Journal of Computational Mathematics
Vol.40, No.6, 2022, 1004–1030.

http://www.global-sci.org/jcm
doi:10.4208/jcm.2104-m2021-0007

A STOCHASTIC TRUST-REGION FRAMEWORK FOR POLICY OPTIMIZATION*

Mingming Zhao, Yongfeng Li and Zaiwen Wen[1)]
Beijing International Center for Mathematical Research, Peking University, Beijing 100871, China
Email: mmz102@pku.edu.cn, yongfengli@pku.edu.cn, wenzw@pku.edu.cn

Abstract

In this paper, we study a few challenging theoretical and numerical issues on the well known trust region policy optimization for deep reinforcement learning. The goal is to find a policy that maximizes the total expected reward when the agent acts according to the policy. The trust region subproblem is constructed with a surrogate function coherent to the total expected reward and a general distance constraint around the latest policy. We solve the subproblem using a preconditioned stochastic gradient method with a line search scheme to ensure that each step promotes the model function and stays in the trust region. To overcome the bias caused by sampling to the function estimations under the random settings, we add the empirical standard deviation of the total expected reward to the predicted increase in a ratio in order to update the trust region radius and decide whether the trial point is accepted. Moreover, for a Gaussian policy which is commonly used for continuous action space, the maximization with respect to the mean and covariance is performed separately to control the entropy loss. Our theoretical analysis shows that the deterministic version of the proposed algorithm tends to generate a monotonic improvement of the total expected reward and the global convergence is guaranteed under moderate assumptions. Comparisons with the state-of-the–art methods demonstrate the effectiveness and robustness of our method over robotic controls and game playings from OpenAI Gym.

Mathematics subject classification: 49L20, 90C15, 90C26, 90C40, 93E20.
Key words: Deep reinforcement learning, Stochastic trust region method, Policy optimization, Global convergence, Entropy control.

1. Introduction

In reinforcement learning, the agent starts from an initial state and interacts with the environment by executing an action from some policy iteratively. At each time step, the environment transforms the current state into the next state with respect to the action selected by the agent and gives back a reward to the agent to evaluate how good the action is, then the agent makes a new action for the next interaction based on the feedback. Repeating the above transition dynamics generates a trajectory where stores the visited states, actions and rewards. During the interactions, the transition probability and the reward function are totally determined by the environment, but the intrinsic mechanism may be mysterious. The policy characterizes the distribution of actions at each possible state. The problem is how to design a policy for the agent to maximize the total expected reward along a trajectory induced by the policy. The state-of-the-art model-free methods for reinforcement learning [28, 33] can be

* Received January 7, 2021 / Accepted April 22, 2021 /
Published online November 16, 2021 /
1) Corresponding author

divided into policy-based and value-based methods. Policy-based methods directly learn or try to approximate the optimal policy by policy improvement and policy evaluation alternatively. They generate a map, i.e., a distribution from states to actions, which can be stochastic or deterministic. That is, they can be applied to both continuous and discrete action spaces. While in value-based methods, the goal is approximating the solution of the optimal Bellman equation based upon the temporal difference learning [33]. They learn a value function defined on the state-action pairs to estimate the maximal expected return of the action taken in the state. Then at each sate, the optimal policy based on the value function predicts a single action by maximizing the values.

The recent progress of deep neural networks [23] provides many scalable and reliable learning based approaches [8, 11, 24, 29, 31] for solving large and complex real-world problems in reinforcement learning. The curse of dimensionality is conquered by expressing the value and/or policy function with a deep neural network from high-dimensional or limited sensory inputs. The deepening expedites the evolution of end-to-end reinforcement learning, also referred as deep reinforcement learning. As a representative and illuminative algorithm in deep value-based methods, deep Q-learning (DQN) [25] has succeeded in many discrete domains such as playing Atari games. The learned agent arrives at a comparable level to that of a professional human games player. They construct a Q-network to receive the raw pictures as inputs, and optimize the weights by minimizing the Bellman residual. DQN can be viewed as a deep value iteration method directly, and some independent improvements including their combinations have been summarized in [14]. The success of DQN and its variants has a restriction on the type of the problem, specifically, the maximal operator in the objective function makes the optimization to be less reliable in continuous and/or large action space. By representing the greedy action selection with a policy network, the deep deterministic policy gradient (DDPG) method [24] successfully extends the algorithmic idea of DQN into the continuous action space. The value network imitates the training in DQN and the policy network is updated by maximizing the estimated values. The two delayed deep deterministic (TD3) policy gradient algorithm [9] substantially improves DDPG by building double deep Q-networks to avoid overestimation in value estimates and delaying the policy updates to reduce the per-update error in DDPG.

Different from the optimization models based on value functions, policy-based algorithms also concentrate on optimizing the policy iteratively. In the policy improvement step, the actor updates the policy by optimizing an appropriate objective function using gradient-based methods. Policy evaluation creates a critic, i.e., a value function, to assess the policy by minimizing the Bellman error associated with the policy, which provides a basis for policy improvement. Thus the policy-based methods are usually classified as actor-critic methods. As the optimization is practically based on the observations, the generalized advantage estimators (GAE) [30] are mostly considered for the bias-variance tradeoff and numerical stability. The discrepancy among the state-of-the-art policy-based methods mainly locates in the actor part, specifically, the surrogate function used for improving the policy. The trust region policy optimization (TRPO) [29] generalizes the proof the policy improvement bound in [16] into general stochastic policies and proposes a trust region model for policy update. The model function is a local approximation of the total expected reward and the Kullback-Leibler (KL) divergence between two policies is considered as a distance constraint. The subproblem under parameterization is highly nonlinear and nonconvex rather than a typical quadratic model as in [3, 6, 37] because the policy is parameterized by a neural network and the trust region constraint is replaced by a distance function of two policies. In order to develop a practical

algorithm, the subproblem is approximately solved in one step by a linearization of the model function and a second-order approximation of the constraint. The proximal policy optimization (PPO) algorithm [31] constructs a surrogate function by modifying the model function in TRPO with a clipped probability ratio, which controls the policy update from one iteration to the next one as the KL divergence constraint does. The surrogate function is maximized by the stochastic gradient methods. Some related developments can be found in [2, 5, 20].

In this paper, we use a similar model as in TRPO and develop a model-free on-policy algorithm within the stochastic trust region framework for policy optimization. In TRPO, the subproblem is solved essentially by only one natural gradient step and the size of the trust region radius is always fixed. However, in our method, the subproblem is approximately solved by a sequence of increasing and feasible directions, and the solving process is terminated until the model function has a sufficient increase and the constraint is satisfied. Additionally, we update the trust region radius adaptively by linearly dependent on the gradient of the model and the adjustment is guided by the coherence between the surrogate function and the total expected reward. We add the empirical standard deviation of the total expected reward to the predicted increase in a ratio to alleviate the bias caused by sampling in function estimations. This revision can be interpreted as releasing the coherence into a related confidence interval. For problems with continuous action spaces, where the Gaussian policy is commonly used, the mean of the policy tends to be sharply updated toward the best observation accompanied with an unexpected drop in variance. To avoid from the premature convergence, we separate the optimization with respect to the mean and variance into two independent parts. This alternating strategy slows down the decline of the entropy to encourage exploration which is crucial in deep reinforcement learning. In the convergence analysis, we provide a theoretical foundation for the trust region framework on policy optimization. With respect to the unparameterized policies, we show that the proposed trust region algorithm generates a sequence of monotonically rising policies. Moreover, specifying the total variation (TV) distance in trust region constraint, we can construct a feasible solution for the subproblem such that the model function has a sufficient improvement. We prove that the ratio can be bounded from below and it goes to one as the trust region radius goes to zero. Therefore, the total expected reward converges to the optimal value under some mild assumptions. Then we extend the results with policy parameterization. Specifically, we consider the Gaussian polices and the KL divergence in continuous action space. The alternating strategy in mean and covariance updates ensures a sufficient improvement in the model function, and the ratio is bounded below. Furthermore, under some continuous assumptions of the objective function, the mean of the policy is proved to converge to a stationary point where the covariance is also assumed to converge. The numerical performance of our stochastic trust region algorithm which is motivated from the theoretical analysis can be significantly better than that of the state-of-the-art methods including TRPO and PPO under certain typical environments.

In Section 2, we introduce basic preliminaries in deep reinforcement learning, and review some related algorithms. The trust region framework for policy optimization is proposed in Section 3.1. In Section 3.2, we give a convergence analysis of our algorithmic framework for problems with parameterized and unparameterized policies, respectively. The alternating strategy for the Gaussian policies is included in the proof. To handle practical problems, we present a stochastic trust region algorithm for policy optimization named STRO in Section 4. Finally, extensive numerical experiments are presented in Section 5 to demonstrate the effectiveness and robustness of our algorithm in MuJoCo [35] and Atari games [4].

2. Preliminaries

2.1. Notations

We consider the reinforcement learning problem defined by an infinite-horizon discounted Markov decision process (MDP). The MDP is denoted as a tuple $(\mathcal{S}, \mathcal{A}, P, r, \rho_0, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are known as the state and action spaces, respectively. $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbf{R}$ is the transition probability distribution, $r : \mathcal{S} \times \mathcal{A} \to \mathbf{R}$ is the bounded reward function, $\rho_0 : \mathcal{S} \to \mathbf{R}$ is the distribution of the initial state s_0, and $\gamma \in (0, 1)$ is the discount factor.

In this paper, we consider the stochastic policy, which maps the state into a distribution over action space:

$$\pi(a|s) : \ \mathcal{S} \times \mathcal{A} \to [0, 1], \quad \sum_{a \in \mathcal{A}} \pi(a|s) = 1, \quad \forall s \in \mathcal{S}.$$

The total expected reward is the expectation of the cumulative discounted reward of a trajectory induced by the policy π:

$$\eta(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right], \tag{2.1}$$

where $s_0 \sim \rho_0, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot|s_t, a_t)$. Moreover, we introduce the unnormalized discounted visitation frequency with respect to π as:

$$\rho_\pi(s) = \sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s_t = s|\pi).$$

We take the following standard definitions of the action value function Q_π and the state value function V_π:

$$Q_\pi(s, a) = \mathbb{E}_\pi \left[\sum_{l=0}^{\infty} \gamma^l r(s_{t+l}, a_{t+l}) \Big| s_0 = s, a_0 = a \right],$$

$$V_\pi(s) = \mathbb{E}_\pi \left[\sum_{l=0}^{\infty} \gamma^l r(s_{t+l}, a_{t+l}) \Big| s_0 = s \right].$$

They satisfy the recursive relationships:

$$Q_\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{P(\cdot|s,a)} \left[V_\pi(s') \right], \ V_\pi(s) = E_{\pi(\cdot|s)} \left[Q_\pi(s, a) \right].$$

The advantage of an action a over a given state s is defined as

$$A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s).$$

2.2. The optimization model

Generally, we consider the optimization

$$\max_{\pi \in \Pi} \ \eta(\pi), \tag{2.2}$$

where $\Pi = \{\pi | \sum_{a \in \mathcal{A}} \pi(a|s) = 1, \pi(a|s) \geq 0, \forall a \in \mathcal{A}, s \in \mathcal{S}\}$. For any fixed policy π, the total expected reward of another policy $\tilde{\pi}$ can be expressed in terms of the expectation of advantage function $A_\pi(s, a)$ over policy $\tilde{\pi}$:

$$\eta(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s, a). \tag{2.3}$$

The derivation can be referred in [16, 29]. The complex dependency of $\rho_{\tilde{\pi}}$ on $\tilde{\pi}$ motivates a popular surrogate function [29] by approximating $\rho_{\tilde{\pi}}$ with ρ_π:

$$\begin{aligned} L_\pi(\tilde{\pi}) &= \eta(\pi) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a) \\ &= \eta(\pi) + \mathbb{E}_{\rho_\pi,\pi}\left[\frac{\tilde{\pi}(a|s)}{\pi(a|s)} A_\pi(s,a)\right]. \end{aligned} \tag{2.4}$$

The second equality follows from importance sampling, where the sampling distribution is π. Apparently, L_π is a linear function in the policy space and intersects η at π. The maximizer of L_π over $\tilde{\pi}$ is exactly the greedy policy update at π in policy iteration, since for each state s, the action with the maximal advantage value is assigned with probability one:

$$\tilde{\pi}^* = \arg\max_{\tilde{\pi}\in\Pi} L_\pi(\tilde{\pi}),\ \tilde{\pi}^*(a|s) = \begin{cases} 1, & a = \arg\max_{a'} A_\pi(s,a'), \\ 0, & \text{otherwise.} \end{cases}$$

Besides, the evaluation of L_π at any policy $\tilde{\pi}$ only requires the expectation over π, which is much cheaper than that of η.

