. For any n, whose sampling probability is $p(c)$, with a probability at least $1 - \delta$, it requires at most

$$\frac{\log(1/\delta)}{\log(1 + \frac{p(n)}{1-p(n)})} \text{ for } n \text{ to be sampled.}$$

Since $\log(1+x) \geq x - \frac{1}{2}x^2$ for all $x > 0$, we have

$$\frac{\log(1/\delta)}{\log(1+\frac{p(n)}{1-p(n)})} \leq \log(1/\delta)\frac{1}{\frac{p(n)}{1-p(n)} - \frac{p(n)^2}{2(1-p(n))^2}} = O(\log(1/\delta)\frac{1-p(n)}{p(n)}).$$

Searching the whole space $\mathcal{N}$ with each new element being found with probability $\frac{N-i}{N}$ at round i, it requires at most

$$O(\sum_{i=1}^{N} \log(\frac{N}{\delta})\frac{N}{N-i}) = O(\log^2(\frac{N}{\delta})N),$$

with a probability at most $1-\delta$.

By Lemma 3 We have with a probability at least $1-\delta$, there exists a sample in each $\mathcal{A}_i$ described above. To proceed, we apply the Lipschitz of the Q function, Lemma 1 follows.

Proof of Theorem 1. Now we proceed to show Theorem 1. We denote $\frac{2k\sqrt{d}L\log^{1/d}(n/\delta)}{n^{1/d}}$ by σ^2. Our global policy network gives a prediction from a linear model. Let our dataset be $\{s_t, a_t^+\}_{t=1}^T$. Let $\epsilon_t = a_t^+ - a_t^*$. Let $\boldsymbol{S} = (s_1, \ldots, s_T)^T$ and $\boldsymbol{a}^+, \boldsymbol{a}^*, \boldsymbol{\epsilon}$ be the corresponding vector for $(a_t^+)_{t=1}^T, (a_t^*)_{t=1}^T, (\epsilon_t)_{t=1}^T$. We have any ϵ_i, ϵ_j are independent for $i \neq j$. We further make an assumption that $\mathbb{E}[\epsilon_t] = 0$. Then we immediately have $\mathbb{E}[\epsilon_i\epsilon_j] = 0$ as well. This assumption is just for simplifying the proof, we can show similar results without assume the unbiasedness as the bias can be bounded.

To proceed, let $\theta^* = \arg\min_\theta \mathbb{E}_s\|s^T\theta - a^*(s)\|_2^2$.

Since the ERM solution is simply OLS (ordinary least square). We have the estimate

$$\hat{\theta} = (\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^T\boldsymbol{a}^+.$$

We have

$$\hat{\theta} - \theta^* = (\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^T(\boldsymbol{a}^* - \boldsymbol{S}\theta^*) + (\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^T\boldsymbol{\epsilon}.$$

Since $\|\epsilon_t\|^2 \leq \sigma^2$, we have $\mathrm{Var}(\epsilon_t) \leq \sigma^2$. We observe that

$$\mathrm{Var}((\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^T\boldsymbol{\epsilon}) \leq \sigma^2((\boldsymbol{S}^T\boldsymbol{S})^{-1}) = \mathcal{O}(\sigma^2 p/T).$$

The generalization error is given by

$$\begin{aligned}
&\mathbb{E}_s\|s^T\hat{\theta} - a^*(s)\|_2^2 \\
&\leq \mathbb{E}_s\|s^T\hat{\theta} - s^T\theta^*\|_2^2 + \mathbb{E}_s\|s^T\theta^* - a^*(s)\|_2^2 \\
&= \mathcal{O}\left(\frac{\sigma^2 p}{T} + \mathbb{E}_s\|s^T(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^T(\boldsymbol{a}^* - \boldsymbol{S}\theta^*)\|_2^2 + \mathbb{E}_s\|s^T\theta^* - a^*(s)\|_2^2\right) \\
&= \tilde{\mathcal{O}}\left(\frac{\sigma^2 p}{T} + \mathbb{E}_s\|s^T\theta^* - a^*(s)\|_2^2\right).
\end{aligned}$$

