Optimization Methods for Reinforcement Learning: Theory and Applications

Carlo Alfano

Linacre College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Trinity 2025

Abstract

Reinforcement learning (RL) is a powerful paradigm for training agents to make optimal decisions in sequential environments through interaction and feedback. A central aspect of RL research lies in the development and application of efficient optimization methods that enable agents to learn complex behaviors. This thesis contributes to this area by introducing novel optimization techniques and by analyzing their applications in various RL settings, encompassing cooperative multi-agent systems, policy optimization with general parameterizations, and preference-based learning.

In the context of cooperative multi-agent reinforcement learning, where multiple agents collaborate to achieve a common goal, a significant challenge arises from the exponential increase in complexity with the number of agents. To address this, we introduce a scalable algorithm based on Natural Policy Gradient, which leverages local information exchange between neighboring agents within a defined range. We theoretically demonstrate that, under standard assumptions on spatial decay of correlations, our algorithm converges to the globally optimal policy with a statistical and computational complexity that remains independent of the number of agents.

This thesis further investigates the use of mirror descent as a versatile framework for policy optimization in reinforcement learning. We develop Approximate Mirror Policy Optimization (AMPO) and establish for it the first linear convergence guarantee for a policy-gradient-based method that accommodates general policy parameterizations. Furthermore, we empirically examine the impact of different mirror maps within the Policy Mirror Descent (PMD) and AMPO frameworks, revealing that the commonly used negative entropy is not always the best choice.

Finally, we extend the application of mirror descent to the domain of preference optimization (PO), a crucial technique for aligning agents with human preferences. In particular, we propose a novel family of algorithms called Mirror Preference Optimization (MPO). Through evolutionary strategies, we identify specialized MPO algorithms tailored to specific characteristics of preference datasets, such as mixed-quality or noisy data. Our findings demonstrate that these discovered algorithms outperform existing state-of-the-art PO methods in both simulated robotic tasks and a large language model alignment task, highlighting the effectiveness of our mirror descent-based approach for preference learning.

Optimization Methods for Reinforcement Learning: Theory and Applications

Carlo Alfano

Linacre College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Trinity 2025

Acknowledgements

The years spent writing this thesis have been the best and hardest of my life, and I would not have made it without all the wonderful people I have met.

First of all I would like to thank my supervisor, Patrick Rebeschini, who gave me this incredible opportunity and mentored me into becoming a scientist. Until the day I retire, I will hear his voice saying "Is this sentence/paragraph clear?" every time I write a paper. I would then like to thank Silvia Sapora, for showing me that research is more fun when you can get angry at python bugs together. I am also thankful to Tom Rainforth and Matthieu Geist for reviewing this thesis and for gifting me a truly pleasant memory of my thesis defense.

Outside of research, I am grateful to all my friends in Oxford, who have made this experience truly magical and have enriched my life more than I could imagine. I am also thankful to my friends in Rome for always welcoming me back and making me feel like I never left.

I am beyond grateful to my family for their unwavering support and encouragement. To my father, for being the first person I call whenever I have a problem and for always having a solution. To my mother, for being the kindest person I know and for teaching me the value of lightheartedness. To my sister, for growing up together and being one of my closest friends.

Lastly, I would like to thank my partner. She has been the person who has taught me the most and has brought me the most joy in these years.

The author would like to thank the Engineering and Physical Sciences Research Council for their support.

Abstract

Reinforcement learning (RL) is a powerful paradigm for training agents to make optimal decisions in sequential environments through interaction and feedback. A central aspect of RL research lies in the development and application of efficient optimization methods that enable agents to learn complex behaviors. This thesis contributes to this area by introducing novel optimization techniques and by analyzing their applications in various RL settings, encompassing cooperative multi-agent systems, policy optimization with general parameterizations, and preference-based learning.

In the context of cooperative multi-agent reinforcement learning, where multiple agents collaborate to achieve a common goal, a significant challenge arises from the exponential increase in complexity with the number of agents. To address this, we introduce a scalable algorithm based on Natural Policy Gradient, which leverages local information exchange between neighboring agents within a defined range. We theoretically demonstrate that, under standard assumptions on spatial decay of correlations, our algorithm converges to the globally optimal policy with a statistical and computational complexity that remains independent of the number of agents.

This thesis further investigates the use of mirror descent as a versatile framework for policy optimization in reinforcement learning. We develop Approximate Mirror Policy Optimization (AMPO) and establish for it the first linear convergence guarantee for a policy-gradient-based method that accommodates general policy parameterizations. Furthermore, we empirically examine the impact of different mirror maps within the Policy Mirror Descent (PMD) and AMPO frameworks, revealing that the commonly used negative entropy is not always the best choice.

Finally, we extend the application of mirror descent to the domain of preference optimization (PO), a crucial technique for aligning agents with human preferences. In particular, we propose a novel family of algorithms called Mirror Preference Optimization (MPO). Through evolutionary strategies, we identify specialized MPO algorithms tailored to specific characteristics of preference datasets, such as mixed-quality or noisy data. Our findings demonstrate that these discovered algorithms outperform existing state-of-the-art PO methods in both simulated robotic tasks and a large language model alignment task, highlighting the effectiveness of our mirror descent-based approach for preference learning.

Contents

1 Introduction

Contents

1.1 Motivation

Many of the most impactful problems that we aim to solve with autonomous agents involve a sequential decision-making process with a long-term goal. Whether it is training a robot to navigate a complex environment, a program to play a video-game, or a system to give personalized recommendations, these agents need to learn how to act optimally in the face of uncertainty. However, for most real-world tasks, the

long-term value of the agent's action is difficult to estimate, and the optimal strategy often requires balancing immediate gains with potential future rewards. Explicitly programming such complex behaviors in an agent is often unfeasible due to the vastness of possible scenarios and the difficulty in anticipating all potential outcomes.

The field of Reinforcement Learning (RL) is designed to tackle this challenge, where agents sequentially interact with an environment—observing its state, taking actions, and receiving rewards—to maximize notions of reward. In particular, RL consists in training an agent by letting it interact with its environment and observing the consequences of its actions, without explicit supervision on what constitutes the correct action in each situation.

RL has been very successful in practice in the last few years, largely due to the integration of deep learning techniques. Deep Reinforcement Learning (DRL) has enabled the development of agents capable of achieving or surpassing human-level performance in complex domains such as games (Silver, Huang, et al. 2016; Silver, Schrittwieser, et al. 2017), finance (Deng et al. 2017), robotics (Kober et al. 2013), autonomous driving (Shalev-Shwartz et al. 2016), and large language model alignment (Guo et al. 2025).

Despite these impressive advances, RL still presents several critical challenges and open questions that warrant further investigation, including issues related to the statistical guarantees of learning algorithms and the behavior of these algorithms in practical applications. This thesis aims to contribute to the field of research that studies optimization algorithms used to train RL agents, with the objective of providing statistical guarantees for their performance or insights on their behavior.

In the first part of this thesis, i.e. Chapter 1, we tackle the cooperative multi-agent setting, where multiple agents collaborate to maximize a collective reward. Given the combinatorial nature of this setting, the number of possible states and actions for the group of agents increases exponentially with the number of agents, leading to a curse of dimensionality that affects both standard convergence guarantees and practical implementations. We show that under standard assumptions on the spatial decay of correlations for the transition dynamics of the environment, it

is possible to design a decentralized algorithm with convergence guarantees that do not incur this curse of dimensionality.

In the second part of this thesis, i.e. Chapters 2 and 3, we investigate the use of the Mirror Descent (MD) (Nemirovski and Yudin 1983) optimization algorithm in policy optimization, which is generally referred to as Policy Mirror Descent (PMD) (Lan 2022; Xiao 2022; Zhan et al. 2023; Y. Li, Lan, and Zhao 2024; Y. Li and Lan 2025). Here, policy optimization refers to the process of finding a policy, i.e. a function that dictates the behavior of the agent, that maximizes the expected cumulative reward of the agent. Our first objective is to generalize the convergence guarantees of the tabular setting, where the policy is expressed by a probability distribution for each state, to the general parameterization setting, where the policy is expressed by a parametrized function that maps a state to a probability distribution. This generalization is of particular interest, as it provides a theoretical justification for parameterizing policies with neural networks. To this end, we introduce Approximate Mirror Policy Optimization (AMPO), the first policy-gradient-based method that accommodates general policy parameterizations and benefits from linear convergence guarantees. Our second objective is to show that different algorithms within the PMD and AMPO classes perform differently in practice, despite having similar convergence guarantees. We demonstrate empirically that standard algorithms within these classes are rarely the best alternatives, and, using evolution strategies, we discover algorithms with superior performance in our benchmarks.

Lastly, we apply MD to preference optimization (PO), where an agent is trained on a preference dataset that contains pairs of task attempts ranked by a judge. We design a new family of algorithms, called Mirror Preference Optimization (MPO), which generalizes many popular algorithms, such as Direct Preference Optimization (DPO) (Rafailov et al. 2024) and Simple Preference Optimization (SimPO) (Meng et al. 2024). To understand the behavior of algorithms within this family, we build a simulated benchmark based on MuJoCo, where task attempts are represented by trajectories and trajectories with higher total reward are preferred. Thanks to the insights obtained on this simulated benchmark, we design a novel

PO algorithm that we test on a large language model (LLM) alignment task, outperforming existing baselines.

The next sections will be dedicated to introducing notation and preliminary notions and will be followed by a summary of the chapters in this thesis.

1.2 Preliminaries

In this section, we provide a detailed explanation of the notation, settings, and frameworks considered in this thesis. We start by introducing the general framework for RL for both single-agent and multi-agent settings. We follow by introducing the mirror descent algorithm and policy gradient methods. Lastly, we present the setting of preference optimization and evolution strategies.

1.2.1 Reinforcement Learning

Single-agent Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$ be a discounted Markov Decision Process (MDP), where $\mathcal{S}$ is a possibly infinite state space, $\mathcal{A}$ is a finite action space, $P(s'|s,a)$ is the transition probability from state s to s' under action a, $r(s,a) \in [0,1]$ is a reward function, γ is a discount factor, and μ is a starting state distribution. The behavior of an agent on an MDP is modeled by a *policy* $\pi \in (\Delta(\mathcal{A}))^{\mathcal{S}}$, where $a \sim \pi(\cdot \mid s)$ is the density of the distribution over actions at state $s \in \mathcal{S}$, and $\Delta(\mathcal{A})$ is the probability simplex over $\mathcal{A}$. For a set of parameters Θ, we denote $\{\pi^\theta : \theta \in \Theta\}$ the class of associated parametrized policies.

Given a policy π, let $V^\pi : \mathcal{S} \to \mathbb{R}$ denote the associated *value function*. Letting s_t and a_t be the current state and action at time t, the value function V^π is defined as the expected discounted cumulative reward with the initial state $s_0 = s$, namely,

$$V^\pi(s) := \mathbb{E}_{a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot|s_t,a_t)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| \pi, s_0 = s \right].$$

Now letting $V^\pi(\mu) := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$, our objective is for the agent to find an optimal policy

$$\pi^\star \in \operatorname{argmax}_{\pi \in (\Delta(\mathcal{A}))^{\mathcal{S}}} V^\pi(\mu). \tag{1.1}$$

Similarly to the value function, for each pair $(s, a) \in \mathcal{S} \times \mathcal{A}$, the state-action value function, or *Q-function*, associated to a policy π is defined as

$$Q^\pi(s,a) := \mathbb{E}_{a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot|s_t,a_t)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid \pi, s_0 = s, a_0 = a \right].$$

The advantage function associated to a policy π for a state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$ is defined as the difference between the Q-function and the value function, that is $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$. Lastly, we define the discounted state visitation distribution by

$$d_\mu^\pi(s) := (1-\gamma)\mathbb{E}_{s_0 \sim \mu} \left[\sum_{t=0}^{\infty} \gamma^t P(s_t = s \mid \pi, s_0) \right], \tag{1.2}$$

where $P(s_t = s \mid \pi, s_0)$ represents the probability of the agent being in state s at time t when following policy π and starting from s_0. The probability $d_\mu^\pi(s)$ represents the time spent on state s when following policy π.

Multi-agent The multi-agent setting can be interpreted as a particular case of the single-agent setting, where the agent consists of a network of smaller agents. Let $\mathcal{G} = (\mathcal{K}, \mathcal{E})$ be an undirected graph describing a network of $|\mathcal{K}| = K$ agents. On this graph, let $d(k, k')$ be the distance between two agents $k, k' \in \mathcal{K}$, that is the length of the shortest path between the two vertices. Let $N_k^r = \{k' \in \mathcal{K} : d(k, k') \leq r\}$ denote the neighborhood of radius r of agent k, with $N_{-k}^r = \mathcal{K} \setminus N_k^r$. Let $\mathcal{S} = \mathcal{S}(1) \times \cdots \times \mathcal{S}(K)$ and $\mathcal{A} = \mathcal{A}(1) \times \cdots \times \mathcal{A}(K)$, where $S(k)$ and $A(k)$ are the state and action spaces associated with agent k. Let $P(s'|s,a) = \prod_{k \in \mathcal{K}} P_k(s'(k)|s,a)$, where P_k is the local transition probability, and $r(s,a) = \frac{1}{K} \sum_{k \in \mathcal{K}} r_k(s(k), a(k))$, where $r_k : \mathcal{S}(k) \times \mathcal{A}(k) \to [0,1]$ is the reward for agent k.

To each agent k is assigned a policy $\pi^k(a(k)|s)$. Given the current global state s, each agent acts independently of the others, that is, $\pi(a|s) = \prod_{k \in \mathcal{K}} \pi^k(a(k)|s)$.

For a policy π and for each agent k, we define the associated localized value,

state-action value, and advantage functions as

$$V_k^\pi(s) = \mathbb{E}_{a_t\sim\pi(\cdot|s_t),s_{t+1}\sim P(\cdot|s_t,a_t)}\left[\sum_{t=0}^{\infty}\gamma^t r_k(s_t(k),a_t(k))\middle|\pi, s_0=s\right],$$
$$Q_k^\pi(s,a) = \mathbb{E}_{a_t\sim\pi(\cdot|s_t),s_{t+1}\sim P(\cdot|s_t,a_t)}\left[\sum_{t=0}^{\infty}\gamma^t r_k(s_t(k),a_t(k))\middle|\pi, s_0=s, a_0=a\right],$$
$$A_k^\pi(s,a) = Q_k^\pi(s,a) - V_k^\pi(s).$$

The respective global functions are defined as the average of the local ones, that is,

$$V^\pi(s) = \frac{1}{K}\sum_{k\in\mathcal{K}} V_k^\pi(s),\ Q^\pi(s,a) = \frac{1}{K}\sum_{k\in\mathcal{K}} Q_k^\pi(s,a),\ A^\pi(s,a) = \frac{1}{K}\sum_{k\in\mathcal{K}} A_k^\pi(s,a).$$

In the following, we will assume to be in the single-agent setting, unless specified.

1.2.2 Mirror descent

Mirror Descent (MD) (Nemirovski and Yudin 1983; Bubeck 2015) is an iterative optimization algorithm used to find a local minimum of a differentiable function, or the global minimum in convex optimization problems. It is a generalization of the gradient descent algorithm (Cauchy et al. 1847) that offers more flexibility by adapting to the geometry of the problem space. The first tools we recall from the MD framework are mirror maps and Bregman divergences (Bregman 1967). Let $\mathcal{Y}\subseteq\mathbb{R}^{|\mathcal{A}|}$ be a convex set. A *mirror map* $h:\mathcal{Y}\to\mathbb{R}$ is a strictly convex, continuously differentiable and essentially smooth function[1] such that $\nabla h(\mathcal{Y}) = \mathbb{R}^{|\mathcal{A}|}$. The convex conjugate of h, denoted by h^*, is given by

$$h^*(x^*) := \sup_{x\in\mathcal{Y}}\langle x^*, x\rangle - h(x), \quad x^*\in\mathbb{R}^{|\mathcal{A}|}.$$

The gradient of the mirror map $\nabla h : \mathcal{Y}\to\mathbb{R}^{|\mathcal{A}|}$ allows to map objects from the primal space $\mathcal{Y}$ to its dual space $\mathbb{R}^{|\mathcal{A}|}$, $x\mapsto\nabla h(x)$, and viceversa for ∇h^*, i.e., $x^*\mapsto\nabla h^*(x^*)$. In particular, from $\nabla h(\mathcal{Y}) = \mathbb{R}^{|\mathcal{A}|}$, we have: for all $(x,x^*)\in\mathcal{Y}\times\mathbb{R}^{|\mathcal{A}|}$,

$$x = \nabla h^*(\nabla h(x)) \quad \text{and} \quad x^* = \nabla h(\nabla h^*(x^*)). \tag{1.3}$$

Furthermore, the mirror map h induces a *Bregman divergence*, defined as

$$\mathcal{D}_h(x,y) := h(x) - h(y) - \langle\nabla h(y), x-y\rangle,$$

[1] h is essentially smooth if $\lim_{x\to\partial\mathcal{Y}}\left\|\nabla h(x)\right\|_2 = +\infty$, where $\partial\mathcal{Y}$ denotes the boundary of $\mathcal{Y}$.

where $\mathcal{D}_h(x, y) \geq 0$ for all $x, y \in \mathcal{Y}$. We can now present the standard MD algorithm. Let $\mathcal{X} \subseteq \mathcal{Y}$ be a convex set and $V : \mathcal{X} \to \mathbb{R}$ be a differentiable function. The MD algorithm can be formalized as the following iterative procedure in order to solve the minimization problem $\min_{x\in\mathcal{X}} V(x)$: for all $t \geq 0$,

$$y^{t+1} = \nabla h(x^t) - \eta_t \nabla V(x)|_{x=x^t}, \tag{1.4}$$

$$x^{t+1} = \text{Proj}^h_{\mathcal{X}}(\nabla h^*(y^{t+1})), \tag{1.5}$$

where η_t is set according to a step-size schedule $(\eta_t)_{t\geq 0}$ and $\text{Proj}^h_{\mathcal{X}}(\cdot)$ is the *Bregman projection*

$$\text{Proj}^h_{\mathcal{X}}(y) := \text{argmin}_{x\in\mathcal{X}} \mathcal{D}_h(x, y). \tag{1.6}$$

Precisely, at time t, $x^t \in \mathcal{X}$ is mapped to the dual space through $\nabla h(\cdot)$, where a gradient step is performed as in (1.4) to obtain y^{t+1}. The next step is to map y^{t+1} back in the primal space using $\nabla h^*(\cdot)$. In case $\nabla h^*(y^{t+1})$ does not belong to $\mathcal{X}$, it is projected as in (1.5).

The MD algorithm in (1.4) and (1.5) can also be formulated with a single expression, as we demonstrate in the following lemma (Bubeck 2015).

Lemma 1.2.1. *Consider the mirror descent update in* (1.4)-(1.5) *for the minimization of a function* $V(\cdot)$. *The mirror descent update can be rewritten as*

$$x^{t+1} \in \operatorname*{argmin}_{x\in\mathcal{X}} \eta_t \langle x, \nabla V(x)|_{x=x^t}\rangle + \mathcal{D}_h(x, x^t). \tag{1.7}$$

Proof. From definition of the Bregman projection step, starting from (1.4) we have

$$\begin{aligned}
x^{t+1} &= \text{Proj}^h_{\mathcal{X}}(\nabla h^*(y^{t+1})) = \operatorname*{argmin}_{x\in\mathcal{X}} \mathcal{D}_h(x, \nabla h^*(y^{t+1})) \\
&\in \operatorname*{argmin}_{x\in\mathcal{X}} \nabla h(x) - \nabla h(\nabla h^*(y^{t+1})) - \langle \nabla h(\nabla h^*(y^{t+1})), x - \nabla h^*(y^{t+1})\rangle \\
&\overset{(1.3)}{\in} \operatorname*{argmin}_{x\in\mathcal{X}} \nabla h(x) - y^{t+1} - \langle y^{t+1}, x - \nabla h^*(y^{t+1})\rangle \\
&\in \operatorname*{argmin}_{x\in\mathcal{X}} \nabla h(x) - \langle x, y^{t+1}\rangle \\
&\overset{(1.4)}{\in} \operatorname*{argmin}_{x\in\mathcal{X}} \nabla h(x) - \langle x, \nabla h(x^t) - \eta_t \nabla V(x)|_{x=x^t}\rangle \\
&\in \operatorname*{argmin}_{x\in\mathcal{X}} \eta_t \langle x, \nabla V(x)|_{x=x^t}\rangle + \nabla h(x) - \nabla h(x^t) - \langle \nabla h(x^t), x - x^t\rangle \\
&\in \operatorname*{argmin}_{x\in\mathcal{X}} \eta_t \langle x, \nabla V(x)|_{x=x^t}\rangle + \mathcal{D}_h(x, x^t),
\end{aligned}$$

where the second and the last lines are both obtained by the definition of the Bregman divergence. □

The one-step update in (1.7) is often taken as the definition of mirror descent (Beck and Teboulle 2003), which provides a proximal view point of mirror descent, i.e., a gradient step in the primal space with a regularization of Bregman divergence.

1.2.3 Policy Gradient Methods

Policy gradient (PG) methods consist in employing differentiable policy parametrizations and in solving the value maximization problem in (1.1) with gradient ascent. Policy parametrizations depend on the environment. In the tabular setting, where the state and action spaces $\mathcal{S}$ and $\mathcal{A}$ are finite, we usually consider softmax policies, where the policy is expressed as:

$$\pi^\theta(a|s) = \frac{\exp(\theta_{s,a})}{\sum_{a' \in \mathcal{A}} \exp(\theta_{s,a'})}, \tag{1.8}$$

for $\theta_{s,a} \in \Theta = \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$. Alternatively, we consider directly parametrized policies, which are defined as

$$\pi^\theta(a|s) = \theta_{s,a}, \tag{1.9}$$

where $\theta \in \Delta(\mathcal{A})^{|\mathcal{S}|}$, i.e., $\theta_{s,a} \geq 0$ and $\sum_{a \in \mathcal{A}} \theta_{s,a} = 1$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$. If the state space $\mathcal{S}$ is continuous, then we consider policies of the form

$$\pi(a|s) = \frac{\exp(f_\theta(s,a))}{\sum_{a' \in \mathcal{A}} \exp(f_\theta(s,a))}, \tag{1.10}$$

where $f_\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a parametrized function. Given a parametrized policy π^θ, the expression of the gradient of the value function $V^\pi(\mu)$ with respect to the policy is given by the policy gradient theorem (Sutton et al. 1999):

$$\nabla_\theta V^\pi(\mu) = \frac{1}{1-\gamma} \mathbb{E}_{s \sim d_\mu^{\pi^\theta}, a \sim \pi^\theta(\cdot|s)} [Q^\pi(s,a) \nabla_\theta \log \pi^\theta(a|s)]. \tag{1.11}$$

If the parametrization explicitly constrains π^t to be in the simplex $\Delta(\mathcal{A})^{|\mathcal{S}|}$, as in softmax policies, the following also holds:

$$\nabla_\theta V^\pi(\mu) = \frac{1}{1-\gamma} \mathbb{E}_{s \sim d_\mu^{\pi^\theta}, a \sim \pi^\theta(\cdot|s)} [A^\pi(s,a) \nabla_\theta \log \pi^\theta(a|s)]. \tag{1.12}$$

This expression of the gradient does not hold, for example, for the direct parametrization, where $\sum_{a\in\mathcal{A}} \nabla_\theta \pi^\theta(a|s) = 0$ is not guaranteed for all $s \in \mathcal{S}$ and $\theta \in \Theta$. Using (1.11), the vanilla PG algorithm can be written as

$$\theta^{t+1} = \theta^t + \frac{\eta^t}{1-\gamma}\mathbb{E}_{s\sim d_\mu^{\pi^\theta}, a\sim\pi^\theta(\cdot|s)}[Q^\pi(s,a)\nabla_\theta \log \pi^\theta(a|s)],$$

where θ^t is the set of parameters at iteration t. Several variations of vanilla PG have been proposed in the literature. We will focus on natural policy gradient (NPG) (S. M. Kakade 2001; A. Agarwal et al. 2021) and on policy mirror descent (PMD). NPG consist in an adaptation of the natural gradient descent algorithm (Amari 1998) to policy optimization, whereby the gradient is pre-multiplied by the inverse of the Fisher information matrix before being added to the current parameter vector θ^t. Let the Fisher information matrix induced by π^θ be defined as

$$F_\mu(\theta) = \mathbb{E}_{s\sim d_\mu^\pi, a\sim\pi^\theta(\cdot|s)}\left[\nabla_\theta \log \pi^\theta(a|s)(\nabla_\theta \log \pi^\theta(a|s))^\top\right].$$

The NPG update, with step-size η, is defined as

$$\theta^{t+1} = \theta^t + \eta^t F_\mu(\theta^t)^{-1}\nabla_\theta V^{\theta^t}(\mu), \tag{1.13}$$

where $F_\mu(\theta^t)^{-1}$ is the Moore-Penrose pseudo-inverse of the Fisher information matrix. S. M. Kakade 2001 shows that, for the softmax parametrization in the tabular setting, the update in (1.13) is equivalent to

$$\theta_{s,a}^{t+1} = \theta_{s,a}^t + \eta^t \frac{A^t(s,a)}{(1-\gamma)}. \tag{1.14}$$

In the continuous state space setting, with policies defined as in (1.10), the update in (1.13) is equivalent to solving the problem

$$w_\star \in \operatorname*{argmin}_w \mathbb{E}_{s\sim d_\mu^{\pi^\theta}, a\sim\pi^\theta(\cdot|s)}\left[\left(A^{\pi^\theta}(s,a) - w\cdot\nabla_\theta \log \pi^\theta(\cdot|s)\right)^2\right] \tag{1.15}$$

and then performing the update

$$\theta^{t+1} = \theta^t + \frac{\eta^t}{1-\gamma} w^\star. \tag{1.16}$$

Policy mirror descent is the adaptation of the mirror descent update in (1.7) to policy optimization. In the setting of direct policy parametrization, its update is defined as

$$\pi^{t+1} \in \operatorname*{argmax}_{\pi} \mathbb{E}_{s\sim d_\mu^{\pi^t}} \left[\eta^t \langle Q_s^t, \pi_s\rangle + D_h(\pi_s, \pi_s^t)\right], \tag{1.17}$$

where we used the expression for the gradient in (1.11) and where we defined $D_h(\pi, \pi^t) = \mathbb{E}_{s\sim d_\mu^{\pi^t}} D_h(\pi_s, \pi_s^t)$. When the mirror map is the negative entropy, that is $h(\pi_s) = \sum_{a\in\mathcal{A}} \pi(a|s) \log \pi(a|s)$, and the Bregman divergence is the KL divergence, the solution to (1.17) becomes

$$\pi^{t+1}(a|s) = \pi^t(a|s) \frac{\exp(\eta^t Q^t(s,a)/(1-\gamma))}{Z_t(s)}$$

where $Z_t(s) = \sum_{a\in\mathcal{A}} \pi^t(a|s)\exp(\eta Q^t(s,a)/(1-\gamma))$ is a normalizing factor and which is equivalent to the NPG update in (1.14).

1.2.4 Preference optimization

Learning from human preferences (Christiano et al. 2017) is a paradigm which enables the alignment of machine learning systems to relative human preferences, without requiring access to absolute rewards. While the framework was developed for robotic and games applications with experiments on MuJoCo simulations and Atari (Akrour et al. 2012; Biyik and Sadigh 2018; Ibarz et al. 2018), this paradigm has been successfully applied to Large Language Models (Team et al. 2023; Achiam et al. 2023). In particular, fine-tuning pre-trained LLMs with human preferences has become a popular strategy to adapt them to specific tasks and to improve their safety and helpfulness. Within this framework, Reinforcement Learning from Human Feedback (RLHF) is one of the most popular methods. It consists in learning a reward function using a preference dataset and then optimizing the estimated reward using Reinforcement Learning methods.

Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$ be an MDP and $\mathcal{D} = \{(s_0^i, \tau_w^i, \tau_l^i)_{i=1}^N\}$ be a preference dataset, where each tuple (s_0, τ_w, τ_l) consists of a starting state s_0 and two trajectories with starting state s_0. Each pair of trajectories is ranked by a judge, who determines a chosen trajectory τ_w ("win") and a rejected trajectory τ_l ("lose"),

based on the cumulative rewards $r(\tau_w)$ and $r(\tau_l)$. Most settings assume the judge ranks trajectories according to the Bradley-Terry model (Bradley and Terry 1952), whereby the probability of choosing τ_w over τ_l is defined as

$$\mathbb{P}(\tau_w \succ \tau_l) = \frac{\exp(r(\tau_w)/\eta)}{\exp(r(\tau_w)/\eta) + \exp(r(\tau_l)/\eta)} = \sigma((r(\tau_w) - r(\tau_l))/\eta), \tag{1.18}$$

where σ is the sigmoid function and η is a temperature parameter. In this thesis, we consider an offline training setting, where the agent aims to solve the optimization problem in (1.1) but only has access to the the dataset $\mathcal{D}$ and cannot collect further data. We also assume the agent does not have access to either the transition probability P, the reward function r, or the MDP $\mathcal{M}$.

There are several algorithms in the literature to optimize the objective in (1.1) using a preference dataset $\mathcal{D}$. We describe here the main training pipeline, consisting of supervised fine-tuning (SFT) and preference optimization (PO).

SFT SFT is an initial alignment phase, where the policy π_0 is trained to imitate high-quality demonstration data. The starting policy π_0 is updated to minimize the cross-entropy loss $\ell(\pi, (s_0, \tau_w, \tau_l)) = -\log(\pi(\tau_w))$. We call *reference policy* π_{ref} the policy obtained at the end of this procedure.

PO Preference Optimization consists in solving a maximum likelihood estimation problem to obtain the reward estimate, which is then used to find an optimal policy. That is, we aim to solve

$$\widehat{r} \in \operatorname*{argmax}_{r_\theta} \mathbb{E}_{(s_0,\tau_w,\tau_l)} \log \sigma((r_\theta(\tau_w) - r_\theta(\tau_l))/\eta), \tag{1.19}$$

for a parametrized reward class $\{r_\theta : \theta \in \Theta\}$. The reward estimate is then used in the policy optimization problem

$$\pi^\star \in \operatorname*{argmax}_{\pi} \mathbb{E}_{s_0\sim\mathcal{D},\tau\sim(\pi,P)} \left[\sum_{t=0}^{T-1} \mathbb{E}_{a\sim\pi(\cdot|s_t)} \widehat{r}(s_t, a) - \beta D_{\texttt{KL}}(\pi(\cdot|\tau), \pi_{\text{ref}}(\cdot|\tau)) \right], \tag{1.20}$$

where $D_{\texttt{KL}}$ represents the KL-divergence and is introduced to prevent the policy from moving too far away from the dataset distribution. The problem in (1.20) is then solved with a RL algorithm, typically Proximal Policy Optimization

(PPO) (Schulman et al. 2017). Rafailov et al. 2024 have proposed an alternative method, Direct Preference Optimization (DPO), which merges (1.19) and (1.20) by using the agent itself to implicitly represent the reward model. It consists in optimizing the objective

$$\pi^\star \in \operatorname*{argmax}_\pi \mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}} \left[\log \sigma \left(\beta \left(\log \frac{\pi(\tau_w)}{\pi_{\text{ref}}(\tau_w)} - \log \frac{\pi(\tau_l)}{\pi_{\text{ref}}(\tau_l)}\right)\right)\right], \tag{1.21}$$

which is obtained by plugging the theoretical solution of (1.20) in the maximum likelihood problem in (1.19).

1.2.5 Evolution Strategies

Evolution Strategies (ES) (Rechenberg 1973; Schwefel 1977) are a family of optimization algorithms that operate like a guided trial-and-error process, modeled on the principles of biological evolution. At each stage, a collection of proposed solutions is tested and ranked based on how well they solve a given problem. The top performers are then used to generate a new set of solutions for the next stage, often by introducing small, random changes (mutation) and mixing elements of the successful solutions. This cycle of variation and selection is repeated to progressively find an optimal solution. The specific ways in which solutions are managed, varied, and combined are what differentiate the various algorithms within the Evolution Strategies class.

In this thesis, we consider the OpenAI-ES (Salimans et al. 2017) algorithm, which has been widely used to meta-learn objectives (Lu et al. 2022; Jackson et al. 2024), as it obtains an unbiased estimate of the gradient. Let $F : \Theta \to \mathbb{R}$ be a function we want to optimize. OpenAI-ES consists in estimating the gradient $\nabla_\theta F(\theta)$, for $\theta \in \Theta$, using:

$$\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I_d)} \left[\frac{\epsilon}{2\sigma}(\widehat{F}(\theta+\sigma\epsilon) - \widehat{F}(\theta-\sigma\epsilon))\right],$$

where $\mathcal{N}(0, I_d)$ is the multivariate normal distribution, d is the number of parameters, $\widehat{F}$ is an estimate of F, and $\sigma > 0$ is a hyperparameter that regulates the variance of the perturbations.

1.3 Contributions

In this section, we will summarize the contributions of each chapter in this thesis.

1.3.1 Dimension-Free Rates for Natural Policy Gradient in Multi-Agent Reinforcement Learning

In Chapter 2, we consider the issue of applying NPG in the multi-agent setting with continuous state spaces. The straightforward application of the update (1.16) incurs prohibitive costs. This is due to the iteration complexity given by standard bounds (A. Agarwal et al. 2021) getting worse by a factor K and to the increased sample and computational complexity of solving problem (1.15) in higher dimension. Additionally, solving (1.15) and (1.16) requires each agent to communicate with every other agent in the network at each iteration, which is rarely viable in real-world applications.

We design a decentralized version of NPG that does not present these issues. To do so, we assume that the network of agents presents some form of decay of correlations, that is we assume that the further two agents are the less they influence each other. In particular, we assume that a version of the Dobrushin condition (Dobrusin 1970; Georgii 2011) holds for the transition dynamics of the network of agents. Let $TV(\mu, \nu) = \sup_{A \in \mathcal{F}} |\mu(A) - \nu(A)|$ be the total variation distance between the probability distributions μ and ν defined on the σ-algebra $\mathcal{F}$.

Assumption 1.3.1. (Spatial Decay of Correlation for the Dynamics) Let $C \in \mathbb{R}^{K \times K}$ be defined as

$$C_{ij} = \sup_{s_j, s_{-j}, a_j, a_{-j}, s'_j, a'_j} TV(P_i(\cdot|s_j, s_{-j}, a_j, a_{-j}), P_i(\cdot|s'_j, s_{-j}, a'_j, a_{-j})),$$

that is the influence that a perturbation of the state and action of agent j has on the transition probability of agent i. Assume that there exists $\beta \geq 0$ such that

$$\max_{k \in \mathcal{K}} \sum_{j \in \mathcal{K}} e^{\beta d(k,j)} C_{kj} \leq \rho,$$

with $\rho < 1/\gamma$, where γ is the discount factor of the MDP.

Our version of decentralized NPG limits the communication range to agents that are at most at distance r. For each agent k, we impose the policy to be local, i.e. $\pi^{\theta_k}(a_k|s) = \pi^{\theta_k}(a_k|s_{N_k^r})$, and define the localized advantage function, localized value function, and localized Q-function, as follows:

$$\begin{aligned}
\widetilde{A}_k^\pi(s_{N_k^r}, a_{N_k^r}) &= \widetilde{Q}_k^\pi(s_{N_k^r}, a_{N_k^r}) - \widetilde{V}_k^\pi(s_{N_k^r}), \\
\widetilde{V}_k^\pi(s_{N_k^r}) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_k(s_k(t), a_k(t)) \Big| \pi, s_{N_k^r}(0) = s_{N_k^r}\right], \\
\widetilde{Q}_k^\pi(s_{N_k^r}, a_{N_k^r}) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_k(s_k(t), a_k(t)) \Big| \pi, s_{N_k^r}(0) = s_{N_k^r}, a_{N_k^r}(0) = a_{N_k^r}\right].
\end{aligned}$$

For each agent k, define the loss function

$$\widetilde{L}_k^r(w, \theta, \nu) = \mathbb{E}_{s,a\sim\nu}\left[\left(\widetilde{A}_k^{\pi_\theta}(s_{N_k^r}, a_{N_k^r}) - \nabla_{\theta_k} \log \pi^{\theta_k}(a_k|s_{N_k^r}) \cdot w\right)^2\right]. \tag{1.22}$$

We can now express the decentralized NPG algorithm. At each step, each agent k aims to solve the minimization problem

$$w_k^{(t)} \in \underset{\|w\|_2 \le W}{\operatorname{argmin}} \widetilde{L}_k^r(w, \theta^{(t)}, d^{(t)})$$

and to update its parameters

$$\theta_k^{(t+1)} = \theta_k^{(t)} + \frac{\eta}{1-\gamma} w_k^{(t)}.$$

For this algorithm, we obtain the same convergence guarantee as the ones obtained by A. Agarwal et al. 2021, plus a term that goes exponentially fast to 0 for an increasing range r.

Qu and N. Li 2019; Qu, Wierman, et al. n.d.; Qu, Lin, et al. 2020, and Lin et al. 2020 are concurrent works that use decay of correlation assumptions to provably avoid the curse of dimensionality in multi-agent RL. Our contributions with respect to these works are described in Appendix A of Chapter 2. In summary, we consider a more general version of the decay of correlation assumption and provide convergence bound w.r.t. the *optimal* policy, while their convergence bounds are w.r.t. a stationary policy.

1.3.2 A Novel Framework for Policy Mirror Descent with General Parameterization and Linear Convergence

In Chapter 3, we introduce Approximate Mirror Policy Optimization (AMPO), a novel framework designed to incorporate general parameterization into PMD in a theoretically sound manner. AMPO is an MD-based method that recovers PMD in different settings, such as tabular MDPs, is capable of generating new algorithms by varying the mirror map, and is amenable to theoretical analysis for any parameterization class. We can formulate AMPO as an approximation of the two-step formulation of MD in (1.4) and (1.5), in the context of policy optimization with direct parametrization. That is, we want to approximate

$$\begin{aligned} f^{t+1}(s,\cdot) &= \nabla h(\pi^t(s,\cdot)) + \eta_t(1-\gamma)\nabla_{\pi(s,\cdot)} V^t(\mu)/d_\mu^t(s) \\ &\overset{(1.11)}{=} \nabla h(\pi^t(s,\cdot)) + \eta_t Q^t(s,\cdot),^2 \end{aligned} \tag{1.23}$$

$$\pi^{t+1}(s,\cdot) = \mathrm{Proj}^h_{\Delta(\mathcal{A})}(\nabla h^*(\eta_t f^{t+1}(s,\cdot))), \tag{1.24}$$

for $f \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$. Given a parameterized function class $\mathcal{F}^\Theta = \{f^\theta : \mathcal{S}\times\mathcal{A} \to \mathbb{R}, \theta \in \Theta\}$, AMPO replaces (1.23) with

$$\theta^{t+1} \in \operatorname*{argmin}_{\theta\in\Theta} \mathbb{E}_{(s,a)\sim v^t}\Big[f^\theta(s,a) - Q^t(s,a) - \eta_t^{-1}\nabla h(\pi^t(s,\cdot))_a\Big]^2 \tag{1.25}$$

and obtains the policy at step $t+1$ as in (1.24). In Chapter 3, we give a detailed discussion on how to solve the projection step in (1.24) efficiently. In particular, when the mirror map is the negative entropy, the projection step becomes a softmax operator.

We provide an analysis of AMPO and establish theoretical guarantees that hold for any parameterization class and any mirror map. More specifically, we show that our algorithm enjoys quasi-monotonic improvements, sublinear convergence when the step-size is non-decreasing, and linear convergence when the step-size is geometrically increasing. To the best of our knowledge, AMPO is the first policy-gradient-based method with linear convergence that can accommodate any parameterization class.

[2]The update is (1.4) up to a scaling $(1-\gamma)/d_\mu^t(s)$ of η_t.

1.3.3 Learning mirror maps in policy mirror descent

In Chapter 4, we investigate how the mirror map influences the performance of PMD algorithms. Our research question stems from the fact that most guarantees for PMD algorithms do not explicitly depend on the mirror map. We report, as an example, a slight modification of the results by Xiao 2022 on the quasi-monotonic updates and the convergence to the optimal policy of PMD for direct policy parametrization.

Theorem 1.3.2 (Xiao 2022)**.** *Following update* (1.17)*, we have that, for all* $t \geq 0$

$$V^{t+1}(\mu) - V^t(\mu) \geq -\frac{1}{1-\gamma} \max_{s\in\mathcal{S}} \left\| \widehat{Q}^t(s,\cdot) - Q^t(s,\cdot) \right\|_\infty \max_{s\in\mathcal{S}} \left\| \pi^{t+1}(\cdot|s) - \pi^t(\cdot|s) \right\|_1, \tag{1.26}$$

where $\|\cdot\|_\infty$ *and* $\|\cdot\|_1$ *represent the* ℓ_∞ *and the* ℓ_1 *norms, respectively, and* $\widehat{Q}$ *is an estimate of the true Q-function. Additionally, at each iteration* $T > 0$*, we have*

$$\begin{aligned} V^\star(\mu) - \sum_{t<T} \frac{\mathbb{E}[V^t(\mu)]}{T} \leq & \frac{1}{T} \underbrace{\left(\frac{\mathbb{E}_{s\sim d_\mu^\star}[\mathcal{D}_h(\pi^\star(\cdot|s), \pi^0(\cdot|s))]}{\eta_t(1-\gamma)} + \frac{1}{(1-\gamma)^2} \right)}_{\textit{convergence rate}} \\ & + \underbrace{4 \frac{\max_{t<T, s\in\mathcal{S}} \left\| \widehat{Q}^t(s,\cdot) - Q^t(s,\cdot) \right\|_\infty}{(1-\gamma)^2}}_{\textit{error floor}}. \end{aligned} \tag{1.27}$$

Given that by setting $\eta_0 = \mathbb{E}_{s\sim d_\mu^\star}[\mathcal{D}_h(\pi_s^\star, \pi_s^0)](1-\gamma)$ we obtain the convergence rate $2(T(1-\gamma)^2)^{-1}$, and that the error floor has no explicit dependence on the mirror map, Equation (1.27) suggests that the mirror map has a mild influence on the performance of PMD. On the other hand, Equation (1.26) suggests that mirror maps that prevent large updates of the policy cause the PMD algorithm to be less prone to performance degradation, as the lower bound is close to 0 when the policy update distance $\max_{s\in\mathcal{S}} \|\pi_s^{t+1} - \pi_s^t\|_1$ is small.

Using ES, we search for the mirror maps that lead to the highest performance of PMD in a variety of settings, obtaining two main insights. The first is that the mirror map plays a crucial role in determining both the convergence speed and the lowest attainable error floor. In particular, we show that using a mirror map that limits large updates throughout training leads to improved performance, as indicated by (1.26). On the other hand, we show that the error floor in (1.27) is

not a good indicator of performance, since the best performing algorithm in our simulations also had the highest theoretical error floor. The second insight is that, while most works focus on the negative entropy mirror map, it is rarely the optimal choice, as we find mirror maps that consistently outperform it.

1.3.4 Meta-Learning Objectives for Preference Optimization

In Chapter 5, we provide a comprehensive analysis of PO algorithms, examining their behavior on automatically generated preference datasets. We perform this analysis in MuJoCo environments and datasets, where the underlying ground-truth reward structure is well defined and offers a clear performance metric to compare agents. In particular, we design a benchmark where a pre-trained agent has to adhere to a new stylistic constraint using a preference dataset, emulating the typical conditions of LLM fine-tuning. On this benchmark, we test eight existing PO algorithms using automatically generated preference datasets with varying levels of data quality, noise levels and initial policy. We see that most existing algorithms struggle when dealing with noise and mixed-quality data.

Moreover, we introduce a framework for finding PO algorithms, which we test on our benchmark. Specifically, we define Mirror Preference Optimization (MPO), a class of PO algorithms based on mirror descent, which recovers DPO when using the negative entropy mirror map. We search this class using ES, optimizing for the final performance of the trained policy, as measured by the ground truth reward. For each setting we consider, we discover an algorithm that significantly outperforms all baselines. Analyzing the discovered algorithms, we find that the main difference between them and the baselines is that they keep optimizing the policy of the agent well after the probability of generating the chosen trajectory has surpassed the probability of generating the rejected one. Lastly, we hand-craft an algorithm within MPO using the takeaways from our analysis on the MuJoCo setting. We test this algorithm on an LLM alignment task, where we significantly outperform baselines.

2

Dimension-Free Rates for Natural Policy Gradient in Multi-Agent Reinforcement Learning

Dimension-Free Rates for Natural Policy Gradient in Multi-Agent Reinforcement Learning

Carlo Alfano
Department of Statistics
University of Oxford

Patrick Rebeschini
Department of Statistics
University of Oxford

Abstract

Cooperative multi-agent reinforcement learning is a decentralized paradigm in sequential decision making where agents distributed over a network iteratively collaborate with neighbors to maximize global (network-wide) notions of rewards. Exact computations typically involve a complexity that scales exponentially with the number of agents. To address this curse of dimensionality, we design a scalable algorithm based on the Natural Policy Gradient framework that uses local information and only requires agents to communicate with neighbors within a certain range. Under standard assumptions on the spatial decay of correlations for the transition dynamics of the underlying Markov process and the localized learning policy, we show that our algorithm converges to the globally optimal policy with a dimension-free statistical and computational complexity, incurring a localization error that does not depend on the number of agents and converges to zero exponentially fast as a function of the range of communication.

1 Introduction

Sequential decision-making is a prominent setting in modern statistical theories and applications, where agents sequentially interact with an environment—observing its state, taking actions, and receiving rewards—to maximize notions of reward. Reinforcement learning is the setting where agents do not have complete knowledge of the environment dynamics, and it has received increased attention due to its recent successes on a variety of domains, e.g. games [Silver et al., 2016, 2017] and autonomous driving [Shalev-Shwartz et al., 2016].

Modern applications typically involve high-dimensional state and action spaces, and classical algorithms often lead to a computational complexity that scales exponentially with the number of degrees of freedom in the model. Understanding which structures can be used to design approximate methods that can overcome this curse of dimensionality while retaining near-optimal statistical guarantees is a matter of paramount importance.

A class of algorithms that have proven successful in the face of high-dimensional models is that of Natural Policy Gradient (NPG) methods [Amari, 1998, Kakade, 2002, Peters and Schaal, 2008, Bhatnagar et al., 2009]. It has recently been shown [Agarwal et al., 2020] that NPG converges to an optimal policy with an iteration complexity that scales only logarithmically with the cardinality of the action space and with no explicit dependence on the cardinality of the state space.

Despite the favorable iteration complexity of NPG, NPG still faces the curse of dimensionality in applications where the computational cost per iteration scales exponentially with the number of degrees of freedom. This is the case in the setting of multi-agent reinforcement learning (MARL), for instance, where agents distributed over a network iteratively interact with each other to maximize global notions of reward. In this setting, the computational complexity of NPG—when applied to the entire network of agents—scales exponentially with the dimension of the model, which corresponds to the number of agents (see Section 3).

Along with the curse of dimensionality, NPG also faces scalability and implementability issues within the MARL framework. Applying NPG to the entire network of agents requires global communication, i.e. it requires each agent to be able to communicate with every other agent in the net-

work, at every time step. This requirement is unrealistic in many multi-agent applications of interest, where the network topology is typically sparse, often grid-like, and where agents are only allowed to perform computation and communication in a decentralized manner, interacting only with neighboring agents within a certain range. These computational and communicational constraints arise, for instance, in the case of sensor networks, e.g. [Rabbat and Nowak, 2004, Nedic and Ozdaglar, 2009], and in the case of intelligent transportation systems, e.g. [Adler and Blue, 2002].

Over the past decades, various approaches have been proposed to address the curse of dimensionality in high-dimensional reinforcement learning models and, before that, in high-dimensional dynamical programming models, where exact knowledge of the probabilistic structure describing the environment is assumed. A popular approach involves designing algorithms that can exploit notions of *locality*, which encodes the assumption that, in some regimes, information can dissipate when it propagates through the network so that global computation and communication are not required to meet a prescribed level of error accuracy. Exploiting locality prompted the use of ad-hoc approximate factorization and truncation techniques, such as expressing the value function as a linear combination of basis functions that only depend on a small subset of local variables [Koller and Parr, 1999, Guestrin et al., 2001b, Koller and Parr, 2013, Yang and Wang, 2019, Jin et al., 2020]. These ideas have been applied to the MARL setting [Guestrin et al., 2001a, 2002, Sunehag et al., 2017, Rashid et al., 2018, Zhang et al., 2018a,b, Zhang and Zavlanos, 2019] and have proven successful in experiments, but lack theoretical guarantees or non-asymptotic analysis. A recent line of work has formally considered spatial decay of correlation assumptions for nearest-neighbors dynamics and designed decentralized algorithms based on policy gradient and actor-critic methods [Qu and Li, 2019, Qu et al., 2020a, Lin et al., 2020, Qu et al., 2020b], establishing non-asymptotic convergence guarantees towards a *stationary* point, but not towards an *optimal* policy.[1] An application of the same principles to the setting of *mean-field* MARL [Yang et al., 2018] can be found in [Gu et al., 2021], where the authors show that a neural network based version of the actor-critic algorithm can achieve global convergence. In this setting, however, agents are considered to be indistinguishable and the transition scheme of an agent is only affected by the mean effect from its neighbors.

In this paper, we design a decentralized algorithm for the MARL setting based on the NPG framework that only uses local computations and communication for neighbors of agents within a certain range. We show that our algorithm can provably exploit spatial decay of correlation properties to overcome the curse of dimensionality, establishing non-asymptotic convergence guarantees to a globally *optimal* policy. In particular, we consider a general formulation of the decay of correlation assumption from statistical mechanics and probability theory [Dobrusin, 1970, Föllmer, 1982, Georgii, 2011], whereby agents have an influence on each other that decays exponentially with their distance on the network. This type of assumption has been previously considered in the learning literature, e.g. in [Mitliagkas and Mackey, 2017, Dagan et al., 2019, Balle et al., 2019, Prasad et al., 2020, Diakonikolas et al., 2021], and also in the MARL setting, c.f. discussion in the previous paragraph. Under this assumption, we derive convergence bounds that are the same as those established for (centralized) NPG in [Agarwal et al., 2020], worsened only by a localization error that decreases exponentially with the radius of the communication range. A key feature of our bounds is that they are *dimension-free*, as they do not depend on the number of agents, and depend only logarithmically on the cardinality of the action space of *individual* agents and do not explicitly depend on the state space of individual agents. The localization radius controls the trade-off between statistical accuracy and computational complexity, as the overall computational cost of our algorithm scales only with respect to the number of agents within the local communication radius, and not with the total number of agents in the network.

Our contribution fits into the more general literature that has shown how spatial decay of correlations can be used to establish dimension-free results in a variety of settings, such as [Gamarnik, 2013, Gamarnik et al., 2014], mixing times in spin systems [Hayes, 2006, Dyer et al., 2006], particle filtering [Rebeschini and Van Handel, 2015], epidemics [Mei et al., 2017], social networks [Chakrabarti et al., 2008], communication networks [Zocca, 2019], queuing networks [Papadimitriou and Tsitsiklis, 1994], and smart transportation [Zhang and Pavone, 2016].

The paper organization is as follows. In Section 2 we describe the MARL framework and we discuss the model assumptions we work with. In Section 3 we describe NPG as presented in [Agarwal et al.,

[1] Appendix A gives a complete comparison of our results against previous findings that exploit the same type of decay of correlation assumption.

2020] and discuss its limitations when applied to MARL. In Section 4 we design a decentralized version of MARL and state our main results. The Appendix contains all the proofs of our statements and elaborates on the model assumptions.

2 Setting

Let $\mathcal{G} = (\mathcal{K}, \mathcal{E})$ be an undirected graph describing a network of $|\mathcal{K}| = K$ agents. On this graph, the distance $d(k, k')$ between two agents $k, k' \in \mathcal{K}$ is defined as the length of the shortest path between the two vertices. Let $N_k^r = \{k' \in \mathcal{K} : d(k, k') \leq r\}$ denote the neighborhood of radius r of agent k, with $N_k = N_k^1$ and $N_{-k}^r = \mathcal{K} \setminus N_k^r$. Let S_k and A_k be the state and action spaces associated with agent k. We consider a Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$: $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_K$ and $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_K$ are, respectively, the global state space and the global action space; $\forall s, s' \in \mathcal{S}, a \in \mathcal{A}$, $P_k(s_k'|s, a)$ is the local transition probability, that is the probability that agent k transitions to state s_k' when the global state and action is (s, a), and $P(s'|s, a) = \prod_{k \in \mathcal{K}} P_k(s_k'|s, a)$ is the global transition probability; $r(s, a) = \frac{1}{K} \sum_{k \in \mathcal{K}} r_k(s_k, a_k)$ is the global (network-wide) reward function that we wish to maximize, where $r_k : \mathcal{S}_k \times \mathcal{A}_k \to [0, 1]$ is the reward for agent k; γ is the discount factor and μ is the starting state distribution. At time t denote the current state and action by $s(t)$ and $a(t)$.

To each agent k is assigned a local differentiable policy parameterized by $\theta_k \in \Theta_k$,

$$\pi_{\theta_k}(a_k|s) = \frac{e^{f_{\theta_k}(s,a_k)}}{\sum_{a' \in \mathcal{A}_k} e^{f_{\theta_k}(s,a')}},$$

which depends on the current global state s. Given the current global state s, each agent acts independently of the others. Denote $\theta = (\theta_1, \ldots, \theta_K)$ and $\Theta = \Theta_1 \times \cdots \times \Theta_K$, then $\pi_\theta(a|s) = \prod_{k \in \mathcal{K}} \pi_{\theta_k}(a_k|s)$.

For a policy π and for each agent k, let $V_k^\pi : \mathcal{S} \to \mathbb{R}$ be the respective value function, which is defined as the expected discounted cumulative reward with starting state $s(0) = s$, namely,

$$V_k^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_k(s_k(t), a_k(t)) \middle| \pi, s(0) = s\right],$$

where $a(t) \sim \pi(\cdot|s(t))$ and $s(t+1) \sim P(\cdot|s(t), a(t))$. Let V^π be the global value function, defined as $V^\pi(s) = \frac{1}{K} \sum_{k \in \mathcal{K}} V_k^\pi(s)$, and $V^\pi(\mu)$ be the expected global value function when the initial state distribution is μ, i.e. $V^\pi(\mu) = \mathbb{E}_{s \sim \mu} V^\pi(s)$. Our objective is to find an optimal policy $\pi^\star \in \operatorname{argmax}_\pi \mathbb{E}_{s \sim \mu} V^\pi(s)$.

For a policy π and for each agent k, let $Q_k^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be the respective Q-function, which is defined as the expected discounted cumulative reward with starting state $s(0) = s$ and starting action $a(0) = a$, namely,

$$Q_k^\pi(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_k(s_k(t), a_k(t)) \middle| \pi, s(0) = s, a(0) = a\right],$$

where $a(t) \sim \pi(\cdot|s(t))$ and $s(t+1) \sim P(\cdot|s(t), a(t))$. Let Q^π be the global value function, defined as $Q^\pi(s, a) = \frac{1}{K} \sum_{k \in \mathcal{K}} Q_k^\pi(s, a)$.

Let $A_k^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be the advantage function for policy π and agent k, representing the advantage of taking the action a at step 0 and then following policy π, with respect to following policy π from the start, and defined as

$$A_k^\pi(s, a) = Q_k^\pi(s, a) - V_k^\pi(s).$$

Let $A^\pi(s, a) = \frac{1}{K} \sum_{k \in \mathcal{K}} A_k^\pi(s, a)$ be the global advantage function.

We define the discounted state visitation distribution [Sutton et al., 1999],

$$d_\rho^\pi(s) = (1 - \gamma) \mathbb{E}_{s(0) \sim \rho} \sum_{t=0}^{\infty} \gamma^t P(s(t) = s|\pi, s(0)),$$

and the discounted state-action visitation distribution,

$$d_\nu^\pi(s,a) = (1-\gamma)\mathbb{E}_{s(0),a(0)\sim\nu}\sum_{t=0}^{\infty}\gamma^t P(s(t)=s, a(t)=a|\pi, s(0), a(0)),$$

where the trajectory $(s(t), a(t))_{t\geq 0}$ is generated by the MDP following policy π. Lastly, a function $f:\Theta\to\mathbb{R}$ is said to be a δ-smooth function of θ if, $\forall\theta,\theta'\in\Theta$,

$$\left\|\nabla f(\theta) - \nabla f(\theta')\right\|_2 \leq \delta\left\|\theta-\theta'\right\|_2.$$

2.1 Natural Policy Gradient

Natural policy gradient (NPG) is a fundamental algorithm in RL, which modifies the vanilla policy gradient algorithm by pre-multiplying the gradient of the value function with the Fisher information matrix induced by the policy. Let π_θ be a differentiable policy and define the Fisher information matrix induced by π_θ as

$$F_\mu(\theta) = \mathbb{E}_{s\sim d_\mu^\pi}\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\nabla_\theta\log\pi_\theta(a|s)(\nabla_\theta\log\pi_\theta(a|s))^\top\right].$$

Given a step-size η, NPG consists in iteratively updating the policy π_θ as

$$\theta^{(t+1)} = \theta^{(t)} + \eta F_\mu(\theta^{(t)})^{-1}\nabla_\theta V^{\theta^{(t)}}(\mu), \tag{1}$$

where $\theta^{(t)}$ is the set of parameters at iteration t, $\nabla_\theta V^\theta(\mu)$ is the gradient of the value function with respect to the policy parameters, and $F_\mu(\theta^{(t)})^{-1}$ is the Moore-Penrose pseudo-inverse of the Fisher information matrix. As shown by Agarwal et al. [2020], the update in (1) is equivalent to solving

$$w_\star \in \operatorname*{argmin}_w \mathbb{E}_{s\sim d_\mu^{\pi_\theta},a\sim\pi_\theta(\cdot|s)}\left[\left(A^{\pi_\theta}(s,a) - w\cdot\nabla_\theta\log\pi_\theta(\cdot|s)\right)^2\right] \tag{2}$$

and then performing the update

$$\theta^{(t+1)} = \theta^{(t)} + \frac{\eta}{1-\gamma}w^\star. \tag{3}$$

Agarwal et al. [2020] provide convergence guarantees for NPG in the general function approximation setting, which we summarize here. Define

$$L(w,\theta,\nu) := \mathbb{E}_{s,a\sim\nu}\left[\left(A^{\pi_\theta}(s,a) - w\cdot\nabla_\theta\log\pi_\theta(a|s)\right)^2\right].$$

Assume that $\log\pi_\theta(a|s)$ is a δ-smooth function of θ and that $\pi^{(0)}$ is the uniform distribution. Let $d^{(t)} = d_\nu^{\pi_t}(s,a)$ and $d^\star(s,a) = d_\mu^{\pi^\star}(s)\pi^\star(a|s)$. Let ν be a distribution of s,a such that

$$\sup_{w\in\mathbb{R}^d}\frac{w^\top\Sigma_{d^\star}^{(t)}w}{w^\top\Sigma_\nu^{(t)}w} \leq \kappa,$$

where

$$\Sigma_\nu^\theta = \mathbb{E}_{s,a\sim\nu}\left[\nabla_\theta\log\pi_\theta(a|s)(\nabla_\theta\log\pi_\theta(a|s))^\top\right]$$

and $\Sigma^{(t)} = \Sigma^{\theta^{(t)}}$. Lastly, assume that

$$\mathbb{E}\left[L(w_\star^{(t)},\theta^{(t)},d^\star)\right] \leq \varepsilon_{\text{bias}}, \tag{4}$$

$$\mathbb{E}\left[L(w^{(t)},\theta^{(t)},d^{(t)}) - L(w_\star^{(t)},\theta^{(t)},d^{(t)})|\theta^{(t)}\right] \leq \varepsilon_{\text{stat}}, \tag{5}$$

where

$$w_\star^{(t)} \in \operatorname*{argmin}_{\|w\|_2\leq W} L(w,\theta^{(t)},d^{(t)})$$

and the expectations are taken w.r.t. the sequence $(w^{(t)})_{t=0,\dots,T-1}$. Given $\eta = \sqrt{2\frac{\log|\mathcal{A}|}{(\delta W^2 T)}}$, for a given parameter W, we have that performing the update in (3) for T iterations benefits of the following guarantee ([Agarwal et al., 2020], Theorem 6.2):

$$\mathbb{E}\left[\min_{t\leq T}\left\{V^{\pi^\star}(\mu) - V^{(t)}(\mu)\right\}\right] \leq \frac{W}{1-\gamma}\sqrt{\frac{2\delta\log|\mathcal{A}|}{T}} + \sqrt{\frac{\kappa\varepsilon_{\text{stat}}}{(1-\gamma)^3}} + \frac{\sqrt{\varepsilon_{\text{bias}}}}{1-\gamma}. \tag{6}$$

As we highlighted in the introduction, the guarantee (6) is particularly suitable for high-dimensional settings, as there is no explicit dependence on $|\mathcal{S}|$ and the dependence on $|\mathcal{A}|$ is only logarithmic. As to implicit dependencies, Agarwal et al. [2020] state that it is reasonable to expect that κ is not a quantity related to $|\mathcal{S}|$. On the other hand, $\varepsilon_{\text{stat}}$ and $\varepsilon_{\text{bias}}$ are constants related to a minimization problem that depends on both $\mathcal{S}$ and $\mathcal{A}$.

3 Curse of Dimensionality and Scalability/Implementabily Issues in MARL

When applied to the MARL setting, with $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_K$ and $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_K$, NPG incurs a curse of dimensionality or scalability and implementabily issues, depending on the approach used for the minimization problem in (8). The guarantee in (6) would yield

$$\mathbb{E}\left[\min_{t<T}\{V^{\pi^\star}(\rho) - V^{(t)}(\rho)\}\right] \leq \frac{WK}{1-\gamma}\sqrt{\frac{2\delta \log \max_{k\in\mathcal{K}} |\mathcal{A}_k|}{T}} + \sqrt{\frac{\kappa\varepsilon_{\text{stat}}}{(1-\gamma)^3}} + \frac{\sqrt{\varepsilon_{\text{bias}}}}{1-\gamma}, \tag{7}$$

where we assume (4), (5), and

$$w_\star^{(t)} \in \underset{\|w\|_2 \leq \sqrt{K}W}{\operatorname{argmin}} \; L(w, \theta^{(t)}, d^{(t)}). \tag{8}$$

With respect to the single agent setting, we increase the maximum norm of $w_\star^{(t)}$ from W to $\sqrt{K}W$, which is analogous to requiring $\|w_{k,\star}^{(t)}\|_2 \leq W$, for each agent $k \in \mathcal{K}$. Not doing so would mean keeping a constant parameter W despite increases in the dimensions, i.e. agents, of problem (8), incurring increases in the bias term.