In deep reinforcement learning, to address the high dimensionality and complexity in real-world tasks, the policy π is usually parameterized by a set of variables, for example, a differentiable neural network weighted by θ. The constraint on the policy, i.e., $\pi \in \Pi$, is guaranteed by the parameterization. For simplicity, we can overload our previous notations to associate with θ rather than π, e.g., $\eta(\theta) := \eta(\pi_\theta)$, $Q_\theta(s_t,a_t) := Q_{\pi_\theta}(s_t,a_t)$, $L_\theta(\tilde{\theta}) := L_\pi(\tilde{\pi})$ and $\rho_\theta(s) := \rho_{\pi_\theta}(s)$. The goal is maximizing the total expected reward (2.1) in the parameterized policy space:

$$\max_\theta\ \eta(\theta). \tag{2.5}$$

The optimal solution θ^* corresponds to the so-called optimal policy π_{θ^*}, which equivalently implies that for any $s \in \mathcal{S},\ a \in \mathcal{A}$:

$$Q_{\theta^*}(s,a) \geq Q_\theta(s,a), \quad V_{\theta^*}(s) \geq V_\theta(s), \quad \forall\theta.$$

In general, for any MDP with a differentiable policy π_θ, the policy gradient [34] is formulated as

$$\nabla\eta(\theta) = \sum_s \rho_\theta(s) \sum_a \nabla\pi_\theta(a|s) A_\theta(s,a) = \mathbb{E}_{\rho_\theta,\pi_\theta}\Big[\nabla \log \pi_\theta(a|s) A_\theta(s,a)\Big]. \tag{2.6}$$

Intuitively, $L_\theta(\tilde{\theta})$ matches $\eta(\tilde{\theta})$ up to the first-order accuracy at θ in the parameterized policy space:

$$L_\theta(\theta) = \eta(\theta), \quad \nabla L_\theta(\theta) = \nabla\eta(\theta). \tag{2.7}$$

2.3. Related algorithms

Policy gradient type methods [17, 33, 34] directly take a stochastic gradient version of (2.6) to update the policy in an incremental manner:

$$\theta_{k+1} = \theta_k + \alpha M(\theta_k)\nabla\eta(\theta_k), \tag{2.8}$$

where $\alpha > 0$ is the step size, and $M(\theta_k)$ is a preconditioning matrix that may associate with θ_k. The policy gradient type algorithms distinguish from each other with the choice of the preconditioning matrix $M(\theta_k)$ [10]. The vanilla policy gradient (VPG) [34] method using $M(\theta_k) = I$

often suffers poor-scaled issues. The natural policy gradient (NPG) algorithm [17] brings the natural gradient techniques originated from the neural networks and takes $M(\theta_k)$ as the inverse of the Fisher information matrix (FIM) of the policy π_{θ_k}, that is,

$$M(\theta_k)^{-1} = \mathbb{E}_{\rho_{\theta_k}, \pi_{\theta_k}} \left[\nabla \log \pi_{\theta_k}(s,a) \nabla \log \pi_{\theta_k}(s,a)^T \right]. \tag{2.9}$$

It can be showed as defining a matrix norm on the parameterized policy space. Generally, a well-tuned update rule for the step size α in (2.8) is crucial for the numerical stability in both VPG and NPG.

In TRPO, they propose to optimize a surrogate function coherent to the total expected reward with a constraint on the KL divergence of two policies, which controls how far the policy is allowed to update. At each iteration, they construct a trust region subproblem

$$\max_{\theta} \quad L_{\theta_k}(\theta), \quad \text{s.t.} \quad \mathbb{E}_{s\sim\rho_{\theta_k}}[D_{KL}(\pi_{\theta_k}(\cdot|s)||\pi_\theta(\cdot|s))] \le \delta, \tag{2.10}$$

where δ is a fixed constant and the KL divergence $D_{KL}(p||q) = \sum\limits_x p(x) \log \frac{p(x)}{q(s)}$. To develop a practical algorithm for solving the subproblem, they take a linear approximation of the model function and a second-order approximation of the constraint. Essentially, TRPO solves the quadratic constrained optimization:

$$\max_{\theta} \quad \nabla L_{\theta_k}(\theta_k)^T(\theta - \theta_k), \quad \text{s.t.} \quad \frac{1}{2}(\theta - \theta_k)^T H(\theta_k)(\theta - \theta_k) \le \delta, \tag{2.11}$$

where $H(\theta_k)$ is the FIM of the policy π_{θ_k} as well as the second-order approximation of the KL divergence at θ_k. Then the solution of (2.11) obtained by the conjugate gradient method [27, 32] is taken as an approximate solution of (2.10). Obviously, the induced update direction is collinear with that of NPG, therefore TRPO can be viewed as a natural policy gradient algorithm with self-adaptive step size.

PPO optimizes the model function in (2.10) with point-wise value clipping to penalize a large policy update. They construct a clipped surrogate function at each iteration:

$$L_k^{clip}(\theta) = \mathrm{E}_{\rho_{\theta_k}, \pi_{\theta_k}} \left[\min(r_k(s,a) A_{\theta_k}(s,a), \mathrm{clip}(r_k(s,a), 1-\epsilon, 1+\epsilon) A_{\theta_k}(s,a)) \right],$$

where $r_k(s,a) = \frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}$ is the probability ratio, $\mathrm{clip}(x, r_1, r_2) = \min(\max(x, r_1), r_2)$, and ϵ is a hyper-parameter. It is also a local estimate of the total expected reward, and has stringent control of the policy update. The maximization of $L_k^{clip}(\theta)$ is approximately solved by multiple epochs of the stochastic gradient methods.

Some recent works focus on applying mirror descent update in solving (2.5). A stochastic mirror descent with a new policy gradient estimator that merges historical gradient information is proposed in [38]. An objective function in a mirror descent style is considered in [36]. At each iteration, multiple SGD updates are performed to approximately solve the optimization problem. Moreover, the mirror descent update in regularized RL is considered in [21].

3. A Trust Region Method for Policy Optimization

As we showed in the last section, the function $L_{\theta_k}(\theta)$ gives a good local approximation to $\eta(\theta)$ around π_{θ_k} as in (2.7) and it has much lower costs than η in evaluation and derivation.

These advantages motivate an underlying exploration of L_{θ_k} to extract potential information for policy improvement. At the k-th iteration, we construct the trust region model as:

$$\max_{\theta} \quad L_{\theta_k}(\theta), \quad \text{s.t. } \mathbb{E}_{s\sim\rho_{\theta_k}}\left[D(\pi_{\theta_k}(\cdot|s), \pi_\theta(\cdot|s))\right] \leq \delta_k, \tag{3.1}$$

where D is a general metric function of two distributions, such as the KL divergence and TV distance. The subproblem (2.10) in TRPO can be viewed as a special case of ours by taking the KL divergence as the distance metric function. We embed the subproblem (3.1) into a general trust region method, so as to monitor the acceptance of the trial point and to adjust the trust region radius δ_k.

3.1. Algorithmic Framework

We now state the trust region framework for policy improvement with a random initialization. At the k-th iteration, the algorithm approximately solves the subproblem (3.1) to obtain a trial point $\tilde{\theta}_{k+1}$, then we resolve two issues: 1) whether to accept $\tilde{\theta}_{k+1}$ as θ_{k+1}; 2) how to update the trust region radius δ_k adaptively. The canonical trust region framework leads us to compute a ratio to evaluate the agreement between the objective function and the surrogate function at $\tilde{\theta}_{k+1}$:

$$r_k = \frac{\eta(\tilde{\theta}_{k+1}) - \eta(\theta_k)}{L_{\theta_k}(\tilde{\theta}_{k+1}) - L_{\theta_k}(\theta_k)}. \tag{3.2}$$

The trial point $\tilde{\theta}_{k+1}$ is accepted if r_k is greater than some positive constant β_0 and it is called a successful iteration, i.e., $\theta_{k+1} = \tilde{\theta}_{k+1}$, otherwise, the iteration is unsuccessful and $\theta_{k+1} = \theta_k$:

$$\theta_{k+1} = \begin{cases} \tilde{\theta}_{k+1}, & r_k \geq \beta_0, \\ \theta_k, & \text{otherwise.} \end{cases} \tag{3.3}$$

The adjustment of the trust region radius δ_k is based on the ratio as:

$$\delta_{k+1} = \begin{cases} \gamma_1\delta_k, & r_k \geq \beta_1, \\ \gamma_2\delta_k, & r_k \in [\beta_0, \beta_1), \\ \gamma_3\delta_k, & \text{otherwise,} \end{cases} \tag{3.4}$$

where $0 < \beta_0 < \beta_1$, and $0 < \gamma_3 < \gamma_2 \leq 1 < \gamma_1$. These tuning parameters control the accuracy of the model by determining how aggressively the trust region radius is updated when an iteration is successful or not. The training process is summarized in Algorithm 3.1.

Algorithm 3.1. A Trust Region Optimization Framework

Require: Set θ_0, δ_0, $k = 0$
1: **while** stopping criterion not met **do**
2: solve (3.1) to obtain a trail point $\tilde{\theta}_{k+1}$;
3: compute the ratio r_k via (3.2);
4: update θ_{k+1} using (3.3);
5: update δ_{k+1} using (3.4);
6: $k = k + 1$;
7: **end while**

3.2. Theoretical Analysis

We next construct the convergence of the trust region framework for policy optimization. The analysis starts from the unparameterized case, then moves to the case that the policy is parameterized as Gaussian distributions. During the discussion, the states set $\mathcal{S}$ is supposed to be finite and the initial state distribution ρ_0 is assumed that

$$\rho_0(s) > 0, \quad \forall s \in \mathcal{S}.$$

Consequently, for any policy π, it holds

$$\rho_\pi(s) \geq \rho_0(s) > 0, \quad \forall s \in \mathcal{S}, \tag{3.5}$$

where $\rho_\pi(\cdot)$ is not a distribution, i.e., $\sum\limits_s \rho_\pi(s) = \frac{1}{1-\gamma}$.

For general policy gradient methods, some theoretical properties conditioned on appropriate value estimation refer to [19, 34, 39]. For mirror descent methods and the KL-divergence regularized methods, the convergence analysis can be found in [21, 26, 38]. As the optimization model in our trust region method is quite different from these works, the analysis technique below are different.

3.2.1. Unparameterized Policy

In this part, we consider a MDP with a finite set of actions $\mathcal{A}$, and focus on the policy π itself, i.e., considering the case that $\theta = \{\pi(a|s) : \forall s \in \mathcal{S}, a \in \mathcal{A}\}$, which means $\pi_\theta = \pi$. Note that our analysis can be similarly extended to continuous action spaces (Eucliean space). Typically, we consider the following subproblem

$$\max_\pi L_{\pi_k}(\pi), \quad \text{s.t.} \quad \mathbb{E}_{s\sim\rho_{\pi_k}}[D_{TV}(\pi_k(\cdot|s)||\pi(\cdot|s))] \leq \delta_k, \tag{3.6}$$

where $\mathrm{D}_{TV}(p||q) = \frac{1}{2}\sum_x |p(x) - q(x)|$ is the total variation distance between two distributions p and q. Usually, a close-form solution of (3.6) is unknown. We next define the so-called policy advantage to introduce the theoretical results that follow.

Definition 3.1 (Policy Advantage). *The policy advantage $\mathbb{A}_\pi(\pi')$ of a policy π' with respect to a policy π is defined by*

$$\mathbb{A}_\pi(\pi') = \mathrm{E}_{s\sim\rho_\pi}\left[\mathrm{E}_{a\sim\pi'(\cdot|s)}\left[A_\pi(s,a)\right]\right].$$

From the definition, it is straightforward to obtain $L_{\pi_k}(\pi) = \eta(\pi_k) + \mathbb{A}_{\pi_k}(\pi)$. In the next lemma, we give an optimality condition of the RL problem (2.2).

