Wasserstein Policy Optimization

MATH 277A Project

Hu Hanyang

2026-06-06

1 Introduction

Reinforcement learning studies sequential decision-making problems in which an agent interacts with an environment and seeks to maximize long-term reward. We model the environment as a discounted Markov decision process with state space $\mathcal{S}$, action space $\mathcal{A}$, transition probability $\mathbb{P}(s'|s,a)$, reward function $r(s,a)$, initial-state distribution p_0, and discount factor $\gamma \in (0,1)$. Given a stochastic policy $\pi(a|s)$, the objective is to maximize the expected discounted return

$$\mathcal{J}[\pi] = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\right], \tag{1.1}$$

where $\tau = (s_0, a_0, s_1, a_1, \ldots)$ denotes a trajectory s.t. $s_0 \sim p_0$, $a_t \sim \pi(\cdot|s_t)$, and $s_{t+1} \sim \mathbb{P}(\cdot|s_t,a_t)$. The goal of policy optimization is to improve π so as to increase $\mathcal{J}[\pi]$.

Definition 1.1 (Value functions and discounted occupancy)**.** *For a policy π, the value function and action-value function are defined by*

$$V^\pi(s) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\,\middle|\, s_0 = s\right], \tag{1.2}$$

$$Q^\pi(s,a) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\,\middle|\, s_0 = s,\ a_0 = a\right]. \tag{1.3}$$

The discounted state-occupancy distribution is

$$d^\pi(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \mathbb{P}^\pi(s_t = s), \tag{1.4}$$

where $\mathbb{P}^\pi(s_t = s)$ denotes the probability of visiting state s at time t under policy π.

A standard approach is to represent the policy by a finite-dimensional parameter vector θ and optimize $\mathcal{J}[\pi_\theta]$ by policy gradient [14]. The key result is the policy gradient theorem, which expresses the gradient of the expected return in terms of the score function weighted by the action-value function.

Theorem 1.2 (Policy gradient)**.** *Let π_θ be a differentiable stochastic policy. Under standard regularity assumptions allowing differentiation under the expectation, the policy gradient can be written as*

$$\nabla_\theta \mathcal{J}[\pi_\theta] = \int \frac{1}{1-\gamma} Q^{\pi_\theta}(s,a) d^{\pi_\theta}(s) \nabla_\theta \pi_\theta(a|s)\, ds\, da. \tag{1.5}$$

Equivalently, using $\nabla_\theta \pi_\theta(a|s) = \pi_\theta(a|s)\nabla_\theta \log \pi_\theta(a|s)$, it can be written in the score function form

$$\nabla_\theta \mathcal{J}[\pi_\theta] = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta},\, a\sim\pi_\theta(\cdot|s)}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \log \pi_\theta(a|s)\right]. \tag{1.6}$$

To see the relation between the two forms, we start from the direct policy derivative expression:

$$\begin{aligned}
\nabla_\theta \mathcal{J}[\pi_\theta] &= \frac{1}{1-\gamma}\int_{\mathcal{S}} d^{\pi_\theta}(s)\int_{\mathcal{A}} Q^{\pi_\theta}(s,a)\nabla_\theta \pi_\theta(a|s)\, da\, ds \\
&= \frac{1}{1-\gamma}\int_{\mathcal{S}} d^{\pi_\theta}(s)\int_{\mathcal{A}} Q^{\pi_\theta}(s,a)\pi_\theta(a|s)\nabla_\theta \log \pi_\theta(a|s)\, da\, ds \\
&= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta},\, a\sim\pi_\theta(\cdot|s)}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \log \pi_\theta(a|s)\right].
\end{aligned} \tag{1.7}$$

This direct policy derivative form is the finite-dimensional counterpart of the functional policy gradient viewpoint mentioned later in this report.

The score function form (1.6) is the one most often used for Monte Carlo estimation, while the direct policy derivative form (1.5) is closer to the functional perspective. Natural policy gradient rescales the vanilla gradient using the Fisher information matrix, thereby taking into account the information

geometric structure of the policy class [2]. Trust-region methods further constrain the policy update using a KL-divergence trust region to improve stability [9], and proximal policy optimization replaces the constrained problem with a clipped surrogate objective that is easier to optimize in practice [11]. These methods have been highly successful, but they still update stochastic policies mainly by reweighting sampled actions according to scalar value estimates.

On the other hand, deterministic policy gradient methods [12] optimize a deterministic policy $a = \pi_\theta(s)$ by differentiating the critic with respect to the action. The deterministic policy gradient theorem follows directly from applying the chain rule evaluated at the current policy action:

$$\nabla_\theta Q^{\pi_\theta}(s, \pi_\theta(s)) = \nabla_\theta \pi_\theta(s) \nabla_a Q^{\pi_\theta}(s, a)\big|_{a=\pi_\theta(s)}. \tag{1.8}$$

Theorem 1.3 (Deterministic policy gradient)**.** *Let $\pi_\theta : \mathcal{S} \to \mathcal{A}$ be a differentiable deterministic policy. Under standard regularity assumptions, the deterministic policy gradient is*

$$\nabla_\theta \mathcal{J}[\pi_\theta] = \frac{1}{1-\gamma} \mathbb{E}_{s\sim d^{\pi_\theta}} \left[\nabla_\theta \pi_\theta(s) \nabla_a Q^{\pi_\theta}(s, a)\big|_{a=\pi_\theta(s)} \right]. \tag{1.9}$$

This approach can be effective in high-dimensional continuous action spaces because it directly uses the local geometry of the learned value function in the action space. However, deterministic policies typically require an additional exploration mechanism, such as manually specified noise during data collection [3, 12]. Related stochastic value-gradient methods also exploit gradients through the value function or dynamics to learn continuous control policies [1], using reparameterization tricks. These methods suggest that gradients of action-value functions can provide useful directions for policy improvement. However, they do not give a general geometric rule for transporting an arbitrary stochastic policy distribution.

Wasserstein policy optimization (WPO) [18] provides such a perspective by viewing a stochastic policy $\pi(\cdot|s)$ as a probability distribution over actions and updating it through optimal transport geometry. Instead of increasing the probability of sampled actions according to their scalar values, WPO moves probability mass in action space along directions determined by $\nabla_a Q^\pi(s, a)$. More broadly, WPO belongs to a family of policy optimization methods that use Wasserstein geometry or Wasserstein gradient flow to improve the policy by transporting probability mass in action space rather than only by pointwise likelihood reweighting. Several other methods, together with WPO, can all be interpreted as different implementations or approximations of Wasserstein gradient flow in policy space [13, 16, 17].

In this project, we provide a relatively self-contained review of the basic concepts in optimal transport and Wasserstein gradient flow in order to understand the update rule of Wasserstein policy optimization. We then implement WPO from scratch and compare it with alternative policy update rules in a unified DDPG-style actor-critic framework. Our implementation is intentionally simpler than the original implementation and does not include all of its additional stabilization tricks. The code for the numerical experiments in this project is available at `https://github.com/hanyang-hu/wpo_exp`.

2 Optimal Transport

In this section, we briefly review the basic optimal transport background needed for Wasserstein policy optimization, focusing on the quadratic Wasserstein distance $\mathcal{W}_2$. The Wasserstein distance measures the minimal cost of transporting mass between probability measures, with the cost reflecting the physics of the underlying space. Unlike divergences that compare probability mass in a point-wise manner, this is particularly appealing for comparing stochastic policies: two policies that assign probability to nearby actions should be considered similar, even if their densities do not overlap exactly. Throughout this section, we adopt the notation of [5] and mainly consider probability measures. We will introduce the Monge and Kantorovich formulations of the optimal transport problem and conclude with Brenier's theorem and the Kantorovich potential for the quadratic-cost optimal transport problem.

