Understanding Curriculum Learning in Policy Optimization for Solving Combinatorial Optimization Problems

Runlong Zhou[*] Yuandong Tian[†] Yi Wu[‡] Simon S. Du[§]

February 3, 2022

Abstract

Over the recent years, reinforcement learning (RL) has shown impressive performance in finding strategic solutions for game environments, and recently starts to show promising results in solving combinatorial optimization (CO) problems, in particular when coupled with curriculum learning to facilitate training. Despite emerging empirical evidence, theoretical study on why RL helps is still at its early stage. This paper presents the first systematic study on policy optimization methods for solving CO problems. We show that CO problems can be naturally formulated as latent Markov Decision Processes (LMDPs), and prove convergence bounds on natural policy gradient (NPG) for solving LMDPs. Furthermore, our theory explains the benefit of curriculum learning: it can find a strong sampling policy and reduce the distribution shift, a critical quantity that governs the convergence rate in our theorem. For a canonical combinatorial problem, Secretary Problem, we formally prove that distribution shift is reduced *exponentially* with curriculum learning. Our theory also shows we can simplify the curriculum learning scheme used in prior work from multi-step to single-step. Lastly, we provide extensive experiments on Secretary Problem and Online Knapsack to empirically verify our findings.

1 Introduction

In recent years, machine learning techniques have shown promising results in solving combinatorial optimization (CO) problems, including traveling salesman problem (TSP, Kool et al. [2019b]), maximum cut [Dai et al., 2018] and satisfiability problem [Selsam et al., 2019]. While in the worst case some CO problems are NP-hard, in practice, the probability that we need to solve the worst-case problem instance is low [Cappart et al., 2021]. Machine learning techniques are able to find generic models that can efficiently solve the majority of a class of CO problems.

This paper concerns the approach of using reinforcement learning (RL) to solve CO problems. Many prior works exist, e.g., using Pointer Network in REINFORCE and Actor-Critic for routing problems [Nazari et al., 2018], combining Graph Attention Network with Monte Carlo Tree Search to solve TSP [Drori et al., 2020] and incorporating Structure-to-Vector Network in Deep Q-networks to solve maximum independent set problems [Cappart et al., 2019]. Compared to supervised learning, RL not only mimics existing heuristics, but also discover novel ones that humans have not thought of, for example chip design [Mirhoseini et al., 2021] and compiler optimization [Zhou et al., 2021c].

RL, as a convenient way to model sequential decision-making, fits well with the sequential nature of many CO problems. When used, RL is often coupled with specialized techniques including (a particular type of) curriculum learning [Kong et al., 2019], human feedback and correction (Pérez-Dattari et al. [2018], Scholten et al. [2019]), and policy aggregation (boosting, Brukhim et al. [2021]).

[*]Tsinghua University. Email: zhourunlongvector@gmail.com

[†]Facebook AI Research. Email: yuandong@fb.com

[‡]Tsinghua University. Email: jxwuyi@gmail.com

[§]University of Washington & Facebook AI Research. Email: ssdu@cs.washington.edu

While these hybrid techniques enjoy empirical success, the theoretical understanding is still limited: it is unclear when and why they improve the performance. In this paper, we particularly focus on *RL with Curriculum Learning* (Bengio et al. [2009], also named "bootstrapping" in Kong et al. [2019]): train the agent from an easy task and gradually increase the difficulty until the target task. Interestingly, these techniques exploit the special structures of CO problems.

Main Contributions. In this paper, we initiate the formal study on using RL for solving COs, with a particular emphasis on understanding the specialised techniques developed in this emerging subarea. Our contributions are summarized below.

- **Formalization.** For CO problems, we want to learn a single policy that enjoys good performance over *a distribution* of CO problem instances. This motivates us to use Latent MDP (LMDP) Kwon et al. [2021] instead of standard Markov Decision Process (MDP) formulation. We give concrete examples, Secretary Problem (SP) and Online Knapsack, to show how LMDP models CO problems. With this formulation, we can systematically analyze the performance of RL algorithms.

- **Provable Efficiency of Policy Optimization.** By leveraging recent theory on policy gradient algorithms for standard MDP Agarwal et al. [2020], we analyze the performance on Natural Policy Gradient (NPG) for LMDP. The performance bound is characterized by the number of iterations, the excess risk of policy evaluation, the transfer error, and the relative condition number κ that characterizes the distribution shift between the sampling policy and the optimal policy. To our knowledge, this is the first performance bound of policy optimization methods on LMDP.

- **Understanding and Simplifying Curriculum Learning.** Using our performance guarantee on NPG for LMDP, we study when and why *curriculum learning* is beneficial for RL in solving CO problems. Our main finding is that the main effect of curriculum learning is to give a stronger sampling policy. Specifically, curriculum learning can change the relative condition number κ, and in turn improve the convergence rate. For the Secretary Problem, we provably show that curriculum learning can *exponentially reduce* κ compared with using the naïve sampling policy. As direct implication of this finding, we show that the original multi-step curriculum learning proposed in Kong et al. [2019] can be significantly simplified: a single-step curriculum learning suffices. Lastly, in order to obtain a complete understanding, we study the failure mode of curriculum learning, in a way to help practitioners to decide whether to use curriculum learning based on their prior knowledge. To verify our theoretical findings, we conduct extensive experiments on two classical CO problems (Secretary Problem and Online Knapsack) and carefully track the dependency between the performance of the policy and κ.

2 Related Work

RL for CO. There has been a rich literature studying RL for CO problems. Bello et al. [2017] proposed a framework to tackle CO problems using RL and neural networks. Kool et al. [2019a] combined REINFORCE and attention technique to learn routing problems. Vesselinova et al. [2020] and Mazyavkina et al. [2020] are taxonomic surveys of contemporary RL approaches for graph problems. Bengio et al. [2020] summarized learning methods, algorithmic structures, learning objective design and discussed generalization and fine-tuning. In particular scaling to larger problems was mentioned as a major challenge.

Our paper is motivated by Kong et al. [2019] who focused on using RL to solve online CO problems, which means that the agent must make sequential and irrevocable decisions. Three online CO problems were mentioned in their paper: Online Matching, Online Knapsack and Secretary Problem (SP). Though matching and knapsack have (pseudo-) polynomial time exact solvers, their online versions can only be solved approximately [Huang et al., 2019, Albers et al., 2020]. For all algorithms provided by Kong et al. [2019], inputs are encoded in a length-independent manner, for example the i-th element of a n-length sequence is encoded by the fraction $\frac{i}{n}$ and other features, so that the agent can generalize to unseen n. In our experiments, we follow the same encoding. Kong et al. [2019] also proposed a bootstrapping / curriculum

learning approach: gradually increase the problem size after the model works sufficiently well on current problem size. This approach facilitated their training, and is one of our main focuses in this paper.
LMDP. We provide the exact definition of LMDP in Sec. 4.1. As studied by Steimle et al. [2021], in the general cases, optimal policies for LMDPs are *history dependent*. This is different from standard MDP cases where there always exists an optimal *stationary (history independent)* policy. They showed that even finding the optimal stationary policy is *NP-hard*. Kwon et al. [2021] investigated the sample complexity and regret bounds of LMDP. Specifically, they presented an exponential lower-bound for a general LMDP, then they derived an algorithm with polynomial sample complexity and sub-linear regret for two special cases (with context hindsight, or δ-strongly separated MDPs). In contrast to their work, we aim to find a *stationary* policy because for many CO problems of interest, such as SP and Online Knapsack, a stationary policy often suffices.
Convergence rate for policy gradient methods. Recently, there is line of work on the convergence rates of policy gradient methods for standard MDP (Bhandari and Russo [2021], Wang et al. [2020], Liu et al. [2020], Ding et al. [2021], Zhang et al. [2021]). For *Softmax tabular parameterization*, NPG can obtain an $O(1/T)$ rate [Agarwal et al., 2020] where T is the number of iterations; with entropy regularization, both PG and NPG achieves linear convergence [Mei et al., 2020, Cen et al., 2021]. For *log-linear policies*, sample-based NPG makes an $O(1/\sqrt{T})$ convergence rate, with assumptions on $\epsilon_{\text{stat}}, \epsilon_{\text{bias}}$ and κ [Agarwal et al., 2020] (see Def. 6); exact NPG with entropy regularization enjoys a linear convergence rate up to ϵ_{bias} [Cayci et al., 2021]. In this paper, we extend the analysis to LMDP.
Curriculum Learning. There are a rich body of literature on Curriculum Learning [Zhou et al., 2021b,a, 2020, Ao et al., 2021, Willems et al., 2020, Graves et al., 2017]. As surveyed in Bengio et al. [2009], Curriculum Learning has been applied to training deep neural networks and non-convex optimizations and improves the convergence in several cases. Narvekar et al. [2020] rigorously modeled curriculum as a directed acyclic graph, and named common types of curriculum. They then surveyed work on curriculum design for RL problems in aspects of task generation, sequencing and transfer learning.

3 Motivating Combinatorial Optimization Problems

In this subsection we introduce two motivating CO problems. We are particularly interested in these problems because they have all been extensively studied in CO and have closed-form, easy-to-implement policies as references. Furthermore, they were studied in Kong et al. [2019], the paper that motivates our work.

3.1 Secretary Problem

In SP, the goal is to maximize the *probability* of choosing the best among n candidates, where n is known. It is assumed that all n candidates have different scores to quantify their professional abilities. They arrive sequentially and when the i-th candidate shows up, the decision maker observes the relative ranking X_i among the first i candidates ($X_i = 1$ means being the best so far, and $X_i = 2$ means being the second best, and so on). A decision that whether to accept or reject the i-th candidate must be made *immediately* when the candidate comes, and such decisions *cannot be revoked.* Once one candidate is accepted, the game ends immediately.

The ordering of the candidates is unknown. There are in total $n!$ permutations, and an instance of SP is drawn from a unknown distribution over these permutations. In the classical SP, each permutation is sampled with equal probability. The optimal solution for the classical SP is the well-known $\frac{1}{\mathrm{e}}$-*threshold strategy*: reject all the first $\left\lfloor \frac{n}{\mathrm{e}} \right\rfloor$ candidates, then accept the first one who is the so-far best. In this paper, we also study some different distributions.

3.2 Online Knapsack (decision version)

In Online Knapsack problems the decision maker observes a known number of n items arriving sequentially, each with a value v_i and a size s_i revealed upon arrival. A decision that whether to accept or reject the i-th item must be made *immediately* when the item arrives, and such decisions *cannot be revoked.* At any time the accepted items should have their total size no larger than a known budget B.

The goal of standard Online Knapsack is to maximize the total value of accepted items. In this paper we study the decision version, denoted as OKD, whose goal is to maximize the *probability* of total value reaching a known target V.

We assume that all values and sizes are sampled independently from two fixed distributions, namely $v_1, v_2, \ldots, v_n \overset{\text{i.i.d.}}{\sim} F_v$ and $s_1, s_2, \ldots, s_n \overset{\text{i.i.d.}}{\sim} F_s$. In Kong et al. [2019] the experiments were carried out with $F_v = F_s = \text{Unif}_{[0,1]}$, and we also study other distributions.

Remark 1. A challenge in OKD is that the reward is sparse: the only reward signal is reward 1 when the total value of accepted items first exceeds V (see the detailed formulation in Sec. 7.2), unlike in Online Knapsack the reward of v_i is given instantly after the i-th item is successfully accepted. This makes random exploration hardly get reward signals, hence underscores the importance of curriculum learning.

4 Problem Setup

In this section, we first introduce LMDP and why it naturally formulates the CO problems.

4.1 Latent Markove Decision Process

Solving a CO problem entails handling a family of problem instances. Each instance can be modeled as a Markov Decision Process. However, for CO problems, we want to find one algorithm that works for a family of problem instances and performs well on average over an (unknown) distribution of this family. To this end, we adopt the concept of Latent MDP which naturally models CO problems.

Latent MDP [Kwon et al., 2021] is a collection of finitely many MDPs $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_M\}$ where $M = |\mathcal{M}|$. All the MDPs share state set $\mathcal{S}$, action set $\mathcal{A}$, initial state s_0 and horizon H. Each MDP $\mathcal{M}_m = (\mathcal{S}, \mathcal{A}, s_0, H, P_m, r_m)$ has its own transition model $P_m : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ and reward function $r_m : \mathcal{S} \times \mathcal{A} \to [0, 1]$. Let $w_1, w_2, \ldots, w_M$ be the mixing weights of MDPs such that $w_m > 0$ for any m and $\sum_{m=1}^{M} w_m = 1$. At the start of every episode, one MDP $\mathcal{M}_m \in \mathcal{M}$ is randomly chosen with probability w_m.

In general, the optimal policy of an LMDP is history-dependent and is PSPACE-hard to find (Cor. 1 and Prop. 3 in Steimle et al. [2021]). So in this paper we care only about finding the optimal *stationary* (history-independent) policy. A stationary policy is defined as $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ where $\Delta(\mathcal{A})$ is the probability simplex over $\mathcal{A}$. Let Π^{s} denote the class of all the stationary policies defined on $\mathcal{M}$.

Entropy regularized value function, Q-function and advantage function. The expected reward of executing π on M_m can be defined via value functions. Taking *entropy regularization* into consideration, we define the value function in a unified way: $V_{m,h}^{\pi,\lambda}(s)$ is defined as the sum of future λ-regularized rewards starting from s and executing π for h steps in M_m, i.e.,

$$V_{m,h}^{\pi,\lambda}(s) := \mathbb{E}\left[\sum_{t=0}^{h-1} r_m^{\pi,\lambda}(s_t, a_t) \,\middle|\, \mathcal{M}_m, \pi, s_0 = s\right],$$

where $r_m^{\pi,\lambda}(s,a) := r_m(s,a) + \lambda \ln \frac{1}{\pi(a|s)}$, and the expectation is with respect to the randomness of trajectory induced by π in M_m. Denote $V_{m,h}^{\pi}(s) = V_{m,h}^{\pi,0}(s)$.