Lemma 3.1. *The policy π is an optimal solution for (2.2) if and only if*

$$\mathbb{A}_\pi^* \triangleq \max_{\pi'} \ \mathbb{A}_\pi(\pi') = \sum_s \rho_\pi(s) \max_{\pi'(\cdot|s)} \sum_a \pi'(a|s) A_\pi(s,a) = 0, \tag{3.7}$$

i.e., $\pi \in \mathrm{argmax}_{\pi'} \ \mathbb{A}_\pi(\pi')$.

Proof. Since $\sum\limits_a \pi(a|s)A_\pi(s,a) = 0$ for any s and (3.5) holds for π, it is obviously that the condition (3.7) is equivalent to, for any policy π',

$$\sum_a \pi'(a|s)A_\pi(s,a) \leq 0, \quad \forall s. \tag{3.8}$$

We first prove the sufficiency part. Combining the conditions (3.5) and (3.8) for any policy π', we have that

$$\eta(\pi') = \eta(\pi) + \sum_s \rho_{\pi'}(s) \sum_a \pi'(a|s) A_\pi(s,a) \leq \eta(\pi).$$

Hence, π is an optimal solution of (2.2).

We then prove the necessary part by contradiction. Suppose that (3.8) is not satisfied. Then there exists a state s and policy π' such that

$$\sum_a \pi'(a|s') A_\pi(s,a) > 0.$$

Define a new policy $\tilde{\pi}$ as

$$\tilde{\pi}(a|s) = \begin{cases} \pi'(a|s), & s = s', \\ \pi(a|s), & s \neq s'. \end{cases}$$

Since (3.5) holds for $\tilde{\pi}$, we obtain

$$\begin{aligned} \eta(\tilde{\pi}) &= \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a) \\ &= \eta(\pi) + \rho_{\tilde{\pi}}(s') \sum_a \pi'(a|s') A_\pi(s',a) > \eta(\pi), \end{aligned}$$

i.e., π is not an optimal solution of (2.2), which completes the proof. □

We next show a lower bound of improvement for the function L in each step.

Lemma 3.2. *Suppose $\{\pi_k\}$ is the sequence generated by the trust region method, then we have*

$$L_{\pi_k}(\pi_{k+1}) - L_{\pi_k}(\pi_k) \geq \min(1, (1-\gamma)\delta_k) \mathbb{A}^*_{\pi_k}.$$

Proof. If the optimal solution of subproblem (3.6) lies in the trust region, i.e.,

$$\mathbb{E}_{s\sim\rho_{\pi_k}}[D_{TV}(\pi_k(\cdot|s)||\pi_{k+1}(\cdot|s))] < \delta_k,$$

we obtain

$$\pi_{k+1} = \text{argmax}_\pi \ L_{\pi_k}(\pi) = \text{argmax}_\pi \ \mathbb{A}_{\pi_k}(\pi).$$

In other words, $L_{\pi_k}(\pi_{k+1}) - L_{\pi_k}(\pi_k) = \mathbb{A}^*_{\pi_k}$.

If the optimal solution of subproblem (3.6) reaches the boundary, it means that $\pi^*_{k+1} = \text{argmax}_\pi \ \mathbb{A}_{\pi_k}(\pi)$ is outside the trust region. Take $\hat{\pi}_{k+1}$ as a feasible convex combination of π_k and π^*_{k+1}, that is, set $\beta = (1-\gamma)\delta_k$ and

$$\hat{\pi}_{k+1} = (1-\beta)\pi_k + \beta \pi^*_{k+1}.$$

Consequently, we have

$$\begin{aligned} &\mathbb{E}_{s\sim\rho_{\pi_k}}[D_{TV}(\pi_k(\cdot|s)||\hat{\pi}_{k+1}(\cdot|s))] = \sum_s \rho_{\pi_k}(s) D_{TV}(\pi_k(\cdot|s)||\hat{\pi}_{k+1}(\cdot|s)) \\ =&\beta \sum_s \rho_{\pi_k}(s) D_{TV}(\pi_k(\cdot|s)||\pi^*_{k+1}(\cdot|s)) \leq \frac{\beta}{1-\gamma} = \delta_k, \end{aligned}$$

where the inequality is based on the facts that the total variation distance is uniformly bounded by one and $\sum_s \rho_{\pi_k}(s) = \frac{1}{1-\gamma}$. Then, it follows that

$$\begin{aligned} &L_{\pi_k}(\pi_{k+1}) - L_{\pi_k}(\pi_k) \geq L_{\pi_k}(\hat{\pi}_{k+1}) - L_{\pi_k}(\pi_k) \\ =&\sum_s \rho_{\pi_k}(s) \sum_a \left(\hat{\pi}_{k+1}(a|s) - \pi_k(a|s)\right) A_{\pi_k}(s,a) \\ =&\beta \sum_s \rho_{\pi_k}(s) \sum_a \pi^*_{k+1}(a|s) A_{\pi_k}(s,a) = (1-\gamma)\delta_k \mathbb{A}^*_{\pi_k}, \end{aligned}$$

which completes the proof. □

The next lemma shows a lower bound of the trust region ratio r_k. If the advantage function $\mathbb{A}^*_\pi$ is lower bounded, then as the trust region radius goes to zero, we have the ratio goes to one.

Lemma 3.3. *The ratio r_k defined in* (3.2) *satisfies that*

$$r_k \geq \min\left(1 - \frac{4\bar{A}\gamma\delta_k^2}{p_0^2(1-\gamma)^2\mathbb{A}^*_{\pi_k}}, 1 - \frac{4\bar{A}\gamma\delta_k}{p_0^2(1-\gamma)^3\mathbb{A}^*_{\pi_k}}\right),$$

where $p_0 = \min_s \rho_0(s)$ and $\bar{A} = \max_{s,a,\pi} |A_\pi(s,a)|$.

Proof. It follows from [29, Theorem 1] that

$$\eta(\pi_{k+1}) \geq L_{\pi_k}(\pi_{k+1}) - \frac{4\bar{A}\gamma\alpha^2}{(1-\gamma)^2}, \tag{3.9}$$

where $\alpha = \max_s D_{TV}(\pi_k(\cdot|s)||\pi_{k+1}(\cdot|s))$. Thus,

$$\begin{aligned} r_k &= \frac{\eta(\pi_{k+1}) - \eta(\pi_k)}{L_{\pi_k}(\pi_{k+1}) - L_{\pi_k}(\pi_k)} \\ &\geq 1 - \frac{4\bar{A}\gamma\alpha^2}{(1-\gamma)^2(L_{\pi_k}(\pi_{k+1}) - L_{\pi_k}(\pi_k))} \\ &\geq 1 - \frac{4\bar{A}\gamma\alpha^2}{(1-\gamma)^2\min(1,(1-\gamma)\delta_k)\mathbb{A}^*_{\pi_k}}. \end{aligned}$$

Since the relationship

$$\delta_k \geq \mathbb{E}_{s\sim\rho_{\pi_k}}[D_{TV}(\pi_k(\cdot|s)||\pi_{k+1}(\cdot|s))] \geq p_0 \max_s [D_{TV}(\pi_k(\cdot|s)||\pi_{k+1}(\cdot|s))]$$

holds, i.e., $\alpha \leq \frac{\delta_k}{p_0}$, then we have that

$$r_k \geq 1 - \frac{4\bar{A}\gamma\delta_k^2}{p_0^2(1-\gamma)^2\min(1,(1-\gamma)\delta_k)\mathbb{A}^*_{\pi_k}}.$$

This completes the proof of the lemma. □

In particular, the inequality (3.9) only provides some descent properties of the objective function η, but it cannot guarantee the convergence directly. Finally, we show our main theorem.

Theorem 3.1 (Convergence). *Suppose that $\{\pi_k\}$ is a sequence generated by our trust region method, then we have the following conclusions.*

1. $\liminf_{k\to\infty} \mathbb{A}^*_{\pi_k} = 0$.
2. $\lim_{k\to\infty} \eta(\pi_k) = \eta(\pi^*)$, *where* π^* *is an optimal solution of* (2.2).

Proof. We prove the first statement by contradiction. Suppose that $\exists\ \epsilon > 0$ and $K \in \mathbb{N}$ such that

$$\mathbb{A}^*_{\pi_k} \geq \epsilon, \quad \forall k > K.$$

Without loss of generality, we can assume $(1-\gamma)\delta_k < 1$ holds for any $k > K$. Then the results in Lemma 3.2 and 3.3 reduce to

$$L_{\pi_k}(\pi_{k+1}) - L_{\pi_k}(\pi_k) \geq (1-\gamma)\delta_k\epsilon \quad \text{and} \quad r_k \geq 1 - \frac{4\bar{A}\gamma\delta_k}{p_0^2(1-\gamma)^3\epsilon},$$

respectively. Take $\bar{\delta} = \frac{p_0^2(1-\gamma)^3\epsilon(1-\beta_1)}{4\bar{A}\gamma}$. Once $\delta_k \leq \bar{\delta}$, then

$$r_k \geq \beta_1 \quad \text{and} \quad \delta_{k+1} \geq \delta_k.$$

Hence, we obtain

$$\delta_k \geq \min(\delta_K, \gamma_2\bar{\delta}), \quad \forall k > K. \tag{3.10}$$

We next claim that $r_k \geq \beta_1$ occurs infinite many times. If not, then there exists $K_1 > K$ such that $r_k < \beta_1$ and $\delta_{k+1} < \delta_k$, $\forall k > K_1$, which conflicts with (3.10). Therefore, we have

$$\eta(\pi_{k+1}) - \eta(\pi_k) = r_k\left(L(\pi_{k+1}) - L(\pi_k)\right) \geq (1-\gamma)\delta_k\epsilon\beta_0, \quad \forall k > K.$$

Since $\eta(\pi_k)$ is monotone and bounded, we have $\delta_k \to 0$ which is a contradiction.

Denote a subsequence $\{k_n\}$ such that $\lim_{n\to\infty} \mathbb{A}^*_{\pi_{k_n}} = 0$. Then the continuity of $\mathbb{A}^*_\pi$ with respect to π indicates that $\lim_{n\to\infty} \pi_{k_n} = \pi^*$. Consequently, the subsequence $\{\eta(\pi_{k_n})\}$ converges since η is continuous with respect to π. Therefore, the full sequence $\{\eta(\pi_k)\}$ converges since $\{\eta(\pi_k)\}$ is monotone and bounded. □

3.2.2. Parameterized Policy

We consider the parameterization of the policy in this part. We mainly focus on the continuous action space and restrict the policy into the Gaussian distribution.

Assumption 3.1. The policy is assumed to be a Gaussian distribution with state-dependent mean vector $\mu(s) \in \mathbb{R}^n$ and a state-independent covariance matrix $\Sigma = \mathrm{diag}(\sigma^2)$, where $\sigma \in \mathbb{R}^n$ is the standard deviation vector and is assumed to be bounded below, i.e. $\sigma^{(i)} \geq \underline{\sigma} > 0, i = 1, \cdots, n$.

The parameterization is $\theta = \{\mu(s), \sigma\}_{s\in\mathcal{S}}$ and the policy $\pi_{\theta_k}(\cdot|s)$ obeys the Gaussian distribution $\mathcal{N}(\mu_k(s), \sigma_k^2)$. For simplicity, in the following discussion, we often use μ to represent the notation $\mu(s)$ by ignoring s. The discussion is based on the model (3.1) with the KL divergence:

$$\max_{\mu,\sigma}\ L_{\mu_k,\sigma_k}(\mu,\sigma), \quad \text{s.t.} \quad \mathrm{E}_{s\sim\rho_{\mu_k,\sigma_k}} D_{KL}(\pi_{\mu_k,\sigma_k}(\cdot|s)||\pi_{\mu,\sigma}(\cdot|s)) \leq \delta_k. \tag{3.11}$$

Since an exact maximizer of (3.11) may be hard to compute, we use an alternative strategy to get an approximate solution. Firstly, by fixing σ_k, we update μ_k by solving the subproblem

$$\max_{\mu}\ L_{\mu_k,\sigma_k}(\mu,\sigma_k), \quad \text{s.t.} \quad \mathrm{E}_{s\sim\rho_{\mu_k,\sigma_k}} D_{KL}(\pi_{\mu_k,\sigma_k}(\cdot|s)||\pi_{\mu,\sigma_k}(\cdot|s)) \leq \delta_k. \tag{3.12}$$

The KL divergence between two Gaussian policies with the same standard deviation can be written as

$$D_{KL}(\pi_{\mu_k,\sigma_k}(\cdot|s)||\pi_{\mu,\sigma_k}(\cdot|s)) = \frac{1}{2}(\mu-\mu_k)^T\Sigma_k^{-1}(\mu-\mu_k).$$

Then, we update σ_k from

$$\max_{\sigma}\ L_{\mu_k,\sigma_k}(\mu_{k+1},\sigma),\quad \text{s.t.}\quad \mathrm{E}_{s\sim\rho_{\mu_k,\sigma_k}} D_{KL}(\pi_{\mu_k,\sigma_k}(\cdot|s)||\pi_{\mu_{k+1},\sigma}(\cdot|s)) \le \delta_k. \tag{3.13}$$

We also make some assumptions on the smoothness of the function L.