In the multi-agent setting, the iteration complexity given by the bound in (7) is worse by a factor K, compared to the single-agent setting. The curse of dimensionality appears when solving the minimization problem in (8), e.g. with gradient descent because the computation of exact gradients involves a sum/integral over $\mathcal{S}\times\mathcal{A}$, which has a dimension that grows exponentially with the number of agents. If the minimization problem is solved with stochastic projected gradient descent, then the problem of computing gradients disappears as we *assume* access to samples to estimate gradients. However, the number of samples required to satisfy (5) becomes $O(K^2\varepsilon_{\text{stat}}^{-2})$, from being $O(\varepsilon_{\text{stat}}^{-2})$ in the single-agent setting, due to the increase in dimensionality of the update w, in particular due to the scaled-up constraint $\|w\|_2 \leq \sqrt{K}W$ that is used by a classical convergence result of stochastic projected gradient descent [Orabona, 2019]. Implementing a sampler could, in turn, involve a curse of dimensionality.

The dependencies of the minimization problem in (8) cause the algorithm to incur additional scalability and implementability issues. As the projection step, the advantage function and the policy gradient depend on the states and actions of the entire network, which do not factorize, each agent would have to communicate to every other agent in the network at each iteration to solve the problem. As mentioned in the introduction, requiring such a level of communication is rarely viable in real-world applications in the decentralized MARL setting.

Remark 3.1. These aforementioned problems do not arise in case of independent agents, where, for each agent k, the local transition probabilities satisfy $P_k(s'_k|s,a) = P_k(s'_k|s_k,a_k)$ and policies satisfy $\pi_{\theta_k}(a_k|s) = \pi_{\theta_k}(a_k|s_k)$. In this setting, as we show in Appendix E, it is possible to show that applying NPG to the whole network of K agents corresponds to running K independent runs of NPG applied to individual agents and to recover the same results of Agarwal et al. [2020] for the individual agents.

4 Exponential decay

In this section we present one of our main results, regarding the decay of correlations. We assume that a version of the Dobrushin condition [Georgii, 2011] holds for the transition dynamics of the network of agents. Let $TV(\mu,\nu) = \sup_{A\in\mathcal{F}} |\mu(A) - \nu(A)|$ be the total variation distance between the probability distributions μ and ν defined on the σ-algebra $\mathcal{F}$.

Assumption 4.1. (Spatial Decay of Correlation for the Dynamics) Let $C \in \mathbb{R}^{K\times K}$ be defined as follows:

$$C_{ij} = \sup_{s_j, s_{-j}, a_j, a_{-j}, s'_j, a'_j} TV(P_i(\cdot|s_j, s_{-j}, a_j, a_{-j}), P_i(\cdot|s'_j, s_{-j}, a'_j, a_{-j})).$$

Assume that there exists $\beta \geq 0$ such that

$$\max_{k\in\mathcal{K}} \sum_{j\in\mathcal{K}} e^{\beta d(k,j)} C_{kj} \leq \rho,$$

with $\rho < 1/\gamma$, where γ is the discount factor of the MDP.

The element (i, j) of the matrix C represents the influence that a perturbation of the state and action of agent j has on the transition probability of agent i. Assumption 4.1 encodes the fact that the transition dynamics of each agent is exponentially less sensible to perturbations of the state and action of further away agents. The requirement $\rho < 1/\gamma$ comes as we need the spatial decay of correlation for the dynamics to be strong enough to induce a spatial decay for the Q-function (see Appendix C). A small value of the discount factor γ eases this requirement since it reduces the effect of perturbations through time. When $\beta = 0$ and $\gamma = 1$, Assumption 4.1 recovers the assumption in Qu et al. [2020a] as a particular case.

Differently from Qu et al. [2020a], we require the Dobrushin condition to hold for the policy as well. This is due to the fact that Assumption 4.1 is sufficient to prove the decay of correlation for the Q-function, on which the algorithm designed by Qu et al. [2020a] is based, but it is not sufficient to prove the decay of correlation for the value function, which instead needs an additional assumption on the policy. Since the NPG framework on which we build upon is based on both the Q-function and the value function, we make the following additional assumptions.

Assumption 4.2. (Spatial Decay of Correlation for the Policy) Assume that there exist $\xi, \beta \geq 0$ such that, $\forall \theta \in \Theta$,

$$\sup_{s_{N_k^r}, s_{N_{-k}^r}, s'_{N_{-k}^r}} TV(\pi_{\theta_k}(\cdot|s_{N_k^r}, s_{N_{-k}^r}), \pi_{\theta_k}(\cdot|s_{N_k^r}, s'_{N_{-k}^r})) \leq \xi e^{-\beta r}.$$

Assumption 4.3. (Local Policy) Assume that, for any neighborhood radius r, the parameters θ_k can be partitioned in $(\theta_k)_{N_k^r}$ and $(\theta_k)_{N_{-k}^r}$ so that, if $(\theta_k)_{N_{-k}^r} = 0$, then

$$\pi_{\theta_k}(a_k|s) = \pi_{\theta_k}(a_k|s_{N_k^r}),$$
$$\nabla_{(\theta_k)_{N_k^r}} \log \pi_{\theta_k}(a_k|s) = \nabla_{(\theta_k)_{N_k^r}} \log \pi_{\theta_k}(a_k|s_{N_k^r}).$$

Assumptions 4.2 and 4.3 impose a design constraint for the policy class $\{\pi_\theta \,|\, \theta \in \Theta\}$ rather than being assumptions on the nature of the environment, as the case for Assumption 4.1. Assumption 4.2 encodes, for the policy, a type of decay of correlation property that is weaker than Assumption 4.1. Assumption 4.2 allows us to consider a policy class that presents properties that are necessary for the optimal policy under Assumption 4.1, as we show in Appendix D. Assumption 4.3 is made to address the communication constraints of the network and requires the possibility of computing the policy and its gradient without access to the information coming from distant agents by setting their associated parameters to 0. In practice, we do only need Assumption 4.3 to hold for the value of r we want Theorem 5.1 to hold for. In Appendix B, we describe a policy class that satisfies both Assumptions 4.2 and 4.3 for any value of r.

To take advantage of the local structure of the network, Lin et al. [2020] define a property regarding the dependence of $Q_k^\pi(s, a)$ on the neighbors of k.

Definition 4.4 ([Lin et al., 2020])**.** The (c, ψ)-exponential decay property for the Q-function holds if, for any agent $k \in \mathcal{K}$ and for any (s, a), $(\widetilde{s}, \widetilde{a}) \in \mathcal{S} \times \mathcal{A}$ such that $s_{N_k^r} = \widetilde{s}_{N_k^r}, a_{N_k^r} = \widetilde{a}_{N_k^r}$, we have that

$$|Q_k^\pi(s, a) - Q_k^\pi(\widetilde{s}, \widetilde{a})| \leq c\psi^{r+1}.$$

In our analysis, we need to define the exponential decay property for the value function as well.

Definition 4.5. The (c', ϕ)-exponential decay property for the value function holds if, for any agent $k \in \mathcal{K}$ and for any $s, \widetilde{s} \in \mathcal{S}$ such that $s_{N_k^r} = \widetilde{s}_{N_k^r}$, we have that

$$|V_k^\pi(s) - V_k^\pi(\widetilde{s})| \leq c'\phi^{r+1}.$$

These two properties mean that the cumulative discounted rewards of agents have an exponential decaying dependence on the states and actions of distant agents. We show that both these properties hold in our setting.

Proposition 4.6. *If Assumptions 4.1 and 4.2 hold, then the exponential decay property holds for both the Q-function and the value function with parameters* $(c, \psi) = \left(\frac{\gamma\rho e^\beta}{1-\gamma\rho}, e^{-\beta}\right)$ *and* $(c', \phi) = \left(\frac{\gamma(\rho+\xi)e^\beta}{1-\gamma(\rho+\xi)}, e^{-\beta}\right)$*, respectively.*

For clarity of exposition, in the rest of the paper we make the following assumption.

Assumption 4.7. Assume that the exponential decay property holds for the Q-function with parameters (c, ψ) and that it holds for the value function with parameters (c', ϕ).

5 Decentralized NPG

We design a decentralized version of NPG that is capable of exploiting the spatial decay of correlation properties that we assume and of avoiding the curse of dimensionality while still approximately converging to a globally optimal policy. We do so by limiting the communication range to agents that are at most at distance r and defining, for each agent k, the localized advantage function, localized value function, localized Q-function, as follows:

$$\widetilde{A}^{\pi}_k(s_{N^r_k}, a_{N^r_k}) = \widetilde{Q}^{\pi}_k(s_{N^r_k}, a_{N^r_k}) - \widetilde{V}^{\pi}_k(s_{N^r_k}),$$

$$\widetilde{V}^{\pi}_k(s_{N^r_k}) = \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r_k(s_k(t), a_k(t))\big|\pi, s_{N^r_k}(0) = s_{N^r_k}\right],$$

$$\widetilde{Q}^{\pi}_k(s_{N^r_k}, a_{N^r_k}) = \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r_k(s_k(t), a_k(t))\big|\pi, s_{N^r_k}(0) = s_{N^r_k}, a_{N^r_k}(0) = a_{N^r_k}\right].$$

Following Assumption 4.3, we set $(\theta_k)_{N^r_{-k}} = 0, \forall k \in \mathcal{K}$ and we never update these parameters, so that the policy of an agent and its gradient do not depend on the states of agents whose distance is greater than r. For each agent k, define the loss function

$$\widetilde{L}^r_k(w, \theta, \nu) = \mathbb{E}_{s,a\sim\nu}\left[\left(\widetilde{A}^{\pi_\theta}_k(s_{N^r_k}, a_{N^r_k}) - \nabla_{(\theta_k)_{N^r_k}}\log\pi_{\theta_k}(a_k|s_{N^r_k})\cdot w\right)^2\right]. \tag{9}$$

The minimization problem that each agent k aims to solve at each step becomes

$$w^\star \in \operatorname*{argmin}_{\|w\|_2\le W}\widetilde{L}^r_k(w, \theta^{(t)}, d^{(t)}). \tag{10}$$

In Section 5.1 we show how to solve this minimization problem in a decentralized manner and that, even if $d^{(t)}$ is a global distribution, it is possible to build a decentralized sampler of it assuming only access to a global clock.

Algorithm 1: Decentralized NPG

Input: Learning rate η; numbers of iterations T; an initialized policy $\pi^{(0)}$.

Set $(\theta_k)_{N^r_{-k}} = 0, \forall k \in \mathcal{K}$;

for $t = 0, \dots, T-1$; $k \in \mathcal{K}$ **do**

  Compute approximately $w^{(t)}_k \in \operatorname{argmin}_{\|w\|_2\le W}\widetilde{L}^r_k(w, \theta^{(t)}, d^{(t)})$;

  Compute the update $(\theta_k)^{(t+1)}_{N^r_k} = (\theta_k)^{(t)}_{N^r_k} + \frac{\eta}{1-\gamma}w^{(t)}_k$.

Exploiting decay of correlation properties and the policy design constraints in Assumption 4.3, Algorithm 1 removes the dependence on K from the iteration complexity bound and addresses the curse of dimensionality and scalability and implementability issues outlined in Section 3.

Theorem 5.1. *Assume that Assumption 4.3 and Assumption 4.7 hold. Assume that* $\log\pi_\theta(a|s)$ *is a* δ*-smooth function of* θ *and that* $\pi^{(0)}$ *is the uniform distribution. Let* $d^{(t)} = d^{\pi_t}_\nu(s,a)$ *and* $d^\star(s,a) = d^{\pi^\star}_\mu(s)\pi^\star(a|s)$*. Let* ν *be a distribution of* s, a *for which there exists* $\kappa' \ge 0$ *such that*

$$\max_{k\in\mathcal{K}}\sup_{w\in\mathbb{R}^d}\frac{w^\top\Sigma^{(t)}_{d^\star,k}w}{w^\top\Sigma^{(t)}_{\nu,k}w} \le \kappa',$$

where, $\forall\theta, \nu, k$

$$\Sigma^{\theta}_{\nu,k} = \mathbb{E}_{s,a\sim\nu}\left[\nabla_{(\theta_k)_{N^r_k}}\log\pi_\theta(a_k|s_{N^r_k})(\nabla_{(\theta_k)_{N^r_k}}\log\pi_\theta(a_k|s_{N^r_k}))^\top\right]$$

and $\Sigma^{(t)}_{\nu,k} \equiv \Sigma^{\theta^{(t)}}_{\nu,k}$*. Let*

$$\max_{k\in\mathcal{K}}\mathbb{E}\left[\widetilde{L}_k(w^{(t)}_{k,\star}, \theta^{(t)}, d^\star)\right] \le \varepsilon_{bias},$$

$$\max_{k\in\mathcal{K}}\mathbb{E}\left[\widetilde{L}_k(w^{(t)}_k, \theta^{(t)}, d^{(t)}) - \widetilde{L}_k(w^{(t)}_{k,\star}, \theta^{(t)}, d^{(t)})\big|\theta^{(t)}\right] \le \varepsilon_{stat},$$

where

$$w_{k,\star}^{(t)} \in \underset{\|w\|_2 \leq W}{\operatorname{argmin}} \widetilde{L}_k(w, \theta^{(t)}, d^{(t)}).$$

Given $\eta = \sqrt{2\frac{\log|\mathcal{A}|}{(\delta K W^2 T)}}$, *Algorithm 1 has the following guarantee:*

$$\mathbb{E}\left[\min_{t<T}\{V^{\pi^\star}(\mu) - V^{(t)}(\mu)\}\right] \leq \frac{W}{1-\gamma}\sqrt{\frac{2\delta \log \max_{k\in\mathcal{K}}|\mathcal{A}_k|}{T}} + \sqrt{\frac{\kappa' \varepsilon_{stat}}{(1-\gamma)^3}} + \frac{\sqrt{\varepsilon_{bias}}}{1-\gamma} + \underbrace{\frac{c\psi^{r+1} + c'\phi^{r+1}}{1-\gamma}}_{\text{localization error}}. \tag{11}$$

Agent-wise, the assumptions in Theorem 5.1 correspond to the assumptions made in Agarwal et al. [2020], in a setting where, for each agent k, the state space and the action space are $\mathcal{S}_{N_k^r}$ and $\mathcal{A}_k$, respectively, where the policy is defined as $\pi_\theta(a|s) = \pi_{\theta_k}(a_k|s_{N_k^r})$ and where the update $w_k^{(t)}$ is bounded by W. Theorem 5.1 shows that Decentralized NPG recovers the iteration complexity of the algorithm in Agarwal et al. [2020], worsened only by the fourth term on the RHS of (11). This localization error is exponentially small in r. Theorem 5.1 provides a dimension-free guarantee on the average expected cumulative rewards of the whole network, as the upper bound in (11) does not depend on the number of agents K in the network and depends only on the logarithm of the cardinality of the action space of an individual agent, with no explicit dependence on the state space.

Theorem 5.1 shows that, under the assumption on spatial decay of correlation, Decentralized NPG solves the curse of dimensionality and the scalability and implementability issues outlined in Section 3. The minimization problem in (10) can be solved approximately in a decentralized manner through stochastic projected gradient descent, as we show in the next section, leading to the same sample complexity as the single agent setting, i.e. $O(\varepsilon_{\text{stat}}^{-2})$. Additionally, Decentralized NPG can be run locally by each agent and only requires information from neighbors within distance r.

The role of the term $\varepsilon_{\text{bias}}$ in (11) has a difference from the role that $\varepsilon_{\text{bias}}$ has in (6) [Agarwal et al., 2020]. They both represent the worst-case error that is made by the agents when they approximate their current advantage function with a linear combination of the elements of the gradient of their current policy and encode the *transfer* error that we make shifting the distribution to $d^\star$. In addition to that, $\varepsilon_{\text{bias}}$ in (11) also encodes the localization error that we make in Algorithm 1 by using the localized loss defined in (9). In Appendix G we give a bound for this localization error of the bias term, showing that the localized bias is at most the non-localized bias, i.e. the bias associated with an infinite range parameter r, plus a quantity that decreases to 0 exponentially fast in r.

5.1 Sample complexity

We provide an unbiased sampler, that is Algorithm 2, and a sample complexity analysis for the optimization problem in (10).

Corollary 5.2. *Consider the setting of Theorem 5.1 and assume access to the sampler in Algorithm 2. If* $\|\nabla_{\theta_k} \log \pi^{(t)}(a|s)\|_2 \leq B$ *and the step-size is set as* $\alpha = W/(8B(BW + (1-\gamma)^{-1})\sqrt{N})$, *then running projected gradient descent on problem (10) for* N *iterations yields*

$$\varepsilon_{stat} \leq \frac{8BW(BW + \frac{1}{1-\gamma})}{\sqrt{N}}.$$

Proof. The proposition follows from a result on stochastic projected gradient descent [Shalev-Shwartz and Ben-David, 2014]. Consider the minimization problem $\min_{x\in C} f(x)$ for a convex function f, then performing the update

$$x_{t+1} = \text{Proj}_C(x_t - \alpha v_t),$$

where $C = \{x : \|x\|_2 \leq W\}$ for some $W \geq 0$, $\text{Proj}_C(\cdot)$ is the projection on C, v_t is such that $\mathbb{E}[v_t|x_t] = \nabla f(x_t)$ and $\|v_t\| \leq \rho$, for N steps and with $\alpha = \sqrt{W^2/(\rho^2 N)}$, gives

$$\mathbb{E}\left[f\left(\frac{1}{N}\sum_{t=1}^{N} x_t\right)\right] - f(x^\star) \leq \frac{W\rho}{\sqrt{N}},$$

Algorithm 2: Sampler

Input: Starting state-action distribution ν, a policy π_θ, access to a global clock.

For each agent $k \in \mathcal{K}$ set $\widehat{Q}_k = 0$, $\widehat{V}_k = 0$, sample $s_k(0), a_k(0) \sim \nu$ and start at state $s_k(0)$.

for $h \geq 0$, $k \in \mathcal{K}$ **do**

- with probability γ, execute $a_k(h)$, transition to $s_k(h+1)$ and sample $a_k(h+1) \sim \pi_{\theta_k}(\cdot|s_{N_k^r}(h+1))$;
- else accept $(s(h), a(h))$ as the sample and exit the loop, each agent only saves $\left(s_{N_k^r}(h), a_{N_k^r}(h)\right)$.

Set SampleQ=True with probability $1/2$;

if *SampleQ=True* **then**

$\forall k \in \mathcal{K}$ execute $a_k(h)$ and then, for every time-step $h' > h$ with termination probability γ, transition to $s_k(h')$, sample $a_k(h'+1) \sim \pi_{\theta_k}(\cdot|s_{N_k^r}(h'+1))$ and set $\widehat{Q}_k = \widehat{Q}_k + r_k(h')$;

else

$\forall k \in \mathcal{K}$ sample $a_k(h) \sim \pi_{\theta_k}(\cdot|s_{N_k^r}(h))$ and execute $a_k(h)$ and then, for every time-step $h' > h$ with termination probability γ, transition to $s_k(h')$, sample $a_k(h'+1) \sim \pi_k(\cdot|s_{N_k^r}(h'+1))$ and set $\widehat{V}_k = \widehat{V}_k + r_k(h')$.

Return: $\left(\widehat{A}_k\left(s_{N_k^r}(h), a_{N_k^r}(h)\right) = 2\left(\widehat{Q}_k - \widehat{V}_k\right)\right)_{k\in\mathcal{K}}$ and $(s(h), a(h))$.

where $x^\star \in \operatorname{argmin}_{x\in C} f(x)$. Noticing that $\widetilde{A}_k^{(t)}(s,a) \leq 2/(1-\gamma)$ and that the sampled gradient is therefore bounded by $8B(BW + (1-\gamma)^{-1})$ gives the corollary. □

The minimization problem in (10) can therefore be solved by each agent k through stochastic projected gradient descent, which only depends on the states and actions of N_k^r and on the parameters $(\theta_k)_{N_k^r}$ and with a sample complexity that does not depend on K.

6 Conclusion

We have investigated applications of the NPG framework to MARL, showing how a standard assumption on the spatial decay of correlation for the dynamics and for the policy on a network of agents, expressed through a form of Dobrushin condition, induces a form of exponential decay in the cumulative rewards that can be exploited by a localized version of NPG to avoid the curse of dimensionality. The version of NPG that we design scales to large networks and yields convergence guarantees to the optimal policy that are analogous to the ones in Agarwal et al. [2020], worsened only by a localization error that decreases exponentially with the communication radius. Our analysis does not consider regularization, which has been shown to accelerate convergence for NPG methods [Geist et al., 2019] and yield linear convergence rates [Cen et al., 2020].

References

Jeffrey L Adler and Victor J Blue. A cooperative multi-agent transportation management and route guidance system. *Transportation Research Part C: Emerging Technologies*, pages 433–454, 2002.

Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. Optimality and approximation with policy gradient methods in markov decision processes. *Conference on Learning Theory*, pages 64–66, 2020.

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10:251–276, 1998.

Borja Balle, Gilles Barthe, Marco Gaboardi, and Joseph Geumlek. Privacy amplification by mixing and diffusion mechanisms. In *Advances in Neural Information Processing Systems*, 2019.

Shalabh Bhatnagar, Richard S Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor–critic algorithms. *Automatica*, pages 2471–2482, 2009.

Shicong Cen, Chen Cheng, Yuxin Chen, Yuting Wei, and Yuejie Chi. Fast global convergence of natural policy gradient methods with entropy regularization. *arXiv preprint arXiv: 1912.02906v2*, 2020.

Deepayan Chakrabarti, Yang Wang, Chenxi Wang, Jurij Leskovec, and Christos Faloutsos. Epidemic thresholds in real networks. *ACM Transactions on Information and System Security*, pages 1–26, 2008.

Yuval Dagan, Constantinos Daskalakis, Nishanth Dikkala, and Siddhartha Jayanti. Learning from weakly dependent data under dobrushin's condition. In *Conference on Learning Theory*, pages 914–928. PMLR, 2019. ISBN 2640-3498.

Ilias Diakonikolas, Daniel M. Kane, Alistair Stewart, and Yuxin Sun. Outlier-robust learning of ising models under dobrushins condition. In *Conference on Learning Theory*, 2021.

RL Dobrusin. Definition of a system of random variables by means of conditional distributions. *Theory of Probability and its Applications*, pages 458–486, 1970.

Martin Dyer, Leslie Ann Goldberg, and Mark Jerrum. *Dobrushin conditions and systematic scan*, pages 327–338. 2006.

Hans Föllmer. A covariance estimate for gibbs measures. *Journal of Functional Analysis*, pages 387–395, 1982.

David Gamarnik. *Correlation decay method for decision, optimization, and inference in large-scale networks*, pages 108–121. 2013.

David Gamarnik, David A Goldberg, and Theophane Weber. Correlation decay in random decision networks. *Mathematics of Operations Research*, pages 229–261, 2014.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. In *International Conference on Machine Learning*, pages 2160–2169, 2019.

Hans-Otto Georgii. *Gibbs measures and phase transitions*. 2011.

Alison L Gibbs and Francis Edward Su. On choosing and bounding probability metrics. *International Statistical Review*, pages 419–435, 2002.

Haotian Gu, Xin Guo, Xiaoli Wei, and Renyuan Xu. Mean-field multi-agent reinforcement learning: A decentralized network approach. *arXiv preprint arXiv: 2108.02731*, 2021.

Carlos Guestrin, Daphne Koller, and Ronald Parr. Multiagent planning with factored mdps. In *Advances in Neural Information Processing Systems*, pages 1523–1530, 2001a.

Carlos Guestrin, Daphne Koller, and Ronald Parr. Max-norm projections for factored mdps. In *International Joint Conference on Artificial Intelligence*, pages 673–680, 2001b.

Carlos Guestrin, Michail Lagoudakis, and Ronald Parr. Coordinated reinforcement learning. In *International Conference on Machine Learning*, pages 227–234, 2002.

Thomas P Hayes. A simple condition implying rapid mixing of single-site dynamics on spin systems. In *Annual IEEE Symposium on Foundations of Computer Science*, pages 39–46, 2006.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In Abernethy Jacob and Agarwal Shivani, editors, *Conference on Learning Theory*, pages 2137–2143, 2020.

S Kakade. A natural policy gradient. *Advances in Neural Information Processing Systems*, 2002.

Daphne Koller and Ron Parr. Policy iteration for factored mdps. *arXiv preprint arXiv:1301.3869*, 2013.

Daphne Koller and Ronald Parr. Computing factored value functions for policies in structured mdps. In *International Joint Conference on Artificial Intelligence*, pages 1332–1339, 1999.

Yiheng Lin, Guannan Qu, Longbo Huang, and Adam Wierman. Distributed reinforcement learning in multi-agent networked systems. *arXiv preprint arXiv:2006.06555*, 2020.

Wenjun Mei, Shadi Mohagheghi, Sandro Zampieri, and Francesco Bullo. On the dynamics of deterministic epidemic propagation over networks. *Annual Reviews in Control*, pages 116–128, 2017.

Ioannis Mitliagkas and Lester Mackey. Improving gibbs sampler scan quality with dogs. In *International Conference on Machine Learning*, pages 2469–2477. PMLR, 2017. ISBN 2640-3498.

Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, pages 48–61, 2009.

Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

Christos H Papadimitriou and John N Tsitsiklis. The complexity of optimal queueing network control. In *IEEE Conference on Structure in Complexity Theory*, pages 318–322, 1994.

Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, pages 1180–1190, 2008.

Adarsh Prasad, Vishwak Srinivasan, Sivaraman Balakrishnan, and Pradeep Ravikumar. On learning ising models under huber's contamination model. *Advances in Neural Information Processing Systems*, 33, 2020.

Guannan Qu and Na Li. Exploiting fast decaying and locality in multi-agent mdp with tree dependence structure. In *IEEE Conference on Decision and Control*, 2019.

Guannan Qu, Yiheng Lin, Adam Wierman, and Na Li. Scalable multi-agent reinforcement learning for networked systems with average reward. *Advances in Neural Information Processing Systems*, pages 2074–2086, 2020a.

Guannan Qu, Adam Wierman, and Na Li. Scalable reinforcement learning of localized policies for multi-agent networked systems. In *Conference on Learning for Dynamics and Control*, 2020b.

Michael Rabbat and Robert Nowak. Distributed optimization in sensor networks. In *International Symposium on Information Processing in Sensor Networks*, pages 20–27, 2004.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 4295–4304, 2018.

Patrick Rebeschini and Ramon Van Handel. Can local particle filters beat the curse of dimensionality? *Annals of Applied Probability*, pages 2809–2866, 2015.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, pages 484–489, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George Van Den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. pages 354–359, 2017.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*, 2017.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pages 1057–1063, 1999.

Lin Yang and Mengdi Wang. Sample-optimal parametric q-learning using linearly additive features. In Chaudhuri Kamalika and Salakhutdinov Ruslan, editors, *International Conference on Machine Learning*, pages 6995–7004, 2019.

Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. Mean field multi-agent reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2018.

Kaiqing Zhang, Zhuoran Yang, and Tamer Basar. Networked multi-agent reinforcement learning in continuous spaces. In *IEEE Conference on Decision and Control*, pages 2771–2776, 2018a.

Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. Fully decentralized multi-agent reinforcement learning with networked agents. In *International Conference on Machine Learning*, pages 5872–5881, 2018b.

Rick Zhang and Marco Pavone. Control of robotic mobility-on-demand systems: a queueing-theoretical perspective. *The International Journal of Robotics Research*, pages 186–203, 2016.

Yan Zhang and Michael M Zavlanos. Distributed off-policy actor-critic reinforcement learning with policy consensus. In *IEEE Conference on Decision and Control*, pages 4674–4679, 2019.

Alessandro Zocca. Temporal starvation in multi-channel csma networks: an analytical framework. *Queueing Systems*, pages 241–263, 2019.

A Comparison to previous works

The results in Qu and Li [2019], Qu et al. [2020a], Lin et al. [2020], Qu et al. [2020b] are closely related to our work, as they also use decay of correlation assumptions to provably avoid the curse of dimensionality in MARL. Our contributions differ from these works in the following main ways.

1. (Continuous spaces) We consider continuous state and action spaces, while they consider finite state and action spaces.

2. (Decay of correlation) We consider a more general version of the decay of correlation property (Assumption 4.1) and, differently from them, we also require a decay of correlation property to hold for the policy (Assumption 4.2). Assumption 4.1 recovers the version they consider in Qu et al. [2020a] in the case $\beta = 0$ and $\gamma = 1$. The generality of our condition allows us to consider transition dynamics that are not truncated, as they do, and to control the truncation of the policy.

3. (Methodology) Our method is based on NPG framework, while their methods is based on policy gradient and actor-critic methods.

4. (Optimality) We present statistical error bounds w.r.t. to the *optimal* policy, while the bounds they give are w.r.t. a stationary policy.

5. (Computational complexity) Our method has a computational complexity that does not depend, for any agent k, on the number of agents K or the number of neighbors $|N_k^r|$. The method in Qu et al. [2020b] is shown to have a computational complexity that scales as $O(\log |\mathcal{S}||\mathcal{A}|)$, hence depending linearly on K, using additional assumptions on the minimum local exploration. The method in Lin et al. [2020] is shown to have computational complexity that scales as $O(\log \max_{k\in\mathcal{K}} |\mathcal{S}_{N_k^r}||\mathcal{A}_{N_k^r}|)$, hence depending linearly on $|N_k^r|$, using additional assumptions on the stationarity and on the mixing rates of the MDP.

6. (Statistical/Iteration Complexity) Under the only assumptions on decay of correlation and local policy, our method has an iteration complexity that scales as $O(\sqrt{\log \max_{k\in\mathcal{K}} |\mathcal{A}_k|})$. The methods in Qu et al. [2020b], Lin et al. [2020] have an iteration complexity that does not depend on the state or action spaces.

B Policy Class Example

Let $\widetilde{r} = \max_{k,k'\in\mathcal{K}} d(k,k')$ be the maximum distance between two agents. Define a set of parameterized differentiable functions $\{f_{(\theta_k)_r} : \mathcal{S}_{N_k^r} \times \mathcal{A}_k \to \mathcal{C} | 0 \leq r \leq \widetilde{r}\}$, where $\mathcal{C} \subset [-C, C]$ and $C > 0$, a set of parameters $\{\alpha_r \geq 0 | 0 \leq r \leq \widetilde{r}\}$ and let, for each agent k,

$$f_{\theta_k}(s, a_k) = \sum_{r=0}^{\widetilde{r}} \alpha_r f_{(\theta_k)_r}(s_{N_k^r}, a_k),$$

$$\pi_{\theta_k}(a_k|s) = \frac{\exp(f_{\theta_k}(s, a_k))}{\sum_{a'\in\mathcal{A}_k} \exp(f_{\theta_k}(s, a'))}.$$

By tuning the parameters α_r, we can make any policy belonging to this policy class respect Assumptions 4.2 and 4.3, as we show in the following. Let $r \in \{0, \dots, \widetilde{r}\}$, let $s, \widetilde{s} \in \mathcal{S}$ be such that

$s_{N_k^r} = \widetilde{s}_{N_k^r}$, then

$$
\begin{aligned}
TV(\pi_{\theta_k}(\cdot|s), \pi_{\theta_k}(\cdot|\widetilde{s})) &= \frac{1}{2}\sum_{a\in\mathcal{A}_k} |\pi_{\theta_k}(a|s) - \pi_{\theta_k}(a|\widetilde{s})| \\
&= \frac{1}{2}\sum_{a\in\mathcal{A}_k} \left| \frac{\exp(f_{\theta_k}(s,a))}{\sum_{a'\in\mathcal{A}_k}\exp(f_{\theta_k}(s,a'))} - \frac{\exp(f_{\theta_k}(\widetilde{s},a))}{\sum_{a'\in\mathcal{A}_k}\exp(f_{\theta_k}(\widetilde{s},a'))} \right| \\
&= \frac{\sum_{a\in\mathcal{A}_k} \left|\sum_{a'\in\mathcal{A}_k}\exp(f_{\theta_k}(s,a))\exp(f_{\theta_k}(\widetilde{s},a')) - \exp(f_{\theta_k}(\widetilde{s},a))\exp(f_{\theta_k}(s,a'))\right|}{2\sum_{a'\in\mathcal{A}_k}\exp(f_{\theta_k}(\widetilde{s},a'))\sum_{a'\in\mathcal{A}_k}\exp(f_{\theta_k}(s,a'))} \\
&\leq \frac{\sum_{a\in\mathcal{A}_k}\sum_{a'\in\mathcal{A}_k} |\exp(f_{\theta_k}(s,a))\exp(f_{\theta_k}(\widetilde{s},a')) - \exp(f_{\theta_k}(\widetilde{s},a))\exp(f_{\theta_k}(s,a'))|}{2\sum_{a'\in\mathcal{A}_k}\exp(f_{\theta_k}(\widetilde{s},a'))\sum_{a'\in\mathcal{A}_k}\exp(f_{\theta_k}(s,a'))} \\
&\leq \frac{\sum_{a\in\mathcal{A}_k} |\exp(f_{\theta_k}(\widetilde{s},a)) - \exp(f_{\theta_k}(s,a))|}{\sum_{a\in\mathcal{A}_k}\exp(f_{\theta_k}(\widetilde{s},a))} \\
&\leq \frac{\sum_{a\in\mathcal{A}_k} |f_{\theta_k}(\widetilde{s},a) - f_{\theta_k}(s,a)|\exp(\sup_{s'\in\{s,\widetilde{s}\}} f_{\theta_k}(s',a))}{\sum_{a\in\mathcal{A}_k}\exp(f_{\theta_k}(\widetilde{s},a))} \\
&\leq e^{2C(\widetilde{r}-r)}\frac{\sum_{a\in\mathcal{A}_k} |f_{\theta_k}(\widetilde{s},a) - f_{\theta_k}(s,a)|\exp(f_{\theta_k}(\widetilde{s},a))}{\sum_{a\in\mathcal{A}_k}\exp(f_{\theta_k}(\widetilde{s},a))} \\
&= e^{2C(\widetilde{r}-r)}\mathbb{E}_{\pi_{\theta_k}}\left| \sum_{r'=r+1}^{\widetilde{r}} \alpha_{r'}\left(f_{(\theta_k)_{r'}}(\widetilde{s}_{N_k^{r'}},a) - f_{(\theta_k)_{r'}}(s_{N_k^{r'}},a)\right)\right| \\
&\leq e^{2C(\widetilde{r}-r)}\sum_{r'=r+1}^{\widetilde{r}} \alpha_{r'}\mathbb{E}_{\pi_{\theta_k}}\left|f_{(\theta_k)_{r'}}(\widetilde{s}_{N_k^{r'}},a) - f_{(\theta_k)_{r'}}(s_{N_k^{r'}},a)\right| \\
&\leq 2Ce^{2C(\widetilde{r}-r)}\sum_{r'=r+1}^{\widetilde{r}} \alpha_{r'}.
\end{aligned}
$$

Setting the parameters $\{\alpha_{r'}\}_{r'\in\{r+1,\dots,\widetilde{r}\}}$ small enough ensures that the policy respects Assumption 4.2. Similarly, Assumption 4.3 is satisfied for a value r of the range parameter if $\alpha_{r'} = 0\ \forall r' \in \{r+1,\dots,\widetilde{r}\}$.

C Proof of Proposition 4.6

C.1 Preliminary Lemmas

To prove Proposition 4.6, we need a series of intermediate results, which we state and prove for completeness. Results similar to Lemmas C.1 and C.2 can be found in Chapter 8 of Georgii [2011], Lemma C.3 is an extension of results from Qu et al. [2020a].

Lemma C.1. *Let $f : \mathcal{Z} \to [m, M]$, where $\mathcal{Z} = \prod_{k\in\mathcal{K}}\mathcal{Z}_k$ and $m, M \in \mathbb{R}$. For every $k \in \mathcal{K}$, let μ_k and ν_k be two distributions on $\mathcal{Z}_k$. Let μ and ν be the respective product distributions. Let $\delta_k(f(z)) = \sup_{z_k, z_{-k}, z_k'} |f(z_k, z_{-k}) - f(z_k', z_{-k})|$. Then:*

$$|\mathbb{E}_{z\sim\mu} f(z) - \mathbb{E}_{z\sim\nu} f(z)| \leq \sum_{k\in\mathcal{K}} TV(\mu_k, \nu_k)\delta_k(f).$$

Proof. We prove Lemma C.1 by induction. Note that

$$TV(\mu,\nu) = \frac{1}{2}\max_{|h|\leq 1} |\mathbb{E}_\mu(h) - \mathbb{E}_\nu(h)|$$

is an equivalent formulation of the total variation distance [Gibbs and Su, 2002]. For $|\mathcal{K}| = 1$, we have that

$$\begin{aligned}
|\mathbb{E}_{\mu_1}(f) - \mathbb{E}_{\nu_1}(f)| &= \left|\mathbb{E}_{\mu_1}\left(f - \frac{M+m}{2}\right) - \mathbb{E}_{\nu_1}\left(f - \frac{M+m}{2}\right)\right| \\
&= \frac{M-m}{2}\left|\mathbb{E}_{\mu_1}\left(\frac{2f}{M-m} - \frac{M+m}{M-m}\right) - \mathbb{E}_{\nu_1}\left(\frac{2f}{M-m} - \frac{M+m}{M-m}\right)\right| \\
&\leq \frac{M-m}{2}\max_{|h|\leq 1}|\mathbb{E}_{\mu_1}(h) - \mathbb{E}_{\nu_1}(h)| \\
&= TV(\mu_1, \nu_1)\delta_1(f).
\end{aligned}$$

As induction assumption, assume that Lemma C.1 holds for $|\mathcal{K}| - 1$. Then:

$$\begin{aligned}
|\mathbb{E}_{z\sim\mu}f(z) - \mathbb{E}_{z\sim\nu}f(z)| &= |\mathbb{E}_{z_1\sim\mu_1}\mathbb{E}_{z_{2:n}\sim\mu_{2:n}}f(z) - \mathbb{E}_{z_1\sim\nu_1}\mathbb{E}_{z_{2:n}\sim\nu_{2:n}}f(z)| \\
&\leq |\mathbb{E}_{z_1\sim\mu_1}\mathbb{E}_{z_{2:n}\sim\mu_{2:n}}f(z) - \mathbb{E}_{z_1\sim\mu_1}\mathbb{E}_{z_{2:n}\sim\nu_{2:n}}f(z)| \\
&\quad + |\mathbb{E}_{z_1\sim\mu_1}\mathbb{E}_{z_{2:n}\sim\nu_{2:n}}f(z) - \mathbb{E}_{z_1\sim\nu_1}\mathbb{E}_{z_{2:n}\sim\nu_{2:n}}f(z)| \\
&\leq \mathbb{E}_{z_1\sim\mu_1}|\mathbb{E}_{z_{2:n}\sim\mu_{2:n}}f(z) - \mathbb{E}_{z_{2:n}\sim\nu_{2:n}}f(z)| \\
&\quad + \left|\mathbb{E}_{z_1\sim\mu_1}\widetilde{f}(z_1) - \mathbb{E}_{z_1\sim\nu_1}\widetilde{f}(z_1)\right|.
\end{aligned}$$

where $\widetilde{f}(z_1) = \mathbb{E}_{z_{2:n}\sim\nu_{2:n}}f(z)$. By induction assumption:

$$|\mathbb{E}_{z_{2:n}\sim\mu_{2:n}}f(z) - \mathbb{E}_{z_{2:n}\sim\nu_{2:n}}f(z)| \leq \sum_{k\neq 1\in\mathcal{K}} TV(\mu_k, \nu_k)\delta_k(f(z_1, \cdot)) \leq \sum_{k\neq 1\in\mathcal{K}} TV(\mu_k, \nu_k)\delta_k(f).$$

Since

$$\begin{aligned}
\delta_1(\widetilde{f}) &= \sup_{z_1, z_1'}|\mathbb{E}_{z_{2:n}\sim\nu_{2:n}}f(z_1, z_{2:n}) - \mathbb{E}_{z_{2:n}\sim\nu_{2:n}}f(z_1', z_{2:n})| \\
&\leq \sup_{z_1, z_1'}\mathbb{E}_{z_{2:n}\sim\nu_{2:n}}|f(z_1, z_{2:n}) - f(z_1', z_{2:n})| \\
&\leq \sup_{z_1, z_1'}\sup_{z_{2:n}}|f(z_1, z_{2:n}) - f(z_1', z_{2:n})| = \delta_1(f),
\end{aligned}$$

we have

$$\begin{aligned}
|\mathbb{E}_{z\sim\mu}f(z) - \mathbb{E}_{z\sim\nu}f(z)| &\leq \mathbb{E}_{z_1\sim\mu_1}\sum_{k\neq 1\in\mathcal{K}} TV(\mu_k, \nu_k)\delta_k(f) + TV(\mu_1, \nu_1)\delta_1(f) \\
&\leq \sum_{k\in\mathcal{K}} TV(\mu_k, \nu_k)\delta_k(f),
\end{aligned}$$

which concludes the induction. □

Lemma C.2. *Consider a Markov Chain with state $z \in \mathcal{Z}$, where $\mathcal{Z} = \prod_{k\in\mathcal{K}}\mathcal{Z}_k$ and $\mathcal{K}$ is defined as in Section 2. Suppose its transition probability factorizes as*

$$P(z(t+1)|z(t)) = \prod_{k\in\mathcal{K}} P_k(z_k(t+1)|z(t)).$$

Let $C \in \mathbb{R}^{K\times K}$ be a matrix whose elements respect the condition

$$C_{ij} \geq \sup_{z_j, z_{-j}, z_j'} TV(P_i(\cdot|z_j, z_{-j}), P_i(\cdot|z_j', z_{-j})).$$

If $\sum_{j\in\mathcal{K}} e^{\beta d(j,k)}C_{kj} \leq \rho$, then, $\forall\mathcal{J} \subseteq \mathcal{K}$,

$$\sup_{z_j, z_{-j}, z_j'} TV(P_i(\cdot|z_J, z_{-J}), P_i(\cdot|z_J', z_{-J})) \leq \sum_{j\in\mathcal{J}} C_{ij}$$

and

$$\sup_{z_J, z_{-J}, z_J'} TV(P_i(\cdot|z_J, z_{-J}), P_i(\cdot|z_J', z_{-J})) \leq \rho e^{-\beta d(\mathcal{J}, i)},$$

where $d(\mathcal{J}, i) = \min_{j\in\mathcal{J}} d(j, i)$.

Proof. We prove the first claim of Lemma C.2 by induction. The first claim clearly holds if $|\mathcal{J}| = 1$. As induction assumption, assume that the first claim holds for a generic $\mathcal{J}$. Then, it holds for $\mathcal{J}' = \mathcal{J} + \{k\}$:

$$\sup_{z_j, z_{-j}, z'_j} TV(P_i(\cdot|z_{J'}, z_{-J'}), P_i(\cdot|z'_{J'}, z_{-J'})) = \sup_{\substack{A \subseteq \mathcal{Z}_i \\ z_j, z_{-j}, z'_j}} |P_i(A|z_{J'}, z_{-J'}) - P_i(A|z'_{J'}, z_{-J'})|$$

$$\leq \sup_{\substack{A \subseteq \mathcal{Z}_i \\ z_j, z_{-j}, z'_j}} |P_i(A|z_{J'}, z_{-J'}) - P_i(A|z'_J, z_{-J})| + \sup_{\substack{A \subseteq \mathcal{Z}_i \\ z_j, z_{-j}, z'_j}} |P_i(A|z'_J, z_{-J}) - P_i(A|z'_{J'}, z_{-J'})|$$

$$\leq \sum_{j \in \mathcal{J}} C_{ij} + C_{ik} = \sum_{j \in \mathcal{J}'} C_{ij}.$$

The second claim follows immediately, since

$$e^{\beta d(\mathcal{J}, i)} \sum_{j \in \mathcal{J}} C_{ij} \leq \sum_{j \in \mathcal{J}} e^{\beta d(j,i)} C_{ij} \leq \sum_{j \in \mathcal{K}} e^{\beta d(j,i)} C_{ij} \leq \rho,$$

and

$$\sum_{j \in \mathcal{J}} C_{ij} \leq \rho e^{-\beta d(\mathcal{J}, i)}.$$

□

Lemma C.3. *Consider the setting of Lemma C.2. For a generic value of r, denote by d_t and $\widetilde{d}_t$ the distribution of $z(t)$ with starting state, respectively, $z = (z_{N_k^r}, z_{N_{-k}^r})$ and $\widetilde{z} = (z_{N_k^r}, \widetilde{z}_{N_{-k}^r})$. Then, if $\sum_{j \in \mathcal{K}} e^{\beta d(j,k)} C_{kj} \leq \rho$, we have that $TV(d_{t,k}, \widetilde{d}_{t,k}) \leq \rho^t e^{-\beta r}$, $\forall k \in \mathcal{K}$.*

Proof. We prove Lemma C.3 by induction. The case where $t = 1$ follows from Lemma C.2. As induction assumption, assume that Lemma C.3 holds for t. Then,

$$\begin{aligned}
&\left| \mathbb{E}_{s \sim d_{t+1,k}(s)} \mathbf{1}_A(s) - \mathbb{E}_{s \sim \widetilde{d}_{t+1,k}} \mathbf{1}_A(s) \right| \\
&= \left| \mathbb{E}_{z \sim d_t} E_{s \sim P_k(\cdot|z)} \mathbf{1}_A(s) - \mathbb{E}_{z \sim \widetilde{d}_t} \mathbb{E}_{s \sim P_k(\cdot|z)} \mathbf{1}_A(s) \right| \\
&\leq \sum_{j \in \mathcal{K}} TV(d_{t,j}, \widetilde{d}_{t,j}) \delta_j (E_{s \sim P_k(\cdot|\cdot)} \mathbf{1}_A(s)) \\
&\leq \sum_{j \in \mathcal{K}} TV(d_{t,j}, \widetilde{d}_{t,j}) C_{kj} \\
&\leq \sum_{j \in \mathcal{K}} \rho^t e^{-\beta(r - d(j,k))} C_{kj} \\
&= \rho^t e^{-\beta r} \sum_{j \in \mathcal{K}} e^{\beta d(j,k)} C_{kj} \leq \rho^{t+1} e^{-\beta r},
\end{aligned}$$

where we used Lemma C.1 in the first inequality. □

Lemma C.4. *Consider the setting of Lemma C.2. Let $P^t(z'|z) = P(z(t) = z'|z(0) = z)$ and*

$$\delta_i P_k^t = \sup_{z_i, z_{-i}, z'_i} TV(P_k^t(\cdot|z_i, z_{-i}), P_k^t(\cdot|z'_i, z_{-i})).$$

If $\sum_{j \in \mathcal{K}} e^{\beta d(j,k)} C_{kj} \leq \rho$, then $\forall k \in \mathcal{K}$

$$\sum_{i \in \mathcal{K}} e^{\beta d(k,i)} \delta_i P_k^t \leq \rho^t.$$

Proof. We prove Lemma C.4 by induction. The claim holds for $t = 1$:

$$\sum_{i \in \mathcal{K}} e^{\beta d(k,i)} \delta_i P_k^t = \sum_{i \in \mathcal{K}} e^{\beta d(k,i)} C_{k,i} \leq \rho$$

As induction assumption, we assume that the claim holds for t. Then, using Lemma C.1,

$$\begin{aligned}
\delta_i P_k^{t+1} &= \sup_{\substack{A\subseteq\mathcal{S}_k \\ z_i,z_{-i},z_i'}} \left|\mathbb{E}_{s\sim P_k^{t+1}(\cdot|z_i,z_{-i})}\mathbf{1}_A(s) - \mathbb{E}_{s\sim P_k^{t+1}(\cdot|z_i',z_{-i})}\mathbf{1}_A(s)\right| \\
&= \sup_{\substack{A\subseteq\mathcal{S}_k \\ z_i,z_{-i},z_i'}} \left|\mathbb{E}_{x\sim P^t(\cdot|z_i,z_{-i})}E_{s\sim P_k(\cdot|x)}\mathbf{1}_A(s) - \mathbb{E}_{x\sim P^t(\cdot|z_i',z_{-i})}\mathbb{E}_{s\sim P_k(\cdot|x)}\mathbf{1}_A(s)\right| \\
&\leq \sup_{z_i,z_{-i},z_i'} \sum_{j\in\mathcal{K}} TV(P_j^t(\cdot|z_i,z_{-i}), P_j^t(\cdot|z_i',z_{-i}))\delta_j(E_{s\sim P_k(\cdot|\cdot)}\mathbf{1}_A(s)) \\
&\leq \sum_{j\in\mathcal{K}} \delta_i P_j^t C_{kj}
\end{aligned}$$

and, using the inverse triangle inequality,

$$\begin{aligned}
\sum_{i\in\mathcal{K}} e^{\beta d(k,i)}\delta_i P_k^{t+1} &\leq \sum_{i\in\mathcal{K}} e^{\beta d(k,i)} \sum_{j\in\mathcal{K}} \delta_i P_j^t C_{kj} \\
&\leq \sum_{j\in\mathcal{K}} e^{\beta d(k,j)} C_{kj} \sum_{i\in\mathcal{K}} e^{\beta(d(k,i)-d(k,j))}\delta_i P_j^t \\
&\leq \sum_{j\in\mathcal{K}} e^{\beta d(k,j)} C_{kj} \sum_{i\in\mathcal{K}} e^{\beta d(j,i)}\delta_i P_j^t \leq \rho^{t+1},
\end{aligned}$$

which concludes the induction. □

C.2 Main Result

Proof of Proposition 4.6. The following holds for every $k \in \mathcal{K}$. Let $s, \widetilde{s} \in \mathcal{S}$, $a, \widetilde{a} \in \mathcal{A}$ be such that $s_{N_k^r} = \widetilde{s}_{N_k^r}$ and $a_{N_k^r} = \widetilde{a}_{N_k^r}$. Notice that

$$\begin{aligned}
&|Q_k^\pi(s,a) - Q_k^\pi(\widetilde{s},\widetilde{a})| \\
&\leq \sum_{t=0}^{\infty} \gamma^t \left|\mathbb{E}\left[r_k(s_k(t),a_k(t))\middle|\pi, s(0)=s, a(0)=a\right] - \mathbb{E}\left[r_k(s_k(t),a_k(t))\middle|\pi, s(0)=\widetilde{s}, a(0)=\widetilde{a}\right]\right| \\
&\leq \sum_{t=1}^{\infty} \gamma^t \left|\mathbb{E}\left[r_k(s_k(t),a_k(t))\middle|\pi, s(0)=s, a(0)=a\right] - \mathbb{E}\left[r_k(s_k(t),a_k(t))\middle|\pi, s(0)=\widetilde{s}, a(0)=\widetilde{a}\right]\right| \\
&\leq \sum_{t=1}^{\infty} \gamma^t TV(d_{t,k}, \widetilde{d}_{t,k}),
\end{aligned}$$

where $d_{t,k}(s_k,a_k)$ and $\widetilde{d}_{t,k}(s_k,a_k)$ are the distributions of s_k, a_k at time t with starting point (s,a) and $(\widetilde{s},\widetilde{a})$, respectively. We use Lemma C.3 to bound $TV(d_{t,k}, \widetilde{d}_{t,k})$. The structure of our MDP implies that:

$$P(s(t+1),a(t+1)|s(t),a(t)) = \prod_{k\in\mathcal{K}} \pi^k(a_k(t+1)|s(t+1))P_k(s_k(t+1)|s(t),a(t)).$$

Let C be defined as in Assumption 4.1 and note that

$$\begin{aligned}
C_{kj} &= \sup_{s_j,s_{-j},a_j,a_{-j},s_j',a_j'} TV(P_k(\cdot|s_j,s_{-j},a_j,a_{-j}), P_k(\cdot|s_j',s_{-j},a_j',a_{-j})) \\
&\geq \sup_{s_j,s_{-j},a_j,a_{-j},s_j',a_j'} TV(P_k(\cdot,\cdot|s_j,s_{-j},a_j,a_{-j}), P_k(\cdot,\cdot|s_j',s_{-j},a_j',a_{-j})).
\end{aligned}$$

Then, if Assumption 4.1 holds, the requirements of Lemma C.3 are satisfied. Therefore, $TV(d_{t,k}, \widetilde{d}_{t,k}) \leq \rho^t e^{-\beta r}$ and

$$|Q_k^\pi(s,a) - Q_k^\pi(\widetilde{s},\widetilde{a})| \leq \sum_{t=1}^{\infty} \gamma^t TV(d_{t,k}, \widetilde{d}_{t,k}) \leq e^{-\beta r} \sum_{t=1}^{\infty} \gamma^t \rho^t = \frac{\gamma\rho e^{-\beta r}}{1-\gamma\rho}.$$

We use this result to prove the exponential decay property for the value function. Let

$$\delta_j Q_k^\pi(s,a) = \sup_{s_j,s_{-j},a_j,a_{-j},s'_j,a'_j} |Q_k^\pi(s_j,s_{-j},a_j,a_{-j}) - Q_k^\pi(s'_j,s_{-j},a'_j,a_{-j})|.$$

Using Lemma C.1 and Assumption 4.2, we have that

$$\begin{aligned}
&|V_k^\pi(s) - V_k^\pi(\widetilde{s})| = \left|\mathbb{E}_{a\sim\pi(\cdot|s)} Q_k^\pi(s,a) - \mathbb{E}_{a\sim\pi(\cdot|\widetilde{s})} Q_k^\pi(\widetilde{s},a)\right| \\
&\leq \left|\mathbb{E}_{a\sim\pi(\cdot|s)} Q_k^\pi(s,a) - \mathbb{E}_{a\sim\pi(\cdot|\widetilde{s})} Q_k^\pi(s,a)\right| + \left|\mathbb{E}_{a\sim\pi(\cdot|\widetilde{s})} Q_k^\pi(s,a) - \mathbb{E}_{a\sim\pi(\cdot|\widetilde{s})} Q_k^\pi(\widetilde{s},a)\right| \\
&\leq \sum_{i\in\mathcal{K}} TV(\pi_i(\cdot|s),\pi_i(\cdot|\widetilde{s}))\delta_i Q_k^\pi(s,a) + \frac{\gamma\rho e^{-\beta r}}{1-\gamma\rho} \\
&\leq \xi e^{-\beta r} \sum_{i\in\mathcal{K}} e^{-\beta d(i,k)} \delta_i Q_k^\pi(s,a) + \frac{\gamma\rho e^{-\beta r}}{1-\gamma\rho}.
\end{aligned}$$

We have already shown that the MDP satisfies the condition of Lemma C.4. Using it we obtain

$$\sum_{i\in\mathcal{K}} e^{\beta d(k,i)} \delta_i(Q_k^\pi(s,\cdot)) \leq \sum_{t=1}^{\infty} \gamma^t \sum_{i\in\mathcal{K}} e^{\beta d(k,i)} \delta_i P_k^t \leq \sum_{t=1}^{\infty} \gamma^t \rho^t = \frac{\gamma\rho}{1-\gamma\rho},$$

where

$$\delta_i P_k^t = \sup_{s_j,s_{-j},a_j,a_{-j},s'_j,a'_j} TV(P_k^t(\cdot,\cdot|s_j,s_{-j},a_j,a_{-j}), P_k^t(\cdot,\cdot|s'_j,s_{-j},a'_j,a_{-j})).$$

Then, we have that

$$|V_k^\pi(s) - V_k^\pi(\widetilde{s})| \leq \frac{\gamma\rho(1+\xi)e^{-\beta r}}{1-\gamma\rho}.$$

□

D Decay for the Optimal Policy

Lemma D.1. *Assume that the exponential decay property holds for the Q-function with parameters (c,ψ). Then the exponential decay property holds also for the value function associated with the optimal policy, with parameters $(3c,\psi)$.*

Proof. The following holds for every $k \in \mathcal{K}$. Let $s,\widetilde{s} \in \mathcal{S}$ be such that $s_{N_k^r} = \widetilde{s}_{N_k^r}$ and let $a_{N_{-k}^r} \in \mathcal{A}_{N_{-k}^r}$.

$$\begin{aligned}
&|V_k^\star(s) - V_k^\star(\widetilde{s})| = \left|\mathbb{E}_{a\sim\pi^\star(\cdot|s)} Q_k^\star(s,a) - \mathbb{E}_{a\sim\pi^\star(\cdot|\widetilde{s})} Q_k^\star(\widetilde{s},a)\right| = \left|\max_a Q_k^\star(s,a) - \max_a Q_k^\star(\widetilde{s},a)\right| \\
&= \Bigg|\max_a Q_k^\star(s,a) - \max_{a_{N_k^r}} Q_k^\star(s,a_{N_k^r},a_{N_{-k}^r}) + \max_{a_{N_k^r}} Q_k^\star(s,a_{N_k^r},a_{N_{-k}^r}) \\
&\quad - \max_a Q_k^\star(\widetilde{s},a) - \max_{a_{N_k^r}} Q_k^\star(\widetilde{s},a_{N_k^r},a_{N_{-k}^r}) + \max_{a_{N_k^r}} Q_k^\star(\widetilde{s},a_{N_k^r},a_{N_{-k}^r})\Bigg| \\
&= \left|\max_{a_{N_k^r}} Q_k^\star(s,a_{N_k^r},a_{N_{-k}^r}) - \max_{a_{N_k^r}} Q_k^\star(\widetilde{s},a_{N_k^r},a_{N_{-k}^r})\right| \\
&\quad + \left|\max_a Q_k^\star(\widetilde{s},a) - \max_{a_{N_k^r}} Q_k^\star(\widetilde{s},a_{N_k^r},a_{N_{-k}^r})\right| + \left|\max_a Q_k^\star(s,a) - \max_{a_{N_k^r}} Q_k^\star(s,a_{N_k^r},a_{N_{-k}^r})\right| \\
&\leq \left|\max_{a_{N_k^r}} Q_k^\star(s,a_{N_k^r},a_{N_{-k}^r}) - \max_{a_{N_k^r}} Q_k^\star(\widetilde{s},a_{N_k^r},a_{N_{-k}^r})\right| + 2c\psi^{r+1}.
\end{aligned}$$

Let $a'_{N_k^r} \in \operatorname{argmax}_{a_{N_k^r}} Q_k^\star(s,a_{N_k^r},a_{N_{-k}^r})$, then

$$\begin{aligned}
&Q_k^\star(s,a'_{N_k^r},a_{N_{-k}^r}) - \max_{a_{N_k^r}} Q_k^\star(\widetilde{s},a_{N_k^r},a_{N_{-k}^r}) \\
&\leq Q_k^\star(\widetilde{s},a'_{N_k^r},a_{N_{-k}^r}) - \max_{a_{N_k^r}} Q_k^\star(\widetilde{s},a_{N_k^r},a_{N_{-k}^r}) + c\psi^{r+1}
\end{aligned}$$

$$\leq \max_{a_{N_k^r}} Q_k^\star(\widetilde{s}, a_{N_k^r}, a_{N_{-k}^r}) - \max_{a_{N_k^r}} Q_k^\star(\widetilde{s}, a_{N_k^r}, a_{N_{-k}^r}) + c\psi^{r+1} = c\psi^{r+1}.$$

The same holds for $\max_{a_{N_k^r}} Q_k^\star(\widetilde{s}, a_{N_k^r}, a_{N_{-k}^r}) - \max_{a_{N_k^r}} Q_k^\star(s, a_{N_k^r}, a_{N_{-k}^r})$. The lemma follows immediately. □

We make an assumption on the minimum influence that the action of an agent has on its expected future rewards. Assumption D.2 and Proposition D.3 hold for any $k \in \mathcal{K}$.

Assumption D.2. Let $\mathcal{A}^\star = \operatorname{argmax}_{a \in \mathcal{A}} Q_k^\star(s, a), \forall s \in \mathcal{S}$. Assume that, if $\widetilde{a}$ is such that $\widetilde{a}_k \notin \mathcal{A}_k^\star$, then

$$|Q_k^\star(s, a) - Q_k^\star(s, \widetilde{a})| \geq R$$

Proposition D.3. *Assume that the exponential decay property holds for the Q-function with parameters (c, ψ) and that Assumption D.2 holds. Let $s, \widetilde{s} \in \mathcal{S}$ be such that $s_{N_k^r} = \widetilde{s}_{N_k^r}$. Let $\mathcal{A}^\star = \operatorname{argmax}_{a \in \mathcal{A}} Q_k^\star(s, a)$ and $\widetilde{a} \in \operatorname{argmax}_{a \in \mathcal{A}} Q_k^\star(\widetilde{s}, a)$. If $r > \log_\psi R/4c$, then $\widetilde{a}_k \in \mathcal{A}_k^\star$.*

Proof. We prove this by contradiction. Lemma D.1 shows that, $\forall r > 0$, if $s, \widetilde{s} \in \mathcal{S}$ are such that $s_{N_k^r} = \widetilde{s}_{N_k^r}$,

$$|V_k^\star(s) - V_k^\star(\widetilde{s})| = \left|\max_a Q_k^\star(s, a) - \max_a Q_k^\star(\widetilde{s}, a)\right| \leq 3c\psi^{r+1}.$$

Let $a \in \operatorname{argmax}_{a \in \mathcal{A}} Q_k^\star(s, a)$ and $\widetilde{a} \in \operatorname{argmax}_{a \in \mathcal{A}} Q_k^\star(\widetilde{s}, a)$. Let $\mathcal{A}^\star = \operatorname{argmax}_{a \in \mathcal{A}} Q_k^\star(s, a)$ and assume that $\widetilde{a}_k \notin \mathcal{A}_k^\star$. Then

$$\begin{aligned}
|Q_k^\star(s, a) - Q_k^\star(\widetilde{s}, \widetilde{a})| &= |Q_k^\star(s, a) - Q_k^\star(s, \widetilde{a}) + Q_k^\star(s, \widetilde{a}) - Q_k^\star(\widetilde{s}, \widetilde{a})| \\
&\geq ||Q_k^\star(s, a) - Q_k^\star(s, \widetilde{a})| - |Q_k^\star(s, \widetilde{a}) - Q_k^\star(\widetilde{s}, \widetilde{a})|| \\
&\geq |Q_k^\star(s, a) - Q_k^\star(s, \widetilde{a})| - c\psi^{r+1} \geq R - c\psi^{r+1}
\end{aligned}$$

where we used Assumption D.2 in the last passage. Then, due to Lemma D.1, if $r > \log_\psi R/4c$ we have a contradiction. □

Proposition D.3 shows that the optimal policy of an agent is not influenced by distant agents. Assumption 4.2 ensures that the policy class we consider respects this condition.