C ONE-STEP ZEROTH-ORDER OPTIMIZATION WITH CONSISTENT ITERATION

Algorithm 2 One-step Zeroth-Order Optimization with Consistent Iteration

Require
Objective function Q, domain $\mathcal{A}$, current point a_0, number of local samples n_1, number of global samples n_2, local scale $\eta > 0$ and step size h, number of steps m.
for $t = 1, \dots n_2$ **do**
  Globally sampling
  Sample a point uniformly in the entire space by

$$a_{t0} \sim \mathcal{U}_{\mathcal{A}}$$

  where $\mathcal{U}_{\mathcal{A}}$ is the uniform distribution over $\mathcal{A}$.
  for $i = 1, \dots, m$ **do**
    Locally sampling
    Sample n_1 points around $a_{t,i-1}$ by

$$\tilde{a}_j = a_{t,i-1} + \mu e_j \text{ for } e_j \sim \mathcal{N}(0, I_d), j = 1, \dots n_1,$$

    where $\mathcal{N}(0, I_d)$ is the standard normal distribution centered at 0.
    Update
    Set $a_{t,i} = a_{t,i-1} + h(\arg\max_{a \in \{\tilde{a}_j\}} Q(a) - a_{t,i-1})$
  end for
end for
return $\max_{a \in \{a_t m\}_{t=1}^{n_2}} Q(a)$.

D PSEUDO CODE OF ZOSPI

Algorithm 3 Zeroth-Order Supervised Policy Improvement (ZOSPI)

Require
  Number of epochs M, size of mini-batch N, momentum $\tau > 0$.
  Random initialized policy network π_θ, target policy network $\pi_{\theta'}$, perturbation network π_ϕ, $\theta' \leftarrow \theta$.
  Two random initialized Q networks, and corresponding target networks, parameterized by w_1, w_2, w_1', w_2'. $w_i' \leftarrow w_i$.
  Empty experience replay buffer $\mathcal{D} = \{\}$.
for iteration $= 1, 2, \dots$ **do**
  for t $= 1, 2, \dots, T$ **do**
    # Interaction
    Rollout with $a_t = \pi_{\theta_t}(s_t) + \pi_{\phi_t}(s_t, \pi_{\theta_t}(s_t))$, store transition (s_t, a_t, s_{t+1}, r_t) in $\mathcal{D}$.
    for epoch $= 1, 2, \dots, M$ **do**
      Sample a mini-batch of transition tuples $\mathcal{D}' = \{(s_{t_j}, a_{t_j}, s_{t_j+1}, r_{t_j})\}_{j=1}^N$.
      # Update Q
      Calculate target Q value $y_j = r_{t_j} + \min_{i=1,2} Q_{w_i'}(s_{t_j+1}, \pi_{\theta'}(s_{t_j}))$.
      Update w_i with one step gradient descent on the loss $\sum_j (y_j - Q_{w_i}(s_{t_j}, a_{t_j}))^2$, $i = 1, 2$.
      # Update π
      Call Algorithm 1 for policy optimization to update θ and ϕ.
    end for
    $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$; $w_i' \leftarrow \tau w_i + (1-\tau) w_i'$
  end for
end for

E IMPLEMENTATION DETAILS

E.1 NETWORK STRUCTURE AND HYPER-PARAMS

In our experiments, we follow Fujimoto et al. (2018a) to use a 3-layer MLP with 256 hidden units for both critic and actor networks. We also follow Fujimoto et al. (2018a) to use 25000 timesteps for worm-up and use a batch-size of $256 : 1$ training-interaction proportion during training.