Assumption 3.2. The Hessian matrix $\nabla^2_\mu L_{\mu_k,\sigma_k}(\mu,\sigma_k)$ is uniformly bounded within the trust region, i.e., $h_k = \max_{\mu\in\mathcal{B}_k} \|\nabla^2_\mu L_{\mu_k,\sigma_k}(\mu,\sigma_k)\| + 1 \le h$ for $\forall k$, where $\mathcal{B}_k = \{\mu|\mathrm{E}_{s\sim\rho_{\mu_k,\sigma_k}} D_{KL}(\pi_{\mu_k,\sigma_k}(\cdot|s)||\pi_{\mu,\sigma_k}(\cdot|s)) \le \delta_k\}$.

According to the definitions of the model function and the Gaussian density function, the Hessian matrix is

$$\nabla^2_\mu L_{\mu_k,\sigma_k}(\mu,\sigma_k) = \sum_s \rho_{\mu_k,\sigma_k}(s)\sum_a \pi_{\mu,\sigma_k}(a|s)\frac{1}{\sigma_k^2}\left[\frac{1}{\sigma_k^2}(a-\mu)(a-\mu)^T - I\right] A_{\mu_k,\sigma_k}(s,a).$$

Note that the advantage function $|A_{\mu,\sigma}(s,a)| \le \bar{A}$ for any s,a,μ and σ. In consideration of Assumption 3.1, the triangle inequality and the consistency of the matrix norm, we have

$$\begin{aligned}
&\|\nabla^2_\mu L_{\mu_k,\sigma_k}(\mu,\sigma_k)\| \\
\le&\sum_s \rho_{\mu_k,\sigma_k}(s)\sum_a \pi_{\mu,\sigma_k}(a|s)\frac{1}{\sigma_k^2}\left[\frac{1}{\sigma_k^2}(a-\mu)^T(a-\mu) + \|I\|\right]|A_{\mu_k,\sigma_k}(s,a)| \\
\le&\sum_s \rho_{\mu_k,\sigma_k}(s)\frac{\bar{A}}{\underline{\sigma}^4}\mathrm{E}_{\mu,\sigma_k}[(a-E(a))^T(a-E(a))] + \sum_s \rho_{\mu_k,\sigma_k}(s)\frac{\bar{A}}{\underline{\sigma}^2}\|I\| \\
=&\frac{1}{1-\gamma}\frac{\bar{A}}{\underline{\sigma}^2}(\dim(\mathcal{A}) + \|I\|).
\end{aligned}$$

The last equality follows from the fact that, for the multi-dimensional Gaussian distribution with diagonal and scalar variance matrix, the expectation

$$\mathrm{E}_{\mu,\sigma}[(a-E(a))^T(a-E(a))] = \sigma^2\dim(\mathcal{A}),$$

where $\mathcal{A}$ is the action space.

The next lemma shows that the solution of the subproblem (3.12) provides a lower bounded model improvement.

Lemma 3.4. *Suppose that Assumptions* $3.1-3.2$ *hold. The maximizer of* (3.12) *satisfies that*

$$L_{\mu_k,\sigma_k}(\mu_{k+1},\sigma_k) - L_{\mu_k,\sigma_k}(\mu_k,\sigma_k) \ge \|g_k\|^2 \min\left(\frac{1}{2h_k}, \frac{\underline{\sigma}\sqrt{\delta_k}}{2\|g_k\|}\right), \tag{3.14}$$

where $g_k = \nabla_\mu L_{\mu_k,\sigma_k}(\mu_k,\sigma_k)$.

Proof. Take $\mu^c_{k+1} := \mu_k + \tau g_k$, where $\tau \in (0, \frac{\underline{\sigma}\sqrt{\delta_k}}{\|g_k\|}]$. Then we have $\mu^c_{k+1} \in \mathcal{B}_k$ and the Taylor expansion of L_{μ_k,σ_k} with Lagrange's form of the remainder indicates that

$$\begin{aligned}
L_{\mu_k,\sigma_k}(\mu^c_{k+1},\sigma_k) =& L_{\mu_k,\sigma_k}(\mu_k,\sigma_k) + \tau\|g_k\|^2 + \frac{1}{2}\tau^2\left\langle g_k, \nabla^2 L_{\mu_k,\sigma_k}(\hat{\mu}_{k+1})g_k\right\rangle \\
\ge& L_{\mu_k,\sigma_k}(\mu_k,\sigma_k) + (\tau - \frac{1}{2}\tau^2 h_k)\|g_k\|^2,
\end{aligned} \tag{3.15}$$

where $\hat{\mu}_{k+1} \in \mathcal{B}_k$ lies between μ_k and μ_{k+1}^c. If $\frac{1}{h_k} \leq \frac{\underline{\sigma}\sqrt{\delta_k}}{\|g_k\|}$, by taking $\tau = \frac{1}{h_k}$, we can obtain

$$L_{\mu_k,\sigma_k}(\mu_{k+1}^c, \sigma_k) - L_{\mu_k,\sigma_k}(\mu_k, \sigma_k) \geq \frac{\|g_k\|^2}{2h_k}. \tag{3.16}$$

Otherwise $\frac{1}{h_k} > \frac{\underline{\sigma}\sqrt{\delta_k}}{\|g_k\|}$, we set $\tau = \frac{\underline{\sigma}\sqrt{\delta_k}}{\|g_k\|}$ and get

$$L_{\mu_k,\sigma_k}(\mu_{k+1}^c, \sigma_k) - L_{\mu_k,\sigma_k}(\mu_k, \sigma_k) \geq \underline{\sigma}\sqrt{\delta_k}\|g_k\| - \frac{1}{2}\underline{\sigma}^2\delta_k h_k \geq \frac{1}{2}\underline{\sigma}\sqrt{\delta_k}\|g_k\|. \tag{3.17}$$

It follows from (3.16) and (3.17) that at the point μ_{k+1}^c,

$$L_{\mu_k,\sigma_k}(\mu_{k+1}^c, \sigma_k) - L_{\mu_k,\sigma_k}(\mu_k, \sigma_k) \geq \|g_k\|^2 \min\left(\frac{1}{2h_k}, \frac{\underline{\sigma}\sqrt{\delta_k}}{2\|g_k\|}\right), \tag{3.18}$$

which completes the proof. □

Then we show a lower bound of the trust region ratio.

Lemma 3.5. *Suppose that Assumptions* 3.1 *and* 3.2 *hold. The ratio defined in* (3.2) *satisfies that*

$$r_k \geq 1 - \frac{8\bar{A}\gamma\delta_k^2 \max(\frac{\|g_k\|}{\underline{\sigma}\sqrt{\delta_k}}, h_k)}{p_0^2(1-\gamma)^2\|g_k\|^2}, \tag{3.19}$$

where $p_0 = \min_s \rho_0(s)$ *and* $\bar{A} = \max_{s,a,\mu,\sigma} |A_{\mu,\sigma}(s,a)|$.

Proof. The explicit formulation of the ratio yields

$$\begin{aligned}
r_k =& \frac{\eta(\mu_{k+1}, \sigma_{k+1}) - \eta(\mu_k, \sigma_k)}{L_{\mu_k,\sigma_k}(\mu_{k+1}, \sigma_{k+1}) - L_{\mu_k,\sigma_k}(\mu_k, \sigma_k)} \\
=& \frac{\eta(\mu_{k+1}, \sigma_{k+1}) - L_{\mu_k,\sigma_k}(\mu_{k+1}, \sigma_{k+1}) + L_{\mu_k,\sigma_k}(\mu_{k+1}, \sigma_{k+1}) - \eta(\mu_k, \sigma_k)}{L_{\mu_k,\sigma_k}(\mu_{k+1}, \sigma_{k+1}) - L_{\mu_k,\sigma_k}(\mu_k, \sigma_k)} \\
\geq& \frac{-\frac{4\bar{A}\gamma\delta_k^2}{p_0^2(1-\gamma)^2} + L_{\mu_k,\sigma_k}(\mu_{k+1}, \sigma_{k+1}) - L_{\mu_k,\sigma_k}(\mu_k, \sigma_k)}{L_{\mu_k,\sigma_k}(\mu_{k+1}, \sigma_{k+1}) - L_{\mu_k,\sigma_k}(\mu_k, \sigma_k)} \\
\geq& 1 - \frac{4\bar{A}\gamma\delta_k^2}{p_0^2(1-\gamma)^2\|g_k\|^2 \min\left(\frac{1}{2h_k}, \frac{\underline{\sigma}\sqrt{\delta_k}}{2\|g_k\|}\right)} \\
\geq& 1 - \frac{8\bar{A}\gamma\delta_k^2 \max(\frac{\|g_k\|}{\underline{\sigma}\sqrt{\delta_k}}, h_k)}{p_0^2(1-\gamma)^2\|g_k\|^2},
\end{aligned} \tag{3.20}$$

where the first inequality follows from [29, Theorem 1] and $\eta(\mu_k, \sigma_k) = L_{\mu_k,\sigma_k}(\mu_k, \sigma_k)$, and the second inequality is derived from Lemma 3.4 and Assumption 3.1. □

Finally, we establish our main convergence result. Ideally, the optimal policy is deterministic, as well as the optimal Gaussian policy, i.e., the mean vector is the optimal action at each state and the standard deviation is zero. It implies that the best action is usually grasped by the mean parameter and the standard deviation plays a role in exploration. Meanwhile, zero variance also induces unbounded values in both numerical experiment and theoretical analysis. Hence, in practice, we only care the learn of the mean vector and make the final standard deviation small enough. Therefore, we concentrate on the behavior of $\|\nabla_\mu \eta(\mu, \sigma)\|$.

Theorem 3.2. *Suppose that Assumptions* 3.1 *and* 3.2 *hold and* $\{(\mu_k, \sigma_k)\}$ *is the sequence generated by the trust region method. Then we have the following conclusions.*

1. *The limit inferior of the norm of gradient with respect to* μ *goes to zero, i.e.,*

$$\liminf_{k\to\infty} \|\nabla_\mu \eta(\mu_k, \sigma_k)\| = 0.$$

2. *Suppose that the gradient* $\nabla_\mu \eta(\mu, \sigma)$ *is Lipschitz continuous with respect to* μ *and* σ*, respectively. If the standard deviation vector* σ_k *converges to* σ^**, i.e.* $\lim_{k\to\infty} \sigma_k = \sigma^*$*, then we have*

$$\lim_{k\to\infty} \|\nabla_\mu \eta(\mu_k, \sigma^*)\| = 0.$$

Remark 3.1. The second conclusion indicates that supposing the standard deviation converges to σ^*, then the mean vector converges to the optimal mean with respect to σ^*. In other words, the algorithm converges in the space of the Gaussian policy and the converged mean vector is close to the optimal action if σ^* is small enough.