E Independent Agents

By completely independent agents we mean agents whose transition dynamics are independent and whose policy is defined, for $s \in \mathcal{S}, a \in \mathcal{A}$, as:

$$\pi_\theta(a|s) = \prod_{k \in \mathcal{K}} \pi_{\theta_k}(a_k|s_k) = \prod_k \frac{e^{f_{\theta_k}(s_k, a_k)}}{\sum_{a' \in \mathcal{A}_k} e^{f_{\theta_k}(s_k, a')}}.$$

In accordance to previous assumptions, we also assume that $\pi_{\theta_k}(a_k|s_k)$ is δ-smooth.

Proof of Remark 3.1. Firstly we show that the two applications of the algorithm coincide. Let

$$L(w) = \mathbb{E}_{s \sim d_\nu^\pi, a \sim \pi_\theta(\cdot|s)} \left[\left(A^{\pi_\theta}(s, a) - w \cdot \nabla_\theta \log \pi_\theta(a|s) \right)^2 \right].$$

In Agarwal et al. [2020], the NPG update is

$$w_\star \in \operatorname*{argmin}_w L(w).$$

We now show that the gradient of the loss function is the same as the one in the single agent setting. This is enough to show that the two algorithms coincide, since every other operation coincides. The only exception is the projection step, problem that can be side-stepped by projecting each single component of the gradient vector instead of the whole vector. For each agent k we have that

$$\begin{aligned}
\nabla_{\theta_k} L(w) &= \mathbb{E}_{s \sim d_\nu^\pi, a \sim \pi_\theta(\cdot|s)} \left[\left(A^{\pi_\theta}(s, a) - w \cdot \nabla_\theta \log \pi_\theta(a|s) \right) \nabla_{\theta_k} \log \pi_\theta(a|s) \right] \\
&= \sum_{j \in \mathcal{K}} \mathbb{E}_{s \sim d_\nu^\pi, a \sim \pi_\theta(\cdot|s)} \left[\left(\frac{1}{K} A_j^{\pi_\theta}(s_j, a_j) - w_j \cdot \nabla_{\theta_j} \log \pi_\theta(a_j|s_j) \right) \nabla_{\theta_k} \log \pi_\theta(a|s) \right] \\
&= E_{s_k \sim d_{\nu,k}^\pi, a_k \sim \pi_{\theta_k}^k(\cdot|s_k)} \left[\left(\frac{1}{K} A_k^{\pi_\theta}(s_k, a_k) - w_k \cdot \nabla_{\theta_k} \log \pi_\theta(a_k|s_k) \right) \nabla_{\theta_k} \log \pi_\theta(a_k|s_k) \right],
\end{aligned}$$

which corresponds to the gradient for the single agent setting.

With regards to guarantees, since the problem is decoupled, for any agent k we have the same result as in Agarwal et al. [2020]:

$$\mathbb{E}\left[\min_{t<T}\{V_k^{\pi^\star}(\rho) - V_k^{(t)}(\rho)\}\right] \leq \frac{W}{1-\gamma}\sqrt{\frac{2\delta \log|\mathcal{A}_k|}{T}} + \sqrt{\frac{\kappa\varepsilon_{\text{stat},k}}{(1-\gamma)^3}} + \frac{\sqrt{\varepsilon_{\text{bias},k}}}{1-\gamma},$$

where we assumed that

$$\mathbb{E}\left[L_k(w_k^{(t)}, \theta_k^{(t)}, d_k^{(t)}) - L(w_{\star,k}^{(t)}, \theta_k^{(t)}, d_k^{(t)})|\theta_k^{(t)}\right] \leq \varepsilon_{\text{stat},k},$$

$$\mathbb{E}\left[L(w_{\star,k}^{(t)}, \theta_k^{(t)}, d_k^\star)\right] \leq \varepsilon_{\text{bias},k},$$

where $d_k^{(t)} = d_\nu^{\pi_k^{(t)}}\pi_k^{(t)}$, $d_k^\star = d_\nu^{\pi_k^\star}\pi_k^\star$ and

$$L_k(w, \theta_k, d) = \mathbb{E}_{s_k,a_k\sim d}\left[\left(A_k^{\pi_{\theta_k}}(s_k, a_k) - w\cdot\nabla_{\theta_k}\log\pi_{\theta_k}(a_k|s_k)\right)^2\right],$$

$$w_{\star,k}^{(t)} \in \operatorname*{argmin}_{\|w\|_2\leq W} L_k(w, \theta_k^{(t)}, d_k^{(t)}).$$

The same result holds for the whole network:

$$\begin{aligned}\mathbb{E}\left[\min_{t<T}\{V^{\pi^\star}(\rho) - V^{(t)}(\rho)\}\right] &= \mathbb{E}\left[\min_{t<T}\left\{\frac{1}{K}\sum_{k\in\mathcal{K}}\left(V_k^{\pi^\star}(\rho) - V_k^{(t)}(\rho)\right)\right\}\right] \\ &\leq \mathbb{E}\left[\min_{t<T}\max_{k\in\mathcal{K}}\left\{V_k^{\pi^\star}(\rho) - V_k^{(t)}(\rho)\right\}\right].\end{aligned}$$

□

F Proof of Theorem 5.1

We follow the proof in Agarwal et al. [2020] modifying it where necessary. We start by proving a modified NPG Regret Lemma.

Lemma F.1. *Consider the setting of Theorem 5.1, then we have that:*

$$\mathbb{E}\left[\min_{t<T}\{V^{\pi^\star}(\rho) - V^{(t)}(\rho)\}\right] \leq \frac{W}{1-\gamma}\sqrt{\frac{2\delta\max_{k\in\mathcal{K}}\log|\mathcal{A}_k|}{T}} + \mathbb{E}\left[\frac{1}{T(1-\gamma)}\sum_{t=0}^{T-1} err_t\right],$$

where

$$err_t = \mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[A^{(t)}(s,a) - \frac{1}{K}\nabla_\theta\log\pi^{(t)}(a|s)\cdot w^{(t)}\right].$$

Proof. We assume $\log\pi_\theta(a|s)$ is a δ-smooth function of θ. By smoothness we have:

$$\begin{aligned}\log\frac{\pi^{(t+1)}(a|s)}{\pi^{(t)}(a|s)} &\geq \nabla_\theta\log\pi^{(t)}(a|s)\cdot(\theta^{(t+1)} - \theta^{(t)}) - \frac{\delta}{2}\left\|\theta^{(t+1)} - \theta^{(t)}\right\|_2^2 \\ &= \eta\nabla_\theta\log\pi^{(t)}(a|s)\cdot w^{(t)} - \eta^2\frac{\delta}{2}\left\|w^{(t)}\right\|_2^2.\end{aligned}$$

Then

$$\begin{aligned}&\frac{1}{K}\mathbb{E}_{s\sim d_\rho^{\pi^\star}}(\text{KL}(\pi_s^\star||\pi_s^{(t)}) - \text{KL}(\pi_s^\star||\pi_s^{(t+1)})) = \frac{1}{K}\mathbb{E}_{s\sim d^\star,a\sim\pi^\star(\cdot|s)}\left[\log\frac{\pi^{(t+1)}(a|s)}{\pi^{(t)}(a|s)}\right] \\ &\geq \frac{\eta}{K}\mathbb{E}_{s\sim d^\star,a\sim\pi^\star(\cdot|s)}\left[\nabla_\theta\log\pi^{(t)}(a|s)\cdot w^{(t)}\right] - \eta^2\frac{\delta}{2K}\left\|w^{(t)}\right\|_2^2\end{aligned}$$

$$
\begin{aligned}
&= \eta \mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[A^{(t)}(s,a)\right] + \eta \mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[\frac{1}{K}\nabla_\theta \log \pi^{(t)}(a|s)\cdot w^{(t)} - A^{(t)}(s,a)\right] \\
&\quad - \eta^2 \frac{\delta}{2K}\left\|w^{(t)}\right\|_2^2 \\
&= (1-\gamma)\eta\left(V^{\pi^\star}(\rho) - V^{(t)}(\rho)\right) - \eta^2\frac{\delta}{2K}\left\|w^{(t)}\right\|_2^2 - \eta\,\mathrm{err}_t,
\end{aligned}
$$

where

$$
\mathrm{err}_t = \mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[A^{(t)}(s,a) - \frac{1}{K}\nabla_\theta \log \pi^{(t)}(a|s)\cdot w^{(t)}\right].
$$

Rearranging

$$
V^{\pi^\star}(\rho) - V^{(t)}(\rho) \le \frac{1}{1-\gamma}\left(\frac{1}{\eta K}\mathbb{E}_{s\sim d_\rho^{\pi^\star}}(\mathrm{KL}(\pi_s^\star||\pi_s^{(t)}) - \mathrm{KL}(\pi_s^\star||\pi_s^{(t+1)})) + \frac{\eta\delta}{2}W^2 + \mathrm{err}_t\right)
$$

and summing over t

$$
\begin{aligned}
V^{\pi^\star}(\rho) - \frac{1}{T}\sum_{t=0}^{T-1} V^{(t)}(\rho) \le& \frac{1}{\eta K T(1-\gamma)}\sum_{t=0}^{T-1}\mathbb{E}_{s\sim d_\rho^{\pi^\star}}(\mathrm{KL}(\pi_s^\star||\pi_s^{(t)}) - \mathrm{KL}(\pi_s^\star||\pi_s^{(t+1)})) \\
&+ \frac{1}{T(1-\gamma)}\sum_{t=0}^{T-1}\left(\frac{\eta\delta}{2}W^2 + \mathrm{err}_t\right) \\
\le& \frac{\mathbb{E}_{s\sim d_\rho^{\pi^\star}}\mathrm{KL}(\pi_s^\star||\pi_s^{(0)})}{\eta K T(1-\gamma)} + \frac{\eta\delta W^2}{2(1-\gamma)} + \frac{1}{T(1-\gamma)}\sum_{t=0}^{T-1}\mathrm{err}_t \\
\le& \frac{\log|\mathcal{A}|}{\eta K T(1-\gamma)} + \frac{\eta\delta W^2}{2(1-\gamma)} + \frac{1}{T(1-\gamma)}\sum_{t=0}^{T-1}\mathrm{err}_t.
\end{aligned}
$$

Optimizing for η, we obtain the lemma. □

We can now prove Theorem 5.1

Proof of Theorem 5.1. After using the NPG Regret Lemma we want to bound the err_t term. We do so by dividing it in 3 parts. Let $(w^{(t)})_{t=1,\dots,T}$ be an update sequence such that, for each t, k, $(w_k^{(t)})_{N_{-k}^r} = 0$, then

$$
\begin{aligned}
\mathrm{err}_t &= \mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[A^{(t)}(s,a) - \frac{1}{K}\nabla_\theta \log \pi^{(t)}(a|s)\cdot w^{(t)}\right] \\
&= \frac{1}{K}\sum_{k\in\mathcal{K}}\mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[A_k^{(t)}(s,a) - \nabla_{\theta_k}\log\pi^{(t)}(a|s)\cdot w_k^{(t)}\right] \\
&= \frac{1}{K}\sum_{k\in\mathcal{K}}\mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[A_k^{(t)}(s,a) - \nabla_{(\theta_k)_{N_k^r}}\log\pi^{(t)}(a|s)\cdot (w_k^{(t)})_{N_k^r}\right] \\
&= \frac{1}{K}\sum_{k\in\mathcal{K}}\mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}}\log\pi^{(t)}(a|s)\cdot (w_{k,\star}^{(t)})_{N_k^r}\right] \\
&\quad + \frac{1}{K}\sum_{k\in\mathcal{K}}\mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[A_k^{(t)}(s,a) - \widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r})\right] \\
&\quad + \frac{1}{K}\sum_{k\in\mathcal{K}}\mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[\nabla_{(\theta_k)_{N_k^r}}\log\pi^{(t)}(a|s)\cdot((w_{k,\star}^{(t)})_{N_k^r} - (w_k^{(t)})_{N_k^r})\right],
\end{aligned}
$$

where $\forall k \in \mathcal{K}$

$$w_{k,\star}^{(t)} \in \underset{\substack{\left\|w\right\|_2 \leq W \\ w(N_{-k}^r)=0}}{\operatorname{argmin}} \mathbb{E}_{s\sim d^{(t)}, a\sim\pi^{(t)}(\cdot|s)} \left[\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a_k|s_{N_k^r}) \cdot w\right]^2.$$

We now analyse each term separately.

Term 1

$$\frac{1}{K}\sum_{k\in\mathcal{K}} \mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)} \left[\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s) \cdot (w_{k,\star}^{(t)})_{N_k^r}\right]$$
$$\leq \frac{1}{K}\sum_{k\in\mathcal{K}} \sqrt{\mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)} \left[\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s) \cdot (w_{k,\star}^{(t)})_{N_k^r}\right]^2} \leq \sqrt{\varepsilon_{\text{bias}}}$$

Term 2 Firstly, we have that $\forall k \in \mathcal{K}$

$$\widetilde{Q}_k^\pi(s_{N_k^r}, a_{N_k^r}) = \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t r_k(s_k(t), a_k(t)) \middle| \pi_{N_k^r}, s_{N_k^r}(0) = s_{N_k^r}, a_{N_k^r}(0) = a_{N_k^r}\right]$$

$$= \mathbb{E}\left[\mathbb{E}\left[\sum_{t=0}^\infty \gamma^t r_k(s_k(t), a_k(t)) \middle| \pi_{N_k^r}, s_{N_k^r}(0) = s_{N_k^r}, a_{N_k^r}(0) = a_{N_k^r}, s_{N_{-k}^r}(0), a_{N_{-k}^r}(0)\right]\right]$$
$$= \mathbb{E}\left[Q_k^\pi\left(s_{N_k^r}, a_{N_k^r}, s_{N_{-k}^r}(0), a_{N_{-k}^r}(0)\right)\right],$$

for some distribution of $s(0)_{N_{-k}^r}$ and $a(0)_{N_{-k}^r}$. Similarly $\forall k \in \mathcal{K}$

$$V_k^\pi(s) - \widetilde{V}_k^\pi(s_{N_k^r}) = \mathbb{E}\left[V_k^\pi(s) - V_k^\pi\left(s_{N_k^r}, s(0)_{N_{-k}^r}\right)\right].$$

So

$$A_k^\pi(s,a) - \widetilde{A}_k^\pi(s_{N_k^r}, a_{N_k^r}) = Q_k^\pi(s,a) - \widetilde{Q}_k^\pi(s_{N_k^r}, a_{N_k^r}) - V_k^\pi(s) + \widetilde{V}_k^\pi(s_{N_k^r}),$$

$$V_k^\pi(s) - \widetilde{V}_k^\pi(s_{N_k^r}) = \mathbb{E}\left[V_k^\pi(s) - V_k^\pi\left(s_{N_k^r}, s(0)_{N_{-k}^r}\right)\right]$$
$$\leq \mathbb{E}\left[c'\xi^{r+1}\right] = c'\xi^{r+1}$$

and

$$Q_k^\pi(s,a) - \widetilde{Q}_k^\pi(s_{N_k^r}, a_{N_k^r}) = \mathbb{E}\left[Q_k^\pi(s,a) - Q_k^\pi\left(s_{N_k^r}, a_{N_k^r}, s(0)_{N_{-k}^r}, a(0)_{N_{-k}^r}\right)\right]$$
$$\leq \mathbb{E}\left[c\rho^{r+1}\right] = c\rho^{r+1}.$$

Then $\forall k \in \mathcal{K}$

$$\left|A_k^\pi(s,a) - \widetilde{A}_k^\pi(s_{N_k^r}, a_{N_k^r})\right| \leq c\rho^{r+1} + c'\xi^{r+1}.$$

Term 3 In this paragraph, we denote, $\forall k \in \mathcal{K}$, $w_{k,\star}^{(t)} = (w_{k,\star}^{(t)})_{N_k^r}$ and $w_k^{(t)} = (w_k^{(t)})_{N_k^r}$ for clarity of exposition. Remember that

$$\Sigma_{\nu,k}^\theta = \mathbb{E}_{s,a\sim\nu}\left[\nabla_{(\theta_k)_{N_k^r}} \log \pi_\theta(a_k|s_{N_k^r}) (\nabla_{(\theta_k)_{N_k^r}} \log \pi_\theta(a_k|s_{N_k^r}))^\top\right]$$

and that we assume, $\forall k$,

$$\sup_{w\in\mathbb{R}^d} \frac{w^\top \Sigma_{d^\star,k}^{(t)} w}{w^\top \Sigma_{\nu,k}^{(t)} w} \leq \kappa'.$$

Then $\forall k \in \mathcal{K}$

$$\begin{aligned}
&\mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[\nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s)\cdot(w_{k,\star}^{(t)} - w_k^{(t)})\right] \\
&\le \sqrt{(w_{k,\star}^{(t)} - w_k^{(t)})^\top \Sigma_{d^\star,k}^{(t)}(w_{k,\star}^{(t)} - w_k^{(t)})} \\
&= \sqrt{\frac{(w_{k,\star}^{(t)} - w_k^{(t)})^\top \Sigma_{d^\star,k}^{(t)}(w_{k,\star}^{(t)} - w_k^{(t)})}{(w_{k,\star}^{(t)} - w_k^{(t)})^\top \Sigma_{\nu,k}^{(t)}(w_{k,\star}^{(t)} - w_k^{(t)})}(w_{k,\star}^{(t)} - w_k^{(t)})^\top \Sigma_{\nu,k}^{(t)}(w_{k,\star}^{(t)} - w_k^{(t)})} \\
&\le \sqrt{\kappa'(w_{k,\star}^{(t)} - w_k^{(t)})^\top \Sigma_{\nu,k}^{(t)}(w_{k,\star}^{(t)} - w_k^{(t)})} \\
&[\text{using that } (1-\gamma)\nu \le d_\nu^{\pi^{(t)}}] \\
&\le \sqrt{\frac{\kappa'}{1-\gamma}(w_{k,\star}^{(t)} - w_k^{(t)})^\top \Sigma_{d^{(t)}}^{(t)}(w_{k,\star}^{(t)} - w_k^{(t)})}.
\end{aligned}$$

Since $w_{k,\star}^{(t)}$ optimizes $\widetilde{L}_k(w, \theta^{(t)}, d^{(t)})$ the first order optimality condition implies that, $\forall w, \forall k \in \mathcal{K}$,

$$(w - w_{k,\star}^{(t)}) \cdot \nabla \widetilde{L}_k(w_{k,\star}^{(t)}, \theta^{(t)}, d^{(t)}) \ge 0.$$

So $\forall w, \forall k \in \mathcal{K}$,

$$\begin{aligned}
&\widetilde{L}_k(w, \theta^{(t)}, d^{(t)}) - \widetilde{L}_k(w_{k,\star}^{(t)}, \theta^{(t)}, d^{(t)}) \\
&= \mathbb{E}_{s\sim d^{(t)}, a\sim\pi^{(t)}(\cdot|s)}\Big[\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s)\cdot w \\
&\qquad + \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s)\cdot w_{k,\star}^{(t)} - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s)\cdot w_{k,\star}^{(t)}\Big]^2 \\
&\quad - \widetilde{L}_k(w_{k,\star}^{(t)}, \theta^{(t)}, d^{(t)}) \\
&= \mathbb{E}_{s\sim d^{(t)}, a\sim\pi^{(t)}(\cdot|s)}\left[\nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s)\cdot w_{k,\star}^{(t)} - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s)\cdot w\right]^2 \\
&+ 2(w - w_{k,\star}^{(t)})\mathbb{E}_{s\sim d^{(t)}, a\sim\pi^{(t)}(\cdot|s)}\left[\left(\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s)\cdot w_{k,\star}^{(t)}\right)\right. \\
&\qquad \left.\cdot \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s)\right] \\
&= (w_{k,\star}^{(t)} - w_k^{(t)})^\top \Sigma_{d^{(t)}}^{(t)}(w_{k,\star}^{(t)} - w_k^{(t)}) + (w - w_{k,\star}^{(t)}) \cdot \nabla \widetilde{L}_k(w_{k,\star}^{(t)}, \theta^{(t)}, d^{(t)}) \\
&\ge (w_{k,\star}^{(t)} - w_k^{(t)})^\top \Sigma_{d^{(t)}}^{(t)}(w_{k,\star}^{(t)} - w_k^{(t)}).
\end{aligned}$$

Therefore

$$\begin{aligned}
&\mathbb{E}\left[\mathbb{E}_{s\sim d^\star, a\sim\pi^\star(\cdot|s)}\left[\nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s)\cdot(w_{k,\star}^{(t)} - w_k^{(t)})\right]\right] \\
&\le \sqrt{\mathbb{E}\left[\frac{\kappa'}{1-\gamma}(\widetilde{L}_k(w, \theta^{(t)}, d^{(t)}) - \widetilde{L}_k(w_{k,\star}^{(t)}, \theta^{(t)}, d^{(t)}))\right]} \\
&= \sqrt{\mathbb{E}\left[\frac{\kappa'}{1-\gamma}\mathbb{E}\left[(\widetilde{L}_k(w, \theta^{(t)}, d^{(t)}) - \widetilde{L}_k(w_{k,\star}^{(t)}, \theta^{(t)}, d^{(t)}))\middle|\theta^{(t)}\right]\right]} \\
&\le \sqrt{\frac{\kappa'\varepsilon_{\text{stat}}}{1-\gamma}},
\end{aligned}$$

which completes the proof. □

G Bias analysis

Let $\widetilde{L}_k^r$ and $w_{k,\star}$ be defined as in Section 5. For every $k \in \mathcal{K}$, let

$$L_k(w, \theta, \nu) = \mathbb{E}_{s,a\sim\nu}\left[\left(A_k^{\pi_\theta}(s,a) - \nabla_{\theta_k} \log \pi_{\theta_k}(a_k|s)\cdot w\right)^2\right].$$

Let $\varepsilon_{\text{bias},r}$ be the smallest possible localized bias term associated to the range parameter r, i.e.

$$\varepsilon_{\text{bias},r} = \max_{k \in \mathcal{K}} \widetilde{L}_k^r(w_{k,\star}, \theta^{(t)}, d^\star),$$

which is the same assumption as in Theorem 9, but with an equality instead of an inequality. Let $\varepsilon_{\text{bias}}$ be the smallest possible non-localized bias term associated with an infinite range parameter, i.e. the case where we make no approximation, namely,

$$\varepsilon_{\text{bias}} = \max_{k \in \mathcal{K}} L_k(w'_{k,\star}, \theta^{(t)}, d^\star)$$

where

$$w'_{k,\star} \in \operatorname*{argmin}_{\|w\|_2 \leq W} L_k(w, \theta^{(t)}, d^{(t)}).$$

Proposition G.1. *Assume that*

$$\left\| \nabla_{\theta_k} \log \pi^{(t)}(a|s) \right\|_2 \leq B$$

and that

$$\left\| \nabla_{(\theta_k)_{N^r_{-k}}} \pi_\theta(a_k|s) \right\|_2 \leq \omega_r.$$

Then

$$\varepsilon_{bias,r} \leq \varepsilon_{bias} + e_r + 2\left(\frac{2}{1-\gamma} + WB\right)\sqrt{\frac{2\kappa' e_r}{1-\gamma}} + \frac{2\kappa' e_r}{1-\gamma},$$

where

$$e_r = \left(\frac{4}{1-\gamma} + 2WB\right) W\omega_r + \left(\frac{4}{1-\gamma} + 2WB\right)\left(c\psi^{r+1} + c'\phi^{r+1}\right).$$

The second assumption can be respected by choosing a policy belonging to the policy class described in Appendix B, as ω_r can be controlled by setting the parameters $\{\alpha_r\}$ small enough. In particular, it is possible to set it to be exponentially small in r. Therefore, the proposition shows that the localized bias is at most the bias associated with an infinite range parameter plus a quantity that goes to 0 exponentially fast in r. We present the proof of this result below.

Proof of Proposition G.1. To prove the proposition we need two intermediate results. For any w,θ and ν, we have that

$$\begin{aligned}
&\left|\widetilde{L}_k^r((w)_{N_k^r}, \theta, \nu) - L_k(w, \theta, \nu)\right| \\
&= \mathbb{E}_{s,a\sim\nu}\left[\left(\widetilde{A}_k^{\pi_\theta}(s_{N_k^r}, a_{N_k^r})\right)^2 - \left(A_k^{\pi_\theta}(s,a)\right)^2\right] \\
&\quad + \mathbb{E}_{s,a\sim\nu}\left[\left(\nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s) \cdot (w)_{N_k^r}\right)^2 - \left(\nabla_{\theta_k} \log \pi_{\theta_k}(a_k|s) \cdot w\right)^2\right]
\end{aligned}$$

$$\begin{aligned}
&+ 2\mathbb{E}_{s,a\sim\nu}\left[A_k^{\pi_\theta}(s,a)\nabla_{\theta_k} \log \pi_{\theta_k}(a_k|s) \cdot w - \widetilde{A}_k^{\pi_\theta}(s_{N_k^r}, a_{N_k^r})\nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s) \cdot (w)_{N_k^r}\right] \\
&\leq \frac{4}{1-\gamma}\left(c\psi^{r+1} + c'\phi^{r+1}\right) + 2W^2 B\omega_r + 2WB\left(c\psi^{r+1} + c'\phi^{r+1}\right) + \frac{4}{1-\gamma} W\omega_r \\
&= e_r.
\end{aligned}$$

Following the same passages in the proof of Theorem 5.1 in Appendix F, we have that

$$\begin{aligned}
&\max_{k\in\mathcal{K}} \mathbb{E}_{s,a\sim d^\star}\left[\nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s_{N_k^r})^\top \left((w'_{k,\star})_{N_k^r} - w_{k,\star}\right)\right] \\
&\leq \sqrt{\frac{\kappa'}{1-\gamma}(\widetilde{L}_k^r((w'_{k,\star})_{N_k^r}, \theta^{(t)}, d^{(t)}) - \widetilde{L}_k^r(w_{k,\star}, \theta^{(t)}, d^{(t)}))} \\
&\text{[using the first result]} \\
&\leq \sqrt{\frac{\kappa'}{1-\gamma}\left|L_k(w'_{k,\star}, \theta^{(t)}, d^{(t)}) - \widetilde{L}_k^r(w_{k,\star}, \theta^{(t)}, d^{(t)})\right| + \frac{\kappa' e_r}{1-\gamma}} \\
&= \sqrt{\frac{\kappa'}{1-\gamma}\left|\min_{\|w\|_2\leq W} L_k(w, \theta^{(t)}, d^{(t)}) - \min_{\|w\|_2\leq W} \widetilde{L}_k^r((w)_{N_k^r}, \theta^{(t)}, d^{(t)})\right| + \frac{\kappa' e_r}{1-\gamma}} \\
&\leq \sqrt{\frac{2\kappa' e_r}{1-\gamma}}.
\end{aligned}$$

We can now prove the proposition.

$$\begin{aligned}
\varepsilon_{\text{bias},r} &= \max_{k\in\mathcal{K}} \mathbb{E}_{s,a\sim d^\star}\left[\left(\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s_{N_k^r}) \cdot w_{k,\star}\right)^2\right] \\
&= \max_{k\in\mathcal{K}} \mathbb{E}_{s,a\sim d^\star}\Bigg[\left(\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s_{N_k^r}) \cdot (w'_{k,\star})_{N_k^r}\right)^2 \\
&\qquad + 2\left(\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s_{N_k^r}) \cdot (w'_{k,\star})_{N_k^r}\right) \\
&\qquad\quad \cdot \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s_{N_k^r})^\top \left((w'_{k,\star})_{N_k^r} - w_{k,\star}\right) \\
&\qquad + \left(\nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s_{N_k^r})^\top \left((w'_{k,\star})_{N_k^r} - w_{k,\star}\right)\right)^2\Bigg] \\
&\leq \max_{k\in\mathcal{K}} \mathbb{E}_{s,a\sim d^\star}\left[\left(\widetilde{A}_k^{(t)}(s_{N_k^r}, a_{N_k^r}) - \nabla_{(\theta_k)_{N_k^r}} \log \pi^{(t)}(a|s_{N_k^r}) \cdot (w'_{k,\star})_{N_k^r}\right)^2\right] \\
&\quad + 2\left(\frac{2}{1-\gamma} + WB\right)\sqrt{\frac{2\kappa' e_r}{1-\gamma}} + \frac{2\kappa' e_r}{1-\gamma} \\
&= \max_{k\in\mathcal{K}} \widetilde{L}_k^r((w'_{k,\star})_{N_k^r}, \theta^{(t)}, d^\star) + 2\left(\frac{2}{1-\gamma} + WB\right)\sqrt{\frac{2\kappa' e_r}{1-\gamma}} + \frac{2\kappa' e_r}{1-\gamma} \\
&\leq \varepsilon_{\text{bias}} + e_r + 2\left(\frac{2}{1-\gamma} + WB\right)\sqrt{\frac{2\kappa' e_r}{1-\gamma}} + \frac{2\kappa' e_r}{1-\gamma}.
\end{aligned}$$

□

3

A Novel Framework for Policy Mirror Descent with General Parameterization and Linear Convergence

A Novel Framework for Policy Mirror Descent with General Parameterization and Linear Convergence

Carlo Alfano
Department of Statistics
University of Oxford
carlo.alfano@stats.ox.ac.uk

Rui Yuan
Stellantis, France*
rui.yuan@stellantis.com

Patrick Rebeschini
Department of Statistics
University of Oxford
patrick.rebeschini@stats.ox.ac.uk

Abstract

Modern policy optimization methods in reinforcement learning, such as TRPO and PPO, owe their success to the use of parameterized policies. However, while theoretical guarantees have been established for this class of algorithms, especially in the tabular setting, the use of general parameterization schemes remains mostly unjustified. In this work, we introduce a framework for policy optimization based on mirror descent that naturally accommodates general parameterizations. The policy class induced by our scheme recovers known classes, e.g., softmax, and generates new ones depending on the choice of mirror map. Using our framework, we obtain the first result that guarantees linear convergence for a policy-gradient-based method involving general parameterization. To demonstrate the ability of our framework to accommodate general parameterization schemes, we provide its sample complexity when using shallow neural networks, show that it represents an improvement upon the previous best results, and empirically validate the effectiveness of our theoretical claims on classic control tasks.

1 Introduction

Policy optimization is one of the most widely-used classes of algorithms for reinforcement learning (RL). Among policy optimization techniques, policy gradient (PG) methods [e.g., 92, 87, 49, 6] are gradient-based algorithms that optimize the policy over a parameterized policy class and have emerged as a popular class of algorithms for RL [e.g., 42, 75, 12, 69, 78, 80, 54].

The design of gradient-based policy updates has been key to achieving empirical success in many settings, such as games [9] and autonomous driving [81]. In particular, a class of PG algorithms that has proven successful in practice consists of building updates that include a hard constraint (e.g., a trust region constraint) or a penalty term ensuring that the updated policy does not move too far from the previous one. Two examples of algorithms belonging to this category are trust region policy optimization (TRPO) [78], which imposes a Kullback-Leibler (KL) divergence [53] constraint on its updates, and policy mirror descent (PMD) [e.g. 88, 54, 93, 51, 89], which applies mirror descent (MD) [70] to RL. Shani et al. [82] propose a variant of TRPO that is actually a special case of PMD, thus linking TRPO and PMD.

From a theoretical perspective, motivated by the empirical success of PMD, there is now a concerted effort to develop convergence theories for PMD methods. For instance, it has been established that PMD converges linearly to the global optimum in the tabular setting by using a geometrically increasing step-size [54, 93], by adding entropy regularization [17], and more generally by adding convex regularization [100]. Linear convergence of PMD has also been established for the negative

*The work was done when the author was affiliated with Télécom Paris.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

entropy mirror map in the linear function approximation regime, i.e., for log-linear policies, either by adding entropy regularization [15], or by using a geometrically increasing step-size [20, 2, 98]. The proofs of these results are based on specific policy parameterizations, i.e., tabular and log-linear, while PMD remains mostly unjustified for general policy parameterizations and mirror maps, leaving out important practical cases such as neural networks. In particular, it remains to be seen whether the theoretical results obtained for tabular policy classes transfer to this more general setting.

In this work, we introduce Approximate Mirror Policy Optimization (AMPO), a novel framework designed to incorporate general parameterization into PMD in a theoretically sound manner. In summary, AMPO is an MD-based method that recovers PMD in different settings, such as tabular MDPs, is capable of generating new algorithms by varying the mirror map, and is amenable to theoretical analysis for any parameterization class. Since the MD update can be viewed as a two-step procedure, i.e., a gradient update step on the dual space and a mapping step onto the probability simplex, our starting point is to define the policy class based on this second MD step (Definition 3.1). This policy class recovers the softmax policy class as a special case (Example 3.2) and accommodates any parameterization class, such as tabular, linear, or neural network parameterizations. We then develop an update procedure for this policy class based on MD and PG.

We provide an analysis of AMPO and establish theoretical guarantees that hold for any parameterization class and any mirror map. More specifically, we show that our algorithm enjoys quasi-monotonic improvements (Proposition 4.2), sublinear convergence when the step-size is non-decreasing, and linear convergence when the step-size is geometrically increasing (Theorem 4.3). To the best of our knowledge, AMPO is the first policy-gradient-based method with linear convergence that can accommodate any parameterization class. Furthermore, the convergence rates hold for any choice of mirror map. The generality of our convergence results allows us not only to unify several current best-known results with specific policy parameterizations, i.e., tabular and log-linear, but also to achieve new state-of-the-art convergence rates with neural policies. Tables 1 and 2 in Appendix A.2 provide an overview of our results. We also refer to Appendix A.2 for a thorough literature review.

The key point of our analysis is Lemma 4.1, which allows us to keep track of the errors incurred by the algorithm (Proposition 4.2). It is an application of the three-point descent lemma by [19, Lemma 3.2] (see also Lemma F.2), which is possible thanks to our formulations of the policy class (Definition 3.1) and the policy update (Line 2 of Algorithm 1). The convergence rates of AMPO are obtained by building on Lemma 4.1 and leveraging modern PMD proof techniques [93].

In addition, we show that for a large class of mirror maps, i.e., the ω-potential mirror maps in Definition 3.5, AMPO can be implemented in $\widetilde{\mathcal{O}}(|\mathcal{A}|)$ computations. We give two examples of mirror maps belonging to this class, Examples 3.6 and 3.7, that illustrate the versatility of our framework. Lastly, we examine the important case of shallow neural network parameterization both theoretically and empirically. In this setting, we provide the sample complexity of AMPO, i.e., $\widetilde{\mathcal{O}}(\varepsilon^{-4})$ (Corollary 4.5), and show how it improves upon previous results.

2 Preliminaries

Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$ be a discounted Markov Decision Process (MDP), where $\mathcal{S}$ is a possibly infinite state space, $\mathcal{A}$ is a finite action space, $P(s' \mid s, a)$ is the transition probability from state s to s' under action a, $r(s, a) \in [0, 1]$ is a reward function, γ is a discount factor, and μ is a target state distribution. The behavior of an agent on an MDP is then modeled by a *policy* $\pi \in (\Delta(\mathcal{A}))^{\mathcal{S}}$, where $a \sim \pi(\cdot \mid s)$ is the density of the distribution over actions at state $s \in \mathcal{S}$, and $\Delta(\mathcal{A})$ is the probability simplex over $\mathcal{A}$.

Given a policy π, let $V^\pi : \mathcal{S} \to \mathbb{R}$ denote the associated *value function*. Letting s_t and a_t be the current state and action at time t, the value function V^π is defined as the expected discounted cumulative reward with the initial state $s_0 = s$, namely,

$$V^\pi(s) := \mathbb{E}_{a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot|s_t, a_t)} \Big[\sum\nolimits_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid \pi, s_0 = s \Big].$$

Now letting $V^\pi(\mu) := \mathbb{E}_{s \sim \mu}[V^\pi(s)]$, our objective is for the agent to find an optimal policy

$$\pi^\star \in \operatorname{argmax}_{\pi \in (\Delta(\mathcal{A}))^{\mathcal{S}}} V^\pi(\mu). \tag{1}$$

As with the value function, for each pair $(s,a) \in \mathcal{S} \times \mathcal{A}$, the state-action value function, or *Q-function*, associated with a policy π is defined as

$$Q^\pi(s,a) := \mathbb{E}_{a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot|s_t,a_t)} \Big[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid \pi, s_0 = s, a_0 = a \Big].$$

We also define the discounted state visitation distribution by

$$d_\mu^\pi(s) := (1-\gamma)\mathbb{E}_{s_0 \sim \mu} \Big[\sum_{t=0}^{\infty} \gamma^t P(s_t = s \mid \pi, s_0) \Big], \tag{2}$$

where $P(s_t = s \mid \pi, s_0)$ represents the probability of the agent being in state s at time t when following policy π and starting from s_0. The probability $d_\mu^\pi(s)$ represents the time spent on state s when following policy π.

The gradient of the value function $V^\pi(\mu)$ with respect to the policy is given by the policy gradient theorem (PGT) [87]:

$$\nabla_s V^\pi(\mu) := \frac{\partial V^\pi(\mu)}{\partial \pi(\cdot \mid s)} = \frac{1}{1-\gamma} d_\mu^\pi(s) Q^\pi(s,\cdot). \tag{3}$$

2.1 Mirror descent

The first tools we recall from the mirror descent (MD) framework are mirror maps and Bregman divergences [14, Chapter 4]. Let $\mathcal{Y} \subseteq \mathbb{R}^{|\mathcal{A}|}$ be a convex set. A *mirror map* $h : \mathcal{Y} \to \mathbb{R}$ is a strictly convex, continuously differentiable and essentially smooth function[1] such that $\nabla h(\mathcal{Y}) = \mathbb{R}^{|\mathcal{A}|}$. The convex conjugate of h, denoted by h^*, is given by

$$h^*(x^*) := \sup_{x \in \mathcal{Y}} \langle x^*, x \rangle - h(x), \quad x^* \in \mathbb{R}^{|\mathcal{A}|}.$$

The gradient of the mirror map $\nabla h : \mathcal{Y} \to \mathbb{R}^{|\mathcal{A}|}$ allows to map objects from the primal space $\mathcal{Y}$ to its dual space $\mathbb{R}^{|\mathcal{A}|}$, $x \mapsto \nabla h(x)$, and viceversa for ∇h^*, i.e., $x^* \mapsto \nabla h^*(x^*)$. In particular, from $\nabla h(\mathcal{Y}) = \mathbb{R}^{|\mathcal{A}|}$, we have: for all $(x, x^*) \in \mathcal{Y} \times \mathbb{R}^{|\mathcal{A}|}$,

$$x = \nabla h^*(\nabla h(x)) \quad \text{and} \quad x^* = \nabla h(\nabla h^*(x^*)). \tag{4}$$

Furthermore, the mirror map h induces a *Bregman divergence* [13, 18] , defined as

$$\mathcal{D}_h(x,y) := h(x) - h(y) - \langle \nabla h(y), x - y \rangle,$$

where $\mathcal{D}_h(x,y) \geq 0$ for all $x, y \in \mathcal{Y}$. We can now present the standard MD algorithm [70, 14]. Let $\mathcal{X} \subseteq \mathcal{Y}$ be a convex set and $V : \mathcal{X} \to \mathbb{R}$ be a differentiable function. The MD algorithm can be formalized[2] as the following iterative procedure in order to solve the minimization problem $\min_{x \in \mathcal{X}} V(x)$: for all $t \geq 0$,

$$y^{t+1} = \nabla h(x^t) - \eta_t \nabla V(x)|_{x=x^t}, \tag{5}$$

$$x^{t+1} = \text{Proj}_{\mathcal{X}}^h(\nabla h^*(y^{t+1})), \tag{6}$$

where η_t is set according to a step-size schedule $(\eta_t)_{t \geq 0}$ and $\text{Proj}_{\mathcal{X}}^h(\cdot)$ is the *Bregman projection*

$$\text{Proj}_{\mathcal{X}}^h(y) := \text{argmin}_{x \in \mathcal{X}} \mathcal{D}_h(x,y). \tag{7}$$

Precisely, at time t, $x^t \in \mathcal{X}$ is mapped to the dual space through $\nabla h(\cdot)$, where a gradient step is performed as in (5) to obtain y^{t+1}. The next step is to map y^{t+1} back in the primal space using $\nabla h^*(\cdot)$. In case $\nabla h^*(y^{t+1})$ does not belong to $\mathcal{X}$, it is projected as in (6).

3 Approximate Mirror Policy Optimization

The starting point of our novel framework is the introduction of a novel parameterized policy class based on the Bregman projection expression recalled in (7).

[1] h is essentially smooth if $\lim_{x \to \partial\mathcal{Y}} \|\nabla h(x)\|_2 = +\infty$, where $\partial\mathcal{Y}$ denotes the boundary of $\mathcal{Y}$.
[2] See a different formulation of MD in (11) and in Appendix B (Lemma B.1).

Algorithm 1: Approximate Mirror Policy Optimization

Input: Initial policy π^0, mirror map h, parameterization class $\mathcal{F}^\Theta$, iteration number T, step-size schedule $(\eta_t)_{t\geq 0}$, state-action distribution sequence $(v^t)_{t\geq 0}$.

for $t = 0$ ***to*** $T-1$ **do**

1 Obtain $\theta^{t+1} \in \Theta$ such that $\theta^{t+1} \in \operatorname{argmin}_{\theta\in\Theta} \left\| f^\theta - Q^t - \eta_t^{-1}\nabla h(\pi^t)\right\|^2_{L_2(v^t)}$.[4]

2 Update $\pi_s^{t+1} \in \operatorname*{argmin}_{p\in\Delta(\mathcal{A})} \mathcal{D}_h(p, \nabla h^*(\eta_t f_s^{\theta^{t+1}})) = \text{Proj}^h_{\Delta(\mathcal{A})}(\nabla h^*(\eta_t f_s^{t+1})),\ \forall s \in \mathcal{S}$.

Definition 3.1. Given a parameterized function class $\mathcal{F}^\Theta = \{f^\theta : \mathcal{S}\times\mathcal{A} \to \mathbb{R}, \theta \in \Theta\}$, a mirror map $h : \mathcal{Y} \to \mathbb{R}$, where $\mathcal{Y} \subseteq \mathbb{R}^{|\mathcal{A}|}$ is a convex set with $\Delta(\mathcal{A}) \subseteq \mathcal{Y}$, and $\eta > 0$, the *Bregman projected policy* class associated with $\mathcal{F}^\Theta$ and h consists of all the policies of the form:

$$\left\{\pi^\theta : \pi_s^\theta = \text{Proj}^h_{\Delta(\mathcal{A})}(\nabla h^*(\eta f_s^\theta)),\ s \in \mathcal{S};\ \theta \in \Theta\right\},$$

where for all $s \in \mathcal{S}$, $\pi_s^\theta, f_s^\theta \in \mathbb{R}^{|\mathcal{A}|}$ denote vectors $[\pi^\theta(a \mid s)]_{a\in\mathcal{A}}$ and $[f^\theta(s,a)]_{a\in\mathcal{A}}$, respectively.

In this definition, the policy is induced by a mirror map h and a parameterized function f^θ, and is obtained by mapping f^θ to $\mathcal{Y}$ with the operator $\nabla h^*(\cdot)$, which may not result in a well-defined probability distribution, and is thus projected on the probability simplex $\Delta(\mathcal{A})$. Note that the choice of h is key in deriving convenient expressions for π^θ. The Bregman projected policy class contains large families of policy classes. Below is an example of h that recovers a widely used policy class [7, Example 9.10].

Example 3.2 (Negative entropy)**.** If $\mathcal{Y} = \mathbb{R}_+^{|\mathcal{A}|}$ and h is the negative entropy mirror map, i.e., $h(\pi(\cdot|s)) = \sum_{a\in\mathcal{A}} \pi(a|s)\log(\pi(a|s))$, then the associated Bregman projected policy class becomes

$$\left\{\pi^\theta : \pi_s^\theta = \frac{\exp(\eta f_s^\theta)}{\|\exp(\eta f_s^\theta)\|_1},\ s \in \mathcal{S};\ \theta \in \Theta\right\}, \tag{8}$$

where the exponential and the fraction are element-wise and $\|\cdot\|_1$ is the ℓ_1 norm. In particular, when $f^\theta(s,a) = \theta_{s,a}$, the policy class (8) becomes the tabular softmax policy class; when f^θ is a linear function, (8) becomes the log-linear policy class; and when f^θ is a neural network, (8) becomes the neural policy class defined by Agarwal et al. [1]. We refer to Appendix C.1 for its derivation.

We now construct an MD-based algorithm to optimize V^{π^θ} over the Bregman projected policy class associated with a mirror map h and a parameterization class $\mathcal{F}^\Theta$ by adapting Section 2.1 to our setting. We define the following shorthand: at each time t, let $\pi^t := \pi^{\theta^t}$, $f^t := f^{\theta^t}$, $V^t := V^{\pi^t}$, $Q^t := Q^{\pi^t}$, and $d_\mu^t := d_\mu^{\pi^t}$. Further, for any function $y : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$ and distribution v over $\mathcal{S}\times\mathcal{A}$, let $y_s := y(s,\cdot) \in \mathbb{R}^{|\mathcal{A}|}$ and $\|y\|^2_{L_2(v)} = \mathbb{E}_{(s,a)\sim v}[(y(s,a))^2]$. Ideally, we would like to execute the exact MD algorithm: for all $t \geq 0$ and for all $s \in \mathcal{S}$,

$$f_s^{t+1} = \nabla h(\pi_s^t) + \eta_t(1-\gamma)\nabla_s V^t(\mu)/d_\mu^t(s) \overset{(3)}{=} \nabla h(\pi_s^t) + \eta_t Q_s^t,\ [3] \tag{9}$$

$$\pi_s^{t+1} = \text{Proj}^h_{\Delta(\mathcal{A})}(\nabla h^*(\eta_t f_s^{t+1})). \tag{10}$$

Here, (10) reflects our Bregman projected policy class 3.1. However, we usually cannot perform the update (9) exactly. In general, if f^θ belongs to a parameterized class $\mathcal{F}^\Theta$, there may not be any $\theta^{t+1} \in \Theta$ such that (9) is satisfied for all $s \in \mathcal{S}$.

To remedy this issue, we propose Approximate Mirror Policy Optimization (AMPO), described in Algorithm 1. At each iteration, AMPO consists in minimizing a surrogate loss and projecting the result onto the simplex to obtain the updated policy. In particular, the surrogate loss in Line 1 of Algorithm 1 is a standard regression problem where we try to approximate $Q^t + \eta_t^{-1}\nabla h(\pi^t)$ with f^{t+1}, and has been studied extensively when f^θ is a neural network [3, 35]. We can then readily use (10) to update π^{t+1} within the Bregman projected policy class defined in 3.1, which gives Line 2 of Algorithm 1.

To better illustrate the novelty of our framework, we provide below two remarks on how the two steps of AMPO relate and improve upon previous works.

[3] The update is (5) up to a scaling $(1-\gamma)/d_\mu^t(s)$ of η_t.

[4] With a slight abuse of notation, denote $\nabla h(\pi^t)(s,a)$ as $[\nabla h(\pi_s^t)]_a$.

Remark 3.3. Line 1 associates AMPO with the *compatible function approximation* framework developed by [87, 42, 1], as both frameworks define the updated parameters θ^{t+1} as the solution to a regression problem aimed at approximating the current Q-function Q^t. A crucial difference is that, Agarwal et al. [1] approximate Q^t linearly with respect to $\nabla_\theta \log \pi^t$ (see (61)), while in Line 1 we approximate Q^t and the gradient of the mirror map of the previous policy with any function f^θ. This generality allows our algorithm to achieve approximation guarantees for a wider range of assumptions on the structure of Q^t. Furthermore, the regression problem proposed by Agarwal et al. [1] depends on the distribution d_μ^t, while ours has no such constraint and allows off-policy updates involving an arbitrary distribution v^t. See Appendix E for more details.

Remark 3.4. Line 2 associates AMPO to previous approximations of PMD [89, 88]. For instance, Vaswani et al. [89] aim to maximize an expression equivalent to

$$\pi^{t+1} \in \operatorname{argmax}_{\pi^\theta \in \Pi(\Theta)} \mathbb{E}_{s\sim d_\mu^t}[\eta_t \langle Q_s^t, \pi_s^\theta\rangle - \mathcal{D}_h(\pi_s^\theta, \pi_s^t)], \tag{11}$$

where $\Pi(\Theta)$ is a given parameterized policy class, while the Bregman projection step of AMPO can be rewritten as

$$\pi_s^{t+1} \in \operatorname{argmax}_{p\in\Delta(\mathcal{A})} \langle \eta_t f_s^{t+1} - \nabla h(\pi_s^t), p\rangle - \mathcal{D}_h(p, \pi_s^t), \quad \forall s \in \mathcal{S}. \tag{12}$$

We formulate this result as Lemma F.1 in Appendix F with a proof. When the policy class $\Pi(\Theta)$ is the entire policy space $\Delta(\mathcal{A})^\mathcal{S}$, (11) is equivalent to the two-step procedure (9)-(10) thanks to the PGT (3). A derivation of this observation is given in Appendix B for completeness. The issue with the update in (11), which is overcome by (12), is that $\Pi(\Theta)$ in (11) is often a non-convex set, thus the three-point-descent lemma [19] cannot be applied. The policy update in (12) circumvents this problem by defining the policy implicitly through the Bregman projection, which is a convex problem and thus allows the application of the three-point-descent lemma [19]. We refer to Appendix F for the conditions of the three-point-descent lemma in details.

3.1 ω-potential mirror maps

In this section, we provide a class of mirror maps that allows to compute the Bregman projection in Line 2 with $\widetilde{\mathcal{O}}(|\mathcal{A}|)$ operations and simplifies the minimization problem in Line 1.

Definition 3.5 (ω-potential mirror map [50])**.** For $u \in (-\infty, +\infty]$, $\omega \leq 0$, let an *ω-potential* be an increasing C^1-diffeomorphism $\phi : (-\infty, u) \to (\omega, +\infty)$ such that

$$\lim_{x\to-\infty} \phi(x) = \omega, \qquad \lim_{x\to u} \phi(x) = +\infty, \qquad \int_0^1 \phi^{-1}(x)dx \leq \infty.$$

For any ω-potential ϕ, the associated mirror map h_ϕ, called *ω-potential mirror map*, is defined as

$$h_\phi(\pi_s) = \sum\nolimits_{a\in\mathcal{A}} \int_1^{\pi(a|s)} \phi^{-1}(x)dx.$$

Thanks to Krichene et al. [50, Proposition 2] (see Proposition C.1 as well), the policy π^{t+1} in Line 2 induced by the ω-potential mirror map can be obtained with $\widetilde{\mathcal{O}}(|\mathcal{A}|)$ computations and can be written as

$$\pi^{t+1}(a \mid s) = \sigma(\phi(\eta_t f^{t+1}(s,a) + \lambda_s^{t+1})), \quad \forall s \in \mathcal{S}, a \in \mathcal{A}, \tag{13}$$

where $\lambda_s^{t+1} \in \mathbb{R}$ is a normalization factor to ensure $\sum_{a\in\mathcal{A}} \pi^{t+1}(a \mid s) = 1$ for all $s \in \mathcal{S}$, and $\sigma(z) = \max(z, 0)$ for $z \in \mathbb{R}$. We call this policy class the *ω-potential policy* class. By using (13) and the definition of the ω-potential mirror map h_ϕ, the minimization problem in Line 1 is simplified to be

$$\theta^{t+1} \in \operatorname{argmin}_{\theta\in\Theta} \left\| f^\theta - Q^t - \eta_t^{-1} \max(\eta_{t-1} f^t, \phi^{-1}(0) - \lambda_s^t)\right\|_{L_2(v^t)}^2, \tag{14}$$

where $\max(\cdot,\cdot)$ is applied element-wisely. The ω-potential policy class allows AMPO to generate a wide range of applications by simply choosing an ω-potential ϕ. In fact, it recovers existing approaches to policy optimization, as we show in the next two examples.

Example 3.6 (Squared ℓ_2-norm)**.** If $\mathcal{Y} = \mathbb{R}^{|\mathcal{A}|}$ and ϕ is the identity function, then h_ϕ is the squared ℓ_2-norm, that is $h_\phi(\pi_s) = \|\pi_s\|_2^2/2$, and ∇h_ϕ is the identity function. So, Line 1 in Algorithm 1 becomes

$$\theta^{t+1} \in \operatorname{argmin}_{\theta\in\Theta} \left\| f^\theta - Q^t - \eta_t^{-1}\pi^t\right\|_{L_2(v^t)}^2. \tag{15}$$

The ∇h_ϕ^* also becomes the identity function, and the policy update is given for all $s \in \mathcal{S}$ by

$$\pi_s^{t+1} = \text{Proj}_{\Delta(\mathcal{A})}^{h_\phi}(\eta_t f_s^{t+1}) = \operatorname*{argmin}_{\pi_s \in \Delta(\mathcal{A})} \left\| \pi_s - \eta_t f_s^{t+1} \right\|_2^2, \tag{16}$$

which is the Euclidean projection on the probability simplex. In the tabular setting, where $\mathcal{S}$ and $\mathcal{A}$ are finite and $f^\theta(s,a) = \theta_{s,a}$, (15) can be solved exactly with the minimum equal to zero, and Equations (15) and (16) recover the projected Q-descent algorithm [93]. As a by-product, we generalize the projected Q-descent algorithm from the tabular setting to a general parameterization class $\mathcal{F}^\Theta$, which is a novel algorithm in the RL literature.

Example 3.7 (Negative entropy)**.** If $\mathcal{Y} = \mathbb{R}_+^{|\mathcal{A}|}$ and $\phi(x) = \exp(x-1)$, then h_ϕ is the negative entropy mirror map from Example 3.2 and Line 1 in Algorithm 1 becomes

$$\theta^{t+1} \in \operatorname{argmin}_{\theta \in \Theta} \left\| f^{t+1} - Q^t - \frac{\eta_{t-1}}{\eta_t} f^t \right\|_{L_2(v^t)}^2. \tag{17}$$

Consequently, based on Example 3.2, we have $\pi_s^{t+1} \propto \exp(\eta_t f_s^{t+1})$ for all $s \in \mathcal{S}$. In this example, AMPO recovers tabular NPG [82] when $f^\theta(s,a) = \theta_{s,a}$, and Q-NPG with log-linear policies [98] when f^θ and Q^t are linear functions for all $t \geq 0$.

We refer to Appendix C.2 for detailed derivations of the examples in this section and an efficient implementation of the Bregman projection step. In addition to the ℓ_2-norm and the negative entropy, several other mirror maps that have been studied in the optimization literature fall into the class of ω-potential mirror maps, such as the Tsallis entropy [74, 57] and the hyperbolic entropy [34], as well as a generalization of the negative entropy [50]. These examples illustrate how the class of ω-potential mirror maps recovers known methods and can be used to explore new algorithms in policy optimization. We leave the study of the application of these mirror maps in RL as future work, both from an empirical and theoretical point of view, and provide additional discussion and details in Appendix C.2.

4 Theoretical analysis

In our upcoming theoretical analysis of AMPO, we rely on the following key lemma.

Lemma 4.1. *For any policies π and $\bar{\pi}$, for any function $f^\theta \in \mathcal{F}^\Theta$ and for $\eta > 0$, we have*

$$\langle \eta f_s^\theta - \nabla h(\bar{\pi}_s), \pi_s - \tilde{\pi}_s \rangle \leq \mathcal{D}_h(\pi_s, \bar{\pi}_s) - \mathcal{D}_h(\tilde{\pi}_s, \bar{\pi}_s) - \mathcal{D}_h(\pi_s, \tilde{\pi}_s), \quad \forall s \in \mathcal{S},$$

where $\tilde{\pi}$ is the Bregman projected policy induced by f^θ and h according to Definition 3.1, that is $\tilde{\pi}_s = \operatorname{argmin}_{p \in \Delta(\mathcal{A})} \mathcal{D}_h(p, \nabla h^*(\eta f_s^\theta))$ for all $s \in \mathcal{S}$.

The proof of Lemma 4.1 is given in Appendix D.1. Lemma 4.1 describes a relation between any two policies and a policy belonging to the Bregman projected policy class associated with $\mathcal{F}^\Theta$ and h. As mentioned in Remark 3.4, Lemma 4.1 is the direct consequence of (12) and can be interpreted as an application of the three-point descent lemma [19], while it cannot be applied to algorithms based on the update in (11) [88, 89] due to the non-convexity of the optimization problem (see also Appendix F). Notice that Lemma 4.1 accommodates naturally with general parameterization also thanks to (12). In contrast, similar results have been obtained and exploited for specific policy and mirror map classes [93, 60, 36, 98], while our result allows any parameterization class $\mathcal{F}^\Theta$ and any choice of mirror map, thus greatly expanding the scope of applications of the lemma. A similar result for general parameterization has been obtained by Lan [55, Proposition 3.5] in the setting of strongly convex mirror maps.

Lemma 4.1 becomes useful when we set $\bar{\pi} = \pi^t$, $f^\theta = f^{t+1}$, $\eta = \eta_t$ and $\pi = \pi^t$ or $\pi = \pi^\star$. In particular, when $\eta_t f_s^{t+1} - \nabla h(\pi_s^t) \approx \eta_t Q_s^\pi$, Lemma 4.1 allows us to obtain telescopic sums and recursive relations, and to handle error terms efficiently, as we show in Appendix D.

4.1 Convergence for general policy parameterization

In this section, we consider the parameterization class $\mathcal{F}^\Theta$ and the fixed but arbitrary mirror map h. We show that AMPO enjoys quasi-monotonic improvement and sublinear or linear convergence, depending on the step-size schedule. The first step is to control the approximation error of AMPO.

(A1) (*Approximation error*). There exists $\varepsilon_{\text{approx}} \geq 0$ such that, for all times $t \geq 0$,

$$\mathbb{E}\big[\, \big\| f^{t+1} - Q^t - \eta_t^{-1} \nabla h(\pi^t) \big\|^2_{L_2(v^t)} \,\big] \leq \varepsilon_{\text{approx}},$$

where $(v^t)_{t \geq 0}$ is a sequence of distributions over states and actions and the expectation is taken over the randomness of the algorithm that obtains f^{t+1}.

Assumption (A1) is common in the conventional compatible function approximation approach[5] [1]. It characterizes the loss incurred by Algorithm 1 in solving the regression problem in Line 1. When the step-size η_t is sufficiently large, Assumption (A1) measures how well f^{t+1} approximates the current Q-function Q^t. Hence, $\varepsilon_{\text{approx}}$ depends on both the accuracy of the policy evaluation method used to obtain an estimate of Q^t [86, 79, 28] and the error incurred by the function $f^\theta \in \mathcal{F}^\Theta$ that best approximates Q^t, that is the representation power of $\mathcal{F}^\Theta$. Later in Section 4.2, we show how to solve the minimization problem in Line 1 when $\mathcal{F}^\Theta$ is a class of shallow neural networks so that Assumption (A1) holds. We highlight that Assumption (A1) is weaker than the conventional assumptions [1, 98], since we do not constrain the minimization problem to be linear in the parameters (see (61)). We refer to Appendix A.2 for a discussion on its technical novelty and Appendix G for a relaxed version of the assumption.

As mentioned in Remark 3.3, the distribution v^t does not depend on the current policy π^t for all times $t \geq 0$. Thus, Assumption (A1) allows for off-policy settings and the use of replay buffers [68]. We refer to Appendix A.3 for details. To quantify how the choice of these distributions affects the error terms in the convergence rates, we introduce the following coefficient.

(A2) (*Concentrability coefficient*). There exists $C_v \geq 0$ such that, for all times t,

$$\mathbb{E}_{(s,a) \sim v^t} \left[\left(\frac{d_\mu^\pi(s) \pi(a \mid s)}{v^t(s,a)} \right)^2 \right] \leq C_v,$$

whenever (d_μ^π, π) is either $(d_\mu^\star, \pi^\star)$, (d_μ^{t+1}, π^{t+1}), $(d_\mu^\star, \pi^t)$, or (d_μ^{t+1}, π^t).

The concentrability coefficient C_v quantifies how much the distribution v^t overlaps with the distributions $(d_\mu^\star, \pi^\star)$, (d_μ^{t+1}, π^{t+1}), $(d_\mu^\star, \pi^t)$ and (d_μ^{t+1}, π^t). It highlights that the distribution v^t should have full support over the environment, in order to avoid large values of C_v. Assumption (A2) is weaker than the previous best-known concentrability coefficient [98, Assumption 9], in the sense that we have the full control over v^t. We refer to Appendix H for a more detailed discussion. We can now present our first result on the performance of Algorithm 1.

Proposition 4.2 (Quasi-monotonic updates)**.** *Let (A1), (A2) be true. We have, for all $t \geq 0$,*

$$\mathbb{E}\left[V^{t+1}(\mu) - V^t(\mu)\right] \geq \mathbb{E}\left[\mathbb{E}_{s \sim d_\mu^{t+1}}\left[\frac{\mathcal{D}_h(\pi_s^{t+1}, \pi_s^t) + \mathcal{D}_h(\pi_s^t, \pi_s^{t+1})}{\eta_t(1-\gamma)}\right]\right] - \frac{2\sqrt{C_v \varepsilon_{\text{approx}}}}{1-\gamma},$$

where the expectation is taken over the randomness of AMPO.

We refer to Appendix D.3 for the proof. Proposition 4.2 ensures that an update of Algorithm 1 cannot lead to a performance degradation, up to an error term. The next assumption concerns the coverage of the state space for the agent at each time t.

(A3) (*Distribution mismatch coefficient*). Let $d_\mu^\star := d_\mu^{\pi^\star}$. There exists $\nu_\mu \geq 0$ such that

$$\sup_{s \in \mathcal{S}} \frac{d_\mu^\star(s)}{d_\mu^t(s)} \leq \nu_\mu, \quad \text{for all times } t \geq 0.$$

Since $d_\mu^t(s) \geq (1-\gamma)\mu(s)$ for all $s \in \mathcal{S}$, obtained from the definition of d_μ in (2), we have that

$$\sup_{s \in \mathcal{S}} \frac{d_\mu^\star(s)}{d_\mu^t(s)} \leq \frac{1}{1-\gamma} \sup_{s \in \mathcal{S}} \frac{d_\mu^\star(s)}{\mu(s)},$$

where assuming boundedness for the term on the right-hand side is standard in the literature on both the PG [e.g., 101, 90] and NPG convergence analysis [e.g., 1, 15, 93]. We refer to Appendix I for details. It is worth mentioning that the quasi-monotonic improvement in Proposition 4.2 holds without (A3).