E.2 MIXTURE DENSITY NETWORKS

Our implementation of ZOSPI with MDN is based on neural network with multiple outputs. For MDN with K Gaussian mixture outputs, the neural network has $3 \times K$-dim output. The first K-dim units are normalized with softmax activation as the probability of selecting the Gaussians, the following K-dim units are corresponding K mean values of the Gaussians, and the last K-dim units are standard deviation of the K Gaussians. In our experiments, we use Diracs instead of Gaussians for parameterization. Therefore, our networks output $2 \times K$-dim units for each action dimension, where the first K-dimensions denote the probabilities and the latter K-dimensions denote the mean values. In our experiments, we use $K = 1$ for Hopper, HalfCheetah and Ant, $K = 5$ for Walker2d, and $K = 5$ for Humanoid. ($K = 5$ and $K = 10$ achieve on-par performance for the Humanoid environment, though the $K = 10$ setting spend roughly one more time computational expense).

More implementation details are provided with the code in the supplementary material.

E.3 RUNNING TIME OF ZOSPI

We conduct our experiments with 8 GTX TITAN X GPUs and 32 Intel(R) Xeon(R) E5-2640 v3 @ 2.60GHz CPUs. The wall clock time of our proposed method is roughly 3-times slower than running TD3, without application of MDNs (i.e., NoD $= 1$). It takes roughly 20 hours to train Hopper with 10 seeds, and takes about 120 hours to train Humanoid when NoD $= 10$ with 10 seeds.

F BETTER EXPLORATION WITH BOOTSTRAPPED NETWORKS

Sample efficient RL requires algorithms to balance exploration and exploitation. One of the most popular way to achieve this is called optimism in face of uncertainty (OFU) Brafman and Tennenholtz (2002); Jaksch et al. (2010); Azar et al. (2017); Jin et al. (2018), which gives an upper bound on Q estimates and applies the optimal action corresponding to the upper bound. The optimal action a_t is given by the following optimization problem:

$$\arg\max_a Q^+(s_t, a), \tag{6}$$

where Q^+ is the upper confidence bound on the optimal Q function. A guaranteed exploration performance requires both a good solution for (6) and a valid upper confidence bound.

While it is trivial to solve (6) in the tabular setting, the problem can be intractable in a continuous action space. Therefore, as shown in the previous section, ZOSPI adopts a local set to approximate policy gradient descent methods in the local region and further applies a global sampling scheme to increase the potential chance of finding a better maxima.

As for the requirement of a valid upper confidence bound, we use bootstrapped Q networks to address the uncertainty of Q estimates as in Osband et al. (2016; 2018); Agarwal et al. (2019); Kumar et al. (2019); Ciosek et al. (2019). Specifically, we keep K estimates of Q, namely $Q_1, \dots Q_K$ with bootstrapped samples from the replay buffer. Let $\overline{Q} = \frac{1}{K}\sum_k Q_k(s, a)$. An upper bound Q^+ is

$$Q^+(s, a) = \overline{Q} + \phi\sqrt{\frac{1}{K}\sum_k [Q_k(s, a) - \overline{Q}]^2}, \tag{7}$$

where ϕ is the hyper-parameter controlling the failure rate of the upper bound. Another issue is on the update of bootstrapped Q networks. Previous methods Agarwal et al. (2019) usually update each Q network with the following target $r_t + \gamma Q_k\left(s_{t+1}, \pi_{\theta_t}(s_{t+1})\right)$, which violates the Bellman

equation as π_{θ_t} is designed to be the optimal policy for Q^+ rather than Q_k. Using π_{θ_t} also introduces extra dependencies among the K estimates. We instead employ a global random sampling method to correct the violation as

$$r_t + \gamma \max_{i=1,\dots n} Q_k\left(s_{t+1}, a_i\right), \quad a_1, \dots a_n \sim \mathcal{U}_{\mathcal{A}}.$$

The correction also reinforces the argument that a global random sampling method yields a good approximation to the solution of the optimization problem (6). The detailed algorithm is provided in Algorithm 4 in Appendix F.1.