Proof. We prove the first statement by contradiction. Suppose that we can find $\epsilon > 0$ and $K \in \mathbb{N}$ such that

$$\|\nabla_\mu \eta(\mu_k, \sigma_k)\| = \|g_k\| \geq \epsilon, \quad \forall k > K. \tag{3.21}$$

Combining (3.21) with the result in Lemma 3.5, we have

$$\begin{aligned} r_k &\geq 1 - \frac{8\bar{A}_k \gamma \delta_k^2 \max(\frac{\|g_k\|}{\underline{\sigma}\sqrt{\delta_k}}, h_k)}{p_0^2 (1-\gamma)^2 \|g_k\|^2} \\ &\geq \min\left(1 - \frac{8\bar{A}\gamma\delta_k^{\frac{3}{2}}}{p_0^2(1-\gamma)^2\|g_k\|\underline{\sigma}}, 1 - \frac{8\bar{A}\gamma\delta_k^2 h}{p_0^2(1-\gamma)^2\|g_k\|^2}\right) \\ &\geq \min\left(1 - \frac{8\bar{A}\gamma\delta_k^{\frac{3}{2}}}{p_0^2(1-\gamma)^2\epsilon\underline{\sigma}}, 1 - \frac{8\bar{A}\gamma\delta_k^2 h}{p_0^2(1-\gamma)^2\epsilon^2}\right) \end{aligned}$$

Take

$$\bar{\delta} = \min\left(\left(\frac{p_0^2(1-\beta_1)(1-\gamma)^2\bar{\sigma}\epsilon}{8\bar{A}\gamma}\right)^{\frac{2}{3}}, \left(\frac{p_0^2(1-\beta_1)(1-\gamma)^2\epsilon^2}{8\bar{A}\gamma h}\right)^{\frac{1}{2}}\right).$$

It follows that for each $\delta_k \leq \bar{\delta}$, the ratio $r_k \geq \beta_1$. According to the update rule (3.4), we have

$$\delta_k \geq \min(\delta_K, \gamma_2 \bar{\delta}), \quad \forall k > K. \tag{3.22}$$

Now, we claim that $r_k \geq \beta_1$ occurs infinite many time. Otherwise, we can find $K_1 > K$ such that for any $k > K_1$, $r_k < \beta_1$ and $\delta_k > \delta_{k+1}$, which conflicts with (3.22). Therefore, we can obtain for $\forall k > K$ and $r_k \geq \beta_0$,

$$\begin{aligned} \eta(\mu_{k+1}, \sigma_{k+1}) - \eta(\mu_k, \sigma_k) &= r_k \left(L_{\mu_k,\sigma_k}(\mu_{k+1}, \sigma_{k+1}) - L_{\mu_k,\sigma_k}(\mu_k, \sigma_k)\right) \\ &\geq r_k \left(L_{\mu_k,\sigma_k}(\mu_{k+1}, \sigma_k) - L_{\mu_k,\sigma_k}(\mu_k, \sigma_k)\right) \\ &\geq \beta_0 \|g_k\|^2 \min\left(\frac{1}{2h}, \frac{\underline{\sigma}\sqrt{\delta_k}}{2\|g_k\|}\right) \\ &\geq \beta_0 \min\left(\frac{\epsilon^2}{2h}, \frac{\underline{\sigma}\epsilon\sqrt{\delta_k}}{2}\right), \end{aligned}$$

where the first inequality is due to the definition of the trust region ratio, the second inequality comes from the update rule of σ (3.13) and the third inequality is from Lemma (3.4). Since η is continuous and upper bounded, we have $\delta_k \to 0$ as $k \to \infty$ which contradicts to (3.22).

In order to construct a contradiction for the second statement, we assume that there exists some $\epsilon > 0$ such that $\|g_{k_i}\| \geq 2\epsilon$, where $\{k_i\}$ is a subsequence of successful iterates. Denote the indices of all successful iterates by $\mathcal{M}$. From the first statement and its proof, we can obtain another subsequence of $\mathcal{M}$, denoted by $\{t_i\}$, where t_i is the first successful iterate such that $t_i > k_i$ and $\|g_{t_i}\| \leq \epsilon$. Let $\mathcal{Q} = \{q \in \mathcal{M} | k_i \leq q < t_i, \text{for some } i = 1, 2, ...\}$. Then for any $q \in \mathcal{Q}$ we have

$$\eta(\mu_{q+1}, \sigma_{q+1}) - \eta(\mu_q, \sigma_q) \geq \beta_0 \min\left(\frac{\epsilon^2}{2h}, \frac{\underline{\sigma}\epsilon\sqrt{\delta_q}}{2}\right).$$

Obviously, the sequence $\{\eta(\mu_k, \sigma_k)\}_{k\in\mathcal{M}}$ is monotonically increasing and upper bounded. Hence, we obtain

$$\lim_{\substack{q\in\mathcal{Q}\\ q\to\infty}} \delta_q = 0, \qquad \lim_{\substack{q\in\mathcal{Q}\\ q\to\infty}} \eta(\mu_q, \sigma_q) = \eta^*.$$

Without loss of generality, we can assume $\frac{\underline{\sigma}\sqrt{\delta_q}}{2} \leq \frac{\epsilon}{2h}$ for all $q \in \mathcal{Q}$. Then for each i, we have

$$\begin{aligned}
&\eta^* - \eta(\mu_{k_i}, \sigma_{k_i}) \geq \eta(\mu_{t_i}, \sigma_{t_i}) - \eta(\mu_{k_i}, \sigma_{k_i}) \\
\geq &\sum_{\substack{q\in\mathcal{Q},\\ k_i\leq q<t_i}} \beta_0 \frac{\underline{\sigma}\epsilon\sqrt{\delta_q}}{2} \geq \frac{\beta_0\underline{\sigma}\epsilon}{2\bar{\sigma}} \sum_{\substack{q\in\mathcal{Q},\\ k_i\leq q<t_i}} \|\mu_{q+1} - \mu_q\| \geq \frac{\beta_0\underline{\sigma}\epsilon}{2\bar{\sigma}}\|\mu_{t_i} - \mu_{k_i}\|,
\end{aligned}$$

where $\bar{\sigma}$ is a upper bound of each element in σ_k which exists due to the convergence of σ_k. Thus, we derive $\|\mu_{t_i} - \mu_{k_i}\| \to 0$, and consequently $\|g_{t_i} - g_{k_i}\| \to 0$, which yields a contradiction to $\|g_{t_i} - g_{k_i}\| \geq \epsilon$ because of the definitions of $\{k_i\}$ and $\{t_i\}$. It concludes that

$$\lim_{k\to\infty} \|\nabla_\mu \eta(\mu_k, \sigma_k)\| = 0.$$

Then we have

$$\|\nabla_\mu \eta(\mu_k, \sigma^*)\| \leq \|\nabla_\mu \eta(\mu_k, \sigma^*) - \nabla_\mu \eta(\mu_k, \sigma_k)\| + \|\nabla_\mu \eta(\mu_k, \sigma_k)\| \to 0,$$

where the first term goes to zero due to the convergence of σ_k and the Lipschitz continuity of $\nabla\eta$. □

In conclusion, we establish the global convergence for trust region method in unparameterized case, and prove that the Gaussian policy class for continuous environments converges to the stationary point. Although the optimal Gaussian policy is supposed to be deterministic, we make an assumption on the lower bound of the variance of the policy due to the analytical and numerical issues. If the the limit of the variance is small enough, then Theorem 3.2 implies that the trust region method converges. In [22], a linear expected regret is proved in TRPO. Actually, this is consistent with our conclusions. The automatic adjustment of the trust region radius, rather than fixed radius in TRPO, is the basis of the convergence of the trust region method.

4. A Stochastic Trust Region Algorithm

The functions discussed before are constructed with expectations and they have to be estimated using sample averages in practice. To be consistent with the notations above, we add a

hat to represent the corresponding estimated value, such as $\hat{\eta}$, $\hat{L}_{\theta_k}$ and so on. Essentially, the optimization in Algorithm 4.1 is constructed with these estimated functions. In consideration of the uncertainty and fluctuations caused by the samples, we develop a stochastic version of Algorithm 4.1 in the following discussion.

4.1. Sample Collection and Advantage Estimation

The original objective function $\eta(\theta)$ denotes the expectation of the cumulative discounted rewards over a trajectory generated by the policy π_θ. Practically, we simulate N interactions with the environment from policy π_{θ_k} to generate τ complete trajectories whose total rewards are denoted as $\{R_i\}_{i=1}^{\tau}$, using Algorithm 4.1. All visited states and actions during the interactions are gathered into a set of sample pairs $B_{\theta_k} = \{(s_i, a_i)\}_{i=1}^{N}$ for estimation. The averaged total reward of these trajectories:

$$\hat{\eta}(\theta_k) = \frac{1}{\tau}\sum_{i=1}^{\tau} R_i \tag{4.1}$$

is called as an empirical estimation of $\eta(\theta_k)$. The sample standard deviation

$$\hat{\sigma}_\eta(\theta_k) = \sqrt{\frac{1}{\tau-1}\sum_{i=1}^{\tau}(R_i - \hat{\eta}(\theta_k))^2} \tag{4.2}$$

characterizes the sample fluctuation near the mean value. The expectations can be approximated with respect to the samples as follows:

$$\hat{L}_{\theta_k}(\theta, B_{\theta_k}) = \hat{\eta}(\theta_k) + \frac{1}{|B_{\theta_k}|}\sum_{(s,a)\in B_{\theta_k}} \frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}\hat{A}_{\theta_k}(s,a), \tag{4.3a}$$

$$\hat{g}_k(\theta, B_{\theta_k}) = \frac{1}{|B_{\theta_k}|}\sum_{(s,a)\in B_{\theta_k}} \frac{\nabla\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}\hat{A}_{\theta_k}(s,a), \tag{4.3b}$$

$$\hat{D}_k(\theta, B_{\theta_k}) = \frac{1}{|B_{\theta_k}|}\sum_{(s,a)\in B_{\theta_k}} D(\pi_{\theta_k}(\cdot|s), \pi_\theta(\cdot|s)), \tag{4.3c}$$

where the advantage estimator $\hat{A}_{\theta_k}$ is constructed using the empirical reward and the value network. We take a version of GAE as:

$$\hat{A}_{\theta_k}(s,a) = r(s,a) + \gamma V_{\phi_k}(s') - V_{\phi_k}(s) + \gamma\lambda\hat{A}_{\theta_k}(s',a'), \tag{4.4}$$

where V_ϕ is the state value function parameterized by ϕ and s' is the state after the transfer from s and a. The hyper-parameter λ controls the trade-off between bias and variance. The value network is trained by minimizing the error between the target value and the predicted value:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|B_{\theta_k}|}\sum_{(s,a)\in B_{\theta_k}} \|V_{\phi_k}(s) + \hat{A}_{\theta_k}(s,a) - V_\phi(s)\|^2. \tag{4.5}$$

Fitting such a value network as the baseline function is the same as the other policy-based methods in deep reinforcement learning [15].

Algorithm 4.1. SAMPLE(θ,N)

$B_\theta = \emptyset$; $\tau = 0$, $R_\tau = 0, t = 0$; randomly initiate the state s_0;
for $i = 1, \cdots, N$ **do**
  if s_t is a terminal state **then**
    $\tau = \tau + 1$; $R_\tau = 0, t = 0$; randomly initiate the state s_0;
  else
    perform one step at s_t using $\pi_\theta(\cdot|s_t)$ and obtain reward $r_t := r(s_t, a_t, s_{t+1})$;
    $R_\tau = R_\tau + \gamma^t r_t$;
    $t = t + 1$;
    $B_\theta \leftarrow B_\theta \cup (s_t, a_t)$;
  end if
end for
$\hat{\eta}(\theta) = \frac{1}{\tau}\sum_{j=1}^{\tau} R_j$;
return B_θ

4.2. Solving the Trust Region Subproblem

Empirically, the trust region model for policy optimization at the k-th iteration is

$$\max_\theta \hat{L}_{\theta_k}(\theta, B_{\theta_k}), \quad \text{s.t.} \quad \hat{D}_k(\theta, B_{\theta_k}) \le \delta_k. \tag{4.6}$$

It is worth noting that the policy distance function in (4.6) can be approximately treated as a re-weighted matrix norm on parameters. In other words, it holds for any metric function D and sample set B:

$$\hat{D}_k(\theta, B) \approx \frac{1}{2}(\theta - \theta_k)^T \nabla^2 \hat{D}_k(\theta_k, B)(\theta - \theta_k),$$

since $\hat{D}_k(\theta, B) \ge 0$, $\nabla^2 \hat{D}_k(\theta_k, B)$ is semi-positive definite, and the first-order Taylor expansion at $\theta = \theta_k$ is equal to zero.