We define the weighted Bregman divergence between the optimal policy $\pi^\star$ and the initial policy π^0 as $\mathcal{D}_0^\star = \mathbb{E}_{s \sim d_\mu^\star}[\mathcal{D}_h(\pi_s^\star, \pi_s^0)]$. We then have our main results below.

[5] An extended discussion of this approach is provided in Appendix G.

Theorem 4.3. *Let (A1), (A2) and (A3) be true. If the step-size schedule is non-decreasing, i.e., $\eta_t \leq \eta_{t+1}$ for all $t \geq 0$, the iterates of Algorithm 1 satisfy: for every $T \geq 0$,*

$$V^\star(\mu) - \frac{1}{T}\sum_{t<T} \mathbb{E}\left[V^t(\mu)\right] \leq \frac{1}{T}\left(\frac{\mathcal{D}_0^\star}{(1-\gamma)\eta_0} + \frac{\nu_\mu}{1-\gamma}\right) + \frac{2(1+\nu_\mu)\sqrt{C_v \varepsilon_{\text{approx}}}}{1-\gamma}.$$

Furthermore, if the step-size schedule is geometrically increasing, i.e., satisfies

$$\eta_{t+1} \geq \frac{\nu_\mu}{\nu_\mu - 1}\eta_t \qquad \forall t \geq 0, \tag{18}$$

we have: for every $T \geq 0$,

$$V^\star(\mu) - \mathbb{E}\left[V^T(\mu)\right] \leq \frac{1}{1-\gamma}\left(1 - \frac{1}{\nu_\mu}\right)^T \left(1 + \frac{\mathcal{D}_0^\star}{\eta_0(\nu_\mu - 1)}\right) + \frac{2(1+\nu_\mu)\sqrt{C_v \varepsilon_{\text{approx}}}}{1-\gamma}.$$

Theorem 4.3 is, to the best of our knowledge, the first result that establishes linear convergence for a PG-based method involving general policy parameterization. For the same setting, it also matches the previous best known $O(1/T)$ convergence [55], without requiring regularization. Lastly, Theorem 4.3 provides a convergence rate for a PMD-based algorithm that allows for arbitrary mirror maps and policy parameterization without requiring the assumption on the approximation error to hold in ℓ_∞-norm, in contrast to Lan [55]. We give here a brief discussion of Theorem 4.3 w.r.t. previous results and refer to Tables 1 and 2 in Appendix A.2 for a detailed comparison.

In terms of iteration complexity, Theorem 4.3 recovers the best-known convergence rates in the tabular setting [93], for both non-decreasing and exponentially increasing step-size schedules. While considering a more general setting, Theorem 4.3 matches or improves upon the convergence rate of previous work on policy gradient methods for non-tabular policy parameterizations that consider constant step-size schedules [60, 82, 61, 90, 1, 89, 16, 55], and matches the convergence speed of previous work that employ NPG, log-linear policies, and geometrically increasing step-size schedules [2, 98].

In terms of generality, the results in Theorem 4.3 hold without the need to implement regularization [17, 100, 15, 16, 54], to impose bounded updates or smoothness of the policy [1, 61], to restrict the analysis to the case where the mirror map h is the negative entropy [60, 36], or to make ℓ_∞-norm assumptions on the approximation error [55]. We improve upon the latest results for PMD with general policy parameterization by Vaswani et al. [89], which only allow bounded step-sizes, where the bound can be particularly small, e.g., $(1-\gamma)^3/(2\gamma|\mathcal{A}|)$, and can slow down the learning process.

When $\mathcal{S}$ is a finite state space, a sufficient condition for ν_μ in (A3) to be bounded is requiring μ to have full support on $\mathcal{S}$. If μ does not have full support, one can still obtain linear convergence for $V^\star(\mu') - V^T(\mu')$, for an arbitrary state distribution μ' with full support, and relate this quantity to $V^\star(\mu) - V^T(\mu)$. We refer to Appendix I for a detailed discussion on the distribution mismatch coefficient.

Intuition. An interpretation of our theory can be provided by connecting AMPO to the Policy Iteration algorithm (PI), which also enjoys linear convergence. To see this, first recall (12)

$$\pi_s^{t+1} \in \text{argmin}_{p \in \Delta(\mathcal{A})} \langle -f_s^{t+1} + \eta_t^{-1}\nabla h(\pi_s^t), p\rangle + \eta_t^{-1}\mathcal{D}_h(p, \pi_s^t), \quad \forall s \in \mathcal{S}.$$

Secondly, solving Line 1 of Algorithm 1 leads to $f_s^{t+1} - \eta_t^{-1}\nabla h(\pi_s^t) \approx Q_s^t$. When the step-size $\eta_t \to \infty$, that is $\eta_t^{-1} \to 0$, the above viewpoint of the AMPO policy update becomes

$$\pi_s^{t+1} \in \text{argmin}_{p \in \Delta(\mathcal{A})} \langle -Q_s^t, p\rangle \iff \pi_s^{t+1} \in \text{argmax}_{p \in \Delta(\mathcal{A})} \langle Q_s^t, p\rangle, \quad \forall s \in \mathcal{S},$$

which is the PI algorithm. Here we ignore the Bregman divergence term $\mathcal{D}_h(\pi, \pi_s^t)$, as it is multiplied by $1/\eta_t$, which goes to 0. So AMPO behaves more and more like PI with a large enough step-size and thus is able to converge linearly like PI.

Proof idea. We provide a sketch of the proof here; the full proof is given in Appendix D. In a nutshell, the convergence rates of AMPO are obtained by building on Lemma 4.1 and leveraging modern PMD proof techniques [93]. Following the conventional compatible function approximation approach [1], the idea is to write the global optimum convergence results in an additive form, that is

$$\text{sub-optimality gap} \leq \text{optimization error} + \text{approximation error}.$$

The separation between the two errors is allowed by Lemma 4.1, while the optimization error is bounded through the PMD proof techniques from Xiao [93] and the approximation error is characterized by Assumption (A1). Overall, the proof consists of three main steps.

Step 1. Using Lemma 4.1 with $\bar{\pi} = \pi^t$, $f^\theta = f^{t+1}$, $\eta = \eta_t$, $\tilde{\pi} = \pi^{t+1}$, and $\pi_s = \pi_s^t$, we obtain

$$\langle \eta_t f_s^{t+1} - \nabla h(\pi_s^t), \pi_s^{t+1} - \pi_s^t \rangle \geq 0,$$

which characterizes the improvement of the updated policy.

Step 2. Assumption (A1), Step 1, the performance difference lemma (Lemma D.4), and Lemma 4.1 with $\bar{\pi} = \pi^t$, $f^\theta = f^{t+1}$, $\eta = \eta_t$, $\tilde{\pi} = \pi^{t+1}$, and $\pi_s = \pi_s^\star$ permit us to obtain the following.

Proposition 4.4. *Let $\Delta_t := V^\star(\mu) - V^t(\mu)$. For all $t \geq 0$, we have*

$$\mathbb{E}\left[\nu_\mu \left(\Delta_{t+1} - \Delta_t\right) + \Delta_t\right] \leq \mathbb{E}\left[\frac{\mathbb{E}_{s\sim d_\mu^\star}\left[\mathcal{D}_h(\pi_s^\star, \pi_s^t)\right]}{(1-\gamma)\eta_t} - \frac{\mathbb{E}_{s\sim d_\mu^\star}\left[\mathcal{D}_h(\pi_s^\star, \pi_s^{t+1})\right]}{(1-\gamma)\eta_t}\right] + (1+\nu_\mu)\frac{2\sqrt{C_v \varepsilon_{\text{approx}}}}{1-\gamma}.$$

Step 3. Proposition 4.4 leads to sublinear convergence using a telescoping sum argument, and to linear convergence by properly defining step-sizes and by rearranging terms into the following contraction,

$$\mathbb{E}\left[\Delta_{t+1} + \frac{\mathbb{E}_{s\sim d_\mu^\star}\left[\mathcal{D}_h(\pi_s^\star, \pi_s^{t+1})\right]}{(1-\gamma)\eta_{t+1}(\nu_\mu - 1)}\right] \leq \left(1 - \frac{1}{\nu_\mu}\right)\mathbb{E}\left[\Delta_t + \frac{\mathbb{E}_{s\sim d_\mu^\star}\left[\mathcal{D}_h(\pi_s^\star, \pi_s^t)\right]}{(1-\gamma)\eta_t(\nu_\mu - 1)}\right] + \left(1 + \frac{1}{\nu_\mu}\right)\frac{2\sqrt{C_v \varepsilon_{\text{approx}}}}{1-\gamma}.$$

4.2 Sample complexity for neural network parameterization

Neural networks are widely used in RL due to their empirical success in applications [67, 68, 84]. However, few theoretical guarantees exist for using this parameterization class in policy optimization [60, 90, 16]. Here, we show how we can use our framework and Theorem 4.3 to fill this gap by deriving a sample complexity result for AMPO when using neural network parameterization. We consider the case where the parameterization class $\mathcal{F}^\Theta$ from Definition 3.1 belongs to the family of shallow ReLU networks, which have been shown to be universal approximators [38, 4, 27, 39]. That is, for $(s,a) \in (\mathcal{S} \times \mathcal{A}) \subseteq \mathbb{R}^d$, define $f^\theta(s,a) = c^\top \sigma(W(s,a) + b)$ with $\theta = (c, W, b)$, where $\sigma(y) = \max(y, 0)$ for all $y \in \mathbb{R}$ is the ReLU activation function and is applied element-wisely, $c \in \mathbb{R}^m$, $W \in \mathbb{R}^{m\times d}$ and $b \in \mathbb{R}^m$.

At each iteration t of AMPO, we set $v^t = d_\mu^t$ and solve the regression problem in Line 1 of Algorithm 1 through stochastic gradient descent (SGD). In particular, we initialize entry-wise W_0 and b as i.i.d. random Gaussian variables from $\mathcal{N}(0, 1/m)$, and c as i.i.d. random Gaussian variables from $\mathcal{N}(0, \epsilon_A)$ with $\epsilon_A \in (0, 1]$. Assuming access to a simulator for the distribution v^t, we run SGD for K steps on the matrix W, that is, for $k = 0, \ldots, K-1$,

$$W_{k+1} = W_k - \alpha\big(f^{(k)}(s,a) - \widehat{Q}^t(s,a) - \eta_t^{-1}\nabla h(\pi_s^t)\big)\nabla_W f^{(k)}(s,a), \tag{19}$$

where $f^{(k)}(s,a) = c^\top \sigma((W_0 + W_k)(s,a) + b)$, $(s,a) \sim v^t$ and $\widehat{Q}^t(s,a)$ is an unbiased estimate of $Q^t(s,a)$ obtained through Algorithm 4. We can then present our result on the sample complexity of AMPO for neural network parameterization, which is based on our convergence Theorem 4.3 and an analysis of neural networks by Allen-Zhu et al. [3, Theorem 1].

Corollary 4.5. *In the setting of Theorem 4.3, let the parameterization class $\mathcal{F}^\Theta$ consist of sufficiently wide shallow ReLU neural networks. Using an exponentially increasing step-size and solving the minimization problem in Line 1 with SGD as in (19), the number of samples required by AMPO to find an ε-optimal policy with high probability is $\widetilde{\mathcal{O}}(C_v^2 \nu_\mu^5 / \varepsilon^4 (1-\gamma)^6)$, where ε has to be larger than a fixed and non-vanishing error floor.*

We provide a proof of Corollary 4.5 and an explicit expression for the error floor in Appendix J. Note that the sample complexity in Corollary 4.5 might be impacted by an additional $\texttt{poly}(\varepsilon^{-1})$ term. We refer to Appendix J for more details and an alternative result (Corollary J.4) which does not include an additional $\texttt{poly}(\varepsilon^{-1})$ term, enabling comparison with prior works.

5 Numerical experiments

We provide an empirical evaluation of AMPO in order to validate our theoretical findings. We note that the scope of this work is mainly theoretical and that we do not aim at establishing state-of-the-art results in the setting of deep RL. Our implementation is based upon the PPO implementation

from PureJaxRL [63], which obtains the estimates of the Q-function through generalized advantage estimation (GAE) [79] and performs the policy update using ADAM optimizer [48] and mini-batches. To implement AMPO, we (i) replaced the PPO loss with the expression to minimize in Equation (14), (ii) replaced the softmax projection with the Bregman projection, (iii) saved the constants λ along the sampled trajectories in order to compute Equation (14). The code is available here.

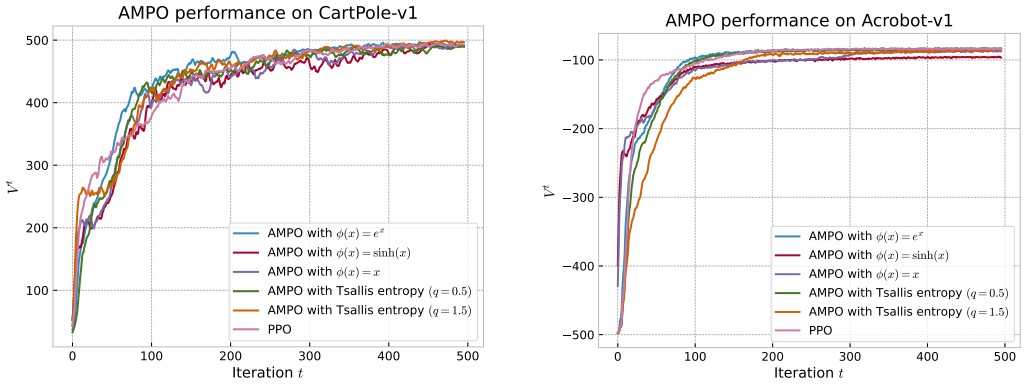

Figure 1: Averaged performance over 50 runs of AMPO in CartPole and Acrobot environments. Note that the maximum values for CartPole and Acrobot are 500 and -80, respectively.

In Figure 1, we show the averaged performance over 100 runs of AMPO in two classic control environments, i.e. CartPole and Acrobot, in the setting of ω-potential mirror maps. In particular, we choose: $\phi(x) = e^x$, which corresponds to the negative entropy (Example 3.7); $\phi(x) = \sinh(x)$, which corresponds to the hyperbolic entropy [34, see also (41)]; $\phi(x) = x$, which corresponds to the Euclidean norm (Example 3.6); and the Tsallis entropy for two values of the entropic index q [74, 57, see also (40)]. We refer to Appendix C.2 for a detailed discussion on these mirror maps. We set the step-size to be constant and of value 1. For a comparison, we also plot the averaged performance over 100 runs of PPO.

The plots in Figure 1 confirm our results on the quasi-monotonicity of the updates of AMPO and on its convergence to the optimal policy. We observe that instances of AMPO with different mirror maps are very competitive as compared to PPO. We also note that, despite the convergence rates in Theorem 4.3 depend on the mirror map only in terms of a $\mathcal{D}_0^\star$ term, different mirror maps may result in different convergence speeds and error floors in practice. In particular, our experiments suggest that the negative entropy mirror map may not be the best choice for AMPO, and that exploring different mirror maps is a promising direction of research.

6 Conclusion

We have introduced a novel framework for RL which, given a mirror map and any parameterization class, induces a policy class and an update rule. We have proven that this framework enjoys sublinear and linear convergence for non-decreasing and geometrically increasing step-size schedules, respectively. Future venues of investigation include studying the sample complexity of AMPO in on-policy and off-policy settings other than neural network parameterization, exploiting the properties of specific mirror maps to take advantage of the structure of the MDP and efficiently including representation learning in the algorithm. We refer to Appendix A.3 for a thorough discussion of future work. We believe that the main contribution of AMPO is to provide a general framework with theoretical guarantees that can help the analysis of specific algorithms and MDP structures. AMPO recovers and improves several convergence rate guarantees in the literature, but it is important to keep in consideration how previous works have exploited particular settings, while AMPO tackles the most general case. It will be interesting to see whether these previous works combined with our fast linear convergence result can derive new efficient sample complexity results.

Acknowledgments and Disclosure of Funding

We thank the anonymous reviewers for their helpful comments. Carlo Alfano was supported by the Engineering and Physical Sciences Research Council and thanks G-Research for partly funding attendance to NeurIPS 2023.

References

[1] Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 2021. (Cited on pages 4, 5, 7, 8, 20, 22, 24, 28, 37, 38, 39, and 41.)

[2] Carlo Alfano and Patrick Rebeschini. Linear convergence for natural policy gradient with log-linear policy parametrization. *arXiv preprint arXiv:2209.15382*, 2022. (Cited on pages 2, 8, 20, 21, 22, 25, 28, 38, and 39.)

[3] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *Advances in Neural Information Processing Systems*, 2019. (Cited on pages 4, 9, 40, and 41.)

[4] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, 2019. (Cited on pages 9 and 43.)

[5] Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 1998. (Cited on page 21.)

[6] Jonathan Baxter and Peter L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 2001. (Cited on page 1.)

[7] Amir Beck. *First-Order Methods in Optimization*. SIAM-Society for Industrial and Applied Mathematics, 2017. (Cited on pages 4 and 23.)

[8] Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 2003. (Cited on pages 21 and 26.)

[9] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019. (Cited on page 1.)

[10] Jalaj Bhandari and Daniel Russo. Global optimality guarantees for policy gradient methods. *arXiv preprint arXiv:1906.01786*, 2019. (Cited on page 23.)

[11] Jalaj Bhandari and Daniel Russo. On the linear convergence of policy gradient methods for finite MDPs. In *International Conference on Artificial Intelligence and Statistics*, 2021. (Cited on pages 22, 23, and 25.)

[12] Shalabh Bhatnagar, Richard S. Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor-critic algorithms. *Automatica*, 2009. (Cited on pages 1 and 22.)

[13] Lev M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 1967. (Cited on page 3.)

[14] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 2015. (Cited on pages 3, 25, 31, and 32.)

[15] Semih Cayci, Niao He, and R Srikant. Linear convergence of entropy-regularized natural policy gradient with linear function approximation. *arXiv preprint arXiv:2106.04096*, 2021. (Cited on pages 2, 7, 8, 22, 25, 38, and 39.)

[16] Semih Cayci, Niao He, and R Srikant. Finite-time analysis of entropy-regularized neural natural actor-critic algorithm. *arXiv preprint arXiv:2206.00833*, 2022. (Cited on pages 8, 9, 20, 22, 24, 39, 43, and 44.)

[17] Shicong Cen, Chen Cheng, Yuxin Chen, Yuting Wei, and Yuejie Chi. Fast global convergence of natural policy gradient methods with entropy regularization. *Operations Research*, 2021. (Cited on pages 1, 8, 22, 25, and 39.)

[18] Yair Censor and Stavros A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, USA, 1997. (Cited on page 3.)

[19] Gong Chen and Marc Teboulle. Convergence analysis of a proximal-like minimization algorithm using bregman functions. *SIAM Journal on Optimization*, 1993. (Cited on pages 2, 5, 6, 21, 31, and 37.)

[20] Zaiwei Chen and Siva Theja Maguluri. Sample complexity of policy-based methods under off-policy sampling and linear function approximation. In *International Conference on Artificial Intelligence and Statistics*, 2022. (Cited on pages 2, 22, 25, and 39.)

[21] Zaiwei Chen, Sajad Khodadadian, and Siva Theja Maguluri. Finite-sample analysis of off-policy natural actor-critic with linear function approximation. *IEEE Control Systems Letters*, 2022. (Cited on page 38.)

[22] Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex sgd. In *Advances in Neural Information Processing Systems*, 2019. (Cited on pages 22 and 23.)

[23] Dongsheng Ding, Kaiqing Zhang, Tamer Basar, and Mihailo Jovanovic. Natural policy gradient primal-dual method for constrained markov decision processes. In *Advances in Neural Information Processing Systems*, 2020. (Cited on page 23.)

[24] Yuhao Ding, Junzi Zhang, and Javad Lavaei. On the global optimum convergence of momentum-based policy gradient. In *International Conference on Artificial Intelligence and Statistics*, 2022. (Cited on page 23.)

[25] Simon Du, Akshay Krishnamurthy, Nan Jiang, Alekh Agarwal, Miroslav Dudik, and John Langford. Provably efficient rl with rich observations via latent state decoding. In *International Conference on Machine Learning*, 2019. (Cited on page 23.)

[26] Simon Du, Sham Kakade, Jason Lee, Shachar Lovett, Gaurav Mahajan, Wen Sun, and Ruosong Wang. Bilinear classes: A structural framework for provable generalization in RL. In *International Conference on Machine Learning*, 2021. (Cited on page 23.)

[27] Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019. (Cited on page 9.)

[28] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, 2018. (Cited on page 7.)

[29] Ilyas Fatkhullin, Jalal Etesami, Niao He, and Negar Kiyavash. Sharp analysis of stochastic optimization under global Kurdyka-łojasiewicz inequality. In *Advances in Neural Information Processing Systems*, 2022. (Cited on page 23.)

[30] Ilyas Fatkhullin, Anas Barakat, Anastasia Kireeva, and Niao He. Stochastic policy gradient methods: Improved sample complexity for fisher-non-degenerate policies. *arXiv preprint arXiv:2302.01734*, 2023. (Cited on page 23.)

[31] Maryam Fazel, Rong Ge, Sham Kakade, and Mehran Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. In *International Conference on Machine Learning*, 2018. (Cited on page 22.)

[32] Boyan Gao, Henry Gouk, Hae Beom Lee, and Timothy M Hospedales. Meta mirror descent: Optimiser learning for fast convergence. *arXiv preprint arXiv:2203.02711*, 2022. (Cited on page 31.)

[33] Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized Markov decision processes. In *International Conference on Machine Learning*, 2019. (Cited on pages 19, 20, 22, 23, and 24.)

[34] Udaya Ghai, Elad Hazan, and Yoram Singer. Exponentiated gradient meets gradient descent. In *International Conference on Algorithmic Learning Theory*, 2020. (Cited on pages 6, 10, and 30.)

[35] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016. (Cited on page 4.)

[36] Yuzheng Hu, Ziwei Ji, and Matus Telgarsky. Actor-critic is implicitly biased towards high entropy optimal policies. In *International Conference on Learning Representations*, 2022. (Cited on pages 6, 8, 20, 22, and 24.)

[37] Feihu Huang, Shangqian Gao, and Heng Huang. Bregman gradient policy optimization. In *International Conference on Learning Representations*, 2022. (Cited on pages 22 and 24.)

[38] Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, 2018. (Cited on page 9.)

[39] Ziwei Ji, Matus Telgarsky, and Ruicheng Xian. Neural tangent kernels, transportation mappings, and universal approximation. In *International Conference on Learning Representations*, 2019. (Cited on pages 9, 38, 43, and 44.)

[40] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, 2020. (Cited on pages 22 and 23.)

[41] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, 2002. (Cited on page 33.)

[42] Sham M. Kakade. A natural policy gradient. *Advances in Neural Information Processing Systems*, 2002. (Cited on pages 1, 5, and 21.)

[43] William Karush. Minima of functions of several variables with inequalities as side conditions. Master's thesis, Department of Mathematics, University of Chicago, Chicago, IL, USA, 1939. (Cited on page 26.)

[44] Michael J. Kearns and Daphne Koller. Efficient reinforcement learning in factored mdps. In *International Joint Conference on Artificial Intelligence*, 1999. (Cited on page 23.)

[45] Sajad Khodadadian, Zaiwei Chen, and Siva Theja Maguluri. Finite-sample analysis of off-policy natural actor-critic algorithm. In *International Conference on Machine Learning*, 2021. (Cited on pages 22 and 24.)

[46] Sajad Khodadadian, Prakirt Raj Jhunjhunwala, Sushil Mahavir Varma, and Siva Theja Maguluri. On the linear convergence of natural policy gradient algorithm. In *IEEE Conference on Decision and Control*, 2021. (Cited on pages 22 and 25.)

[47] Sajad Khodadadian, Prakirt Raj Jhunjhunwala, Sushil Mahavir Varma, and Siva Theja Maguluri. On linear and super-linear convergence of natural policy gradient algorithm. *Systems and Control Letters*, 2022. (Cited on pages 22 and 25.)

[48] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. (Cited on page 10.)

[49] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, 2000. (Cited on page 1.)

[50] Walid Krichene, Syrine Krichene, and Alexandre Bayen. Efficient bregman projections onto the simplex. In *IEEE Conference on Decision and Control*, 2015. (Cited on pages 5, 6, 27, 29, and 30.)

[51] Jakub Grudzien Kuba, Christian A Schroeder De Witt, and Jakob Foerster. Mirror learning: A unifying framework of policy optimisation. In *International Conference on Machine Learning*, 2022. (Cited on pages 1, 20, and 23.)

[52] Harold W. Kuhn and Albert W. Tucker. Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, 1951. (Cited on page 26.)

[53] Solomon Kullback and Richard A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 1951. (Cited on page 1.)

[54] Guanghui Lan. Policy mirror descent for reinforcement learning: Linear convergence, new sampling complexity, and generalized problem classes. *Mathematical programming*, 2022. (Cited on pages 1, 8, 21, 22, 25, and 39.)

[55] Guanghui Lan. Policy optimization over general state and action spaces. *arXiv preprint arXiv:2211.16715*, 2022. (Cited on pages 6, 8, 21, and 24.)

[56] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020. (Cited on page 23.)

[57] Yan Li and Guanghui Lan. Policy mirror descent inherently explores action space. *arXiv preprint arXiv:2303.04386*, 2023. (Cited on pages 6, 10, and 30.)

[58] Yan Li, Tuo Zhao, and Guanghui Lan. Homotopic policy mirror descent: Policy convergence, implicit regularization, and improved sample complexity. *arXiv preprint arXiv:2201.09457*, 2022. (Cited on pages 22 and 25.)

[59] Zhize Li, Hongyan Bao, Xiangliang Zhang, and Peter Richtarik. PAGE: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In *International Conference on Machine Learning*, 2021. (Cited on page 23.)

[60] Boyi Liu, Qi Cai, Zhuoran Yang, and Zhaoran Wang. Neural trust region/proximal policy optimization attains globally optimal policy. *Advances in Neural Information Processing Systems*, 2019. (Cited on pages 6, 8, 9, 22, and 24.)

[61] Yanli Liu, Kaiqing Zhang, Tamer Basar, and Wotao Yin. An improved analysis of (variance-reduced) policy gradient and natural policy gradient methods. *Advances in Neural Information Processing Systems*, 2020. (Cited on pages 8, 22, 23, 24, and 38.)

[62] Stanislaw Lojasiewicz. Une propriété topologique des sous-ensembles analytiques réels. *Les équations aux dérivées partielles*, 1963. (Cited on page 22.)

[63] Chris Lu, Jakub Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered policy optimisation. *Advances in Neural Information Processing Systems*, 2022. (Cited on page 10.)

[64] Saeed Masiha, Saber Salehkaleybar, Niao He, Negar Kiyavash, and Patrick Thiran. Stochastic second-order methods improve best-known sample complexity of SGD for gradient-dominated functions. In *Advances in Neural Information Processing Systems*, 2022. (Cited on page 23.)

[65] Jincheng Mei, Chenjun Xiao, Csaba Szepesvari, and Dale Schuurmans. On the global convergence rates of softmax policy gradient methods. In *International Conference on Machine Learning*, 2020. (Cited on page 22.)

[66] Jincheng Mei, Yue Gao, Bo Dai, Csaba Szepesvari, and Dale Schuurmans. Leveraging non-uniformity in first-order non-convex optimization. In *International Conference on Machine Learning*, 2021. (Cited on page 23.)

[67] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. (Cited on page 9.)

[68] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 2015. (Cited on pages 7 and 9.)

[69] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, 2016. (Cited on page 1.)

[70] Arkadi Nemirovski and David B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley Interscience, 1983. (Cited on pages 1, 3, and 21.)

[71] Yurii E. Nesterov and Boris T. Polyak. Cubic regularization of Newton method and its global performance. *Mathematical Programming*, 2006. (Cited on page 23.)

[72] Gergely Neu, Anders Jonsson, and Vicenç Gómez. A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*, 2017. (Cited on page 22.)

[73] Lam M. Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *International Conference on Machine Learning*, 2017. (Cited on pages 22 and 23.)

[74] Francesco Orabona. A modern introduction to online learning, 2020. URL https://open.bu.edu/handle/2144/40900. (Cited on pages 6, 10, 30, and 31.)

[75] Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 2008. (Cited on pages 1 and 22.)

[76] Boris T. Polyak. Gradient methods for the minimisation of functionals. *USSR Computational Mathematics and Mathematical Physics*, 1963. (Cited on page 22.)

[77] Bruno Scherrer. Approximate policy iteration schemes: A comparison. In *International Conference on Machine Learning*, 2014. (Cited on page 39.)

[78] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, 2015. (Cited on pages 1, 19, and 22.)

[79] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016. (Cited on pages 7 and 10.)

[80] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. (Cited on pages 1, 19, and 22.)

[81] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016. (Cited on page 1.)

[82] Lior Shani, Yonathan Efroni, and Shie Mannor. Adaptive trust region policy optimization: Global convergence and faster rates for regularized MDPs. In *AAAI Conference on Artificial Intelligence*, 2020. (Cited on pages 1, 6, 8, 21, 22, and 24.)

[83] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, 2014. (Cited on page 31.)

[84] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 2017. (Cited on page 9.)

[85] Wen Sun, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Model-based rl in contextual decision processes: Pac bounds and exponential improvements over model-free approaches. In *Conference on Learning Theory*, 2019. (Cited on page 23.)

[86] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018. (Cited on pages 7 and 18.)

[87] Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, 1999. (Cited on pages 1, 3, and 5.)

[88] Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror descent policy optimization. In *International Conference on Learning Representations*, 2022. (Cited on pages 1, 5, 6, and 19.)

[89] Sharan Vaswani, Olivier Bachem, Simone Totaro, Robert Müller, Shivam Garg, Matthieu Geist, Marlos C Machado, Pablo Samuel Castro, and Nicolas Le Roux. A general class of surrogate functions for stable and efficient reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, 2022. (Cited on pages 1, 5, 6, 8, 19, and 23.)

[90] Lingxiao Wang, Qi Cai, Zhuoran Yang, and Zhaoran Wang. Neural policy gradient methods: Global optimality and rates of convergence. In *International Conference on Learning Representations*, 2020. (Cited on pages 7, 8, 9, 22, and 24.)

[91] Weiran Wang and Miguel A Carreira-Perpinán. Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application. *arXiv preprint arXiv:1309.1541*, 2013. (Cited on page 29.)

[92] Ronald J. Williams and Jing Peng. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 1991. (Cited on page 1.)

[93] Lin Xiao. On the convergence rates of policy gradient methods. *Journal of Machine Learning Research*, 2022. (Cited on pages 1, 2, 6, 7, 8, 9, 19, 20, 21, 22, 24, 25, 28, 29, 31, 33, 37, 38, 39, and 40.)

[94] Tengyu Xu, Zhe Wang, and Yingbin Liang. Improving sample complexity bounds for (natural) actor-critic algorithms. In *Advances in Neural Information Processing Systems*, 2020. (Cited on pages 22 and 24.)

[95] Long Yang, Yu Zhang, Gang Zheng, Qian Zheng, Pengfei Li, Jianhang Huang, and Gang Pan. Policy optimization with stochastic mirror descent. *AAAI Conference on Artificial Intelligence*, 2022. (Cited on pages 22 and 24.)

[96] Batuhan Yardim, Semih Cayci, Matthieu Geist, and Niao He. Policy mirror ascent for efficient and independent learning in mean field games. In *International Conference on Machine Learning*, 2023. (Cited on page 23.)

[97] Rui Yuan, Robert M. Gower, and Alessandro Lazaric. A general sample complexity analysis of vanilla policy gradient. In *International Conference on Artificial Intelligence and Statistics*, 2022. (Cited on pages 22 and 23.)

[98] Rui Yuan, Simon Shaolei Du, Robert M. Gower, Alessandro Lazaric, and Lin Xiao. Linear convergence of natural policy gradient methods with log-linear policies. In *International Conference on Learning Representations*, 2023. (Cited on pages 2, 6, 7, 8, 20, 21, 22, 24, 25, 28, 31, 38, 39, and 40.)

[99] Andrea Zanette, Ching-An Cheng, and Alekh Agarwal. Cautiously optimistic policy optimization and exploration with linear function approximation. In *Conference on Learning Theory*, 2021. (Cited on pages 20, 22, and 24.)

[100] Wenhao Zhan, Shicong Cen, Baihe Huang, Yuxin Chen, Jason D. Lee, and Yuejie Chi. Policy mirror descent for regularized reinforcement learning: A generalized framework with linear convergence. *SIAM Journal on Optimization*, 33(2):1061–1091, 2023. (Cited on pages 1, 8, 22, 25, and 39.)

[101] Junyu Zhang, Alec Koppel, Amrit Singh Bedi, Csaba Szepesvari, and Mengdi Wang. Variational policy gradient method for reinforcement learning with general utilities. In *Advances in Neural Information Processing Systems*, 2020. (Cited on pages 7 and 23.)

[102] Junyu Zhang, Chengzhuo Ni, Zheng Yu, Csaba Szepesvari, and Mengdi Wang. On the convergence and sample efficiency of variance-reduced policy gradient method. In *Advances in Neural Information Processing Systems*, 2021. (Cited on page 23.)

Appendix

Table of Contents

Here we provide the related work discussion, the deferred proofs from the main paper and some additional noteworthy observations.

A Related work

We provide an extended discussion for the context of our work, including a comparison of different PMD frameworks and a comparison of the convergence theories of PMD in the literature. Furthermore, we discuss future work, such as extending our analysis to the dual averaging updates and developing sample complexity analysis of AMPO.

A.1 Comparisons with other policy mirror descent frameworks

In this section, we give a comparison of AMPO with some of the most popular policy optimization algorithms in the literature. First, recall AMPO's update through (12), that is, for all $s \in \mathcal{S}$,

$$\pi_s^{t+1} \in \underset{\pi_s \in \Delta(\mathcal{A})}{\operatorname{argmax}} \langle \eta_t f_s^{t+1} - \nabla h(\pi_s^t), \pi_s \rangle - \mathcal{D}_h(\pi_s, \pi_s^t), \tag{20}$$

where $\eta_t f_s^{t+1} - \nabla h(\pi_s^t) \approx \eta_t Q_s^t$ following Line 1 of Algorithm 1. The proof of (20) can be found in Lemma F.1 in Appendix F.

Generalized Policy Iteration (GPI) [86]. The update consists of evaluating the Q-function of the policy and obtaining the new policy by acting greedily with respect to the estimated Q-function. That is, for all $s \in \mathcal{S}$,

$$\pi_s^{t+1} \in \underset{\pi_s \in \Delta(\mathcal{A})}{\operatorname{argmax}} \langle Q_s^t, \pi_s \rangle. \tag{21}$$

AMPO behaves like GPI when we perfectly approximate f_s^{t+1} to the value of Q_s^t (e.g. when we consider the tabular case) and $\eta_t \to +\infty$ (or $\eta_t^{-1} \to 0$) which is the case with the use of geometrically increasing step-size schedule.

Mirror Descent Modified Policy Iteration (MD-MPI) [33]. Consider the full policy space $\Pi = \Delta(\mathcal{A})^{\mathcal{S}}$. The MD-MPI's update is as follows:

$$\pi_s^{t+1} \in \operatorname*{argmax}_{\pi_s \in \Delta(\mathcal{A})} \langle Q_s^t, \pi_s \rangle - \mathcal{D}_h(\pi_s, \pi_s^t), \quad \forall s \in \mathcal{S}. \tag{22}$$

In this case, the PMD framework of Xiao [93], which is a special case of AMPO, recovers MD-MPI with the fixed step-size $\eta_t = 1$. Consequently, Assumption (A1) holds with $\varepsilon_{\text{approx}} = 0$, and we obtain the sublinear convergence of MD-MPI through Theorem 4.3, which is

$$V^\star(\mu) - \frac{1}{T}\sum_{t<T} \mathbb{E}\left[V^t(\mu)\right] \leq \frac{1}{T}\left(\frac{\mathcal{D}_0^\star}{(1-\gamma)} + \frac{\nu_\mu}{1-\gamma}\right).$$

As explained later in Appendix H that the distribution mismatch coefficient ν_μ is upper bounded by $\mathcal{O}(\frac{1}{1-\gamma})$, we obtain an average regret of MD-MPI as $\mathcal{O}(\frac{1}{(1-\gamma)^2 T})$, which matches the convergence results in Geist et al. [33, Corollary 3].

Trust Region Policy Optimization (TRPO) [78]. The TRPO's update is as follows:

$$\begin{aligned} \pi^{t+1} \in \operatorname*{argmax}_{\pi \in \Pi} \mathbb{E}_{s\sim d_\mu^t}\left[\langle A_s^t, \pi_s \rangle\right], \\ \text{such that } \mathbb{E}_{s\sim d_\mu^t}\left[D_h(\pi_s^t, \pi_s)\right] \leq \delta, \end{aligned} \tag{23}$$

where $A_s^t = Q_s^t - V^t$ represents the advantage function, h is the negative entropy and $\delta > 0$. Like GPI, TRPO is equivalent to AMPO when at each time t, the admissible policy class is $\Pi^t = \{\pi \in \Delta(\mathcal{A})^{\mathcal{S}} : \mathbb{E}_{s\sim d_\mu^t} D_h(\pi_s^t, \pi_s) \leq \delta\}$, and we perfectly approximate Q_s^t with $\eta_t \to +\infty$.

Proximal Policy Optimization (PPO) [80]. The PPO's update consists of maximizing a surrogate function depending on the policy gradient with respect to the new policy. Namely,

$$\pi^{t+1} \in \operatorname*{argmax}_{\pi \in \Pi} \mathbb{E}_{s\sim d_\mu^t}\left[L(\pi_s, \pi_s^t)\right], \tag{24}$$

with

$$L(\pi_s, \pi_s^t) = \mathbb{E}_{a\sim\pi^t}[\min\left(r^\pi(s,a)A^t(s,a), \text{clip}(r^\pi(s,a), 1\pm\epsilon)A^t(s,a)\right)],$$

where $r^\pi(s,a) = \pi(s,a)/\pi^t(s,a)$ is the probability ratio between the current policy π^t and the new one, and the function $\text{clip}(r^\pi(s,a), 1\pm\epsilon)$ clips the probability ratio $r^\pi(s,a)$ to be no more than $1+\epsilon$ and no less than $1-\epsilon$. PPO has also a KL variation [80, Section 4], where the objective function L is defined as

$$L(\pi_s, \pi_s^t) = \eta_t \langle A_s^t, \pi_s \rangle - D_h(\pi_s^t, \pi_s),$$

where h is the negative entropy. In an exact setting and when $\Pi = \Delta(\mathcal{A})^{\mathcal{S}}$, the KL variation of PPO still differs from AMPO because it inverts the terms in the Bregman divergence penalty.

Mirror Descent Policy Optimization (MDPO.) [88]. The MDPO's update is as follows:

$$\pi^{t+1} \in \operatorname*{argmax}_{\pi \in \Pi} \mathbb{E}_{s\sim d_\mu^t}[\langle \eta_t A_s^t, \pi_s \rangle - D_h(\pi_s, \pi_s^t)], \tag{25}$$

where Π is a parameterized policy class. While it is equivalent to AMPO in an exact setting and when $\Pi = \Delta(\mathcal{A})^{\mathcal{S}}$, as we show in Appendix B, the difference between the two algorithms lies in the approximation of the exact algorithm.

Functional Mirror Ascent Policy Gradient (FMA-PG) [89]. The FMA-PG's update is as follows:

$$\begin{aligned} \pi^{t+1} &\in \operatorname*{argmax}_{\pi^\theta:\, \theta\in\Theta} \mathbb{E}_{s\sim d_\mu^t}[V^t(\mu) + \langle \eta_t \nabla_{\pi_s} V^t(\mu)\big|_{\pi=\pi^t}, \pi_s^\theta - \pi_s^t\rangle - D_h(\pi_s^\theta, \pi_s^t)] \\ &\in \operatorname*{argmax}_{\pi^\theta:\, \theta\in\Theta} \mathbb{E}_{s\sim d_\mu^t}[\langle \eta_t Q_s^t, \pi_s^\theta\rangle - D_h(\pi_s^\theta, \pi_s^t)], \end{aligned} \tag{26}$$

The second line is obtained by the definition of V^t and the policy gradient theorem (3). The discussion is the same as the previous algorithm.

Mirror Learning [51]. The on-policy version of the algorithm consists of the following update:

$$\pi^{t+1} = \underset{\pi \in \Pi(\pi^t)}{\operatorname{argmax}} \mathbb{E}_{s \sim d^t_\mu}[\langle Q^t_s, \pi_s \rangle - D(\pi_s, \pi^t_s)], \tag{27}$$

where $\Pi(\pi^t)$ is a policy class that depends on the current policy π^t and the drift functional D is defined as a map $D : \Delta(\mathcal{A}) \times \Delta(\mathcal{A}) \to \mathbb{R}$ such that $D(\pi_s, \bar{\pi}_s) \geq 0$ and $\nabla_{\pi_s} D(\pi_s, \bar{\pi}_s)\big|_{\pi_s = \bar{\pi}_s} = 0$. The drift functional D recovers the Bregman divergence as a particular case, in which case Mirror Learning is equivalent to AMPO in an exact setting and when $\Pi = \Delta(\mathcal{A})^{\mathcal{S}}$. Again, the main difference between the two algorithms lies in the approximation of the exact algorithm.

A.2 Discussion on related work

Our Contributions. Our work provides a framework for policy optimization – AMPO. For AMPO, we establish in Theorem 4.3 both $\mathcal{O}(1/T)$ convergence guarantee by using a non-decreasing step-size and linear convergence guarantee by using a geometrically increasing step-size. Our contributions to the prior literature on sublinear and linear convergence of policy optimization methods can be summarized as follows.

- The generality of our framework allows Theorem 4.3 to unify previous results in the literature and generate new theoretically sound algorithms under one guise. Both the sublinear and the linear convergence analysis of natural policy gradient (NPG) with softmax tabular policies [93] or with log-linear policies [2, 98] are special cases of our general analysis. As mentioned in Appendix A.1, MD-MPI [33] in the tabular setting is also a special case of AMPO. Thus, Theorem 4.3 recovers the best-known convergence rates in both the tabular setting [33, 93] and the non-tabular setting [16, 2, 98]. AMPO also generates new algorithms by selecting mirror maps, such as the ϵ-negative entropy mirror map in Appendix C.2 associated with Algorithm 2, and generalizes the projected Q-descent algorithm [93] from the tabular setting to a general parameterization class $\mathcal{F}^{\Theta}$.
- As discussed in Section 4.1, the results of Theorem 4.3 hold for a general setting with fewer restrictions than in previous work. The generality of the assumptions of Theorem 4.3 allows the application of our theory to specific settings, where existing sample complexity analyses could be improved thanks to the linear convergence of AMPO. For instance, since Theorem 4.3 holds for any structural MDP, AMPO could be applied directly to the linear MDP setting to derive a sample complexity analysis of AMPO which could improve that of Zanette et al. [99] and Hu et al. [36]. As we discuss in Appendix A.3, this is a promising direction for future work.
- From a technical point of view, our main contributions are: Definition 3.1 introduces a novel way of incorporating general parameterization into the policy; the update in Line 1 of Algorithm 1 simplifies the policy optimization step into a regression problem; and Lemma 4.1 establishes a key result for policies belonging to the class in Definition 3.1. Together, these innovations have allowed us to establish new state-of-the-art results in Theorem 4.3 by leveraging the modern PMD proof techniques of Xiao [93].

In particular, our technical novelty with respect to Xiao [93], Alfano and Rebeschini [2], and Yuan et al. [98] can be summarized as follows.

- In terms of algorithm design, AMPO is an innovation. The PMD algorithm proposed by Xiao [93] is strictly limited to the tabular setting and, although it is well defined for any mirror map, it cannot include general parameterization. Alfano and Rebeschini [2] and Yuan et al. [98] propose a first generalization of the PMD algorithm in the function approximation regime thanks to the linear compatible function approximation framework [1], but are limited to considering the log-linear policy parameterization and the entropy mirror map. On the contrary, AMPO solves the problem of incorporating general parameterizations in the policy thanks to Definition 3.1 and the extension of the compatible function approximation framework from linear to nonlinear, which corresponds to the parameter update in Line 1 of Algorithm 1. This innovation is key to the generality of the algorithm, as it allows AMPO to employ any mirror map and any parameterization class. Moreover, AMPO is computationally efficient for a large class of mirror maps (see Appendix C.2 and Algorithms 2 and 3). Our design is readily applied to deep RL, where the policy is usually parameterized by a neural network whose

last layer is a softmax transformation. Our policy definition can be implemented in this setting by replacing the softmax layer with a Bregman projection, as shown in Example 3.7.

- Regarding the assumptions necessary for convergence guarantees, we have weaker assumptions. Xiao [93] requires an ℓ_∞-norm on the approximation error of Q^t, i.e., $\|\widehat{Q}^t - Q^t\|_\infty \leq \varepsilon_{\text{approx}}$, for all $t \leq T$. Alfano and Rebeschini [2] and Yuan et al. [98] require an L_2-norm bound on the error of the linear approximation of Q^t, i.e., $\|w^\top \phi - Q^t\|^2_{L_2(v^t)} \leq \varepsilon_{\text{approx}}$ for some feature mapping $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ and vector $w \in \mathbb{R}^d$, for all $t \leq T$. Our approximation error $\varepsilon_{\text{approx}}$ in Assumption (A1) is an improvement since it does not require the bound to hold in ℓ_∞-norm, and allows any regression model instead of linear function approximation, especially neural networks, which greatly increases the representation power of $\mathcal{F}^\Theta$ and expands the range of applications. We further relax Assumption (A1) in Appendix G and show that the approximation error bound can be larger for earlier iterations. In addition, we improve the concentrability coefficients of Yuan et al. [98] by defining the expectation under an arbitrary state-action distribution v^t instead of the state-action visitation distribution with a fixed initial state-action pair (see Yuan et al. [98, Equation (4)]).
- As for the analysis of the algorithm, while we borrow tools from Xiao [93], Alfano and Rebeschini [2], and Yuan et al. [98], our results are not simple extensions. In fact, without our work, it is not clear from Xiao [93], Alfano and Rebeschini [2], and Yuan et al. [98] whether PMD could have theoretical guarantees in a setting with general parameterization and an arbitrary mirror map. The two main problems on this front are the non-convexity of the policy class, which prevents the use of the three-point descent lemma by Chen and Teboulle [19, Lemma 3.2] (or by Xiao [93, Lemma 6]), and the fact that the three-point identity used by Alfano and Rebeschini [2, Equation 4] holds only for the negative entropy mirror map. Our Lemma 4.1 successfully addresses general policy parameterization and arbitrary mirror maps thanks to the design of AMPO. Additionally, we provide a sample complexity analysis of AMPO when employing shallow neural networks that improves upon previous state-of-the-art results in this setting. We further improve this sample complexity analysis in Appendix J, where we consider an approximation error assumption that is weaker than Assumption (A1) (see Appendix G).

We also include a comparison wih Lan [55]. Our diffences can be outlined in two points.

- Lan [55] propose a PMD algorithm (Algorithm 2 in their paper) that can accommodate general parameterization and arbitrary mirror maps. As AMPO, it involves a two-step procedure where the first step is to find an approximation of $Q^t - \eta_t^{-1} \nabla h(\pi^t)$ and the second step is to find the policy through a Bregman projection. However, it is unclear how to implement their algorithm in practice, as they do not propose a specific method to perform either step. We provide an explicit implementation of AMPO and identify a class of mirror maps that is computationally efficient for AMPO (see Appendix C.2 and Algorithms 2 and 3).
- In terms of theoretical analysis, they assume for their results that the approximation error is bounded in ℓ_∞-norm over the action space. Let

$$\varepsilon_{\text{det}} = \mathbb{E}_{s \sim v^\star} \left[\left\| \mathbb{E}[f_s^{t+1}] - Q_s^t - \eta_t^{-1} \nabla h(\pi_s^t) \right\|_\infty \right],$$
$$\varepsilon_{sto} = \mathbb{E}_{s \sim v^\star} \left[\left\| \mathbb{E}[f_s^{t+1}] - f_s^{t+1} \right\|_\infty^2 \right],$$

  where the expectation $\mathbb{E}[f_s^{t+1}]$ is taken w.r.t. the stochasticity of the algorithm employed to obtain f^{t+1}. Lan [55] assume that both ε_{det} and ε_{sto} are bounded for all iterations t. In contrast, our assumptions are weaker as they are required to hold for the $L_2(v)$-norm we define in Section 3. Additionally, Lan [55] establishes a $\mathcal{O}(1/\sqrt{T})$ convergence rate for their algorithm without regularization and a $\mathcal{O}(1/T)$ convergence rate in the regularized case, in both cases using bounded step-sizes. We improve upon these results by obtaining a $\mathcal{O}(1/T)$ convergence rate without regularization and a linear convergence rate.

Related literature. Recently, the impressive empirical success of policy gradient (PG)-based methods has catalyzed the development of theoretically sound algorithms for policy optimization. In particular, there has been a lot of attention around algorithms inspired by mirror descent (MD) [70, 8] and, more specifically, by natural gradient descent [5]. These two approaches led to policy mirror descent (PMD) methods [82, 54] and natural policy gradient (NPG) methods [42], which, as first shown by

Neu et al. [72], is a special case of PMD. For instance, PMD and NPG are the building blocks of the state-of-the-art policy optimization algorithms, TRPO [78] and PPO [80]. Leveraging various techniques from the MD literature, it has been established that PMD, NPG, and their variants converge to the global optimum in different settings. We refer to global optimum convergence as an analysis that guarantees that $V^\star(\mu) - \mathbb{E}\left[V^T(\mu)\right] \leq \epsilon$ after T iterations with $\epsilon > 0$. As an important variant of NPG, we will also discuss the literature of the convergence analysis of natural actor-critic (NAC) [75, 12]. The comparison of AMPO with different methods will proceed from the tabular case to different function approximation regimes.

Sublinear convergence analyses of PMD, NPG and NAC. For softmax tabular policies, Shani et al. [82] establish a $\mathcal{O}(1/\sqrt{T})$ convergence rate for unregularized NPG and $\mathcal{O}(1/T)$ for regularized NPG. Agarwal et al. [1], Khodadadian et al. [45] and Xiao [93] improve the convergence rate for unregularized NPG and NAC to $\mathcal{O}(1/T)$ and Xiao [93] extends the same convergence rate to projected Q-descent. The same convergence rate is established by MD-MPI [33] through the PMD framework.

In the function approximation regime, Zanette et al. [99] and Hu et al. [36] achieve $\mathcal{O}(1/\sqrt{T})$ convergence rate by developing variants of PMD methods for the linear MDP [40] setting. The same $\mathcal{O}(1/\sqrt{T})$ convergence rate is obtained by Agarwal et al. [1] for both log-linear and smooth policies, while Yuan et al. [98] improve the convergence rate to $\mathcal{O}(1/T)$ for log-linear policies. For smooth policies, the convergence rate is later improved to $\mathcal{O}(1/T)$ either by adding an extra Fisher-non-degeneracy condition on the policies [61] or by analyzing NAC under Markovian sampling [94]. Yang et al. [95] and Huang et al. [37] consider Lipschitz and smooth policies [97], obtain $\mathcal{O}(1/\sqrt{T})$ convergence rates for PMD-type methods and faster $\mathcal{O}(1/T)$ convergence rates by applying the variance reduction techniques SARAH [73] and STORM [22], respectively. As for neural policy parameterization, Liu et al. [60] establish a $\mathcal{O}(1/\sqrt{T})$ convergence rate for two-layer neural PPO. The same $\mathcal{O}(1/\sqrt{T})$ convergence rate is established by Wang et al. [90] for two-layer neural NAC, which is later improved to $\mathcal{O}(1/T)$ by Cayci et al. [16], using entropy regularization.

We highlight that all of the above sublinear convergence analyses, for both softmax tabular policies and the function approximation regime, are obtained either by using a decaying step-size or a constant step-size. Under these step-size schemes, our AMPO's $\mathcal{O}(1/T)$ sublinear convergence rate is the state of the art: it recovers the best-known convergence rates in the tabular setting [33, 93] without regularization; it improves the $\mathcal{O}(1/\sqrt{T})$ convergence rate of Zanette et al. [99] and Hu et al. [36] to $\mathcal{O}(1/T)$ for the linear MDP setting; it recovers the best-known convergence rates for the log-linear policies [98]; it matches the $\mathcal{O}(1/T)$ sublinear convergence rate for smooth and Fisher-non-degenerate policies [61] and the same convergence rate of Yang et al. [95] and Huang et al. [37] for Lipschitz and smooth policies without introducing variance reduction techniques; it matches the previous best-known convergence result in the neural network settings [16] without regularization; lastly, it goes beyond all these results by allowing general parameterization. We refer to Table 1 for an overview of recent sublinear convergence analyses of NPG/PMD.

Linear convergence analysis of PMD, NPG, NAC and other PG methods. In the softmax tabular policy settings, the linear convergence guarantees of NPG and PMD are achieved by either adding regularization [17, 100, 54, 58] or by varying the step-sizes [11, 46, 47, 93].

In the function approximation regime, the linear convergence guarantees are achieved for NPG with log-linear policies, either by adding entropy regularization [15] or by choosing geometrically increasing step-sizes [2, 98]. It can also be achieved for NAC with log-linear policy by using adaptive increasing step-sizes [20].

Again, our AMPO's linear convergence rate is the state of the art: not only it recovers the best-known convergence rates in both the tabular setting [93] and the log-linear policies [2, 98] without regularization [17, 100, 54, 58], nor adaptive step-sizes [11, 46, 47, 20], but also it achieves the new state-of-the-art linear convergence rate for PG-based methods with general parameterization, including the neural network parameterizations. We refer to Table 2 for an overview of recent linear convergence analyses of NPG/PMD.

Alternatively, by exploiting a Polyak-Lojasiewicz (PL) condition [76, 62], fast linear convergence results can be achieved for PG methods under different settings, such as linear quadratic control problems [31] and softmax tabular policies with entropy regularization [65, 97]. The PL condition

is extensively studied by Bhandari and Russo [10] to identify more general MDP settings. Like the cases of NPG and PMD, linear convergence of PG can also be obtained for the softmax tabular policy without regularization by choosing adaptive step sizes through exact line search [11] or by exploiting non-uniform smoothness [66]. When the PL condition is relaxed to other weaker conditions, PG methods combined with variance reduction methods such as SARAH [73] and PAGE [59] can also achieve linear convergence. This is shown by Fatkhullin et al. [29, 30] when the PL condition is replaced by the weak PL condition [97], which is satisfied by Fisher-non-degenerate policies [24]. It is also shown by Zhang et al. [102], where the MDP satisfies some hidden convexity property that contains a similar property to the weak PL condition studied by Zhang et al. [101]. Lastly, linear convergence is established for the cubic-regularized Newton method [71], a second-order method, applied to Fisher-non-degenerate policies combined with variance reduction [64].

Outside of the literature focusing on finite time convergence guarantees, Vaswani et al. [89] and Kuba et al. [51] provide a theoretical analysis for variations of PMD and show monotonic improvements for their frameworks. Additionally, Kuba et al. [51] give an infinite time convergence guarantee for their framework.

A.3 Future work

Our work opens several interesting research directions in both algorithmic and theoretical aspects.

From an algorithmic point of view, the updates in Lines 1 and 2 of AMPO are not explicit. This might be an issue in practice, especially for large scale RL problems. It would be interesting to design efficient regression solver for minimizing the approximation error in Line 1 of Algorithm 1. For instance, by using the dual averaging algorithm [7, Chapter 4], it could be possible to replace the term $\nabla h(\pi_s^t)$ with f_s^t for all $s \in \mathcal{S}$, to make the computation of the algorithm more efficient. That is, it could be interesting to consider the following variation of Line 1 in Algortihm 1:

$$\left\| f^{t+1} - Q^t - \frac{\eta_{t-1}}{\eta_t} f^t \right\|_{L_2(v^t)}^2 \leq \varepsilon_{\text{approx}}. \tag{28}$$

Notice that (28) has the same update as (17), however, (28) is not restricted to using the negative entropy mirror map. To efficiently solve the regression problem in Line 1 of Algorithm 1, one may want to apply modern variance reduction techniques [73, 22, 59]. This has been done by Liu et al. [61] for NPG method.

From a theoretical point of view, it would be interesting to derive a sample complexity analysis for AMPO in specific settings, by leveraging its linear convergence. As mentioned for the linear MDP [40] in Appendix A.2, one can apply the linear convergence theory of AMPO to other structural MDPs, e.g., block MDP [25], factored MDP [44, 85], RKHS linear MDP and RKHS linear mixture MDP [26], to build new sample complexity results for these settings, since the assumptions of Theorem 4.3 do not impose any constraint on the MDP. On the other hand, it would be interesting to explore the interaction between the Bregman projected policy class and the expected Lipschitz and smooth policies [97] and the Fish-non-degenerate policies [61] to establish new improved sample complexity results in these settings, again thanks to the linear convergence theory of AMPO.

Additionally, it would be interesting to study the application of AMPO to the offline setting. In the main text, we have discussed how to extend Algorithm 1 and Theorem 4.3 to the offline setting, where v^t can be set as the state-action distribution induced by an arbitrary behavior policy that generates the data. However, we believe that this direction requires further investigation. One of the major challenges of offline RL is dealing with the distribution shifts that stem from the mismatch between the trained policy π^t and the behaviour policy. Several methods have been introduced to deal with this issue, such as constraining the current policy to be close to the behavior policy [56]. We leave introducing offline RL techniques in AMPO as future work.

Another direction for future work is extending the policy update of AMPO to mirror descent algorithm based on value iteration and Bellman operators, such as MD-MPI [33], in order to extend existing results to the general parameterization setting. Other interesting settings that have been addressed using the PMD framework are mean-field games [96] and constrained MDPs [23]. We hope to build on the existing literature for these settings and see whether our results can bring any improvements.

Finally, this work theoretically indicates that, perhaps the most important future work of PMD-type algorithms is to design efficient policy evaluation algorithms to make the estimation of the Q-function

Table 1: Overview of sublinear convergence results for NPG and PMD methods with constant step-size in different settings. The dark blue cells contain our new results. The light blue cells contain previously known results that we recover as special cases of our analysis. The pink cells contain previously known results that we improve upon by providing a faster convergence rate. The white cells contain existing results that have already been improved by other literature or that we could not recover under our general analysis.

Algorithm	**Rate**	**Comparisons to our works**
Setting: Softmax tabular policies		
Adaptive TRPO [82]	$\mathcal{O}(1/\sqrt{T})$	They employ regularization
Tabular off-policy NAC [45]	$\mathcal{O}(1/T)$	We have a weaker approximation error Assumption (A1) with L_2 instead of ℓ_∞ norm
Tabular NPG [1]	$\mathcal{O}(1/T)$	
MD-MPI [33]	$\mathcal{O}(1/T)$	We match their results when $f^\theta(s,a) = \theta_{s,a}$.
Tabular NPG/ projected Q-descent [93]	$\mathcal{O}(1/T)$	We recover their results when $f^\theta(s,a) = \theta_{s,a}$; we have a weaker approximation error Assumption (A1) with L_2 instead of ℓ_∞ norm.
Setting: Log-linear policies		
Q-NPG [1]	$\mathcal{O}(1/\sqrt{T})$	
Q-NPG/NPG [98]	$\mathcal{O}(1/T)$	We recover their results when $f^\theta(s,a)$ is linear to θ.
Setting: Softmax two-layer neural policies		
Neural PPO [60]	$\mathcal{O}(1/\sqrt{T})$	
Neural NAC [90]	$\mathcal{O}(1/\sqrt{T})$	
Regularized neural NAC [16]	$\mathcal{O}(1/T)$	We match their results without regularization.
Setting: Linear MDP		
NPG [99, 36]	$\mathcal{O}(1/\sqrt{T})$	
Setting: Smooth policies		
NPG [1]	$\mathcal{O}(1/\sqrt{T})$	
NAC under Markovian sampling [94]	$\mathcal{O}(1/T)$	
NPG with Fisher-non-degenerate policies [61]	$\mathcal{O}(1/T)$	
Setting: Lipschitz and Smooth policies		
Variance reduced PMD [95, 37]	$\mathcal{O}(1/T)$	We match their results without variance reduction.
Setting: Bregman projected policies with general parameterization and mirror map		
Regularized PMD [55]	$\mathcal{O}(1/T)$	We match their results without regularization; we have a weaker approximation error Assumption (A1) with L_2 instead of ℓ_∞ norm.
AMPO (Theorem 4.3, this work)	$\mathcal{O}(1/T)$	

Table 2: Overview of linear convergence results for NPG and PMD methods in different settings. The darker cells contain our new results. The light cells contain previously known results that we recover as special cases of our analysis, and extend the permitted concentrability coefficients settings. The white cells contain existing results that we could not recover under our general analysis.

Algorithm	Reg.	C.S.	A.I.S.	N.I.S.*	Error assumption**
Setting: Softmax tabular policies					
NPG [17]	✓	✓			ℓ_∞
PMD [100]	✓	✓			ℓ_∞
NPG [54]	✓			✓	ℓ_∞
NPG [58]	✓			✓	ℓ_∞
NPG [11]			✓		
NPG [46, 47]			✓		ℓ_∞
NPG / Projected Q-descent [93]				✓	ℓ_∞
Setting: Log-linear policies					
NPG [15]	✓	✓			L_2
Off-policy NAC [20]			✓		ℓ_∞
Q-NPG [2]				✓	L_2
Q-NPG/NPG [98]				✓	L_2
Setting: Bregman projected policies with general parameterization and mirror map					
AMPO (Theorem 4.3, this work)				✓	L_2

* **Reg.**: regularization; **C.S.**: constant step-size; **A.I.S.**: Adaptive increasing step-size; **N.I.S.**: Non-adaptive increasing step-size.
** **Error assumption.**: ℓ_∞ means that the approximation error assumption uses the ℓ_∞-norm; and L_2 means that the approximation error assumption uses the weaker L_2 norm.

as accurate as possible, such as using offline data for training, and to construct adaptive representation learning for $\mathcal{F}^\Theta$ to closely approximate Q-function, so that ϵ_{approx} is guaranteed to be small. This matches one of the most important research questions for deep Q-learning type algorithms for general policy optimization problems.