To begin with, we define the push-forward operator, which describes how measures are transported through a map.

Definition 2.1 (Push-forward)**.** *For measurable $T : \mathcal{X} \to \mathcal{Y}$, the push-forward measure $\beta = T_\# \alpha \in \mathcal{P}(\mathcal{Y})$ of some $\alpha \in \mathcal{P}(\mathcal{X})$ satisfies*

$$\forall\, h \in \mathcal{C}(\mathcal{Y}), \qquad \int_{\mathcal{Y}} h(y) d\beta(y) = \int_{\mathcal{X}} h(T(x)) d\alpha(x). \tag{2.1}$$

Here, $\mathcal{P}(\cdot)$ denotes probability measures and $\mathcal{C}(\cdot)$ the set of continuous test functions. Equivalently, for any measurable set $B \subseteq \mathcal{Y}$, one has $\beta(B) = \alpha(T^{-1}(B))$.

Definition 2.2 (Monge problem)**.** *Given two measures $\alpha \in \mathcal{P}(\mathcal{X})$ and $\beta \in \mathcal{P}(\mathcal{Y})$, and a cost function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, the Monge problem seeks a transport map $T : \mathcal{X} \to \mathcal{Y}$ such that $\beta = T_{\#}\alpha$ while minimizing the total transportation cost:*

$$\inf_{T_{\#}\alpha=\beta} \int_{\mathcal{X}} c(x, T(x)) d\alpha(x). \tag{2.2}$$

The Monge formulation is intuitive, but it can be too restrictive: for instance, a feasible transport map $T : \mathcal{X} \to \mathcal{Y}$ may not exist. The Kantorovich relaxation replaces the map T with a transport plan π, which describes how mass at each point x may be split and sent to different locations y.

Definition 2.3 (Kantorovich problem)**.** *Given two measures $\alpha \in \mathcal{P}(\mathcal{X})$ and $\beta \in \mathcal{P}(\mathcal{Y})$, and a cost function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, the Kantorovich problem minimizes the transportation cost over all couplings π between α and β:*

$$\inf_{\pi \in \Pi(\alpha,\beta)} \int_{\mathcal{X}\times\mathcal{Y}} c(x, y) d\pi(x, y), \tag{2.3}$$

where

$$\Pi(\alpha, \beta) := \{\pi \in \mathcal{P}(\mathcal{X} \times \mathcal{Y}) \,|\, P_{\mathcal{X}\#}(\pi) = \alpha, P_{\mathcal{Y}\#}(\pi) = \beta\} \tag{2.4}$$

denotes the set of measures on $\mathcal{X} \times \mathcal{Y}$ whose marginals are α and β. Here, $P_{\mathcal{X}} : \mathcal{X} \times \mathcal{Y} \to \mathcal{X}$ and $P_{\mathcal{Y}} : \mathcal{X} \times \mathcal{Y} \to \mathcal{Y}$ are canonical projections.

Consider a cost of the form $c(x, y) = \|x - y\|^p$ in the Euclidean space $\Omega = \mathcal{X} = \mathcal{Y} \subseteq \mathbb{R}^n$ with $p \geq 1$, we can define the p-Wasserstein distance via the Kantorovich formulation

$$\mathcal{W}_p(\alpha, \beta) = \left(\inf_{\pi \in \Pi(\alpha,\beta)} \int_{\Omega\times\Omega} c(x, y) d\pi(x, y) \right)^{1/p}, \tag{2.5}$$

where $\alpha, \beta \in \mathcal{P}_p(\Omega) := \{\mu \in \mathcal{P}(\Omega) \,|\, \int_\Omega \|x\|^p d\mu(x) < \infty\}$. Under the usual assumptions, $\mathcal{W}_p$ defines a metric on $\mathcal{P}_p(\Omega)$. Interestingly, when $p = 2$, the Kantorovich and Monge problems are equivalent. Furthermore, the optimal transport map is characterized as the gradient of a convex function [5, 8].

Theorem 2.4 (Brenier's theorem)**.** *Let $\alpha, \beta \in \mathcal{P}_2(\Omega)$ where α is absolutely continuous with respect to the Lebesgue measure. Then there exists a convex function $\phi : \Omega \to \mathbb{R}$ such that the map $T = \nabla\phi$ is a push-forward from α to β, i.e. $T_{\#}\alpha = \beta$, and $(id, T)_{\#}\alpha \in \Pi(\alpha, \beta)$ minimizes the Kantorovich problem (2.3) with quadratic cost $c(x, y) = \|x - y\|^2$.*

Let $\varphi(x) = \frac{1}{2}\|x\|^2 - \phi(x)$, then $\nabla\varphi(x) = x - \nabla\phi(x)$ and we have $T = \nabla\phi = x - \nabla\varphi(x)$. The function φ is known as the Kantorovich potential associated with the quadratic-cost optimal transport problem.

3 Wasserstein Gradient Flow

In Euclidean space, gradient flow gives the continuous-time limit of gradient descent: it describes the path that decreases an objective most rapidly under the Euclidean metric. Analogously, once $\mathcal{W}_2$ defines a distance between probability measures, it induces a notion of steepest descent for objectives defined on distributions. The resulting Wasserstein gradient flow describes how a probability measure evolves over time by moving its mass along a velocity field during optimization. Since $\mathcal{W}_2$ defines a metric on $\mathcal{P}_2(\Omega)$, we can discretize the Wasserstein gradient flow via the Minimizing Movement Scheme [8]

$$\mu_{k+1}^\tau \in \arg\min_\mu F(\mu) + \frac{\mathcal{W}_2^2(\mu, \mu_k^\tau)}{2\tau} \tag{3.1}$$

where $F : \mathcal{P}_2(\Omega) \to \mathbb{R}$ is the objective functional and $\tau > 0$ is the step size.

To characterize the update direction, we need a notion of derivative for functionals defined on probability measures. This is given by the first variation.

Definition 3.1 (First variation)**.** *Let $F : \mathcal{P}_2(\Omega) \to \mathbb{R}$ be a functional. Its first variation is a function $\frac{\delta F}{\delta \mu}(\mu) : \Omega \to \mathbb{R}$ satisfying, for any smooth perturbation $\xi \in \mathcal{M}_0(\Omega)$ s.t. $\mu + \epsilon\xi \in \mathcal{P}(\Omega)$ for sufficiently small $\epsilon > 0$,*

$$\frac{d}{d\epsilon} F(\mu + \epsilon\xi)\bigg|_{\epsilon=0} = \int_\Omega \frac{\delta F}{\delta \mu}(\mu)(x) d\xi(x). \tag{3.2}$$

Here $\mathcal{M}_0(\Omega)$ denotes the space of signed measures on Ω with zero total mass, i.e., $\xi(\Omega) = 0$.

Equivalently, the first variation gives the first-order expansion

$$F(\mu + \epsilon\xi) = F(\mu) + \epsilon \int_\Omega \frac{\delta F}{\delta \mu}(\mu) d\xi(x) + o(\epsilon). \tag{3.3}$$

The first variation provides an optimality condition for the minimizing movement step (3.1). More specifically, the first variation of $F(\mu) + \frac{\mathcal{W}_2^2(\mu, \mu_k^\tau)}{2\tau}$ must be constant at $\mu = \mu_{k+1}^\tau$ so that the first-order term in (3.3) vanishes. The first variation of $\mu \mapsto \frac{1}{2}\mathcal{W}_2^2(\mu, \nu)$ is characterized by the following proposition.