We now present a feasible stochastic method for solving (4.6) by sequentially generating increasing directions using the CG algorithm. The process starts from $\theta_{k,1} = \theta_k$. At the l-th inner iteration, we randomly sample a subset $b_l \subseteq B_{\theta_k}$ to obtain $\hat{g}_k(\theta_{k,l}, b_l)$ and $\hat{H}_k(b_l) = \nabla^2 \hat{D}_k(\theta_k, b_l)$, and compute the direction

$$d_l = \hat{H}_k^{-1}(b_l)\hat{g}_k(\theta_{k,l}, b_l).$$

Then we update the parameter as

$$\theta_{k,l+1} = \theta_{k,l} + \alpha d_l \tag{4.7}$$

such that the following two conditions are satisfied:

$$\hat{L}_{\theta_k}(\theta_{k,l+1}, b_l) \ge \hat{L}_{\theta_k}(\theta_{k,l}, b_l) + \tau d_l^T \hat{g}_k(\theta_{k,l}, b_l), \quad \text{and} \quad \hat{D}_k(\theta_{k,l+1}, B_{\theta_k}) \le \delta_k, \tag{4.8}$$

where $\tau \in (0, 1)$. The conditions in (4.8) ensure an improvement of $\hat{L}_{\theta_k}$ on b_l and the constraint in (4.6) holds at each step.

The theoretical analysis in Lemma 3.2 suggests that the improvement of L_{θ_k} is supposed to be larger than $(1 - \gamma)\delta_k A^*_{\theta_k}$. However, it is unaccessible in practice. Empirically, we terminate

the iteration and denote the trial point $\tilde{\theta}_{k+1} := \theta_{k,l+1}$ once the following conditions hold:

$$\frac{|\hat{L}_{\theta_k}(\theta_{k,l+1}, B_{\theta_k}) - \hat{L}_{\theta_k}(\theta_{k,l}, B_{\theta_k})|}{1 + |\hat{L}_{\theta_k}(\theta_{k,l}, B_{\theta_k})|} \leq \epsilon \quad \text{or} \quad \frac{|\mathrm{Ent}(\theta_{k,l+1}, B_{\theta_k}) - \mathrm{Ent}(\theta_k, B_{\theta_k})|}{1 + |\mathrm{Ent}(\theta_k, B_{\theta_k})|} \geq \epsilon, \tag{4.9}$$

where $\epsilon > 0$ is a small constant and $\mathrm{Ent}(\theta_k, B_{\theta_k})$ is the sample-averaged entropy of the policy π_{θ_k}, i.e.,

$$\mathrm{Ent}(\theta_k, B_{\theta_k}) = \frac{1}{|B_{\theta_k}|} \sum_{s \in B_{\theta_k}} \mathcal{H}(\pi_{\theta_k}(\cdot|s)) \quad \text{and} \quad \mathcal{H}(p) = -\int p(x) \log p(x) dx.$$

The entropy characterizes the randomness and the exploration ability of the policy. Under the random circumstances, the policy inadvertently collapses on the best observed action with respect to the estimated advantage value which may be significantly wrong. Although the distance constraint recedes this effect to some extent, the policy is driven to favor the best actions it has visited, albeit slowly [1]. Based on these concerns, we take the second condition in (4.9) to prevent excessively dependency on the advantage estimators and limited observations. The pseudocode for solving (4.6) is outlined in Algorithm 4.2.

Algorithm 4.2. InnerSolu(θ_k,δ_k,B_{θ_k},ξ,τ)

1: $\theta_{k,1} = \theta_k$;
2: **for** $l = 1, 2, \ldots$ **do**
3: randomly sample data $b_l \subseteq B_{\theta_k}$;
4: compute $\hat{g}_k(\theta_{k,l}, b_l)$ and $\hat{H}_k(b_l)$ by (4.3);
5: compute $d_l = \hat{H}_k^{-1}(b_l)\hat{g}_k(\theta_{k,l}, b_l)$ using the CG algorithm;
6: set $\theta_{k,l+1} = \theta_{k,l} + \alpha d_l$ such that (4.8) holds;
7: **if** $l \ \% \ l_0 = 0$ **then**
8: If (4.9) holds, break;
9: **end if**
10: **end for**
11: **return** $\tilde{\theta}_{k+1} = \theta_{k,l+1}$

4.3. Alternating Strategy For Gaussian Policy

For continuous action spaces, the multivariate Gaussian distribution is adopted in many cases. The neural network whose weight is denoted as θ^μ maps from state to the mean of the Gaussian distribution, $\mu(s; \theta^\mu)$. The covariance matrix is assumed to be a diagonal matrix. Namely, we take a state-independent vector θ^σ to represent the log-standard deviations instead of the standard deviations, since the log-standard deviations are free to take values from $(-\infty, \infty)$. Therefore, the policy is characterized by the Gaussian distribution

$$\pi_\theta(\cdot|s) \sim N(\mu(s; \theta^\mu), \mathrm{diag}(\exp(2\theta^\sigma))), \quad \forall s \in \mathcal{S},$$

where $\theta = (\theta^\mu, \theta^\sigma)$. It can be verified that the action can be characterized as:

$$a = \mu(s; \theta^\mu) + \exp(\theta^\sigma) \odot \varepsilon, \quad \varepsilon \sim N(0, I), \tag{4.10}$$

where $\odot$ is the element-wise product of two vectors. Ideally, the optimal Gaussian policy is supposed to be deterministic, that is, the mean is the optimal action and the covariance is zero.

As shown in (4.10), the covariance term affects the exploration of the samples as well as the quality of the optimization. An optimistic situation is that the log-standard deviations gradually decreases with the iterative process and is finally controlled by a lower bound, since extremely small covariance is undesirable for numerical stability in practice. In order to reduce the interactions between the mean and log-standard deviations in optimization and regulate the decay of the log-standard deviations more precisely, we can approximately solve the trust region problem (4.6) by alternating the update with respect to the mean and log-standard deviations independently. For the mean parameter, we formulate the trust region model as follows:

$$\max_{\theta^\mu} \ \hat{L}_{\theta_k}\left(\theta = (\theta^\mu, \theta_k^\sigma), B_{\theta_k}\right), \quad \text{s.t.} \quad \hat{D}_k(\theta, B_{\theta_k}) \le \delta_k, \tag{4.11}$$

As we assumed in the theoretical analysis, the update of the log-standard deviations is supposed to monotonically improve L_{θ_k} and ensure that the new policy is close to the current policy. For simplicity, we can update θ^σ by approximately solving

$$\max_{\theta^\sigma} \ \hat{L}_{\theta_k}\left(\theta = (\tilde{\theta}_{k+1}^\mu, \theta^\sigma), B_{\theta_k}\right), \quad \text{s.t.} \quad \|\theta_k^\sigma - \theta^\sigma\|_\infty \le \bar{A}_k, \tag{4.12}$$

where $\tilde{\theta}_{k+1}^\mu$ is the solution of (4.11). Then we denote the solution of (4.12) as $\tilde{\theta}_{k+1}^\sigma$ and the trial point $\tilde{\theta}_{k+1} = (\tilde{\theta}_{k+1}^\mu, \tilde{\theta}_{k+1}^\sigma)$.

Essentially, the infinite norm constraint controls the distance between two Gaussian policies and plays a similar role in regulating the entropy loss for the Gaussian policy since

$$\mathrm{Ent}(\theta_k, B_{\theta_k}) = 1^T \theta_k^\sigma + \frac{n}{2}(1 + \log 2\pi),$$

where n is the dimension of the action space. We can take the bound $\bar{A}_k$ related to the entropy of the current policy, i.e., $\epsilon_k \propto |\mathrm{Ent}(\theta_k, B_{\theta_k})|$. The subproblem (4.11) is solved by following the process in Algorithm 4.2, while the solution of the subproblem (4.12) is approximated using the projected gradient method. In our experiments, we find that the alternative update for the Gaussian policy is helpful in releasing the log-standard deviations from the extreme decline caused by the estimations and encouraging the mean parameter to update toward the accurate optimal actions.

4.4. Algorithmic Development

According to the Algorithm 3.1, if the iteration is successful, we exploit the samples for testing $\tilde{\theta}_{k+1}$ in the next iteration without additional simulations. However, once a rejection of $\tilde{\theta}_{k+1}$ arises, the policy remains unchanged, i.e., $\theta_{k+1} = \theta_k$. Apparently, the samples in B_{θ_k} can be used to construct the trust region model (4.6) in the next iteration. In such a situation, apart from updating the trust region radius δ_k using (3.4), we simulate another N sample pairs from $\pi_{\theta_{k+1}}$ and merge it with B_{θ_k} to be $B_{\theta_{k+1}}$. Thereby after a rejection, the sample size is enlarged and the estimations are supposed to be more accurate. Ideally, when the sample size is sufficiently large and the trust region gets small enough, an acceptance is very likely to occur with high probability. In consideration of data storage, once the sample size reaches an upper limit $N_{\max}$ after several consecutive rejections, a mandatory acceptance is enforced such that the best performed policy among the last few rejected iterations is taken as the next iteration:

$$(\theta_{k+1}, B_{\theta_{k+1}}) = \arg\max_{(z,T_z)\in H} \ \hat{\eta}(z), \tag{4.13}$$

where H is the set of rejected iterations and T_z is the sample set with respect to the policy π_z.

Undesirably, the landscape of η becomes much more noisy in low-sampled regime as the training progresses [15]. More seriously, the total expected rewards $\eta(\theta_k)$ and $\eta(\tilde{\theta}_{k+1})$ are estimated from different sample sets, which is different from that of the general deep learning problems. These challenges make the estimations to be accompanied with large variance. Therefore, we take the sample standard deviation into the ratio (3.2) for empirical stability:

$$r_k = \frac{\hat{\eta}(\tilde{\theta}_{k+1}) - \hat{\eta}(\theta_k)}{\hat{\sigma}_\eta(\theta_k) + \hat{L}_{\theta_k}(\tilde{\theta}_{k+1}, B_k) - \hat{L}_{\theta_k}(\theta_k, B_k)}. \tag{4.14}$$

As a generalization of the standard ratio in trust region methods, this revision checks the agreement between the objective function and surrogate function in a confidence interval.

In view of the oscillations under sampling, a small negative ratio is permitted in acceptance. Namely the update criterion for θ_{k+1} is modified from (3.3) as:

$$\theta_{k+1} = \begin{cases} \tilde{\theta}_{k+1}, & r_k \geq 0 > \beta_0, \\ \theta_k, & r_k \leq \beta_0 \ \text{ and } \ |B_{\theta_k}| < N_{\max}, \\ \hat{\theta}_{k+1}, & \text{otherwise.} \end{cases} \tag{4.15}$$

We select the trust region radius depending linearly on the norm of the gradient [37], i.e., $\delta_k = \mu_k \|g_k(\theta_k)\|$. The adjustment of the coefficient μ_k is based on the ratio as:

$$\mu_{k+1} = \begin{cases} \min(\gamma_1 \mu_k, \mu_{\max}), & r_k \geq \beta_1, \\ \max(\gamma_2 \mu_k, \mu_{\min}), & r_k \in [\beta_0, \beta_1), \\ \max(\gamma_3 \mu_k, \mu_{\min}), & \text{otherwise,} \end{cases} \tag{4.16}$$

where $\beta_1 \geq 0 > \beta_0$ and $\mu_{\max} > \mu_{\min} > 0$. The empirical algorithm is summarized in Algorithm 4.3.