B Equivalence of (9)-(10) and (11) in the tabular case

To demonstrate the equivalence between the two-step update (9)-(10) and the one-step update (11) for policy mirror descent in the tabular case, it is sufficient to validate the following lemma, which comes from the optimization literature. The proof of this lemma can be found in Bubeck [14, Chapter 4.2]. However, for the sake of completeness, we present the proof here.

Lemma B.1 (Right after Theorem 4.2 in Bubeck [14])**.** *Consider the mirror descent update in* (5)-(6) *for the minimization of a function $V(\cdot)$, that is,*

$$y^{t+1} = \nabla h(x^t) - \eta_t \nabla V(x)|_{x=x^t}, \tag{29}$$
$$x^{t+1} = Proj^h_{\mathcal{X}}(\nabla h^*(y^{t+1})). \tag{30}$$

Then the mirror descent update can be rewritten as

$$x^{t+1} \in \operatorname*{argmin}_{x\in\mathcal{X}} \eta_t \langle x, \nabla V(x)|_{x=x^t}\rangle + \mathcal{D}_h(x, x^t). \tag{31}$$

Proof. From definition of the Bregman projection step, starting from (29) we have

$$
\begin{aligned}
x^{t+1} &= \mathrm{Proj}^h_{\mathcal{X}}(\nabla h^*(y^{t+1})) = \operatorname*{argmin}_{x\in\mathcal{X}} \mathcal{D}_h(x, \nabla h^*(y^{t+1})) \\
&\in \operatorname*{argmin}_{x\in\mathcal{X}} \nabla h(x) - \nabla h(\nabla h^*(y^{t+1})) - \left\langle \nabla h(\nabla h^*(y^{t+1})), x - \nabla h^*(y^{t+1}) \right\rangle \\
&\overset{(4)}{\in} \operatorname*{argmin}_{x\in\mathcal{X}} \nabla h(x) - y^{t+1} - \left\langle y^{t+1}, x - \nabla h^*(y^{t+1}) \right\rangle \\
&\in \operatorname*{argmin}_{x\in\mathcal{X}} \nabla h(x) - \langle x, y^{t+1} \rangle \\
&\overset{(29)}{\in} \operatorname*{argmin}_{x\in\mathcal{X}} \nabla h(x) - \langle x, \nabla h(x^t) - \eta_t \nabla V(x)|_{x=x^t} \rangle \\
&\in \operatorname*{argmin}_{x\in\mathcal{X}} \eta_t \langle x, \nabla V(x)|_{x=x^t} \rangle + \nabla h(x) - \nabla h(x^t) - \left\langle \nabla h(x^t), x - x^t \right\rangle \\
&\in \operatorname*{argmin}_{x\in\mathcal{X}} \eta_t \langle x, \nabla V(x)|_{x=x^t} \rangle + \mathcal{D}_h(x, x^t),
\end{aligned}
$$

where the second and the last lines are both obtained by the definition of the Bregman divergence. □

The one-step update in (31) is often taken as the definition of mirror descent [8], which provides a proximal view point of mirror descent, i.e., a gradient step in the primal space with a regularization of Bregman divergence.

C AMPO for specific mirror maps

In this section, we give the derivations for Example 3.2, which is based on the Karush-Kuhn-Tucker (KKT) conditions [43, 52], and then provide details about the ω-potential mirror map class from Section 3.1.

C.1 Derivation of Example 3.2

We give here the derivation of Example 3.2. Let h be the negative entropy mirror map, that is

$$
h(\pi_s) = \sum_{a\in\mathcal{A}} \pi(a \mid s) \log(\pi(a \mid s)), \qquad \forall \pi_s \in \Delta(\mathcal{A}) \text{ and } \forall s \in \mathcal{S}.
$$

For every state $s \in \mathcal{S}$, we solve the minimization problem

$$
\pi^\theta_s \in \operatorname*{argmin}_{\pi_s\in\Delta(\mathcal{A})} \mathcal{D}_h(\pi_s, \nabla h^*(\eta f^\theta_s))
$$

through the KKT conditions. We formalize it as

$$
\begin{aligned}
\pi^\theta_s \in \operatorname*{argmin}_{\pi_s\in\mathbb{R}^{|\mathcal{A}|}} \; & \mathcal{D}_h(\pi_s, \nabla h^*(\eta f^\theta_s)) \\
\text{subject to } & \langle \pi_s, \mathbf{1} \rangle = 1, \\
& \pi(a \mid s) \geq 0, \qquad \forall\, a \in \mathcal{A},
\end{aligned}
$$

where $\mathbf{1}$ denotes a vector in $\mathbb{R}^{|\mathcal{A}|}$ with coordinates equal to 1 element-wisely. The conditions then become

$$
\begin{aligned}
\text{(stationarity)} \qquad & \log(\pi^\theta_s) - \eta f^\theta_s + (\lambda_s + 1)\mathbf{1} - c_s = 0, \\
\text{(complementary slackness)} \qquad & c^a_s \pi^\theta(a \mid s) = 0, \quad \forall\, a \in \mathcal{A}, \\
\text{(primal feasibility)} \qquad & \langle \pi^\theta_s, \mathbf{1} \rangle = 1,\ \pi^\theta(a \mid s) \geq 0, \qquad \forall\, a \in \mathcal{A}, \\
\text{(dual feasibility)} \qquad & c^a_s \geq 0, \quad \forall\, a \in \mathcal{A},
\end{aligned}
$$

where $\log(\pi_s)$ is applied element-wisely, $\lambda_s \in \mathbb{R}$ and $c^a_s \in \mathbb{R}$ are the dual variables, and c_s devotes the vector $[c^a_s]_{a\in\mathcal{A}}$. It is easy to verify that the solution

$$
\pi^\theta_s = \frac{\exp(\eta f^\theta_s)}{\|\exp(\eta f^\theta_s)\|_1},
$$

with $\lambda_s = \log\left(\sum_{a\in\mathcal{A}} \exp(\eta f^\theta(s,a))\right) - 1$ and $c_s^a = 0$ for all $a \in \mathcal{A}$, satisfies all the conditions.

When $f^\theta(s,a) = \theta_{s,a}$ we obtain the tabular softmax policy $\pi^\theta(a \mid s) \propto \exp(\eta\theta_{s,a})$. When $f^\theta(s,a) = \theta^\top\phi(s,a)$ is a linear function, for $\theta \in \mathbb{R}^d$ and for a feature function $\phi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^d$, we obtain the log-linear policy $\pi^\theta(a \mid s) \propto \exp(\eta\theta^\top\phi(s,a))$. When $f^\theta : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$ is a neural network, we obtain the softmax neural policy $\pi^\theta(a \mid s) \propto \exp(\eta f^\theta(s,a))$.

C.2 More on ω-potential mirror maps

In this section, we provide details about the ω-potential mirror map class from Section 3.1, including the derivation of (14), several instantiations of ω-potential mirror map mentioned in Section 3.1 with their derivations, and an iterative algorithm to find approximately the Bregman projection induced by ω-potential mirror map when an exact solution is not available.

We give a different but equivalent formulation of Proposition 2 of Krichene et al. [50].

Proposition C.1. *For $u \in (-\infty, +\infty]$ and $\omega \le 0$, an increasing C^1-diffeomorphism $\phi : (-\infty, u) \to (\omega, +\infty)$ is called an ω-potential if*

$$\lim_{x\to-\infty}\phi(x) = \omega, \quad \lim_{x\to u}\phi(x) = +\infty, \quad \int_0^1 \phi^{-1}(x)dx \le \infty.$$

Let the mirror map h_ϕ be defined as

$$h_\phi(\pi_s) = \sum_{a\in\mathcal{A}} \int_1^{\pi(a|s)} \phi^{-1}(x)dx.$$

We have that π_s^t is a solution to the Bregman projection

$$\min_{\pi\in\Delta_s} \mathit{Proj}^h_{\Delta(\mathcal{A})}(\nabla h^*_\phi(\eta_{t-1}f_s^t)),$$

if and only if there exist a normalization constant $\lambda_s^t \in \mathbb{R}$ such that

$$\pi^t(a \mid s) = \sigma(\phi(\eta_{t-1}f^t(s,a) + \lambda_s^t)), \quad \forall a \in \mathcal{A}, \tag{32}$$

and $\sum_{a\in\mathcal{A}} \pi^t(a \mid s) = 1$, where for all $s \in \mathcal{S}$ and $\sigma(z) = \max(z, 0)$ for $z \in \mathbb{R}$.

We can now use Proposition C.1 to derive (14).

Consider an ω-potential mirror map h_ϕ associated with an ω-potential ϕ. By definition, we have

$$\nabla h_\phi(\pi_s^t) = [\phi^{-1}(\pi^t(a \mid s))]_{a\in\mathcal{A}}. \tag{33}$$

Plugging (32) into (33), we have

$$\begin{aligned}
\nabla h_\phi(\pi_s^t) &\overset{(33)}{=} [\phi^{-1}(\pi^t(a \mid s))]_{a\in\mathcal{A}} \\
&\overset{(32)}{=} \left[\phi^{-1}\left(\sigma\left(\phi(\eta_{t-1}f^t(s,a) + \lambda_s^t)\right)\right)\right]_{a\in\mathcal{A}} \\
&= \left[\max\left(\phi^{-1}\left(\phi(\eta_{t-1}f^t(s,a) + \lambda_s^t)\right), \phi^{-1}(0)\right)\right]_{a\in\mathcal{A}} \\
&= \left[\max\left(\eta_{t-1}f^t(s,a) + \lambda_s^t, \phi^{-1}(0)\right)\right]_{a\in\mathcal{A}} \\
&= \left[\max\left(\eta_{t-1}f^t(s,a), \phi^{-1}(0) - \lambda_s^t\right)\right]_{a\in\mathcal{A}} + \lambda_s^t,
\end{aligned}$$

where the third line is obtained by using the increasing property of ϕ^{-1}, as ϕ is increasing. Finally, plugging the above expression of $\nabla h_\phi(\pi_s^t)$ into Line 1, we obtain (14), which is

$$\theta^{t+1} \in \operatorname{argmin}_{\theta\in\Theta} \mathbb{E}_{(s,a)\sim v^t}\left[(f^\theta(s,a) - Q^t(s,a) - \eta_t^{-1}\max(\eta_{t-1}f^t(s,a), \phi^{-1}(0) - \lambda_s^t))^2\right],$$

where the term λ_s^t is dropped, as it is constant over actions and does not affect the resulting policy.

Once (14) is obtained, we can instantiate AMPO for mirror maps belonging to this class. We highlight that due to the definition of the Bregman divergence, two mirror maps that only differ for a constant term are equivalent and generate the same algorithm. We start with the negative entropy, which leads to a closed solution for λ_s^t and therefore for the Bregman projection.

Negative entropy. Let $\phi(x) = \exp(x-1)$, which is an ω-potential with $\omega = 0$, $u = +\infty$, and

$$\int_0^1 \phi^{-1}(x)dx \;=\; \int_0^1 \log(x) + 1 dx \;=\; [x\log(x)]_0^1 = 0 \leq +\infty.$$

The mirror map h_ϕ becomes the negative entropy, as

$$h_\phi(\pi_s) = \sum_{a\in\mathcal{A}} \int_1^{\pi(a|s)} (\log(x)+1)dx = \sum_{a\in\mathcal{A}} \pi(a \mid s)\log(\pi(a \mid s)),$$

and the associated Bregman divergence becomes the KL divergence, i.e., $D_{h_\phi}(\pi_s, \bar{\pi}_s) = \mathtt{KL}(\pi_s, \bar{\pi}_s)$. Equation (17) follows from Equation (14) by the fact that

$$\phi^{-1}(0) = \log(0) + 1 = -\infty,$$

which means that $\max(\eta_{t-1}f^t, \phi^{-1}(0) - \lambda_s^t) = \eta_{t-1}f^t$ element-wisely.

As we showed in Appendix C.1, the Bregman projection (Line 2 of Algorithm 1) for the negative entropy has a closed form which is $\pi_s^{t+1} \propto \exp(\eta_t f_s^{t+1})$.

In NPG with softmax tabular policies [1, 93], at time t, the updates for the policies have

$$\pi_s^{t+1} \propto \pi_s^t \odot \exp(\eta_t Q_s^t), \quad \forall s \in \mathcal{S}. \tag{34}$$

When considering AMPO with $f^\theta(s,a) = \theta_{s,a} \in \mathbb{R}$, from (17), we obtain that for all $s, a \in \mathcal{S}, \mathcal{A}$, we have

$$\begin{aligned}\theta^{t+1} \in \operatorname*{argmin}_{\theta\in\mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}} \left\|\theta - Q^t - \frac{\eta_{t-1}}{\eta_t}\theta^t\right\|_{L_2(v^t)}^2 &\iff f^{t+1}(s,a) = Q^t(s,a) + \frac{\eta_{t-1}}{\eta_t} f^t(s,a) \\ &\iff \eta_t f^{t+1}(s,a) = \eta_t Q^t(s,a) + \eta_{t-1} f^t(s,a).\end{aligned}$$

With the above expression, we have the AMPO's updates for the policies rewritten as

$$\begin{aligned}\pi_s^{t+1} &\propto \exp(\eta_t f_s^{t+1}) \\ &= \exp(\eta_t Q_s^t + \eta_{t-1} f_s^t) \\ &\propto \pi_s^t \odot \exp(\eta_t Q_s^t), \quad \forall s \in \mathcal{S},\end{aligned}$$

which recovers (34). In particular, the summation in the second line is element-wise and the third line is obtained because of $\pi_s^t \propto \exp(\eta_{t-1} f_s^t)$, as shown in Appendix C.1.

In Q-NPG with log-linear policies [1, 98, 2], at time t, the updates for the policies have

$$\pi^{t+1}(a \mid s) \propto \pi^t(a \mid s)\exp(\eta_t \phi(s,a)^\top w^t), \tag{35}$$

where $\phi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^d$ is a feature map, and

$$w^t \in \operatorname*{argmin}_{w\in\mathbb{R}^d} \mathbb{E}_{(s,a)\sim d_\mu^t}\left[(Q^t(s,a) - \phi(s,a)^\top w)^2\right]. \tag{36}$$

Like in the tabular case, when considering AMPO with $\theta \in \mathbb{R}^d$, $f^\theta(s,a) = \phi(s,a)^\top\theta$ and $v^t = d_\mu^t$, from (17), we obtain that for all $s, a \in \mathcal{S}, \mathcal{A}$, we have

$$\begin{aligned}\theta^{t+1} &\in \operatorname*{argmin}_{\theta\in\mathbb{R}^d} \mathbb{E}_{(s,a)\sim d_\mu^t}\left[\left(\phi(s,a)^\top\theta - Q^t(s,a) - \frac{\eta_{t-1}}{\eta_t}\phi(s,a)^\top\theta^t\right)^2\right] \\ &\in \operatorname*{argmin}_{\theta\in\mathbb{R}^d} \mathbb{E}_{(s,a)\sim d_\mu^t}\left[\left(Q^t(s,a) - \phi(s,a)^\top\Big(\underbrace{\theta - \frac{\eta_{t-1}}{\eta_t}\theta^t}_{w}\Big)\right)^2\right].\end{aligned}$$

Compared the above form with (36), we obtain that

$$w^t = \theta^{t+1} - \frac{\eta_{t-1}}{\eta_t}\theta^t \quad\iff\quad \eta_t\theta^{t+1} = \eta_t w^t + \eta_{t-1}\theta^t. \tag{37}$$

Algorithm 2: Bregman projection for ϵ-negative entropy

Input: vector to project $x \in \mathbb{R}^{|\mathcal{A}|}$, parameter ϵ.

Initialize $y = \exp(x)$ element-wisely.

Let $y^{(i)}$ be the i-th smallest element of y.

Let $i^\star$ be the smallest index for which

$$(1 + \epsilon(|\mathcal{A}| - i + 1))y^{(i)} - \epsilon \sum_{j \geq i} y^{(j)} > 0.$$

Set

$$\lambda = \frac{\sum_{i \geq i^\star} y^{(i)}}{1 + \epsilon(|\mathcal{A}| - i^\star + 1)}.$$

Return: the projected vector $\left(\sigma\left(-\epsilon + y_a/\lambda\right)\right)_{a \in \mathcal{A}}$.

So, the AMPO's updates for the policies can be rewritten as

$$\begin{aligned} \pi^{t+1}(a \mid s) &\propto \exp(\eta_t \phi(s,a)^\top \theta^{t+1}) \\ &\overset{(37)}{=} \exp(\phi(s,a)^\top (\eta_t w^t + \eta_{t-1}\theta^t)) \\ &\propto \pi^t(a \mid s) \exp(\eta_t \phi(s,a)^\top w^t), \end{aligned}$$

where the last line is obtained because of $\pi_s^t \propto \exp(\eta_{t-1} f_s^t)$, as shown in Appendix C.1, and we recover (35).

We next present the squared ℓ_2-norm and the ϵ-negative entropy. For these two mirror maps, the Bregman projection can be computed exactly but has no closed form.

Squared ℓ_2-norm. Let ϕ be the identity function. The mirror map h_ϕ becomes the squared ℓ_2-norm, up to a constant term, as

$$h_\phi(\pi_s) = \sum_{a \in \mathcal{A}} \int_1^{\pi(a|s)} x \, dx = \frac{1}{2} \sum_{a \in \mathcal{A}} \left(\pi(a \mid s)^2 - 1\right).$$

The associated Bregman divergence becomes the squared Euclidean distance, i.e., $D_{h_\phi}(\pi_s, \bar{\pi}_s) = \frac{1}{2}\left\|\pi_s - \bar{\pi}_s\right\|_2^2$, and $\nabla h^*(\cdot)$ is the identity function. The update in (15) follows immediately and the Bregman projection step with the Euclidean distance becomes, for all $s \in \mathcal{S}$,

$$\pi_s^{t+1} = \text{Proj}_{\Delta(\mathcal{A})}^{h_\phi}(\nabla h^*(\eta_t f_s^{t+1})) = \text{Proj}_{\Delta(\mathcal{A})}^{l_2}(\eta_t f_s^{t+1}) = \operatorname*{argmin}_{p \in \Delta(\mathcal{A})} \left\|p - \eta_t f_s^{t+1}\right\|_2^2. \tag{38}$$

In the projected-Q descent for tabular policies developed by Xiao [93], at time t, the updates for the policies are

$$\pi_s^{t+1} \in \operatorname*{argmin}_{p \in \Delta(\mathcal{A})} \left\|\pi_s^t + \eta_t Q_s^t - p\right\|_2^2, \quad \forall s \in \mathcal{S}. \tag{39}$$

When considering AMPO with $f^\theta(s,a) = \theta_{s,a}$ and $\Theta = \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}|}$, (15) is solved with

$$f^{t+1}(s,a) = \theta_{s,a}^{t+1} = Q^t(s,a) + \eta_t^{-1} \pi^t(a \mid s).$$

Plugging the above expression into (38), we recover (39).

Notice that the Euclidean projection onto the probability simplex can be obtained exactly, as shown by Wang and Carreira-Perpinán [91].

ϵ-negative entropy [50]. Let $\epsilon \geq 0$ and define the ϵ-exponential potential as $\phi(x) = \exp(x-1) - \epsilon$. The mirror map h_ϕ becomes

$$h_\phi(\pi_s) = \sum_{a \in \mathcal{A}} \int_1^{\pi(a|s)} (\log(x+\epsilon)+1) dx = \sum_{a \in \mathcal{A}} \left[(\pi(a \mid s) + \epsilon) \ln(\pi(a \mid s) + \epsilon) - (1+\epsilon)\ln(1+\epsilon)\right].$$

Algorithm 3: Bregman projection for ω-potential mirror maps

Input: vector to project $x \in \mathbb{R}^{|\mathcal{A}|}$, ω-potential ϕ, precision ε.

Initialize

$$\bar{\nu} = \phi^{-1}(1) - \max_{a \in \mathcal{A}} x_a$$
$$\underline{\nu} = \phi^{-1}(1/|\mathcal{A}|) - \max_{a \in \mathcal{A}} x_a$$

Define $\tilde{x}(\nu) = (\sigma(\phi(x_a + \nu)))_{a \in \mathcal{A}}$.

while $\|\tilde{x}(\bar{\nu}) - \tilde{x}(\underline{\nu})\|_1 > \varepsilon$ **do**

  Let $\nu^+ \leftarrow (\bar{\nu} + \underline{\nu})/2$

  if $\sum_{a \in \mathcal{A}} \tilde{x}_a(\nu^+) > 1$ **then**

    $\bar{\nu} \leftarrow \nu^+$

  else

    $\underline{\nu} \leftarrow \nu^+$

Return $\tilde{x}(\bar{\nu})$

An exact solution to the associated projection can then be found in $\widetilde{\mathcal{O}}(|\mathcal{A}|)$ computations using Algorithm 2, which has been proposed by Krichene et al. [50, Algorithm 4]. Additionally, following (14), the regression problem in Line 1 of Algorithm 1 becomes

$$\theta^{t+1} \in \operatorname*{argmin}_{\theta \in \Theta} \left\| f^\theta - Q^t - \eta_t^{-1} \max(\eta_{t-1} f^t, 1 + \log(\epsilon) - \lambda_s^t) \right\|^2_{L_2(v^t)},$$

where λ_s^t can be obtained through Algorithm 2.

The Bregman projection for generic mirror maps can be computed approximately in $\widetilde{\mathcal{O}}(|\mathcal{A}|)$ computations through a bisection algorithm. Krichene et al. [50] propose one such algorithm, which we report in Algorithm 3 for completeness. We next provide two mirror maps that have appeared before in the optimization literature, but do not lead to an exact solution to the Bregman projection. We leave them as object for future work.

Negative Tsallis entropy [74, 57]. Let $q > 0$ and define ϕ as

$$\phi_q(x) = \begin{cases} \exp(x-1) & \text{if } q = 1, \\ \left[\sigma\left(\frac{(q-1)x}{q}\right)\right]^{\frac{1}{q-1}} & \text{else.} \end{cases}$$

The mirror map h_{ϕ_q} becomes the negative Tsallis entropy, that is

$$h_{\phi_q}(\pi_s) = \sum \pi(a \mid s) \log_q(\pi(a \mid s)), \tag{40}$$

where, for $y > 0$,

$$\log_q(y) = \begin{cases} \log(y) & \text{if } q = 1, \\ \frac{-y^{q-1}}{q-1} & \text{else.} \end{cases}$$

If $q \neq 1$ and following (14), the regression problem in Line 1 of Algorithm 1 becomes

$$\theta^{t+1} \in \operatorname*{argmin}_{\theta \in \Theta} \left\| f^\theta - Q^t - \frac{\eta_{t-1}}{\eta_t} f^t \right\|^2_{L_2(v^t)},$$

Hyperbolic entropy [34]. Let $b > 0$ and define ϕ as

$$\phi_b(x) = b \sinh(x)$$

The mirror map h_{ϕ_b} becomes the hyperbolic entropy, that is

$$h_{\phi_b}(\pi_s) = \sum_{a \in \mathcal{A}} \pi(a \mid s) \operatorname{arcsinh}(\pi(a \mid s)/b) - \sqrt{\pi(a \mid s)^2 + b^2}, \tag{41}$$

and, following (14), the regression problem in Line 1 of Algorithm 1 becomes

$$\theta^{t+1} \in \operatorname*{argmin}_{\theta \in \Theta} \left\| f^\theta - Q^t - \eta_t^{-1} \max(\eta_{t-1} f^t, -\lambda_s^t) \right\|^2_{L_2(v^t)}.$$

Hyperbolic-tangent entropy. Inspired by the hyperbolic entropy, we consider ϕ as

$$\phi(x) = \tanh(x)/2 + 0.5$$

The mirror map h_ϕ becomes

$$h_\phi(\pi_s) = \frac{1}{2}\sum_{a\in\mathcal{A}}(2\pi(a \mid s) - 1)\operatorname{arctanh}(2\pi(a \mid s) - 1) + \frac{1}{2}\log \pi(a \mid s)(1 - \pi(a \mid s)), \quad (42)$$

which, to the best of our knowledge, is not equivalent to any mirror map studied in the literature. Following (14), the regression problem in Line 1 of Algorithm 1 becomes

$$\theta^{t+1} \in \operatorname*{argmin}_{\theta\in\Theta} \left\| f^\theta - Q^t - \frac{\eta_{t-1}}{\eta_t} f^t \right\|^2_{L_2(v^t)}.$$

Regarding the limitations of the ω-potential mirror map class, we are aware of two previously used mirror maps that cannot be recovered using ω-potentials: $h(x) = \frac{1}{2}x^\top \mathbf{A}x$ [32], for some positive-definite matrix $\mathbf{A}$, which generates the Mahalanobis distance, and p-norms, i.e. $h(x) = \|x\|_p^2$ [74]. Note that the case where $h(x) = \|x\|_p^p$ can be recovered by setting $\phi(x) = (px)^{p/(1-p)}$.

We note that tuning the mirror map and the step-size can lead AMPO to encompass the case of deterministic policies, which can be obtained when using softmax policies by sending the step-size to infinity, effectively turning the softmax operator into a max operator. Another simple way of introducing deterministic policies in our framework is to choose the mirror map to be the Euclidean norm and to choose the step-size large enough. Doing so will cause the Bregman projection to put all the probability on the action that corresponds to the maximum value of f_s^θ. Our results hold in this setting because our analysis does not use the policy gradient theorem (3), which has a different expression for deterministic policies [83].

D Deferred proofs from Section 4.1

D.1 Proof of Lemma 4.1

Here we provide the proof of Lemma 4.1, an application of the three-point descent lemma that accommodates arbitrary parameterized functions. Lemma 4.1 is the key tool for our analysis of AMPO. It is a generalization of both Xiao [93, Equation (44)] and Yuan et al. [98, Equation (50)] thanks to our two-step PMD framework. First, we recall some technical conditions of the mirror map [14, Chapter 4].

Suppose that $\mathcal{Y} \subset \mathbb{R}^{|\mathcal{A}|}$ is a closed convex set, we say a function $h : \mathcal{Y} \to \mathbb{R}$ is a *mirror map* if it satisfies the following properties:

(i) h is strictly convex and differentiable;

(ii) h is essentially smooth, i.e., the graident of h diverges on the boundary of $\mathcal{Y}$, that is $\lim_{x\to\partial\mathcal{Y}} \|\nabla h(x)\| \to \infty$;

(iii) the gradient of h takes all possible values, that is $\nabla h(\mathcal{Y}) = \mathbb{R}^{|\mathcal{A}|}$.

To prove Lemma 4.1, we also need the following rather simple properties, i.e., the three-point identity and the generalized Pythagorean theorem, satisfied by the Bregman divergence. We provide their proofs for self-containment.

Lemma D.1 (Three-point identity, Lemma 3.1 in Chen and Teboulle [19])**.** *Let h be a mirror map. For any a, b in the relative interior of $\mathcal{Y}$ and $c \in \mathcal{Y}$, we have that:*

$$\mathcal{D}_h(c, a) + \mathcal{D}_h(a, b) - \mathcal{D}_h(c, b) = \langle \nabla h(b) - \nabla h(a), c - a\rangle . \quad (43)$$

Proof. Using the definition of the Bregman divergence $\mathcal{D}_h$, we have

$$\langle \nabla h(a), c - a\rangle = h(c) - h(a) - \mathcal{D}_h(c, a), \quad (44)$$
$$\langle \nabla h(b), a - b\rangle = h(a) - h(b) - \mathcal{D}_h(a, b), \quad (45)$$
$$\langle \nabla h(b), c - b\rangle = h(c) - h(b) - \mathcal{D}_h(c, b). \quad (46)$$

Subtracting (44) and (45) from (46) yields (43). □

Lemma D.2 (Generalized Pythagorean Theorem of Bregman divergence, Lemma 4.1 in Bubeck [14]). *Let $\mathcal{X} \subseteq \mathcal{Y}$ be a closed convex set. Let h be a mirror map defined on $\mathcal{Y}$. Let $x \in \mathcal{X}$, $y \in \mathcal{Y}$ and $y^\star = \text{Proj}^h_{\mathcal{X}}(y)$, then*

$$\langle \nabla h\left(y^\star\right) - \nabla h(y), y^\star - x\rangle \leq 0,$$

which also implies

$$\mathcal{D}_h\left(x, y^\star\right) + \mathcal{D}_h\left(y^\star, y\right) \leq \mathcal{D}_h(x, y). \tag{47}$$

Proof. From the definition of $y^\star$, which is

$$y^\star \in \operatorname*{argmin}_{y' \in \mathcal{X}} \mathcal{D}_h(y', y),$$

and from the first-order optimality condition [14, Proposition 1.3], with

$$\nabla_{y'} \mathcal{D}_h(y', y) = \nabla h(y') - \nabla h(y), \quad \text{for all } y' \in \mathcal{Y},$$

we have

$$\langle \nabla_{y'} \mathcal{D}_h(y', y)|_{y'=y^\star}, y^\star - x\rangle \leq 0 \quad \Longrightarrow \quad \langle \nabla h\left(y^\star\right) - \nabla(y), y^\star - x\rangle \leq 0,$$

which implies (47) by applying the definition of Bregman divergence and rearranging terms. □

Now we are ready to prove Lemma 4.1.

Lemma D.3 (Lemma 4.1). *Let $\mathcal{Y} \subset \mathbb{R}^{|\mathcal{A}|}$ be a closed convex set with $\Delta(\mathcal{A}) \subseteq \mathcal{Y}$. For any policies $\pi \in \Delta(\mathcal{A})^{\mathcal{S}}$ and $\bar{\pi}$ in the relative interior of $\Delta(\mathcal{A})^{\mathcal{S}}$, any function f^θ with $\theta \in \Theta$, any $s \in \mathcal{S}$ and for $\eta > 0$, we have that,*

$$\langle \eta f^\theta_s - \nabla h(\bar{\pi}_s), \pi_s - \tilde{\pi}_s\rangle \leq \mathcal{D}_h(\pi_s, \bar{\pi}_s) - \mathcal{D}_h(\tilde{\pi}_s, \bar{\pi}_s) - \mathcal{D}_h(\pi, \tilde{\pi}_s),$$

where $\tilde{\pi}$ is induced by f^θ and η according to Definition 3.1, that is, for all $s \in \mathcal{S}$,

$$\tilde{\pi}_s = \text{Proj}^h_{\Delta(\mathcal{A})}\left(\nabla h^*(\eta f^\theta_s)\right) = \operatorname*{argmin}_{\pi'_s \in \Delta(\mathcal{A})} \mathcal{D}_h(\pi'_s, \nabla h^*(\eta f^\theta_s)). \tag{48}$$

Proof. For clarity of exposition, let $p_s = \nabla h^*(\eta f^\theta_s)$. Plugging $a = \bar{\pi}_s$, $b = p_s$ and $c = \pi_s$ in the three-point identity lemma D.1, we obtain

$$\mathcal{D}_h(\pi_s, \bar{\pi}_s) - \mathcal{D}_h(\pi_s, p_s) + \mathcal{D}_h(\bar{\pi}_s, p_s) = \langle \nabla h(\bar{\pi}_s) - \nabla h(p_s), \bar{\pi}_s - \pi_s\rangle. \tag{49}$$

Similarly, plugging $a = \bar{\pi}_s$, $b = p_s$ and $c = \tilde{\pi}_s$ in the three-point identity lemma D.1, we obtain

$$\mathcal{D}_h(\tilde{\pi}_s, \bar{\pi}_s) - \mathcal{D}_h(\tilde{\pi}_s, p_s) + \mathcal{D}_h(\bar{\pi}_s, p_s) = \langle \nabla h(\bar{\pi}_s) - \nabla h(p_s), \bar{\pi}_s - \tilde{\pi}_s\rangle. \tag{50}$$

From (49), we have

$$\begin{aligned}
& \mathcal{D}_h(\pi_s, \bar{\pi}_s) - \mathcal{D}_h(\pi_s, p_s) + \mathcal{D}_h(\bar{\pi}_s, p_s) \\
= \; & \langle \nabla h(\bar{\pi}_s) - \nabla h(p_s), \bar{\pi}_s - \pi_s\rangle \\
= \; & \langle \nabla h(\bar{\pi}_s) - \nabla h(p_s), \bar{\pi}_s - \tilde{\pi}_s\rangle + \langle \nabla h(\bar{\pi}_s) - \nabla h(p_s), \tilde{\pi}_s - \pi_s\rangle \\
\overset{(50)}{=} \; & \mathcal{D}_h(\tilde{\pi}_s, \bar{\pi}_s) - \mathcal{D}_h(\tilde{\pi}_s, p_s) + \mathcal{D}_h(\bar{\pi}_s, p_s) + \langle \nabla h(\bar{\pi}_s) - \nabla h(p_s), \tilde{\pi}_s - \pi_s\rangle.
\end{aligned}$$

By rearranging terms, we have

$$\mathcal{D}_h(\pi_s, \bar{\pi}_s) - \mathcal{D}_h(\tilde{\pi}_s, \bar{\pi}_s) - \mathcal{D}_h(\pi_s, p_s) + \mathcal{D}_h(\tilde{\pi}_s, p_s) = \langle \nabla h(\bar{\pi}_s) - \nabla h(p_s), \tilde{\pi}_s - \pi_s\rangle. \tag{51}$$

From the Generalized Pythagorean Theorem of the Bregman divergence in Lemma D.2, also known as non-expansivity property, and from the fact that $\tilde{\pi}_s = \text{Proj}^h_{\Delta(\mathcal{A})}(p_s)$, we have that

$$\mathcal{D}_h(\pi_s, \tilde{\pi}_s) + \mathcal{D}_h(\tilde{\pi}_s, p_s) \leq \mathcal{D}_h(\pi_s, p_s) \quad \Longleftrightarrow \quad -\mathcal{D}_h(\pi_s, p_s) + \mathcal{D}_h(\tilde{\pi}_s, p_s) \leq -\mathcal{D}_h(\pi_s, \tilde{\pi}_s).$$

Plugging the above inequality into the left hand side of (51) yields

$$\mathcal{D}_h(\pi_s, \bar{\pi}_s) - \mathcal{D}_h(\tilde{\pi}_s, \bar{\pi}_s) - \mathcal{D}_h(\pi_s, \tilde{\pi}_s) \geq \langle \nabla h(\bar{\pi}_s) - \nabla h(p_s), \tilde{\pi}_s - \pi_s\rangle,$$

which concludes the proof with $\nabla h(p_s) = \eta f^\theta_s$. □

We also provide an alternative proof of Lemma 4.1 later in Appendix F.

D.2 Bounding errors

In this section, we will bound error terms of the type

$$\mathbb{E}_{s\sim d_\mu^\pi, a\sim\pi_s}\left[Q^t(s,a) + \eta_t^{-1}[\nabla h(\pi_s^t)]_a - f^{t+1}(s,a)\right], \tag{52}$$

where $(d_\mu^\pi, \pi) \in \{(d_\mu^\star, \pi^\star), (d_\mu^{t+1}, \pi^{t+1}), (d_\mu^\star, \pi^t), (d_\mu^{t+1}, \pi^t)\}$. These error terms appear in the forthcoming proofs of our results and directly induce the error floors in the convergence rates.

In the rest of Appendix D, let $q^t : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$ such that, for every $s \in \mathcal{S}$,

$$q_s^t := f_s^{t+1} - \eta_t^{-1}\nabla h(\pi_s^t) \in \mathbb{R}^{|\mathcal{A}|}.$$

So (52) can be rewritten as

$$\mathbb{E}_{s\sim d_\mu^\pi, a\sim\pi_s}\left[Q^t(s,a) + \eta_t^{-1}[\nabla h(\pi_s^t)]_a - f^{t+1}(s,a)\right] = \mathbb{E}_{s\sim d_\mu^\pi, a\sim\pi_s}\left[Q^t(s,a) - q^t(s,a)\right]. \tag{53}$$

To bound it, let $(v^t)_{t\geq 0}$ be a sequence of distributions over states and actions. By using Cauchy-Schwartz's inequality, we have

$$\begin{aligned}
&\mathbb{E}_{s\sim d_\mu^\pi, a\sim\pi_s}\left[Q^t(s,a) - q^t(s,a)\right] \\
&\quad= \int_{s\in\mathcal{S}, a\in\mathcal{A}} \frac{d_\mu^\pi(s)\pi(a\mid s)}{\sqrt{v^t(s,a)}} \cdot \sqrt{v^t(s,a)}(Q^t(s,a) - q^t(s,a)) \\
&\quad\leq \sqrt{\int_{s\in\mathcal{S}, a\in\mathcal{A}} \frac{\left(d_\mu^\pi(s)\pi(a\mid s)\right)^2}{v^t(s,a)} \cdot \int_{s\in\mathcal{S}, a\in\mathcal{A}} v^t(s,a)(Q^t(s,a) - q^t(s,a))^2} \\
&\quad= \sqrt{\mathbb{E}_{(s,a)\sim v^t}\left[\left(\frac{d_\mu^\pi(s)\pi(a\mid s)}{v^t(s,a)}\right)^2\right] \cdot \mathbb{E}_{(s,a)\sim v^t}\left[(Q^t(s,a) - q^t(s,a))^2\right]} \\
&\quad\leq \sqrt{C_v \mathbb{E}_{(s,a)\sim v^t}\left[(Q^t(s,a) - q^t(s,a))^2\right]},
\end{aligned}$$

where the last line is obtained by Assumption (A2). Using the concavity of the square root and Assumption (A1), we have that

$$\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\pi, a\sim\pi_s}\left[Q^t(s,a) - q^t(s,a)\right]\right] \leq \sqrt{C_v \varepsilon_{\text{approx}}}. \tag{54}$$

D.3 Quasi-monotonic updates – Proof of Proposition 4.2

In this section, we show Proposition 4.2 with its proof that the AMPO updates guarantee a quasi-monotonic property, i.e., a non-decreasing property up to a certain error floor due to the approximation error, which allows us to establish an important recursion about the AMPO iterates next. First, we recall the performance difference lemma [41] which is the second key tool for our analysis and a well known result in the RL literature. Here we use a particular form of the lemma presented by Xiao [93, Lemma 1].

Lemma D.4 (Performance difference lemma, Lemma 1 in [93])**.** *For any policy $\pi, \pi' \in \Delta(\mathcal{A})^{\mathcal{S}}$ and $\mu \in \Delta(\mathcal{S})$,*

$$V^\pi(\mu) - V^{\pi'}(\mu) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\mu^\pi}\left[\left\langle Q_s^{\pi'}, \pi_s - \pi_s'\right\rangle\right].$$

For clarity of exposition, we introduce the notation

$$\tau := \frac{2\sqrt{C_v \varepsilon_{\text{approx}}}}{1-\gamma}.$$

Proposition 4.2 characterizes the non-decreasing property of AMPO. The error bound (54) in Appendix D.2 will be used to prove the the result.

Proposition D.5 (Proposition 4.2)**.** *For the iterates of Algorithm 1, at each time $t \geq 0$, we have*

$$\mathbb{E}[V^{t+1}(\mu) - V^t(\mu)] \geq \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\frac{\mathcal{D}_h(\pi_s^{t+1}, \pi_s^t) + \mathcal{D}_h(\pi_s^t, \pi_s^{t+1})}{\eta_t(1-\gamma)}\right]\right] - \tau.$$

Proof. Using Lemma 4.1 with $\bar{\pi} = \pi^t$, $f^\theta = f^{t+1}$, $\eta = \eta_t$, thus $\tilde{\pi} = \pi^{t+1}$ by Definition 3.1 and Algorithm 1, and $\pi_s = \pi_s^t$, we have

$$\langle \eta_t q_s^t, \pi_s^t - \pi_s^{t+1}\rangle \leq \mathcal{D}_h(\pi_s^t, \pi_s^t) - \mathcal{D}_h(\pi_s^{t+1}, \pi_s^t) - \mathcal{D}_h(\pi_s^t, \pi_s^{t+1}). \tag{55}$$

By rearranging terms and noticing $\mathcal{D}_h(\pi_s^t, \pi_s^t) = 0$, we have

$$\langle \eta_t q_s^t, \pi_s^{t+1} - \pi_s^t\rangle \geq \mathcal{D}_h(\pi_s^{t+1}, \pi_s^t) + \mathcal{D}_h(\pi_s^t, \pi_s^{t+1}) \geq 0. \tag{56}$$

Then, by the performance difference lemma D.4, we have

$$\begin{aligned}
(1-\gamma)\mathbb{E}[V^{t+1}(\mu) - V^t(\mu)] &= \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\langle Q_s^t, \pi_s^{t+1} - \pi_s^t\rangle\right]\right] \\
&= \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\langle q_s^t, \pi_s^{t+1} - \pi_s^t\rangle\right]\right] \\
&\quad + \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\langle Q_s^t - q_s^t, \pi_s^{t+1} - \pi_s^t\rangle\right]\right] \\
&\overset{(55)}{\geq} \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\frac{\mathcal{D}_h(\pi_s^{t+1}, \pi_s^t) + \mathcal{D}_h(\pi_s^t, \pi_s^{t+1})}{\eta_t}\right]\right] \\
&\quad - \left|\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\langle Q_s^t - q_s^t, \pi_s^{t+1} - \pi_s^t\rangle\right]\right]\right| \\
&\geq \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\frac{\mathcal{D}_h(\pi_s^{t+1}, \pi_s^t) + \mathcal{D}_h(\pi_s^t, \pi_s^{t+1})}{\eta_t}\right]\right] - \tau(1-\gamma),
\end{aligned}$$

which concludes the proof after dividing both sides by $(1-\gamma)$. The last line follows from

$$\begin{aligned}
\left|\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\langle Q_s^t - q_s^t, \pi_s^{t+1} - \pi_s^t\rangle\right]\right]\right| &\leq \left|\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}, a\sim\pi_s^{t+1}}\left[Q^t(s,a) - q^t(s,a)\right]\right]\right| \\
&\quad + \left|\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}, a\sim\pi_s^{t}}\left[Q^t(s,a) - q^t(s,a)\right]\right]\right| \qquad (57) \\
&\overset{(54)}{\leq} 2\sqrt{C_1\varepsilon_{\text{error}}} = \tau(1-\gamma), \qquad (58)
\end{aligned}$$

where both terms are upper bounded by $\sqrt{C_v\varepsilon_{\text{approx}}}$ through (54) with $(d_\mu^\pi, \pi) = (d_\mu^{t+1}, \pi^{t+1})$ and $(d_\mu^\pi, \pi) = (d_\mu^{t+1}, \pi^t)$, respectively. □

D.4 Main passage – An important recursion about the AMPO method

In this section, we show an important recursion result for the AMPO updates, which will be used for both the sublinear and the linear convergence analysis of AMPO.

For clarity of exposition in the rest of Appendix D, let

$$\nu_t := \left\|\frac{d_\mu^\star}{d_\mu^{t+1}}\right\|_{L_\infty} := \sup_{s\in\mathcal{S}} \frac{d_\mu^\star(s)}{d_\mu^{t+1}(s)}.$$

For two different time $t, t' \geq 0$, let $\mathcal{D}_{t'}^t$ denote the expected Bregman divergence between the policy π^t and policy $\pi^{t'}$, where the expectation is taken over the discounted state visitation distribution of the optimal policy $d_\mu^\star$, that is,

$$\mathcal{D}_{t'}^t := \mathbb{E}_{s\sim d_\mu^\star}\left[\mathcal{D}_h(\pi_s^t, \pi_s^{t'})\right].$$

Similarly, let $\mathcal{D}_t^\star$ denote the expected Bregman divergence between the optimal policy $\pi^\star$ and π^t, that is,

$$\mathcal{D}_t^\star := \mathbb{E}_{s\sim d_\mu^\star}\left[\mathcal{D}_h(\pi_s^\star, \pi_s^t)\right].$$

Let $\Delta_t := V^\star(\mu) - V^t(\mu)$ be the optimality gap.

We can now state the following important recursion result for the AMPO method.

Proposition D.6 (Proposition 4.4). *Consider the iterates of Algorithm 1, at each time $t \geq 0$, we have*

$$\mathbb{E}\left[\frac{\mathcal{D}_t^{t+1}}{(1-\gamma)\eta_t} + \nu_\mu\left(\Delta_{t+1} - \Delta_t\right) + \Delta_t\right] \leq \mathbb{E}\left[\frac{\mathcal{D}_t^\star}{(1-\gamma)\eta_t} - \frac{\mathcal{D}_{t+1}^\star}{(1-\gamma)\eta_t}\right] + (1+\nu_\mu)\tau.$$

Proof. Using Lemma 4.1 with $\bar{\pi} = \pi^t$, $f^\theta = f^{t+1}$, $\eta = \eta_t$, and thus $\tilde{\pi} = \pi^{t+1}$ by Definition 3.1 and Algorithm 1, and $\pi_s = \pi_s^\star$, we have that

$$\langle \eta_t q_s^t, \pi_s^\star - \pi_s^{t+1}\rangle \leq \mathcal{D}_h(\pi^\star, \pi^t) - \mathcal{D}_h(\pi^\star, \pi^{t+1}) - \mathcal{D}_h(\pi^{t+1}, \pi^t),$$

which can be decomposed as

$$\langle \eta_t q_s^t, \pi_s^t - \pi_s^{t+1}\rangle + \langle \eta_t q_s^t, \pi_s^\star - \pi_s^t\rangle \leq \mathcal{D}_h(\pi^\star, \pi^t) - \mathcal{D}_h(\pi^\star, \pi^{t+1}) - \mathcal{D}_h(\pi^{t+1}, \pi^t).$$

Taking expectation with respect to the distribution $d_\mu^\star$ over states and with respect to the randomness of AMPO and dividing both sides by η_t, we have

$$\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle q_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] + \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle q_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right] \leq \frac{1}{\eta_t}\mathbb{E}[\mathcal{D}_t^\star - \mathcal{D}_{t+1}^\star - \mathcal{D}_t^{t+1}]. \quad (59)$$

We lower bound the two terms on the left hand side of (59) separately. For the first term, we have that

$$\begin{aligned}
\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle q_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] &\overset{(56)}{\geq} \left\|\frac{d_\mu^\star}{d_\mu^{t+1}}\right\|_{L_\infty}^{-1} \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\langle q_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] \\
&= \nu_{t+1}\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\langle Q_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] \\
&\quad + \nu_{t+1}\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\langle q_s^t - Q_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] \\
&\overset{(a)}{=} \nu_{t+1}(1-\gamma)\mathbb{E}\left[V^t(\mu) - V^{t+1}(\mu)\right] \\
&\quad + \nu_{t+1}\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^{t+1}}\left[\langle q_s^t - Q_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] \\
&\overset{(57)}{\geq} \nu_{t+1}(1-\gamma)\mathbb{E}\left[V^t(\mu) - V^{t+1}(\mu)\right] - \nu_{t+1}\tau(1-\gamma) \\
&= \nu_{t+1}(1-\gamma)\mathbb{E}\left[\Delta_{t+1} - \Delta_t\right] - \nu_{t+1}\tau(1-\gamma),
\end{aligned}$$

where (a) follows from Lemma D.4. For the second term, we have that

$$\begin{aligned}
\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle q_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right] &= \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle Q_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right] + \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle q_s^t - Q_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right] \\
&\overset{(b)}{=} \mathbb{E}[\Delta_t](1-\gamma) + \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle q_s^t - Q_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right] \\
&\overset{(c)}{\geq} \mathbb{E}[\Delta_t](1-\gamma) - \tau(1-\gamma),
\end{aligned}$$

where (b) follows from Lemma D.4 and (c) follows similarly to (57), i.e., by applying (54) twice with $(d_\mu^\pi, \pi) = (d_\mu^\star, \pi^\star)$ and $(d_\mu^\pi, \pi) = (d_\mu^\star, \pi^t)$.

Plugging the two bounds in (59), dividing both sides by $(1-\gamma)$ and rearranging, we obtain

$$\mathbb{E}\left[\frac{\mathcal{D}_t^{t+1}}{(1-\gamma)\eta_t} + \nu_{t+1}\left(\Delta_{t+1} - \Delta_t - \tau\right) + \Delta_t\right] \leq \mathbb{E}\left[\frac{\mathcal{D}_t^\star}{(1-\gamma)\eta_t} - \frac{\mathcal{D}_{t+1}^\star}{(1-\gamma)\eta_t}\right] + \tau.$$

From Proposition 4.2, we have that $\Delta_{t+1} - \Delta_t - \tau \leq 0$. Consequently, since $\nu_{t+1} \leq \nu_\mu$ by the definition of ν_μ in Assumption (A3), one can lower bound the left hand side of the above inequality by replacing ν_{t+1} by ν_μ, that is,

$$\mathbb{E}\left[\frac{\mathcal{D}_t^{t+1}}{(1-\gamma)\eta_t} + \nu_\mu\left(\Delta_{t+1} - \Delta_t - \tau\right) + \Delta_t\right] \leq \mathbb{E}\left[\frac{\mathcal{D}_t^\star}{(1-\gamma)\eta_t} - \frac{\mathcal{D}_{t+1}^\star}{(1-\gamma)\eta_t}\right] + \tau,$$

which concludes the proof. □

D.5 Proof of the sublinear convergence analysis

In this section, we derive the sublinear convergence result of Theorem 4.3 with non-decreasing step-size.

Proof. Starting from Proposition D.6

$$\mathbb{E}\left[\frac{\mathcal{D}_t^{t+1}}{(1-\gamma)\eta_t}+\nu_\mu\left(\Delta_{t+1}-\Delta_t\right)+\Delta_t\right]\leq\mathbb{E}\left[\frac{\mathcal{D}_t^\star}{(1-\gamma)\eta_t}-\frac{\mathcal{D}_{t+1}^\star}{(1-\gamma)\eta_t}\right]+(1+\nu_\mu)\tau.$$

If $\eta_t\leq\eta_{t+1}$,

$$\mathbb{E}\left[\frac{\mathcal{D}_t^{t+1}}{(1-\gamma)\eta_t}+\nu_\mu\left(\Delta_{t+1}-\Delta_t\right)+\Delta_t\right]\leq\mathbb{E}\left[\frac{\mathcal{D}_t^\star}{(1-\gamma)\eta_t}-\frac{\mathcal{D}_{t+1}^\star}{(1-\gamma)\eta_{t+1}}\right]+(1+\nu_\mu)\tau. \tag{60}$$

Summing up from 0 to $T-1$ and dropping some positive terms on the left hand side and some negative terms on the right hand side, we have

$$\sum_{t<T}\mathbb{E}\left[\Delta_t\right]\leq\frac{\mathcal{D}_0^\star}{(1-\gamma)\eta_0}+\nu_\mu\Delta_0+T(1+\nu_\mu)\tau\leq\frac{\mathcal{D}_0^\star}{(1-\gamma)\eta_0}+\frac{\nu_\mu}{1-\gamma}+T(1+\nu_\mu)\tau.$$

Notice that $\Delta_0\leq\frac{1}{1-\gamma}$ as $r(s,a)\in[0,1]$. By dividing T on both side, we yield the proof of the sublinear convergence

$$V^\star(\mu)-\frac{1}{T}\sum_{t<T}\mathbb{E}\left[V^t(\mu)\right]\leq\frac{1}{T}\left(\frac{\mathcal{D}_0^\star}{(1-\gamma)\eta_0}+\frac{\nu_\mu}{1-\gamma}\right)+(1+\nu_\mu)\tau.$$

□

D.6 Proof of the linear convergence analysis

In this section, we derive the linear convergence result of Theorem 4.3 with exponentially increasing step-size.

Proof. Starting from Proposition D.6 by dropping $\frac{\mathcal{D}_t^{t+1}}{(1-\gamma)\eta_t}$ on the left hand side, we have

$$\mathbb{E}\left[\nu_\mu\left(\Delta_{t+1}-\Delta_t\right)+\Delta_t\right]\leq\mathbb{E}\left[\frac{\mathcal{D}_t^\star}{(1-\gamma)\eta_t}-\frac{\mathcal{D}_{t+1}^\star}{(1-\gamma)\eta_t}\right]+(1+\nu_\mu)\tau.$$

Dividing ν_μ on both side and rearranging, we obtain

$$\mathbb{E}\left[\Delta_{t+1}+\frac{\mathcal{D}_{t+1}^\star}{(1-\gamma)\nu_\mu\eta_t}\right]\leq\left(1-\frac{1}{\nu_\mu}\right)\mathbb{E}\left[\Delta_t+\frac{\mathcal{D}_t^\star}{(1-\gamma)\eta_t(\nu_\mu-1)}\right]+\left(1+\frac{1}{\nu_\mu}\right)\tau.$$

If the step-sizes satisfy $\eta_{t+1}(\nu_\mu-1)\geq\eta_t\nu_\mu$ with $\nu_\mu\geq1$, then

$$\mathbb{E}\left[\Delta_{t+1}+\frac{\mathcal{D}_{t+1}^\star}{(1-\gamma)\eta_{t+1}(\nu_\mu-1)}\right]\leq\left(1-\frac{1}{\nu_\mu}\right)\mathbb{E}\left[\Delta_t+\frac{\mathcal{D}_t^\star}{(1-\gamma)\eta_t(\nu_\mu-1)}\right]+\left(1+\frac{1}{\nu_\mu}\right)\tau.$$

Now we need the following simple fact, whose proof is straightforward and thus omitted.

Suppose $0<\alpha<1, b>0$ and a nonnegative sequence $\{a_t\}_{t\geq0}$ satisfies

$$a_{t+1}\leq\alpha a_t+b\qquad\forall t\geq0.$$

Then for all $t\geq0$,

$$a_t\leq\alpha^t a_0+\frac{b}{1-\alpha}.$$

The proof of the linear convergence analysis follows by applying this fact with $a_t=\mathbb{E}\left[\Delta_t+\frac{\mathcal{D}_t^\star}{(1-\gamma)\eta_t(\nu_\mu-1)}\right]$, $\alpha=1-\frac{1}{\nu_\mu}$ and $b=\left(1+\frac{1}{\nu_\mu}\right)\tau$. □

E Discussion of the first step (Line 1) of AMPO – the compatible function approximation framework

Starting from this section, some additional remarks about AMPO are in order. In particular, we discuss in detail the novelty of the first step (Line 1) and the second step (Line 2) of AMPO in this and the next section, respectively. Afterwards, we provide an extensive justification of the assumptions used in Theorem 4.3 in Appendices G to I.

As mentioned in Remark 3.3, Agarwal et al. [1] study NPG with smooth policies through compatible function approximation and propose the following algorithm. Let $\{\pi^\theta : \theta \in \Theta\}$ be a policy class such that $\log \pi^\theta(a \mid s)$ is a β-smooth function of θ for all $s \in \mathcal{S}$, $a \in \mathcal{A}$. At each iteration t, update

$$\theta^{t+1} = \theta^t + \eta w^t,$$

with

$$w^t \in \operatorname*{argmin}_{\|w\|_2 \leq W} \left\| A^t - w^\top \nabla_\theta \log \pi^t \right\|_{L_2(d_\mu^t \cdot \pi^t)}, \tag{61}$$

where $W > 0$ and $A^t(s,a) = Q^t(s,a) - V^t(s)$ represents the advantage function. While both the algorithm proposed by Agarwal et al. [1] and AMPO involve regression problems, the one in (61) is restricted to linearly approximate A^t with $\nabla_\theta \log \pi^t$, whereas the one in Line 1 of Algorithm 1 is relaxed to approximate A^t with an arbitrary class of functions $\mathcal{F}^\Theta$. Additionally, (61) depends on the distribution d_μ^t, while Line 1 of Algorithm 1 does not and allows off-policy updates involving an arbitrary distribution v^t, as v^t is independent of the current policy π^t.

F Discussion of the second step (Line 2) of AMPO – the Bregman projection

As mentioned in Remark 3.4, we can rewrite the second step (Line 2) of AMPO through the following lemma.

Lemma F.1. *For any policy $\bar{\pi}$, for any function $f^\theta \in \mathcal{F}^\Theta$ and for $\eta > 0$, we have, for all $s \in \mathcal{S}$,*

$$\tilde{\pi}_s \in \operatorname*{argmin}_{p \in \Delta(\mathcal{A})} \mathcal{D}_h(p, \nabla h^*(\eta f_s^\theta)) \iff \tilde{\pi}_s \in \operatorname*{argmin}_{p \in \Delta(\mathcal{A})} \langle -\eta f_s^\theta + \nabla h(\bar{\pi}_s), p \rangle + \mathcal{D}_h(p, \bar{\pi}_s).$$

Equations (12) and (20) are obtained by choosing $\tilde{\pi}_s = \pi_s^{t+1}$ and $\bar{\pi}_s = \pi_s^t$ for all $s \in \mathcal{S}$, $\eta = \eta_t$, $\theta = \theta^{t+1}$, and by changing the sign of the expression on the right in order to obtain an argmax.

Proof. Starting from the definition of $\tilde{\pi}$, we have

$$\begin{aligned}
\tilde{\pi}_s &\in \operatorname*{argmin}_{p \in \Delta(\mathcal{A})} \mathcal{D}_h(p, \nabla h^*(\eta f_s^\theta)) \\
&\in \operatorname*{argmin}_{p \in \Delta(\mathcal{A})} h(p) - h(\nabla h^*(\eta f_s^\theta)) - \langle \nabla h(\nabla h^*(\eta f_s^\theta)), p - \nabla h^*(\eta f_s^\theta) \rangle \\
&\in \operatorname*{argmin}_{p \in \Delta(\mathcal{A})} h(p) - \langle \eta f_s^\theta, p \rangle \\
&\in \operatorname*{argmin}_{p \in \Delta(\mathcal{A})} \langle -\eta f_s^\theta + \nabla h(\bar{\pi}_s), p \rangle + h(p) - h(\bar{\pi}_s) - \langle \nabla h(\bar{\pi}_s), p - \bar{\pi}_s \rangle \\
&\in \operatorname*{argmin}_{p \in \Delta(\mathcal{A})} \langle -\eta f_s^\theta + \nabla h(\bar{\pi}_s), p \rangle + \mathcal{D}_h(p, \bar{\pi}_s),
\end{aligned} \tag{62}$$

where the second and the last lines are obtained using the definition of the Bregman divergence, and the third line is obtained using (4) ($\nabla h(\nabla h^*(x^*)) = x^*$ for all $x^* \in \mathbb{R}^{|\mathcal{A}|}$). □

Lemmas F.1 and B.1 share a similar result, as they both rewrite the Bregman projection into the MD updates. However, the MD updates in Lemma B.1 are exact, while the MD updates in AMPO involve approximation (Line 1).

Next, we provide an alternative proof for Lemma 4.1 to show that it is the direct consequence of Lemma F.1. The proof will involve the application of the three-point descent lemma [19, Lemma 3.2]. Here we adopt its slight variation by following Lemma 6 in Xiao [93].

Lemma F.2 (Three-point decent lemma, Lemma 6 in Xiao [93]). *Suppose that $\mathcal{C} \subset \mathbb{R}^m$ is a closed convex set, $f : \mathcal{C} \to \mathbb{R}$ is a proper, closed* [6] *convex function, $\mathcal{D}_h(\cdot, \cdot)$ is the Bregman divergence generated by a mirror map h. Denote* $\operatorname{rint}\operatorname{dom} h$ *as the relative interior of* $\operatorname{dom} h$. *For any $x \in \operatorname{rint}\operatorname{dom} h$, let*

$$x^+ \in \arg\min_{u \in \operatorname{dom} h \cap \mathcal{C}} \{f(u) + \mathcal{D}_h(u, x)\}.$$

Then $x^+ \in \operatorname{rint}\operatorname{dom} h \cap \mathcal{C}$ and for any $u \in \operatorname{dom} h \cap \mathcal{C}$,

$$f(x^+) + \mathcal{D}_h(x^+, x) \leq f(u) + \mathcal{D}_h(u, x) - \mathcal{D}_h(u, x^+).$$

We refer to Yuan et al. [98, Lemma 11] for a proof of Lemma F.2.

Lemma 4.1 is obtained by simply applying the three-point descent lemma, Lemma F.2, to (62) with $x^+ = \tilde{\pi}_s$, $f(u) = \langle -\eta f_s^\theta + \nabla h(\bar{\pi}_s), u \rangle$, $u = \pi$ and $x = \bar{\pi}_s$ and rearranging terms.

In contrast, it may not be possible to apply Lemma F.2 to (11), as $\Pi(\Theta)$ is often non-convex.

G Discussion on Assumption (A1) – the approximation error

The compatible function approximation approach [1, 61, 15, 21, 2, 98] has been introduced to deal with large state and action spaces, in order to reduce the dimension of the problem and make the computation feasible. As mentioned in the *proof idea* in Page 8, this framework consists in upper-bounding the sub-optimality gap with an optimization error plus an approximation error. Consequently, it is important that both error terms converge to 0 in order to achieve convergence to a global optimum.

Assumptions similar to Assumption (A1) are common in the compatible function approximation literature. Assumption (A1) encodes a form of realizability assumption for the parameterization class $\mathcal{F}^\Theta$, that is, we assume that for all $t \leq T$ there exists a function $f^\theta \in \mathcal{F}^\Theta$ such that

$$\left\| f^\theta - Q^t - \eta_t^{-1} \nabla h(\pi^t) \right\|_{L_2(v^t)}^2 \leq \varepsilon_{\text{approx}}.$$

When $\mathcal{F}^\Theta$ is a class of sufficiently large shallow neural networks, this realizability assumption holds as it has been shown that shallow neural networks are universal approximators [39]. It is, however, possible to relax Assumption (A1). In particular, the condition

$$\frac{1}{T} \sum_{t<T} \sqrt{\mathbb{E}\left[\mathbb{E} \left\| f^{t+1} - Q^t - \eta_t^{-1} \nabla h(\pi^t) \right\|_{L_2(v^t)}^2 \right]} \leq \sqrt{\varepsilon_{\text{approx}}} \tag{63}$$

can replace Assumption (A1) and is sufficient for the sublinear convergence rate in Theorem 4.3 to hold. Equation (63) shows that the realizability assumption does not need to hold for all $t < T$, but only needs to hold on average over T iterations. Similarly, the condition

$$\sum_{t \leq T} \left(1 - \frac{1}{\nu_\mu}\right)^{T-t} \frac{1}{\nu_\mu} \sqrt{\mathbb{E}\left[\mathbb{E} \left\| f^{t+1} - Q^t - \eta_t^{-1} \nabla h(\pi^t) \right\|_{L_2(v^t)}^2 \right]} \leq \sqrt{\varepsilon_{\text{approx}}} \tag{64}$$

can replace Assumption (A1) and is sufficient for the linear convergence rate in Theorem 4.3 to hold. Additionally, requiring, for all $t < T$,

$$\mathbb{E}\left[\mathbb{E} \left\| f^{t+1} - Q^t - \eta_t^{-1} \nabla h(\pi^t) \right\|_{L_2(v^t)}^2 \right] \leq \frac{\nu_\mu^2}{T^2} \left(1 - \frac{1}{\nu_\mu}\right)^{-2(T-t)} \varepsilon_{\text{approx}} \tag{65}$$

is sufficient for Equation (64) to hold. Equation (65) shows that the error floor in the linear convergence rate is less influenced by approximation errors made in early iterations, which are discounted by the term $\left(1 - \frac{1}{\nu_\mu}\right)$. On the other hand, the realizability assumption becomes relevant once the algorithm approaches convergence, i.e., when $t \simeq T$ and $Q^t \simeq Q^\star$, as the discount term $\left(1 - \frac{1}{\nu_\mu}\right)$ is applied fewer times.