F.1 Algorithm 4: ZOSPI with Bootstrapped Q networks

Algorithm 4 ZOSPI with UCB Exploration

Require

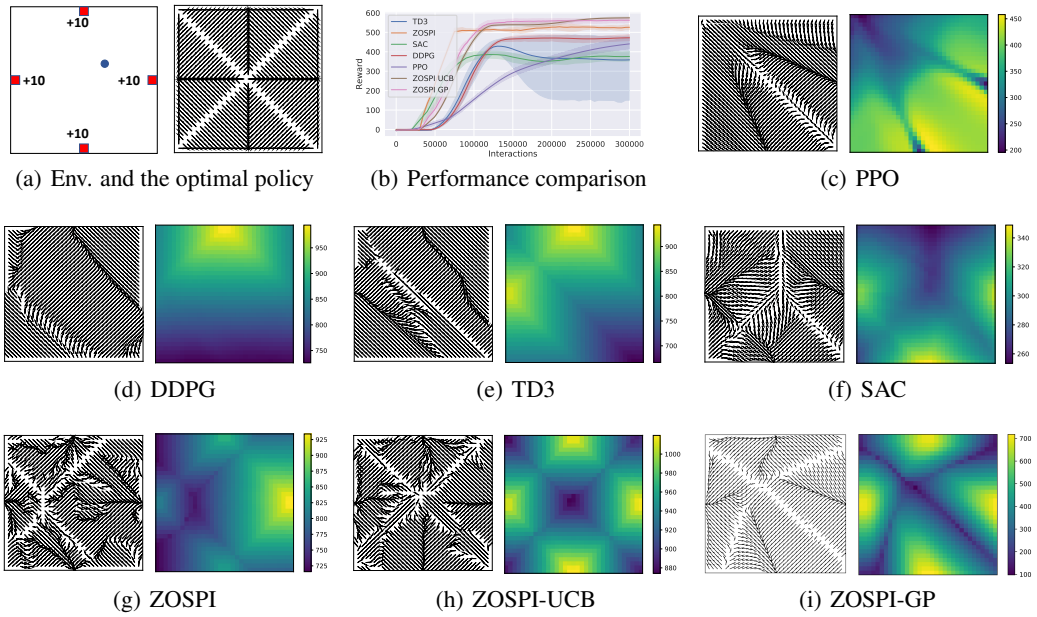
- The number of epochs M, the size of mini-batch N, momentum $\tau > 0$ and the number of Bootstrapped Q-networks K.
- Random initialized policy network π_{θ_1}, target policy network $\pi_{\theta'_1}$, $\theta'_1 \leftarrow \theta_1$.
- K random initialized Q networks, and corresponding target networks, parameterized by $w_{k,1}, w'_{k,1}$, $w'_{k,1} \leftarrow w_{k,1}$ for $k = 1, \dots, K$.

for iteration $= 1, 2, \dots$ **do**
  for t $= 1, 2, \dots, T$ **do**
    # Interaction
    Run policy $\pi_{\theta'_t}$, and collect transition tuples $(s_t, a_t, s'_t, r_t, m_t)$.
    for epoch $j = 1, 2, \dots, M$ **do**
      Sample a mini-batch of transition tuples $\mathcal{D}_j = \{(s, a, s', r, m)_i\}_{i=1}^N$.
      # Update Q
      for $k = 1, 2, \dots, K$ **do**
        Calculate the k-th target Q value $y_{ki} = r_i + \max_l Q_{w'_{k,t}}(s'_i, a'_l)$, where $a'_l \sim \mathcal{U}_{\mathcal{A}}$.
        Update $w_{k,t}$ with loss $\sum_{i=1}^N m_{ik}(y_{ki} - Q_{w'_{k,t}}(s_i, a_i))^2$.
      end for
      # Update π
      Calculate the predicted action $a_0 = \pi_{\theta'_t}(s_i)$
      Sample actions $a_l \sim \mathcal{U}_{\mathcal{A}}$
      Select $a^+ \in \{a_l\} \cup \{a_0\}$ as the action with maximal $Q^+(s_t, a)$ defined in (7).
      Update policy network with Eq.(2).
    end for
    $\theta'_{t+1} \leftarrow \tau\theta_t + (1-\tau)\theta'_t$.
    $w'_{k,t+1} \leftarrow \tau w_{k,t} + (1-\tau)w'_{k,t}$.
    $w_{k,t+1} \leftarrow w_{k,t}$; $\theta_{t+1} \leftarrow \theta_t$.
  end for
end for