Algorithm 4.3. STRO

Require: Set μ_0, θ_0, ϕ_0, N, N_{max}, σ, τ.
$k = 0$, $B_{\theta_k} = \text{SAMPLE}(\theta_k, N)$, $H = \emptyset$;
while stopping criterion not met **do**
  compute a trial point $\tilde{\theta}_{k+1} = \text{InnerSolu}(\theta_k, \delta_k, B_{\theta_k}, \sigma, \tau)$;
  $T_{\tilde{\theta}_{k+1}} = \text{SAMPLE}(\tilde{\theta}_{k+1}, N)$;
  compute the ratio r_k via (4.14);
  update μ_{k+1} by (4.16);
  if $r_k \geq \beta_0$ **then**
    $\theta_{k+1} = \tilde{\theta}_{k+1}$ and $B_{\theta_{k+1}} = T_{\tilde{\theta}_{k+1}}$;
  else if $|S_k| < N_{\max}$ **then**
    $\theta_{k+1} = \theta_k$ and $B_{\theta_{k+1}} = \text{SAMPLE}(\theta_{k+1}, N) \cup B_{\theta_k}$;
    $H = H \cup \{(\tilde{\theta}_{k+1}, T_{\tilde{\theta}_{k+1}})\}$;
  else
    update $(\theta_{k+1}, B_{\theta_{k+1}})$ by (4.13); $H = H \setminus \{(\theta_{k+1}, B_{\theta_{k+1}})\}$;
  end if
  update ϕ_k by (4.5);
  $k = k + 1$;
end while

5. Experiments

To investigate the effectiveness and robustness of our stochastic trust region method for policy optimization among the state-of-the-art deep reinforcement learning algorithms, we make a comprehensive comparison using OpenAI's Baselines [7] and Spinningup[1]. The discrepancy between the implementations in these two repositories are primarily in the preprocessing of the environments and the structure of the networks. In Baselines, some additional wrappers are included for normalizing the environment informations and the size of the networks varies for the algorithms, while in Spinningup the observations and the reward function are taken from the environments directly and the acquiescent network architecture is shared among the methods. As an on-policy algorithm, we compare our method with TRPO and PPO on a range of continuous control tasks and Atari game playings using the benchmark suite OpenAI Gym. In order to facilitate comparisons, we take the distance function in our method as the KL divergence since it is differentiable and is commonly used. We should point out that there is a gap between the implemented PPO in Baselines and the theoretical algorithm in [31]. Ilyas et al. [15] state that, the optimization which is not part of the core algorithm develop the practical success of PPO to a large extent and the theoretical framework might be negligible in practice. To study the learning capability of the policy-based and values-based methods, we take a comparison with DDPG, TD3, and soft actor-critic (SAC) method [12] which is an energy-based off-policy algorithm. They are known to perform well in continuous controls at the expense of large interactions and long training time. Since these methods are incapable to handle discrete problems directly, we only compare with them in continuous tasks.

5.1. Continuous Controls

We test nine representative robotic locomotion experiments using the simulator MuJoCo[2] in Gym. The problem is simulating a robot to win highest returns with fluent and safe movements. The unknown dynamics, non-smooth reward shape and the high dimensionality make the problems being challenging.

For these continuous tasks, we use the Gaussian distribution to characterize the conditional probability $\pi_\theta(a|s)$. As we described in section 4.3, the policy is defined by the normal distribution $N(\mu(s;\theta^\mu), \mathrm{diag}(\exp(2\theta^\sigma)))$. The mean function $\mu(\cdot;\theta^\mu)$ is parameterized by a neural network with $tanh$ units. Moreover, for the value network where the parameter is denoted as ϕ, we use the same architecture as the mean function except that the dimension of the last layer is one. We update the parameter $\theta = [\theta^\mu, \theta^\sigma]$ by the alternative models in (4.11) and (4.12), and train the value network using the Adam [18] method simultaneously.

During our experiments, we take $N = 2048$ for each simulation and set the initial trust region coefficient $\mu_0 = 0.05$, the adjustment factors $\gamma_1 = 2$, $\gamma_2 = 0.8$, $\gamma_3 = 0.6$, $\mu_{\max} = 0.1$ and $\mu_{\min} = 0.01$. Empirically, the trial point is accepted once $\eta(\tilde{\theta}_{k+1}) > \eta(\theta_k)$, i.e., $\beta_1 = 0$. The compulsive acceptance takes place after four consecutive rejections. Fig. 5.1 presents the training curves over these environments in Baselines. Each task is run for one million time steps over five random seeds of the network initialization and Gym simulator. The horizontal and vertical axes are training time steps and the empirical total expected rewards, respectively. The solid curves represent the mean values of five independent simulations and the shaded areas correspond to the standard deviation. Table 5.1 summaries the maximal averaged return and a single standard

[1] An educational resource produced by OpenAI, https://spinningup.openai.com.
[2] http://www.mujoco.org.

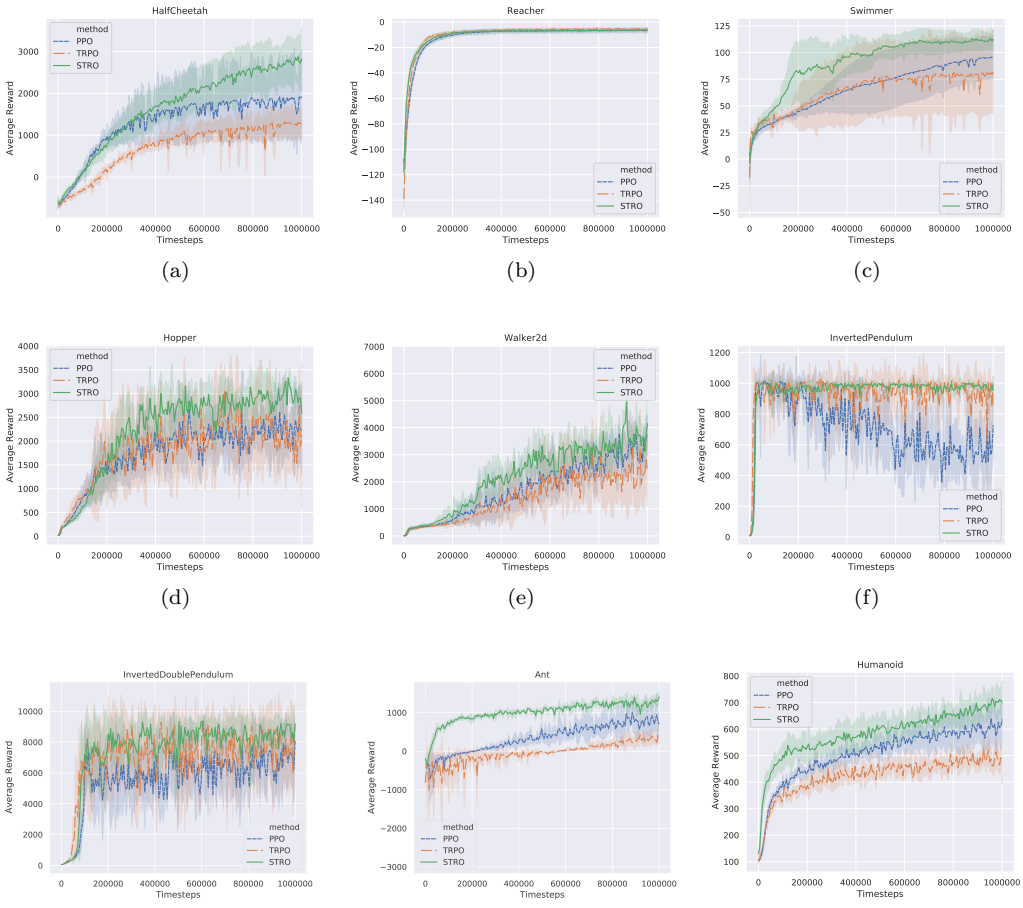


Fig. 5.1. Training curves on MuJoCo-v2 continuous control benchmarks under Baselines.

deviation over five trials. A higher maximal average return indicates that the algorithm has ability to capture the better agent. We find that our method outperforms or matches TRPO and PPO in almost the same amount of time over all experimented environments.

Table 5.1: Max Average Reward ± standard deviation over 5 trials of 1e6 time steps under Baselines. The last column lists the average running time of these three methods.

Env	PPO	TRPO	STRO	Time(min)
HalfCheetah	1816±798	1406±390	**2985**±491	36\35\54
Reacher	-5±1	-3±0	-4±0	34\38\53
Swimmer	94±18	97±24	**117**±9	36\43\46
Hopper	2475±91	3591±107	3463±81	35\43\48
Walker2d	3681±794	3935±936	**5849**±2063	39\43\51
InvPend	1000±0	1000±0	**1000**±0	41\33\49
InvDoubPend	9328±1	9340±7	**9346**±3	43\36\49
Ant	793±206	666±79	**1512**±101	43\43\46
Humanoid	679±64	627±11	**784**±36	47\44\ 56

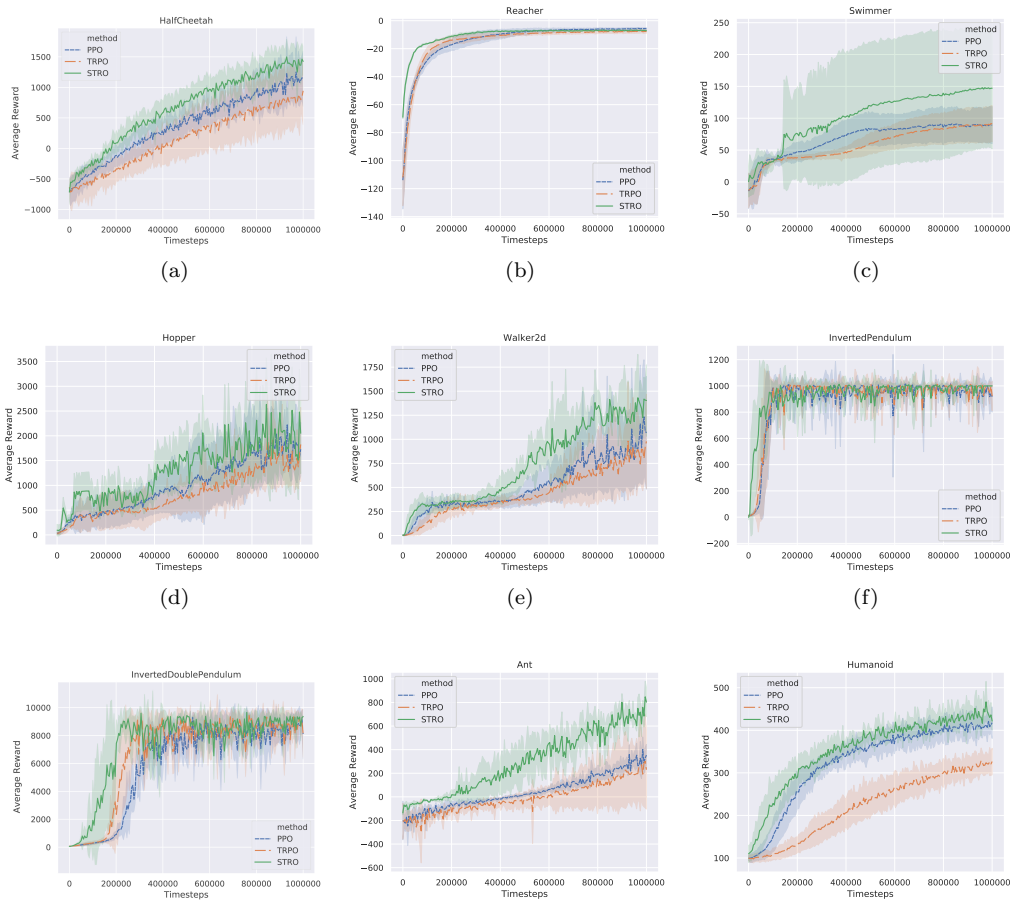
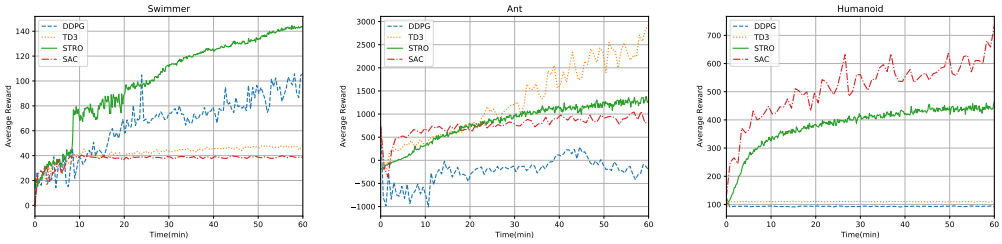


Fig. 5.2. Training curves on MuJoCo-v2 continuous control benchmarks with Spinningup.

As the recent studies [13, 15] demonstrate that the experimental techniques in the environment wrappers, such as observation normalization, reward clipping, etc., have a dramatic effect on the numerical performance, we make another comparison in Spinningup where no extra modification is applied in the environments and the network structure is shared among all algorithms. From the results in Fig 5.2, our algorithm still surpass TRPO and PPO on most experiments as in Baselines. Because the implementations are slightly different between Baselines and Spinningup, the values of the average reward in Fig 5.1 and Fig 5.2 are different in some tasks, even for the same algorithm. Generally, as illustrated in Table 5.1, our method takes around 50 minutes in Baselines for each task over one random seed on average, slightly slower than PPO and TRPO.

