Curriculum Reinforcement Learning using Optimal Transport via Gradual Domain Adaptation

Anonymous Author(s)
Affiliation
Address
email

Abstract

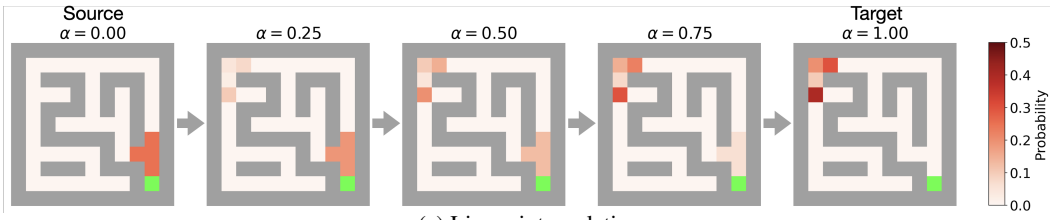
Curriculum Reinforcement Learning (CRL) aims to create a sequence of tasks, starting from easy ones and gradually learning towards some difficult tasks. In this work, we focus on the idea of framing CRL as interpolations between a *source* (auxiliary) and a *target* task distribution. Although existing studies have shown the great potential of this idea, it remains unclear how to formally quantify and generate the movement between task distributions. Inspired by the insights from gradual domain adaptation in semi-supervised learning, we create a natural curriculum by breaking down the potentially large task distributional shift in CRL into smaller shifts. We propose GRADIENT, which formulates CRL as an optimal transport problem with a tailored distance metric between tasks. Specifically, we generate a sequence of task distributions as a geodesic interpolation between the source and target distributions, i.e., the Wasserstein barycenters. Different from many existing methods, our algorithm considers a task-dependent contextual distance metric and is capable of handling non-parametric distributions in both continuous and discrete context settings. In addition, we theoretically show that GRADIENT enables smooth transferring between subsequent stages in the curriculum under certain conditions. Our empirical results demonstrate that the proposed algorithm achieves high performance in terms of learning efficiency and asymptotic performance in a wide range of tasks.

1 Introduction

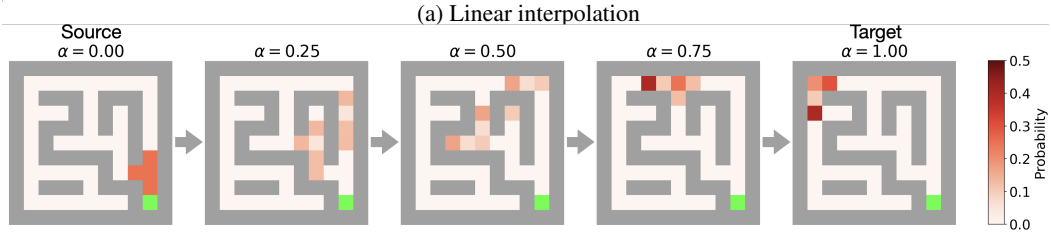
Reinforcement Learning (RL) [1] has demonstrated great potential in solving complex decision-making tasks, including but not limited to video games [2], chess [3], and robotic manipulation [4]. Among them, various prior works highlight the daunting challenges resulting from sparse rewards. For example, in the maze navigation task, since the agent needs to navigate from the initial position to the goal to receive a positive reward, the task requires a large amount of randomized exploration. One solution to address this issue is the Curriculum Reinforcement Learning (CRL) [5], of which the objective is to create a sequence of environments to facilitate the learning of some difficult tasks.

Although there is a wide range of different interpretations of CRL, we focus on one interpretation of CRL, which interprets the curriculum as a sequence of task distributions that interpolate between a *source* (auxiliary) task distribution and a *target* task distribution [6, 7, 8, 9, 10, 11]. This interpretation allows more general presentations of tasks and a wider range of objectives such as generalization (uniform distribution over the task collection) or learning to complete difficult tasks (a subset of the task collection). Again using the maze navigation as an example, given a fixed maze layout and goal,

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



(a) Linear interpolation



(b) GRADIENT interpolation (ours)

Figure 1: Intermediate task distributions generated by (a) linear interpolation and (b) our method for the maze navigation task. The green cell represents the goal. The red color cells represent the initial positions (the darker the color is, the higher the probability is). The source task distribution is the first column, and the target task distribution is the last column. In (a), linear interpolations do not cover the space where the source and the target have zero probability, which does not benefit the learning. In contrast, (b) GRADIENT interpolations creates a curriculum that gradually morphs from the source to the target, covering tasks of intermediate difficulty and improving the learning efficiency.

a task is then defined by a start position, and a task distribution is a categorical distribution over all the possible start positions. With a target distribution putting mass over start positions far away from the goal position, a natural curriculum to accelerate the learning is to first put start positions close to the goal position, and gradually move them towards the target distribution.