Proposition 3.2. *The first variation of $\mu \mapsto \frac{1}{2}\mathcal{W}_2^2(\mu, \nu)$ is given by the Kantorovich potential*

$$\frac{\delta}{\delta \mu} \frac{1}{2} \mathcal{W}_2^2(\mu, \nu) = \varphi. \tag{3.4}$$

We shall verify this result intuitively.[1] To begin with, we perturb μ by a smooth displacement field $v : \Omega \to \mathbb{R}^n$ such that $v(x) = 0$ for $x \in \partial\Omega$, then the map $S_\epsilon : x \mapsto x + \epsilon v(x)$ defines a perturbed measure $\mu_\epsilon = (S_\epsilon)_\# \mu$ through push-forward. Although μ_ϵ is not generally of the exact form $\mu + \epsilon\xi$ for some $\xi \in \mathcal{M}_0(\Omega)$, it has such a form to first order: for any smooth test function h, we can expand

$$\begin{aligned} \int_\Omega h(x) d\mu_\epsilon(x) &= \int_\Omega h(S_\epsilon(x)) d\mu(x) \\ &= \int_\Omega h(x + \epsilon v(x)) d\mu(x) \\ &= \int_\Omega h(x) d\mu(x) + \epsilon \int_\Omega \langle \nabla h(x), v(x) \rangle d\mu(x) + o(\epsilon). \end{aligned} \tag{3.5}$$

Therefore, let $\xi = -\nabla \cdot (\mu v)$ and ρ be the density of μ, using integration-by-parts, we have

$$\begin{aligned} \int_\Omega h(x) d\xi(x) &= \int_\Omega h(x)[-\nabla \cdot (\rho(x) v(x))] dx \\ &= \cancel{- \int_{\partial\Omega} h(x) \rho(x) v(x) \cdot \mathbf{n}(x) dS} + \int_\Omega \rho(x) \langle \nabla h(x), v(x) \rangle dx \end{aligned} \tag{3.6}$$

since $v(x)$ vanishes at the boundary $\partial\Omega$. Furthermore, consider $h \equiv 1$, we have

$$\xi(\Omega) = \int_\Omega d\xi(x) = \int_\Omega \langle \nabla 1, v(x) \rangle d\mu(x) = 0 \tag{3.7}$$

i.e., $\mu_\epsilon = \mu + \epsilon\xi + o(\epsilon)$ in distribution sense and indeed $\xi \in \mathcal{M}_0(\Omega)$. Now, define $G(\mu) = \frac{1}{2}\mathcal{W}_2^2(\mu, \nu)$ and let $T : \Omega \to \Omega$ be the optimal transport from μ to ν. By Theorem 2.4, we can write $T(x) = x - \nabla\varphi(x)$ where φ is the Kantorovich potential. We have $(T \circ S_\epsilon^{-1})_\# \mu_\epsilon = T_\# \mu = \nu$. By Definition 2.2,

$$G(\mu_\epsilon) \le \int_\Omega \frac{1}{2} \|S_\epsilon(x) - T(x)\|^2 d\mu(x). \tag{3.8}$$

Since $S_\epsilon(x) = x + \epsilon v(x)$, we can expand and use (3.6) to obtain

$$\begin{aligned} G(\mu_\epsilon) &\le \int_\Omega \left[\frac{1}{2} \|x - T(x)\|^2 + \epsilon \langle x - T(x), v(x) \rangle + o(\epsilon) \right] d\mu(x) \\ &= G(\mu) + \epsilon \int_\Omega \langle \nabla\varphi(x), v(x) \rangle d\mu(x) + o(\epsilon) \\ &= G(\mu) + \epsilon \int_\Omega \varphi(x) d\xi(x) + o(\epsilon). \end{aligned} \tag{3.9}$$

[1] More rigorous discussions can be found in [7].

Applying the same comparison in the reverse direction yields equality, hence justifying Proposition (3.2).

The optimality condition for the minimizing movement step (3.1) is then given by

$$\frac{\delta F}{\delta \mu} + \frac{\varphi}{\tau} = \text{const} \tag{3.10}$$

where φ is the Kantorovich potential associated with the transport from μ_{k+1}^τ to μ_k^τ. Therefore, the "velocity" at point x from μ_{k+1}^τ to μ_k^τ is given by

$$-v^\tau(x) = \frac{T(x) - x}{\tau} = -\frac{\nabla \varphi(x)}{\tau} = \nabla \frac{\delta F}{\delta \mu}(\mu_{k+1}^\tau)(x). \tag{3.11}$$

As $\tau \to 0$, the sequence $\{\mu_k^\tau\}$ converges to a curve μ_t transported by a velocity field v_t

$$v_t = -\nabla \frac{\delta F}{\delta \mu}(\mu_t). \tag{3.12}$$

Since the measure evolves by transporting mass along the velocity field v_t, it satisfies a continuity equation. To see this, let X_t denote the flow generated by v_t, namely

$$\frac{d}{dt} X_t(x) = v_t(X_t(x)). \tag{3.13}$$

Since $\mu_t = (X_t)_\# \mu_0$, for every smooth test function h, we have

$$\begin{aligned} \frac{d}{dt} \int_\Omega h(x)\, d\mu_t(x) &= \frac{d}{dt} \int_\Omega h(X_t(x))\, d\mu_0(x) \\ &= \int_\Omega \left\langle \nabla h(X_t(x)), \frac{d}{dt} X_t(x) \right\rangle d\mu_0(x) \\ &= \int_\Omega \langle \nabla h(X_t(x)), v_t(X_t(x)) \rangle\, d\mu_0(x) \\ &= \int_\Omega \langle \nabla h(x), v_t(x) \rangle\, d\mu_t(x). \end{aligned} \tag{3.14}$$

Let ρ_t be the density of μ_t, i.e. $d\mu_t(x) = \rho_t(x)\, dx$, then the previous identity becomes

$$\int_\Omega h(x) \partial_t \rho_t(x)\, dx = \int_\Omega \langle \nabla h(x), v_t(x) \rangle\, \rho_t(x)\, dx. \tag{3.15}$$

By integration-by-parts, assuming the boundary term vanishes (e.g., $\rho_t v_t \cdot \mathbf{n} = 0$ on $\partial\Omega$), we have

$$\int_\Omega \langle \nabla h(x), v_t(x) \rangle\, \rho_t(x)\, dx = \cancel{\int_{\partial\Omega} h(x) \rho_t(x) v_t(x) \cdot \mathbf{n}(x) dS} - \int_\Omega h(x) \nabla \cdot (\rho_t(x) v_t(x))\, dx. \tag{3.16}$$

which implies

$$\int_\Omega h(x) \left[\partial_t \rho_t(x) + \nabla \cdot (\rho_t(x) v_t(x)) \right] dx = 0. \tag{3.17}$$

Since this holds for every smooth test function h, we obtain the continuity equation

$$\partial_t \rho_t + \nabla \cdot (\rho_t v_t) = 0. \tag{3.18}$$

Substituting (3.12), we obtain

$$\partial_t \rho_t - \nabla \cdot \left(\rho_t \nabla \frac{\delta F}{\delta \rho}(\rho_t) \right) = 0. \tag{3.19}$$

Here we slightly abuse notation by identifying μ_t with its density ρ_t. Under this identification, the measure-level first variation $\frac{\delta F}{\delta \mu}(\mu_t)$ is written as the density-level first variation $\frac{\delta F}{\delta \rho}(\rho_t)$.