For any $\mathcal{M}_m, \pi, h$, let

$$H_{m,h}^{\pi}(s) := \mathbb{E}\left[\sum_{t=0}^{h-1} \mathcal{H}(\pi(\cdot|s_t)) \,\middle|\, \mathcal{M}_m, \pi, s_0 = s\right],$$

where $\mathcal{H}(\pi(\cdot|s)) = \sum_{a \in \mathcal{A}} \pi(a|s) \ln \frac{1}{\pi(a|s)}$. Note that $0 \le \mathcal{H}(\pi(\cdot|s)) \le \ln |\mathcal{A}|$. In fact, $V_{m,h}^{\pi,\lambda}(s) = V_{m,h}^{\pi}(s) + \lambda H_{m,h}^{\pi}(s)$.

Denote $V^{\pi,\lambda} := \sum_{m=1}^{M} w_m V_{m,H}^{\pi,\lambda}(s_0)$ and $V^{\pi} = V^{\pi,0}$. We need to find $\pi^\star = \arg\max_{\pi\in\Pi^{\mathrm{s}}} V^{\pi}$, and denote $V^\star = V^{\pi^\star}$. Under regularization, we seek for $\pi_\lambda^\star = \arg\max_{\pi\in\Pi^{\mathrm{s}}} V^{\pi,\lambda}$ instead. Denote $V^{\star,\lambda} = V^{\pi_\lambda^\star,\lambda}$. Since

$$V^\star \leq V^{\pi^\star,\lambda} \leq V^{\star,\lambda} \leq V^{\pi_\lambda^\star,\lambda} + \lambda H \ln|\mathcal{A}|,$$

the regularized optimal policy $\pi_\lambda^\star$ can be nearly optimal as long as the regularization coefficient λ is small enough. For notational ease, we abuse $\pi^\star$ with $\pi_\lambda^\star$, and $d^\star$ with $d^{\pi_\lambda^\star}$.

The Q-function can be defined in a similar manner:

$$Q_{m,h}^{\pi,\lambda}(s,a) := \mathbb{E}\left[\sum_{t=0}^{h-1} r_m^{\pi,\lambda}(s_t,a_t) \;\middle|\; \mathcal{M}_m, \pi, (s_0,a_0)=(s,a)\right],$$

and the advantage function is defined as

$$A_{m,h}^{\pi,\lambda}(s,a) := Q_{m,h}^{\pi,\lambda}(s,a) - V_{m,h}^{\pi,\lambda}(s).$$

Modeling SP as LMDP. For SP, each instance is a permutation of length n, and each round an instance is drawn from an unknown distribution over all permutations. In the i-th step for $i \in [n]$, the state encodes the i-th candidate and relative ranking so far. The transition is deterministic according to the problem definition. A reward of one is given if and only if the agent accepts the best candidate in this instance.

Modeling ODK as LMDP. For OKD, each instance is a sequence of items with values and sizes drawn from unknown distributions. In the i-the step for $i \in [n]$, the states encodes the information of i-th item's value and size, the remaining budget, and the remaining target value to fulfill. The transition is also deterministic according to the problem definition, and a reward of one is given if and only if the agent obtains the target value.

4.2 Log-linear policy

Suppose we have a feature mapping function $\phi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^d$ where d denotes the dimension of feature space. Throughout the paper we assume that $\|\phi(s,a)\|_2 \leq B$[1]. Each policy in the log-linear policy class is of the form:

$$\pi_\theta(a|s) = \frac{\exp(\theta^\top \phi(s,a))}{\sum_{a'\in\mathcal{A}} \exp(\theta^\top \phi(s,a'))}, \text{ where } \theta \in \mathbb{R}^d.$$

Remark 2. Log-linear parameterization is a generalization of *Softmax tabular parameterization* by setting $d = |\mathcal{S}||\mathcal{A}|$ and $\phi(s,a) = \text{One-hot}(s,a)$. We prefer log-linear policies in that they are "scalable": if ϕ extracts important features from different $\mathcal{S}\times\mathcal{A}$s with a fixed dimension $d \ll |\mathcal{S}||\mathcal{A}|$, then a single π_θ can generalize.

4.3 Algorithm components

In this subsection we will introduce some necessary notions used by our main algorithm and main theorem.

Definition 1 (State(-action) Visitation Distribution)**.** *For any starting state s_0, the state visitation distribution and state-action visitation distribution at time step h with respect to π in $\mathcal{M}_m$ are defined as*

$$d_{m,h}^{s_0,\pi}(s) := \mathbb{P}(s_h = s \mid \mathcal{M}_m, \pi, s_0),$$
$$d_{m,h}^{s_0,\pi}(s,a) := \mathbb{P}(s_h = s, a_h = a \mid \mathcal{M}_m, \pi, s_0).$$

We will encounter a grafted distribution $\widetilde{d}_{m,h}^{s_0,\pi}(s,a) = d_{m,h}^{s_0,\pi}(s) \circ \text{Unif}_{\mathcal{A}}(a)$ which in general cannot be the state-action visitation distribution with respect to any policy. However, it can be attained by first acting

[1]For vector v, $\|v\|_2 = \sqrt{\sum_i v_i^2}$.

under π for h steps to get states then sample actions from the uniform distribution $\mathrm{Unif}_{\mathcal{A}}$. This distribution will be useful when we apply a variant of NPG, where the sampling policy is fixed.

Since s_0 is fixed throughout the paper, we omit the superscript s_0. Further we denote $d^\star_{m,h} = d^{\pi^\star}_{m,h}$ and $d^\theta_{m,h} = d^{\pi_\theta}_{m,h}$. We take $d^\clubsuit$ as a shorthand for the collection of $\{d^\clubsuit_{m,h}\}_{1\le m\le M, 0\le h\le H-1}$, here $\clubsuit$ can be any superscript.

We also need the following definitions for NPG. We note that these definitions different from the standard versions for MDP. Here, we also need to incorporate weights $\{w_m\}$ in the definitions to deal with LMDP.

Definition 2 (Compatible Function Approximation Loss)**.**

$$L(g;\theta,v) := \sum_{m=1}^M w_m \sum_{h=1}^H \mathop{\mathbb{E}}_{s,a\sim v_{m,H-h}} \left[\left(A^{\pi_\theta,\lambda}_{m,h}(s,a) - g^\top \nabla_\theta \ln \pi_\theta(a|s)\right)^2\right].$$

Definition 3 (Generic Fisher Information Matrix)**.**

$$\Sigma^\theta_v := \sum_{m=1}^M w_m \sum_{h=1}^H \mathop{\mathbb{E}}_{s,a\sim v_{m,H-h}} \left[\nabla_\theta \ln \pi_\theta(a|s) \left(\nabla_\theta \ln \pi_\theta(a|s)\right)^\top\right].$$

Particularly, denote $F(\theta) = \Sigma^\theta_{d^\theta}$ as the Fisher information matrix induced by π_θ.

Our main result is based on the Lyapunov potential function which depicts the closeness between $\pi^\star$ and any policy π:

Definition 4 (Lyapunov Potential Function [Cayci et al., 2021])**.** *We define the potential function $\Phi : \Pi^{\mathrm{s}} \to \mathbb{R}$ as follows: for any $\pi \in \Pi^{\mathrm{s}}$,*

$$\Phi(\pi) = \sum_{m=1}^M w_m \sum_{h=0}^{H-1} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,h}} \left[\ln \frac{\pi^\star(a|s)}{\pi(a|s)}\right].$$

Finally, we present the well-known Policy Gradient Theorem for finite horizon LMDP. The full statement and proof are deferred to Sec. D.

Theorem 5 (Policy Gradient Theorem)**.** *For any policy π_θ parameterized by θ,*

$$\nabla_\theta V^{\pi_\theta,\lambda} = \sum_{m=1}^M w_m \sum_{h=1}^H \mathop{\mathbb{E}}_{s,a\sim d^\theta_{m,H-h}} \left[Q^{\pi_\theta,\lambda}_{m,h}(s,a) \nabla_\theta \ln \pi_\theta(a|s)\right].$$

5 Learning Procedure

5.1 Natural Policy Gradient

The learning procedure generates a series of parameters and policies. Starting from θ_0, the algorithm updates the parameter by setting $\theta_{t+1} = \theta_t + \eta g_t$, where η is a predefined constant learning rate, and g_t is some specific update weight. Denote $\pi_t = \pi_{\theta_t}$, $V^{t,\lambda} = V^{\pi_t,\lambda}$ and $A^{t,\lambda}_{m,h} = A^{\pi_t,\lambda}_{m,h}$ for convenience.

In this paper we adopt the method of Natural Policy Gradient [Kakade, 2002]. According to Agarwal et al. [2020], the exact update weight of NPG satisfies $g_t = F(\theta_t)^\dagger \nabla_\theta V^{t,\lambda}$ and it is equivalent to $g_t \in \arg\min_g L(g;\theta_t, d^{\theta_t})$ (Lem. 13). When we only have samples, we use the approximate version of NPG: $g_t \approx \arg\min_{g\in\mathcal{G}} L(g;\theta_t,d^{\theta_t})$, where $\mathcal{G} = \{x : \|x\|_2 \le G\}$ for some hyper-parameter G.

We also use another type of NPG in experiments: instead of sampling from d^{θ_t} using the current policy π_t, we use a *fixed* sampling policy π_s to sample from $\widetilde{d}^{\pi_s}$. The update rule is $g_t \approx \arg\min_{g\in\mathcal{G}} L(g;\theta_t,\widetilde{d}^{\pi_s})$.

The main algorithm is shown in Alg. 1. It admits two types of training:

Algorithm 1 `NPG`: Sample-based NPG.

```
Input: Environment $E$; learning rate $\eta$; episode number $T$; batch size $N$; initialization $\theta_0$; sampler $\pi_s$; regularization coefficient $\lambda$; entropy clip bound $U$; optimization domain $\mathcal{G}$.
for $t \leftarrow 0, 1, \ldots, T-1$ do
  Initialize $\widehat{F}_t \leftarrow 0^{d\times d}, \widehat{\nabla}_t \leftarrow 0^d$.
  for $n \leftarrow 0, 1, \ldots, N-1$ do
    for $h \leftarrow 0, 1, \ldots, H-1$ do
      if $\pi_s$ is not None then
        $s_h, a_h, \widehat{A}_{H-h}(s_h, a_h) \leftarrow$ Sample $(E, \pi_s, \text{True}, \pi_t, h, \lambda, U)$ (see Alg. 3).
        // $s, a \sim \widetilde{d}^{\pi_s}_{m,h}$, estimate $A^{t,\lambda}_{m,H-h}(s, a)$.
      else
        $s_h, a_h, \widehat{A}_{H-h}(s_h, a_h) \leftarrow$ Sample $(E, \pi_t, \text{False}, \pi_t, h, \lambda, U)$.
        // $s, a \sim d^{\theta_t}_{m,h}$, estimate $A^{t,\lambda}_{m,H-h}(s, a)$.
      end if
    end for
    Update:
      $$\widehat{F}_t \leftarrow \widehat{F}_t + \sum_{h=0}^{H-1} \nabla_\theta \ln \pi_{\theta_t}(a_h|s_h) \left(\nabla_\theta \ln \pi_{\theta_t}(a_h|s_h)\right)^\top,$$
      $$\widehat{\nabla}_t \leftarrow \widehat{\nabla}_t + \sum_{h=0}^{H-1} \widehat{A}_{H-h}(s_h, a_h) \nabla_\theta \ln \pi_{\theta_t}(a_h|s_h).$$
  end for
  Call any solver to get $\widehat{g}_t \leftarrow \arg\min_{g\in\mathcal{G}} g^\top \widehat{F}_t g - 2g^\top \widehat{\nabla}_t$.
  Update $\theta_{t+1} \leftarrow \theta_t + \eta \widehat{g}_t$.
end for
Return: $\theta_T$.
```

- If $\pi_s =$ None, it calls Alg. 3 (a sampler for state-action pairs and estimator for the advantage function, deferred to App. A) to sample $s, a \sim d^{\theta_t}$;

- If $\pi_s \neq$ None, it then calls Alg. 3 to sample $s, a \sim \widetilde{d}^{\pi_s}$.

In both cases, we denote d^t as the sampling distribution and Σ_t as the induced Fisher Information Matrix used in step t, i.e. $d^t = d^{\theta_t}, \Sigma_t = F(\theta_t)$ if $\pi_s =$ None, and $d^t = \widetilde{d}^{\pi_s}, \Sigma_t = \Sigma^{\theta_t}_{\widetilde{d}^{\pi_s}}$ otherwise. The update rule can be written in a unified way as $g_t \approx \arg\min_{g\in\mathcal{G}} L(g; \theta_t, d^t)$. In our problem, this is equivalent to solving a quadratic optimization problem and we can use existing solvers.

Remark 3. Alg. 1 is different from Alg. 4 of Agarwal et al. [2020] that: we use a "batched" update while they used successive Projected Gradient Descents (PGD). This is an important implementation technique to speed up training in our experiments. Our implementation is provided in the supplementary materials.

5.2 Curriculum Learning

We use Curriculum Learning to facilitate the training process. Bengio et al. [2009] suggested a method of using gradually more difficult examples to speed training up. Kong et al. [2019] also set up a *series of curricula* (mentioned as "bootstrapping") when tackling CO problems: suppose the problem based on sequences with length n, they first trained on $n = 10$ and once the algorithm performed well they increased n by 10, until n reached the final value. Our analysis below shows there is no need to increase problem scales gradually, instead a single-step curriculum is sufficient. We also verify our simplification through experiments.