We now compare with DDPG, TD3 and SAC. For these methods, in addition to the one million time steps for deterministic evaluation, i.e., no noise in sampling, they require another several million interactions for training which may take about several hours, and even longer in complex problems. In other words, the empirical convergence rate of these methods is much slower than that of policy-based methods. Due to the remarkable difference in the computational time among these algorithms, we compare their learning ability within a specified time. Since



(a) (b) (c)

Fig. 5.3. One hour training curves on MuJoCo-v2 continuous control benchmarks under SpinningUp.

there is no standard implementation of TD3 and SAC in Baselines as a benchmark, we take the comparisons in Spinningup and terminate the algorithms after one hour training. We test these methods on nine robotic locomotion experiments. Specifically, the numerical results on *Swimmer*, *Ant* and *Humanoid* are reported in Fig 5.3. Our method is always better than DDPG on most environments except in $HalfCheetah$. Moreover, in some problems our algorithm can surpass SAC and TD3, and in others it is slightly inferior than the best performed method. The comparisons demonstrate that our method is comparable with state-of-the-art deep reinforcement learning algorithms under such time constraint. Notably, for the other three methods, the simulations are deterministic which has inherent advantages compared to ours, sometimes even significant.

Table 5.2: Max Average Reward ± standard deviation over 5 trials of 1e7 time steps.

Environment	PPO	TRPO	STRO
BeamRider	3113±375	760±30	**3836**±903
Bowling	61±11	58±18	**81**±18
FishingDerby	11±18	-82±2	**25**±0
MsPacman	2037±153	1538±159	**2558**±442
Pong	19±0	3±7	**20**±0
Seaquest	1100±317	692±92	**1396**±398
SpaceInvaders	965±108	540±20	**1010**±81
PrivateEye	100±0	88±16	100±0
Freeway	30±0	28±3	**31**±0

5.2. Discrete Control

To evaluate our method on discrete problems, we randomly test 9 Atari games which have partially observed states in the form of images and discrete action spaces. The complex observations, high dimensionality, delayed reward and many other challenging elements make them to be extremely tough to learn. Different from the robotic locomotion experiments, the game images are preprocessed and feed into a convolutional neural network with $softmax$ operator in the last layer to represent a categorical policy. The value network has a similar structure as the policy network except the dimension of the output. Our method is implemented in Baselines, and the network architecture is the same as the one in PPO. For all tested games, we take $N = 2048$ for each simulation and initialize the trust region coefficient $\mu_0 = 0.01$. We set

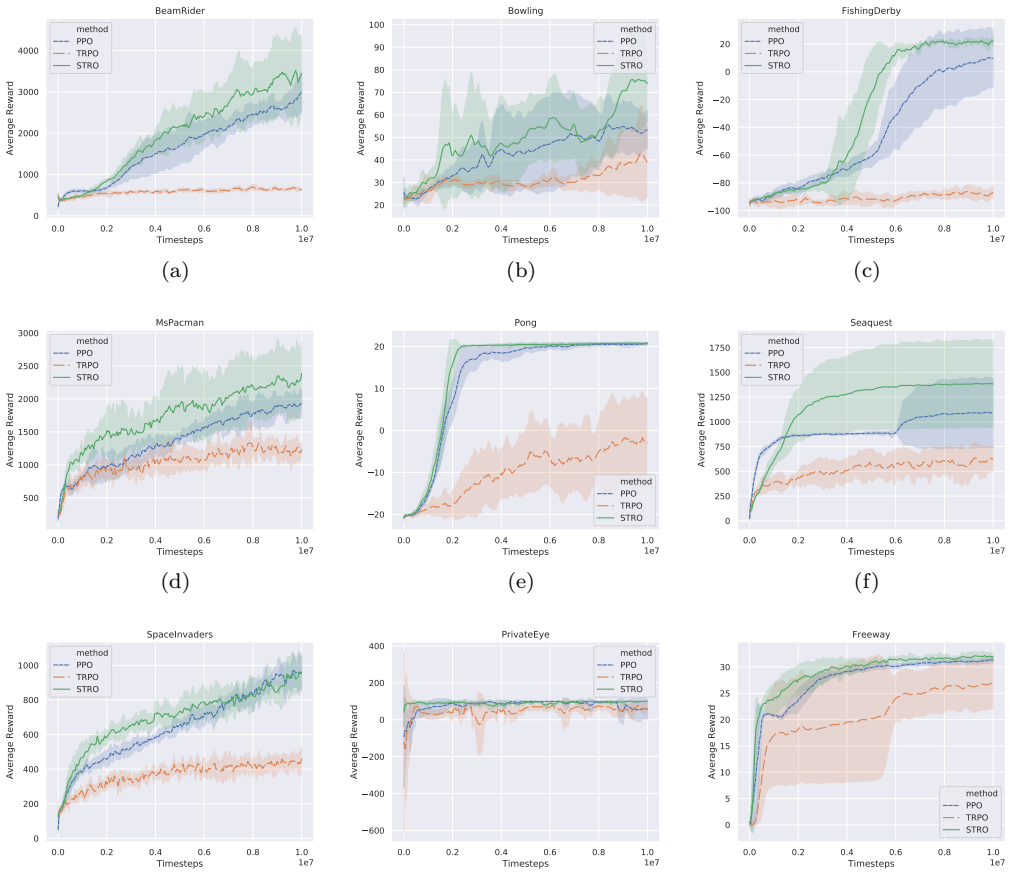


Fig. 5.4. Training curves on Atari games over 16 parallel programmings.

$\mu_{\max} = 0.5$ and $\mu_{\min} = 0.005$. The maximal average return over 100 episodes on several games is reported in Table 5.2, and the learning curves are plotted in Fig 5.4 to illustrate the generalization of our method. Generally, our algorithm reaches higher or almost the same average reward in all tested environments.

6. Conclusion

We propose a stochastic trust-region framework for policy optimization and a decoupled update for the Gaussian policy to avoid premature convergence. For the unparameterized policies, we prove the global convergence of the proposed algorithm under mild and feasible assumptions. Moreover, in the parameterized case, we show that the mean of the Gaussian policies converges to the stationary point where the covariance of the policies is assumed to converge. In robotic locomotion using a general-purpose policy network, we successfully learn better controllers than PPO and TRPO. Our method is able to surpass DDPG in most tested environments under given time constraints. Meanwhile, we show that our algorithm is comparable with the start-of-the-art deep reinforcement learning methods, such as TD3 and SAC. Our method is also suitable for discrete tasks such as playing some Atari games, and it can outperform PPO

and TRPO in quite a few tasks.

Acknowledgments. The computational results were obtained at GPUs supported by the National Engineering Laboratory for Big Data Analysis and Applications and the High-performance Computing Platform of Peking University.

References

[1] A. Abdolmaleki, J.T. Springenberg, J. Degrave, S. Bohez, Y. Tassa, D. Belov, N. Heess and Martin Riedmiller, Relative entropy regularized policy iteration. *arXiv:1812.02256*, 2018.

[2] J. Achiam, D. Held, A. Tamar and P. Abbeel, Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 22–31. JMLR. org, 2017.

[3] A.S. Bandeira, K. Scheinberg and Luís N Vicente, Convergence of trust-region methods based on probabilistic models. *SIAM Journal on Optimization*, **24**:3 (2014), 1238–1264.

[4] M. G Bellemare, Y. Naddaf, J. Veness and M. Bowling, The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, **47** (2013), 253–279.

[5] G. Chen, Y.M. Peng and M.J. Zhang, An adaptive clipping approach for proximal policy optimization. *arXiv preprint arXiv:1804.06461*, 2018.

[6] R.B. Chen, M. Menickelly and K. Scheinberg, Stochastic optimization using a trust-region method and random models. *Mathematical Programming*, **169**:2 (2018), 447–487.

[7] P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y.H. Wu and P. Zhokhov, Openai baselines. https://github.com/openai/baselines, 2017.

[8] Y. Duan, X. Chen, R. Houthooft, J. Schulman and P. Abbeel, Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, (2016), 1329–1338.

[9] S. Fujimoto, H. van Hoof and D. Meger. Addressing function approximation error in actor-critic methods. *arXiv:1802.09477*, 2018.

[10] T. Furmston, G. Lever and D. Barber, Approximate newton methods for policy search in Markov decision processes. *The Journal of Machine Learning Research*, **17**:1 (2016), 8055–8105.

[11] T, Haarnoja, H.R. Tang, P. Abbeel and S. Levine, Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1352–1361. JMLR. org, 2017.

[12] T. Haarnoja, A. Zhou, P. Abbeel and S. Levine, Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv:1801.01290*, 2018.

[13] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup and D. Meger, Deep reinforcement learning that matters. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[14] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar and D. Silver, Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[15] A. Ilyas, L. Engstrom, S. Santurkar, D. Tsipras, F. Janoos, L. Rudolph and A. Madry, Are deep policy gradient algorithms truly policy gradient algorithms? *arXiv:1811.02553*, 2018.

[16] S. Kakade and J. Langford, Approximately optimal approximate reinforcement learning. In *ICML*, **2** (2002), 267–274.

[17] S. M Kakade, A natural policy gradient. In *Advances in neural information processing systems*, (2002), 1531–1538.

[18] D. P Kingma and J. Ba, Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

[19] H. Kumar, A. Koppel and A. Ribeiro. On the sample complexity of actor-critic method for reinforcement learning with function approximation. *arXiv preprint arXiv:1910.08412*, 2019.

[20] T. Kurutach, I. Clavera, Y. Duan, A. Tamar and P. Abbeel, Model-ensemble trust-region policy optimization. *arXiv:1802.10592*, 2018.

[21] G.H Lan, Policy mirror descent for reinforcement learning: Linear convergence, new sampling complexity, and generalized problem classes. *arXiv preprint arXiv:2102.00135*, 2021.

[22] N. Lazić, B. Hao, Y. Abbasi-Yadkori, D. Schuurmans and C. Szepesvári, Optimization issues in kl-constrained approximate policy iteration. *arXiv preprint arXiv:2102.06234*, 2021.

[23] Y. LeCun, Y. Bengio and G. Hinton, Deep learning. *nature*, **521**:7553 (2015), 436.

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

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. A Rusu, J. Veness, M. G Bellemare, A. Graves, M. Riedmiller, A. K Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg and D. Hassabis, Human-level control through deep reinforcement learning. *Nature*, **518**:7540 (2015), 529.

[26] G. Neu, A. Jonsson and V. Gómez, A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*, 2017.

[27] J. Nocedal and S.J. Wright, Numerical optimization. 2006.

[28] B. Recht, A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2018.

[29] J. Schulman, S. Levine, P. Abbeel, M. Jordan and P. Moritz, Trust region policy optimization. In *International Conference on Machine Learning*, (2015), 1889–1897.

[30] J. Schulman, P. Moritz, S. Levine, M. Jordan and P. Abbeel, High-dimensional continuous control using generalized advantage estimation. *arXiv:1506.02438*, 2015.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford and O. Klimov, Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

[32] W.Y. Sun and Y.X. Yuan, *Optimization theory and methods: nonlinear programming*, volume 1. Springer Science & Business Media, 2006.

[33] R. S Sutton, A. G Barto, *Reinforcement learning: An introduction*. MIT press, 1998.

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

[35] E. Todorov, T. Erez and Y. Tassa, Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, (2012), 5026–5033. IEEE.

[36] M. Tomar, L. Shani, Y. Efroni and M. Ghavamzadeh, Mirror descent policy optimization. *arXiv preprint arXiv:2005.09814*, 2020.

[37] X.Y. Wang and Y.X. Yuan, Stochastic trust region methods with trust region radius depending on probabilistic models. *arXiv:1904.03342*, 2019.

[38] L. Yang, G. Zheng, H.T. Zhang, Y. Zhang, Q. Zheng, J. Wen and G. Pan, Policy optimization with stochastic mirror descent. *arXiv preprint arXiv:1906.10462*, 2019.

[39] K.Q Zhang, A. Koppel, H. Zhu and T. Basar, Global convergence of policy gradient methods to (almost) locally optimal policies. *SIAM Journal on Control and Optimization*, **58**:6 (2020), 3586–3612.