Riemannian Proximal Policy Optimization

Shijun Wang[1], Baocheng Zhu[2], Chen Li[2], Mingzhe Wu[2], James Zhang[3], Wei Chu[4] & Yuan Qi[4]

[1] Dept. of Artificial Intelligence, Ant Financial Services Group, Seattle, USA

[2] Dept. of Artificial Intelligence, Ant Financial Services Group, Shanghai, China

[3] Dept. of Artificial Intelligence, Ant Financial Services Group, New York, USA

[4] Dept. of Artificial Intelligence, Ant Financial Services Group, Hangzhou, China

Correspondence: Shijun Wang, Dept. of Artificial Intelligence, Ant Financial Services Group, Seattle, USA.

Abstract

In this paper, we propose a general Riemannian proximal optimization algorithm with guaranteed convergence to solve Markov decision process (MDP) problems. To model policy functions in MDP, we employ Gaussian mixture model (GMM) and formulate it as a non-convex optimization problem in the Riemannian space of positive semidefinite matrices. For two given policy functions, we also provide its lower bound on policy improvement by using bounds derived from the Wasserstein distance of GMMs. Preliminary experiments show the efficacy of our proposed Riemannian proximal policy optimization algorithm.

Keywords: Markov decision process, Riemannian proximal policy optimization, Gaussian mixture model, Wasserstein distance

1. Introduction

Reinforcement learning studies how agents explore/exploit environment, and take actions to maximize long-term reward. It has broad applications in robot control and game playing(Mnih et al., 2015; Silver et al., 2016; Argall et al., 2009; Silver et al., 2017). Value iteration and policy gradient methods are mainstream methods for reinforcement learning (Sutton & Barto, 2018; Li, 2017).

Policy gradient methods learn optimal policy directly from past experience or on the fly. It maximizes expected discounted reward through a parametrized policy whose parameters are updated using gradient ascent. Traditional policy gradient methods suffer from three well-known obstacles: high-variance, sample inefficiency and difficulty in tuning learning rate. To make the learning algorithm more robust and scalable to large datasets, Schulman et al. proposed trust region policy optimization algorithm (TRPO)(Schulman et al., 2015). TRPO searches for the optimal policy by maximizing a surrogate function with constraints placed upon the KL divergence between old and new policy distributions, which guarantees monotonically improvements. To further improve the data efficiency and reliable performance, proximal policy optimization algorithm (PPO) was proposed which utilizes first-order optimization and clipped probability ratio between the new and old policies (Schulman et al., 2017). TRPO was also extended to constrained reinforcement learning. Achiam et al. proposed constrained policy optimization (CPO) which guarantees near-constraint satisfaction at each iteration (Achiam et al., 2017).

Although TRPO, PPO and CPO have shown promising performance on complex decision-making problems, such as continuous-control tasks and playing Atari, as other neural network based models, they face two typical challenges: the lack of interpretability, and difficult to converge due to the nature of non-convex optimization in high dimensional parameter space. For many real applications, data lying in a high dimensional ambient space usually have a much lower intrinsic dimension. It may be easier to optimize the policy function in low dimensional manifolds.

In recent years, many optimization methods are generalized from Euclidean space to Riemannian space due to manifold structures existed in many machine learning problems(Absil et al., 2007, 2009; Vandereycken, 2013; Huang et al., 2015; Zhang et al., 2016). In this paper, we leverage merits of TRPO, PPO, and CPO and propose a new algorithm called Riemannian proximal policy optimization (RPPO) by taking manifold learning into account for policy optimization. In order to estimate the policy, we need a density-estimation function. Options we have include kernel density estimation, neural networks, Gaussian mixture model (GMM), etc. In this study we choose GMM due to its good analytical characteristics, universal representation power and low computational cost compared with neural networks. It is well-known that the covariance matrices of GMM lie in a Riemannian manifold of positive semidefinite matrices.

To be more specific, we model policy functions using GMM first. Secondly, to optimize GMM and learn the optimal policy functions efficiently, we formulate it as a non-convex optimization problem in the Riemannian space. By this way, our method gains advantages in improving both interpretability and speed of convergence. Please note that our RPPO algorithm can be easily extended to any other non-GMM density estimators, as long as their parameter space is Riemannian. In addition, GMM has been applied to reinforcement learning previously by embedding GMM in the Q-learning framework(Agostini & Celaya, 2010). But it also suffers from the headache of Q-learning that it can hardly handle problems with large continuous state-action space.

2. Preliminaries

2.1 Reinforcement Learning

In this study, we consider the following Markov decision process (MDP) which is defined as a tuple (S, A, P, r, γ), where S is the set of states, A is the set of actions, $P : S \times A \times S \rightarrow [0, 1]$ is the transition probability function, $r : S \times A \times S \rightarrow \mathcal{R}$ is the reward function, and γ is the discount factor which balances future rewards and immediate ones.

To make optimal decisions for MDP problems, reinforcement learning was proposed to learn optimal value function or policy. A value function is an expected, discounted accumulative reward function of a state or state-action pair by following a policy π. Here we define state value function as $v_\pi(s) = E_{\tau \sim \pi}[r(\tau) \mid s_0 = s]$ where $\tau = (s_0, a_0, s_1, ...)$ denotes a trajectory by playing policy π, $a_t \sim \pi(a_t \mid s_t)$, and $s_{t+1} \sim P(s_{t+1} \mid s_t, a_t)$. Similarly we define state-action value function as: $q_\pi(s, a) = E_{\tau \sim \pi}[r(\tau) \mid s_0 = s, a_0 = a]$. We also define advantage function as $A_\pi(s, a) = q_\pi(s, a) - v_\pi(s)$.