Algorithm 2 Curriculum learning framework.

```
Input: Environment $E$; learning rate $\eta$; episode number $T$; batch size $N$; sampler type $samp \in \{$ pi_s, pi_t $\}$; regularization coefficient $\lambda$; entropy clip bound $U$; optimization domain $\mathcal{G}$.
Construct an environment $E'$ with a task easier than $E$. This environment should have optimal policy similar to that of $E$.
$\theta_s \leftarrow$ NPG $(E', \eta, T, N, 0^d, \text{None}, \lambda, U, \mathcal{G})$ (see Alg. 1).
if $samp =$ pi_s then
    $\theta_T \leftarrow$ NPG $(E, \eta, T, N, 0^d, \pi_s, \lambda, U, \mathcal{G})$.
else
    $\theta_T \leftarrow$ NPG $(E, \eta, T, N, \theta_s, \text{None}, \lambda, U, \mathcal{G})$.
end if
Return: $\theta_T$.
```

The training framework is shown in Alg. 2. We first construct an easy environment and get the (near-)optimal policy π_s of it. Then, we either use π_s to sample data in the target environment, or simply continue training π_s on-policy.

6 Performance Analysis

6.1 Natural Policy Gradient for Latent MDP

Let $g_t^\star \in \arg\min_{g\in\mathcal{G}} L(g; \theta_t, d^t)$ denote the true minimizer. We have the following definitions.

Definition 6. *Define for* $0 \le t \le T$*:*

1. *(Excess risk)* $\epsilon_{\text{stat}} := \max_t \mathbb{E}[L(g_t; \theta_t, d^t) - L(g_t^\star; \theta_t, d^t)]$*;*
2. *(Transfer error)* $\epsilon_{\text{bias}} := \max_t \mathbb{E}[L(g_t^\star; \theta_t, d^\star)]$*;*
3. *(Relative condition number)*

$$\kappa := \max_t \mathbb{E}\left[\sup_{x\in\mathbb{R}^d} \frac{x^\top \Sigma_{d^\star}^{\theta_t} x}{x^\top \Sigma_t x}\right].$$

Note that term inside the expectation is a random quantity as θ_t *is random.*

In the above conditions, the expectation is with respect to the randomness in the sequence of weights $g_0, g_1, \ldots, g_T$.

All the quantities are commonly used in literature (Wang et al. [2020], Agarwal et al. [2020], Cayci et al. [2021], Liu et al. [2020]). ϵ_{stat} is due to that the minimizer g_t from samples may not be the minimizer of the population loss L. ϵ_{bias} quantifies the approximation error due to the use of feature maps. κ characterizes the distribution mismatch between d^t and $d^\star$. This is a key quantity in curriculum learning and we will study it in more details in the following sections.

Thm. 7 shows the convergence rate of Alg. 1, and its proof is deferred to Sec. B.1.

Theorem 7. *With notations in Def. 6, our algorithm enjoys the following performance bound*

$$\mathbb{E}\left[\min_{0\le t\le T} V^{\star,\lambda} - V^{t,\lambda}\right] \le \frac{\lambda(1-\eta\lambda)^{T+1}\Phi(\pi_0)}{1-(1-\eta\lambda)^{T+1}} + \eta\frac{B^2G^2}{2} + \sqrt{H\epsilon_{\text{bias}}} + \sqrt{H\kappa\epsilon_{\text{stat}}}.$$

Remark 4. ① This is the first result for LMDP and sample-based NPG with entropy regularization. The proof centers around the Lyapunov potential functions.

② For any fixed $\lambda > 0$ we have a linear convergence, which generally matches the result of discounted infinite horizon MDP (Thm. 1 in Cayci et al. [2021]); if we take $\lambda \to 0$, limit gives an $O(\frac{1}{\eta T} + \eta)$ bound (which implies an $O(1/\sqrt{T})$ rate), matching the result in Agarwal et al. [2020].

③ ϵ_{stat} can be reduced using a larger batch size N (Lem. 19). The typical scaling is $\epsilon_{\text{stat}} = O(1/\sqrt{N})$.

④ If some d_t (especially the initialization d_0) is far away from $d^\star$, κ may be extremely large (Sec. 6.2 as an example). This theorem shows, we only need find a policy whose κ is small. Therefore, if we can find such a policy with a single curriculum, we do not need the multi-step curriculum learning procedure used in Kong et al. [2019].

⑤ ϵ_{bias} is relevant to the feature mapping design. *Softmax tabular parameterization* leads to $\epsilon_{\text{bias}} = 0$. Another special case is a *linear MDP with low rank transitions*: if we use the features from the linear MDP then $\epsilon_{\text{bias}} = 0$ (Rem. 6.4 in Agarwal et al. [2020]). In such cases, κ would contribute to the *dominant term* of the constant gap.

6.2 Curriculum learning for Secretary Problem

For SP, dynamic programming shows that there exists a threshold policy that is optimal [Beckmann, 1990]. Suppose the threshold is $p \in (0,1)$, then under the formulation in Sec. 7.1, the policy is: accept if and only if $\frac{i}{n} > p$ and $x_i = 1$. For the classical SP where all the $n!$ instances have equal probability to be sampled, the optimal threshold is $\frac{1}{\mathrm{e}}$.

To show that curriculum learning makes the training converge faster, Thm. 7 gives a direct hint: *curriculum learning produces a good sampler leading to much smaller κ than that of a naïve random sampler.* Here we focus on the cases where *samp* = `pi_s`. We show Thm. 8 to characterize κ in SP. Its proof is deferred to Sec. B.2.

Theorem 8. *Assume that each candidate is independent from others and the i-th candidate has probability P_i of being the best so far (Sec. 7.1). Assume the optimal policy is a p-threshold policy and the sampling policy is a q-threshold policy. There exists a policy parameterization that: up to multiplicative factors in* $[1,2]$,

$$\kappa_{\text{curl}} = \begin{cases} \prod_{j=\lfloor nq\rfloor+1}^{\lfloor np\rfloor} \frac{1}{1-P_j}, & q \le p, \\ 1, & q > p, \end{cases} \tag{1}$$

$$\kappa_{\text{naïve}} = 2^{\lfloor np\rfloor} \max\left\{1, \max_{i \ge \lfloor np\rfloor+2} \prod_{j=\lfloor np\rfloor+1}^{i-1} 2(1-P_j)\right\}, \tag{2}$$

where κ_{curl} corresponds to κ induced by the q-threshold policy and $\kappa_{\text{naïve}}$ corresponds to κ induced by the naïve random sampling policy.

To understand how curriculum learning influences κ, below we apply Thm. 8 to three concrete cases.

6.2.1 The classical case: an exponential improvement

Here we look into the classical SP as a special case: all the $n!$ permutations are sampled with equal probability. The probability series for this case is $P_i = \frac{1}{i}$. Substituting them into Eq. 1 and Eq. 2 directly gives:

$$\kappa_{\text{curl}} = \begin{cases} \frac{\lfloor n/\mathrm{e}\rfloor}{\lfloor nq\rfloor}, & q \le \frac{1}{\mathrm{e}}, \\ 1, & q > \frac{1}{\mathrm{e}}, \end{cases} \qquad \kappa_{\text{naïve}} = 2^{n-1}\frac{\lfloor n/\mathrm{e}\rfloor}{n-1}.$$

Except for the corner case where $q < \frac{1}{n}$, we have that $\kappa_{\text{curl}} = O(n)$ while $\kappa_{\text{naïve}} = \Omega(2^n)$. Notice that any distribution with $P_i \le \frac{1}{i}$ leads to an exponential improvement.

6.2.2 The General Case

Now we try to loosen the condition where $P_i \le \frac{1}{i}$. Let us consider the case where $P_i \le \frac{1}{2}$ for $i \ge 2$ (by definition P_1 is always equal to 1). Eq. 1 and Eq. 2 now become:

$$\kappa_{\text{curl}} \le \begin{cases} 2^{\lfloor np\rfloor - \lfloor nq\rfloor}, & q \le p, \\ 1, & q > p, \end{cases}, \qquad \kappa_{\text{naïve}} \ge 2^{\lfloor np\rfloor}.$$

A direct implication is that $\kappa_{\text{curl}} \leq \kappa_{\text{naïve}}$ always holds, and sometime the difference is huge.

6.2.3 Failure Mode of Curriculum Learning

Lastly we show further relaxing the assumption on P_i leads to cases which are not in our favor. The extreme case is that all $P_i = 1$, i.e., the best candidate always comes as the last one. Suppose $q < \frac{n-1}{n}$, then $d^{\pi_q}\left(\frac{n}{n}\right) = 0$. Hence

$$\kappa_{\text{curl}} = \infty, \quad \kappa_{\text{naïve}} = 2^{n-1}.$$

From Eq. 2 it can be seen that $\kappa_{\text{naïve}} \leq 2^{n-1}$. Similar as in Sec. 3 of Beckmann [1990], the optimal threshold p satisfies:

$$\sum_{i=\lfloor np \rfloor+2}^{n} \frac{P_i}{1-P_i} \leq 1 < \sum_{i=\lfloor np \rfloor+1}^{n} \frac{P_i}{1-P_i}.$$

So letting $P_n > \frac{1}{2}$ results in $p \in [\frac{n-1}{n}, 1)$. Further, if $q < \frac{n-1}{n}$ and $P_j > 1 - 2^{-\frac{n}{n-\lfloor nq \rfloor - 1}}$ for any $\lfloor nq \rfloor + 1 \leq j \leq n-1$, then from Eq. 1, $\kappa_{\text{curl}} > 2^n > \kappa_{\text{naïve}}$.

Remark 5. These cases show that one can always adverserially manipulate the curriculum learning. Alternatively, sometimes there is hardly any reasonable curriculum.

7 Experiments

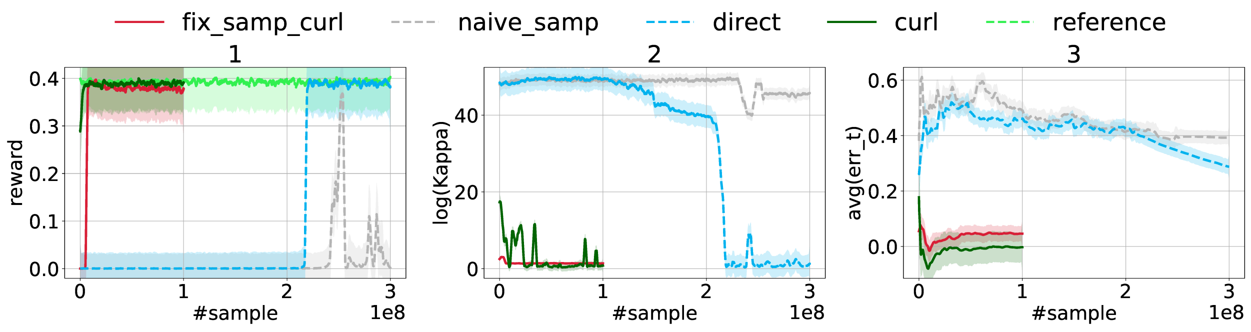

Figure 1: One experiment of SP. **Dashed lines represent only final phase training and solid lines represent curriculum learning.** The shadowed area shows the 95% confidence interval for the expectation. `fix_samp_curl` means running Alg. 2 with $samp = \texttt{pi\_s}$; `naive_samp` means running Alg. 1 with $\pi_s =$ naïve random policy; `direct` means running Alg. 1 with $\pi_s =$ None; `curl` means running Alg. 2 with $samp = \texttt{pi\_t}$. The reference policy is the optimal threshold policy.

The experiments' formulations are modified from Kong et al. [2019]. Since in all the experiments there are exactly two actions, we can use $\phi(s) = \phi(s, \text{accept}) - \phi(s, \text{reject})$ instead of $\phi(s, \text{accept})$ and $\phi(s, \text{reject})$. Now the policy is $\pi_\theta(\text{accept}|s) = \frac{\exp(\theta^\top \phi(s))}{\exp(\theta^\top \phi(s))+1}$ and $\pi_\theta(\text{reject}|s) = \frac{1}{\exp(\theta^\top \phi(s))+1}$. In curriculum training the entire training process splits into two phase. We call the training on curriculum (small scale instances) "warm-up phase" and the training on large scale instances "final phase". If the training is directly on large scale instances, we also call it "final phase" for convenience. As will be seen in the following, we ran more than one experiments for each problem. In one experiment there are more than one training processes to show the effect of different samplers and regularization coefficients. All the trainings in the same experiment have the common distributions over LMDPs for final phase and warm-up phase (if any) respectively.

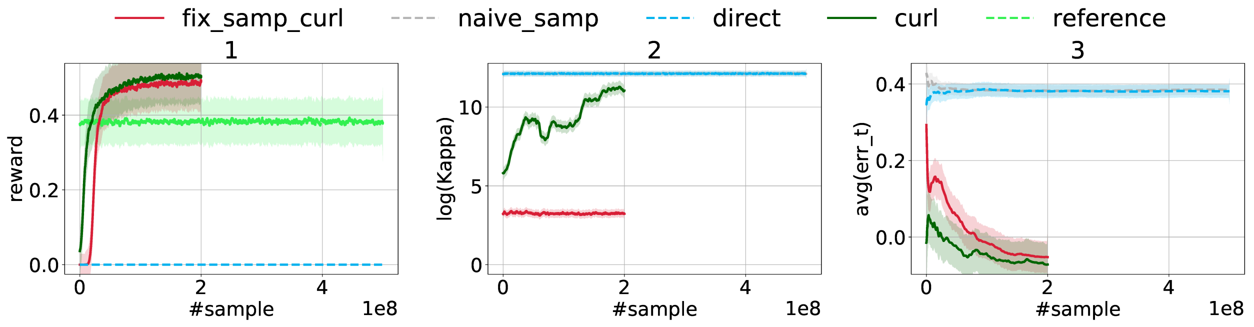

Figure 2: One experiment of OKD. Legend description is the same as that of Fig. 1. The reference policy is the bang-per-buck algorithm for Online Knapsack (Sec. 3.1 of Kong et al. [2019], we calculated the ratio with ternary search).