G Gaussian Processes for Continuous Control

Different from previous policy gradient methods, the self-supervised learning paradigm of ZOSPI permits it to learn both its actor and critic with a regression formulation. Such a property enables the learning of actor in ZOSPI to be implemented with either parametric models like neural networks or non-parametric models like Gaussian Processes (GP). Although plenty of previous works have discussed the application of GP in RL by virtue of its natural uncertainty capture ability, most of these works are limited to model-based methods or discrete action spaces for value estimation Kuss and Rasmussen (2004); Engel et al. (2005); Kuss (2006); Levine et al. (2011); Grande et al. (2014); Fan et al. (2018). On the other hand, ZOSPI formulates the policy optimization in continuous control tasks as a regression objective, therefore empowers the usage of GP policy in continuous control tasks.

As a first attempt of applying GP policies in continuous control tasks, we simply alter the actor network with a GP to interact with the environment and collect data, while the value approximator is still parameterized by a neural network. We leave the investigation of better consolidation design in future work.



(a) Env. and the optimal policy (b) Performance comparison (c) PPO

(d) DDPG (e) TD3 (f) SAC

(g) ZOSPI (h) ZOSPI-UCB (i) ZOSPI-GP

Figure 6: Visualization of learned policies on the FSM environment. (a) the FSM environment and its optimal solution, where the policy should find the nearest reward region and move toward it; (b) learning curves of different approaches; (c)-(i) visualize the learned policies and corresponding value functions. We run multiple repeat experiments and show the most representative and well-performing learned value function and policy of each method.

H EXPERIMENTS ON THE FOUR-SOLUTION-MAZE.

The Four-Solution-Maze (FSM) environment is a diagnostic environment where four positive reward regions with a unit side length are placed in the middle points of 4 edges of a $N \times N$ map. An agent starts from a uniformly initialized position in the map and can then move in the map by taking actions according to the location observations (current coordinates x and y). Valid actions are limited to $[-1, 1]$ for both x and y axes. Each game consists of $2N$ timesteps for the agent to navigate in the map and collect rewards. In each timestep, the agent will receive a $+10$ reward if it is inside one of the 4 reward regions or a tiny penalty otherwise. For simplicity, there are no obstacles in the map, the optimal policy thus will find the nearest reward region, directly move towards it, and stay in the region till the end. Figure 6(a) visualizes the environment and the ground-truth optimal solution.

Although the environment is simple, we found it extremely challenging due to existence of multiple sub-optimal policies that only find some but not all four reward regions. We do not conduct grid search on hyper-parameters of the algorithms compared in our experiments but set them to default setting across all experiments. Though elaborated hyper-parameter tuning may benefit for certain environment.

On this environment we compare ZOSPI to on-policy and off-policy SOTA policy gradient methods in terms of the learning curves, each of which is averaged by 5 runs. The results are presented in Figure 6(b). And learned policies from different methods are visualized in Figure 6(c)-6(i). For each method we plot the predicted behaviors of its learned policy at grid points using arrows (although the environment is continuous in the state space), and show the corresponding value function of its learned policy with a colored map. All policies and value functions are learned with 0.3M interactions except for SAC whose figures are learned with 1.2M interactions as it can find 3 out of 4 target regions when more interactions are provided.

We use 4 bootstrapped Q networks for the upper bound estimation in consideration of both better value estimation and computational cost for ZOSPI with UCB. And in ZOSPI with GP, a GP model is used to replace the actor network in data-collection, $i.e.$,exploration. The sample efficiency of ZOSPI

is much higher than that of other methods. Noticeably ZOSPI with UCB exploration is the only method that can find the optimal solution, *i.e.*, a policy directs to the nearest region with a positive reward. All other methods get trapped in sub-optimal solutions by moving to only part of reward regions they find instead of moving toward the nearest one.