In reinforcement learning, we try to find or learn an optimal policy π which maximizes a given performance metric $J(\pi)$. Infinite horizon discounted accumulative return is widely used to evaluate the performance of a given policy which is defined as: $J(\pi) = \underset{\tau \sim \pi}{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t, s_{t+1})\right]$, where $r(s_t, a_t, s_{t+1})$ is the reward from s_t to s_{t+1} by taking action a_t. Please note that the expectation operation is performed over the distribution of trajectories.

In the work of (Kakade & Langford, 2002; Schulman et al., 2017), for two given policies π and π', their expected accumulative returns over infinite horizon can be linked by the advantage functions: $J(\pi') = J(\pi) + \underset{\tau \sim \pi'}{E}\left[\sum_{t=0}^{\infty} \gamma^t A_\pi(s_t, a_t)\right]$. By introducing discounted visit frequencies $\rho_\pi(s) = P(s_0 = s) + \gamma P(s_1 = s) + \gamma^2 P(s_2 = s) + ...$ (Schulman et al., 2015; Achiam et al., 2017), where $s_0 \sim \rho_0$, $\rho_0 : S \rightarrow \mathcal{R}$ is distribution of initial state s_0, we have $J(\pi') = J(\pi) + \sum_s \rho_{\pi'}(s) \sum_a \pi'(a \mid s) A_\pi(s_t, a_t)$. To reduce the complexity of searching for a new policy π' which increases $J(\pi')$, we replace discounted visit frequencies $\rho_{\pi'}(s)$ to be optimized with old discounted visit frequencies $\rho_\pi(s)$: $L(\pi') = L(\pi) + \sum_s \rho_\pi(s) \sum_a \pi'(a \mid s) A_\pi(s_t, a_t)$.

Assume that the policy functions $\pi(a \mid s)$ are parametrized by a vector θ, $\pi(a \mid s) = \pi_\theta(a \mid s)$. Searching for new policy π' is equivalent to searching new parameters θ' in the parameter space. So we have $L(\pi') = L(\theta') = L(\theta) + \sum_s \rho_{\pi_\theta}(s) \sum_a \pi_{\theta'}(a \mid s) A_{\pi_\theta}(s, a)$.

2.2 Riemannian Space

Here we give a brief introduction to Riemannian space, for more details see (Eisenhart, 2016). Let M be a connected and finite dimensional manifold with dimensionality of m. We denote by $T_p M$ the tangent space of M at p. Let M be endowed with a Riemannian metric $\langle ., . \rangle$, with corresponding norm denoted by $\| . \|$, so that M is now a Riemannian manifold (Eisenhart, 2016). We use $l(\gamma) = \int_a^b \| \gamma'(t) \| dt$ to denote length of a piecewise smooth curve $\gamma : [a, b] \longrightarrow M$ joining θ' to θ, i.e., such that $\gamma(a) = \theta'$ and $\gamma(b) = \theta$. Minimizing this length functional over the set of all piecewise smooth curves passing θ' and θ, we get a Riemannian distance $d(\theta', \theta)$ which induces original topology on M. Take $\theta \in M$, the exponential map $exp_\theta : T_\theta M \longrightarrow M$ is defined by $exp_\theta v = \gamma_v(1, \theta)$ which maps a tangent vector v at θ to M along the curve γ. For any $\theta' \in M$ we define the exponential inverse map $exp_{\theta'}^{-1} : M \longrightarrow T_{\theta'} M$ which is C^∞ and maps a point θ' on M to a tangent vector at θ with $d(\theta', \theta) = \| exp_{\theta'}^{-1}\theta \|$. We assume (M, d) is a complete metric space, bounded and all closed subsets of M are compact. For a given convex function $f : M \rightarrow R$ at $\theta' \in M$, a vector $s \in T_{\theta'} M$ is called subgradient of f at $\theta' \in M$ if $f(\theta) \geq f(\theta') + \langle s, exp_{\theta'}^{-1}\theta \rangle$, for all $\theta \in M$. The set of all subgradients of f at $\theta' \in M$ is called subdifferential of f at $\theta' \in M$ which is denoted by $\partial f(\theta')$. If M is a Hadamard manifold which is complete, simply connected and has everywhere non-positive sectional curvature, the subdifferential of f at any point on M is non-empty (Ferreira & Oliveira, 2002).

3. Modeling Policy Function Using Gaussian Mixture Model

To model policy functions, we employ the Gaussian mixture model which is a widely used and statistically mature method for clustering and density estimation. The policy function can be modeled as $\pi(a \mid s) = \Sigma_{i=1}^K \alpha(s) \mathcal{N}(a; \mu_i(s), S_i(s))$, where $\mathcal{N}$ is a (multivariate) Gaussian distribution with mean $\mu \in R^d$ and covariance matrix $S > 0$, K is number of components in the mixture model, $\alpha = (\alpha_1, \alpha_2, ..., \alpha_K)$ are mixture component weights which sum to 1. In the following, we drop s in

GMM to make it simple and parameters of GMM still depend on state variable s implicitly.

Let's define $\theta = ((\alpha_1, \mu_1, S_1), (\alpha_2, \mu_2, S_2), ..., (\alpha_K, \mu_K, S_K))$ which parametrizes the policy function. For a given policy function $\pi_\theta(a \mid s)$, we would like to find a new policy $\pi'_{\theta'}(a \mid s)$ which has higher performance evaluated by $L(\pi') = L(\pi) + \sum_s \rho_\pi(s) \sum_a \pi'(a \mid s) A_\pi(s, a)$ within close proximity of the old policy to avoid dramatic policy updates.

Define $g(\theta') = -L(\theta')$, and $\varphi(\theta') = \frac{\beta}{2} d^2(\theta', \theta)$ which searches new θ' near the proximity of the old parameter θ. $d(\theta', \theta)$ is a similarity metric which can be Euclidean distance, KL divergence, J-S divergence or the 2nd Wasserstein distance(Arjovsky et al., 2017).