Finally, although Assumption (A1) holds for the softmax tabular policies and for the neural network parameterization, it remains an open question whether Assumption (A1) is necessary to achieve the global optimum convergence, especially when the representation power of $\mathcal{F}^\Theta$ cannot guarantee a small approximation error.

[6] A convex function f is proper if $\operatorname{dom} f$ is nonempty and for all $x \in \operatorname{dom} f$, $f(x) > -\infty$. A convex function is closed, if it is lower semi-continuous.

H Discussion on Assumption (A2) – the concentrability coefficients

In our convergence analysis, Assumptions (A2) and (A3) involve the concentrability coefficient C_v and the distribution mismatch coefficient ν_μ, which are potentially large. We give extensive discussions on them in this and the next section, respectively.

As discussed in Yuan et al. [98, Appendix H], the issue of having (potentially large) concentrability coefficient (Assumptions (A2)) is unavoidable in all the fast linear convergence analysis of approximate PMD due to the approximation error $\varepsilon_{\text{approx}}$ of the Q-function [17, 100, 54, 16, 93, 20, 2, 98]. Indeed, in the fast linear convergence analysis of PMD, the concentrability coefficient is always along with the approximation error $\varepsilon_{\text{approx}}$ under the form of $C_v\varepsilon_{\text{approx}}$, which is the case in Theorem 4.3. To not get the concentrability coefficient involved yet maintain the linear convergence of PMD, one needs to consider the exact PMD in the tabular setting [see 93, Theorem 10]. Consequently, the PMD update is deterministic and the full policy space $\Delta(\mathcal{A})^{\mathcal{S}}$ is considered. In this setting, at each time t, it exists θ^{t+1} such that, for any state-action distribution v^t,

$$\left\| f^{t+1} - Q^t - \eta_t^{-1}\nabla h(\pi^t) \right\|^2_{L_2(v^t)} = 0 = \varepsilon_{\text{approx}},$$

and C_v is ignored in the convergence analysis thanks to the vanishing of $\varepsilon_{\text{approx}}$. We note that the PMD analysis in the seminal paper by Agarwal et al. [1] does not use such a coefficient, but a condition number instead. The condition number is controllable to be relatively small, so that the error term in their PMD analysis is smaller than ours. However, their PMD analysis has only a sublinear convergence rate, while ours enjoys a fast linear convergence rate. It remains an open question whether one can both avoid using the concentrability coefficient and maintain the linear convergence of PMD.

Now we compare our concentrability coefficient C_v with others used in the fast linear convergence analysis of approximate PMD [17, 100, 54, 15, 93, 20, 2, 98]. To the best of our knowledge, the previously best-known concentrability coefficient C_v was the one used by Yuan et al. [98, Appendix H]. As they discuss, their concentrability coefficient involved the weakest assumptions on errors among Lan [54], Xiao [93] and Chen and Theja Maguluri [20] by using the L_2-norm instead of the ℓ_∞-norm over the approximation error $\varepsilon_{\text{approx}}$. Additionally, it did not impose any restrictions on the MDP dynamics compared to Cayci et al. [15], as the concentrability coefficient of Yuan et al. [98] was independent from the iterates.

Indeed, Yuan et al. [98] choose v^t such that, for all $(s,a) \in \mathcal{S} \times \mathcal{A}$,

$$v^t(s,a) = (1-\gamma)\,\mathbb{E}_{(s_0,a_0)\sim\nu}\left[\sum_{t'=0}^{\infty}\gamma^{t'}P(s_{t'}=s, a_{t'}=a \mid \pi^t, s_0, a_0)\right],$$

where ν is an initial state-action distribution chosen by the user. In this setting, we have

$$v^t(s,a) \geq (1-\gamma)\nu(s,a).$$

From the above lower bound of v^t, we obtain that

$$\begin{aligned}\mathbb{E}_{(s,a)\sim v^t}\left[\left(\frac{d_\mu^\pi(s)\pi(a \mid s)}{v^t(s,a)}\right)^2\right] &= \int_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{d_\mu^\pi(s)^2\pi(a \mid s)^2}{v^t(s,a)} \\ &\leq \int_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{1}{v^t(s,a)} \leq \frac{1}{(1-\gamma)\min_{(s,a)\in\mathcal{S}\times\mathcal{A}}\nu(s,a)},\end{aligned}$$

where the finite upper bound is independent to t.

As mentioned right after Assumption (A2), the assumption on our concentrability coefficient C_v is weaker than the one in Yuan et al. [98, Assumption 9], as we have the full control over v^t while Yuan et al. [98] only has the full control over the initial state-action distribution ν. In particular, our concentrability coefficient C_v recovers the previous best-known one in Yuan et al. [98] as a special case. Consequently, our concentrability coefficient C_v becomes the "best" with the full control over v^t when other concentrability coefficients are infinite or require strong assumptions [77].

In general, for the ratio $\mathbb{E}_{(s,a)\sim v^t}\left[\left(\frac{d_\mu^\pi(s)\pi(a|s)}{v^t(s,a)}\right)^2\right]$ to have a finite upper bound C_v, it is important that v^t covers well the state and action spaces so that the upper bound is independent to t. However, the

upper bound $\frac{1}{(1-\gamma)\min_{(s,a)\in\mathcal{S}\times\mathcal{A}}\nu(s,a)}$ in Yuan et al. [98] is very pessimistic. Indeed, when π^t and π^{t+1} converge to $\pi^\star$, one reasonable choice of v^t is to choose $v^t \in \{d_\mu^\star \cdot \pi^\star, d_\mu^{t+1} \cdot \pi^{t+1}, d_\mu^\star \cdot \pi^t, d_\mu^{t+1} \cdot \pi^t\}$ such that C_v is close to 1.

We also refer to Yuan et al. [98, Appendix H] for more discussions on the concentrability coefficient.

I Discussion on Assumption (A3) – the distribution mismatch coefficients

In this section, we give further insights on the distribution mismatch coefficient ν_μ in Assumption (A3). As mentioned right after (A3), we have that

$$\sup_{s\in\mathcal{S}} \frac{d_\mu^\star(s)}{d_\mu^t(s)} \leq \frac{1}{1-\gamma} \sup_{s\in\mathcal{S}} \frac{d_\mu^\star(s)}{\mu(s)} := \nu'_\mu,$$

which is a sufficient upper bound for ν_μ. As discussed in Yuan et al. [98, Appendix H],

$$1/(1-\gamma) \leq \nu'_\mu \leq 1/((1-\gamma)\min_s \mu(s)).$$

The upper bound $1/((1-\gamma)\min_s \mu(s))$ of ν'_μ is very pessimistic and the lower bound $\nu'_\mu = 1/(1-\gamma)$ is often achieved by choosing $\mu = d_\mu^\star$.

Furthermore, if μ does not have full support on the state space, i.e., the upper bound $1/((1-\gamma)\min_s \mu(s))$ might be infinite, one can always convert the convergence guarantees for some state distribution $\mu' \in \Delta(\mathcal{S})$ with full support such that

$$\begin{aligned} V^\star(\mu) - \mathbb{E}[V^T(\mu)] &= \mathbb{E}\left[\int_{s\in\mathcal{S}} \frac{\mu(s)}{\mu'(s)} \mu'(s) \left(V^\star(s) - V^T(s)\right)\right] \\ &\leq \sup_{s\in\mathcal{S}} \frac{\mu(s)}{\mu'(s)} \left(V^\star(\mu') - \mathbb{E}[V^T(\mu')]\right). \end{aligned}$$

Then by the linear convergence result of Theorem 4.3, we only transfer the original convergence guarantee to $V^\star(\mu') - \mathbb{E}[V^T(\mu')]$ up to a scaling factor $\sup_{s\in\mathcal{S}} \frac{\mu(s)}{\mu'(s)}$ with an arbitrary distribution μ' such that ν'_μ is finite.

Finally, if d_μ^t converges to $d_\mu^\star$ which is the case of AMPO through the proof of our Theorem 4.3, then $\sup_{s\in\mathcal{S}} \frac{d_\mu^\star(s)}{d_\mu^t(s)}$ converges to 1. This might imply superlinear convergence results as discussed in Xiao [93, Section 4.3]. In this case, the notion of the distribution mismatch coefficients ν_μ no longer exists for the superlinear convergence analysis.

We also refer to Yuan et al. [98, Appendix H] for more discussions on the distribution mismatch coefficient.

J Sample complexity for neural network parameterization

We prove here Corollary 4.5 through a result by Allen-Zhu et al. [3, Theorem 1 and Example 3.1]. We first give a simplified version of this result and then we show how to use it to prove Corollary 4.5.

Consider learning some unknown distribution $\mathcal{D}$ of data points $z = (x, y) \in \mathbb{R}^d \times \mathcal{Y}$, where x is the input point and y is the label. Without loss of generality, assume $\|x\|_2 = 1$ and $x_d = 1/2$. Consider a loss function $L : \mathbb{R}^k \times \mathcal{Y} \to \mathbb{R}$ such that for every $y \in \mathcal{Y}$, the function $L(\cdot, y)$ is non-negative, convex, 1-Lipschitz continuous and $L(0, y) \in [0, 1]$. This includes both the cross-entropy loss and the L_2-regression loss (for bounded $\mathcal{Y}$).

Let $g : \mathbb{R} \to \mathbb{R}$ be a smooth activation function such that $g(z) = e^z$, $\sin(z)$, $\texttt{sigmoid}(z)$, $\tanh(z)$ or is a low degree polynomial.

Define $F^\star : \mathbb{R}^d \to \mathbb{R}^k$ such that $OPT = \mathbb{E}_{\mathcal{D}}[L(F^\star(x), y)]$ is the smallest population error made by a neural network of the form $F^\star = A^\star g(W^\star x)$, where $A^\star \in \mathbb{R}^{k\times p}$ and $W^\star \in \mathbb{R}^{p\times d}$. Assume for simplicity that the rows of W^* have ℓ_2-norm 1 and each element of A^* is less or equal than 1.

Define a ReLU neural network $F(x, W_0) = A_0\sigma(W_0 x + b_0)$, where $A_0 \in \mathbb{R}^{k\times m}$, $W_0 \in \mathbb{R}^{m\times d}$, the entries of W_0 and b_0 are i.i.d. random Gaussians from $\mathcal{N}(0, 1/m)$ and the entries of A are i.i.d. random Gaussians from $\mathcal{N}(0, \varepsilon_A)$, for $\varepsilon_A \in (0, 1]$. We train the weights W of this neural network through stochastic gradient descent over a dataset with N i.i.d. samples from $\mathcal{D}$, i.e., we update $W_{t+1} = W_t - \eta g_t$, where $\mathbb{E}[g_t] = \nabla\mathbb{E}_{\mathcal{D}}[L(F(x, W_0 + W_t), y)]$.

Theorem J.1 (Theorem 1 of Allen-Zhu et al. [3]). *Let $\varepsilon \in (0, O(1/pk))$, choose $\varepsilon_A = \varepsilon/\widetilde{\Theta}(1)$ for the initialization and learning rate $\eta = \widetilde{\Theta}\left(\frac{1}{\varepsilon km}\right)$. SGD finds a set of parameters such that*

$$\frac{1}{J}\sum_{n=0}^{J-1}\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[L\left(F(x; W^{(0)} + W_t), y\right)\right] \le OPT + \varepsilon$$

with probability $1 - e^{-c\log^2 m}$ over the random initialization, for a sufficiently large constant c, with

$$\text{size } m = \frac{\texttt{poly}(k, p)}{\texttt{poly}(\varepsilon)} \text{ and sample complexity } \min\{N, J\} = \frac{\texttt{poly}(k, p, \log m)}{\varepsilon^2}.$$

Theorem J.1 shows that it is possible to achieve the population error OPT by training a two-layer ReLU network with SGD, and quantifies the number of samples needed to do so.

We make the following assumption to address the population error in our setting.

Assumption J.2. Let $g : \mathbb{R} \to \mathbb{R}$ be a smooth activation function such that $g(z) = e^z$, $\sin(z)$, $\texttt{sigmoid}(z)$, $\tanh(z)$ or is a low degree polynomial. For all time-steps t, we assume that there exists a target network $F^{\star,t} : \mathbb{R}^d \to \mathbb{R}^k$, with

$$F^{\star,t} = (f_1^{\star,t}, \ldots, f_k^{\star,t}) \quad \text{and} \quad f_r^{\star,t}(x) = \sum_{i=1}^{p} a_{r,i}^{\star,t} g(\langle w_{1,i}^{\star,t}, x\rangle)\langle w_{2,i}^{\star,t}, x\rangle$$

where $w_{1,i}^{\star,t} \in \mathbb{R}^d$, $w_{2,i}^{\star,t} \in \mathbb{R}^d$, and $a_{r,i}^{\star,t} \in \mathbb{R}$, such that

$$\mathbb{E}\left[\left\|F^{\star,t} - Q^t - \eta_t^{-1}\nabla h(\pi^t)\right\|_{L_2(v^t)}^2\right] \le OPT.$$

We assume for simplicity $\|w_{1,i}^{\star,t}\|_2 = \|w_{2,i}^{\star,t}\|_2 = 1$ and $|a_{r,i}^{\star,t}| \le 1$.

Assumptions similar to Assumption J.2 have already been made in the literature, such as the bias assumption in the compatible function approximation framework studied by [1]. The term OPT represents the minimum error incurred by a target network parameterized as $F^{\star,t}$ when solving the regression problem in Line 1 of Algorithm 1.

We are now ready to prove Corollary 4.5, which uses Algorithm 4 to obtain an unbiased estimate of the current Q-function. We assume to be in the same setting as Theorem J.1

Proof of Corollary 4.5. We aim to find a policy π^T such that

$$V^\star(\mu) - \mathbb{E}\left[V^T(\mu)\right] \le \varepsilon. \tag{66}$$

Suppose the total number of iterations, that is policy optimization steps, in AMPO is T. We need the bound in Assumption (A1) to hold for all T with probability $1 - e^{-c\log^2 m}$, which means that at each iteration the bound should hold with probability $1 - T^{-1}e^{-c\log^2 m}$. Through Algorithm 4, the expected number of samples needed to obtain an unbiased estimate of the current Q-function is $(1-\gamma)^{-1}$. Therefore, using Theorem J.1, at each iteration of AMPO we need at most

$$\frac{\texttt{poly}(k, p, \log m, \log T)}{\varepsilon_{\text{approx}}^2(1-\gamma)}$$

samples for SGD to find parameters that satisfy Assumption (A1) with probability $1 - T^{-1}e^{-c\log^2 m}$. To obtain (66), we need

$$\frac{1}{1-\gamma}\left(1 - \frac{1}{\nu_\mu}\right)^T\left(1 + \frac{\mathcal{D}_0^\star}{\eta_0(\nu_\mu - 1)}\right) \le \frac{\varepsilon}{2} \quad \text{and} \quad \frac{2(1+\nu_\mu)\sqrt{C_v\varepsilon_{\text{approx}}}}{1-\gamma} \le \frac{\varepsilon}{2}. \tag{67}$$

Algorithm 4: Sampler for an unbiased estimate $\widehat{Q^t}(s,a)$ of $Q^t(s,a)$

Input: Initial state-action couple (s_0, a_0), policy π^t, discount factor $\gamma \in [0,1)$

Initialize $\widehat{Q^t}(s_0,a_0) = r(s_0,a_0)$, the time step $n=0$.
while *True* **do**
  With probability γ**:**
    Sample $s_{n+1} \sim P(\cdot \mid s_n, a_n)$
    Sample $a_{n+1} \sim \pi^t(\cdot|s_{n+1})$
    $\widehat{Q^t}(s_0,a_0) \leftarrow \widehat{Q^t}(s_0,a_0) + r(s_{n+1},a_{n+1})$
    $n \leftarrow n+1$
  Otherwise with probability $(1-\gamma)$**:**
    break ▷ Accept $\widehat{Q}_{s_h,a_h}(\theta)$

Output: $\widehat{Q^t}(s_0,a_0)$

Solving for T and $\varepsilon_{\text{approx}}$ and multiplying them together, we obtain the sample complexity of AMPO, that is

$$\widetilde{\mathcal{O}}\left(\frac{\texttt{poly}(k,p,\log m)C_v^2\nu_\mu^5}{\varepsilon^4(1-\gamma)^6}\right).$$

Due to the statement of Theorem J.1, we cannot guarantee the approximation error incurred by the learner network to be smaller than OPT. Consequently, we have that

$$\varepsilon \geq \frac{4(1+\nu_\mu)\sqrt{C_v OPT}}{1-\gamma}.$$

A similar bound can be applied to any proof that contains the bias assumption introduced by [1]. □

We can obtain an improvement over Corollary 4.5 by using the relaxed assumptions in Appendix G, in particular using the condition in (65).

Corollary J.3. *In the setting of Theorem 4.3, replace Assumption (A1) with the condition*

$$\mathbb{E}\left[\mathbb{E}\left\|f^{t+1} - Q^t - \eta_t^{-1}\nabla h(\pi^t)\right\|^2_{L_2(v^t)}\right] \leq \frac{\nu_\mu^2}{T^2}\left(1-\frac{1}{\nu_\mu}\right)^{-2(T-t)}\varepsilon_{\text{approx}}, \tag{68}$$

for all $t<T$. Let the parameterization class $\mathcal{F}^\Theta$ consist of sufficiently wide shallow ReLU neural networks. Using an exponentially increasing step-size and solving the minimization problem in Line 1 with SGD as in (19), the number of samples required by AMPO to find an ε-optimal policy with high probability is $\widetilde{\Theta}(C_v^2\nu_\mu^4/\varepsilon^4(1-\gamma)^6)$.

Proof. The proof follow that of Corollary 4.5. Using Theorem J.1, at each iteration t of AMPO, we need at most

$$\frac{T^2}{\nu_\mu^2}\left(1-\frac{1}{\nu_\mu}\right)^{2(T-t)}\frac{\texttt{poly}(k,p,\log m,\log T)}{\varepsilon^2_{\text{approx}}(1-\gamma)}$$

samples for SGD to find parameters that satisfy condition (68) with probability $1 - T^{-1}e^{-c\log^2 m}$. Summing over T total iterations of AMPO we obtain that the total number of samples needed is

$$\begin{aligned}
&\sum_{t\leq T}\frac{T^2}{\nu_\mu^2}\left(1-\frac{1}{\nu_\mu}\right)^{2(T-t)}\frac{\texttt{poly}(k,p,\log m,\log T)}{\varepsilon_{\text{approx}}^2(1-\gamma)}\\
&\quad=\frac{T^2}{\nu_\mu^2}\left(1-\frac{1}{\nu_\mu}\right)^{2T}\frac{\texttt{poly}(k,p,\log m,\log T)}{\varepsilon_{\text{approx}}^2(1-\gamma)}\sum_{t\leq T}\left(1-\frac{1}{\nu_\mu}\right)^{-2t}\\
&\quad=\frac{T^2}{\nu_\mu^2}\left(1-\frac{1}{\nu_\mu}\right)^{2T}\frac{\texttt{poly}(k,p,\log m,\log T)}{\varepsilon_{\text{approx}}^2(1-\gamma)}\frac{\left(\left(1-\frac{1}{\nu_\mu}\right)^{-2(T+1)}-1\right)}{\left(\left(\frac{1}{1-\nu_\mu}\right)^{-2}-1\right)}\\
&\quad\leq\mathcal{O}\left(\frac{T^2}{\nu_\mu^2}\frac{\texttt{poly}(k,p,\log m,\log T)}{\varepsilon_{\text{approx}}^2(1-\gamma)}\right)
\end{aligned}$$

Replacing T and $\varepsilon_{\text{approx}}$ with the solutions of (67) gives the result. □

At this stage, it is important to note that choosing a method different from the one proposed by Allen-Zhu et al. [4] to solve Line 1 in Algorithm 1 of our paper with neural networks can lead to alternative, and possibly better, sample complexity results for AMPO. For example, we can obtain a sample complexity result for AMPO that does not involve a target network using results from [39] and [16], although this requires introducing more notation and background results compared to Corollary 4.5 (since in [16] they employ a temporal-difference-based algorithm, that is Algorithm 3 in their work, to obtain a neural network estimate $\widehat{Q}^t$ of Q^t, while in [39] they provide a method based on Fourier transforms to approximate a target function through shallow ReLU networks). We outline below the steps in order to do so (and additional details including the precise statements of the results we use and how we use them are provided thereafter for the sake of completeness).

Step 1) We first split the approximation error in Assumption (A1) into a critic error $\mathbb{E}[\sqrt{\|\widehat{Q}^t - Q^t\|^2_{L_2(v^t)}}] \leq \varepsilon_{\text{critic}}$ and an actor error $\mathbb{E}[\sqrt{\|f^{t+1} - \widehat{Q}^t - \eta_t^{-1}\nabla h(\pi^t)\|^2_{L_2(v^t)}}] \leq \varepsilon_{\text{actor}}$. In this case, the linear convergence rate in our Theorem 4.3 becomes

$$V^\star(\mu) - \mathbb{E}\left[V^T(\mu)\right] \leq \frac{1}{1-\gamma}\left(1-\frac{1}{\nu_\mu}\right)^T\left(1+\frac{\mathcal{D}_0^\star}{\eta_0(\nu_\mu-1)}\right) + \frac{2(1+\nu_\mu)\sqrt{C_v}(\varepsilon_{\text{critic}}+\varepsilon_{\text{actor}})}{1-\gamma}.$$

[We can obtain this alternative statement by modifying the passages in Appendix D.2. In particular, writing $f^{t+1} - Q^t - \eta_t^{-1}\nabla h(\pi^t) = (f^{t+1} - \widehat{Q}^t - \eta_t^{-1}\nabla h(\pi^t)) + (Q^t - \widehat{Q}^t)$ and bounding the two terms with the same procedure in Appendix D.2 leads to this alternative expression for the error.]

We will next deal with the critic error and actor error separately.

Step 2) Critic error. Under a realizability assumption that we provide below along with the statement of the theorem (Assumption 2 in [16]), Theorem 1 from [16] gives that the sample complexity required to obtain $\mathbb{E}[\sqrt{\|\widehat{Q}^t - Q^t\|^2_{L_2(d_\mu^t\cdot\pi^t)}}] \leq \varepsilon$ is $\widetilde{O}(\varepsilon^{-4}(1-\gamma)^{-2})$, while the required network width is $\widetilde{O}(\varepsilon^{-2})$.

Step 3) Actor error. Using Theorem E.1 from [39], we obtain that $\mathbb{E}[\sqrt{\|f^{t+1} - \widehat{Q}^t - \eta_t^{-1}\nabla h(\pi^t)\|^2_{L_2(v^t)}}]$ can be made arbitrarily small by tuning the width of f^{t+1}, without using further samples.

Step 4) Replacing Equation (67) with the sample complexity of the critic, we obtain the following corollary on the sample complexity of AMPO, which does not depend on the error made by a target network.

Corollary J.4. *In the setting of Theorem 4.3, let the parameterization class $\mathcal{F}^\Theta$ consist of sufficiently wide shallow ReLU neural networks. Using an exponentially increasing step-size and using the techniques above to update f^θ, the number of samples required by AMPO to find an ε-optimal policy with high probability is $\widetilde{\mathcal{O}}(C_v^2\nu_\mu^5/\varepsilon^4(1-\gamma)^7)$.*

To the best of our knowledge, this result improves upon the previous best result on the sample complexity of a PG method with neural network parameterization [16], i.e., $\widetilde{\mathcal{O}}(C_v^2/\varepsilon^6(1-\gamma)^9)$.

We now provide the statements of the aforementioned results we used.

Recalling Theorem 1 in [16] and its assumptions. Consider the following space of mappings:

$$\mathcal{H}_{\bar{\nu}} = \{v : \mathbb{R}^d \to \mathbb{R}^d : \sup_{w \in \mathbb{R}^d} \|v(w)\|_2 \leq \bar{\nu}\},$$

and the function class:

$$\mathcal{F}_{\bar{\nu}} = \Big\{g(\cdot) = \mathbb{E}_{w_0 \sim \mathcal{N}(0, I_d)}[\langle v(w_0), \cdot\rangle \mathbb{I}\{\langle w_0, \cdot\rangle > 0\}] : v \in \mathcal{H}_{\bar{\nu}}\Big\}.$$

Consider the following realizability assumption for the Q-function.

Assumption J.5 (Assumption 2 in [16]). For any $t \geq 0$, we assume that $Q^t \in \mathcal{F}_{\bar{\nu}}$ for some $\bar{\nu} > 0$.

Theorem J.6 (Theorem 1 in [16]). *Under Assumption 2 in [16], for any error probability $\delta \in (0,1)$, let*

$$\ell(m', \delta) = 4\sqrt{\log(2m'+1)} + 4\sqrt{\log(T/\delta)},$$

and $R > \bar{\nu}$. Then, for any target error $\varepsilon > 0$, number of iterations $T' \in \mathbb{N}$, network width

$$m' > \frac{16\Big(\bar{\nu} + \big(R + \ell(m', \delta)\big)\big(\bar{\nu} + R\big)\Big)^2}{(1-\gamma)^2\varepsilon^2},$$

and step-size

$$\alpha_C = \frac{\varepsilon^2(1-\gamma)}{(1+2R)^2},$$

Algorithm 3 in [16] yields the following bound:

$$\mathbb{E}\Big[\sqrt{\|\widehat{Q}^t - Q^t\|^2_{L_2(d_\mu^t \cdot \pi^t)}}\mathbb{I}_{A_2}\Big] \leq \frac{(1+2R)\bar{\nu}}{\varepsilon(1-\gamma)\sqrt{T'}} + 3\varepsilon,$$

where A_2 holds with probability at least $1-\delta$ over the random initializations of the critic network $\widehat{Q}^t$.

As indicated in [16], a consequence of this result is that in order to achieve a target error less than $\varepsilon > 0$, a network width of $m' = \widetilde{O}\Big(\frac{\bar{\nu}^4}{\varepsilon^2}\Big)$ and iteration complexity $O\Big(\frac{(1+2\bar{\nu})^2\bar{\nu}^2}{(1-\gamma)^2\varepsilon^4}\Big)$ suffice.

The statement of Theorem 1 in [16] can be readily applied to obtain the sample complexity of the critic.

Recalling Theorem E.1 in [39] and its assumptions Let $g : \mathbb{R}^n \to \mathbb{R}$ be given and define the modulus of continuity ω_g as

$$\omega_g(\delta) := \sup_{x, x' \in \mathbb{R}^n} \{g(x) - g(x') : \max(\|x\|_2, \|x'\|_2) \leq 1 + \delta, \|x - x'\|_2 \leq \delta\}.$$

If g is continuous, then ω_g is not only finite for all inputs, but moreover $\lim_{\delta \to 0} \omega_g(\delta) \to 0$.

Denote $\|p\|_{L_1} = \int |p(w)|dw$. Define a sample from a signed density $p : \mathbb{R}^{n+1} \to \mathbb{R}$ with $\|p\|_{L_1} < \infty$ as (w, b, s), where $(w, b) \in \mathbb{R}$ is sampled from the probability density $|p|/\|p\|_{L_1}$ and $s = sign(p(w, b))$

Theorem J.7 (Theorem E.1 in [39]). *Let $g : \mathbb{R}^n \to \mathbb{R}$, $\delta > 0$ and $\omega_g(\delta)$ be as above and define for $x \in \mathbb{R}^n$*

$$M := \sup_{\|x\| \leq 1+\delta} |g(x)|, \qquad g_{|\delta}(x) = f(x)\mathbb{I}[\|x\| \leq 1 + \delta], \qquad \alpha := \frac{\delta}{\sqrt{\delta} + \sqrt{2\log(2M/\omega_g(\delta))}}.$$

Let G_α be a gaussian distribution on $\mathbb{R}^n$ with mean 0 and variance $\alpha^2\mathcal{I}$. Define the Gaussian convolution $l = g_{|\delta} * G_\alpha$ with Fourier transform $\widehat{l}$ satisfying radial decomposition $\widehat{l}(w) = |\widehat{l}(w)| \exp(2\pi i \theta_h(w))$. Let P be a probability distribution supported on $\|x\| \leq 1$. Additionally define

$$c := g(0)g(0) \int |\widehat{l}(w)| \big[\cos(2\pi(\theta_l(w) - \|w\|_2)) - 2\pi\|w\|_2 \sin(2\pi(\theta_l(w) - \|w\|_2)) \big] dw$$

$$a = \int w|\widehat{l}(w)|dw$$

$$r = \sqrt{n} + 2\sqrt{\log \frac{24\pi^2(\sqrt{d}+7)^2\|g_{|\delta}\|_{L_1}}{\omega_g(\delta)}}$$

$$p := 4\pi^2|\widehat{l}(w)| \cos(2\pi(\|w\|_2 - b))\mathbb{I}[|b| \leq \|w\| \leq r],$$

and for convenience create fake (weight, bias, sign) triples

$$(w, b, s)_{m+1} := (0, c, m\, sign(c)), \quad (w, b, s)_{m+2} := (a, 0, m), \quad (w, b, s)_{m+3} := (-a, 0, -m).$$

Then

$$|c| \leq M + 2\sqrt{n}\|g_{|\delta}\|_{L_1}(2\pi\alpha^2)^{-d/2},$$

$$\|p\|_{L_1} \leq 2\|g_{|\delta}\|_{L_1}\sqrt{\frac{(2\pi)^3 n}{(2\pi\alpha^2)^{n+1}}},$$

and with probability at least $1 - 3\lambda$ *over a draw of* $((s_j, w_j, b_j))_{j=1}^m$ *from* p

$$\sqrt{\left\| g - \frac{1}{m}\sum_{j=1}^{m+3} s_j \sigma(\langle w_j, x\rangle + b_j) \right\|_{L(P)}} \leq 3\omega_g(\delta) + \frac{r\|p\|_{L_1}}{\sqrt{m}} \left[1 + \sqrt{2\log(1/\lambda)}\right].$$

We can then characterize the error of the actor by choosing $x = (s, a)$, $g = \widehat{Q}^t + \eta_t^{-1}\nabla h(\pi^t)$, and $f^{t+1} = \frac{1}{m}\sum_{j=1}^{m+3} s_j \sigma(\langle w_j, x\rangle + b_j)$. We can then make the actor error arbitrarily small by tuning the network width m and δ (note that, since both $\widehat{Q}^t$ and f^t are continuous neural networks, g is a continuous function).

4

Learning Mirror Maps in Policy Mirror Descent

LEARNING MIRROR MAPS IN POLICY MIRROR DESCENT

Carlo Alfano[*]
Department of Statistics
University of Oxford

Sebastian Towers[†]
FLAIR
University of Oxford

Silvia Sapora[†]
FLAIR, Department of Statistics
University of Oxford

Chris Lu
FLAIR
University of Oxford

Patrick Rebeschini
Department of Statistics
University of Oxford

ABSTRACT

Policy Mirror Descent (PMD) is a popular framework in reinforcement learning, serving as a unifying perspective that encompasses numerous algorithms. These algorithms are derived through the selection of a mirror map and enjoy finite-time convergence guarantees. Despite its popularity, the exploration of PMD's full potential is limited, with the majority of research focusing on a particular mirror map—namely, the negative entropy—which gives rise to the renowned Natural Policy Gradient (NPG) method. It remains uncertain from existing theoretical studies whether the choice of mirror map significantly influences PMD's efficacy. In our work, we conduct empirical investigations to show that the conventional mirror map choice (NPG) often yields less-than-optimal outcomes across several standard benchmark environments. Using evolutionary strategies, we identify more efficient mirror maps that enhance the performance of PMD. We first focus on a tabular environment, i.e. Grid-World, where we relate existing theoretical bounds with the performance of PMD for a few standard mirror maps and the learned one. We then show that it is possible to learn a mirror map that outperforms the negative entropy in more complex environments, such as the MinAtar suite. Additionally, we demonstrate that the learned mirror maps generalize effectively to different tasks by testing each map across various other environments.

1 INTRODUCTION

Policy gradient (PG) methods (Williams & Peng, 1991; Sutton et al., 1999; Konda & Tsitsiklis, 2000; Baxter & Bartlett, 2001) are some of the most widely-used mechanisms for policy optimization in reinforcement learning (RL). These algorithms are gradient-based methods that optimize over a class of parameterized policies and have become a popular choice for RL problems, both in theory (Kakade, 2002; Peters & Schaal, 2008; Bhatnagar et al., 2009; Schulman et al., 2015; Mnih et al., 2016; Schulman et al., 2017; Lan, 2022a) and in practice (Shalev-Shwartz et al., 2016; Berner et al., 2019; Ouyang et al., 2022).

Among PG methods, some of the most successful algorithms are those that employ some form of regularization in their updates, ensuring that the newly updated policy retains some degree of similarity to its predecessor. This principle has been implemented in different ways. For instance, trust region policy optimization (TRPO) Schulman et al. (2015) imposes a Kullback-Leibler divergence Kullback & Leibler (1951) hard constraint for its updates, while proximal policy optimization (PPO) (Schulman et al., 2017) uses a clipped objective to penalize large updates. A framework that has recently attracted attention and belongs to this heuristic is that of policy mirror descent (PMD) (Tomar et al., 2022; Lan, 2022a; Xiao, 2022; Kuba et al., 2022; Vaswani et al., 2022; Alfano et al., 2023), which applies mirror descent Nemirovski & Yudin (1983) to RL to regularize the policy updates.

PMD consists of a wide class of algorithms, each derived by selecting a *mirror map* that introduces distinct regularization characteristics. In recent years, PMD has been investigated through numerical

[*]Correspondence to carlo.alfano@stats.ox.ac.uk.
[†]Equal contribution.

experiments (Tomar et al., 2022), but it has mainly been analysed from a theoretical perspective. To the best of our knowledge, research has been mostly focused either on the particular case of the negative entropy mirror map, which generates the natural policy gradient (NPG) algorithm (Kakade, 2002; Agarwal et al., 2021), or on finding theoretical guarantees for a generic mirror map.

NPG has been proven to converge to the optimal policy, up to a difference in expected return (or *error floor*), in several settings, e.g. using tabular, linear, general parameterization (Agarwal et al., 2021) or regularizing rewards (Cen et al., 2021). It has been shown that NPG benefits from implicit regularization (Hu et al., 2022), that it can exploit optimism (Zanette et al., 2021; Liu et al., 2023), and some of its variants have been evaluated in simulations (Vaswani et al., 2022; 2023). When considering other specific mirror maps investigated in the RL literature, the Tsallis entropy has been noted for enhancing performance in offline settings (Tomar et al., 2022) and for offering improved sample efficiency in online settings (Li & Lan, 2023), when compared to the negative entropy.

There is a substantial body of theoretical research focused on the general case of mirror maps. This research demonstrates that PMD achieves convergence to the optimal policy under the same conditions as NPG, as evidenced by various studies (Xiao, 2022; Lan, 2022b; Yuan et al., 2023; Alfano et al., 2023). Except for the setting where we have access to the true value of the policy, convergence guarantees in stochastic settings are subject to an error floor due to the inherent randomness or bias within the algorithm. In the majority of PMD analyses involving generic mirror maps, both the convergence rate and the error floor show mild dependence on the specific choice of mirror map. Typically, the effect of the mirror map appears explicitly as a multiplicative factor in the convergence rate and it appears implicitly in the error floor. These analyses often rely on *upper bounds*, meaning they may not accurately reflect the algorithms' actual performance in applications.

In this work, we contribute to the literature with an empirical investigation of PMD, with the objectives of finding a mirror map that consistently outperforms the negative entropy mirror map and of understanding how the theoretical guarantees of PMD relate to simulations. We first consider a set of tabular environments, i.e. Grid-World (Oh et al., 2020), where we compare the empirical results to the theoretical guarantees given by Xiao (2022), which we can compute as we have full control over the environment. In this setting, we provide a learned mirror map which outperforms the negative entropy in all tested environments. Our experiments suggest that the error floor appearing in prototypical PMD convergence guarantees is not a good performance indicator: our learned mirror map achieves the best value while presenting the worst theoretical error floor, implying that the *upper* bounds typically considered in the PMD literature are loose with respect to the choice of the mirror map. Additionally, our experiments indicate that having small policy updates leads to smoother value improvements over time with less instances of performance degradation, as suggested by the monotonic policy improvement property given by Xiao (2022). We then consider two non-tabular settings, i.e. the Basic Control Suite and the MinAtar Suite, which are more realistic but also more complex, and therefore prevent us from computing the exact theoretical guarantees. We learn a mirror map for each of these environments and, also in this case, show that the learned mirror maps lead to a higher performance of PMD than the negative entropy. Moreover, we show that the learned mirror maps generalize well to other tasks, by testing each of the learned mirror maps on all the other environments. Lastly, we tackle continuous control tasks in MuJoCo (Todorov et al., 2012), where we show that the mirror map learned on one environment surpasses the negative entropy across several environments.

To establish our findings, we employ the standard formulation of PMD (Xiao, 2022) for the tabular setting and the continuous control tasks, and we used a generalized version, Approximate Mirror Policy Optimization (AMPO) (Alfano et al., 2023), for the non-tabular setting with discrete action spaces. To allow optimization over the space of mirror maps, we introduce parameterization schemes for mirror maps, one for PMD and one for AMPO. Specifically, we propose a parameterization for ω-potentials, which have been shown to induce a wide class of mirror maps (Krichene et al., 2015). We use evolution strategies (ES) to search for the mirror map that maximizes the performance of PMD and AMPO over an environment.

AMPO (Alfano et al., 2023) is a recently-proposed PMD framework designed to integrate general parameterization schemes, in our case neural networks, and arbitrary mirror maps. It benefits from theoretical guarantees, as Alfano et al. (2023) show that AMPO has quasi-monotonic updates as well as sub-linear and linear convergence rates, depending on the step-size schedule. These desirable properties make AMPO particularly suitable for our numerical investigation.

ES are a type of population-based stochastic optimization algorithm that leverages random noise to generate a diverse pool of candidate solutions, and have been successfully applied to a variety of tasks (Real et al., 2019; Salimans et al., 2017; Such et al., 2018). The main idea consists in iteratively selecting higher-performing individuals, w.r.t. a fitness function, resulting in a gradual convergence towards the optimal solution. ES algorithms are gradient-free and have been shown to be well-suited for optimisation problems where the objective function is noisy or non-differentiable and the search space is large or complex (Beyer, 2000; Lu et al., 2022; 2023). We use ES to search over the parameterized classes of mirror maps we introduce, by defining the fitness of a particular mirror map as the value of the last policy outputted by PMD and AMPO, for fixed hyper-parameters.

The rest of the paper is organized as follows. In Section 2, we introduce the setting of RL as well as the PMD and AMPO algorithms. We describe the methodology behind our numerical experiments in Section 3, which are then discussed in Section 4. Finally, we discuss related works from the literature on automatic discovery of machine learning algorithms in Appendix A and give our conclusions in Section 5.

2 Preliminaries

2.1 Reinforcement Learning

Define a discounted Markov Decision Process (MDP) as the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$, where $\mathcal{S}$ and $\mathcal{A}$ are respectively the state and action spaces, $P(s' \mid s, a)$ is the transition probability from state s to s' when taking action a, $r(s, a) \in [0, 1]$ is the reward function, γ is a discount factor, and μ is a starting state distribution. A *policy* $\pi \in (\Delta(\mathcal{A}))^{\mathcal{S}}$, where $\Delta(\mathcal{A})$ is the probability simplex over $\mathcal{A}$, represents the behavior of an agent on an MDP, whereby at state $s \in \mathcal{S}$ the agents takes actions according to the probability distribution $\pi(\cdot \mid s)$.

Our objective is for the agent to find a policy that maximizes the expected discounted cumulative reward for the starting state distribution μ. That is, we want to find

$$\pi^\star \in \operatorname*{argmax}_{\pi \in (\Delta(\mathcal{A}))^{\mathcal{S}}} \mathbb{E}_{s\sim\mu}[V^\pi(s)]. \tag{1}$$

Here $V^\pi : \mathcal{S} \to \mathbb{R}$ denotes the *value function* associated with policy π and is defined as

$$V^\pi(s) := \mathbb{E}\Big[\sum\nolimits_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid \pi, s_0 = s\Big],$$

where s_t and a_t are the current state and action at time t and the expectation is taken over the trajectories generated by $a_t \sim \pi(\cdot|s_t)$ and $s_{t+1} \sim P(\cdot|s_t, a_t)$.

Similarly to the value function, we define the Q-function associated with a policy π as

$$Q^\pi(s, a) := \mathbb{E}\Big[\sum\nolimits_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid \pi, s_0 = s, a_0 = a\Big],$$

where the expectation is once again taken over the trajectories generated by the policy π. When the state and action spaces are finite, the Q-function can be expressed as $Q^\pi = (I - \gamma P^\pi)^{-1} r$, where P^π is a square matrix where the position $((s, a), (s', a'))$ is occupied by $\pi(a'|s')P(s'|s, a)$. We also define the discounted state visitation distribution as

$$d_\mu^\pi(s) := (1 - \gamma)\mathbb{E}_{s_0\sim\mu}\Big[\sum\nolimits_{t=0}^{\infty} \gamma^t P(s_t = s \mid \pi, s_0)\Big],$$

where $P(s_t = s \mid \pi, s_0)$ represents the probability of the agent being in state s at time t when following policy π and starting from s_0. The probability distribution over states $d_\mu^\pi(s)$ represents the proportion of time spent on state s when following policy π.

2.2 Policy Mirror Descent

We review the PMD framework, starting from mirror maps (Bubeck, 2015). Let $\mathcal{X} \subseteq \mathbb{R}^{\mathcal{A}}$ be a convex set. A *mirror map* $h : \mathcal{X} \to \mathbb{R}$ is a strictly convex, continuously differentiable and essentially smooth function[1] that satisfies $\nabla h(\mathcal{X}) = \mathbb{R}^{\mathcal{A}}$. In particular, we consider mirror maps belonging to the ω-potential mirror map class, which contains most mirror maps used in the literature.

[1] A function h is *essentially smooth* if $\lim_{x\to\partial\mathcal{X}} \|\nabla h(x)\|_2 = +\infty$, where $\partial\mathcal{X}$ denotes the boundary of $\mathcal{X}$.

Definition 2.1 (ω-potential mirror map (Krichene et al., 2015)). For $u \in (-\infty, +\infty]$, $\omega \leq 0$, an *ω-potential* is defined as an increasing C^1-diffeomorphism $\phi : (-\infty, u) \to (\omega, +\infty)$ such that

$$\lim_{x \to -\infty} \phi(x) = \omega, \ \lim_{x \to u} \phi(x) = +\infty, \ \int_0^1 \phi^{-1}(x)dx \leq \infty.$$

For any ω-potential ϕ, we define the associated mirror map h_ϕ as

$$h_\phi(\pi_s) = \sum\nolimits_{a \in \mathcal{A}} \int_1^{\pi(a|s)} \phi^{-1}(x)dx.$$

When $\phi(x) = e^x$ we recover the negative entropy mirror map, which is the standard choice of mirror map in the RL literature (Agarwal et al., 2021; Tomar et al., 2022; Hu et al., 2022; Vaswani et al., 2022; 2023), while we recover the ℓ_2-norm when $\phi(x) = x$ (see Appendix B.1). Mirror maps belonging to this class are particularly advantageous as they can be defined by a single scalar function ϕ, without having to account for the dimension of the action space $\mathcal{A}$. The *Bregman divergence* (Bregman, 1967; Censor & Zenios, 1997) induced by the mirror map h is defined as

$$\mathcal{D}_h(x, y) := h(x) - h(y) - \langle \nabla h(y), x - y \rangle,$$

where $\mathcal{D}_h(x, y) \geq 0$ for all $x, y \in \mathcal{X}$. As we further discuss in Appendix C, the Bregman divergence measures how far two points are in a geometry induced by the mirror map. Given a starting policy π^0, a learning rate η and a mirror map h, PMD can be formalized as an iterative algorithm: for all iterations $t \geq 0$,

$$\pi^{t+1} \in \text{argmax}_{\pi \in (\Delta(\mathcal{A}))^{\mathcal{S}}} \mathbb{E}_{s \sim d_\mu^t}[\eta_t \langle Q_s^t, \pi_s \rangle - \mathcal{D}_h(\pi_s, \pi_s^t)], \tag{2}$$

where we used the shorthand: $V^t := V^{\pi^t}$, $Q^t := Q^{\pi^t}$, $d_\mu^t := d_\mu^{\pi^t}$ and $y_s := y(s, \cdot) \in \mathbb{R}^{\mathcal{A}}$, for any function $y : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. PMD benefits from several theoretical guarantees and, in particular, Xiao (2022) shows that PMD enjoys quasi-monotonic updates and convergence to the optimal policy. We give here a slight modification of the statements of these results. Specifically, we do not upper-bound a term regarding the distance between subsequent policies in the result on quasi-monotonic updates, which we use to draw connections between theory and practice. Additionally, to obtain the convergence rate for PMD with constant step-size in the setting where we do not have access to the true Q-function, we combine the analyses on the sublinear convergence of PMD and linear convergence of inexact PMD given by Xiao (2022). We provide a proof in Appendix D.

Theorem 2.2 (Xiao (2022)). *Following update (2), we have that, for all $t \geq 0$*

$$V^{t+1}(\mu) - V^t(\mu) \geq -\frac{1}{1-\gamma} \max_{s \in \mathcal{S}} \left\| \widehat{Q}_s^t - Q_s^t \right\|_\infty \max_{s \in \mathcal{S}} \left\| \pi_s^{t+1} - \pi_s^t \right\|_1, \tag{3}$$

where $\|\cdot\|_\infty$ and $\|\cdot\|_1$ represent the ℓ_∞ and the ℓ_1 norms, respectively, and $\widehat{Q}$ is an estimate of the true Q-function. Additionally, at each iteration $T > 0$, we have

$$V^\star(\mu) - \sum_{t<T} \mathbb{E}[V^t(\mu)] \leq \frac{1}{T} \left(\frac{\mathbb{E}_{s \sim d_\mu^\star}[\mathcal{D}_h(\pi_s^\star, \pi_s^0)]}{\eta_t(1-\gamma)} + \frac{1}{(1-\gamma)^2} \right) + 4 \frac{\max_{t<T, s \in \mathcal{S}} \left\| \widehat{Q}_s^t - Q_s^t \right\|_\infty}{(1-\gamma)^2}. \tag{4}$$

The statement of Theorem 2.2 is similar to many results in the literature on PMD and NPG (Agarwal et al., 2021; Xiao, 2022; Lan, 2022b; Hu et al., 2022). That is, the convergence guarantee involves two terms, i.e. a convergence rate, which involves the Bregman divergence between the optimal policy and the starting policy, and an error floor, which involves the estimation error $\max_{s \in \mathcal{S}} \|\widehat{Q}_s^t - Q_s^t\|_\infty$. Given that by setting $\eta_0 = \mathbb{E}_{s \sim d_\mu^\star}[\mathcal{D}_h(\pi_s^\star, \pi_s^0)](1-\gamma)$ we obtain the convergence rate $2(T(1-\gamma)^2)^{-1}$, and that the error floor has no explicit dependence on the mirror map, Equation (4) suggests that the mirror map has a mild influence on the performance of PMD. The only way the mirror map seems to affect the convergence guarantee is, implicitly, by changing the path of the algorithm and therefore influencing the estimation error. Similar observations can be made for several results in the PMD literature that share the same structure of convergence guarantees. On the other hand, Equation (3) suggests that mirror maps that prevent large updates of the policy cause the PMD algorithm to be less prone to performance degradation, as the lower bound is close to 0 when the policy update distance $\max_{s \in \mathcal{S}} \|\pi_s^{t+1} - \pi_s^t\|_1$ is small. One of the contributions of

our work is to challenge and refute the conclusion that the mirror map has little influence on the convergence of PMD, highlighting a gap between theoretical guarantees, which are *based on upper bounds*, and the actual performance observed in PMD-based methodologies. Our empirical studies reveal that the choice of mirror map significantly influences both the speed of convergence and the minimum achievable error floor in PMD. Additionally, we provide evidence that a mirror map that prevents large updates throughout training leads to a better performance, as suggested by (3).

When applying PMD to continuous control tasks, we replace the tabular policy in (2) with a parametrized one. That is, for all iterations $t \geq 0$, we obtain the updated policy as

$$\pi_{\theta^{t+1}} \in \operatorname{argmax}_{\pi_\theta : \theta \in \Theta} \mathbb{E}_{s \sim d_\mu^t}[\eta_t \mathbb{E}_{a \sim \pi_\theta(\cdot|s)}(Q^t(s,a)) - \mathcal{D}_h(\pi_\theta(\cdot|s), \pi^t(\cdot|s))], \tag{5}$$

where $\{\pi_\theta : \theta \in \Theta\}$ is a class of parametrized policies.

2.3 Approximate Mirror Policy Optimization

AMPO is a theoretically sound framework for deep reinforcement learning based on mirror descent, as it inherits the quasi-monotonic updates and convergence guarantees from the tabular case (Alfano et al., 2023). Given a parameterized function class $\mathcal{F}^\Theta = \{f^\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R}, \theta \in \Theta\}$, an initial scoring function f^{θ^0}, a step-size η and an ω-potential mirror map h_ϕ, AMPO can be described, for all iterations t, by the two-step update

$$\pi^t(a \mid s) = \sigma(\phi(\eta f^t(s,a) + \lambda_s^t)) \qquad \forall s \in \mathcal{S}, a \in \mathcal{A}, \tag{6}$$

$$\theta^{t+1} \in \operatorname*{argmin}_{\theta \in \Theta} \mathbb{E}_{s \sim d_\mu^t, a \sim \pi(\cdot|s)} \left[\left(f^\theta(s,a) - Q^t(s,a) - \eta^{-1} \max(\eta f^t(s,a) + \lambda_s^t, \phi^{-1}(0)) \right)^2 \right] \tag{7}$$

where $\lambda_s^t \in \mathbb{R}$ is a normalization factor to ensure $\pi_s^t \in \Delta(\mathcal{A})$ for all $s \in \mathcal{S}$, $f^t := f^{\theta^t}$, and $\sigma(z) = \max(z, 0)$ for $z \in \mathbb{R}$. Theorem 1 by Krichene et al. (2015) ensures that there always exists a normalization constant $\lambda_s^t \in \mathbb{R}$. As shown by Alfano et al. (2023), AMPO recovers the standard formulation of PMD in (2) in the tabular setting. Assuming for simplicity that $\phi(x) > 0$ for all $x \in \mathbb{R}$, the minimization problem in (7) implies that, at each iteration t, f^t is an approximation of the sum of the Q-functions up to that point, that is $f^t \simeq \sum_{i=0}^{t-1} Q^i$. Therefore, the scoring function f^t serves as an estimator of the value of an action.

3 Methodology

Denote by $\mathcal{H}$ the class of ω-potentials mirror maps. Our objective is to search for the mirror map that maximizes the value of the last policy outputted by our mirror descent based algorithms, for a fixed time horizon T. That is, we want to find

$$h^\star \in \operatorname*{argmax}_{h \in \mathcal{H}} \mathbb{E}\left[V^T(\mu)\right], \tag{8}$$

where the expectation is taken over the randomness of the policy optimization algorithm and the policy updates are based on the mirror map h. Depending on the setting, we parameterize the mirror map by parameterizing either ϕ^{-1} or ϕ as monotonically increasing functions. One of the primary objectives of this work it to motivate further research into the choice of mirror maps, as influenced by the choice of the function ϕ, moving beyond the conventional use of $\phi(x) = e^x$. This is achieved by examining how different choices of ϕ influence the trajectory of training and demonstrating that, in many cases, there is a mirror map that outperforms the negative entropy by a large margin.

3.1 Policy Mirror Descent

We parameterize ϕ^{-1} as a one layer neural network with 126 hiden units, where all kernels are non-negative and the activation functions are equally split among the following convex and concave monotonic non-linearities: x^3, $(x)_+^2$, $(x)_+^{1/2}$, $(x)_+^{1/3}$, $\log((x)_+ + 10^{-3})$ and e^x, where $(x)_+ = \max(x, 0)$. To ensure that we are able to recover the negative entropy and the ℓ_2-norm, we add $ax + b\log(x)$ to the final output, where $a, b \geq 0$

To search for the best mirror map within this class, we employ a slight variation of the OpenAI-ES strategy (Salimans et al., 2017), adapted to the multi-task setting (Jackson et al., 2024). Denote by ψ the parameters of the mirror map and by $F(\psi)$ the objective function in (8). Given a distribution of tasks $\mathcal{E}$, we estimate the gradient $\nabla_\psi F(\psi)$ as

$$E_{\epsilon\sim\mathcal{N}(0,I_d)}\left[E_{e\sim\mathcal{E}}\left[\frac{\epsilon}{2\sigma}(F_e(\psi+\sigma\epsilon)-F_e(\psi-\sigma\epsilon))\right]\right], \tag{9}$$

where $\mathcal{N}(0, I_d)$ is the multivariate normal distribution, d is the number of parameters, and $\sigma > 0$ is a hyperparameter regulating the variance of the perturbations. To account for different reward scales across tasks, we perform a rank transformation of the objective functions, whereby, for each sampled task e, we return 1 for the higher performing member between $\psi + \sigma\epsilon$ and $\psi - \sigma\epsilon$, and 0 for the other. We note that, when the distribution of tasks $\mathcal{E}$ covers a single task, we recover the standard OpenAI-ES strategy.

3.2 Approximate Mirror Policy Optimization

As non-tabular environments with discrete action space, we consider the Basic Control Suite (BCS) and the MinAtar suite. Given the higher computational cost of simulations on these environments w.r.t. to the tabular setting, we replace the neural network parameterization for the mirror map with one with fewer parameters, in order to reduce the dimension of the mirror map class we search over. We define the parameterized class $\Phi = \{\phi_\psi : \mathbb{R} \to [0,1], \psi \in \mathbb{R}^n_+\}$, with

$$\phi_\psi(x) = \begin{cases} 0 & \text{if } x \le 0, \\ \frac{x}{\psi_1 n} & \text{if } 0 < x \le \psi_1, \\ \frac{j}{n} + \frac{x - \sum_{i=1}^{j}\psi_i}{n\psi_{j+1}} & \text{if } \sum_{i=1}^{j}\psi_i < x \le \sum_{i=1}^{j+1}\psi_i, \\ 1 & \text{if } x > 1, \end{cases}$$

where $1 \le j \le n-1$. In other words, ϕ is defined as a piece-wise linear function with n steps where, for all $j \le n$, $\phi(\sum_{i=1}^{j}\psi_i) = j/n$ and subsequent points are interpolated with a straight segment. This is illustrated in Figure 1. We note that the ω-potentials within Φ violate some of the constraints in Definition 2.1, as they are non-decreasing instead of increasing and $\lim_{x\to\infty}\phi(x) = 1$ for all $\phi \in \Phi$. In Appendix B.2, we show that, if $\phi \in \Phi$, we can construct an ω-potential ϕ' that satisfies the constraints in Definition 2.1 such that ϕ and ϕ' induce the same policies along the path of AMPO.

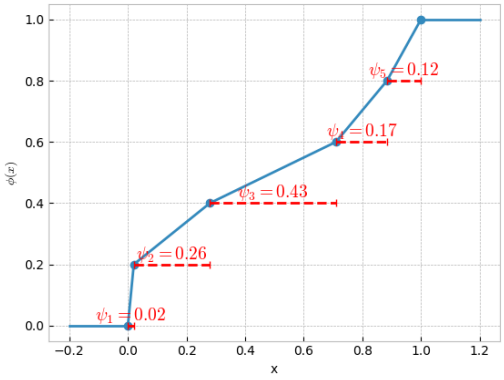

Figure 1: A plot visually demonstrating the parameterization for ϕ.

To effectively learn the hyperparameters, we employ the Separable Covariance Matrix Adaptation Evolution Strategy (sep-CMA) (Ros & Hansen, 2008), a variant of the popular algorithm CMA-ES (Hansen & Ostermeier, 2001). CMA-ES is, essentially, a second-order method adapted for gradient free optimization. At every generation, it samples n new points from a normal distribution, parameterized by a mean vector m_k and a covariance matrix C_k. That is, the samples for generation k, $x_1^k, \dots, x_n^k$, are distributed i.i.d. according to $x_i^k \sim \mathcal{N}(m_k, C_k)$. For each generation k, denote the $m \le n$ best performing samples as $x_1^{*k}, \dots, x_m^{*k}$. Then update the mean vector as $m_{k+1} = \sum_{i=1}^{m} w_i x_i^{*k}$, where $\sum_{i=1}^{m} w_i = 1$, so that the next generation is distributed around the weighted mean of the best performing samples. C_{k+1} is also updated to reflect the covariance structure of $x_1^{*k}, \dots, x_m^{*k}$, in a complex way beyond the scope of this text. Due to the need to update C_k based on covariance information, CMA-ES exhibits a quadratic scaling behavior with respect to the dimensionality of the search space, potentially hindering its efficiency in high-dimensional settings. To improve computational efficiency, we adopt sep-CMA, which introduces a diagonal constraint on the covariance matrix and reduces the computational complexity of the algorithm.

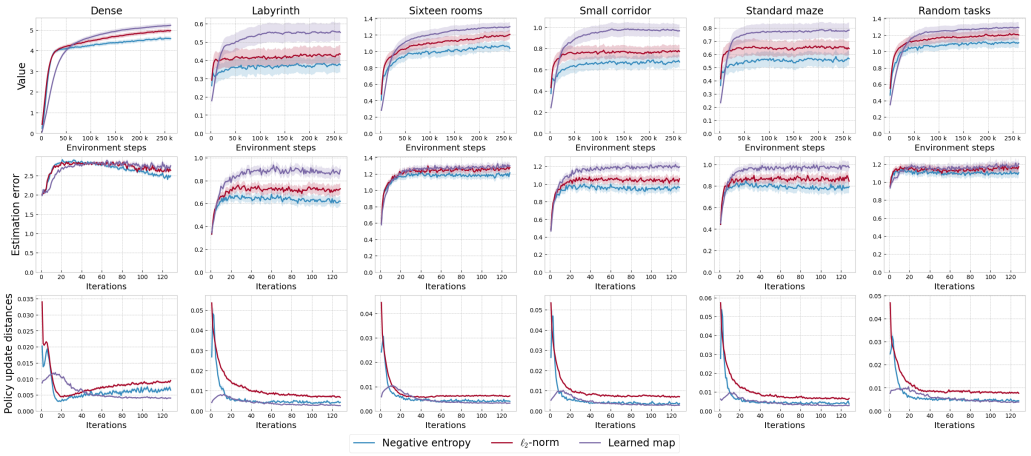

Figure 2: Comparison between the learned map and the negative entropy and ℓ_2-norm mirror maps across a range of held-out configurations of Grid-World. We display the average over 256 runs and report the standard error as a shaded region. The column "Random tasks", reports the averaged metrics for 256 randomly sampled configurations of Grid-World.

4 Experiments

In this section, we discuss the results of our numerical experiments. We start by presenting the tabular setting, where we track errors in order to understand what properties are desirable in a mirror map, and proceed by showing our results in the non-tabular setting.

4.1 Tabular setting: Grid-World

As tabular setting we adopt a discounted and infinite horizon version of Grid-World (Oh et al., 2020), which is a large class of tabular MDPs.

Model architecture and training We define the tabular policy as a one-layer neural network with a softmax head, which takes as input a one-hot encoding of the environment state and outputs a distribution over the action space. We train the policy using the PMD update in (2), where we solve the minimization problem through stochastic gradient descent and estimate the Q-function through generalized advantage estimation (GAE) (Schulman et al., 2016). We perform a simple grid-search over the hyperparameters to maximize the performance for the negative entropy and the ℓ_2-norm mirror maps, in order to have a fair comparison. We report the chosen hyperparameters in Appendix E. The training procedure is implemented in Jax, using evosax (Lange, 2022a) for the evolution. We run on four A40 GPUs, and the optimization process takes roughly 12 hours.

Environment We adopt the version of Grid-World implemented by Jackson et al. (2024), and adapt it to the discounted and infinite horizon setting. We learn a single mirror map by training PMD on a continuous distribution of Grid-World environments, and test PMD with the learned mirror map on five held-out configurations from previous publications (Oh et al., 2020; Chevalier-Boisvert et al., 2024) and on 256 randomly sampled configurations. For all PMD iterations t, we track two quantities that appear in Theorem 2.2, that is the estimation error $\max_{s\in\mathcal{S}}\|\widehat{Q}_s^t - Q_s^t\|_\infty$, and the distance between policy updates $\max_{s\in\mathcal{S}}\|\pi_s^{t+1} - \pi_s^t\|_1$. To obtain the true Q-function, we compute the transition matrix P^{π^t} and use the formula specified in Section 2, that is $Q^t = (I - \gamma P^{\pi^t})^{-1} r$. PMD is run for 128 iterations and $2^{18} \simeq 250k$ total environment steps for all Grid-World configurations.