7.1 Secretary Problem

State and action spaces. States with $X_i > 1$ are the same. To make the problem "scale-invariant", we use $\frac{i}{n}$ to represent i. So the states are $(\frac{i}{n}, x_i = \mathbb{1}[X_i = 1])$. There is an additional terminal state $g = (0, 0)$. For each state, the agent can either accept or reject.
Transition and reward. Any action in g leads back to g. Once the agent accepts the i-th candidate, the state transits into g, and reward is 1 if i is the best in the instance. If the agent rejects, then the state goes to $(\frac{i+1}{n}, x_{i+1})$ if $i < n$ and g if $i = n$. For all other cases, rewards are 0.
Feature mapping. Recall that all states are of the form (f, x) where $f \in [0, 1],\ x \in \{0, 1\}$. We set a degree d_0 and the feature mapping is constructed as the collection of polynomial bases with degree less than d_0 $(d = 2d_0)$:

$$\phi(f, x) = (1, f, \ldots, f^{d_0-1}, x, fx, \ldots, f^{d_0-1}x).$$

LMDP distribution. We model the distribution as follows: for each i, we can have $x_i = 1$ with probability P_i and is independent from other i'. By definition, $P_1 = 1$ while other P_i can be arbitrary. The classical SP satisfies $P_i = \frac{1}{i}$. We also experimented on three other distributions (so in total there are four experiments), each with a series of numbers $p_2, p_3, \ldots, p_n \overset{\text{i.i.d.}}{\sim} \text{Unif}_{[0,1]}$ and set $P_i = \frac{1}{i^{2p_i+0.25}}$.

For each experiment, we run eight setups, each with different combinations of sampler policies, initialization policies of the final phase, and the value of regularization coefficient λ. For the warm-up phases we set $n = 10$ and for final phases $n = 100$.
Results. We show one of the four experiments in Fig. 1 (different optimal policy for warm-up and final phases), and the rest are deferred to Sec. C. A relevant quantity is err_t (defined in Lem. 15) which shows the true transfer error and is upper-bounded by $\sqrt{H\epsilon_{\text{bias}}} + \sqrt{H\kappa\epsilon_{\text{stat}}}$. Aside from reward and $\ln \kappa$, we plot the weighted average of err_t according to the proof of Thm. 7: $\text{avg}(\text{err}_t) = \frac{\sum_{i=0}^{t}(1-\eta\lambda)^{t-i}\ \text{err}_i}{\sum_{i=0}^{t}(1-\eta\lambda)^{t-i}}$. The experiments clearly demonstrate that curriculum learning can boost the performance by large margin and curriculum learning indeed dramatically reduces κ.

7.2 Online Knapsack (decision version)

State and action spaces. The states are represented as

$$\left(\frac{i}{n}, s_i, v_i, \frac{\sum_{j=1}^{i-1} x_j s_j}{B}, \frac{\sum_{j=1}^{i-1} x_j v_j}{V}\right),$$

where $x_j = \mathbb{1}[\text{item } j \text{ was successfully chosen}]$ for $1 \le j \le i - 1$ (in the instance). There is an additional terminal state $g = (0, 0, 0, 0, 0)$. For each state (including g for simplicity), the agent can either accept or reject.

Transition and reward. The transition is implied by the definition of the problem. Any action in terminal state g leads back to g. The item is successfully chosen if and only if the agent accepts and the budget is sufficient. A reward of 1 is given only the first time $\sum_{j=1}^{i} x_i v_i \geq V$, and then the state goes to g. For all other cases, reward is 0.
Feature mapping. Suppose the state is (f, s, v, r, q). We set a degree d_0 and the feature mapping is constructed as the collection of polynomial bases with degree less than d_0 ($d = d_0^5$): $\phi(f, s, v, r, q) = (f^{i_f} s^{i_s} v^{i_v} r^{i_r} q^{i_q})_{i_f, i_s, i_v, i_r, i_q}$ where $i_{\clubsuit} \in \{0, 1, \ldots, d_0 - 1\}$.
LMDP distribution. In Sec. 3.2 the values and sizes are sampled from F_v ans F_s. If F_v or F_s is not $\mathrm{Unif}_{[0,1]}$, we model the distribution as: first set a granularity $gran$ and take $gran$ numbers $p_1, p_2, \ldots, p_{gran} \overset{\text{i.i.d.}}{\sim} \mathrm{Unif}_{[0,1]}$. p_i represents the (unnormalized) probability that $x \in (\frac{i-1}{gran}, \frac{i}{gran})$. To sample, we take $i \sim \mathrm{Multinomial}(p_1, p_2, \ldots, p_{gran})$ and return $x \sim \frac{i-1+\mathrm{Unif}_{[0,1]}}{gran}$.

For each experiment, we ran four setups, each with different combinations of sampler policies and initialization policies of the final phase. For the warm-up phases $n = 10$ and for final phases we set $n = 100$ in all experiments, while B and V vary. In one experiment it satisfies that $\frac{B}{n}$ are close for warm-up and final, and $\frac{V}{B}$ increases from warm-up to final.
Results. We show one of the three experiments in Fig. 2 ($F_v = F_s = \mathrm{Unif}_{[0,1]}$), and the rest are deferred to Sec. C. $\ln\kappa$ and $\mathrm{avg}(\ \mathrm{err}_t)$ are with respect to the reference policy, a bang-per-buck algorithm, which is not the optimal policy. Thus, they are only for reference. The experiments again demonstrate the effectiveness of curriculum learning and curriculum learning indeed dramatically reduces κ.

8 Conclusion

We showed CO problems could be naturally formulated as LMDPs, and we analyzed the convergence rate of NPG for LMDPs. Our theory shows the main benefit of curriculum learning is finding a stronger sampling strategy. Our empirical results also corroborated our findings. Our work is the first attempt to systematically study techniques devoted to using RL to solve CO problems, which we believe is a fruitful direction worth further investigations.

References

Alekh Agarwal, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift, 2020.

Susanne Albers, Arindam Khan, and Leon Ladewig. Improved online algorithms for knapsack and gap in the random order model, 2020.

Shuang Ao, Tianyi Zhou, Guodong Long, Qinghua Lu, Liming Zhu, and Jing Jiang. CO-PILOT: COllaborative planning and reinforcement learning on sub-task curriculum. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=uz_2t6VZby.

M.J. Beckmann. Dynamic programming and the secretary problem. *Computers & Mathematics with Applications*, 19(11):25–28, 1990. ISSN 0898-1221. doi: https://doi.org/10.1016/0898-1221(90)90145-A. URL https://www.sciencedirect.com/science/article/pii/089812219090145A.

Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning, 2017.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. URL https://doi.org/10.1145/1553374.1553380.

Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon, 2020.

Jalaj Bhandari and Daniel Russo. On the linear convergence of policy gradient methods for finite mdps. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 2386–2394. PMLR, 13–15 Apr 2021. URL https://proceedings.mlr.press/v130/bhandari21a.html.

Nataly Brukhim, Elad Hazan, and Karan Singh. A boosting approach to reinforcement learning, 2021.

Quentin Cappart, Emmanuel Goutierre, David Bergman, and Louis-Martin Rousseau. Improving optimization bounds using machine learning: Decision diagrams meet deep reinforcement learning, 2019.

Quentin Cappart, Didier Chételat, Elias B. Khalil, Andrea Lodi, Christopher Morris, and Petar Veličković. Combinatorial optimization and reasoning with graph neural networks. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4348–4355. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/595. URL https://doi.org/10.24963/ijcai.2021/595. Survey Track.

Semih Cayci, Niao He, and R. Srikant. Linear convergence of entropy-regularized natural policy gradient with linear function approximation, 2021.

Shicong Cen, Chen Cheng, Yuxin Chen, Yuting Wei, and Yuejie Chi. Fast global convergence of natural policy gradient methods with entropy regularization, 2021.

Hanjun Dai, Elias B. Khalil, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs, 2018.

Yuhao Ding, Junzi Zhang, and Javad Lavaei. On the global convergence of momentum-based policy gradient, 2021.

Iddo Drori, Anant Kharkar, William R. Sickinger, Brandon Kates, Qiang Ma, Suwen Ge, Eden Dolev, Brenda L Dietrich, David P. Williamson, and Madeleine Udell. Learning to solve combinatorial optimization problems on real-world graphs in linear time. *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 19–24, 2020.

Alex Graves, Marc G. Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks, 2017.

Zhiyi Huang, Binghui Peng, Zhihao Gavin Tang, Runzhou Tao, Xiaowei Wu, and Yuhao Zhang. Tight competitive ratios of classic matching algorithms in the fully online model. *ArXiv*, abs/1810.07903, 2019.

Sham M Kakade. A natural policy gradient. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002. URL https://proceedings.neurips.cc/paper/2001/file/4b86abe48d358ecf194c56c69108433e-Paper.pdf.

Weiwei Kong, Christopher Liaw, Aranyak Mehta, and D. Sivakumar. A new dog learns old tricks: Rl finds classic optimization algorithms. In *ICLR*, 2019.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *ICLR*, 2019a.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems!, 2019b.

Jeongyeol Kwon, Yonathan Efroni, Constantine Caramanis, and Shie Mannor. Rl for latent mdps: Regret guarantees and a lower bound, 2021.

Yanli Liu, Kaiqing Zhang, Tamer Basar, and Wotao Yin. An improved analysis of (variance-reduced) policy gradient and natural policy gradient methods. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7624–7636. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/56577889b3c1cd083b6d7b32d32f99d5-Paper.pdf.

Nina Mazyavkina, Sergey Sviridov, Sergei Ivanov, and Evgeny Burnaev. Reinforcement learning for combinatorial optimization: A survey, 2020.

Jincheng Mei, Chenjun Xiao, Csaba Szepesvari, and Dale Schuurmans. On the global convergence rates of softmax policy gradient methods. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6820–6829. PMLR, 13–18 Jul 2020. URL http://proceedings.mlr.press/v119/mei20b.html.

Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang, Ebrahim M. Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nazi, Jiwoo Pak, Andy Tong, Kavya Srinivasa, Will Hang, Emre Tuncer, Quoc V. Le, James Laudon, Richard Ho, Roger Carpenter, and Jeff Dean. A graph placement methodology for fast chip design. *Nature*, 594 7862:207–212, 2021.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *J. Mach. Learn. Res.*, 21: 181:1–181:50, 2020.

M. Nazari, Afshin Oroojlooy, Lawrence V. Snyder, and Martin Takác. Reinforcement learning for solving the vehicle routing problem. In *NeurIPS*, 2018.

Rodrigo Pérez-Dattari, Carlos Celemin, Javier Ruiz del Solar, and Jens Kober. Interactive learning with corrective feedback for policies based on deep neural networks. In *ISER*, 2018.

Jan Scholten, Daan Wout, Carlos Celemin, and Jens Kober. Deep reinforcement learning with feedback-based exploration. *2019 IEEE 58th Conference on Decision and Control (CDC)*, Dec 2019. doi: 10.1109/cdc40024.2019.9029503. URL http://dx.doi.org/10.1109/CDC40024.2019.9029503.

Daniel Selsam, Matthew Lamm, Benedikt Bünz, Percy Liang, Leonardo de Moura, and David L. Dill. Learning a sat solver from single-bit supervision, 2019.

Lauren N. Steimle, David L. Kaufman, and Brian T. Denton. Multi-model markov decision processes. *IISE Transactions*, 53(10):1124–1139, 2021. doi: 10.1080/24725854.2021.1895454. URL https://doi.org/10.1080/24725854.2021.1895454.

Natalia Vesselinova, Rebecca Steinert, Daniel F. Perez-Ramirez, and Magnus Boman. Learning combinatorial optimization on graphs: A survey with applications to networking. *IEEE Access*, 8:120388–120416, 2020.

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. URL https://openreview.net/forum?id=BJgQfkSYDS.

Lucas Willems, Salem Lahlou, and Yoshua Bengio. Mastering rate based curriculum learning, 2020.

Junzi Zhang, Jongho Kim, Brendan O'Donoghue, and Stephen Boyd. Sample efficient reinforcement learning with reinforce. In *AAAI*, 2021.

Tianyi Zhou, Shengjie Wang, and Jeffrey Bilmes. Curriculum learning by dynamic instance hardness. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8602–8613. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/62000dee5a05a6a71de3a6127a68778a-Paper.pdf.

Tianyi Zhou, Shengjie Wang, and Jeff Bilmes. Curriculum learning by optimizing learning dynamics. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 433–441. PMLR, 13–15 Apr 2021a. URL https://proceedings.mlr.press/v130/zhou21a.html.

Tianyi Zhou, Shengjie Wang, and Jeff Bilmes. Robust curriculum learning: from clean label detection to noisy label self-correction. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=lmTWnm3coJJ.

Yanqi Zhou, Sudip Roy, Amirali Abdolrashidi, Daniel Wong, Peter Ma, Qiumin Xu, Hanxiao Liu, Phitchaya Mangpo Phothilimthana, Shen Wang, Anna Goldie, Azalia Mirhoseini, and James Laudon. Transferable graph optimizers for ml compilers, 2021c.