However, most of the existing methods interpreting the curriculum as shifting distributions use Kullback–Leibler (KL) divergence to measure the distance between distributions, which imposes several restrictions. First, constrained by either the formulation or the computational feasibility, existing methods often assume the distribution to be parametrized, namely Gaussian [6, 7, 8, 11], which limits the usage in many cases. Second, most of the algorithms using KL divergence implicitly restricts the similarity between tasks to be measured by the l_2 distance between two continuous environment parameters, which ignores the metric information and is insufficient when the environment parameter is discrete.

In light of the aforementioned issues with the existing CRL method, we propose GRADIENT, an algorithm that creates a sequence of task distributions that gradually morph from the source to the target distribution using Optimal Transport (OT). In this work, we first define a distance metric between the individual task, then we can find a series of task distributions that interpolates between the easy and the difficult task distribution. GRADIENT is able to deal with both discrete and continuous environment parameter space, and arbitrary distributions (represented either by explicit categorical distributions or implicit empirical distributions of particles). Under mild conditions, GRADIENT provably ensures a *smooth* adaptation from the learned policy of one stage in the curriculum to the next.

We summarize our main contributions as follows:

1. We propose GRADIENT, a novel CRL framework based on optimal transport formulation to generate gradually morphing intermediate task distributions. As a result, GRADIENT requires little effort to transfer between subsequent stages and therefore improves the learning efficiency towards difficult tasks.
2. We develop π-contextual-distance to measure the task similarity and compute the Wasserstein barycenters as intermediate task distributions. Our proposed method is able to deal with both continuous and discrete context space as well non-parametric distributions. We also prove the theoretical bound of policy transfer which leads to practical insights.
3. We demonstrate empirically that our method improves the learning efficiency and asymptotic performance in a wide range of tasks compared with state-of-the-art CRL baselines.

2 Related Work

Curriculum learning in RL Curriculum reinforcement learning (CRL) [5, 12] focuses on the generation of training environments for RL agents. There are several objectives in CRL: improving learning efficiency towards difficult tasks (*time-to-threshold*), maximum return (*asymptotic performance*), or transfer performance in unseen tasks (*generalization*). From a domain randomization perspective, Active Domain Randomization [4, 13] uses curricula to diversify the physical parameters of the simulator to facilitate the generalization in sim-to-real transfer. From a game-theoretical perspective, adversarial training is also developed to improve the robustness of RL agents in unseen environments [14, 15, 16]. From a "intrinsic motivation" perspective, methods have been proposed to create curricula even in the absence of a target task to be accomplished [17, 11, 18].

In this work, we focus on another stream of works that interprets CRL as an explicit interpolation between an auxiliary task distribution and a difficult task distribution [6, 7, 8, 9]. Self-Paced Reinforcement Learning (SPRL) [6] is proposed to generate intermediate distributions by measuring the task distribution similarity using Kullback–Leibler (KL) divergence. However, as we will show in this paper, KL divergence brings several shortcomings that may impede the usage of the algorithms. First, although the formulation of [6, 7, 8, 9] does not restrict the distribution class, the algorithm realization requires the explicit computation of KL divergence, which is analytically tractably only for a restricted family of distributions. Second, KL divergence implicitly constrains the distance measure between two contexts to be l_2 distance, which is not applicable in discrete context settings. In this work, we use Wasserstein distance instead of KL divergence to measure the distance between distributions. Unlike KL divergence, Wasserstein distance takes into consideration the metric information and opens up a wide variety of task distance measures. Prior to our work, WB-SPDL [19] replaces KL divergence with 2-Wasserstein distance under euclidean distance. Despite both using Wasserstein distance, our work is different in three aspects mainly: 1) the definition of distance function (l_2 only v.s. our task-informed contextual-distance), 2) the definition of barycenters (local interpolation between the current distribution, target distribution, and value distribution v.s. our geodesics between the source and target distribution), 3) the algorithm design (solving a complex bilevel optimization problem by approximations v.s. our solution using off-the-shelf OT solvers).

Gradual domain adaptation in Semi-Supervised Learning Gradual domain adaptation (GDA) [20, 21, 22, 23, 24, 25, 26, 27] considers the problem of transferring a classifier trained in a source domain with labeled data, to a target domain with unlabelled data, given only unlabelled data that gradually shifts from the source to the target domain. GDA solves this problem by designing a sequence of learning tasks, in which the classifier is retrained using the pseudolabels created using the classifier from the last stage in the sequence. Most of the existing literature assumes that there exist intermediate domains. However, there are a few works aiming to tackle the problem when intermediate domains, or the index (i.e., stage in the curriculum), are not readily available. A coarse-to-fine framework is proposed to sort and index intermediate domain data [27]. Another work [26] proposes to create virtual samples from intermediate distributions by interpolating representations of examples from source and target domains and suggests using the optimal transport map to create interpolate data in semi-supervised learning. A recent work [21] demonstrates theoretically that the optimal path of samples is the geodesic interpolation defined by the optimal transport map. Our work is inspired by the "divide and conquer" methodology in GDA and also uses geodesics as our curriculum plan (although in a different task space).