We show the results of our simulations in Figure 2. As shown by the first row, the learned mirror map, the ℓ_2-norm, and the negative entropy consistently rank first, second and third, respectively, in all tested configurations, in terms of final value. These results advocate the effectiveness of our methodology, as we are able to find a mirror map that outperforms the benchmark mirror maps in all tested environments. We highlight that the first five environments in Figure 2 are not part of the task

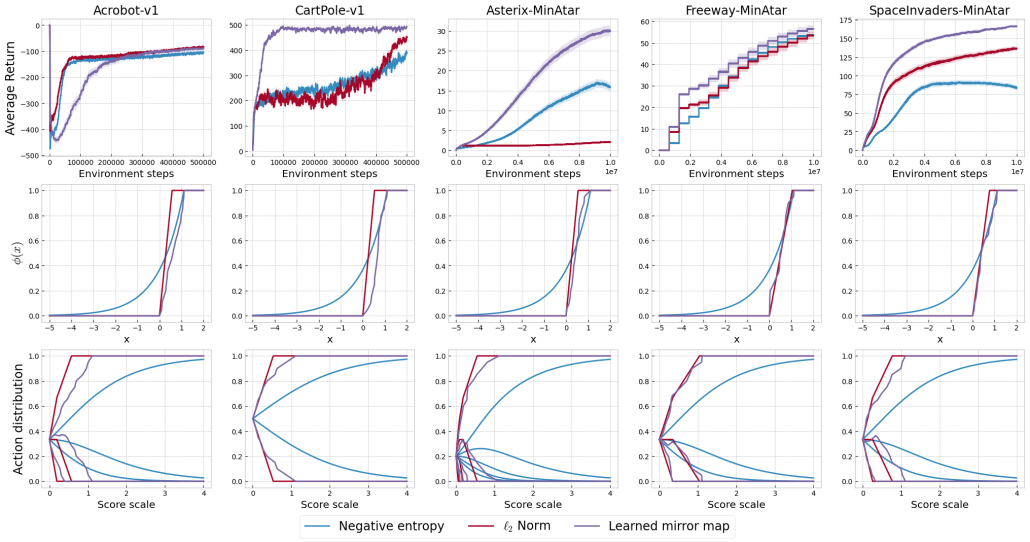

Figure 3: Comparison between the learned mirror map, the ℓ_2-norm and the negative entropy across a range of standard environments. The top plots present the performance of AMPO for all mirror maps, reporting the average over 100 realizations and a shaded region denoting the standard error around the average. The middle plots report the ω-potentials that induce the mirror maps. The bottom plots report the policy distribution according to (6), for each mirror map and score scales. The score scales are obtained by multiplying the vector $[1, \dots, |\mathcal{A}|]$ by a variable $c \in [0, 4]$.

distribution used during the evolution, demonstrating the generalizability of our approach. Moreover, we have that the ℓ_2-norm consistently outperforms the negative entropy, further proving that the negative entropy is not always the best choice of mirror map. Another shared property among all training curves, is that the learned mirror map presents a slower convergence in the initial iterations w.r.t. the ℓ_2-norm and to the negative entropy, as testified by a lower value, but convergence to a higher value in the long run. Lastly, we note that in all environments most of the value improvement happens in the first 50k environment steps.

To gain a better understanding of why the learned mirror maps outperforms the negative entropy and the ℓ_2-norm, and to draw connections with the theoretical results outlined in Section 2.2, we report the estimation error and the distance between policy updates for all mirror maps. The first conclusion that we draw is that a smaller estimation error does not seem to be related to a higher performance. On the contrary, the second row of Figure 2 shows how the three mirror maps consistently have the same ranking in both value and estimation error, which is exactly the opposite of what Theorem 2.2 would suggest. On the other hand, the first and third rows of Figure 2 show how the lower bound on the performance improvement in (3) brings some valid insight on the behaviour of PMD during its first iterations, which are the ones that bring the largest improvement. In all configurations, we have that in the initial iterations of PMD the learned mirror map induces the smallest policy update distances as well as the performance curve with fewer dips in value, while both the negative entropy and the ℓ_2-norm induce larger policy update distances and performance curves with several dips in value. This observation confirms the behaviour described by (3), whereby a small distance between policy updates prevents large performance degradation in policy updates.

4.2 Non-tabular setting: Basic Control and MinAtar suites

Model architecture and training We define the scoring function as a deep neural network, which we train using the AMPO update in (7) and (6), where (7) is solved through Adam and the Q-function is estimated through GAE. We optimize the hyper-parameters of AMPO for the negative entropy mirror map for each suite, using the hyper-parameter tuning framework Optuna (Akiba et al., 2019). This is done to ensure we are looking at a fair benchmark of performance when using the negative entropy mirror map. We report the chosen hyper-parameters in Appendix E. We then initialize the

Table 1: The table contains, for each entry, the value of the final policy outputted by AMPO trained on the environment corresponding to the column with the mirror map learned on the environment corresponding to the row. The last row represents the performance of AMPO with the negative entropy for the corresponding column environments. The value is averaged over 100 runs. Green cells correspond to a value higher than that associated to the negative entropy.

	Acrobot	CartPole	Asterix	Freeway	SpaceInvaders
Acrobot	-88.49	476.41	24.03	56.00	144.01
CartPole	-83.76	499.93	27.26	52.26	100.07
Asterix	-103.55	490.86	30.22	58.56	122.00
Freeway	-82.51	457.47	3.20	58.21	143.93
SpaceInvaders	-78.29	489.56	4.36	22.27	170.24
Negative entropy	-105.63	359.14	17.80	53.69	81.77

parameterized mirror map to be an approximation of the negative entropy[2] and run Sep-CMA-ES on each environment separately. The whole training procedure is implemented in JAX, using gymnax environments (Lange, 2022b) and evosax (Lange, 2022a) for the evolution. We run on 8 GTX 1080Ti GPUs, and the optimization process takes roughly 48 hours for a single environment.

Environments We test AMPO on the Basic Controle Suite (BCS) and the MinAtar Suite. For BCS, we run the evolution for 600 generation, each with 500k timesteps. For MinAtar, we run the evolution for 500 generation with 1M timesteps, then run 100 more generations with 10M timesteps.

Our empirical results are illustrated in Figure 3, where we show the performance of AMPO for the learned mirror map, for the negative entropy and for the ℓ_2-norm. For all environments and mirror maps, we use the hyper-parameters returned by Optuna for AMPO with the negative entropy. Our learned mirror map leads to a better overall performance in all environments, apart from Acrobot, where it ties with the ℓ_2-norm. We observe the largest improvement in performance on Asterix and SpaceInvaders, where the average return for the learned mirror map is more than double the one for the negative entropy. Figure 3 also suggests that different mirror maps may result in different error floors, as shown by the performance curves in Acrobot, Asterix and SpaceInvaders, where the negative entropy converges to a lower point than the learned mirror map.

The second and third row of Figure 3 illustrate the properties of the learned mirror map, in comparison to the negative entropy and the ℓ_2-norm. In particular, the second row shows the corresponding ω-potential, while the third row shows the policy distribution induced by the mirror map according to (6), depending on the scores assigned by the scoring function to each action. A shared property among all the learned mirror maps is that they all lead to assigning 0 probability to the worst actions for relatively small score scales, whilst the negative entropy always assigns positive weights to all actions. In more complex environments, where the evaluation of a certain action may be strongly affected by noise or where the optimal state may be combination locked (Misra et al., 2020), this behaviour may lead to a critical lack of exploration. However, it appears that this is not the case in these environments, and we hypothesise that by setting the probability of the worst actions to 0 the learned mirror maps avoid wasting samples and hence can converge to the optimal policy more rapidly.

Our last result consists in testing each learned mirror map across the other environments we consider. In Table 1, we report the value of the final policy outputted by AMPO for all learned maps, plus the negative entropy, and for all environments, averaged over 100 runs. The table shows that the learned mirror maps generalize well to different environments and to different sets of hyper-parameters, which are shared within BCS and MinAtar but not between them. In particular, we have that the mirror maps learned in Acrobot and Asterix outperform the negative entropy in all environments, those learned in CartPole and Freeway outperform the negative entropy in 4 out of 5 environments, and that learned in SpaceInvaders outperforms the negative entropy in 3 out of 5 environments. These results show that our methodology can be useful in practice, as it benefits from good generalization across tasks.

[2]We achieve this by assigning $\psi_i \propto \log(i/(i-1))$, for $i = 2, \dots, n$, and setting $\psi_1 \propto 3\log(10)$.

4.3 CONTINUOUS CONTROL: MUJOCO

Model architecture and training We define the policy as a deep neural network and optimize it using the PMD update in (5), which is computed using Adam and the estimate of the Q-function obtained through GAE. As previously discussed, we ensure a fair comparison by tuning the hyper-paramers to maximize the performance of the negative entropy. We report the chosen hyper-parameters in Appendix E. The mirror map is initialized to be close to the negative entropy and is trained for 256 generations using OpenAI-ES on a single MuJoCo environment. We run on eight A40 GPUs, and the optimization process takes roughly 24 hours.

Environments We use the `brax` library (Freeman et al., 2021) to simulate three MuJoCo environments, i.e. Hopper, Halfcheetah, and Ant. The mirror map is learned on Hopper and is then tested on all environments, using 10^7 environment steps and 488 PMD update steps.

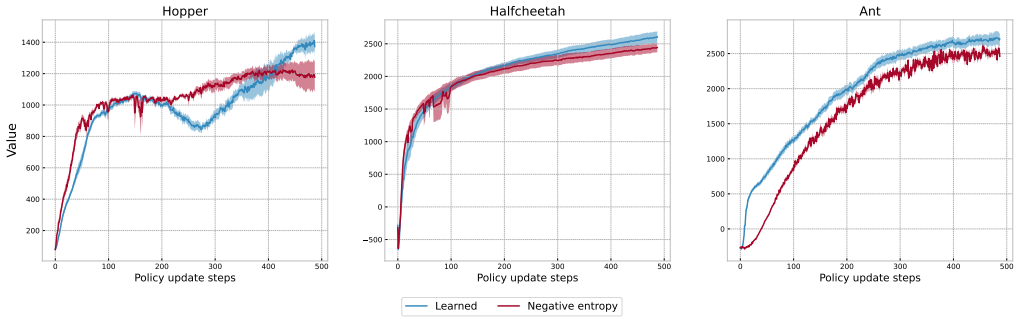

Figure 4: Comparison between the mirror map learned on Hopper and the negative entropy on three MuJoCo environments. The plots present the performance of PMD for both mirror maps, reporting the average over 8 realizations and a shaded region denoting the standard error around the average.

Figure 4 shows that the mirror map learned on Hopper significantly improves in terms of final performance upon the negative entropy in both train and test environments. This result confirms the ability of our methodology to learn generalizable mirror maps, even in complex continuous control tasks.

5 CONCLUSION

Our study presents an empirical examination of PMD, where we successfully test the possibility of learning a mirror map that outperforms the negative entropy in both the tabular and non-tabular settings. In particular, we have shown that the learned mirror maps perform well on a set of configurations in Grid-World and that they can generalize to different tasks in BCS, in MinAtar, and in MuJoCo. Additionally, we have compared the theoretical findings established in the literature and the actual performance of PMD methods in the tabular setting, highlighting how the estimation error is not a good indicator of performance and validating the intuition that small policy updates lead to less instances of performance degradation. Our findings indicate that the choice of mirror map significantly impacts PMD's effectiveness, an aspect not adequately reflected by existing convergence guarantees.

Our research introduces several new directions for inquiry. From a theoretical perspective, obtaining convergence guarantees that reflect the impact of the mirror map is an area for future exploration. On the practical side, investigating how temporal awareness (Jackson et al., 2024) or specific environmental challenges, such as exploration, robustness to noise, or credit assignment (Osband et al., 2019), can inform the choice of a mirror map to improve performance represents another research focus.

ACKNOWLEDGMENTS

Carlo Alfano is funded by the Engineering and Physical Sciences Research Council. Patrick Rebeschini was funded by UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant number EP/Y028333/1]. For the purpose of Open Access, the authors have applied a CC BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

REPRODUCIBILITY STATEMENT

The implementation of our experiments can be found at https://github.com/c-alfano/Learning-mirror-maps.

REFERENCES

Agarwal, A., Kakade, S. M., Lee, J. D., and Mahajan, G. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 2021.

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. Optuna: A next-generation hyperparameter optimization framework, 2019.

Alfano, C., Yuan, R., and Rebeschini, P. A novel framework for policy mirror descent with general parametrization and linear convergence. *Advances in Neural Information Processing Systems*, 2023.

Baxter, J. and Bartlett, P. L. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 2001.

Belousov, B. and Peters, J. f-divergence constrained policy improvement. *arXiv preprint arXiv:1801.00056*, 2017.

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Beyer, H.-G. Evolutionary algorithms in noisy environments: theoretical issues and guidelines for practice. *Computer Methods in Applied Mechanics and Engineering*, 2000.

Bhatnagar, S., Sutton, R. S., Ghavamzadeh, M., and Lee, M. Natural actor-critic algorithms. *Automatica*, 2009.

Bregman, L. M. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 1967.

Bubeck, S. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 2015.

Cen, S., Cheng, C., Chen, Y., Wei, Y., and Chi, Y. Fast global convergence of natural policy gradient methods with entropy regularization. *Operations Research*, 2021.

Censor, Y. and Zenios, S. A. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, USA, 1997.

Chevalier-Boisvert, M., Dai, B., Towers, M., Perez-Vicente, R., Willems, L., Lahlou, S., Pal, S., Castro, P. S., and Terry, J. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *Advances in Neural Information Processing Systems*, 2024.

Freeman, C. D., Frey, E., Raichuk, A., Girgin, S., Mordatch, I., and Bachem, O. Brax - a differentiable physics engine for large scale rigid body simulation, 2021. URL http://github.com/google/brax.

Hansen, N. and Ostermeier, A. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 2001.

Houthooft, R., Chen, Y., Isola, P., Stadie, B., Wolski, F., Jonathan Ho, O., and Abbeel, P. Evolved policy gradients. *Advances in Neural Information Processing Systems*, 2018.

Hu, Y., Ji, Z., and Telgarsky, M. Actor-critic is implicitly biased towards high entropy optimal policies. In *International Conference on Learning Representations*, 2022.

Jackson, M. T., Jiang, M., Parker-Holder, J., Vuorio, R., Lu, C., Farquhar, G., Whiteson, S., and Foerster, J. Discovering general reinforcement learning algorithms with adversarial environment design. *Advances in Neural Information Processing Systems*, 2023.

Jackson, M. T., Lu, C., Kirsch, L., Lange, R. T., Whiteson, S., and Foerster, J. N. Discovering temporally-aware reinforcement learning algorithms. In *International Conference on Learning Representations*, 2024.

Kakade, S. M. A natural policy gradient. *Advances in Neural Information Processing Systems*, 2002.

Kirsch, L., van Steenkiste, S., and Schmidhuber, J. Improving generalization in meta reinforcement learning using learned objectives. In *International Conference on Learning Representations*, 2020.

Konda, V. and Tsitsiklis, J. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, 2000.

Krichene, W., Krichene, S., and Bayen, A. Efficient bregman projections onto the simplex. In *IEEE Conference on Decision and Control*, 2015.

Kuba, J. G., De Witt, C. A. S., and Foerster, J. Mirror learning: A unifying framework of policy optimisation. In *International Conference on Machine Learning*, 2022.

Kullback, S. and Leibler, R. A. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 1951.

Lan, G. Policy mirror descent for reinforcement learning: Linear convergence, new sampling complexity, and generalized problem classes. *Mathematical programming*, 2022a.

Lan, G. Policy optimization over general state and action spaces. *arXiv preprint arXiv:2211.16715*, 2022b.

Lange, R. T. evosax: Jax-based evolution strategies. *arXiv preprint arXiv:2212.04180*, 2022a.

Lange, R. T. gymnax: A JAX-based reinforcement learning environment library, 2022b. URL http://github.com/RobertTLange/gymnax.

Li, Y. and Lan, G. Policy mirror descent inherently explores action space. *arXiv preprint arXiv:2303.04386*, 2023.

Liu, Q., Weisz, G., György, A., Jin, C., and Szepesvari, C. Optimistic natural policy gradient: a simple efficient policy optimization framework for online rl. *Advances in Neural Information Processing Systems*, 2023.

Lu, C., Kuba, J., Letcher, A., Metz, L., Schroeder de Witt, C., and Foerster, J. Discovered policy optimisation. *Advances in Neural Information Processing Systems*, 2022.

Lu, C., Willi, T., Letcher, A., and Foerster, J. N. Adversarial cheap talk. In *International Conference on Machine Learning*, 2023.

Misra, D., Henaff, M., Krishnamurthy, A., and Langford, J. Kinematic state abstraction and provably efficient rich-observation reinforcement learning. In *International conference on machine learning*, 2020.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, 2016.

Nemirovski, A. and Yudin, D. B. *Problem Complexity and Method Efficiency in Optimization*. Wiley Interscience, 1983.

Oh, J., Hessel, M., Czarnecki, W. M., Xu, Z., van Hasselt, H. P., Singh, S., and Silver, D. Discovering reinforcement learning algorithms. *Advances in Neural Information Processing Systems*, 2020.

Orabona, F. A modern introduction to online learning, 2020. URL https://open.bu.edu/handle/2144/40900.

Osband, I., Doron, Y., Hessel, M., Aslanides, J., Sezener, E., Saraiva, A., McKinney, K., Lattimore, T., Szepesvari, C., Singh, S., et al. Behaviour suite for reinforcement learning. In *International Conference on Learning Representations*, 2019.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 2022.

Peters, J. and Schaal, S. Natural actor-critic. *Neurocomputing*, 2008.

Real, E., Aggarwal, A., Huang, Y., and Le, Q. V. Regularized evolution for image classifier architecture search. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

Ros, R. and Hansen, N. A simple modification in cma-es achieving linear time and space complexity. In *International conference on parallel problem solving from nature*. Springer, 2008.

Salimans, T., Ho, J., Chen, X., Sidor, S., and Sutskever, I. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning*, 2015.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

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

Shalev-Shwartz, S., Shammah, S., and Shashua, A. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

Such, F. P., Madhavan, V., Conti, E., Lehman, J., Stanley, K. O., and Clune, J. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning, 2018.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, 1999.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

Tomar, M., Shani, L., Efroni, Y., and Ghavamzadeh, M. Mirror descent policy optimization. In *International Conference on Learning Representations*, 2022.

Vaswani, S., Bachem, O., Totaro, S., Müller, R., Garg, S., Geist, M., Machado, M. C., Castro, P. S., and Le Roux, N. A general class of surrogate functions for stable and efficient reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Vaswani, S., Kazemi, A., Babanezhad, R., and Roux, N. L. Decision-aware actor-critic with function approximation and theoretical guarantees. *Advances in Neural Information Processing Systems*, 2023.

Williams, R. J. and Peng, J. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 1991.

Xiao, L. On the convergence rates of policy gradient methods. *Journal of Machine Learning Research*, 2022.

Yuan, R., Du, S. S., Gower, R. M., Lazaric, A., and Xiao, L. Linear convergence of natural policy gradient methods with log-linear policies. In *International Conference on Learning Representations*, 2023.

Zanette, A., Cheng, C.-A., and Agarwal, A. Cautiously optimistic policy optimization and exploration with linear function approximation. In *Conference on Learning Theory*, 2021.

A Related Works

Discovering reinforcement learning algorithm has been an active area of research in the last years, where researchers have shown that handcrafted algorithms are not always optimal and can be outperformed by automatically discovered algorithms. Oh et al. (2020) obtained an algorithm capable of solving GridWorld and Atari tasks without relying on common notions in the field, such as Q-value functions. Houthooft et al. (2018) used ES to meta-train a policy loss network that outperforms PPO and Kirsch et al. (2020) discovered a new loss function for deterministic policies. Most closely related to our work are Lu et al. (2022) and Jackson et al. (2024), who employ ES to discover RL algorithms within the Mirror Learning class (Kuba et al., 2022). Given a function f, the class of algorithms they consider follows the policy update step

$$\pi_{\theta^{t+1}} \in \text{argmax}_{\pi_\theta : \theta \in \Theta} \mathbb{E}_{s \sim d_\mu^t} \left[\eta_t \mathbb{E}_{a \sim \pi_\theta(\cdot|s)}(Q^t(s,a)) - \mathbb{E}_{a \sim \pi^t(\cdot|s)} \left[f\left(\frac{\pi_\theta(\cdot|s)}{\pi^t(\cdot|s)} \right) \right] \right],$$

This update is similar the the one in (5) but replaces the Bregman divergence with a different penalty term called *drift*, which recovers the f-divergences when f is a convex function. To the best of our knowledge, this class of algorithms has fewer theoretical guarantees than PMD, in particular we are not aware of finite-time convergence guarantees or convergence guarantees involving approximation or estimation errors (Belousov & Peters, 2017; Kuba et al., 2022).

B Further discussion on ω-potentials

B.1 Negative entropy and ℓ_2-norm

If $\phi(x) = e^{x-1}$, then the associated mirror map h_ϕ is the negative entropy. We have that

$$\int_0^1 \phi^{-1}(x)dx = \int_0^1 \log(x)dx = [x\log(x) - x]_0^1 = -1 \leq +\infty.$$

The mirror map h_ϕ becomes the negative entropy, up to a constant, as

$$h_\phi(\pi_s) = \sum_{a \in \mathcal{A}} \int_1^{\pi(a|s)} \log(x)dx = |\mathcal{A}| - 1 + \sum_{a \in \mathcal{A}} \pi(a \mid s) \log(\pi(a \mid s)).$$

If $\phi(x) = x$, then the associated mirror map h_ϕ is the ℓ_2-norm. We have that

$$\int_0^1 \phi^{-1}(x)dx = \int_0^1 x dx = \left[\frac{x^2}{2} \right]_0^1 = \frac{1}{2} \leq +\infty.$$

The mirror map h_ϕ becomes the ℓ_2-norm, up to a constant, as

$$h_\phi(\pi_s) = \sum_{a \in \mathcal{A}} \int_1^{\pi(a|s)} x dx = \frac{1}{2} \sum_{a \in \mathcal{A}} \pi(a \mid s)^2 - 1.$$

B.2 Parametric ϕ

We show here that the parametric class of ω-potentials we introduce in Section 3 results in a well defined algorithm when used for AMPO, even if it breaks some of the constraints in Definition 2.1. In particular, ω-potentials within Φ are not C^1-diffeomorphisms and are only non-decreasing. We can afford not meeting the first constraint as the proof for Theorem 1 by Krichene et al. (2015), which establishes the existence of the normalization constant in Equation (6), only requires Lipschitz-continuity. As to the second constraint, we build an augmented class Φ' that contains increasing ω-potentials and show that it results in the same updates for AMPO as Φ.

Let Δ_n be the n-dimensional probability simplex. We define the parameterized class $\Phi' = \{\phi'_\psi : \mathbb{R} \to [0,1], \psi \in \Delta_n\}$, with

$$\phi'_\psi(x) = \begin{cases} e^x - 1 & \text{if } x \leq 0, \\ \frac{x}{\psi_1 n} & \text{if } 0 < x \leq \psi_1, \\ \frac{j}{n} + \frac{x - \sum_{i=1}^j \psi_i}{n\psi_{j+1}} & \text{if } \sum_{i=1}^j \psi_i < x \leq \sum_{i=1}^{j+1} \psi_i, \\ x & \text{if } x > 1, \end{cases}$$

where $1 \leq j \leq n-1$. The functions within Φ' are increasing and continuous. We start by showing that $\phi'_\psi \in \Phi'$ and $\phi_\psi \in \Phi$ are equivalent for Equation (6), for the same parameters ψ. At each iteration t, we have that

$$\begin{aligned}\pi^{t+1}(a \mid s) &= \sigma(\phi'(\eta f^{t+1}(s,a) + (\lambda')_s^{t+1})) \\ &= \sigma(\phi(\eta f^{t+1}(s,a) + (\lambda')_s^{t+1})) \\ &= \sigma(\phi(\eta f^{t+1}(s,a) + \lambda_s^{t+1})).\end{aligned}$$

This due to the following two facts. For $x \leq 0$, $\sigma(\phi'(x))$ and $\sigma(\phi(x))$ are the same function. Also, $\sigma(\phi'(\eta f^{t+1}(s,a) + (\lambda')_s^{t+1}))$ has to be less or equal to 1, as a result of the projection, meaning that $\eta f^{t+1}(s,a) + (\lambda')_s^{t+1} \leq 1$, where $\sigma(\phi'(\cdot))$ and $\sigma(\phi(\cdot))$ are equivalent.

Since ϕ and ϕ' induce the same policy, share that same normalization constant, and have the same value at 0, they also induce the same expression for Equation (7).

C Further discussion on Bregman divergence

To provide an intuition on the Bregman divergence, we report here some discussion from Orabona (2020). Given a mirror map h, the associated Bregman divergence between two points $x, y \in \mathcal{X}$ is defined as

$$\mathcal{D}_h(x,y) := h(x) - h(y) - \langle \nabla h(y), x - y \rangle.$$

Since h is convex, the Bregman divergence is always non-negative for $x, y \in \mathcal{X}$. We can build more intuition for the Bregman divergence using Taylor's theorem. Assume that h is twice differentiable in an open ball B around y and $x \in B$. Then there exists $0 \leq \alpha \leq 1$ such that

$$B_\psi(x;y) = h(x) - h(y) - \nabla h(y)^\top (x-y) = \frac{1}{2}(x-y)^\top \nabla^2 h(z)(x-y),$$

where $z = \alpha x + (1-\alpha)y$. Therefore, the Bregman divergence can be approximated by squared local norms that depend on the Hessian of the mirror map h. This means that the Bregman divergence will behave differently on different areas of $\mathcal{X}$, depending on the value of Hessian of the mirror map.

D Proof of Theorem 2.2

In this section we outline how we use the proofs given by Xiao (2022) in order to obtain Theorem 2.2.

D.1 Preliminary Lemmas

We start by presenting two preliminary lemmas, which are ubiquitous in the literature on PMD. The first characterizes the difference in value between two policies, while the second characterizes the PMD update.

Lemma D.1 (Performance difference lemma, Lemma 1 in Xiao (2022))**.** *For any policy $\pi, \pi' \in \Delta(\mathcal{A})^{\mathcal{S}}$ and $\mu \in \Delta(\mathcal{S})$,*

$$V^\pi(\mu) - V^{\pi'}(\mu) = \frac{1}{1-\gamma}\mathbb{E}_{s \sim d_\mu^\pi}\left[\langle Q_s^{\pi'}, \pi_s - \pi'_s \rangle\right].$$

Lemma D.2 (Three-point decent lemma, Lemma 6 in Xiao (2022))**.** *Suppose that $\mathcal{C} \subset \mathbb{R}^m$ is a closed convex set, $f : \mathcal{C} \to \mathbb{R}$ is a proper, closed [3] convex function, $\mathcal{D}_h(\cdot,\cdot)$ is the Bregman divergence generated by a mirror map h. Denote* $\operatorname{rint}\operatorname{dom} h$ *as the relative interior of* $\operatorname{dom} h$*. For any $x \in \operatorname{rint}\operatorname{dom} h$, let*

$$x^+ \in \arg\min_{u \in \operatorname{dom} h \cap \mathcal{C}} \{f(u) + \mathcal{D}_h(u,x)\}.$$

Then $x^+ \in \operatorname{rint}\operatorname{dom} h \cap \mathcal{C}$ and for any $u \in \operatorname{dom} h \cap \mathcal{C}$,

$$f(x^+) + \mathcal{D}_h(x^+, x) \leq f(u) + \mathcal{D}_h(u,x) - \mathcal{D}_h(u, x^+).$$

[3] A convex function f is proper if $\operatorname{dom} f$ is nonempty and for all $x \in \operatorname{dom} f$, $f(x) > -\infty$. A convex function is closed, if it is lower semi-continuous.

D.2 QUASI-MONOTONIC UPDATES

We first prove that PMD enjoys quasi-monotonic updates (Equation (3)), that is PMD updates have an upper bound on how much they can deteriorate performance.

Proposition D.3 (Lemma 11 in Xiao (2022))**.** *At each time $t \geq 0$, we have*

$$\langle \eta_t \widehat{Q}_s^t, \pi_s^{t+1} - \pi_s^t \rangle \geq 0.$$

Additionally, we have that

$$V^{t+1}(\mu) - V^t(\mu) \geq -\frac{1}{1-\gamma} \max_{s \in \mathcal{S}} \left\| \widehat{Q}_s^t - Q_s^t \right\|_\infty \max_{s \in \mathcal{S}} \left\| \pi_s^{t+1} - \pi_s^t \right\|_1. \tag{10}$$

Proof. Using Lemma D.2 with $x^+ = \pi_s^{t+1}$, $\mathcal{C} = \Delta(\mathcal{A})$, $f(u) = \langle \widehat{Q}_s^t, u \rangle$, $x = \pi^t$ and $u = \pi^{t+1}$, we obtain

$$\langle \eta_t \widehat{Q}_s^t, \pi_s^t - \pi_s^{t+1} \rangle \leq \mathcal{D}_h(\pi_s^t, \pi_s^t) - \mathcal{D}_h(\pi_s^{t+1}, \pi_s^t) - \mathcal{D}_h(\pi_s^t, \pi_s^{t+1}). \tag{11}$$

By rearranging terms and noticing $\mathcal{D}_h(\pi_s^t, \pi_s^t) = 0$, we have

$$\langle \eta_t \widehat{Q}_s^t, \pi_s^{t+1} - \pi_s^t \rangle \geq \mathcal{D}_h(\pi_s^{t+1}, \pi_s^t) + \mathcal{D}_h(\pi_s^t, \pi_s^{t+1}) \geq 0. \tag{12}$$

Equation (10) can be obtained using the performance difference lemma and (12):

$$\begin{aligned}
V^{t+1}(\mu) - V^t(\mu) &= \frac{1}{1-\gamma} \mathbb{E}_{s \sim d_\mu^{t+1}} \left[\langle Q_s^t, \pi_s^{t+1} - \pi_s^t \rangle \right] \\
&= \frac{1}{1-\gamma} \mathbb{E}_{s \sim d_\mu^{t+1}} \left[\langle \widehat{Q}_s^t, \pi_s^{t+1} - \pi_s^t \rangle \right] \\
&\quad + \frac{1}{1-\gamma} \mathbb{E}_{s \sim d_\mu^{t+1}} \left[\langle \widehat{Q}_s^t - Q_s^t, \pi_s^{t+1} - \pi_s^t \rangle \right] \\
&\geq -\frac{1}{1-\gamma} \mathbb{E}_{s \sim d_\mu^{t+1}} \left[\left\| \widehat{Q}_s^t - Q_s^t \right\|_\infty \left\| \pi_s^{t+1} - \pi_s^t \right\|_1 \right].
\end{aligned}$$

□

D.3 CONVERGENCE GUARANTEE

We can now prove the convergence guarantee reported in Equation (4). The proof of our result is obtained combining the analysis from the sublinear convergence guarantee with the analysis for the inexact linear convergence guarantee given by (Xiao, 2022, Theorems 8 and 13). For two different time $t, t' \geq 0$, denote the expected Bregman divergence between the policy π^t and policy $\pi^{t'}$, where the expectation is taken over the discounted state visitation distribution of the optimal policy $d_\mu^\star$, by

$$\mathcal{D}_{t'}^t := \mathbb{E}_{s \sim d_\mu^\star} \left[\mathcal{D}_h(\pi_s^t, \pi_s^{t'}) \right].$$

Similarly, denote the expected Bregman divergence between the optimal policy $\pi^\star$ and π^t by

$$\mathcal{D}_t^\star := \mathbb{E}_{s \sim d_\mu^\star} \left[\mathcal{D}_h(\pi_s^\star, \pi_s^t) \right].$$

Theorem D.4 (Theorems 8 and 13 in Xiao (2022))**.** *Consider the PMD update in (2), at each iteration $T \geq 0$, we have*

$$V^\star(\mu) - \sum_{t<T} \mathbb{E}[V^t(\mu)] \leq \frac{1}{T} \left(\frac{\mathbb{E}[\mathcal{D}_0^\star]}{\eta_t(1-\gamma)} + \frac{1}{(1-\gamma)^2} \right) + \frac{4}{(1-\gamma)^2} \mathbb{E} \left[\max_{s \in \mathcal{S}} \left\| \widehat{Q}_s^t - Q_s^t \right\|_\infty \right].$$

Proof. Using Lemma D.2 with $x^+ = \pi_s^{t+1}$, $\mathcal{C} = \Delta(\mathcal{A})$, $f(u) = \langle \widehat{Q}_s^t, u \rangle$, $x = \pi^t$ and $u = \pi^{t+1}$, we have that

$$\langle \eta_t \widehat{Q}_s^t, \pi_s^\star - \pi_s^{t+1} \rangle \leq \mathcal{D}_h(\pi^\star, \pi^t) - \mathcal{D}_h(\pi^\star, \pi^{t+1}) - \mathcal{D}_h(\pi^{t+1}, \pi^t),$$

which can be decomposed as

$$\langle \eta_t \widehat{Q}_s^t, \pi_s^t - \pi_s^{t+1} \rangle + \langle \eta_t \widehat{Q}_s^t, \pi_s^\star - \pi_s^t \rangle \leq \mathcal{D}_h(\pi^\star, \pi^t) - \mathcal{D}_h(\pi^\star, \pi^{t+1}) - \mathcal{D}_h(\pi^{t+1}, \pi^t).$$

Taking expectation with respect to the distribution $d_\mu^\star$ over states and with respect to the randomness of PMD and dividing both sides by η_t, we have

$$\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle \widehat{Q}_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] + \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle \widehat{Q}_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right] \leq \frac{1}{\eta_t}\mathbb{E}[\mathcal{D}_t^\star - \mathcal{D}_{t+1}^\star - \mathcal{D}_t^{t+1}]. \quad (13)$$

We lower bound the two terms on the left hand side of (13) separately. For the first term, we have that

$$\begin{aligned}
\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle \widehat{Q}_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] &\overset{(a)}{\geq} \frac{1}{1-\gamma}\mathbb{E}\left[\mathbb{E}_{s\sim d_{d_\mu^\star}^{t+1}}\left[\langle \widehat{Q}_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] \\
&= \frac{1}{1-\gamma}\mathbb{E}\left[\mathbb{E}_{s\sim d_{d_\mu^\star}^{t+1}}\left[\langle Q_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] + \frac{1}{1-\gamma}\mathbb{E}\left[\mathbb{E}_{s\sim d_{d_\mu^\star}^{t+1}}\left[\langle \widehat{Q}_s^t - Q_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] \\
&\overset{(b)}{=} \mathbb{E}\left[V^t(d_\mu^\star) - V^{t+1}(d_\mu^\star)\right] + \frac{1}{1-\gamma}\mathbb{E}\left[\mathbb{E}_{s\sim d_{d_\mu^\star}^{t+1}}\left[\langle \widehat{Q}_s^t - Q_s^t, \pi_s^t - \pi_s^{t+1}\rangle\right]\right] \\
&\geq \mathbb{E}\left[V^t(d_\mu^\star) - V^{t+1}(d_\mu^\star)\right] - \frac{1}{1-\gamma}\mathbb{E}\left[\mathbb{E}_{s\sim d_{d_\mu^\star}^{t+1}}\left[\left\|\widehat{Q}_s^t - Q_s^t\right\|_\infty \left\|\pi_s^{t+1} - \pi_s^t\right\|_1\right]\right] \\
&\geq \mathbb{E}\left[V^t(d_\mu^\star) - V^{t+1}(d_\mu^\star)\right] - \frac{2}{1-\gamma}\mathbb{E}\left[\max_{s\in\mathcal{S}}\left\|\widehat{Q}_s^t - Q_s^t\right\|_\infty\right]
\end{aligned}$$

where (a) follows from Lemmas D.3 and the fact that $d_{d_\mu^\star}^{t+1}(s) \geq (1-\gamma)d_\mu^\star(s)\ \forall s \in \mathcal{S}$, and (b) follows from D.1. For the second term, we have that

$$\begin{aligned}
\mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle \widehat{Q}_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right] &= \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle Q_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right] + \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle \widehat{Q}_s^t - Q_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right] \\
&\overset{(b)}{=} \mathbb{E}[V^\star(\mu) - V^t(\mu)](1-\gamma) + \mathbb{E}\left[\mathbb{E}_{s\sim d_\mu^\star}\left[\langle \widehat{Q}_s^t - Q_s^t, \pi_s^\star - \pi_s^t\rangle\right]\right], \\
&\geq \mathbb{E}[V^\star(\mu) - V^t(\mu)](1-\gamma) - \frac{2}{1-\gamma}\mathbb{E}\left[\max_{s\in\mathcal{S}}\left\|\widehat{Q}_s^t - Q_s^t\right\|_\infty\right],
\end{aligned}$$

where (b) follows from Lemma D.1.

Plugging the two bounds in (13), dividing both sides by $(1-\gamma)$ and rearranging, we obtain

$$\begin{aligned}
\frac{\mathbb{E}[\mathcal{D}_t^{t+1}]}{\eta_t(1-\gamma)} + \mathbb{E}[V^\star(\mu) - V^t(\mu)] \leq \frac{\mathbb{E}[\mathcal{D}_t^\star - \mathcal{D}_{t+1}^\star]}{\eta_t(1-\gamma)} + \frac{\mathbb{E}\left[V^t(d_\mu^\star) - V^{t+1}(d_\mu^\star)\right]}{1-\gamma} \\
+ \frac{4}{(1-\gamma)^2}\mathbb{E}\left[\max_{s\in\mathcal{S}}\left\|\widehat{Q}_s^t - Q_s^t\right\|_\infty\right].
\end{aligned}$$

Summing up from 0 to $T-1$ and dropping some positive terms on the left hand side and some negative terms on the right hand side, we have

$$V^\star(\mu) - \sum_{t<T}\mathbb{E}[V^t(\mu)] \leq \frac{\mathbb{E}[\mathcal{D}_0^\star]}{\eta_t(1-\gamma)} - \frac{\mathbb{E}\left[V^0(d_\mu^\star) - V^T(d_\mu^\star)\right]}{1-\gamma} + \frac{4}{(1-\gamma)^2}\mathbb{E}\left[\max_{s\in\mathcal{S}}\left\|\widehat{Q}_s^t - Q_s^t\right\|_\infty\right].$$

Notice that $\mathbb{E}\left[V^0(d_\mu^\star) - V^T(d_\mu^\star)\right] \leq \frac{1}{1-\gamma}$ as $r(s,a) \in [0,1]$. By dividing T on both side, we yield the statement. □

E Training Details

We give the hyper-parameters we use for training in Tables 2, 3 and 4. Hyper-parameter tuning was performed differently for each method. For Gridworld, we performed a grid search over the hyper-parameters and selected those that maximized the averaged performance of the negative entropy and ℓ_2-norm over 256 randomly sampled Gridworld environments. For Basic Control Suite and MinAtar, we used the optuna library to search over the hyper-parameters to maximize the average performance of the negative entropy over all environments. We used "CmaEsSampler" as option for the sampler. Lastly, we used Weights and Biases to optimize the hyperparameters for MuJoCo in order to maximimize the average performance of the negative entropy over all environments.

Table 2: Hyper-parameter settings of PMD for different sets of environments

Parameter	Grid-World	MuJoCo
Number of environment steps	$2^{18} \simeq 2.5e5$	1e7
Number of environments	64	2048
Unroll length	32	10
Number of minibatches	1	128
Number of update epochs	32	8
Adam learning rate	-	1e-4
Sgd learning rate	40	-
Gamma	0.99	0.99
Max grad norm	-	1.0
η	0.1	0.5

Table 3: Hyper-parameter settings of AMPO for different sets of environments

Parameter	BCS	MinAtar
Number of environment steps	5e5	1e7
Number of environments	4	256
Unroll length	128	128
Number of minibatches	4	8
Number of update epochs	16	8
Adam learning rate	4e-3	7e-4
Gamma	0.99	0.99
Max grad norm	1.4	1
AMPO learning rate	0.9	0.9

Table 4: Hyper-parameter settings of OpenAI-ES, which we used in the tabular setting, and of Sep-CMA-ES, which we used in the non-tabular case.

	OpenAI-ES	Sep-CMA-ES
Population Size	512	128
Number of generations	512 (256 for MuJoCo)	600
Sigma init	0.5	2
Sigma Decay	0.995	-
Learning rate	0.01	-

F COMPUTATIONAL COSTS OF ES

The computational costs associated with ES are inevitable in this field of research that focuses on automatically discovering optimizers. In the literature, the most popular methods to discover optimizers are meta-gradients (Oh et al., 2020; Jackson et al., 2023) and ES (Lu et al., 2022). Discovering algorithms using meta-gradients consists in introducing some meta-parameters that influence the agent training procedure, training a batch of agents, and differentiating through the training procedure w.r.t. the meta-parameters to maximize the final performance of the agents. ES consists in running several training procedures in parallel and estimating the meta-gradient as in (9), therefore avoiding the need for differentiation. This property is particularly desirable in settings, like ours, where the agent is updated many times and differentiating through the whole training procedure is unfeasible due to memory constraints, meaning that it is necessary to limit the differentiation to the final updates. We decided to employ ES due to this property and because it has been found to be more successful in discovering optimizers (Jackson et al., 2024). Additionally, our experiments are implemented in JAX, which through parallelism and just in time compilation renders the computational costs of ES feasible.

5

Meta-Learning Objectives for Preference Optimization

Meta-Learning Objectives for Preference Optimization

Carlo Alfano[*]
Department of Statistics
University of Oxford

Silvia Sapora[*]
Department of Statistics
University of Oxford

Jakob N. Foester
Department of Engineering
University of Oxford

Patrick Rebeschini
Department of Statistics
University of Oxford

Yee Whye Teh
Department of Statistics
University of Oxford

Abstract

Evaluating preference optimization (PO) algorithms on LLM alignment is a challenging task that presents prohibitive costs, noise, and several variables like model size and hyper-parameters. In this work, we show that it is possible to gain insights on the efficacy of PO algorithm on simpler benchmarks. We design a diagnostic suite of MuJoCo tasks and datasets, which we use to systematically evaluate PO algorithms, establishing a more controlled and cheaper benchmark. We then propose a novel family of PO algorithms based on mirror descent, which we call Mirror Preference Optimization (MPO). Through evolutionary strategies, we search this class to discover algorithms specialized to specific properties of preference datasets, such as mixed-quality or noisy data. We demonstrate that our discovered PO algorithms outperform all known algorithms in the targeted MuJoCo settings. Finally, based on the insights gained from our MuJoCo experiments, we design a PO algorithm that significantly outperform existing baselines in an LLM alignment task.

1 Introduction

Learning from human preferences (Christiano et al., 2017) is a paradigm which enables the alignment of machine learning systems to relative human preferences, without requiring access to absolute rewards. While the framework was developed for robotic and games applications with experiments on MuJoCo simulations and Atari (Akrour et al., 2012; Biyik & Sadigh, 2018; Ibarz et al., 2018), this paradigm has been successfully applied to Large Language Models (Team et al., 2023; Achiam et al., 2023). In particular, fine-tuning pre-trained LLMs with human preferences has become a popular strategy to adapt them to specific tasks and to improve their safety and helpfulness.

Within this framework, Reinforcement Learning from Human Feedback (RLHF) is one of the most popular methods. It consists in learning a reward function using a preference dataset and then optimizing the estimated reward using Reinforcement Learning methods such as Proximal Policy Optimization (PPO) (Schulman et al., 2017). However, the training pipeline of RLHF is quite complex, which is why implicit approaches such as Direct Preference Optimisation (DPO) (Schulman et al., 2017) have gained traction thanks to their simplicity. These methods do not learn a reward model but estimate it implicitly using the policy of the agent. Many follow ups to DPO have been proposed (Yuan et al., 2023; Zhao et al., 2023; Azar et al., 2024; Xu et al., 2024a; Hong et al., 2024; Park et al., 2024; Meng et al., 2024), but comparing their performance in LLM alignment is a complex task that incurs high costs, noise, and the inherent difficulty in judging a response better than another.

In this work, we provide a comprehensive analysis of PO algorithms, examining their behavior on automatically generated preference datasets. We return to the roots of RLHF by performing this analysis in MuJoCo environments and datasets, where the underlying ground-truth reward structure is well defined and offers a clear performance metric to compare agents. In particular, we design a task where a pre-trained agent has to adhere to a new stylistic constraint, emulating the typical conditions of LLM fine-tuning. Our findings indicate that many PO algorithms present distinct failure modes when applied to specific mixed-quality or noisy datasets.

[*]Equal contribution, order decided by coin flip.

Moreover, we introduce a framework for finding PO algorithms. Specifically, we define a class of PO algorithms based on mirror descent (Nemirovski & Yudin, 1983), which generalizes DPO and ORPO for particular choices of the mirror map. We then show that this class can be easily parametrized and searched using evolutionary strategies (ES), optimizing for the final performance of the trained policy, as measured by the ground truth reward.

For each setting we consider, we discover an algorithm that significantly outperforms all baselines. Analyzing the discovered algorithms, we find that the main difference between them and the baselines is that they keep optimizing the policy of the agent well after the probability of generating the chosen trajectory has surpassed the probability of generating the rejected one. We use this insight to design a new PO algorithm, Temporally-Aware Mirror Preference Optimization (TA-MPO), which demonstrate promising results in an LLM alignment task. We summarize our contributions below.

1. We perform a systematic evaluation of eight existing PO algorithms on automatically generated preference datasets with varying levels of data quality, noise levels and initial policy. We see that most existing algorithms struggle when dealing with noise and mixed-quality data.
2. We introduce a novel family of offline PO algorithms using mirror descent, named Mirror Preference Optimization (MPO), which can be easily parameterized and explored via ES.
3. For both noisy and mixed-quality settings, we find and describe a PO algorithm within our framework that largely outperforms all the considered baselines in our MuJoCo benchmark.
4. We demonstrate that takeaways from our analysis on the MuJoCo setting, as well as the characteristics of the discovered PO algorithms, can be successfully transferred onto LLM tasks. In particular, we show that our TA-MPO algorithm significantly improves upon the baselines.

Overall, we show that it is possible to gain relevant insights on LLM alignment algorithms through a systematic analysis in simple, computationally inexpensive environments such as MuJoCo.

2 Preliminaries

Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, T, \mu)$ denote an episodic Markov Decision Process, where $\mathcal{S}$ and $\mathcal{A}$ are respectively the state and action spaces, $P(s' \mid s, a)$ is the transition probability from state s to s' when taking action a, $r(s, a) \in [0, 1]$ is the reward function, T is the maximum episode length, and μ is a starting state distribution. A policy $\pi \in (\Delta(\mathcal{A}))^{\mathcal{S}}$, where $\Delta(\mathcal{A})$ is the probability simplex over $\mathcal{A}$, represents the behavior of an agent on an MDP, whereby at state $s \in \mathcal{S}$ the agents takes actions according to the probability distribution $\pi(\cdot \mid s)$. Let $\tau = \{(s_t, a_t)\}_{t=0}^{T-1}$ denote a trajectory of length T and, with a slight overload of notation, let $\pi(\tau) = \prod_{t=0}^{T-1} \pi(a_t|s_t)$ and $r(\tau) = \sum_{t=0}^{T-1} r(s_t, a_t)$. Lastly, let $\pi(\cdot|\tau)$ be a distribution over $(\Delta(\mathcal{A}))^T$ defined as $\pi(\cdot|s_0) \times \cdots \times \pi(\cdot|s_{N-1})$.

Our objective is to find a policy $\pi^\star$ that maximizes the expected cumulative reward of an episode, that is

$$\pi^\star \in \operatorname*{argmax}_\pi \mathbb{E}_{\tau\sim(\mu,\pi,P)} r(\tau) := \operatorname*{argmax}_\pi \mathbb{E}_{s_0\sim\mu, a_t, s_{t+1}} \sum_{t=0}^{T-1} r(s_t, a_t), \tag{1}$$

where $a_t \sim \pi(\cdot|s_t)$ and $s_{t+1} \sim P(\cdot|s_t, a_t)$. Let $\mathcal{D} = \{(s_0^i, \tau_w^i, \tau_l^i)_{i=1}^N\}$ be a preference dataset, where each tuple (s_0, τ_w, τ_l) consists of a starting state s_0 and two trajectories with starting state s_0. Each pair of trajectories is ranked by a judge, who determines a chosen trajectory τ_w ("win") and a rejected trajectory τ_l ("lose"), based on the cumulative rewards $r(\tau_w)$ and $r(\tau_l)$. Most settings assume the judge ranks trajectories according to the Bradley-Terry model (Bradley & Terry, 1952), whereby the probability of choosing τ_w over τ_l is defined as

$$\mathbb{P}(\tau_w \succ \tau_l) = \frac{\exp(r(\tau_w))}{\exp(r(\tau_w)) + \exp(r(\tau_l))} = \sigma(r(\tau_w) - r(\tau_l)), \tag{2}$$

where σ is the sigmoid function. In this work, we consider an offline training setting, where the agent aim to solve the optimization problem in (1) but only has access to the the dataset $\mathcal{D}$ and cannot collect further data. We also assume the agent does not have access to either the transition probability P, the reward function r, or the MDP $\mathcal{M}$.

2.1 Alignment to preference feedback

There are several algorithms in the literature to optimize the objective in (1) using a preference dataset $\mathcal{D}$. We describe supervised fine-tuning (SFT), DPO and ORPO, as they are among the most popular and as many methods can be seen as a variation of one of these algorithms.

SFT SFT is an initial alignment phase, where the policy π_0 is trained to imitate high-quality demonstration data. The starting policy π_0 is updated to minimize the cross-entropy loss $\ell(\pi, (s_0, \tau_w, \tau_l)) = -\log(\pi(\tau_w))$. We call *reference policy* π_{ref} the policy obtained at the end of this procedure.

DPO Direct Preference Optimization (DPO) consists in solving a maximum likelihood estimation problem and a policy optimization problem in a single step. The maximum likelihood estimation problem is the one to find an estimate of the true reward function that governs how the preferences are expressed, that is

$$\widehat{r} \in \operatorname*{argmax}_{r_\theta} \mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}} \sigma(r_\theta(\tau_w) - r_\theta(\tau_l)), \tag{3}$$

for a parametrized reward class $\{r_\theta : \theta \in \Theta\}$.The policy optimization problem is the one to maximize the expected reward, that is

$$\pi^\star \in \operatorname*{argmax}_{\pi} \mathbb{E}_{s_0\sim\mathcal{D},\tau\sim(\pi,P)} \left[\sum_{t=0}^{T-1} \mathbb{E}_{a\sim\pi(\cdot|s_t)} \widehat{r}(s_t, a) - \beta D_{\text{KL}}(\pi(\cdot|\tau), \pi_{\text{ref}}(\cdot|\tau)) \right], \tag{4}$$

where D_{KL} represents the KL-divergence and is introduced to prevent the policy from moving too far away from the dataset distribution.

DPO merges these two problems by using the agent itself to implicitly represent the reward model. It consists in optimizing the objective

$$\pi^\star \in \operatorname*{argmax}_{\pi} \mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}} \left[\log\sigma \left(\beta \left(\log \frac{\pi(\tau_w)}{\pi_{\text{ref}}(\tau_w)} - \log \frac{\pi(\tau_l)}{\pi_{\text{ref}}(\tau_l)} \right) \right) \right], \tag{5}$$

which is obtained by plugging the theoretical solution of (4) in the maximum likelihood problem in (3). Refer to Appendix D for details. Thanks to its simplicity, DPO has been widely adopted to fine-tune LLMs (Yuan et al., 2024; Jiang et al., 2024).

A known issue of DPO is that it pushes probability mass away from the preference dataset and to unseen responses (Xu et al., 2024b), which can cause the final policy to deviate significantly from the reference policy, even when the reference policy aligns well with human preferences. To mitigate this risk, DPO is usually applied for a few epochs.

ORPO ORPO further simplifies the training pipeline and addresses the distribution shift issue present in DPO. It merges the SFT and DPO steps into one, optimizing the unified objective

$$\pi^\star \in \operatorname*{argmax}_{\pi} \mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}} \left[\underbrace{\log \pi(\tau_w)}_{\text{SFT}} + \lambda \underbrace{\log\sigma \left(\log \left(\text{odds}_\pi(\tau_w) \right) - \log \left(\text{odds}_\pi(\tau_l) \right) \right)}_{\text{preference optimization}} \right] \tag{6}$$

where $\text{odds}_\pi(\tau) = \pi(\tau)/(1 - \pi(\tau))$. ORPO gets rid of the need for a reference model by adding an SFT term to the preference optimization objective function, and uses this term to prevent the optimized policy from moving too far away from the dataset distribution. Additionally, the SFT term prevents pushing probability mass away from the preference dataset, addressing the distribution shift issue present in DPO.

Research on preference optimization has been very active and many methods have been proposed. We present a summary of some among the most popular algorithms in Table 3 and a brief discussion in Appendix A. Beyond these implicit algorithms, there are several other methods that explicitly solve the maximum likelihood problem in (3) and use the learned reward model to optimize the objective in (4) with an RL algorithm. Overall, RLHF is a superior approach and the industry standard, but is more computationally expensive and complex to implement. For a detailed discussion and comparison between DPO-like methods and PPO, refer to Appendix G.

2.2 Mirror Maps

We review the concept of mirror map, which will be needed when describing our methodology. For a convex set $\mathcal{X} \subseteq \mathbb{R}^{|\mathcal{A}|}$, a *mirror map* $h : \mathcal{X} \to \mathbb{R}$ is defined as a strictly convex, continuously differentiable and essentially smooth function* function that satisfies $\nabla h(\mathcal{X}) = \mathbb{R}^{|\mathcal{A}|}$. Essentially, a mirror map is a function whose gradient allows bijective mapping between the primal space $\mathcal{X}$ and the dual space $\mathbb{R}^{|\mathcal{A}|}$. The specific class of mirror maps that we are going to use is the ω-potential mirror map class, to which most mirror maps considered in the literature belong.

Definition 2.1 (ω-potential mirror map Krichene et al. (2015))**.** For $u \in (-\infty, +\infty]$, $\omega \leq 0$, an *ω-potential* is defined as an increasing C^1-diffeomorphism $\phi : (-\infty, u) \to (\omega, +\infty)$ such that

$$\lim_{x \to -\infty} \phi(x) = \omega, \ \lim_{x \to u} \phi(x) = +\infty, \ \int_0^1 \phi^{-1}(x)dx \leq \infty.$$

For any ω-potential ϕ, the associated mirror map is $h_\phi(\pi(\cdot|s)) = \sum_{a \in \mathcal{A}} \int_1^{\pi(a|s)} \phi^{-1}(x)dx$. When $\phi(x) = e^{x-1}$ we recover the negative entropy mirror map, while we recover the ℓ_2-norm when $\phi(x) = 2x$ (refer to Appendix E). Mirror maps in this class are simple to implement in practice, where $\mathcal{A}$ is often large, as they can be parametrized by a scalar function instead of a multi-dimentional one. Additionally, the same ω-potential ϕ can be used to generate mirror maps for different action spaces, allowing the insights obtained for one action space to easily generalize to others. An ω-potential mirror map h_ϕ induces a *Bregman divergence* (Bregman, 1967), which is defined as

$$\mathcal{D}_{h_\phi}(\pi(\cdot|s), \pi'(\cdot|s)) := h_\phi(\pi(\cdot|s)) - h_\phi(\pi'(\cdot|s)) - \langle \nabla h_\phi(\pi'(\cdot|s)), \pi(\cdot|s) - \pi'(\cdot|s) \rangle,$$

where $\mathcal{D}_{h_\phi}(\pi(\cdot|s), \pi'(\cdot|s)) \geq 0$ for all $x, y \in \mathcal{Y}$. When $\phi(x) = e^{x-1}$, $\mathcal{D}_{h_\phi}$ is equivalent to the KL-divergence, while we recover the Euclidean distance when $\phi(x) = 2x$ (refer to Appendix E). When the Bregman divergence is employed as a regularization term in optimization problems, tuning the mirror map allows us to control the geometry of the updates of the parameters to be optimized, determining when to take large or small updates based on the current value of the parameters.

2.3 Evolution Strategies

OpenAI-ES (Salimans et al., 2017) is a popular method to be able to optimize non-differentiable functions and it has been widely used to meta-learn objectives (Lu et al., 2022; Jackson et al., 2024), as it obtains an unbiased estimate of the gradient (unlike second order gradient methods). The gradient $\nabla_\zeta F(\zeta)$ is estimated using:

$$\mathbb{E}_{\epsilon \sim \mathcal{N}(0, I_d)} \left[\frac{\epsilon}{2\sigma} (\widehat{F}(\zeta + \sigma\epsilon) - \widehat{F}(\zeta - \sigma\epsilon)) \right],$$

where $\mathcal{N}(0, I_d)$ is the multivariate normal distribution, d is the number of parameters, $\widehat{F}$ is an estimate of F, and $\sigma > 0$ is a hyperparameter regulating the variance of the perturbations.

3 Mirror Preference Optimization

We introduce Mirror Preference Optimization (MPO), a new framework for preference optimization that generalizes DPO and ORPO. We start by replacing the KL-divergence penalty term in the objective in (4) with a Bregman divergence and aim to solve the problem

$$\pi^\star \in \operatorname*{argmax}_\pi \mathbb{E}_{s_0 \sim \mathcal{D}, \tau \sim (\pi, P)} \left[\sum_{t=0}^{T-1} \mathbb{E}_{a \sim \pi(\cdot|s_t)} r(s_t, a) - \beta D_h(\pi(\cdot|\tau), \pi_{\text{ref}}(\cdot|\tau)) \right], \tag{7}$$

where D_h is the Bregman divergence induced by a mirror map h. This new objective allows us to enforce different types of regularization, which, as we show later in the paper, can be tailored to account for specific properties of the preference dataset. Following the same intuition used to obtain the DPO objective, we have the following result.

Theorem 3.1. *Let h_ϕ be a 0-potential mirror map and $\pi^\star$ be a solution to the optimization problem in (7). If $\pi_{\text{ref}}(a|s) > 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}$, we have that*

$$r(\tau) = \beta \phi^{-1}(\pi^\star(\tau)) - \beta \phi^{-1}(\pi_{\text{ref}}(\tau)) + c(s_0), \tag{8}$$

where $c(s_0)$ is a normalization constant that depends only on s_0.

*A function h is *essentially smooth* if $\lim_{x \to \partial\mathcal{X}} \|\nabla h(x)\|_2 = +\infty$, where $\partial\mathcal{X}$ denotes the boundary of $\mathcal{X}$.

We provide a proof for Theorem 3.1 in Appendix D. The next step is to model the reward using a classification problem based on the reward difference rather than the maximum likelihood problem in (3), as suggested by Tang et al. (2024). That is, our aim is to solve the optimization problem

$$\widehat{r} \in \operatorname{argmax}_{r_\theta} \mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}} g(r_\theta(\tau_w) - r_\theta(\tau_l)), \tag{9}$$

where g is an increasing function. We give further details on this interpretation of reward modeling in Appendix C. By plugging (8) in the optimization problem in (9), we obtain the objective:

$$\pi^\star \in \operatorname{argmax}_\pi \mathbb{E}_{\mathcal{D}}\big[g\big(\beta(\phi^{-1}(\pi(\tau_w)) - \phi^{-1}(\pi_{\text{ref}}(\tau_w)) - \phi^{-1}(\pi(\tau_l)) + \phi^{-1}(\pi_{\text{ref}}(\tau_l)))\big)\big], \tag{10}$$

where $\mathbb{E}_{\mathcal{D}}$ is equivalent to $\mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}}$.

Two-step MPO (2S-MPO) We can use the objective in (10) to define a class of two-step PO algorithms, which consist of a preliminary SFT phase to obtain the reference policy π_{ref} and a PO phase which optimizes (10). When $\phi = e^x$, the (10) is equivalent to (5) and we recover DPO.

One-step MPO (1S-MPO) By adding an SFT term to (10) and by setting the π_{ref} be the uniform distribution, we obtain a class of one-step PO algorithms. These algorithms consists in a single phase, where we optimize the objective

$$\pi^\star \in \operatorname{argmax}_\pi \mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}}\big[\psi(\pi(\tau_w)) + \lambda g\left(\phi^{-1}(\pi(\tau_w)) - \phi^{-1}(\pi(\tau_l))\right)\big], \tag{11}$$

where ψ is an ω-potential. Here, the terms $-\phi^{-1}(\pi_{\text{ref}}(\tau_w))$ and $\phi^{-1}(\pi_{\text{ref}}(\tau_l))$ have canceled out due to π_{ref} being uniform. We note that setting π_{ref} to be the uniform distribution is equivalent to replacing the Bregman divergence penalty in (7) with the mirror map $h(\pi(\cdot|\tau))$, which enforces a form of entropy regularization. When $\psi(x) = \log(x)$ and $\phi^{-1}(x) = \log(x/(1-x))$, (11) recovers the ORPO objective in (6).

Temporally-Aware MPO (TA-MPO) Lastly we design a variation of MPO that gradually switches from SFT to PO. TA-MPO consists of single-phase algorithms that optimize the objective

$$\pi^\star \in \operatorname{argmax}_\pi \mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}}\big[(1-\alpha(t))\psi(\pi(\tau_w)) + \alpha(t) g\left(\phi^{-1}(\pi(\tau_w)) - \phi^{-1}(\pi(\tau_l))\right)\big], \tag{12}$$

where $\alpha : [0,1] \to [0,1]$ is an increasing function of the percentage of training progress.

The objectives in (10), (11), and (12) allow us to implement a variety of preference optimization algorithms, while benefiting from a theoretical justification. In the following, we will show that it is possible to parametrize and optimize g, ψ, and ϕ^{-1} to obtain new algorithms that outperform baselines.

3.1 Meta Learning PO objectives

To search the space of PO algorithms we have defined, we employ a neural network parametrization for g, ψ, and ϕ^{-1}, which we optimize using evolutionary strategies (Salimans et al., 2017).

Similarly to Alfano et al. (2024), we parameterize g, ψ and ϕ^{-1} as a one layer neural network with 126 hidden units and non-negative kernels, where the activation functions are equally split among:

$$x,\ (x)_+^2,\ x^3,\ (x)_+^{1/2},\ (x)_+^{1/3},\ \log((x)_+),\ e^x,\ \tanh(x),\ \log(\text{clip}(x)/(1-\text{clip}(x))),$$

where $(x)_+ = \max(x, 0)$ and $\text{clip}(x) = \max(\min(x, 1), 0)$. The non-negative kernels and the increasing activation functions guarantee the monotonicity of g, ψ, and ϕ^{-1}, while the several different activation functions facilitate expressing complex functions. To ensure that we are able to recover the DPO and ORPO objectives, we add $a\log(x)$, $b\log(x)$ and $c\log(x/(1-x))$ to the final outputs of g, ψ and ϕ^{-1}, respectively, where $a, b, c \geq 0$.