Theorem 3.3 (Wasserstein gradient flow)**.** *Let $F : \mathcal{P}_2(\Omega) \to \mathbb{R}$ be a sufficiently regular functional admitting a first variation. The Wasserstein gradient flow of F is given by*

$$\frac{\partial \rho}{\partial t} = \nabla \cdot \left(\rho \nabla \frac{\delta F}{\delta \rho} \right). \tag{3.20}$$

Along sufficiently regular solutions, the energy F is non-increasing in time.

For example, the following functional

$$F(\rho) = \underbrace{\int_\Omega \rho(x) \log \rho(x) dx}_{\text{entropy}} + \underbrace{\int_\Omega V(x)\rho(x) dx}_{\text{potential}}, \tag{3.21}$$

has Wasserstein gradient flow

$$\partial_t \rho = \underbrace{\Delta \rho}_{\text{diffusion}} + \underbrace{\nabla \cdot (\rho \nabla V)}_{\text{drift}} \tag{3.22}$$

which is equivalent to the Fokker–Planck equation of an SDE, and admits a unique minimum over $\mathcal{P}(\Omega)$ [8]. However, for general functionals, we may only guarantee that the Wasserstein gradient flow is a descent dynamics for F. Indeed,

$$\begin{aligned} \frac{d}{dt} F(\rho_t) &= \int_\Omega \frac{\delta F}{\delta \rho}(\rho_t)\, \partial_t \rho_t \, dx \\ &= \int_\Omega \frac{\delta F}{\delta \rho}(\rho_t) \nabla \cdot \left(\rho_t \nabla \frac{\delta F}{\delta \rho}(\rho_t) \right) dx \\ &= \cancel{\int_{\partial\Omega} \frac{\delta F}{\delta \rho}(\rho_t) \left(\rho_t \nabla \frac{\delta F}{\delta \rho}(\rho_t) \right) \cdot \mathbf{n} dS} - \int_\Omega \rho_t \left\| \nabla \frac{\delta F}{\delta \rho}(\rho_t) \right\|^2 dx \le 0, \end{aligned} \tag{3.23}$$

where the integration-by-parts step uses suitable boundary conditions.

4 Wasserstein Policy Optimization

In RL, we may consider the expected return as a functional with respect to a stochastic policy π

$$\mathcal{J}[\pi] = \mathbb{E}_\tau \left[\sum_{t=0}^\infty \gamma^t r_t \right] = \sum_{t=0}^\infty \gamma^t \int r_t \mathbb{P}(s_0) \prod_{t'=0}^{t} \pi(a_{t'}|s_{t'}) \prod_{t'=0}^{t-1} \mathbb{P}(s_{t'+1}|s_{t'}, a_{t'}) d\tau_{0:t} \tag{4.1}$$

where τ denotes the trajectory $s_0, a_0, \dots$ and $\tau_{0:t}$ denotes the truncated trajectory $s_0, a_0, \dots, s_t, a_t$.

Theorem 4.1 (Functional policy gradient)**.** *The first variation of the expected return (4.1) is given by*

$$\frac{\delta J[\pi]}{\delta \pi}(s,a) = Q^\pi(s,a) \sum_{t=0}^\infty \gamma^t \mathbb{P}^\pi(s_t = s) = \frac{1}{1-\gamma} Q^\pi(s,a) d^\pi(s) \tag{4.2}$$

where $\mathbb{P}^\pi(s_t = s)$ denotes the probability that state s is visited at time t under the policy π and d^π denotes the discounted state occupancy function.

Proof. We perturb the stochastic policy π by $\delta\pi$ such that $\int_{\mathcal{A}} \delta\pi(a|s)\, da = 0$ for all $s \in \mathcal{S}$. For sufficiently small $\epsilon > 0$, define $\pi_\epsilon(a|s) = \pi(a|s) + \epsilon\delta\pi(a|s)$, then

$$\mathcal{J}[\pi_\epsilon] - \mathcal{J}[\pi] = \sum_{t=0}^\infty \gamma^t \int r_t p_0(s_0) \left[\prod_{k=0}^{t} \pi_\epsilon(a_k|s_k) - \prod_{k=0}^{t} \pi(a_k|s_k) \right] \prod_{k=0}^{t-1} \mathbb{P}(s_{k+1}|s_k, a_k) d\tau_{0:t}. \tag{4.3}$$

Expanding the product to first order in ϵ gives

$$\prod_{k=0}^{t} (\pi + \epsilon\delta\pi)(a_k|s_k) = \prod_{k=0}^{t} \pi(a_k|s_k) + \epsilon \sum_{\ell=0}^{t} \delta\pi(a_\ell|s_\ell) \prod_{\substack{k=0 \\ k\neq\ell}}^{t} \pi(a_k|s_k) + o(\epsilon). \tag{4.4}$$

Switching the order of summation, we can obtain

$$\begin{aligned}
\mathcal{J}[\pi_\epsilon] - \mathcal{J}[\pi] &= \epsilon \sum_{t=0}^{\infty} \gamma^t \sum_{\ell=0}^{t} \int r_t p_0(s_0) \delta\pi(a_\ell|s_\ell) \prod_{\substack{k=0 \\ k\neq\ell}}^{t} \pi(a_k|s_k) \prod_{k=0}^{t-1} \mathbb{P}(s_{k+1}|s_k, a_k) d\tau_{0:t} + o(\epsilon) \\
&= \epsilon \sum_{\ell=0}^{\infty} \sum_{t=\ell}^{\infty} \gamma^t \int r_t p_0(s_0) \delta\pi(a_\ell|s_\ell) \prod_{\substack{k=0 \\ k\neq\ell}}^{t} \pi(a_k|s_k) \prod_{k=0}^{t-1} \mathbb{P}(s_{k+1}|s_k, a_k) d\tau_{0:t} + o(\epsilon) \\
&= \epsilon \sum_{\ell=0}^{\infty} \int \mathbb{P}^\pi(s_\ell = s) \delta\pi(a_\ell|s_\ell) \left[\int \sum_{t=\ell}^{\infty} \gamma^t r_t \mathbb{P}^\pi(\tau_{\ell+1:t}|s_\ell, a_\ell) d\tau_{\ell+1:t} \right] ds_\ell da_\ell + o(\epsilon) \\
&= \epsilon \sum_{\ell=0}^{\infty} \int \mathbb{P}^\pi(s_\ell = s) \delta\pi(a|s) \left[\mathbb{E}_\tau \left[\sum_{t=\ell}^{\infty} \gamma^t r_t \,\middle|\, s_\ell = s,\ a_\ell = a \right] \right] da\, ds + o(\epsilon) \\
&= \epsilon \int \sum_{\ell=0}^{\infty} \mathbb{P}^\pi(s_\ell = s) \delta\pi(a|s) \gamma^\ell Q^\pi(s, a) da\, ds + o(\epsilon)
\end{aligned} \tag{4.5}$$

since

$$\mathbb{P}^\pi(s_\ell = s) = \int p_0(s_0) \prod_{k=0}^{\ell-1} [\pi(a_k|s_k) \mathbb{P}(s_{k+1}|s_k, a_k)] d\tau_{0:\ell-1} \tag{4.6}$$

$$\mathbb{P}^\pi(\tau_{\ell+1:t}|s_\ell, a_\ell) = \prod_{k=\ell+1}^{t} \pi(a_k|s_k) \prod_{k=\ell}^{t-1} \mathbb{P}(s_{k+1}|s_k, a_k) \tag{4.7}$$