We would like to optimize the following problem with corresponding constraints from GMMs:

$$\min_{\theta' = ((\alpha'_1, \mu'_1, S'_1), (\alpha'_2, \mu'_2, S'_2), ..., (\alpha'_K, \mu'_K, S'_K))} f(\theta') = g(\theta') + \varphi(\theta') \tag{1}$$

$$= -L(\theta) - \sum_s \rho_{\pi_\theta}(s) \sum_a \left(\Sigma_{i=1}^K \alpha'_i \mathcal{N}(a; \mu'_i, S'_i) \right) A_{\pi_\theta}(s, a) + \frac{\beta}{2} d^2(\theta', \theta),$$

s.t. $\Sigma_{i=1}^K \alpha'_i = 1, S'_i > 0, i = 1, 2, ..., K$.

We employ a reparametrization method to make the Gaussian distributions zero-centered. We augment action variables by 1 and define a new variable vector as $a = [a, 1]^\top$ with new covariance matrix $S = \begin{bmatrix} S + \mu\mu^\top & \mu \\ \mu^\top & 1 \end{bmatrix}$(Hosseini & Sra, 2015).

In the Optimization Problem (1), there is a simplex constraint $\alpha \in \Delta_K$. To convert it to a unconstrained problem, we first define $\eta_k = log(\frac{\alpha_k}{\alpha_K})$, for $k = 1, 2, ..., K - 1$, and let $\eta_K = 0$ be a constant (Hosseini & Sra, 2015). Then we have the following unconstrained optimization problem:

$$\min_{\theta' = \{\eta' = \{\eta'_i\}_{i=1}^{K-1}, S' = \{S'_i > 0\}_{i=1}^K\}} f(\theta') = g(\theta') + \varphi(\theta') \tag{2}$$

$$= -L(\theta) - \sum_s \rho_{\pi_\theta}(s) \sum_a \left(\Sigma_{i=1}^K \frac{exp(\eta'_k)}{\Sigma_{j=1}^K exp(\eta'_j)} \mathcal{N}(a; S'_i) \right) A_{\pi_\theta}(s, a) + \frac{\beta}{2} d^2(\theta', \theta).$$

4. Riemannian Proximal Method for Policy Optimization

4.1 Riemannian Proximal Method for a Class of Non-convex Problems

In this section, following the work of (Khamaru & Wainwright, 2018), we tackle a more general class of functions of the form: $f(\theta) = g(\theta) - h(\theta) + \varphi(\theta)$. We assume the following assumptions hold:

Assumption 1:

  a) The function g is continuously differentiable and its gradient vector field is Lipschitz continuous with constant $L \geq 0$.

  b) The function h is continuous and convex.

  c) The function $\varphi(\theta)$ is proper, convex and lower semi-continuous.

  d) The function f is bounded below over a complete Riemannian manifold M of dimension m.

  e) Solution set of min $f(\theta)$ is non-empty and its optimum value is denoted as f^*.

Lemma 1. Under Assumption 1, assume u_k and v_k are subgradients of the convex functions h and φ, respectively. We have $\theta_{k+1} = exp_{\theta_k}(-\alpha_k(\nabla g(\theta_k) - u_k + v_{k+1}))$, and $f(\theta_k) - f(\theta_{k+1}) \geq \frac{1}{2\alpha_k} d^2(\theta_k, \theta_{k+1})$, where α_k is the step size.

Proof of Lemma 1 can be found in the Appendix.

Theorem 1. Under Assumption 1, the following statements hold for any sequence $\{\theta_k\}_{k \geq 0}$ generated by Algorithm 1:

  a) Any limit point of the sequence $\{\theta_k\}_{k \geq 0}$ is a critical point, and the sequence of function values $\{f(\theta_k)\}_{k \geq 0}$ is strictly decreasing and convergent.

b) For $k = 0, 1, 2, ..., N$, we have $\sum_{k=0}^{N} d^2(\theta_k, \theta_{k+1}) \leq \frac{2(f(\theta_0) - f^*)}{L}$. In addition, if the function h is M_h-smooth: $\|u_{k+1} - u_k\| \leq M_h \times d(\theta_k, \theta_{k+1})$ where M_h is a constant, then

$$\sum_{k=0}^{N} \frac{1}{(L + M_h + \frac{1}{\alpha_k})^2} \|\nabla f(\theta_{k+1})\|_2^2 \leq \frac{2(f(\theta_0) - f^*)}{L} \tag{3}$$

Proof of Theorem 1 can be found in the Appendix.

Algorithm 1: Riemannian proximal optimization algorithm

1. Given an initial point $\theta_0 \in M$ and choose step size $\alpha_k \in (0, \frac{1}{L}]$, $k \in \{0, 1, 2, ...\}$.

2. For $k = 0, 1, 2, ..., $ do

   Choose subgradient $u_k \in \partial h(\theta_k)$, where $\partial h(\theta)$ denotes subdifferential (or subderivatives) of the convex function h at the point θ. Update

   $$\theta_{k+1} = \min_{\theta \in M} \left\{ \varphi(\theta) + \frac{1}{2\alpha_k} d^2(\theta, \theta_k - \alpha_k (\nabla g(\theta_k) - u_k)) \right\}. \tag{4}$$

4.2 Lower Bound of Policy Improvement

Assume we have two policy functions $\pi'(a \mid s) = \Sigma_i \alpha_i' \mathcal{N}(a; S_i')$ and $\pi(a \mid s) = \Sigma_i \alpha_i \mathcal{N}(a; S_i)$ parameterized by GMMs with parameters $\theta' = \{\alpha' = \{\alpha_i'\}_{i=1}^K, S' = \{S_i' > 0\}_{i=1}^K\}$ and $\theta = \{\alpha = \{\alpha_i\}_{i=1}^K, S = \{S_i > 0\}_{i=1}^K\}$, we would like to bound the performance improvement of $\pi'(a \mid s)$ over $\pi(a \mid s)$ under limitation of the proximal operator.