Appendix

A Skipped Algorithm

Algorithm 3 `Sample`: Sampler for $s \sim d_{m,h}^{\pi_{\text{samp}}}$ where $m \sim$ Multinomial $(w_1, \ldots, w_M)$, $a \sim \text{Unif}_{\mathcal{A}}$ if $unif =$ True and $a \sim \pi_{\text{samp}}(\cdot|s)$ otherwise, and estimate of $A_{m,H-h}^{t,\lambda}(s,a)$.

```
Input: Environment $E$; sampler policy $\pi_{\text{samp}}$; whether to sample uniform actions after state $unif$; current
policy $\pi_t$; time step $h$; regularization coefficient $\lambda$; entropy clip bound $U$.
$E$.reset().
for $i \leftarrow 0, 1, \ldots, h-1$ do
    $s_i \leftarrow E$.get_state().
    Sample action $a_i \sim \pi_{\text{samp}}(\cdot|s_i)$ and $E$.execute($a_i$).
end for
$s_h \leftarrow E$.get_state().
if $unif =$ True then
    $a_h \sim \text{Unif}_{\mathcal{A}}$.
else
    $a_h \sim \pi_{\text{samp}}(\cdot|s_h)$.
end if
$(s, a) \leftarrow (s_h, a_h)$.
Get a random number $p \sim \text{Unif}[0, 1]$.
if $p < \frac{1}{2}$ then
    Override $a_h \sim \pi_t(\cdot|s_h)$.
    Set importance weight $C \leftarrow -2$.
    $r_h \leftarrow E$.execute($a_h$).
    Initialize cumulative reward $R \leftarrow r_h + \lambda\mathcal{H}(\pi_t(\cdot|s_h))$.
else
    $C \leftarrow 2$.
    $r_h \leftarrow E$.execute($a_h$).
    $R \leftarrow r_h + \lambda \min\{\ln \frac{1}{\pi_t(a_h|s_h)}, U\}$.
end if
for $i \leftarrow h+1, h+2, \ldots, H-1$ do
    $s_i \leftarrow E$.get_state().
    $a_i \sim \pi_t(\cdot|s_i)$ and $r_h \leftarrow E$.execute($a_i$).
    $R \leftarrow R + r_i + \lambda\mathcal{H}(\pi_t(\cdot|s_i))$.
end for
Return: $s, a, \widehat{A}_{H-h}^{t,\lambda}(s,a) = CR$.
```

B Main Result

B.1 Natural Policy Gradient for Latent MDP

Theorem 7 (Restatement of Thm. 7)**.** *With notations in Def. 6,*

$$\mathbb{E}\left[\min_{0 \le t \le T} V^{\star,\lambda} - V^{t,\lambda}\right] \le \frac{\lambda(1-\eta\lambda)^{T+1}\Phi(\pi_0)}{1-(1-\eta\lambda)^{T+1}} + \eta\frac{B^2G^2}{2} + \sqrt{H\epsilon_{\text{bias}}} + \sqrt{H\kappa\epsilon_{\text{stat}}}.$$

Proof. Here we make shorthands for the sub-optimality gap and potential function: $\Delta_t := V^{\star,\lambda} - V^{t,\lambda}$ and $\Phi_t := \Phi(\pi_t)$. From Lem. 15 we have

$$\eta \Delta_t \leq (1-\eta\lambda)\Phi_t - \Phi_{t+1} + \eta \ \mathrm{err}_t + \eta^2 \frac{B^2G^2}{2}.$$

Taking expectation over the update weights, using Lem. 16 and Jensen's inequality, we have

$$\mathbb{E}[\eta\Delta_t] \leq (1-\eta\lambda)\,\mathbb{E}[\Phi_t] - \mathbb{E}[\Phi_{t+1}] + \eta C + \eta^2\frac{B^2G^2}{2},$$

where $C = \sqrt{H\epsilon_{\text{bias}}} + \sqrt{H\kappa\epsilon_{\text{stat}}}$. So

$$\begin{aligned}\mathbb{E}\left[\eta\sum_{t=0}^{T}(1-\eta\lambda)^{T-t}\Delta_t\right] &\leq \sum_{t=0}^{T}(1-\eta\lambda)^{T-t+1}\,\mathbb{E}[\Phi_t] - \sum_{t=0}^{T}(1-\eta\lambda)^{T-t}\,\mathbb{E}[\Phi_{t+1}] \\ &\quad + \left(\eta C + \eta^2\frac{B^2G^2}{2}\right)\sum_{t=0}^{T}(1-\eta\lambda)^{T-t} \\ &= (1-\eta\lambda)^{T+1}\Phi_0 - \mathbb{E}[\Phi_{T+1}] + \left(\eta C + \eta^2\frac{B^2G^2}{2}\right)\sum_{t=0}^{T}(1-\eta\lambda)^{T-t} \\ &\leq (1-\eta\lambda)^{T+1}\Phi_0 + \left(\eta C + \eta^2\frac{B^2G^2}{2}\right)\sum_{t=0}^{T}(1-\eta\lambda)^{T-t},\end{aligned}$$

where the last step uses the fact that $\Phi(\pi) \geq 0$. This is a weighted average, so by normalizing the coefficients,

$$\begin{aligned}\mathbb{E}\left[\min_{0\leq t\leq T}\Delta_t\right] &\leq \frac{(1-\eta\lambda)^{T+1}\Phi_0}{\eta\sum_{t=0}^{T}(1-\eta\lambda)^{T-t}} + \eta\frac{B^2G^2}{2} + C \\ &= \frac{\lambda(1-\eta\lambda)^{T+1}\Phi_0}{1-(1-\eta\lambda)^{T+1}} + \eta\frac{B^2G^2}{2} + C.\end{aligned}$$

This completes the proof. □

B.2 Curriculum learning and the constant gap for Secretary Problem

Theorem 8 (Formal statement of Thm. 8)**.** *For SP, set samp =* `pi_s` *in Alg. 2. Assume that each candidate is independent from others and the i-th candidate has probability P_i of being the best so far (Sec. 7.1). Assume the optimal policy is a p-threshold policy and the sampling policy is a q-threshold policy. There exists a policy parameterization and quantities*

$$k_{\text{curl}} = \begin{cases}\prod_{j=\lfloor nq\rfloor+1}^{\lfloor np\rfloor}\frac{1}{1-P_j}, & q\leq p,\\ 1, & q>p,\end{cases} \qquad k_{\text{naïve}} = 2^{\lfloor np\rfloor}\max\left\{1, \max_{i\geq\lfloor np\rfloor+2}\prod_{j=\lfloor np\rfloor+1}^{i-1}2(1-P_j)\right\},$$

such that $k_{\text{curl}} \leq \kappa_{\text{curl}} \leq 2k_{\text{curl}}$ and $k_{\text{naïve}} \leq \kappa_{\text{naïve}} \leq 2k_{\text{naïve}}$.

Proof. We need to calculate three state-action visitation distributions: that induced by the optimal policy, $d^\star$; that induced by the sampler which is the optimal for the curriculum, $\widetilde{d}^{\text{curl}}$; and that induced by the naïve random sampler, $\underset{\sim}{d}^{\text{naïve}}$. This then boils down to calculating the state(-action) visitation distribution under two types of policies: any threshold policy and the naïve random policy.

For any policy π, denote $d^\pi\left(\frac{i}{n}\right)$ as the probability for the agent acting under π to see states $\frac{i}{n}$ with arbitrary x_i. We do not need to take the terminal state g into consideration, since it stays in a zero-reward loop and contributes 0 to $L(g;\theta,d)$. We use the LMDP distribution described in Sec. 7.1.

Denote π_p as the p-threshold policy, i.e. accept if and only if $\frac{i}{n} > p$ and $x_i = 1$. Then

$$\begin{aligned}
d^{\pi_p}\left(\frac{i}{n}\right) &= \mathbb{P}(\text{reject all previous } i-1 \text{ states}|\pi_p) \\
&= \prod_{j=1}^{i-1}\left(\mathbb{P}\left(\frac{j}{n},1\right)\mathbb{1}\left[\frac{j}{n}\le p\right] + 1 - \mathbb{P}\left(\frac{j}{n},1\right)\right) \\
&= \prod_{j=\lfloor np\rfloor+1}^{i-1}\left(1-\mathbb{P}\left(\frac{j}{n},1\right)\right) \\
&= \prod_{j=\lfloor np\rfloor+1}^{i-1}(1-P_j).
\end{aligned}$$

Denote $\pi_{\text{naïve}}$ as the naïve random policy, i.e., accept with probability $\frac{1}{2}$ regardless of the state. Then

$$d^{\pi_{\text{naïve}}}\left(\frac{i}{n}\right) = \mathbb{P}(\text{reject all previous } i-1 \text{ states}|\pi_{\text{naïve}}) = \frac{1}{2^{i-1}}.$$

For any π, we can see that the state visitation distribution satisfies $d^\pi\left(\frac{i}{n},1\right) = P_i d^\pi\left(\frac{i}{n}\right)$ and $d^\pi\left(\frac{i}{n},0\right) = (1-P_i)d^\pi\left(\frac{i}{n}\right)$.

To show the possible largest difference, we use a parameterization that for each state s, $\phi(s) = \text{One-hot}(s)$. The policy is then

$$\pi_\theta(\text{accept}|s) = \frac{\exp(\theta^\top\phi(s))}{\exp(\theta^\top\phi(s))+1}, \quad \pi_\theta(\text{reject}|s) = \frac{1}{\exp(\theta^\top\phi(s))+1}.$$

Denote $\pi_\theta(s) = \pi_\theta(\text{accept}|s)$, we have

$$\nabla_\theta \ln\pi_\theta(\text{accept}|s) = (1-\pi_\theta(s))\phi(s), \quad \nabla_\theta \ln\pi_\theta(\text{reject}|s) = -\pi_\theta(s)\phi(s).$$

Now suppose the optimal threshold and the threshold learned through curriculum are p and q, then

$$\begin{aligned}
\Sigma^\theta_{d^\star} &= \sum_{s\in\mathcal{S}} d^{\pi_p}(s)\left(\pi_p(s)(1-\pi_\theta(s))^2 + (1-\pi_p(s))\pi_\theta(s)^2\right)\phi(s)\phi(s)^\top, \\
\Sigma^\theta_{\widetilde{d}^{\text{curl}}} &= \sum_{s\in\mathcal{S}} d^{\pi_q}(s)\left(\frac{1}{2}(1-\pi_\theta(s))^2 + \frac{1}{2}\pi_\theta(s)^2\right)\phi(s)\phi(s)^\top, \\
\Sigma^\theta_{\widetilde{d}^{\text{naïve}}} &= \sum_{s\in\mathcal{S}} d^{\text{naïve}}(s)\left(\frac{1}{2}(1-\pi_\theta(s))^2 + \frac{1}{2}\pi_\theta(s)^2\right)\phi(s)\phi(s)^\top.
\end{aligned}$$

Denote $\kappa_\clubsuit(\theta) = \sup_{x\in\mathbb{R}^d}\frac{x^\top\Sigma^\theta_{d^\star}x}{x^\top\Sigma^\theta_{\widetilde{d}^\clubsuit}x}$. From parameterization we know all $\phi(s)$ are orthogonal. Abusing π_q with π_{curl}, we have

$$\kappa_\clubsuit(\theta) = \max_{s\in\mathcal{S}}\frac{d^{\pi_p}(s)\left(\pi^\star(s)(1-\pi_\theta(s))^2 + (1-\pi^\star(s))\pi_\theta(s)^2\right)}{d^\clubsuit(s)\left(\frac{1}{2}(1-\pi_\theta(s))^2 + \frac{1}{2}\pi_\theta(s)^2\right)}.$$

We can separately consider each $s\in\mathcal{S}$ because of the orthogonal features. Observe that $\pi_p(s)\in\{0,1\}$, so for $s\in\mathcal{S}$, its corresponding term in $\kappa_\clubsuit(\theta)$ is maximized when $\pi_\theta(s) = 1-\pi_p(s)$ and is equal to $2\frac{d^{\pi_p}(s)}{d^\clubsuit(s)}$. By definition, $\kappa_\clubsuit = \max_{0\le t\le T}\mathbb{E}[\kappa_\clubsuit(\theta_t)]$. Since $\theta_0 = 0^d$, we have $\kappa_\clubsuit \ge \kappa_\clubsuit(0^d)$ where $\pi_\theta(s) = \frac{1}{2}$ and the corresponding term is $\frac{d^{\pi_p}(s)}{d^\clubsuit(s)}$. So

$$\max_{s\in\mathcal{S}}\frac{d^{\pi_p}(s)}{d^\clubsuit(s)} \le \kappa_\clubsuit \le 2\max_{s\in\mathcal{S}}\frac{d^{\pi_p}(s)}{d^\clubsuit(s)}.$$

We now have an order-accurate result $k_{\clubsuit} = \max_{s \in \mathcal{S}} \frac{d^{\pi_p}(s)}{d^{\clubsuit}(s)}$ for $\kappa_{\clubsuit}$. Direct computation gives

$$k_{\text{curl}} = \begin{cases} \prod_{j=\lfloor nq \rfloor+1}^{\lfloor np \rfloor} \frac{1}{1-P_j}, & q \leq p, \\ 1, & q > p, \end{cases}$$

$$k_{\text{naïve}} = 2^{\lfloor np \rfloor} \max \left\{ 1, \max_{i \geq \lfloor np \rfloor + 2} \prod_{j=\lfloor np \rfloor+1}^{i-1} 2(1-P_j) \right\}.$$

This completes the proof. $\square$

C Full Experiments

Here are all the experiments not shown in Sec. 7. For legend description please refer to the caption of Fig. 1. For experiment data (code, checkpoints, logging data and policy visualization) please refer to the supplementary files.

C.1 Secretary Problem

Fig. 3 (with its full view Fig. 4), Fig. 5, Fig. 6, along with Fig. 1 (with seed 2018011309) show four experiments of SP. They shared a learning rate of 0.2, batch size of 100 per step in horizon, final $n = 100$ and warm-up $n = 10$ (if applied curriculum learning). [2]

The experiment in Fig. 3 was done in the classical SP environment, i.e., all permutations have probability $\frac{1}{n!}$ to be sampled. Experiments Fig. 1, Fig. 5 and Fig. 6 were done with other distributions (see LMDP distribution of Sec. 7.1): the only differences are the random seeds, which we fixed and used to generate P_is for reproducibility.

The experiment of classical SP was run until the direct training of $n = 100$ converges, while all other experiments were run to a maximum episode of 30000 (hence sample number of $THb = 30000 \times 100 \times 100 = 3 \times 10^8$).

The optimal policy was derived from dynamic programming.