We may conclude that

$$\frac{\delta \mathcal{J}[\pi]}{\delta \pi}(s, a) = \sum_{\ell=0}^{\infty} \gamma^\ell \mathbb{P}^\pi(s_\ell = s) Q^\pi(s, a). \tag{4.8}$$

from Definition 3.1. □

Similar to policy gradient, since $\int_{\mathcal{A}} \delta\pi(a|s)\, da = 0$, we can replace $Q^\pi(s, a)$ with the advantage function $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$. In fact, consider a parameterized policy π_θ, we have

$$\begin{aligned}
\nabla_\theta \mathcal{J}[\pi_\theta] &= \int \frac{\delta \mathcal{J}[\pi_\theta]}{\delta \pi}(s, a) \nabla_\theta \pi_\theta(a|s) ds da \\
&= \int \frac{1}{1-\gamma} Q^\pi(s, a) d^\pi(s) \nabla_\theta \pi_\theta(a|s) ds da
\end{aligned} \tag{4.9}$$

which coincides with the classical policy gradient [14] (i.e., Theorem 1.2). Therefore, Theorem 4.1 can be interpreted as the functional generalization of Theorem 1.2.

Wasserstein policy optimization (WPO) [6] is motivated by applying the Wasserstein gradient flow perspective (Theorem 3.3) to the policy functional $\mathcal{J}[\pi]$. More specifically, from Theorem 4.1, the Wasserstein policy improvement dynamics should take the form[2]

$$\partial_t \pi_t(a|s) + \nabla_a \cdot (\pi_t(a|s) v_t(s, a)) = 0, \qquad v_t(s, a) = \nabla_a \frac{\delta \mathcal{J}}{\delta \pi} \propto \nabla_a Q^{\pi_t}(s, a) \tag{4.10}$$

if we treat $d^\pi(s)/(1-\gamma)$ as a positive state-dependent factor since $d^\pi(s)$ typically emerges implicitly in the update as the sampling frequency. Thus, WPO differs from the classical policy gradient: instead of reweighting sampled actions according to $Q^\pi(s, a)$ [10, 14], it transports probability mass in action space along the direction $\nabla_a Q^\pi(s, a)$. In this sense, WPO inherits the use of action-value gradients from deterministic policy gradient (DPG) methods [12] while still optimizing stochastic policies as probability distributions.

[2]Notice the change of sign since Theorem 3.3 was written for minimization while we want to maximize the functional $\mathcal{J}$ of the expected return.

In practice, however, the policy is represented by a finite-dimensional parametric model π_θ (e.g., a neural network). Therefore, WPO needs to project the nonparametric Wasserstein flow back onto the parametric policy class. The idea is to choose a parameter update $\Delta\theta$ such that the change induced by the parametric policy $\pi_{\theta+\Delta\theta}$ best matches the ideal nonparametric flow. This projection can be formulated as

$$\Delta\theta = \arg\min_{\Delta\theta} D_{\mathrm{KL}}\left[\pi_\theta \,\middle\|\, \pi_\theta + \frac{\partial \pi_\theta}{\partial t}dt - \nabla_\theta \pi_\theta \Delta\theta\right]. \tag{4.11}$$

Here, $\frac{\partial \pi_\theta}{\partial t}dt$ represents the infinitesimal change prescribed by the ideal Wasserstein gradient flow, while $\nabla_\theta \pi_\theta \Delta\theta$ represents the change achievable by moving the parameter θ. Notice that the RHS of (4.11) is written for a fixed state s because the Wasserstein gradient flow acts on the conditional action distribution $\pi_\theta(\cdot|s)$. In practice, these state-wise projected directions are estimated over a batch of state samples encountered by the policy and aggregated (thereby implicitly incorporating the factor $d^\pi(s)$).

For a parametric distribution p_θ, the KL divergence can be approximated by its second-order Taylor expansion [4]

$$\begin{aligned}
D_{\mathrm{KL}}[p_\theta \,\|\, p_{\theta+\Delta\theta}] &= \mathbb{E}_{z\sim p_\theta}\left[\log p_\theta(z) - \log p_{\theta+\Delta\theta}(z)\right] \\
&\approx \mathbb{E}_{z\sim p_\theta}\left[\log p_\theta(z) - \left(\log p_\theta(z) + \nabla_\theta \log p_\theta(z)^T \Delta\theta + \frac{1}{2}\Delta\theta^T \nabla_\theta^2 \log p_\theta(z) \Delta\theta\right)\right] \\
&= \cancel{-\mathbb{E}_{z\sim p_\theta}\left[\nabla_\theta \log p_\theta(z)\right]^T \Delta\theta} - \frac{1}{2}\Delta\theta^T \mathbb{E}_{z\sim p_\theta}\left[\nabla_\theta^2 \log p_\theta(z)\right]\Delta\theta \\
&= \frac{1}{2}\Delta\theta^T \mathbb{E}_{z\sim p_\theta}\left[\nabla_\theta \log p_\theta(z) \nabla_\theta \log p_\theta(z)^\top\right]\Delta\theta \\
&= \frac{1}{2}\Delta\theta^T \mathcal{F}_\theta \Delta\theta.
\end{aligned} \tag{4.12}$$

Since the residual $\frac{\partial \pi_\theta}{\partial t}dt - \nabla_\theta \pi_\theta \Delta\theta$ is usually small, we have

$$\begin{aligned}
D_{\mathrm{KL}}\left[\pi_\theta \,\middle\|\, \pi_\theta + \frac{\partial \pi_\theta}{\partial t}dt - \nabla_\theta \pi_\theta \Delta\theta\right] &\approx \frac{1}{2}\begin{pmatrix} dt \\ -\Delta\theta \end{pmatrix}^T \begin{pmatrix} \mathcal{F}_{tt} & \mathcal{F}_{t\theta}^T \\ \mathcal{F}_{t\theta} & \mathcal{F}_{\theta\theta} \end{pmatrix} \begin{pmatrix} dt \\ -\Delta\theta \end{pmatrix} \\
\mathcal{F}_{tt} &= \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\left(\frac{\partial \log \pi_\theta(a|s)}{\partial t}\right)^2\right], \\
\mathcal{F}_{t\theta} &= \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\frac{\partial \log \pi_\theta(a|s)}{\partial t}\nabla_\theta \log \pi_\theta(a|s)\right], \\
\mathcal{F}_{\theta\theta} &= \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\nabla_\theta \log \pi_\theta(a|s) \nabla_\theta \log \pi_\theta(a|s)^\top\right].
\end{aligned} \tag{4.13}$$

where $\mathcal{F}_{tt}, \mathcal{F}_{t\theta}, \mathcal{F}_{\theta\theta}$ are the Fisher information matrices. Therefore, $\Delta\theta \approx \mathcal{F}_{\theta\theta}^{-1}\mathcal{F}_{t\theta}dt$.[3]

[3]The WPO paper writes the update $\Delta\theta \approx \mathcal{F}_{\theta\theta}^{-1}\mathcal{F}_{t\theta}$ in a form without an explicit dt factor.