In this study, we choose the Wasserstein distance to measure discrepancy between policy functions $\pi'(a \mid s)$ and $\pi(a \mid s)$ due to its robustness. For two distributions μ_0 and μ_1 of dimension n, the Wasserstein distance is defined as $W_2(\mu_0, \mu_1) = \inf_{p \in \Pi(\mu_0, \mu_1)} \int_{R^n \times R^n} \| x - y \|^2 p(x, y) dx dy$ which seeks a joint probability distribution Π in R^{2n} whose marginals along coordinates x and y coincide with μ_0 and μ_1, respectively. For two Gaussian distributions $N_0(\mu_0, S_0)$ and $N_1(\mu_1, S_1)$, its Wasserstein distance is $W_2(N_0, N_1)^2 = \| \mu_0 - \mu_1 \|^2 + trace(S_0 + S_1 - 2(S_0^{1/2} S_1 S_0^{1/2})^{1/2})$.

First we have the following lemma:

Lemma 2. Given two policies parametrized by GMMs $\pi'(a \mid s) = \Sigma_i \alpha_i' \mathcal{N}(a; S_i')$ and $\pi(a \mid s) = \Sigma_i \alpha_i \mathcal{N}(a; S_i)$, let $f(\pi') = \beta \Sigma_s \rho_\pi(s) \Sigma_a \left(\Sigma_{i=1}^K \frac{exp(\eta_i')}{\Sigma_{j=1}^K exp(\eta_j')} \mathcal{N}(a; S_i') \right) A_\pi(s, a) - D_{W_2}^\pi(\pi', \pi)$, where $D_{W_2}^\pi(\pi', \pi) = \Sigma_s \rho^\pi(s) W_2(\pi', \pi)$, W_2 defines Wasserstein distance between two GMMs. Then exist $\tilde{\pi} = \arg\max_{\pi'} f(\pi')$, and $f(\tilde{\pi}) \geq f(\pi) = 0$.

Lemma 2 can be simply proved by applying Theorem 1.

To reduce computational complexity, we employ discrete Wasserstein distance by embedding GMMs to a manifold of probability densities with Gaussian mixture structure as proposed by (Chen et al., 2019). To be more specific, the discrete Wasserstein distance between two GMMs $\pi'(a \mid s)$ and $\pi(a \mid s)$ is:

$$W_2(\pi', \pi)^2 = \sum_{i,j} c^*(i, j) W_2(\mathcal{N}(a; S_i'), \mathcal{N}(a; S_j))^2, \tag{5}$$

where $c^*(i, j) = \arg\min_{c \in \prod(\alpha', \alpha)} \Sigma_{i,j} c(i, j) W_2(\mathcal{N}(a; S_i'), \mathcal{N}(a; S_j))^2$.

Lemma 3. Given two policies parametrized by GMMs $\pi'(a \mid s) = \Sigma_i \alpha_i' \mathcal{N}(a; S_i')$ and $\pi(a \mid s) = \Sigma_i \alpha_i \mathcal{N}(a; S_i)$, their total variation distance can be bounded as follows:

$$D_{TV}(\pi'(a \mid s), \pi(a \mid s)) \leq B_{TV}(\pi', \pi) = \sum_{i,j} d^*(i, j) B_{TV}(\mathcal{N}(a; S_i^{\pi'}), \mathcal{N}(a; S_j^\pi)), \tag{6}$$

where $B_{TV}(N_0(\mu, S_0), N_1(\mu, S_1)) = \frac{3}{2} \min\{1, \| S_0^{-1} S_1 - I \|_F\}$ for Gaussian distributions $N_0(\mu, S_0)$ and $\mathcal{N}(\mu, S_1)$(Devroye et al., 2018), and $d^*(i, j) = \arg\min_{d \in \prod(\alpha', \alpha)} \Sigma_{i,j} d(i, j) B_{TV}(\mathcal{N}(a; S_i'), \mathcal{N}(a; S_j))$. Please note the bound $B_{TV}(\pi', \pi)$ actually is

the Wasserstein distance between the discrete distributions $\alpha' = \{\alpha'_1, \alpha'_2, ..., \alpha'_K\}$ and $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_K\}$ with pairwise cost defined by the bound of total variation distance between two Gaussian distributions $B_{TV}(N_i(\mu, S_i), N_j(\mu, S_j))$, $i, j = 1, 2, ..., K$.

With Lemma 2 and Lemma 3, we have the following theorem:

Theorem 2. Given two policy functions π' and π parametrized by GMMs $\pi'(a \mid s) = \Sigma_i \alpha'_i \mathcal{N}(a; S'_i)$ and $\pi(a \mid s) = \Sigma_i \alpha_i \mathcal{N}(a; S_i)$, assume policy $\tilde{\pi}$ is parametrized by $\tilde{\theta} = \arg\max_{\pi'} f(\pi')$ as shown in Lemma 2, then we have the following bound for any π' within proximity of $\tilde{\pi}$:

$$J(\pi') - J(\pi) \geq -2B_{TV}^{\pi'}(\pi', \tilde{\pi})M^{\tilde{\pi}} + \frac{1}{\beta}D_{W_2}^{\pi}(\tilde{\pi}, \pi) - \frac{2\gamma\epsilon_{\pi}^{\tilde{\pi}}}{(1 - \gamma)}B_{TV}^{\pi}(\tilde{\pi}, \pi), \tag{7}$$

where $\epsilon_{\pi}^{\tilde{\pi}} = \max_s \mid E_{a \sim \tilde{\pi}} A^{\pi}(s, a) \mid$, $M^{\tilde{\pi}} = \max_{s,a} |A^{\tilde{\pi}}(s, a)|$, $B_{TV}^{\pi'}(\pi', \tilde{\pi}) = \sum_s \rho^{\pi'}(s)B_{TV}(\pi', \tilde{\pi})$ and $B_{TV}^{\pi}(\tilde{\pi}, \pi) = \sum_s \rho^{\pi}(s)B_{TV}(\tilde{\pi}, \pi)$ ($B_{TV}(\pi', \tilde{\pi})$ and $B_{TV}(\tilde{\pi}, \pi)$ follow the bound definition $B_{TV}(\pi', \pi)$ in Lemma 3), $B_{W_2}^{\pi}(\tilde{\pi}, \pi) = \sum_s \rho^{\pi}(s)W_{d2}(\tilde{\pi}, \pi)$.