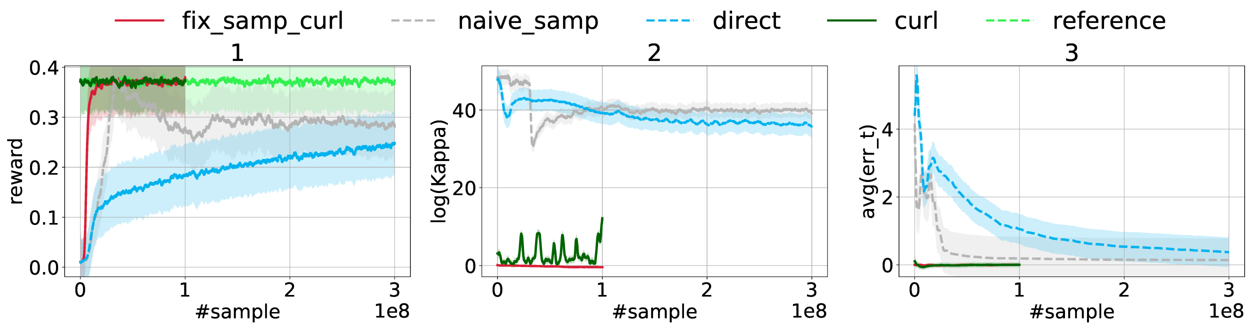

Figure 3: Classical SP, truncated to 3×10^8 samples.

[2]All the four trainings shown in the figures have their counterparts with regularization ($\lambda = 0.01$). Check the supplementary files and use TensorBoard for visualization.

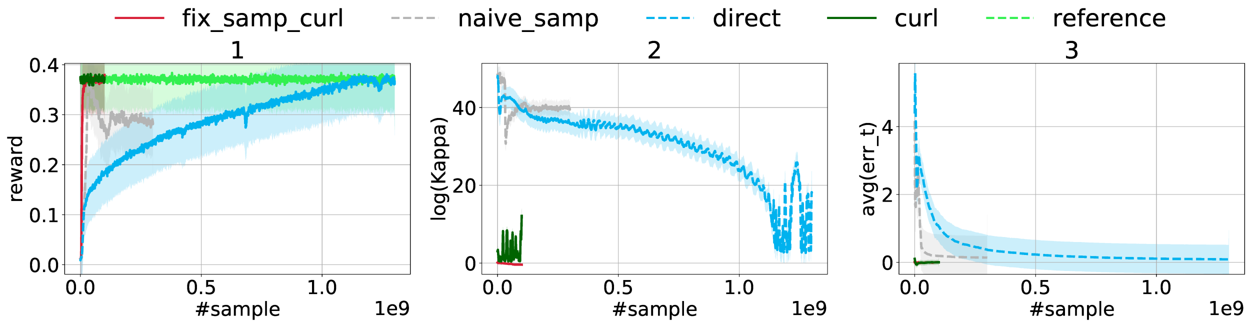

Figure 4: Classical SP, full view.

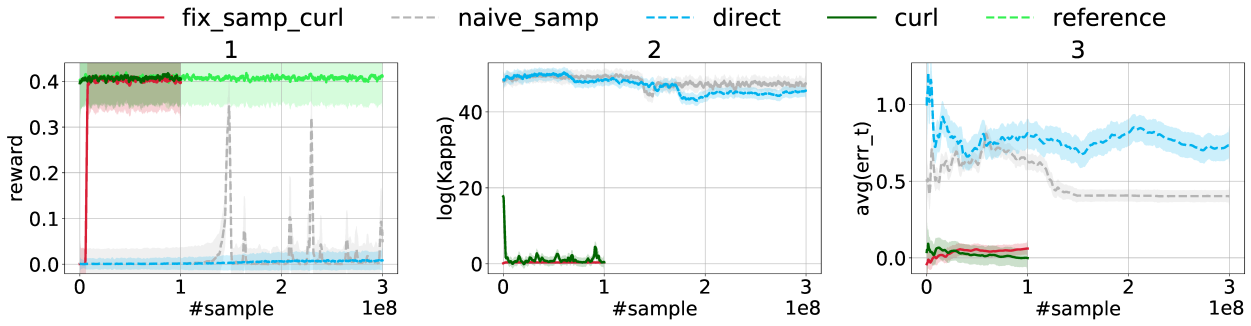

Figure 5: SP, with seed 20000308.

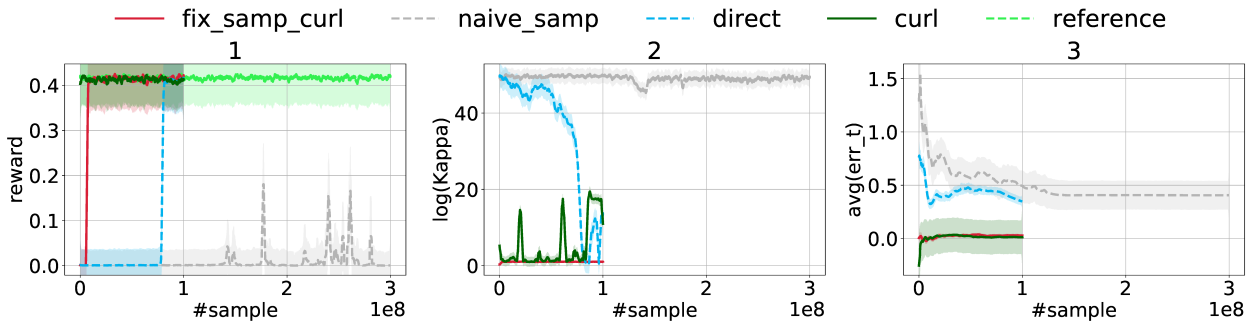

Figure 6: SP, with seed 19283746.

C.2 Online Knapsack (decision version)

Fig. 7, Fig. 8, along with Fig. 2 (with $F_v = F_s = \text{Unif}_{[0,1]}$) show three experiments of OKD. They shared a learning rate of 0.1, batch size of 100 per step in horizon, final $n = 100$ and warm-up $n = 10$ (if applied curriculum learning).

Experiments Fig. 7 and Fig. 8 were done with other value and size distributions (see LMDP distribution of Sec. 7.2): the only differences are the random seeds, which we fixed and used to generate F_v and F_s for reproducibility.

All experiments were run to a maximum episode of 50000 (hence sample number of $THb = 50000 \times 100 \times 100 = 5 \times 10^8$).

The reference policy is a bang-per-buck algorithm (Sec. 3.1 of Kong et al. [2019]): given a threshold r, accept i-th item if $\frac{v_i}{s_i} \geq r$. We searched for the optimal r with respect to Online Knapsack because we found that in general the reward is unimodal to r and contains no "plain area", so we can easily apply ternary search (the reward of OKD contains "plain area").

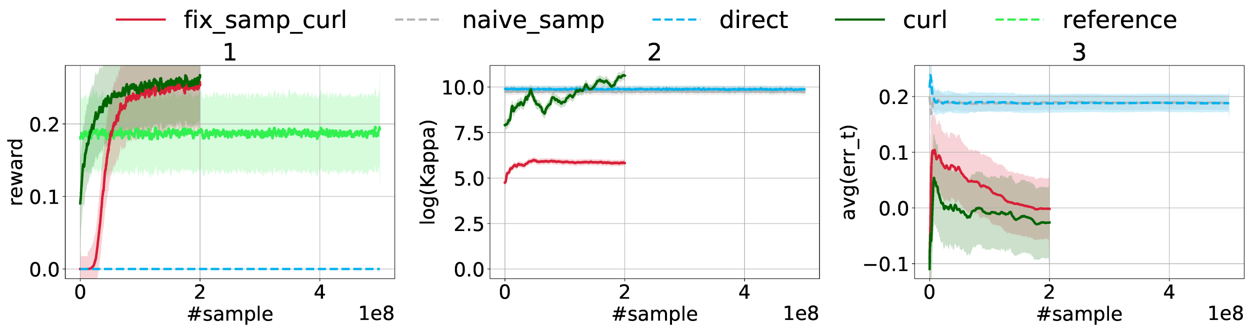

Figure 7: OKD, with seed 2018011309.

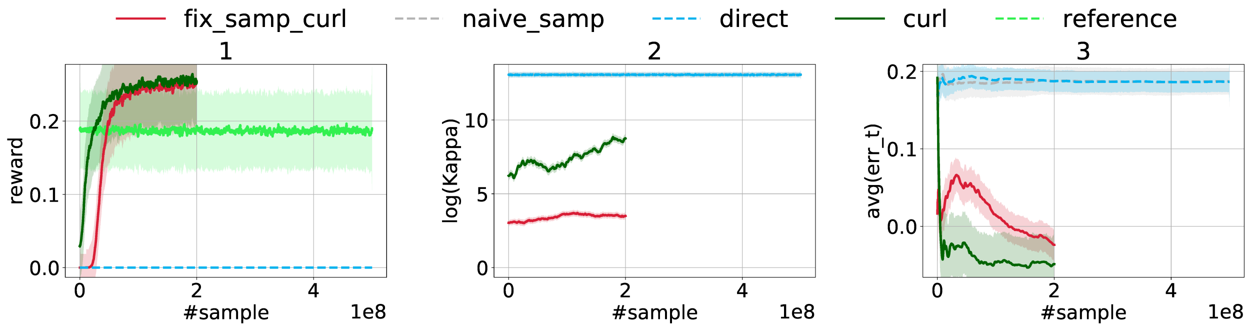

Figure 8: OKD, with seed 20000308.

D Lemmas

Theorem 5 (Full statement of Policy Gradient Theorem). *For any policy π_θ parameterized by θ, and any $1 \leq m \leq M$,*

$$\nabla_\theta V_{m,H}^{\pi_\theta,\lambda}(s_0) = \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d_{m,H-h}^{\theta}} \left[Q_{m,h}^{\pi_\theta,\lambda}(s,a) \nabla_\theta \ln \pi_\theta(a|s) \right].$$

As a result,

$$\nabla_\theta V^{\pi_\theta,\lambda} = \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d_{m,H-h}^{\theta}} \left[Q_{m,h}^{\pi_\theta,\lambda}(s,a) \nabla_\theta \ln \pi_\theta(a|s) \right].$$

Proof. For any $1 \le h \le H$ and $s \in \mathcal{S}$, since $V^{\pi_\theta,\lambda}_{m,h}(s) = \sum_{a\in\mathcal{A}} \pi_\theta(a|s) Q^{\pi_\theta,\lambda}_{m,h}(s,a)$, we have

$$\nabla_\theta V^{\pi_\theta,\lambda}_{m,h}(s) = \sum_{a\in\mathcal{A}} \left(Q^{\pi_\theta,\lambda}_{m,h}(s,a) \nabla_\theta \pi_\theta(a|s) + \pi_\theta(a|s) \nabla_\theta Q^{\pi_\theta,\lambda}_{m,h}(s,a) \right).$$

Hence

$$\begin{aligned}
\sum_{h=1}^{H} \sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \nabla_\theta V^{\pi_\theta,\lambda}_{m,h}(s) &= \sum_{h=1}^{H} \sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \sum_{a\in\mathcal{A}} \left(Q^{\pi_\theta,\lambda}_{m,h}(s,a) \nabla_\theta \pi_\theta(a|s) + \pi_\theta(a|s) \nabla_\theta Q^{\pi_\theta,\lambda}_{m,h}(s,a) \right) \\
&= \sum_{h=1}^{H} \sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s) Q^{\pi_\theta,\lambda}_{m,h}(s,a) \nabla_\theta \ln \pi_\theta(a|s) \\
&\quad + \sum_{h=1}^{H} \sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s) \nabla_\theta Q^{\pi_\theta,\lambda}_{m,h}(s,a) \\
&= \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^\theta_{m,H-h}} \left[Q^{\pi_\theta,\lambda}_{m,h}(s,a) \nabla_\theta \ln \pi_\theta(a|s) \right] \\
&\quad + \sum_{h=1}^{H} \sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s) \nabla_\theta Q^{\pi_\theta,\lambda}_{m,h}(s,a).
\end{aligned}$$

Next we focus on the second term. From the Bellman equation,

$$\begin{aligned}
\nabla_\theta Q^{\pi_\theta,\lambda}_{m,h}(s,a) &= \nabla_\theta \left(r_\theta(s,a) - \lambda \ln \pi_\theta(a|s) + \sum_{s'\in\mathcal{S}} P(s'|s,a) V^{\pi_\theta,\lambda}_{m,h-1}(s') \right) \\
&= -\lambda \nabla_\theta \ln \pi_\theta(a|s) + \sum_{s'\in\mathcal{S}} P(s'|s,a) \nabla_\theta V^{\pi_\theta,\lambda}_{m,h-1}(s').
\end{aligned}$$

Particularly, $\nabla_\theta Q^{\pi,\lambda}_{i,1}(s,a) = -\lambda \nabla_\theta \ln \pi_\theta(a|s)$. So

$$\begin{aligned}
&\sum_{h=1}^{H} \sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s) \nabla_\theta Q^{\pi_\theta,\lambda}_{m,h}(s,a) \\
&= \sum_{h=1}^{H} \sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s) \left(-\lambda \nabla_\theta \ln \pi_\theta(a|s) + \sum_{s'\in\mathcal{S}} P(s'|s,a) \nabla_\theta V^{\pi_\theta,\lambda}_{m,h-1}(s') \right) \\
&= -\lambda \sum_{h=1}^{H} \sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \underbrace{\sum_{a\in\mathcal{A}} \nabla_\theta \pi_\theta(a|s)}_{=\mathbf{0}} + \sum_{h=2}^{H} \sum_{s'\in\mathcal{S}} \nabla_\theta V^{\pi_\theta,\lambda}_{m,h-1}(s') \underbrace{\sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s) P(s'|s,a)}_{=d^\theta_{m,H-h+1}(s')} \\
&= \sum_{h=2}^{H} \sum_{s'\in\mathcal{S}} d^\theta_{m,H-h+1}(s') \nabla_\theta V^{\pi_\theta,\lambda}_{m,h-1}(s') \\
&= \sum_{h=1}^{H} \sum_{s\in\mathcal{S}} d^\theta_{m,H-h}(s) \nabla_\theta V^{\pi_\theta,\lambda}_{m,h}(s) - \nabla_\theta V^{\pi_\theta,\lambda}_{m,H}(s_0),
\end{aligned}$$

where we used the definition of d. So by rearranging the terms, we complete the proof. $\square$