By direct computation, we have (assuming $\nabla_\theta \pi(a|s)$ vanishes at $\partial\mathcal{A}$)

$$
\begin{aligned}
\mathcal{F}_{t\theta} &= \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\frac{\partial \log \pi_\theta(a|s)}{\partial t}\nabla_\theta \log \pi_\theta(a|s)\right] \\
&= \int \frac{\partial \pi_\theta(a|s)}{\partial t}\nabla_\theta \log \pi_\theta(a|s)\, da \\
&\propto -\int \nabla_a \cdot \left(\pi_\theta(a|s)\nabla_a Q^\pi(s,a)\right)\nabla_\theta \log \pi_\theta(a|s)\, da \\
&= -\int \left[\pi_\theta(a|s)\Delta_a Q^\pi(s,a) + \nabla_a \pi_\theta(a|s)\cdot\nabla_a Q^\pi(s,a)\right]\nabla_\theta \log \pi_\theta(a|s)\, da \\
&= -\int \nabla_\theta \pi_\theta(a|s)\Delta_a Q^\pi(s,a)\, da - \int \left(\nabla_a \pi_\theta(a|s)\cdot\nabla_a Q^\pi(s,a)\right)\nabla_\theta \log \pi_\theta(a|s)\, da \\
&= \int \left(\nabla_a \nabla_\theta \pi_\theta(a|s)\right)^\top \nabla_a Q^\pi(s,a)\, da - \cancel{\int_{\partial\mathcal{A}} \nabla_\theta \pi_\theta(a|s)\nabla_a Q^\pi(s,a)\cdot \mathbf{n}(a)\, dS(a)} \\
&\qquad - \int \left(\nabla_a \pi_\theta(a|s)\cdot\nabla_a Q^\pi(s,a)\right)\nabla_\theta \log \pi_\theta(a|s)\, da \\
&= \int \left[\left(\nabla_a \nabla_\theta \pi_\theta(a|s)\right)^\top - \pi_\theta(a|s)^{-1}\nabla_\theta \pi_\theta(a|s)\left(\nabla_a \pi_\theta(a|s)\right)^\top\right]\nabla_a Q^\pi(s,a)\, da \\
&= \int \pi_\theta(a|s)\left[\left(\nabla_a \nabla_\theta \log \pi_\theta(a|s)\right)^\top\right]\nabla_a Q^\pi(s,a)\, da \\
&= \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\left(\nabla_a \nabla_\theta \log \pi_\theta(a|s)\right)^\top \nabla_a Q^\pi(s,a)\right] \\
&= \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\nabla_\theta \nabla_a \log \pi_\theta(a|s)\nabla_a Q^\pi(s,a)\right]
\end{aligned}
\tag{4.14}
$$

Therefore, the *Wasserstein Policy Optimization* update for a single state input s is given by

$$
\theta \leftarrow \theta + \eta \mathcal{F}_{\theta\theta}^{-1}\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\nabla_\theta \nabla_a \log \pi_\theta(a|s)\nabla_a Q^\pi(s,a)\right]. \tag{4.15}
$$

where $\eta > 0$ is the step size (i.e., learning rate).

5 Experiments

5.1 Toy Example

We reproduce the illustrative mixture-of-Gaussians experiment from the original Wasserstein Policy Optimization paper [6]. We consider the state-independent action-value function

$$
Q(a) = -\frac{1}{100}a^4 + a^2,
$$

which is non-concave and has two symmetric global maxima at $a = \pm\sqrt{50}$. The goal is to optimize a one-dimensional stochastic policy represented by a mixture of Gaussians,

$$
\pi(a) = \sum_{i=1}^{K} \rho_i \mathcal{N}(a \mid \mu_i, \sigma_i^2), \qquad \sum_{i=1}^{K}\rho_i = 1,
$$

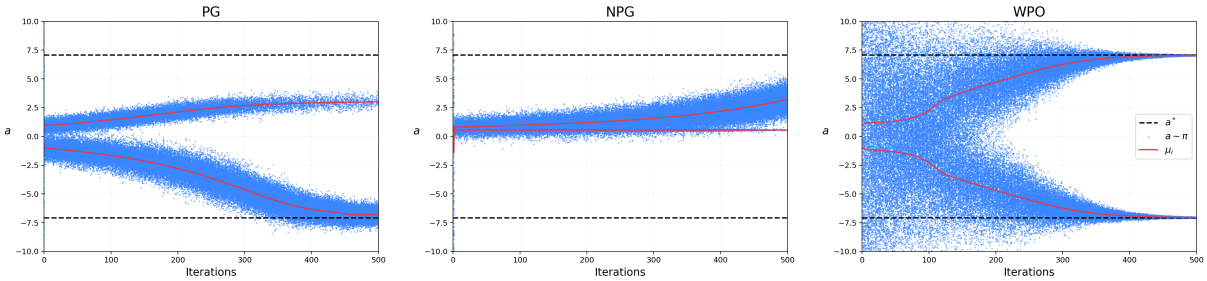
where ρ_i, μ_i, and σ_i denote the mixture weight, mean, and standard deviation of the i-th component. This numerical example is used as a controlled setting to compare the qualitative behavior of the standard policy gradient updates and WPO.

To enforce the constraint $\sum_{i=1}^{K}\rho_i = 1$, we parameterize the mixture weights by logits α_i:

$$
\rho_i = \frac{\exp(\alpha_i)}{\sum_{j=1}^{K}\exp(\alpha_j)}.
$$

In addition, we parameterize $\sigma_i = \exp(\ell_i)$. The policy parameters are therefore

$$
\theta = (\mu_1, \ell_1, \alpha_1, \ldots, \mu_K, \ell_K, \alpha_K).
$$



(a) Policy Gradient (b) Natural Policy Gradient (c) Wasserstein Policy Optimization

Figure 1: One-dimensional mixture-of-Gaussians toy example under PG, NPG, and WPO. Blue dots denote sampled actions from the current policy, red curves show the evolution of the component means, and dashed black lines mark the two target modes $\pm\sqrt{50}$. PG slowly moves the component means toward the two modes but shrinks its variance early. NPG becomes overconfident early and collapses toward one side of the action space. WPO converges stably toward both target modes.

In our implementation, the standard policy gradient update, the natural policy gradient update, and the WPO update are estimated using automatic differentiation and Monte Carlo sampling:

$$\Delta_{\text{PG}} \approx \frac{1}{M}\sum_{m=1}^{M} Q(a_m)\nabla_\theta \log \pi_\theta(a_m), \qquad a_m \sim \pi_\theta,$$

$$\Delta_{\text{NPG}} \approx \mathcal{F}_{\theta\theta}^{-1}\frac{1}{M}\sum_{m=1}^{M} Q(a_m)\nabla_\theta \log \pi_\theta(a_m), \qquad a_m \sim \pi_\theta,$$

$$\Delta_{\text{WPO}} \approx \mathcal{F}_{\theta\theta}^{-1}\frac{1}{M}\sum_{m=1}^{M} \nabla_\theta\nabla_a \log \pi_\theta(a_m)\nabla_a Q(a_m), \qquad a_m \sim \pi_\theta.$$