Proofs of Lemma 3 and Theorem 2 are shown in the appendix.

4.3 Implementation of the Riemannian proximal policy optimization method

Recall that in the optimization problem (2), we are trying to optimize the following objective function:

$$\min_{\theta' = \{\eta' = \{\eta'_i\}_{i=1}^{K-1}, S' = \{S'_i > 0\}_{i=1}^K\}} f(\theta') = g(\theta') + \varphi(\theta').$$

1) *Riemannian Gradient*

$$grad_{S'_i}g(\theta') = \frac{\partial g(\theta')}{\partial S'_i} = -\Sigma_{i=1}^K (\Sigma_s \rho_{\pi_\theta}(s)\Sigma_a A_{\pi_\theta}(s, a))\frac{exp(\eta'_i)}{\Sigma_{j=1}^K exp(\eta'_j)} \times \frac{\partial \mathcal{N}(a; S'_i)}{\partial S'_i}, \tag{8}$$

$$\frac{\partial \mathcal{N}(a; S'_i)}{\partial S'_i} = \mathcal{N}(a; S'_i) \times \frac{1}{2}\left[-S'^{-1}_i + S'^{-1}_i aa^\top S'^{-1}_i\right], i = 1, 2, ..., K. \tag{9}$$

Let $a_i = -(\Sigma_s \rho_{\pi_\theta}(s)\Sigma_a A_{\pi_\theta}(s, a))\mathcal{N}(a; S'_i)$, $m = 1, 2, ..., K$,

$$grad_{\eta'_m}g(\theta') = \frac{\partial g(\theta')}{\partial \eta'_m} = \frac{a_m exp(\eta'_m)}{(\Sigma_j exp(\eta'_j))} - \Sigma_i \frac{1}{(\Sigma_j exp(\eta'_j))^2}\{a_i exp(\eta'_i) \times exp(\eta'_m)\}. \tag{10}$$

For Euclidean distance $d(\theta', \theta) = \frac{\beta}{2} \parallel \theta' - \theta \parallel_2^2$, we have

$$\partial_{S'_i}\varphi(\theta') = \frac{\partial}{\partial S'_i}(\frac{\beta}{2}\sum_{i=1}^K d^2(S'_i, S_i)) = \beta(S'_i - S_i), i = 1, 2, ..., K. \tag{11}$$

For discrete Wasserstein distance $d = W_{d2}^2(\theta', \theta)$, we have

$$\partial_{S'_i}\varphi(\theta') = \frac{\beta}{2}\frac{\partial}{\partial S'_i}(\sum_{i,j} c^*(i, j)trace(S'_i + S_j - 2(S'^{1/2}_i S_j S'^{1/2}_i)^{1/2})), \tag{12}$$

where $c^*(i, j) = \arg\min_{c \in \prod(\alpha', \alpha)} \sum_{i,j} c(i, j)W_2(\mathcal{N}(a; S'_i), \mathcal{N}(a; S_j))^2$, $i = 1, 2, ..., K$.

2) *Retraction*

With $S'_{i,t}$ and $grad_{S'_{i,t}}g(\theta')$ shown above at iteration t, we would like to calculate $S'_{i,t+1}$ using retraction. From (Cheng, 2013), for any tangent vector $\eta \in T_W M$, where W is a point in Riemannian space M, its retraction $R_W(\eta) := \arg\min_{X \in M} \parallel W + \eta - X \parallel_F$. For our case $R_{S'_{i,t}}\left(-\alpha_t(grad_{S'_{i,t}}g(\theta') + \partial_{S'_{i,t}}\varphi(\theta'))\right) = \Sigma_{i=1}^n \sigma_i q_i q_i^\top$, where σ_i and q_i are the i-th eigenvalues and eigenvector of matrix $S'_{i,t} - \alpha_t(grad_{S'_{i,t}}g(\theta') + \partial_{S'_{i,t}}\varphi(\theta'))$.

η_i, $i = 1, 2, ..., K - 1$ are updated using standard gradient decent method in the Euclidean space. The calculation and retraction shown above are repeated until $f(\theta')$ converges.

5. Experimental Results

5.1 Simulation Environments and Baseline Methods

We choose TRPO and PPO, which are well-known excelling at continuous-control tasks, as baseline algorithms. Each algorithm runs on the following 3 environments in OpenAI Gym MuJoCo simulator (Todorov et al., 2012): InvertedPendulum-v2, Hopper-v2, and Walker2d-v2 with increasing task complexity regarding size of state and action spaces. For each run, we compute the average reward for every 50 episodes, and report the mean reward curve and parameters statistics for comparison.

5.2 Preliminary Results

Table 1. Number of parameters of each algorithm on each environment with dimensions listed.