Lemma 12. *Suppose $\Gamma \in \mathbb{R}^{n\times m}, D = \operatorname{diag}(d_1, d_2, \ldots, d_m) \in \mathbb{R}^{m\times m}$ where $d_i \ge 0$ and $q \in \mathbb{R}^m$, then $x = (\Gamma D \Gamma^\top)^\dagger \Gamma D q$ is a solution to the equation $\Gamma D \Gamma^\top x = \Gamma D q$.*

Proof. Denote $D^{1/2} = \mathrm{diag}(\sqrt{d_1}, \sqrt{d_2}, \ldots, \sqrt{d_m}), P = \Gamma D^{1/2}, p = D^{1/2}q$, then the equation is reduced to $PP^\top x = Pp$. Suppose the singular value decomposition of P is $U\Sigma V^\top$ where $U \in \mathbb{R}^{n\times n}, \Sigma \in \mathbb{R}^{n\times m}, V \in \mathbb{R}^{m\times m}$ where U and V are unitary, and singular values are $\sigma_1, \sigma_2, \ldots, \sigma_k$. So $PP^\top = U(\Sigma\Sigma^\top)U^\top$ and $(PP^\top)^\dagger = U(\Sigma\Sigma^\top)^\dagger U^\top$. Notice that

$$\Sigma\Sigma^\top = \mathrm{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_k^2, 0, \ldots, 0) \in \mathbb{R}^{n\times n},$$

we can then derive the pseudo-inverse of this particular diagonal matrix as

$$(\Sigma\Sigma^\top)^\dagger = \mathrm{diag}(\sigma_1^{-2}, \sigma_2^{-2}, \ldots, \sigma_k^{-2}, 0, \ldots, 0).$$

It is then easy to verify that $(\Sigma\Sigma^\top)(\Sigma\Sigma^\top)^\dagger\Sigma = \Sigma$. Finally,

$$\begin{aligned} PP^\top x &= (PP^\top)[(PP^\top)^\dagger Pp] \\ &= U(\Sigma\Sigma^\top)U^\top U(\Sigma\Sigma^\top)^\dagger U^\top U\Sigma V^\top p \\ &= U(\Sigma\Sigma^\top)(\Sigma\Sigma^\top)^\dagger \Sigma V^\top p \\ &= U\Sigma V^\top p \\ &= Pp. \end{aligned}$$

This completes the proof. □

Lemma 13 (NPG Update Rule)**.** *The update rule $\theta \leftarrow \theta + \eta F(\theta)^\dagger \nabla_\theta V^{\pi_\theta,\lambda}$ where*

$$F(\theta) = \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^\theta_{m,H-h}} \left[\nabla_\theta \ln \pi_\theta(a|s) \left(\nabla_\theta \ln \pi_\theta(a|s)\right)^\top\right]$$

is equivalent to $\theta \leftarrow \theta + \eta g^\star$, where $g^\star$ is a minimizer of the function

$$L(g) = \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^\theta_{m,H-h}} \left[\left(A^{\pi_\theta,\lambda}_{m,h}(s,a) - g^\top \nabla_\theta \ln \pi_\theta(a|s)\right)^2\right].$$

Proof.

$$\nabla_g L(g) = -2\sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^\theta_{m,H-h}} \left[\left(A^{\pi_\theta,\lambda}_{m,h}(s,a) - g^\top \nabla_\theta \ln \pi_\theta(a|s)\right) \nabla_\theta \ln \pi_\theta(a|s)\right].$$

Suppose $g^\star$ is any minimizer of $L(g)$, we have $\nabla_g L(g^\star) = \mathbf{0}$, hence

$$\begin{aligned} &\sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^\theta_{m,H-h}} \left[\left(g^{\star\top} \nabla_\theta \ln \pi_\theta(a|s)\right) \nabla_\theta \ln \pi_\theta(a|s)\right] \\ &= \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^\theta_{m,H-h}} \left[A^{\pi_\theta,\lambda}_{m,h}(s,a) \nabla_\theta \ln \pi_\theta(a|s)\right] \\ &= \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^\theta_{m,H-h}} \left[Q^{\pi_\theta,\lambda}_{m,h}(s,a) \nabla_\theta \ln \pi_\theta(a|s)\right]. \end{aligned}$$

Since $(u^\top v)v = (vv^\top)u$, then

$$F(\theta)g^\star = \nabla_\theta V^{\pi_\theta,\lambda}.$$

Now we assign $1, 2, \ldots, MHSA$ as indices to all $(m, h, s, a) \in \{1, \ldots, M\} \times \{1, \ldots, H\} \times \mathcal{S} \times \mathcal{A}$, and set

$$\begin{aligned}
\gamma_j &= \nabla_\theta \ln \pi_\theta(a|s), \\
d_j &= w_m d^\theta_{m,H-h}(s, a), \\
q_j &= Q^{\pi_\theta,\lambda}_{m,h}(s, a),
\end{aligned}$$

where j is the index assigned to (m, h, s, a). Then $F(\theta) = \Phi D \Phi^\top$ and $\nabla_\theta V^\theta = \Phi D q$ where

$$\begin{aligned}
\Gamma &= [\gamma_1, \gamma_2, \ldots, \gamma_{MHSA}] \in \mathbb{R}^{d \times MHSA}, \\
D &= \operatorname{diag}(d_1, d_2, \ldots, d_{MHSA}) \in \mathbb{R}^{MHSA \times MHSA}, \\
q &= [q_1, q_2, \ldots, q_{MHSA}]^\top \in \mathbb{R}^{MHSA}.
\end{aligned}$$

We now conclude the proof by utilizing Lem. 12. □

Lemma 14 (Performance Difference Lemma). *For any two policies π_1 and π_2, and any $1 \le m \le M$,*

$$V^{\pi_1,\lambda}_{m,H}(s_0) - V^{\pi_2,\lambda}_{m,H}(s_0) = \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a \sim d^{\pi_1}_{m,H-h}} \left[A^{\pi_2,\lambda}_{m,h}(s, a) + \lambda \ln \frac{\pi_2(a|s)}{\pi_1(a|s)} \right].$$

As a result,

$$V^{\pi_1,\lambda} - V^{\pi_2,\lambda} = \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a \sim d^{\pi_1}_{m,H-h}} \left[A^{\pi_2,\lambda}_{m,h}(s, a) + \lambda \ln \frac{\pi_2(a|s)}{\pi_1(a|s)} \right].$$

Proof. By definition of the value function, we have

$$\begin{aligned}
&V^{\pi_1,\lambda}_{m,H}(s_0) - V^{\pi_2,\lambda}_{m,H}(s_0) \\
&= \mathbb{E}\left[\sum_{h=0}^{H-1} r_m(s_h, a_h) - \lambda \ln \pi_1(a_h|s_h) \;\middle|\; \mathcal{M}_m, \pi_1, s_0 \right] - V^{\pi_2,\lambda}_{m,H}(s_0) \\
&= \mathbb{E}\left[\sum_{h=0}^{H-1} r_m(s_h, a_h) - \lambda \ln \pi_1(a_h|s_h) + V^{\pi_2,\lambda}_{m,H+1-h}(s_{h+1}) - V^{\pi_2,\lambda}_{m,H-h}(s_h) \;\middle|\; \mathcal{M}_m, \pi_1, s_0 \right] \\
&= \mathbb{E}\left[\sum_{h=0}^{H-1} \mathbb{E}\left[r_m(s_h, a_h) - \lambda \ln \pi_2(a_h|s_h) + V^{\pi_2,\lambda}_{m,H+1-h}(s_{h+1}) \;\middle|\; \mathcal{M}_m, \pi_2, s_h, a_h \right] \;\middle|\; \mathcal{M}_m, \pi_1, s_0 \right] \\
&\quad + \mathbb{E}\left[\sum_{h=0}^{H-1} -V^{\pi_2,\lambda}_{m,H-h}(s_h) + \lambda \ln \frac{\pi_2(a_h|s_h)}{\pi_1(a_h|s_h)} \;\middle|\; \mathcal{M}_m, \pi_1, s_0 \right],
\end{aligned}$$

where the last step uses law of iterated expectations. Since

$$\mathbb{E}\left[r_m(s_h, a_h) - \lambda \ln \pi_2(a_h|s_h) + V^{\pi_2,\lambda}_{m,H+1-h}(s_{h+1}) \;\middle|\; \mathcal{M}_m, \pi_2, s_h, a_h \right] = Q^{\pi_2,\lambda}_{m,H-h}(s_h, a_h),$$

finally we have

$$\begin{aligned}
V^{\pi_1,\lambda}_{m,H}(s_0) - V^{\pi_2,\lambda}_{m,H}(s_0) &= \mathbb{E}\left[\sum_{h=0}^{H-1} Q^{\pi_2,\lambda}_{m,H-h}(s_h, a_h) - V^{\pi_2,\lambda}_{m,H-h}(s_h) + \lambda \ln \frac{\pi_2(a_h|s_h)}{\pi_1(a_h|s_h)} \;\middle|\; \mathcal{M}_m, \pi_1, s_0 \right] \\
&= \mathbb{E}\left[\sum_{h=0}^{H-1} A^{\pi_2,\lambda}_{m,H-h}(s_h, a_h) + \lambda \ln \frac{\pi_2(a_h|s_h)}{\pi_1(a_h|s_h)} \;\middle|\; \mathcal{M}_m, \pi_1, s_0 \right] \\
&= \sum_{h=0}^{H-1} \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} d^{\pi_1}_{m,h}(s, a) \left(A^{\pi_2,\lambda}_{m,H-h}(s, a) + \lambda \ln \frac{\pi_2(a|s)}{\pi_1(a|s)} \right).
\end{aligned}$$

The proof is completed by reversing the order of h. □

Lemma 15 (Lyapunov Drift). *Suppose the update rule is $\theta_{t+1} = \theta_t + \eta g_t$, and define*

$$\mathrm{err}_t := \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,H-h}} \left[A^{t,\lambda}_{m,h}(s,a) - g_t^\top \nabla_\theta \ln \pi_t(a|s) \right].$$

We have that:

$$\Phi(\pi_{t+1}) - \Phi(\pi_t) \le -\eta\lambda\Phi(\pi_t) + \eta\ \mathrm{err}_t - \eta\left(V^{\star,\lambda} - V^{t,\lambda}\right) + \frac{\eta^2 B^2 \|g_t\|_2^2}{2}.$$

Proof. Denote $\Phi_t := \Phi(\pi_t)$. This proof follows a similar manner as in that of Lem. 6 in Cayci et al. [2021]. By smoothness (see Rem. 6.7 in Agarwal et al. [2020]),

$$\begin{aligned}
\ln \frac{\pi_t(a|s)}{\pi_{t+1}(a|s)} &\le (\theta_t - \theta_{t+1})^\top \nabla_\theta \ln \pi_t(a|s) + \frac{B^2}{2}\|\theta_{t+1} - \theta_t\|_2^2 \\
&= -\eta g_t^\top \nabla_\theta \ln \pi_t(a|s) + \frac{\eta^2 B^2 \|g_t\|_2^2}{2}.
\end{aligned}$$

By the definition of Φ,

$$\begin{aligned}
\Phi_{t+1} - \Phi_t &= \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,H-h}} \left[\ln \frac{\pi_t(a|s)}{\pi_{t+1}(a|s)} \right] \\
&\le -\eta \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,H-h}} \left[g_t^\top \nabla_\theta \ln \pi_t(a|s) \right] + \frac{\eta^2 B^2 \|g_t\|_2^2}{2}.
\end{aligned}$$

By the definition of err_t, Lem. 14 and again the definition of Φ, we finally have

$$\begin{aligned}
\Phi_{t+1} - \Phi_t \le\ & \eta \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,H-h}} \left[A^{t,\lambda}_{m,h}(s,a) - g_t^\top \nabla_\theta \ln \pi_t(a|s) \right] \\
& - \eta \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,H-h}} \left[A^{t,\lambda}_{m,h}(s,a) + \lambda \ln \frac{\pi_t(a|s)}{\pi^\star(a|s)} \right] \\
& - \eta\lambda \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,H-h}} \left[\ln \frac{\pi^\star(a|s)}{\pi_t(a|s)} \right] + \frac{\eta^2 B^2 \|g_t\|_2^2}{2} \\
=\ & \eta\ \mathrm{err}_t - \eta\left(V^{\star,\lambda} - V^{t,\lambda}\right) - \eta\lambda\Phi_t + \frac{\eta^2 B^2 \|g_t\|_2^2}{2},
\end{aligned}$$

which completes the proof. □

Lemma 16. *Recall that $g_t^\star$ is the true minimizer of $L(g;\theta_t,d^t)$ in domain $\mathcal{G}$.* err_t *defined in Lem. 15 satisfies*

$$\mathrm{err}_t \le \sqrt{HL(g_t^\star;\theta_t,d^\star)} + \sqrt{H\kappa(L(g_t;\theta_t,d^t) - L(g_t^\star;\theta_t,d^t))}.$$

Proof. The proof is similar to that of Thm. 6.1 in Agarwal et al. [2020]. We make the following decomposition of err_t:

$$\mathrm{err}_t = \underbrace{\sum_{m=1}^{M} w_m \sum_{h=0}^{H-1} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,h}} \left[A^{t,\lambda}_{m,h}(s,a) - g_t^{\star\top} \nabla_\theta \ln \pi_t(a|s) \right]}_{①} + \underbrace{\sum_{m=1}^{M} w_m \sum_{h=0}^{H-1} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,h}} \left[(g_t^\star - g_t)^\top \nabla_\theta \ln \pi_t(a|s) \right]}_{②}.$$