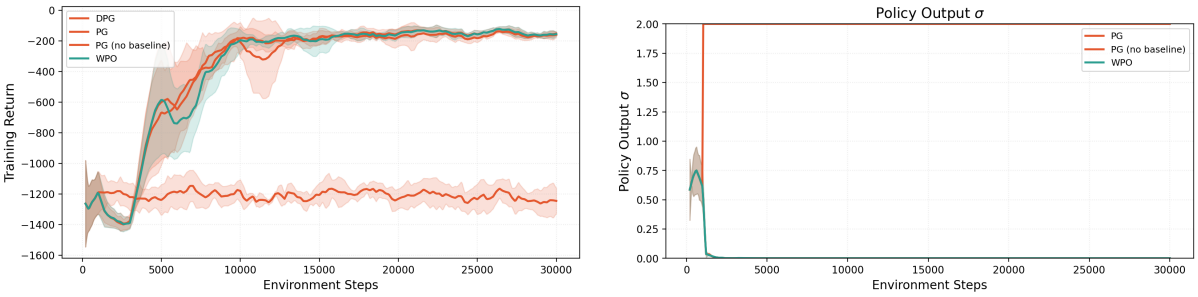
We include the natural policy gradient update [2] as an additional baseline in order to isolate the effect of the Fisher-preconditioned natural update from the Wasserstein gradient flow direction used by WPO. Similar to the original paper [6], we initialize the policy with $K = 2$, $\alpha_i = 0$, $\sigma_i = 10$, and $\mu_i = \pm 1$. For all methods, we use batch size $M = 1024$, step size $\eta = 0.005$, and run the optimization for 500 iterations. For the NPG and WPO updates, we add a damping factor of 0.01 to the empirical Fisher matrix $\mathcal{F}_{\theta\theta}$ before inversion to improve numerical stability.[4]

The results are illustrated in Fig. 1. Despite differences in implementation details, we observe behavior that is qualitatively consistent with the original WPO paper [6]. Under PG, the two component means gradually move toward the two modes, but the variances shrink early during training, which can reduce exploration and slow subsequent convergence. Under NPG, the policy becomes overconfident early and effectively collapses toward a single mode, failing to capture the symmetric bi-modal structure of the objective. This is consistent with Kakade's claim that natural policy gradient can push the policy toward greedy optimal actions, especially in regions far from the maximum [2]. In contrast, WPO transports probability mass toward both optima in a stable and balanced manner. This supports the interpretation that WPO updates the policy by moving probability mass along the gradient of $Q(a)$, rather than merely reweighting sampled actions by their scalar values.

5.2 Inverted Pendulum

The previous toy example studies the behavior of WPO in a state-independent one-dimensional setting. To further evaluate whether WPO works on more realistic reinforcement learning problems, we consider the inverted pendulum environment from Gymnasium [15]. This environment is a standard benchmark in which the agent must learn a feedback policy that balances the pendulum by applying

[4]In practice, we observed that our WPO implementation is more numerically robust than the NPG implementation with respect to the choice of Fisher damping factor and learning rate. We hypothesize that this is because NPG update directions are weighted by $Q(a)$, which can have high variance for tail samples.



(a) Training return on the inverted pendulum.

(b) Evolution of std for stochastic policy outputs.

Figure 2: Inverted pendulum results. Figure (a) compares the training returns of PG, DPG, WPO, and a variant of PG without baseline regularization. The unregularized PG update fails in this off-policy actor-critic setting. Figure (b) shows the evolution of the stochastic policy standard deviation. Without baseline regularization, the stochastic policy update can drive the policy noise to unstable values, whereas the regularized PG update and WPO both rapidly converge toward nearly deterministic policies.

continuous torques. Unlike the toy example, the value of an action depends on the current state and on the long-term effect of the resulting trajectory.

In this experiment, we compare different update strategies, including vanilla policy gradient [10], deterministic policy gradient [3, 12], and WPO [6] under the same actor-critic framework similar to [3]. For PG and WPO, the policy is modeled as a state-dependent diagonal Gaussian distribution,

$$\pi_\theta(a \mid s) = \prod_{d=1}^{D} \mathcal{N}(a_d \mid [\mu_\theta(s)]_d, [\sigma_\theta(s)]_d^2),$$

where the mean and standard deviation are outputs of a neural network. For DDPG, the policy is a deterministic neural network $a = \pi_\theta(s)$. Following the original WPO paper [6], we avoid explicitly forming the full Fisher information matrix $\mathcal{F}_{\theta\theta}$ for the neural network policy π_θ. Instead, a simplified scaling trick is applied for Gaussian policy outputs, which scales the log-likelihood gradients by σ_i^2. This ensures that $\nabla_a \log \pi_\theta(a|s)$ does not explode as the policy converges to a deterministic result. More specifically, the following gradients are modified during back-propagation

$$\overline{\nabla_\mu} \log \mathcal{N}(a \mid \mu, \sigma^2) = \sigma^2 \nabla_\mu \log \mathcal{N}(a \mid \mu, \sigma^2),$$
$$\overline{\nabla_\sigma} \log \mathcal{N}(a \mid \mu, \sigma^2) = \frac{1}{2}\sigma^2 \nabla_\sigma \log \mathcal{N}(a \mid \mu, \sigma^2).$$

The DDPG-style actor-critic algorithm that we use in this project is illustrated in Algorithm 1. The main difference between the methods is the actor update: PG updates the stochastic policy using log-likelihood gradients weighted by the critic value, DPG updates the deterministic action directly through $\nabla_a Q_\psi(s, a)$, and WPO uses the action-value function gradient as the transport direction together with the scaling trick described above.

For all methods, the actor and critic are represented by neural networks with one hidden layer of 256 units and ReLU activation. We train each method for 30000 environment steps, corresponding to 150 episodes, with 1000 warm-up steps and a replay-buffer batch size of 128. The discount factor is $\gamma = 0.99$, and the target networks are updated by Polyak averaging with coefficient $\tau = 0.005$. Both the actor and critic are optimized using Adam with learning rate $\eta = 10^{-3}$, and their gradients are clipped with maximum norm 10. For policy gradient and WPO using stochastic policies, we use $M = 32$ action samples per state to estimate both the stochastic Bellman target and the stochastic policy update. For DPG, exploration is performed using an Ornstein–Uhlenbeck noise process (see Algorithm 1).

The results are illustrated in Fig. 2. For the default stochastic policy gradient update, we found that the raw critic-weighted likelihood-gradient estimator completely failed in this off-policy actor-critic setting. Since the critic is learned from experience replay and can be inaccurate early in training, directly weighting $\nabla_\theta \log \pi_\theta(a \mid s)$ by the scalar value $Q_\psi(s, a)$ can produce high-variance policy gradients. Empirically, this leads to unstable updates of the Gaussian policy outputs, and the policy standard deviation can explode rapidly.

To stabilize policy gradient updates, we subtract a sample-based baseline from the critic values before computing the weighted policy gradient estimate. Specifically, for each replay state s_i, we estimate the state-value baseline by

$$b_i = V^\pi(s_i) = \mathbb{E}_{a\sim\pi_\theta(\cdot|s_i)}\left[Q_\psi(s_i, a)\right] \approx \frac{1}{M}\sum_{k'=1}^{M} Q_\psi(s_i, \tilde{a}_i^{(k')})$$

since $\tilde{a}_i^{(k)} \sim \pi_\theta(\cdot \mid s_i)$. The resulting baseline-regularized policy gradient update is

$$\Delta_{\mathrm{PG}} = \frac{1}{NM}\sum_{i=1}^{N}\sum_{k=1}^{M} \underbrace{\left(Q_\psi(s_i, \tilde{a}_i^{(k)}) - b_i\right)}_{\text{advantage}} \nabla_\theta \log \pi_\theta(\tilde{a}_i^{(k)} \mid s_i).$$

This modification does not change the expected update direction Δ_{PG} when the baseline is independent of the sampled action, but it reduces the variance of the finite-sample update. This regularization substantially improved PG performance. Moreover, WPO achieves comparable (if not better) learning behavior to DPG and regularized PG by using the action-value function gradient $\nabla_a Q_\psi(s, a)$ to define a more structured transport direction in action space.