Environments	Number of parameters			Dim. of states	Dim. of actions
	RPPO	TRPO	PPO		
InvertedPendulum-v2	104	124,026	124,026	4	1
Hopper-v2	599	5,281,434	5,281,434	11	3
Walker2d-v2	599	40,404,522	40,404,522	17	6

In Fig. 1 we show mean reward (column1) for PPO, RPPO and TRPO algorithms on three MuJoCo environments, screenshots (column2) and probability density of GMM (column3) for RPPO on each environment. From the learning curves, we can see that as the state-action dimension of environment increases (shown in Table 1), both the convergence speed and the reward improvement slow down. This is because the higher dimension the environment sits, the more difficult the optimization task is for the algorithm. Correspondingly, in the GMM plot, S, A represent the state and the action dimensions respectively, and the probability density is shown in z axis. In the density plot, we can see that as the environment complexity increases, the density pattern becomes more diverse, and non-diagonal matrix terms also show its importance. The probability density of GMM shows that RPPO learns meaningful structure of policy functions.

TRPO and PPO are pure neural-network-based models with numerous parameters. This makes the model highly vulnerable to overfitting, poor network architecture design and the hyper-parameters tuning. RPPO achieves better robustness by having much fewer parameters. In Table 1 we compare the number of parameters of each algorithm on each environment. It can be seen that GMM has $10^3 \sim 10^5$ order fewer parameters as compared with TRPO and PPO.
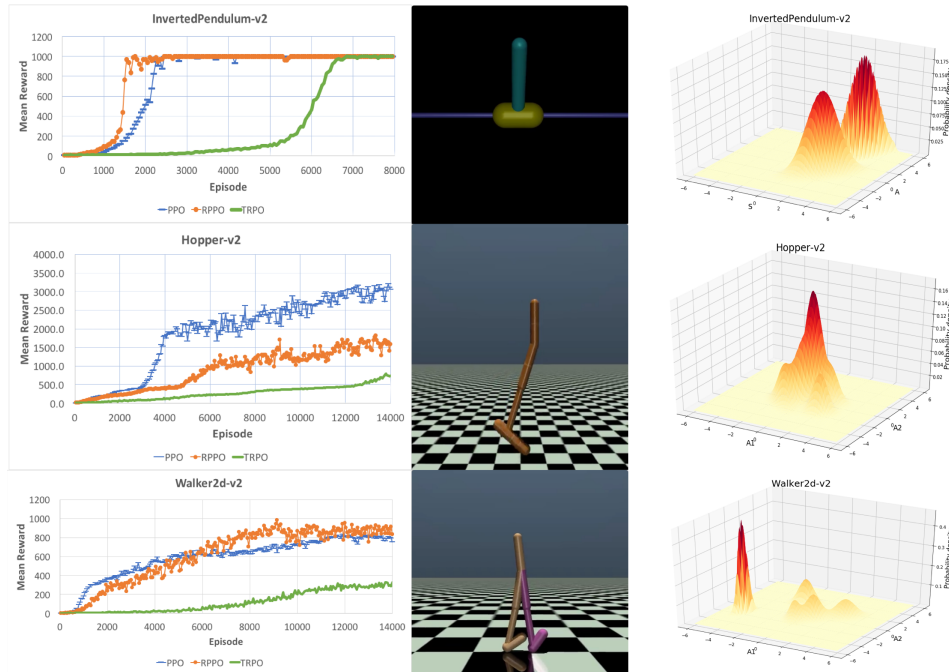

Figure 1. Comparison of PPO, RPPO and TRPO on three MuJoCo environments.

6. Conclusion

We proposed a general Riemannian proximal optimization algorithm with guaranteed convergence to solve Markov decision process (MDP) problems. To model policy functions in MDP, we employed the Gaussian mixture model (GMM) and formulated it as a non-convex optimization problem in the Riemannian space of positive semidefinite matrices. Preliminary experiments on benchmark tasks in OpenAI Gym MuJoCo (Todorov et al., 2012) show the efficacy of the proposed RPPO algorithm.

In Sec. 4.1, the algorithm 1 we proposed is capable of optimizing a general class of non-convex functions of the form $f(\theta) = g(\theta) - h(\theta) + \varphi(\theta)$. Due to page limit, in this study we focus on $f(\theta) = g(\theta) + \varphi(\theta)$ as shown in the Optimization problem (2). In the future, it would be interesting to incorporate constraints in MDP problems like constrained policy optimization (Achiam et al., 2017) and encode them as a concave function $-h(\theta)$ in our RPPO algorithm.

References

Absil, P. A., Baker, C. G., & Gallivan, K. A. (2007). Trust-region methods on riemannian manifolds. *Foundations of Computational Mathematics*, *7*(3), 303-330. https://doi.org/10.1007/s10208-005-0179-9

Absil, P. A., Mahony, R., & Sepulchre, R. (2009). *Optimization algorithms on matrix manifolds*. Princeton University Press. https://doi.org/10.1515/9781400830244

Achiam, J., Held, D., Tamar, A., & Abbeel, P. (2017). Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning, 70*, 22-31. JMLR. org.

Agostini, A., & Celaya, E. (2010). Reinforcement learning with a gaussian mixture model. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, (pp. 1-8). IEEE. https://doi.org/10.1109/IJCNN.2010.5596306

Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and autonomous systems*, *57*(5), 469-483. https://doi.org/10.1016/j.robot.2008.10.024

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

Chen, Y., Georgiou, T. T., & Tannenbaum, A. (2019). Optimal transport for gaussian mixture models. *IEEE Access*, *7*, 6269-6278. https://doi.org/10.1109/ACCESS.2018.2889838

Cheng, L. (2013). Riemannian similarity learning. In *International Conference on Machine Learning* (pp. 540-548).

Devroye, L., Mehrabian, A., & Reddad, T. (2018). The total variation distance between high-dimensional gaussians. *arXiv preprint arXiv:1810.08693*.

Eisenhart, L. P. (2016). *Riemannian geometry*. Princeton university press.

Ferreira, O., & Oliveira, P. (2002). Proximal point algorithm on riemannian manifolds. *Optimization*, *51*(2), 257-270. https://doi.org/10.1080/02331930290019413

Hosseini, R., & Sra, S. (2015). Matrix manifold optimization for gaussian mixtures. In *Advances in Neural Information Processing Systems* (pp. 910-918).