Since $\sum_{m=1}^{M} w_m \sum_{h=0}^{H-1} \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} d^\star_{m,h}(s,a) = H$, normalize the coefficients and apply Jensen's inequality, then

$$\begin{aligned}
① &\le \sqrt{\sum_{m=1}^{M} w_m \sum_{h=0}^{H-1} \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} d^\star_{m,h}(s,a)} \cdot \sqrt{\sum_{m=1}^{M} w_m \sum_{h=0}^{H-1} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,h}} \left[\left(A^{t,\lambda}_{m,h}(s,a) - g_t^{\star\top}\nabla_\theta \ln \pi_t(a|s)\right)^2\right]} \\
&= \sqrt{HL(g_t^\star;\theta_t,d^\star)}.
\end{aligned}$$

Similarly[3],

$$\begin{aligned}
② &\le \sqrt{H\sum_{m=1}^{M} w_m \sum_{h=0}^{H-1} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,h}} \left[\left((g_t^\star - g_t)^\top \nabla_\theta \ln \pi_t(a|s)\right)^2\right]} \\
&= \sqrt{H\sum_{m=1}^{M} w_m \sum_{h=0}^{H-1} \mathop{\mathbb{E}}_{(s,a)\sim d^\star_{m,h}} \left[(g_t^\star - g_t)^\top \nabla_\theta \ln \pi_t(a|s)(\nabla_\theta \ln \pi_t(a|s))^\top (g_t^\star - g_t)\right]} \\
&= \sqrt{H\|g_t^\star - g_t\|^2_{\Sigma^t_{d^\star}}} \\
&\le \sqrt{H\kappa\|g_t^\star - g_t\|^2_{\Sigma_t}}.
\end{aligned}$$

Due to that $g_t^\star$ minimizes $L(g;\theta_t,d^t)$ over the set $\mathcal{G}$, the first-order optimality condition implies that

$$(g - g_t^\star)^\top \nabla_g L(g_t^\star;\theta_t,d^t) \ge 0$$

for any g. Therefore,

$$\begin{aligned}
&L(g;\theta_t,d^t) - L(g_t^\star;\theta_t,d^t) \\
&= \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^t_{m,H-h}} \left[\left(A^{t,\lambda}_{m,h}(s,a) - g_t^{\star\top}\nabla \ln \pi_t(a|s) + (g_t^\star - g)^\top \nabla \ln \pi_t(a|s)\right)^2\right] - L(g_t^\star;\theta_t,d^t) \\
&= \sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^t_{m,H-h}} \left[\left((g_t^\star - g)^\top \nabla_\theta \ln \pi_t(a|s)\right)^2\right] \\
&\quad + (g - g_t^\star)^\top \left(-2\sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^t_{m,H-h}} \left[\left(A^{t,\lambda}_{m,h}(s,a) - g_t^{\star\top}\nabla_\theta \ln \pi_t(a|s)\right)\nabla_\theta \ln \pi_t(a|s)\right]\right) \\
&= \|g_t^\star - g\|^2_{\Sigma_t} + (g - g_t^\star)^\top \nabla_g L(g_t^\star;\theta_t,d^t) \\
&\ge \|g_t^\star - g\|^2_{\Sigma_t}.
\end{aligned}$$

So finally we have

$$\mathrm{err}_t \le \sqrt{HL(g_t^\star;\theta_t,d^\star)} + \sqrt{H\kappa(L(g_t;\theta_t,d^t) - L(g_t^\star;\theta_t,d^t))}.$$

This completes the proof. □

Lemma 17 (Hoeffding's Inequality)**.** *Suppose $X_1, X_2, \dots, X_n$ are i.i.d. random variables taking values in $[a,b]$, with expectation μ. Let $\bar{X}$ denote their average, then for any $\epsilon \ge 0$,*

$$\mathbb{P}\left(\left|\bar{X} - \mu\right| \ge \epsilon\right) \le 2\exp\left(-\frac{2n\epsilon^2}{(b-a)^2}\right).$$

[3] For vector v, $\|v\|_A = \sqrt{v^\top A v}$ for a symmetric positive semi-definite matrix A.

Lemma 18. *For any policy π, any state $s \in \mathcal{S}$ and any $U \geq \ln|\mathcal{A}| - 1$,*

$$0 \leq \sum_{a\in\mathcal{A}} \pi(a|s) \ln \frac{1}{\pi(a|s)} - \sum_{a\in\mathcal{A}} \pi(a|s) \min\left\{\ln \frac{1}{\pi(a|s)}, U\right\} \leq \frac{|\mathcal{A}|}{\mathrm{e}^{U+1}}.$$

Proof. The first inequality is straightforward, so we focus on the second part. Set $\mathcal{A}' = \{a \in \mathcal{A} \,:\, \ln\frac{1}{\pi(a|s)} > U\} = \{a \in \mathcal{A} \,:\, \pi(a|s) < \frac{1}{\mathrm{e}^U}\}$ and $p = \sum_{a\in\mathcal{A}'} \pi(a|s)$, then

$$\begin{aligned}
\sum_{a\in\mathcal{A}} \pi(a|s) \ln \frac{1}{\pi(a|s)} - \sum_{a\in\mathcal{A}} \pi(a|s) \min\left\{\ln \frac{1}{\pi(a|s)}, U\right\} &= \sum_{a\in\mathcal{A}'} \pi(a|s) \ln \frac{1}{\pi(a|s)} - \sum_{a\in\mathcal{A}'} \pi(a|s) U \\
&= p \sum_{a\in\mathcal{A}'} \frac{\pi(a|s)}{p} \ln \frac{1}{\pi(a|s)} - pU \\
&\leq p \ln\left(\sum_{a\in\mathcal{A}'} \frac{\pi(a|s)}{p} \frac{1}{\pi(a|s)}\right) - pU \\
&\leq p \ln \frac{|\mathcal{A}|}{p} - pU,
\end{aligned}$$

where the penultimate step comes from concavity of $\ln x$ and Jensen's inequality. Let $f(p) = p\ln\frac{|\mathcal{A}|}{p} - pU$, then $f'(p) = \ln|\mathcal{A}| - U - 1 - \ln p$. Recall that $U \geq \ln|\mathcal{A}| - 1$, so $f(p)$ increases when $p \in (0, \frac{|\mathcal{A}|}{\mathrm{e}^{U+1}})$ and decreases when $p \in (\frac{|\mathcal{A}|}{\mathrm{e}^{U+1}}, 1)$. Since $f(\frac{|\mathcal{A}|}{\mathrm{e}^{U+1}}) = \frac{|\mathcal{A}|}{\mathrm{e}^{U+1}}$ we complete the proof. $\square$

Lemma 19 (Loss Function Concentration)**.** *If set π_s = None and $U \geq \ln|\mathcal{A}| - 1$, then with probability $1 - 2(T+1)\exp\left(-\frac{2N\epsilon^2}{C^2}\right)$, the update weight sequence of Alg. 1 satisfies: for any $0 \leq t \leq T$,*

$$L(\widehat{g}_t; \theta_t, d^{\theta_t}) - L(g_t^\star; \theta_t, d^{\theta_t}) \leq 2\epsilon + \frac{8\lambda GB|\mathcal{A}|}{\mathrm{e}^{U+1}},$$

where

$$C = 16HGB[1 + \lambda U + H(1 + \lambda \ln|\mathcal{A}|)] + 4HG^2B^2.$$

If $\pi_s \neq$ None and $\lambda = 0$, then with probability $1 - 2(T+1)\exp\left(-\frac{2N\epsilon^2}{C^2}\right)$, the update weight sequence of Alg. 1 satisfies: for any $0 \leq t \leq T$,

$$L(\widehat{g}_t; \theta_t, \widetilde{d}^{\pi_s}) - L(g_t^\star; \theta_t, \widetilde{d}^{\pi_s}) \leq 2\epsilon,$$

where

$$C = 16H^2GB + 4HG^2B^2.$$

Proof. We first prove the π_s = None case. For time step t, Alg. 1 samples HN trajectories. Abusing the notation, denote

$$\begin{aligned}
\widehat{F}_t &= \frac{1}{N}\sum_{n=1}^{N}\sum_{h=0}^{H-1} \nabla_\theta \ln \pi_\theta(a_{n,h}|s_{n,h}) \left(\nabla_\theta \ln \pi_\theta(a_{n,h}|s_{n,h})\right)^\top, \\
\widehat{\nabla}_t &= \frac{1}{N}\sum_{n=1}^{N}\sum_{h=0}^{H-1} \widehat{A}_{n,H-h}(s_{n,h}, a_{n,h}) \nabla_\theta \ln \pi_\theta(a_{n,h}|s_{n,h}), \\
\widehat{L}(g) &= \underbrace{\sum_{m=1}^{M} w_m \sum_{h=1}^{H} \mathop{\mathbb{E}}_{s,a\sim d^{\theta_t}_{m,H-h}} \left[A^{t,\lambda}_{m,h}(s,a)^2\right]}_{①} + \underbrace{g^\top \widehat{F}_t g - 2g^\top \widehat{\nabla}_t}_{②}.
\end{aligned}$$

Notice that ① is a constant. From Alg. 1, $\widehat{g}_t$ is the minimizer of ② (hence $\widehat{L}(g)$) inside the ball $\mathcal{G}$. From $\nabla_\theta \ln \pi_\theta(a|s) = \phi(s,a) - \mathbb{E}_{a'\sim\pi_\theta(\cdot|s)}[\phi(s,a')]$, $\|\phi(s,a)\|_2 \le B$, $\|g\|_2 \le G$, we know that $\left|g^\top \nabla_\theta \ln \pi_\theta(a|s)\right| \le 2GB$. So $0 \le g^\top \widehat{F}_t g \le 4HG^2B^2$. From Alg. 3, we know that any sampled $\widehat{A}$ satisfies $|\widehat{A}| \le 2[1+\lambda U + H(1+\lambda \ln|\mathcal{A}|)]$. So $|g^\top \widehat{\nabla}_t| \le 4HGB[1+\lambda U + H(1+\lambda\ln|\mathcal{A}|)]$. We first have that

$$-8HGB[1+\lambda U + H(1+\lambda\ln|\mathcal{A}|)] \le ② \le 8HGB[1+\lambda U + H(1+\lambda\ln|\mathcal{A}|)] + 4HG^2B^2. \tag{3}$$

To apply any standard concentration inequality, we next need to calculate the expectation of ②. According to Monte Carlo sampling and Lem. 18, for any $1 \le m \le M, 1 \le h \le H$ and $(s,a) \in \mathcal{S}\times\mathcal{A}$, we have

$$A^{t,\lambda}_{m,h}(s,a) - \frac{\lambda|\mathcal{A}|}{\mathrm{e}^{U+1}} \le \mathbb{E}\left[\widehat{A}^{t,\lambda}_{m,h}(s,a)\right] \le A^{t,\lambda}_{m,h}(s,a).$$

Denote ∇_t as the exact policy gradient at time step t, then

$$\left|\mathbb{E}\left[g^\top\widehat{\nabla}_t\right] - g^\top\nabla_t\right| \le \|g\|_2 \left\|\mathbb{E}\left[\widehat{\nabla}_t\right] - \nabla_t\right\|_2 \le \|g\|_2 \cdot H\|\nabla_\theta \ln\pi_\theta(a|s)\|_2 \left\|\mathbb{E}\left[\widehat{A}(s,a)\right] - A(s,a)\right\|_\infty \le \frac{2\lambda GB|\mathcal{A}|}{\mathrm{e}^{U+1}}.$$

Since Monte Carlo sampling correctly estimates state-action visitation distribution, $\mathbb{E}\left[\widehat{F}_t\right] = F(\theta_t)$. Notice that $g^\top\widehat{F}_t g$ is linear in entries of $\widehat{F}_t$, we have $\mathbb{E}\left[g^\top\widehat{F}_t g\right] = g^\top F(\theta_t) g$. Now we are in the position to show that

$$\left|\mathbb{E}\left[\widehat{L}(g)\right] - L(g)\right| \le \frac{4\lambda GB|\mathcal{A}|}{\mathrm{e}^{U+1}}.$$

Hoeffding's inequality (Lem. 17) gives

$$\mathbb{P}\left(\left|\widehat{L}(g) - \mathbb{E}\left[\widehat{L}(g)\right]\right| \ge \epsilon\right) \le 2\exp\left(-\frac{2N\epsilon^2}{C^2}\right).$$

where from Eq. 3,

$$C = 16HGB[1+\lambda U + H(1+\lambda\ln|\mathcal{A}|)] + 4HG^2B^2.$$

After applying union bound for all t, with probability $1 - 2(T+1)\exp\left(-\frac{2N\epsilon^2}{C^2}\right)$ the following holds for any $g \in \mathcal{G}$:

$$\left|\widehat{L}(g;\theta_t,d^{\theta_t}) - L(g;\theta_t,d^{\theta_t})\right| \le \epsilon + \frac{4\lambda GB|\mathcal{A}|}{\mathrm{e}^{U+1}}.$$

Hence

$$\begin{aligned} L(\widehat{g}_t;\theta_t,d^{\theta_t}) &\le \widehat{L}(\widehat{g}_t;\theta_t,d^{\theta_t}) + \epsilon + \frac{4\lambda GB|\mathcal{A}|}{\mathrm{e}^{U+1}} \\ &\le \widehat{L}(g_t^\star;\theta_t,d^{\theta_t}) + \epsilon + \frac{4\lambda GB|\mathcal{A}|}{\mathrm{e}^{U+1}} \\ &\le L(g_t^\star;\theta_t,d^{\theta_t}) + 2\epsilon + \frac{8\lambda GB|\mathcal{A}|}{\mathrm{e}^{U+1}}. \end{aligned}$$

For $\pi_s \ne$ None and $\lambda = 0$, we notice that $|\widehat{A}| \le 2H$ and hence $-8H^2GB \le ② \le 8H^2GB + 4HG^2B^2$. Moreover, $\mathbb{E}\left[\widehat{A}^{t,\lambda}_{m,h}(s,a)\right] = A^{t,\lambda}_{m,h}(s,a)$. So by slightly modifying the proof we can get the result. □