We also observe that, for both WPO and the baseline-regularized PG update, the stochastic policy quickly approaches a nearly deterministic policy. This is likely because the inverted pendulum task is simple enough that an almost deterministic feedback controller is sufficient. This observation is consistent with the original WPO paper [6], which reports that WPO without KL regularization can prematurely collapse to a deterministic policy and fail to learn in more challenging settings.

6 Conclusion

In this project, we briefly reviewed basic notions from optimal transport and Wasserstein gradient flows to better understand Wasserstein policy optimization (WPO) [6], which allows using gradients of the value in the action space to update an arbitrary stochastic policy.

We evaluated WPO in two numerical settings. In the one-dimensional mixture-of-Gaussians toy example, our implementation reproduced the qualitative behavior reported in the original paper: WPO moved probability mass toward both optima in a stable and balanced manner, whereas policy gradient and natural policy gradient were more prone to premature variance shrinkage or mode collapse. In the inverted pendulum experiment, we compared different policy update rules within a unified DDPG-style actor-critic framework. Our implementation is simpler than the official WPO implementation and does not include additional stabilizing techniques such as KL regularization. Nevertheless, it shows that WPO can learn a stochastic policy off the shelf and achieve performance at least comparable to deterministic policy gradient, without requiring manually specified exploration noise. The experiment also confirms the observation that, without further regularization, WPO tends to converge quickly toward a nearly deterministic policy.

On the theoretical side, recent work has begun to study convergence guarantees for Wasserstein policy optimization, including linear convergence results under entropy regularization [18]. While a full theoretical treatment is beyond the scope of this project, these results suggest that the Wasserstein viewpoint may provide a useful framework for analyzing policy optimization beyond its empirical motivation.

Algorithm 1 DDPG-style actor-critic algorithm with PG, DPG, and WPO policy updates

Require: Initialize policy π_θ, critic Q_ψ, target policy $\pi_{\theta'}$, target critic $Q_{\psi'}$ with parameters θ, ψ, $\theta' \leftarrow \theta$, and $\psi' \leftarrow \psi$ respectively. Initialize replay buffer $\mathcal{D}$. Select Method $\in \{\text{PG}, \text{DPG}, \text{WPO}\}$

for episode $= 1, \ldots, M$ **do**
  Initialize exploration Ornstein-Uhlenbeck process noise $\mathcal{N}$ ▷ Only when Method = DPG.
  Receive initial observation state s_1
  for $t = 1, \ldots, T$ **do**
    if Method = DPG **then**
      Select action $a_t = \pi_\theta(s_t) + \mathcal{N}_t$ ▷ Add exploration noise.
    else
      Sample action $a_t \sim \pi_\theta(\cdot \mid s_t)$
    end if
    Execute action a_t, observe reward r_t and next state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in replay buffer $\mathcal{D}$
    Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from $\mathcal{D}$
    if Method = DPG **then**
      Set $y_i = r_i + \gamma Q_{\psi'}(s_{i+1}, \pi_{\theta'}(s_{i+1}))$
    else ▷ M is the number of action samples.
      Set $y_i = r_i + \gamma \frac{1}{M} \sum_{k=1}^{M} Q_{\psi'}(s_{i+1}, a_{i+1}^{(k)})$ where $a_{i+1}^{(k)} \sim \pi_{\theta'}(\cdot \mid s_{i+1})$
    end if
    Update critic by minimizing the mean-squared Bellman error $L(\psi) = \frac{1}{N} \sum_i \left(y_i - Q_\psi(s_i, a_i)\right)^2$
    if Method = PG **then**
      Sample $\tilde{a}_i^{(k)} \sim \pi_\theta(\cdot \mid s_i)$ for $k = 1, \ldots M$
      Update policy with the estimated policy gradient

$$\Delta_{\text{PG}} = \frac{1}{NM} \sum_i \sum_k Q_\psi(s_i, \tilde{a}_i^k) \nabla_\theta \log \pi_\theta(\tilde{a}_i^k \mid s_i).$$

    else if Method = DPG **then**
      Update policy with the estimated deterministic policy gradient

$$\Delta_{\text{DPG}} = \frac{1}{N} \sum_i \nabla_a Q_\psi(s_i, a)\big|_{a=\pi_\theta(s_i)} \nabla_\theta \pi_\theta(s_i).$$

    else if Method = WPO **then**
      Sample $\tilde{a}_i^{(k)} \sim \pi_\theta(\cdot \mid s_i)$ for $k = 1, \ldots, M$
      Update policy with Wasserstein policy optimization ▷ $\overline{\nabla_\theta}$ denotes the scaled gradient.

$$\Delta_{\text{WPO}} = \frac{1}{NM} \sum_i \sum_k \overline{\nabla_\theta} \nabla_a \log \pi_\theta(\tilde{a}_i^{(k)} \mid s_i) \nabla_a Q_\psi(s_i, \tilde{a}_i^{(k)}).$$

    end if
    Update policy parameters $\theta \leftarrow \theta + \eta \Delta_{\text{Method}}$
    Update target networks $\psi' \leftarrow \tau\psi + (1-\tau)\psi'$ and $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$.
  end for
end for

References

[1] Nicolas Heess, Greg Wayne, David Silver, Timothy Lillicrap, Yuval Tassa, and Tom Erez. Learning continuous control policies by stochastic value gradients, 2015.

[2] Sham Kakade. A natural policy gradient. In *Proceedings of the 15th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, page 1531–1538, Cambridge, MA, USA, 2001. MIT Press.

[3] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.

[4] Razvan Pascanu and Yoshua Bengio. Revisiting natural gradient for deep networks. *arXiv preprint arXiv:1301.3584*, 2013.

[5] Gabriel Peyré and Marco Cuturi. Computational optimal transport, 2020.

[6] David Pfau, Ian Davies, Diana Borsa, Joao G. M. Araujo, Brendan Tracey, and Hado van Hasselt. Wasserstein policy optimization, 2025.

[7] Filippo Santambrogio. *Optimal transport for applied mathematicians*. Progress in Nonlinear Differential Equations and Their Applications. Birkhauser, Basel, Switzerland, 1 edition, October 2015.

[8] Filippo Santambrogio. Euclidean, Metric, and Wasserstein gradient flows: an overview, 2016.

[9] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2017.

[10] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018.

[11] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[12] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page I–387–I–395. JMLR.org, 2014.

[13] Jun Song, Niao He, Lijun Ding, and Chaoyue Zhao. Provably convergent policy optimization via metric-aware trust region methods, 2023.

[14] Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'99, page 1057–1063, Cambridge, MA, USA, 1999. MIT Press.

[15] Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A standard interface for reinforcement learning environments, 2025.

[16] Ruiyi Zhang, Changyou Chen, Chunyuan Li, and Lawrence Carin. Policy optimization as wasserstein gradient flows, 2018.

[17] Zhaoyu Zhu, Shuhan Zhang, Rui Gao, and Shuang Li. Wasserstein proximal policy gradient, 2026.

[18] David Šiška and Yufei Zhang. A note on convergence of wasserstein policy optimization, 2026.