Huang, W., Gallivan, K. A., & Absil, P. A. (2015). A broyden class of quasi-newton methods for riemannian optimization. *SIAM Journal on Optimization*, *25*(3), 1660-1685. https://doi.org/10.1137/140955483

Kakade, S., & Langford, J. (2002). Approximately optimal approximate reinforcement learning. In *Proceedings of the 19th International Conference on Machine Learning*, (pp. 267-274).

Khamaru, K., & Wainwright, M. J. (2018). Convergence guarantees for a class of non-convex and non-smooth optimization problems. *arXiv preprint arXiv:1804.09629*.

Li, Y. (2017). Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529. https://doi.org/10.1038/nature14236

Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P. (2015). Trust region policy optimization. In *International Conference on Machine Learning* (pp. 1889-1897).

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *CoRR*, *abs/1707.06347*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... Lanctot, M. et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, *529*(7587), 484.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., ... Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *Nature*, *550*(7676), 354. https://doi.org/10.1038/nature24270

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Todorov, E., Erez, T., & Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (pp. 5026-5033). IEEE. https://doi.org/10.1109/IROS.2012.6386109

Vandereycken, B. (2013). Low-rank matrix completion by Riemannian optimization. *SIAM Journal on Optimization*, *23*(2), 1214-1236. https://doi.org/10.1137/110845768

Wang, Q., Xiong, J., Han, L., Liu, H., & Zhang, T., et al. (2018). Exponentially weighted imitation learning for batched historical data. In *Advances in Neural Information Processing Systems* (pp. 6288-6297).

Zhang, H., Reddi, S. J., & Sra, S. (2016). Riemannian svrg: Fast stochastic optimization on riemannian manifolds. In *Advances in Neural Information Processing Systems* (pp. 4592-4600).

Appendix

1. Proof of Lemma 1:

First let's define a convex majorant $q(w, \theta_k)$ of the function f as follows:

$$q(w, \theta_k) = g(\theta_k) - h(\theta_k) + \langle \nabla g(\theta_k) - u_k, w \rangle + \frac{1}{2\alpha_k} \|w\|_2^2 + \varphi(exp_{\theta_k}(w)), \tag{13}$$

where $w \in T_{\theta_k} M$. Note that minimizer of $q(w, \theta_k)$ is the same as $\theta_{k+1} = \min_{\theta \in M} \left\{ \varphi(\theta) + \frac{1}{2\alpha_k} d^2(\theta, \theta_k - \alpha_k (\nabla g (\theta_k) - u_k)) \right\}$.

The optimality condition of θ_{k+1} guarantees that there exists subgradient $v_{k+1} \in \partial \varphi(\theta_{k+1})$ satisfying the following equation:

$$\nabla g(\theta_k) - u_k + v_{k+1} + \frac{1}{\alpha_k} w = 0. \tag{14}$$

Let $w = exp_{\theta_k}^{-1} \theta_{k+1}$, we have

$$\theta_{k+1} = exp_{\theta_k}(-\alpha_k(\nabla g(\theta_k) - u_k + v_{k+1})). \tag{15}$$

From convexity of the function φ, for any $\theta \in M$ and $v_{k+1} \in \partial \varphi(\theta_{k+1})$ we have $\varphi(\theta_k) \geq \varphi(\theta_{k+1}) + \langle v_{k+1}, w \rangle, w \in T_{\theta_{k+1}} M$.

To prove the second inequality in Lemma 1, we have

$$\begin{aligned} f(\theta_k) - q(w_k, \theta_k) =& g(\theta_k) - h(\theta_k) + \varphi(\theta_k) - q(w_k, \theta_k) \\ \geq & g(\theta_k) - h(\theta_k) + \varphi(\theta_{k+1}) + \langle v_{k+1}, w \rangle - q(w_k, \theta_k) \\ \geq & g(\theta_k) - h(\theta_k) + \varphi(\theta_{k+1}) + \langle v_{k+1}, w \rangle - (g(\theta_k) - h(\theta_k) + \langle \nabla g(\theta_k) - u_k, w_k \rangle + \frac{1}{2\alpha_k} \|w_k\|_2^2 + \varphi(exp_{\theta_k}(w_k))) \\ \geq & \varphi(\theta_{k+1}) + \langle v_{k+1}, w \rangle - (\langle \nabla g(\theta_k) - u_k, w_k \rangle + \frac{1}{2\alpha_k} \|w_k\|_2^2 + \varphi(exp_{\theta_k}(w_k))). \end{aligned} \tag{16}$$

Since $w = -w_k = \alpha_k(\nabla g(\theta_k) - u_k + v_{k+1})$, we have

$$f(\theta_k) - q(w_k, \theta_k) \geq \langle -\nabla g(\theta_k) + u_k - v_{k+1}, w_k \rangle - \frac{1}{2\alpha_k} \|w_k\|_2^2 \geq \frac{1}{\alpha_k} \langle w_k, w_k \rangle - \frac{1}{2\alpha_k} \|w_k\|_2^2 = \frac{1}{2\alpha_k} \|w_k\|_2^2. \tag{17}$$

Recall that $q(w_k, \theta_k)$ is a majorant for the function f, so

$$f(\theta_k) - f(\theta_{k+1}) \geq f(\theta_k) - q(w_k, \theta_k) \geq \frac{1}{2\alpha_k} \|w_k\|_2^2 = \frac{1}{2\alpha_k} d^2(\theta_k, \theta_{k+1}). \tag{18}$$

2. Proof of Theorem 1:

First we would like to prove the convergence of function value. Since the sequence $\{f(\theta_k)\}_{k \geq 0}$ is bounded below, if $\theta_k = \theta_{k+1}$ for some k, the convergence of the sequence $\{f(\theta_k)\}_{k \geq 0}$ is trivial. Let's assume that $\theta_k \neq \theta_{k+1}$ for all $k = 0, 1, 2, ...$ Under the above assumption, Lemma 1 ensures that $f(\theta_k) > f(\theta_{k+1})$. Consequently, there must exist some scalar $\bar{f}$ which is the limit of $f(\theta_k)$, $\lim_{k \to \infty} f(\theta_k) = \bar{f}$.