To search for the best g, ψ and ϕ^{-1} within this class, we employ the OpenAI-ES strategy. Denote by ζ the parameters of g, ψ and ϕ^{-1} and by π^ζ the final policy obtained optimizing the objective in (11) when using the parametrized ψ and ϕ^{-1}. Lastly, let $F(\zeta)$ be the expected cumulative reward of π^ζ, i.e. $F(\zeta) = \mathbb{E}_{\tau\sim(\mu,\pi^\zeta,P)} r(\tau)$. We then use Adam (Kingma & Ba, 2015) to update the parameters ζ using the estimated gradient. In practice, to compute (19), we sample 128 values of ϵ to obtain 256 perturbed objective functions. We then train 256 agents with the perturbed objective functions on an a preference dataset. To measure the value of each agent, i.e. $\widehat{F}(\zeta')$ for all perturbed ζ', we sample 100 trajectories on the target environment for each agent, and take the average cumulative reward as estimate for the value of the agent. Refer to Appendix F for further discussion and details on the ES methodology.

We consider both the case where we fix $g = \log\sigma$ and learn ψ and ϕ^{-1}, and the case where we learn all three functions. We perform the evolution on both the two-step and one-step MPO classes.

4 MuJoCo Experiments

Our first set of experiments is carried out on continuous RL tasks in MuJoCo. In particular, we show the performance of all the algorithms presented in Table 3 across several settings and we compare it with the performance of our discovered objectives. To maximize computational efficiency, all our MuJoCo experiments are implemented in JAX (Bradbury et al., 2018) using the `brax` (Freeman et al., 2021) and evosax (Lange, 2022) libraries. We report our hyper-parameters in Appendix J.

4.1 Tasks

To reproduce the typical conditions of LLM fine-tuning, which involve a pre-trained model, we consider a setting where the task is to adapt a pre-trained agent to meet the original objective while adhering to an additional stylistic constraint. Specifically, in the `Ant` environment, we start from an agent that has been pre-trained on the standard `Ant` goal of moving forward and enforce the objective of avoiding the use of one of its legs. This is accomplished by introducing the `Three-legged-ant` (`TLA`) environment, a modified version of `Ant` where utilizing the fourth leg results in significant penalties.

The offline preference optimization task is defined as follows. We train one agent (the original agent) in the original `Ant` environment, achieving a reward of 6000, and another (the target agent) in the `TLA` environment, which achieves a reward of 3900. For comparison, the original agent achieves a reward of 1700 in the `TLA` environment. We then generate a preference dataset of 1280 rows, each with two trajectories of length 1000 starting from the same state. Each trajectory is generated by either the original or the target agent, depending on the current setting. A Bradley-Terry judge ranks each pair of trajectories and declares a winner, based on their true cumulative reward. We consider three variations of the preference dataset, each meant to represent a common issue of real world data.

- **Base dataset**: for each pair of trajectories, one is generated by the original agent and one by the target agent.
- **Noisy dataset**: same as the base dataset but each chosen/rejected pair of labels given by the judge is flipped with probability ε.
- **Mixed-quality dataset**: each trajectory in the dataset is generated by an agent selected at random between the original and the target one. The resulting dataset will consist of, approximately, 25% comparisons between two trajectories from the target agent, 50% comparisons between trajectories of different agents, and 25% comparisons between two trajectories of the original agent.

We also consider training a randomly initialized agent on the `Hopper` environment. This setting addresses the case where a behavior has to be learned from the preference dataset and there is no prior knowledge of the task available. We report the results of the experiments for this task in Appendix H.3.

4.2 Results

We provide the results of our experiments for TLA in Table 1, which reports the performance of several PO algorithm and of our discovered objectives. We performed a hyperparameter search for each algorithm-dataset combination and only report the performance of the best hyperparameters. All algorithms are run for 12 epochs over the preference dataset, with the exception of DPO, IPO, SimPO and R-DPO, which are run for 2 epochs after 10 epochs of SFT. We provide an additional noisy setting ($\varepsilon = 0.2$) and performance for other existing algorithms in Table 4 in Appendix H.1.

We notice that none of the human-designed algorithms manages to recover the performance of the target agent and that most of them experience a drop in performance in mixed-quality and noisy settings.

Importance of SFT The first group within Table 1 shows that SFT plays a key role in the performance across different settings. SimPO, which does not contain an SFT term nor an SFT step, is at the top of the leaderboard on the base and the mixed-quality setting but performs poorly on all noisy settings. IPO and DPO, which do not contain an SFT term but have an SFT step, are among the top performers on the base, mixed-quality and low noise settings. Their performance finally drops when the noise level reaches 0.3. Lastly, the algorithms that present an SFT term in their objectives, e.g. CPO, ORPO and, obviously, SFT, exhibit a subotpimal performance in the base and mixed-quality settings but are much more robust to noise than the other algorithms.

Table 1: **Three Legged Ant (TLA)**. Performance of existing and discovered MPO algorithms on TLA, for various dataset settings. For each algorithm-dataset combination, we report the average value and standard error of 25 trained agents. For each discovered MPO algorithm, we specify on which setting it was discovered and report its performance across all settings (with fixed hyperparameters). We underline the highest (or two highest if their confidence interval overlaps) average performance among the human-designed algorithm and report in bold the overall highest, for each setting.

	Base	Mixed Quality	Noisy ($\varepsilon = 0.1$)	Noisy ($\varepsilon = 0.3$)
RRHF (Yuan et al., 2023)	2789 ± 285	2245 ± 134	1730 ±442	330 ± 552
SLiC-HF (Zhao et al., 2023)	3255 ± 66	2478 ± 54	2329 ±289	1135 ± 224
DPO (Rafailov et al., 2024)	3528 ± 58	2766 ± 89	3082 ±80	1519 ±140
IPO (Azar et al., 2024)	3618 ± 44	2937 ± 85	3162 ± 66	1133 ± 115
CPO (Xu et al., 2024a)	3450 ± 55	2322 ± 208	2967 ± 58	2000 ±35
ORPO (Hong et al., 2024)	3087 ± 322	2500 ± 71	2841 ± 50	1953 ± 37
R-DPO (Park et al., 2024)	2606 ± 65	2107 ± 40	2099 ± 50	1667 ± 24
SimPO (Meng et al., 2024)	3683 ±78	3117 ± 185	2314 ± 752	-3828 ± 341
SFT	3287 ± 62	2344 ± 40	2733 ±45	2049 ±33
Our algorithms ↓				
LPO	3774 ± 102	2841 ± 46	3617 ± 69	1569 ± 156
With $g = \log \sigma$				
1S-MPO (mixed-quality)	3206 ± 330	3153 ± 274	1319 ± 714	-3967 ± 382
1S-MPO (noisy, $\varepsilon = 0.1$)	3789 ± 60	3210 ± 60	**3813 ± 47**	3279 ± 83
2S-MPO (mixed-quality)	3595 ± 57	2785 ± 78	3197 ± 58	1687 ± 58
2S-MPO (noisy, $\varepsilon = 0.1$)	3551 ± 58	2784 ± 63	3190 ± 62	1569 ± 122
With parametrized g				
1S-MPO (mixed-quality)	3560 ± 333	**3627 ± 79**	3371 ± 410	2681 ± 251
2S-MPO (mixed-quality)	3736 ± 51	3202 ± 64	3488 ± 73	2253 ± 125
1S-MPO (noisy, $\varepsilon = 0.1$)	**3861 ± 79**	3075 ± 87	3724 ± 59	1771 ± 107
2S-MPO (noisy, $\varepsilon = 0.1$)	3701 ± 52	3178 ± 59	3490 ± 95	2074 ± 136
1S-MPO (noisy, $\varepsilon = 0.3$)	**3931 ± 69**	3244 ± 55	**3834 ± 82**	**3417 ± 82**
Temporally-aware				
TA-MPO (1)	3577 ± 45	2730 ± 63	3106 ± 62	1971 ± 35
TA-MPO (2)	3625 ± 49	3088 ± 69	3443 ± 53	1988 ± 39
TA-MPO (3)	3352 ± 57	2256 ± 44	2725 ± 46	1923 ± 27

Keep optimizing Furthermore, while RRHF and SliC-HF are very similar, SliC-HF allows the PO part of the objective to be clipped when $\pi(\tau_w) > \pi(\tau_l) + \delta$, rather than when $\pi(\tau_w) > \pi(\tau_l)$. This modification leads to a higher performance on all tasks, demonstrating that it is beneficial to keep optimizing the policy even if $\pi(\tau_w) > \pi(\tau_l)$. To further stress this point, we consider the objective

$$\pi^\star \in \operatorname{argmax}_\pi \mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}}\left[\lambda \log \pi_\theta(\tau_w|x) - \log \pi_\theta(\tau_l|x)\right],$$

which we call Linear Preference Optimization (LPO). LPO corresponds to SLiC-HF with $\delta = +\infty$ and obtains better results than most of the existing algorithms in Table 1, confirming that it is important to design objectives that do not flatten when $\pi(\tau_w) > \pi(\tau_l)$.

Discovered objectives Table 1 also reports the performance of our discovered objectives, for both the case where we set the monotonic transformation g to be the logarithmic function and where we parametrize and learn it. We learn a separate objective for each dataset setting and report the performance of each learned objective on all settings. Differently from the human-designed algorithms, the discovered objectives recover the performance of the target agent in multiple instances and are more robust to the mixed-quality and noisy settings. We note that allowing the evolution procedure to learn g leads to a better performance in all dataset settings. Additionally, we have that the objectives discovered within the two-step MPO class always have a lower performance than those within the one-step MPO class. This is probably due to the ability to modify the SFT term in the one-step MPO class, which is not present in the two-step class.

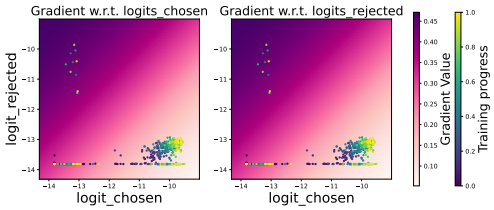

Figure 1: SimPO

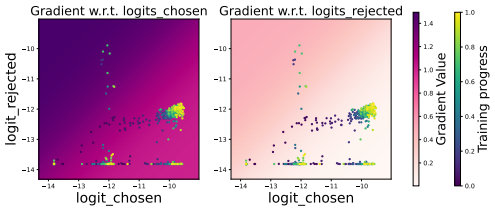

Figure 2: ORPO

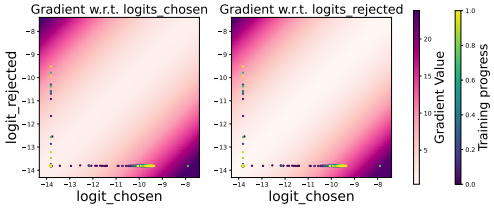

Figure 3: Discovered 1S-MPO (shuffled)

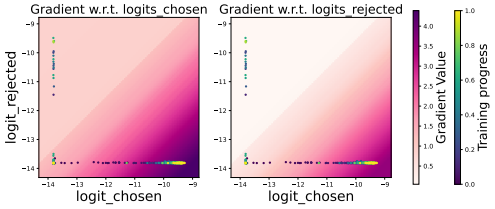

Figure 4: Discovered 1S-MPO (noisy)

Figure 5: Absolute value of the gradient of SimPO, ORPO, 1S-MPO (shuffled), and 1S-MPO (noisy, $\epsilon = 0.3$). The dots are sampled datapoints from the training distribution.

When exploring the one-step MPO class with $g = \log \sigma$, our discovery procedure always recovers a variation of CPO. That is, we obtain an objective that can be approximated as

$$\pi^\star \in \operatorname{argmax}_\pi \mathbb{E}_{(s_0,\tau_w,\tau_l)\sim\mathcal{D}}\left[\alpha \log(\pi(\tau_w)) + \lambda \log \sigma\big(\beta \log(\pi(\tau_w)) - \beta \log(\pi(\tau_l))\big)\right],$$

where the coefficients α and β depend on the setting. In particular, we have a low value for α and a high value for β in the noisy settings, while we observe the opposite in the mixed-quality setting. These results confirm the observations made on the hand-crafted objectives, whereby objectives with an SFT term are more robust to noise and objectives without are more robust to mixed-quality trajectories. When we search the two-step MPO class with $g = \log \sigma$, we recover DPO.

Figure 5 provides a visualization of the gradient of some of the objectives discovered when we parametrize and meta-learn g. In particular, we show the objectives discovered within the one-step MPO class on the mixed-quality and noisy ($\varepsilon = 0.3$) settings. For comparison, we provide the same plots for the ORPO and SimPO objectives. The hand-crafted algorithms present a larger gradient when $\pi(\tau_w) < \pi(\tau_l)$ and a smaller one when $\pi(\tau_w) > \pi(\tau_l)$, that is, they induce large updates when the data-point contradicts the current behaviour of the agent, and small otherwise.

The objective discovered on the noisy dataset has the opposite behavior, meaning that it only optimizes the more robust-to-noise SFT term when $\pi(\tau_w) < \pi(\tau_l)$. As $\log(\pi(\tau_w)) - \log(\pi(\tau_l))$ becomes larger, it shifts toward increasingly large updates thanks to the PO term. A similar pattern appears in the mixed-quality dataset, where the objective also increases its updates as the difference $\log(\pi(\tau_w)) - \log(\pi(\tau_l))$ grows. The key distinction between these two losses is that the shuffled loss triggers high-gradient updates even when $\pi(\tau_w) < \pi(\tau_l)$. We highlight once more that both discovered objectives advocate to keep optimizing the policy even when $\pi(\tau_w) > \pi(\tau_l)$.

4.3 Including temporal awareness

We build an algorithm within the TA-MPO family using the insights gained in the previous section. In particular, we keep the standard SFT loss, which was rediscovered in all our experiments, and use SimPO for the PO component of TA-MPO, given its high performance and its ability to continue optimizing the policy even when $\pi(\tau_w) > \pi(\tau_l)$. That is, we define the objective

$$\pi^\star \in \operatorname{argmax}_\pi \mathbb{E}_{\mathcal{D}}\left[(1-\alpha(t)) \log \pi_\theta(\tau_w|x) + \alpha(t) \log \sigma(\beta(\log \pi_\theta(\tau_w|x) - \log \pi_\theta(\tau_l|x)) - \gamma)\right]. \quad (13)$$

We try three expressions for α, designed to put most of the weight on the SFT component of (13) at the start of training and switch to PO towards the end of training:

$$(1):\ \alpha(t) = t; \quad (2):\ \alpha(t) = t^2; \quad (3):\ \alpha(t) = \sigma(20(x - 0.75)).$$

Table 1 shows promising results for temporally-aware PO algorithms, as the objective in (13) with the version (2) of α matches or surpasses all human-designed algorithms in all settings.

Table 2: **Alpaca Eval LLM results**. We report win-rates and standard error (length controlled win-rates and standard error in parenthesis) for three combinations of base model and preference dataset. We report in bold font the highest winrate (length-controlled winrate) for each column.

	gemma-7b[†], dpo-mix-7k[‡]	gemma-7b, capybara-7k[§]	mistral-7b[¶], dpo-mix-7k
CPO	28.9±1.6 (21.9±0.2)	29.2±1.6 (25.3±0.3)	31.0±1.6 (21.7±0.3)
ORPO	27.4±1.6 (21.5±0.2)	28.2±1.6 (24.2±0.3)	28.0±1.6 (21.1±0.3)
DPO	30.7±1.6 (**31.0±0.3**)	37.9±1.7 (32.8±0.2)	32.8±1.6 (29.5±0.3)
SimPO	32.5±1.7 (25.8±0.2)	27.3±1.6 (23.6±0.2)	28.1±1.6 (22.6±0.3)
TA-MPO (1)	31.5±1.6 (25.0±0.2)	34.2±1.7 (30.0±0.2)	39.6±1.7 (30.4±0.2)
TA-MPO (2)	27.8±1.6 (23.2±0.3)	33.4±1.7 (29.6±0.2)	36.6±1.7 (27.7±0.2)
TA-MPO (3)	**35.4±1.7** (29.1±0.1)	**39.4±1.7 (33.8±0.1)**	**41.6±1.7 (30.6±0.2)**

5 Experiments: LLM transfer

We show that the insights obtained in the MuJoCo environments can be transferred to the LLM alignment setting. In particular, we test the TA-MPO algorithm in (13), which is designed to continue to optimize the policy even when $\pi(\tau_w) > \pi(\tau_l)$, as all our discovered algorithms do. We evaluate TA-MPO on LLM alignment, comparing its performance against baselines for three combinations of base model and preference dataset, as shown in Table 2.

To tune the LLMs, we modify the Alignment Handbook library (Tunstall et al.) to include the TA-MPO objective in (13). We evaluate the tuned LLMs against GPT-4, using the AlpacaEval library (Li et al., 2023) and `Llama-3.1-70B-Instruct` as a judge. For all combinations of starting LLM, dataset, and PO algorithm, we perform 4 update epochs and set the learning rate to 5e-5 and β to 0.05. In the case of DPO and SimPO, we performed 3 epochs of SFT, with learning rate 5e-5, and 1 epoch of DPO/SimPO, with learning rate 5e-7 and $\beta = 0.05$. Refer to Appendix K for further details.

Table 2 shows the effectiveness of the TA-MPO objective in (13), which presents a high winrate for all schedules of α. In particular, TA-MPO with the sigmoid schedule for α has the highest winrate in all tasks and the highest length controlled winrate in two out of three tasks. Additionally, TA-MPO requires only one stage of training, while DPO and SimPO require two, i.e. SFT and PO.

We also tested the static algorithms discovered in the MuJoCo environment. Those with $g = \log \sigma$, which rediscovered CPO, exhibited performance similar to CPO itself. In contrast, algorithms with a parameterized g function, which learned to greedily optimize the policy even when $\pi(\tau_w) > \pi(\tau_l)$, performed poorly. We hypothesize that these algorithms overfit to the TLA task, where our datasets sufficiently cover the state-action space. On the other hand, in LLM tuning—where data is sparse relative to the environment—greedy methods are more susceptible to over-optimization. At the same time, the objectives discovered on TLA are only used to logits within -14 and -8, as shown in Figure 5, and have less regular shape outside of this subset. Despite this, our experiments with LLMs demonstrate that knowledge and insights gained from MuJoCo can still transfer effectively to other tasks.

6 Conclusion

We have introduced a novel framework for Preference Optimization algorithms, as well as a methodology for the automatic discovery of PO algorithms using evolutionary strategies. Through a systematic evaluation across diverse settings in MuJoCo environments, we have demonstrated that the performance of our discovered objectives consistently exceeds the performance of existing methods, particularly in noisy and mixed-quality datasets where many baselines underperform. Our analysis in MuJoCo also revealed a common shortcoming among current baselines: truncating the loss whenever $\pi(\tau_w) > \pi(\tau_l)$. Using this insight, we proposed a temporally-aware algorithm, TA-MPO, that avoids such loss truncation and gradually switches from the SFT step to the PO step. We then tested this objective on an LLM fine-tuning task, achieving significant improvements over existing methods, thereby confirming the broader applicability of our approach.

[†] https://huggingface.co/google/gemma-7b
[‡] https://huggingface.co/datasets/argilla/dpo-mix-7k
[§] https://huggingface.co/datasets/argilla/distilabel-capybara-dpo-7k-binarized
[¶] https://huggingface.co/mistralai/Mistral-7B-v0.3

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Riad Akrour, Marc Schoenauer, and Michèle Sebag. April: Active preference-learning based reinforcement learning, 2012. URL https://arxiv.org/abs/1208.0984.

Carlo Alfano, Sebastian Towers, Silvia Sapora, Chris Lu, and Patrick Rebeschini. Meta-learning the mirror map in policy mirror descent. *arXiv preprint arXiv:2402.05187*, 2024.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, 2024.

Erdem Biyik and Dorsa Sadigh. Batch active preference-based learning of reward functions. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto (eds.), *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pp. 519–528. PMLR, 29–31 Oct 2018. URL https://proceedings.mlr.press/v87/biyik18a.html.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 1952.

Lev M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 1967.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 2017.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017. URL https://arxiv.org/abs/1703.03400.

C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax - a differentiable physics engine for large scale rigid body simulation, 2021. URL http://github.com/google/brax.

Alexander David Goldie, Chris Lu, Matthew Thomas Jackson, Shimon Whiteson, and Jakob Nicolaus Foerster. Can learned optimization make reinforcement learning less difficult?, 2024. URL https://arxiv.org/abs/2407.07082.

Jiwoo Hong, Noah Lee, and James Thorne. Reference-free monolithic preference optimization with odds ratio. *arXiv preprint arXiv:2403.07691*, 2024.

Audrey Huang, Wenhao Zhan, Tengyang Xie, Jason D Lee, Wen Sun, Akshay Krishnamurthy, and Dylan J Foster. Correcting the mythos of kl-regularization: Direct alignment without overparameterization via chi-squared preference optimization. *arXiv preprint arXiv:2407.13399*, 2024.

Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in atari, 2018. URL https://arxiv.org/abs/1811.06521.

Matthew Thomas Jackson, Chris Lu, Louis Kirsch, Robert Tjarko Lange, Shimon Whiteson, and Jakob Nicolaus Foerster. Discovering temporally-aware reinforcement learning algorithms. In *International Conference on Learning Representations*, 2024.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Louis Kirsch, Sjoerd van Steenkiste, and Jürgen Schmidhuber. Improving generalization in meta reinforcement learning using learned objectives, 2020. URL https://arxiv.org/abs/1910.04098.

Walid Krichene, Syrine Krichene, and Alexandre Bayen. Efficient bregman projections onto the simplex. In *IEEE Conference on Decision and Control*, 2015.

Robert Tjarko Lange. evosax: Jax-based evolution strategies. *arXiv preprint arXiv:2212.04180*, 2022.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval, 2023.

Bo Liu, Xidong Feng, Jie Ren, Luo Mai, Rui Zhu, Haifeng Zhang, Jun Wang, and Yaodong Yang. A theoretical understanding of gradient bias in meta-reinforcement learning, 2022.

Chris Lu, Jakub Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered policy optimisation. *Advances in Neural Information Processing Systems*, 2022.

Chris Lu, Samuel Holt, Claudio Fanconi, Alex J Chan, Jakob Foerster, Mihaela van der Schaar, and Robert Tjarko Lange. Discovering preference optimization algorithms with and for large language models. *arXiv preprint arXiv:2406.08414*, 2024.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*, 2024.

Luke Metz, C. Daniel Freeman, Samuel S. Schoenholz, and Tal Kachman. Gradients are not all you need, 2022.

Arkadi Nemirovski and David B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley Interscience, 1983.

Junhyuk Oh, Matteo Hessel, Wojciech M Czarnecki, Zhongwen Xu, Hado P van Hasselt, Satinder Singh, and David Silver. Discovering reinforcement learning algorithms. *Advances in Neural Information Processing Systems*, 2020.

Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. Disentangling length from quality in direct preference optimization. *arXiv preprint arXiv:2403.19159*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 2024.

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

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

Yuda Song, Gokul Swamy, Aarti Singh, Drew Bagnell, and Wen Sun. The importance of online data: Understanding preference fine-tuning via coverage. In *ICML 2024 Workshop: Aligning Reinforcement Learning Experimentalists and Theorists*, 2024.

Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Remi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Avila Pires, and Bilal Piot. Generalized preference optimization: A unified approach to offline alignment. In *Forty-first International Conference on Machine Learning*, 2024.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alvaro Bartolome, Alexander M. Rush, and Thomas Wolf. The Alignment Handbook. URL https://github.com/huggingface/alignment-handbook.

Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. Beyond reverse kl: Generalizing direct preference optimization with diverse divergence constraints. In *The Twelfth International Conference on Learning Representations*, 2023.

P.J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990. doi: 10.1109/5.58337.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. In *Forty-first International Conference on Machine Learning*, 2024a.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is dpo superior to ppo for llm alignment? a comprehensive study, 2024b. URL https://arxiv.org/abs/2404.10719.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E Weston. Self-rewarding language models. In *Forty-first International Conference on Machine Learning*, 2024.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023.

Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023.

A Baseline PO Algorithms

Table 3: Overview of popular PO algorithms. The objective is to be maximized and $(\tau_w, \tau_l) \sim \mathcal{D}$.

Method	Objective
RRHF (Yuan et al., 2023)	$\lambda \log \pi_\theta(\tau_w\|x) - \max\left(0, -\frac{1}{\|\tau_w\|}\log \pi_\theta(\tau_w\|x) + \frac{1}{\|\tau_l\|}\log \pi_\theta(\tau_l\|x)\right)$
SLiC-HF (Zhao et al., 2023)	$\lambda \log \pi_\theta(\tau_w\|x) - \max(0, \delta - \log \pi_\theta(\tau_w\|x) + \log \pi_\theta(\tau_l\|x))$
DPO (Rafailov et al., 2024)	$\log \sigma\left(\beta \log \frac{\pi_\theta(\tau_w\|x)}{\pi_{\text{ref}}(\tau_w\|x)} - \beta \log \frac{\pi_\theta(\tau_l\|x)}{\pi_{\text{ref}}(\tau_l\|x)}\right)$
IPO (Azar et al., 2024)	$-\left(\log \frac{\pi_\theta(\tau_w\|x)}{\pi_{\text{ref}}(\tau_w\|x)} - \log \frac{\pi_\theta(\tau_l\|x)}{\pi_{\text{ref}}(\tau_l\|x)} - \frac{1}{2\tau}\right)^2$
CPO (Xu et al., 2024a)	$\log \pi_\theta(\tau_w\|x) + \log \sigma(\beta \log \pi_\theta(\tau_w\|x) - \beta \log \pi_\theta(\tau_l\|x))$
ORPO (Hong et al., 2024)	$\log \pi(\tau_w) + \lambda \log \sigma\left(\log\left(\text{odds}_\pi(\tau_w)\right) - \log\left(\text{odds}_\pi(\tau_l)\right)\right)$
R-DPO (Park et al., 2024)	$\log \sigma\left(\beta \log \frac{\pi_\theta(\tau_w\|x)}{\pi_{\text{ref}}(\tau_w\|x)} - \beta \log \frac{\pi_\theta(\tau_l\|x)}{\pi_{\text{ref}}(\tau_l\|x)} + (\alpha\|\tau_w\| - \alpha\|\tau_l\|)\right)$
SimPO (Meng et al., 2024)	$\log \sigma\left(\frac{\beta}{\|\tau_w\|}\log \pi_\theta(\tau_w\|x) - \frac{\beta}{\|\tau_l\|}\log \pi_\theta(\tau_l\|x) - \gamma\right)$

These methods include RRHF, which uses length-normalized log-likelihood, and SLiC-HF, which uses direct log-likelihood and incorporates SFT. The comparison also includes IPO, a theoretically-based approach that handles pairwise preferences differently than DPO. Another method, CPO, combines sequence likelihood as a reward with an SFT objective. Finally, R-DPO modifies the original DPO by adding regularization to prevent length exploitation.

B Related Work

Automatic Discovery of Preference Optimization Loss Functions Several works in the literature have shown that it is possible to discover machine learning algorithms that outperform algorithms manually designed by researchers (Oh et al., 2020; Lu et al., 2022; Jackson et al., 2024; Alfano et al., 2024). An approach particularly relevant to our method is DiscoPOP by Lu et al. (2024), which leverages an LLM to discover objective functions for LLM tuning. They consider a different space of objective functions from us, as they replace the log-sigmoid in (5) with a generic loss function, following the framework built by Tang et al. (2024). Additionally, instead of searching over a space of parametrized functions, they ask the LLM to generate loss functions in code space. This distinction suggests that our approaches could be complementary, as the model discovered by DiscoPOP could be paired with our learned mirror map. Lastly, DiscoPOP optimizes its objective function directly on the final task, whereas we adopt a two-stage process—optimizing the loss function on a separate task (MuJoCo) and later transferring it to the LLM setting. This transferability underscores the broader applicability of our approach.

Generalisations of DPO A generalization of DPO alternative to ours is f-DPO (Wang et al., 2023), which consists in replacing the KL-divergence in (1) with an f-divergence and then apply the same heuristic as DPO to obtain the final objective function. We note that the KL-divergence is the only f-divergence to be also a Bregman divergence, and vice-versa. They empirically demonstrate that different f-divergences lead to different balances between alignment performance and generation diversity, highlighting the trade-offs inherent to this class of algorithms. Huang et al. (2024) further explore this class of PO algorithm and individuate an f-divergence for which f-DPO is robust to overoptimization.

C Reward Modeling

In Equation (9), we utilize the interpretation of reward modeling as a binary classification problem given by Tang et al. (2024), which we summarize here. Let $z = (\tau_1, \tau_2)$ be a pair of trajectories and $\ell \in \{-1, 1\}$ be the associated label that states whether τ_1 is preferred to τ_2 ($\ell = 1$) or not ($\ell = -1$). We want to find a function $\hat{\ell}(z) \in \mathbb{R}$ such that $\text{sign}(\hat{\ell}(z))$ is a good estimate of ℓ. For a dataset

$\mathcal{D} = \{z_i, \ell_i\}_{i=1}^N$, the classification loss (or 0-1 loss) is

$$L(\widehat{\ell}, \mathcal{D}) = \mathbb{E}_{\mathcal{D}}\left[1 - \text{sign}\left(\hat{\ell}(z) \cdot \ell\right)\right], \tag{14}$$

which is often approximated with a surrogate

$$L_f(\widehat{\ell}, \mathcal{D}) = \mathbb{E}_{\mathcal{D}}\left[f\left(\hat{\ell}(z) \cdot \ell\right)\right], \tag{15}$$

for a function $f : \mathbb{R} \to \mathbb{R}$. This approximation is possible because, when f is decreasing (or convex), (14) and (15) have the same minimizer, as we prove in the following. Denote $p_1(z) = \mathbb{P}(\ell = 1|z)$, then the conditional surrogate loss at z is

$$L_f(\widehat{\ell}, x) = p_1(z) f\left(\hat{\ell}(z)\right) + (1 - p_1(z)) f\left(-\hat{\ell}(z)\right). \tag{16}$$

The minimizer $\widehat{\ell}^\star$ of (15) is such that $\widehat{\ell}^\star(z)$ minimizes (16). While the minimizer to (16) might not be computable explicitly, we can show $\widehat{\ell}^\star(z) > 0 \iff p_1(z) > 1/2$ when f is a decreasing function, meaning that $\widehat{\ell}^\star$ is also the minimizer of (14). Firstly, we have that

$$\begin{aligned} L_f(\widehat{\ell}, x) - L_f(-\widehat{\ell}, x) &= p_1(z) f\left(\hat{\ell}(z)\right) + (1 - p_1(z)) f\left(-\hat{\ell}(z)\right) \\ &\quad - p_1(z) f\left(-\hat{\ell}(z)\right) - (1 - p_1(z)) f\left(\hat{\ell}(z)\right) \\ &= (2p_1(z) - 1)\left(f\left(\hat{\ell}(z)\right) - f\left(-\hat{\ell}(z)\right)\right) \end{aligned} \tag{17}$$

Since f is decreasing, we have for all $\widehat{\ell}(z) > 0$ that $f(\widehat{\ell}(z)) < f(-\widehat{\ell}(z))$. Plugging this into (17), we obtain that, if $p_1(z) > 1/2$, $L_f(\widehat{\ell}, x) < L_f(-\widehat{\ell}, x)$ for all $\widehat{\ell}(z) > 0$. Therefore, the minimizer $\widehat{\ell}^\star(z)$ of (16) must be positive. The opposite can be proved in the same manner.

Equation (9) can be obtained by setting $f = -g$, for an increasing function g, and

$$\widehat{\ell}(z) = \widehat{r}(\tau_1) - \widehat{r}(\tau_2).$$

D Proof of Theorem 3.1

We provide here a proof for our main result, i.e. Theorem 3.1. The proof to obtain the DPO objective in (5) follows by taking $\phi = e^x$.

Theorem D.1 (Theorem 3.1)**.** *Let h_ϕ be a 0-potential mirror map and $\pi^\star$ be a solution to the optimization problem in (7). If $\pi_{\text{ref}}(a|s) > 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}$, we have that*

$$r(\tau) = \phi^{-1}(\pi^\star(\tau)) - \phi^{-1}(\pi_{\text{ref}}(\tau)) + c(s_0), \tag{18}$$

for all trajectories τ, where $c(s_0)$ is a normalization constant that depends only on s_0.

Proof. We use the KKT conditions to solve (7), i.e.

$$\pi^\star \in \operatorname*{argmax}_\pi \mathbb{E}_{s_0 \sim \mathcal{D}, \tau \sim (\pi, P)}\left[\sum_{t=0}^{T-1} \mathbb{E}_{a \sim \pi(\cdot|s_t)}[r(s_t, a)] - \beta D_h(\pi(\cdot|\tau), \pi_{\text{ref}}(\cdot|\tau))\right]$$

We use the stationarity condition to obtain the equation

$$\begin{aligned} &\nabla_{\pi(\tau)}\left[\sum_{t=0}^{T-1} \mathbb{E}_{a \sim \pi(\cdot|s_t)}[r(s_t, a)] - \beta D_h(\pi(\cdot|\tau), \pi_{\text{ref}}(\cdot|\tau)) - \lambda \sum_{\tau' : s_0 \in \tau'} \pi(\tau') - \lambda + \sum_{\tau' : s_0 \in \tau'} \alpha(\tau')\pi(\tau')\right] \\ &= r(\tau) - \beta\phi^{-1}(\pi(\tau)) + \beta\phi^{-1}(\pi_{\text{ref}}(\tau)) - \lambda + \alpha(\tau) = 0, \end{aligned}$$

for all initial states $s_0 \in \mathcal{S}$ and for all trajectories τ starting from s_0. Rearranging, we obtain that

$$\pi(\tau) = \phi\big((r(\tau) + \beta\phi^{-1}(\pi_{\text{ref}}(\tau)) - \lambda + \alpha(\tau))/\beta\big).$$

Since $0 \notin \operatorname{dom} \phi^{-1}$, due to the definition of a 0-potential, and $\pi_{\text{ref}}(\tau) > 0$, we have that $\pi(\tau) > 0$ for all trajectories τ. Invoking the complementary slackness condition, whereby $\alpha(\tau)\pi(\tau) = 0$ for all trajectories τ, we have that $\alpha(\tau) = 0$ for all trajectories τ. Therefore, we have that

$$r(\tau) - \beta\phi^{-1}(\pi(\tau)) + \beta\phi^{-1}(\pi_{\text{ref}}(\tau)) - \lambda = 0$$

The theorem statement is obtained by rearranging the last equation and denoting $c(s_0) = \lambda$ □

E Further discussion of ω-potentials

We show here two examples of Bregman divergence induced by an ω-potential mirror map, that is when $\phi(x) = e^{x-1}$ and when $\phi(x) = x$. If $\phi(x) = e^{x-1}$, the associated mirror map is defined as

$$\begin{aligned} h_\phi(\pi(\cdot|s)) &= \sum_{a\in\mathcal{A}} \int_1^{\pi(a|s)} \phi^{-1}(x)dx = \sum_{a\in\mathcal{A}} \int_1^{\pi(a|s)} (\log(x)+1)dx \\ &= \sum_{a\in\mathcal{A}} \pi(a \mid s)\log(\pi(a \mid s)) - \pi(a \mid s) + \pi(a \mid s) \\ &= \sum_{a\in\mathcal{A}} \pi(a \mid s)\log(\pi(a \mid s)), \end{aligned}$$

which is the negative entropy. Plugging this expression in the definition of Bregman divergence we obtain

$$\begin{aligned} \mathcal{D}_h(x,y) &= h(x) - h(y) - \langle \nabla h(y), x-y\rangle \\ &= \sum_{a\in\mathcal{A}} x_a \log(x_a) - y_a\log(y_a) - (\log(y_a)-y_a)(x_a-y_a) \\ &= \sum_{a\in\mathcal{A}} x_a \log(x_a/y_a), \end{aligned}$$

which is the definition of the KL-divergence. If $\phi(x) = 2x$, the associated mirror map is defined as

$$h_\phi(\pi(\cdot|s)) = \sum_{a\in\mathcal{A}} \int_1^{\pi(a|s)} \phi^{-1}(x)dx = \sum_{a\in\mathcal{A}} \int_1^{\pi(a|s)} 2xdx = \sum_{a\in\mathcal{A}} \pi(a \mid s)^2,$$

which is the ℓ_2-norm. Plugging this expression in the definition of Bregman divergence we obtain

$$\mathcal{D}_h(x,y) = h(x) - h(y) - \langle \nabla h(y), x-y\rangle = \sum_{a\in\mathcal{A}} x_a^2 - y_a^2 - (2y_a)(x_a-y_a) = \sum_{a\in\mathcal{A}} (x_a-y_a)^2,$$

which is the definition of the Euclidean distance.

F Further discussion on Evolution Strategies

Evolution Strategies (ES) represent a powerful, backpropagation-free method for optimizing complex functions, that has been particularly successful in the context of long-horizon, noisy, and bi-level optimization tasks such as RL and meta-RL. ES, and in particular the OpenAI-ES algorithm (Salimans et al., 2017), rely on perturbation-based sampling to estimate gradients without requiring backpropagation through the entire computational graph. This feature makes ES well-suited for tasks with long computational graphs, for instance algorithms with many updates, where, due to memory constraints, traditional gradient-based methods have to resort to gradient truncation, introducing bias (Werbos, 1990; Metz et al., 2022; Liu et al., 2022).

In our setting, we use ES to search for the best ψ and ϕ^{-1} within the parametrized class introduced in Section 3.1, so that an agent trained using the objective in (11) achieves the highest value. Denote by ζ the parameters of ψ and ϕ^{-1} and by π^ζ the final policy obtained optimizing the objective in (11) when using the parametrized ψ and ϕ^{-1}. Lastly, let $F(\zeta)$ be the expected cumulative reward (or value) of π^ζ, i.e. $F(\zeta) = \mathbb{E}_{\tau\sim(\mu,\pi^\zeta,P)} r(\tau)$. At each iteration, we estimate the gradient $\nabla_\zeta F(\zeta)$ as

$$\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I_d)}\left[\frac{\epsilon}{2\sigma}(\widehat{F}(\zeta+\sigma\epsilon) - \widehat{F}(\zeta-\sigma\epsilon))\right], \tag{19}$$

where $\mathcal{N}(0, I_d)$ is the multivariate normal distribution, d is the number of parameters, $\widehat{F}$ is an estimate of F, and $\sigma > 0$ is a hyperparameter regulating the variance of the perturbations.

G Further discussion on Online vs Offline Methods

In the domain of RL and preference optimization, the choice between online and offline algorithms presents a critical trade-off, influencing computational efficiency, data requirements, and generalization capabilities. Online methods, such as PPO, iteratively collect and incorporate new data during

training. These inherently support exploration of the environment, enabling the discovery of novel strategies or behaviors that are not captured in pre-existing datasets. However, they need feedback for each generated "trajectory" (or response, in the LLM case), which might be expensive to obtain. Online methods are also more complex and particularly sensitive to hyperparameters, often requiring meticulous tuning for stability and efficiency.

Offline algorithms, such as DPO and its variants, rely entirely on pre-collected datasets. These methods are designed for efficiency and simplicity: they don't require any additional feedback from users and are therefore particularly effective in scenarios where feedback is delayed or unavailable. However, the reliance on static datasets means offline methods may struggle to generalize beyond the training data, particularly if the distribution shift between the training dataset and test time distribution is significant. Additionally, the performance of the algorithm is closely tied to the quality of the training dataset: noisy, biased, or corrupt datasets can severely degrade performance, as these methods cannot mitigate such issues through exploration or resampling.

In summary, RLHF (i.e., online) is considered the superior approach, particularly when substantial amounts of online labels are accessible. This makes it the industry standard (Xu et al., 2024b). While DPO has been theoretically equated to optimizing using PPO and a reward model trained on an offline dataset, recent empirical research (Tang et al., 2024) has challenged this notion. These studies have demonstrated that online methods, such as PPO, consistently outperform offline methods like DPO. This superiority is attributed to the benefits of on-policy sampling.

While DPO has occasionally outperformed PPO, it's important to note that several studies (Xu et al., 2024b; Song et al., 2024) have consistently shown PPO's overall superiority. DPO's relative strength lies in its simpler training regime, which avoids the complexities associated with reward model inaccuracies. However, DPO's performance is significantly limited by its sensitivity to distribution shift, especially when the offline preference data lacks diversity (Song et al., 2024). This limitation becomes particularly evident when querying the model with out-of-distribution data, a common challenge for methods relying solely on offline data. To mitigate this issue, DPO-iter (Xu et al., 2024b), which incorporates online data, has been proposed as a potential solution.

H MuJoCo Additional Results

H.1 TLA

We display additional results for the TLA task in Table 4. With respect to Table 1, we replace the mixed-quality setting with a noisy setting where the noise parameter ε is set to 0.2. We also include the KTO (Ethayarajh et al., 2024) and f-DPO (Jensen-Shannon) (Wang et al., 2023) algorithms.

H.2 Simplified expression for discovered objectives

Below, we report a simplified version of the objectives discovered for the shuffled and noisy ($\epsilon = 0.3$) settings, when g is parametrized:

$$\begin{aligned}\mathcal{L}(\tau_w, \tau_l) =& 0.82\mathcal{L}_{\text{SFT}}(\tau_w) + 1.7(\log(\pi_\theta(\tau_w)) - \log(\pi_\theta(\tau_l))) \\ &+ 0.33(\log(\pi_\theta(\tau_w)) - \log(\pi_\theta(\tau_l)))^2 + 0.36(\log(\pi_\theta(\tau_w)) - \log(\pi_\theta(\tau_l)))^3 \\ \mathcal{L}(\tau_w, \tau_l) =& 0.82\mathcal{L}_{\text{SFT}}(\tau_w) \\ &+ \max(1.39(\log(\pi_\theta(\tau_w)) - \log(\pi_\theta(\tau_l)))^2, 0.12(\log(\pi_\theta(\tau_w)) - \log(\pi_\theta(\tau_l))))\end{aligned}$$

H.3 Hopper Tasks

We consider a second set of simulations on MuJoCo, based on the Hopper environment. As for the previous sections, our experiments are implemented in JAX (Bradbury et al., 2018) using the `brax` (Freeman et al., 2021) and evosax (Lange, 2022) libraries. We report our hyper-parameters in Appendix J.

Differently from the `TLA` task, we consider a setting where the agent is randomly initialized and needs to learn a policy from scratch. On `Hopper`, we train an agent with an expected cumulative reward of 2100 (the expert agent) and an agent with an expected cumulative reward of 900 (the bad agent). We generate the preference datasets in the same way we do for `TLA`, with the exceptions that the

Table 4: **Three Legged Ant (TLA)**. Performance of existing and discovered MPO algorithms on TLA. For each algorithm-dataset combination, we report the average value and standard error of 25 trained agents. For each discovered MPO algorithm, we specify on which setting it was discovered and report its performance across all settings (with fixed hyperparameters). We report in bold the highest (or two highest if their confidence interval overlaps) average performance, for each setting.

	Base	Noisy ($\varepsilon = 0.1$)	Noisy ($\varepsilon = 0.2$)	Noisy ($\varepsilon = 0.3$)
RRHF	2789 ± 285	1730 ±442	749 ± 498	330 ± 552
SLiC-HF	3255 ± 66	2329 ±289	1964 ± 116	1135 ± 224
DPO	3528 ± 58	3082 ±80	2530 ± 97	1519 ±140
IPO	3618 ± 44	3162 ± 66	2392 ±136	1133 ± 115
CPO	3450 ± 55	2967 ± 58	2427 ± 44	2000 ±35
ORPO	3087 ± 322	2841 ± 50	2359 ± 43	1953 ± 37
R-DPO	2606 ± 65	2099 ± 50	1740 ± 36	1667 ± 24
SimPO	3683 ±78	2314 ± 752	118 ± 715	-3828 ± 341
SFT	3287 ± 62	2733 ±45	2345 ± 37	2049 ±33
KTO	1534 ± 34	1551 ± 31	1531 ± 49	1442 ± 35
f-DPO (Jensen-Shannon)	3621 ± 50	3192 ± 76	2494 ± 101	1633 ± 123
Our algorithms ↓				
LPO	3774 ± 102	3617 ± 69	2705 ± 370	1569 ± 156
With $g = \log \sigma$				
1S-MPO (mixed-quality)	3206 ± 330	1319 ± 714	-1625 ± 944	-3967 ± 382
1S-MPO (noisy, $\varepsilon = 0.1$)	3789 ± 60	**3813 ± 47**	3280 ± 83	3279 ± 83
2S-MPO (mixed-quality)	3595 ± 57	3197 ± 58	2487 ± 110	1687 ± 58
2S-MPO (noisy, $\varepsilon = 0.1$)	3551 ± 58	3190 ± 62	2552 ± 94	1569 ± 122
With parametrized g				
1S-MPO (mixed-quality)	3560 ± 333	3371 ± 410	3230 ± 259	2681 ± 251
2S-MPO (mixed-quality)	3736 ± 51	3488 ± 73	2992 ± 87	2253 ± 125
1S-MPO (noisy, $\varepsilon = 0.1$)	**3861 ± 79**	3724 ± 59	2845 ± 365	1771 ± 107
2S-MPO (noisy, $\varepsilon = 0.1$)	3701 ± 52	3490 ± 95	2886 ± 127	2074 ± 136
1S-MPO (noisy, $\varepsilon = 0.3$)	**3931 ± 69**	**3834 ± 82**	**3735 ± 84**	**3417 ± 82**

Table 5: **Hopper**. Perfomance of existing MPO algorithms on the Hopper setting. The agent is randomly initialised.

	Base	Mixed Quality	Noisy ($\epsilon = 0.1$)
DPO	1796 ± 78	458 ± 82	693 ± 131
IPO	2049 ± 13	1606 ± 91	739 ± 118
CPO	2078 ± 12	1078 ± 35	1813 ± 33
ORPO	2022 ± 15	1039 ± 20	1710 ± 33
SimPO	2027 ± 15	1460 ± 94	1794 ± 65

number of rows is 5120 and the trajectories are generated by either the expert or the bad agent. We consider the same three variations of the preference datasets used in `TLA`, where the expert agent corresponds to the target agent, and the bad agent corresponds to the original agent. We include more data compared to the TLA setting as it takes more datapoints for the agent to learn from scratch rather than to adapt to a slightly different objective (like in the TLA case).

Our Hopper results confirm our conclusions in the TLA setting. All algorithms but DPO come close to matching the performance of the expert agent (2100), with CPO being the best. We can see SimPO is the only algorithm that significantly outperforms the bad agent (performance of 900) in the Mixed Quality setting (the γ for SimPO was set very high, $\gamma = 10$, as a lower value significantly limited performance).

H.4 Further MuJoCo Analsys

In addition to the noisy dataset, we also considered a **bad judge** setting, where the judge would be more likely to swap the label of a pair of trajectories if their ground truth rewards were closer to each other. This is practically implemented as an increase in the temperature of the Bradley-Terry judge. However, we did not notice significantly different results compared to the simple noisy setting, therefore detailed results are not reported.

I Further discussion on Meta-Learning Algorithms

Meta-learning, or "learning to learn", has been extensively employed to automate the design of algorithms that can either adapt rapidly with minimal data samples or generalize effectively to unseen data, tasks, or environments. The development of broadly applicable algorithms is particularly critical in the context of preference optimization for LLMs. Here, LLMs are fine-tuned on relatively small datasets of offline data but must generalize to a virtually infinite range of potential user queries. Prior work in meta-learning has demonstrated success in developing generalizable optimization algorithms and loss functions (Lu et al., 2022; Jackson et al., 2024; Lu et al., 2024; Goldie et al., 2024; Kirsch et al., 2020).

At its core, meta-learning is defined as a bilevel optimization problem with an inner and an outer loop. The inner loop consists in an iterative optimization algorithm that trains agents to solve a predetermined task given a set of meta-parameters. The outer loop consists in evaluating the agents trained in the inner loop and update the meta-parameters accordingly, following some optimization method like second order gradient descent (Finn et al., 2017). The evaluation of the agents is typically done on a held-out dataset in supervised learning or by sampling trajectories on the environment simulator in RL (Lu et al., 2022; Jackson et al., 2024). In our setting, the inner loop is the offline preference optimization algorithm, while the outer loop is the agent evaluation on the environment (online) and the update of the meta-parameters ζ.

J MuJoCo Hyper-parameters

We give the hyper-parameters we use for training. The hyper-parameters specific to each algorithm are tuned for each task-data type combination. All the experiments were conducted on 4 NVIDIA L40S GPUs.

Table 6: Hyper-parameter settings for PO.

Parameter	Value
Number of epochs	12
Minibatch size	2
Learning rate	1e-3
Max gradient norm	1.3

Table 7: Hyper-parameter settings of OpenAI-ES.

Parameter	`Hopper`	`TLA`
Population Size	256	256
Number of generations	128	256
Sigma init	0.03	0.03
Sigma Decay	0.999	0.999
Learning rate	0.02	0.02

K LLM Hyper-parameters

We give the hyper-parameters we use for LLM training. All the experiments were conducted on 4 NVIDIA L40S GPUs.

Table 8: Hyper-parameter settings for LLM Training.

Parameter	Value
Gradient Accumulation Step	32
Batch Size	2
Total Batch Size	64
LoRA	Yes
LoRA Rank	128
LoRA Alpha	256
Lora Dropout	0.05
Max length	2048

6 Conclusion

In this last chapter, we summarize the main contributions of this thesis and outline open questions for future research.

6.1 Contributions

This thesis has explored and advanced the theoretical and algorithmic foundations of optimization methods for reinforcement learning, with a particular focus on policy gradient techniques. The central theme throughout has been the development of scalable, provably efficient optimization algorithms that address core challenges in modern RL—namely, scalability in multi-agent systems, expressivity in policy parameterizations, and alignment with human preferences.

In the multi-agent setting, we introduced a decentralized natural policy gradient algorithm that leverages assumptions about spatial decay of correlations to achieve convergence rates that are dimension-free with respect to the number of agents. This result provides an important step toward making reinforcement learning feasible in large-scale cooperative systems, offering both theoretical guarantees and practical scalability.

In the single-agent context, we proposed Approximate Mirror Policy Optimization (AMPO), a versatile framework that extends mirror descent techniques

to general policy parameterizations. This method is the first to achieve linear convergence guarantees in this setting, bridging a critical gap between theory and practice in policy gradient optimization. Our analysis also revealed how the choice of mirror map—an under-explored hyperparameter—has a significant effect on empirical performance, motivating more principled approaches to its selection.

Building upon these insights, we explored the domain of preference optimization, an increasingly important area for aligning RL agents with human feedback. By proposing the Mirror Preference Optimization (MPO) framework, we unified and generalized several state-of-the-art algorithms and demonstrated how evolutionary strategies can be used to discover robust optimization strategies tailored to specific data characteristics. The resulting algorithms not only outperform existing baselines in simulated environments but also provide useful insights that can be transferred in real-world tasks, such as large language model alignment.

6.2 Future directions

The contributions of this thesis open up several directions for future research, which we discuss below.

Can we design scalable multi-agent algorithms in more general settings? The algorithm designed in Chapter 2 is tailored to a specific kind of decay of correlations and to the setting of homogeneous cooperative agents. It would be interesting to understand whether it is possible to design scalable algorithm in different MARL setting, such as partially observable MDPs (Åström 1965) or heterogeneus or competing agents. Additionally, we could investigate settings with weaker or non-global spatial decay assumptions.

Can we improve the implementation of the AMPO algorithm? The main issue in the implementation of AMPO is that the updates in (1.25) and (1.24) are not explicit. One way to simplify the expression in (1.25) would be to leverage dual

averaging (Nesterov 2009) instead of mirror descent, obtaining the objective

$$\theta^{t+1} \in \operatorname*{argmin}_{\theta\in\Theta} \mathbb{E}_{(s,a)\sim v^t}\Big[f^\theta(s,a) - Q^t(s,a) - \frac{\eta_{t-1}}{\eta_t} f^t(s,a)\Big]^2.$$

A future direction is to understand whether this update enjoys the same convergence guarantees as AMPO.

Can we export the theoretical analysis of AMPO to other setting? One could apply the linear convergence theory of AMPO to other structural MDPs, e.g., linear MDP (Jin et al. 2020), block MDP (Du, Krishnamurthy, et al. 2019), factored MDP (Kearns and Koller 1999; W. Sun et al. 2019), RKHS linear MDP and RKHS linear mixture MDP (Du, S. Kakade, et al. 2021), to build new sample complexity results for these settings, since the assumptions of Theorem 4.3 in Chapter 3 do not impose any constraint on the MDP. On the other hand, it would be interesting to explore the interaction between the Bregman projected policy class and the expected Lipschitz and smooth policies (Yuan et al. 2022) and the Fisher-non-degenerate policies (Liu et al. 2020) to establish new improved sample complexity results in these settings, again thanks to the linear convergence theory of AMPO.

Another direction for future work is extending the policy update of AMPO to mirror descent algorithm based on value iteration and Bellman operators, such as MD-MPI (Geist et al. 2019), in order to extend existing results to the general parameterization setting.

Can we give a theoretical characterization for the impact of mirror maps in RL? While Chapter 4 provides some insights on the behavior of PMD for different mirror maps, a theoretical understanding of this phenomenon is still missing. An interesting direction could be to explore the implicit regularization properties of mirror maps in RL, as it has been done before in convex optimization (Vaskevicius et al. 2020; F. Wu and Rebeschini 2021; H. Sun et al. 2023).

Can the mirror map be adapted to the environment during training? All our analysis have been done with a mirror map that is kept constant throughout training. One could design an algorithm with a mirror map that changes during training in order to adapt to specific environment properties—e.g., sparsity or symmetry. To do so, one would have to innovate current proof techniques for mirror descent that rely on telescopic sums or recursions, to account for the changing mirror map.

Can we understand why some preference optimization algorithm work better than other? In Chapter 5 we provided a comparison of several preference optimization algorithms, together with some insights on the cause of the better performance of some algorithms. However, our investigation was limited to DPO-style algorithms and did not consider approaches that first train a reward model and then use it to train the agent through RL. In particular, we did not address the fact that this second class of approaches has usually better results than the first (Swamy et al. 2025). It would be interesting to provide more understanding of this phenomenon, even in a toy environment like MuJoCo.

References

Achiam, Josh et al. (2023). "Gpt-4 technical report". In: *arXiv preprint arXiv:2303.08774*.

Agarwal, Alekh et al. (2021). "On the Theory of Policy Gradient Methods: Optimality, Approximation, and Distribution Shift". In: *Journal of Machine Learning Research*.

Akrour, Riad, Marc Schoenauer, and Michèle Sebag (2012). "APRIL: Active Preference-learning based Reinforcement Learning". In: *arXiv preprint arXiv:1208.0984*.

Amari, Shun-Ichi (1998). "Natural gradient works efficiently in learning". In: *Neural computation*.

Åström, Karl Johan (1965). "Optimal control of Markov processes with incomplete state information I". In: *Journal of mathematical analysis and applications*.

Beck, Amir and Marc Teboulle (2003). "Mirror descent and nonlinear projected subgradient methods for convex optimization". In: *Operations Research Letters*.

Biyik, Erdem and Dorsa Sadigh (2018). "Batch Active Preference-Based Learning of Reward Functions". In: *Proceedings of The 2nd Conference on Robot Learning*.

Bradley, Ralph Allan and Milton E Terry (1952). "Rank analysis of incomplete block designs: I. The method of paired comparisons". In: *Biometrika*.

Bregman, Lev M. (1967). "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming". In: *USSR Computational Mathematics and Mathematical Physics*.

Bubeck, Sébastien (2015). "Convex optimization: Algorithms and complexity". In: *Foundations and Trends in Machine Learning*.

Cauchy, Augustin et al. (1847). "Méthode générale pour la résolution des systemes d'équations simultanées". In: *Comp. Rend. Sci. Paris*.

Christiano, Paul F et al. (2017). "Deep reinforcement learning from human preferences". In: *Advances in neural information processing systems*.

Deng, Yue et al. (2017). "Deep Direct Reinforcement Learning for Financial Signal Representation and Trading". In: *IEEE Transactions on Neural Networks and Learning Systems*.

Dobrusin, RL (1970). "Definition of a system of random variables by means of conditional distributions". In: *Theory of Probability and its Applications*.

Du, Simon, Sham Kakade, et al. (2021). "Bilinear Classes: A Structural Framework for Provable Generalization in RL". In: *International Conference on Machine Learning*.

Du, Simon, Akshay Krishnamurthy, et al. (2019). "Provably efficient RL with Rich Observations via Latent State Decoding". In: *International Conference on Machine Learning*.

Geist, Matthieu, Bruno Scherrer, and Olivier Pietquin (2019). "A Theory of Regularized Markov Decision Processes". In: *International Conference on Machine Learning*.

Georgii, Hans-Otto (2011). *Gibbs measures and phase transitions*.

Guo, Daya et al. (2025). "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning". In: *arXiv preprint arXiv:2501.12948*.

Ibarz, Borja et al. (2018). "Reward learning from human preferences and demonstrations in Atari". In: *arXiv preprint arXiv:1811.06521*.

Jackson, Matthew Thomas et al. (2024). "Discovering Temporally-Aware Reinforcement Learning Algorithms". In: *International Conference on Learning Representations*.

Jin, Chi et al. (2020). "Provably efficient reinforcement learning with linear function approximation". In: *Conference on Learning Theory*.

Kakade, Sham M (2001). "A natural policy gradient". In: *Advances in neural information processing systems*.

Kearns, Michael J. and Daphne Koller (1999). "Efficient Reinforcement Learning in Factored MDPs". In: *International Joint Conference on Artificial Intelligence*.

Kober, Jens, J. Andrew Bagnell, and Jan Peters (2013). "Reinforcement learning in robotics: A survey". In: *The International Journal of Robotics Research*, pp. 1238–1274.

Lan, Guanghui (2022). "Policy mirror descent for reinforcement learning: Linear convergence, new sampling complexity, and generalized problem classes". In: *Mathematical programming*.

Li, Yan and Guanghui Lan (2025). "Policy mirror descent inherently explores action space". In: *SIAM Journal on Optimization*.

Li, Yan, Guanghui Lan, and Tuo Zhao (2024). "Homotopic policy mirror descent: policy convergence, algorithmic regularization, and improved sample complexity". In: *Mathematical Programming*.

Lin, Yiheng et al. (2020). "Distributed reinforcement learning in multi-agent networked systems". In: *arXiv preprint arXiv:2006.06555*.

Liu, Yanli et al. (2020). "An improved analysis of (variance-reduced) policy gradient and natural policy gradient methods". In: *Advances in Neural Information Processing Systems*.

Lu, Chris et al. (2022). "Discovered policy optimisation". In: *Advances in Neural Information Processing Systems*.

Meng, Yu, Mengzhou Xia, and Danqi Chen (2024). "Simpo: Simple preference optimization with a reference-free reward". In: *arXiv preprint arXiv:2405.14734*.

Nemirovski, Arkadi and David B. Yudin (1983). *Problem Complexity and Method Efficiency in Optimization*. Wiley Interscience.

Nesterov, Yurii (2009). "Primal-dual subgradient methods for convex problems". In: *Mathematical programming*.

Qu, Guannan and Na Li (2019). "Exploiting Fast Decaying and Locality in Multi-Agent MDP with Tree Dependence Structure". In: *IEEE Conference on Decision and Control*.

Qu, Guannan, Yiheng Lin, et al. (2020). "Scalable Multi-Agent Reinforcement Learning for Networked Systems with Average Reward". In: *Advances in Neural Information Processing Systems*.

Qu, Guannan, Adam Wierman, and Na Li (n.d.). "Scalable reinforcement learning of localized policies for multi-agent networked systems". In: *Conference on Learning for Dynamics and Control*.

Rafailov, Rafael et al. (2024). "Direct preference optimization: Your language model is secretly a reward model". In: *Advances in Neural Information Processing Systems*.

Rechenberg, Ingo (1973). "Evolutionsstrategie". In: *Optimierung technischer Systeme nach Prinzipien derbiologischen Evolution*.

Salimans, Tim et al. (2017). "Evolution strategies as a scalable alternative to reinforcement learning". In: *arXiv preprint arXiv:1703.03864*.

Schulman, John et al. (2017). "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347*.

Schwefel, Hans-Paul (1977). "Numerische optimierung von computer-modellen mittels der evolutionsstrategie". In: *Cybernetics and System*.

Shalev-Shwartz, Shai, Shaked Shammah, and Amnon Shashua (2016). "Safe, Multi-Agent, Reinforcement Learning for Autonomous Driving". In: *arXiv preprint arXiv:1610.03295*.

Silver, David, Aja Huang, et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *Nature*.

Silver, David, Julian Schrittwieser, et al. (2017). "Mastering the game of Go without human knowledge". In: *Nature*.

Sun, Haoyuan et al. (2023). "A unified approach to controlling implicit regularization via mirror descent". In: *Journal of Machine Learning Research*.

Sun, Wen et al. (2019). "Model-based RL in Contextual Decision Processes: PAC bounds and Exponential Improvements over Model-free Approaches". In: *Conference on Learning Theory*.

Sutton, Richard S et al. (1999). "Policy gradient methods for reinforcement learning with function approximation". In: *Advances in neural information processing systems*.

Swamy, Gokul et al. (2025). "All roads lead to likelihood: The value of reinforcement learning in fine-tuning". In: *arXiv preprint arXiv:2503.01067*.

Team, Gemini et al. (2023). "Gemini: a family of highly capable multimodal models". In: *arXiv preprint arXiv:2312.11805*.

Vaskevicius, Tomas, Varun Kanade, and Patrick Rebeschini (2020). "The statistical complexity of early-stopped mirror descent". In: *Advances in Neural Information Processing Systems*.

Wu, Fan and Patrick Rebeschini (2021). "Implicit regularization in matrix sensing via mirror descent". In: *Advances in Neural Information Processing Systems*.

Xiao, Lin (2022). "On the Convergence Rates of Policy Gradient Methods". In: *Journal of Machine Learning Research*.

Yuan, Rui, Robert M. Gower, and Alessandro Lazaric (2022). "A general sample complexity analysis of vanilla policy gradient". In: *International Conference on Artificial Intelligence and Statistics*.

Zhan, Wenhao et al. (2023). "Policy Mirror Descent for Regularized Reinforcement Learning: A Generalized Framework with Linear Convergence". In: *SIAM Journal on Optimization*.