Due to page limit, the proof of stationarity of limit points is omitted.

Now let's establish the bound. Since $f^* = \min f(\theta)$ is finite, by utilizing Lemma 1, we have

$$f(\theta_0) - f^* \geq f(\theta_0) - f(\theta_{N+1}) = \sum_{k=0}^{N} (f(\theta_k) - f(\theta_{k+1})) \geq \sum_{k=0}^{N} \frac{1}{2\alpha_k} d^2(\theta_k, \theta_{k+1}). \tag{19}$$

Note that $\alpha_k \in (0, \frac{1}{L}]$, $k \in \{0, 1, 2, ...\}$, so $\sum_{k=0}^{N} d^2(\theta_k, \theta_{k+1}) \leq \frac{2(f(\theta_0) - f^*)}{L}$.

Recall that the function h is M_h-smooth,

$$\|\nabla f(\theta_{k+1})\|_2 = \|\nabla g(\theta_{k+1}) - u_{k+1} + v_{k+1}\|_2 = \|\nabla g(\theta_{k+1}) - u_{k+1} - (\nabla g(\theta_k) - u_k + \frac{1}{\alpha_k} d(\theta_k, \theta_{k+1}))\|_2 \tag{20}$$

$$\leq \|\nabla g(\theta_{k+1}) - \nabla g(\theta_k)\| + \|u_{k+1} - u_k\| + \frac{1}{\alpha_k} d(\theta_k, \theta_{k+1}) \leq (L + M_h + \frac{1}{\alpha_k}) d(\theta_k, \theta_{k+1})$$

$$\tag{21}$$

So

$$\frac{2(f(\theta_0) - f^*)}{L} \geq \sum_{k=0}^{N} d^2(\theta_k, \theta_{k+1}) \geq \sum_{k=0}^{N} \frac{1}{(L + M_h + \frac{1}{\alpha_k})^2} \|\nabla f(\theta_{k+1})\|_2^2 \tag{22}$$

3. Proof of Lemma 3:

For two Gaussian distributions $N_0(\mu, S_0)$ and $N_1(\mu, S_1)$ with the same mean, their total variation distance is bounded by (Devroye et al., 2018):

$$D_{TV}(N_0(\mu, S_0), N_1(\mu, S_1)) \leq B_{TV}(N_0(\mu, S_0), N_1(\mu, S_1)) = \frac{3}{2} \min\{1, \| S_0^{-1} S_1 - I \|_F\}. \tag{23}$$

By using Wasserstein metric, we have

$$D_{TV}(\pi'(a \mid s), \pi(a \mid s)) = \frac{1}{2} \int | \Sigma_i \alpha_i' \mathcal{N}(a; S_i') - \Sigma_i \alpha_i \mathcal{N}(a; S_i) | \, da \leq \sum_{i,j} d^*(i, j) B_{TV}(\mathcal{N}(a; S_i'), \mathcal{N}(a; S_j)), \tag{24}$$

where $d^*(i, j) = \underset{d \in \prod(\alpha', \alpha)}{\arg \min} \sum_{i,j} d(i, j) B_{TV}(\mathcal{N}(a; S_i'), \mathcal{N}(a; S_j))$.

4. Proof of Theorem 2:

Our proof follows the idea proposed by Wang et al. (Wang et al., 2018). From Corollary 1 in (Achiam et al., 2017), we have

$$J(\pi') - J(\pi) \geq L_\pi(\pi') - \frac{2\gamma \epsilon_\pi^{\pi'}}{(1 - \gamma)} D_{TV}^\pi(\pi', \pi), \tag{25}$$

where $\epsilon_\pi^{\pi'} = \max_s | E_{a \sim \pi'} A^\pi(s, a) |$, $D_{TV}^\pi(\pi', \pi) = \sum_s \rho_\pi(s) D_{TV}(\pi'(a \mid s), \pi(a \mid s))$ and $L_\pi(\pi') = \sum_s \rho_\pi(s) \sum_a \pi'(a|s) A_\pi(s, a)$.

Similarly, we also have

$$|J(\pi') - J(\pi)| = \sum_s \rho_\pi(s) |\sum_a (\pi'(a|s) - \pi(a|s)) A_\pi(s, a)| \leq \sum_s \rho_\pi(s) \sum_a |\pi'(a|s) - \pi(a|s)| |A_\pi(s, a)| \leq 2 D_{TV}^{\pi'}(\pi', \pi) M^\pi, \tag{26}$$

where $M^\pi = \max_{s,a} |A^\pi(s, a)|$. Then

$$J(\pi') - J(\pi) = (J(\pi') - J(\tilde{\pi})) + (J(\tilde{\pi}) - J(\pi)) \geq -2 D_{TV}^{\pi'}(\pi', \tilde{\pi}) M^{\tilde{\pi}} + L_\pi(\tilde{\pi}) - \frac{2\gamma \epsilon_\pi^{\tilde{\pi}}}{(1 - \gamma)} D_{TV}^\pi(\tilde{\pi}, \pi) \tag{27}$$

$$\geq -2 D_{TV}^{\pi'}(\pi', \tilde{\pi}) M^{\tilde{\pi}} + \frac{1}{\beta} D_{W_2}^\pi(\tilde{\pi}, \pi) - \frac{2\gamma \epsilon_\pi^{\tilde{\pi}}}{(1 - \gamma)} D_{TV}^\pi(\tilde{\pi}, \pi).$$