3 Preliminary

3.1 Contextual Markov Decision Process

A Contextual Markov Decision Process (CMDP) extends the standard single-task MDP to a multi-task setting. In this work, we consider discounted infinite-horizon CMDPs, represented as a tuple $M = (\mathcal{S}, \mathcal{C}, \mathcal{A}, R, P, p_0, \rho, \gamma)$. Here, $\mathcal{S}$ is the state space, $\mathcal{C}$ is the context space, $\mathcal{A}$ is the action space, $R : \mathcal{S} \times \mathcal{A} \times \mathcal{C} \mapsto \mathbb{R}$ is the context-dependent reward function, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{C} \mapsto \Delta(\mathcal{S})$ is the context-dependent transition function, $p_0 : \mathcal{C} \mapsto \Delta(\mathcal{S})$ is the context-dependent initial state

distribution, $\rho \in \Delta(\mathcal{C})$ is the context distribution and $\gamma \in (0, 1)$ is the discount factor. Note that goal-conditioned reinforcement learning [10] can be considered as a special case of the CMDP.

To sample a trajectory $\boldsymbol{\tau} := \{s_t, a_t, r_t\}_{t=0}^{\infty}$ in CMDPs, the context $c \sim \rho$ is randomly generated by the environment at the beginning of each episode. With the initial state $s_0 \sim p_0(\cdot \mid c)$, at each time step t, the agent follows a policy π to select an action $a_t \sim \pi(s_t)$ and receives a reward $R(s_t, a_t, c)$. Then the environment transits to the next state $s_{t+1} \sim P(\cdot \mid s_t, a_t, c)$. Contextual reinforcement learning naturally extends the original RL objective to include the context distribution ρ. To find the optimal policy, we need to solve the following optimization problem:

$$\max_{\pi} V^{\pi}(\rho) = \max_{\pi} \mathbb{E}_{\boldsymbol{\tau}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, c) \mid c \sim \rho; \pi \right] \tag{1}$$

3.2 Optimal Transport

Wasserstein distance The basic problem in optimal transport [28], termed as Kantorovich problem [29], aims to find the optimal coupling θ^* which minimizes the transportation cost between measures $\mu, \nu \in \mathcal{M}(\mathcal{C})$. Therefore, the *Wasserstein distance* defines the distance between probability distribution:

$$\mathcal{W}_d(\mu, \nu) = \inf_{\theta \in \Theta(\mu,\nu)} \int_{\mathcal{C}\times\mathcal{C}} d(c_s, c_t)\, \mathrm{d}\theta(c_s, c_t), \text{ subject to } \Theta = \left\{\theta : \gamma^{\mathcal{C}}_{\#}\theta = \mu, \gamma^{\mathcal{C}}_{\#}\theta = \nu\right\} \tag{2}$$

where $\mathcal{C}$ is the support space, $\Theta(\mu, \nu)$ is the set of all couplings between μ and ν, $d(\cdot, \cdot) : \mathcal{C} \times \mathcal{C} \mapsto \mathbb{R}_{\geq 0}$is a distance function, $\gamma^{\mathcal{C}}$ is the projection from $\mathcal{C} \times \mathcal{C}$ onto $\mathcal{C}$, and $T_{\#}P$ generally denotes the push-forward measure of P by a map T. This optimization is well-defined and the optimal θ always exists under mild conditions [28].

Wasserstein Geodesics and Wasserstein Barycenter To construct a curriculum that allows the agent to efficiently move to the difficult task distribution. We want to follow the *Wasserstein geodesic* [30], the shortest path [21] between the source and the target distributions, under the Wasserstein distance. Meanwhile, while the original Wasserstein barycenter [31] problem concerns the Frèchet mean of multiple probability measures in a space endowed with the Wasserstein metric, we consider only two distributions μ_0 and μ_1, and the set of barycenters of them is just the geodesic curve given by McCann's interpolation [32]. Thus, the interpolation between two given distributions μ_0 and μ_1 is defined as the minimizer of following Wasserstein barycenter objective $f(\nu)$: $f(\nu_\alpha) := (1 - \alpha)\mathcal{W}(\mu_0, \nu_\alpha) + \alpha\mathcal{W}(\nu_\alpha, \mu_1)$ where $\alpha \in [0, 1]$ specifies one unique interpolation distribution on the geodesic. While the computational cost of Wasserstein distance objective eq.(2) could be a potential obstacle, we follow the entropic optimal transport and utilize the celebrated Sinkhorn's algorithm [33]. Moreover, we adopt smoothing bias to solve for scalable debiased Sinkhorn barycenters [34].

4 Curriculum Reinforcement Learning using Optimal Transport

We first formulate the curriculum generation as an interpolation problem of probability distributions in Section 4.1. Then we propose a distance measure between contexts in Section 4.2 in order to compute the Wasserstein barycenter. Next, we show our main algorithm, GRADIENT, in Section 4.3 and an associated theorem that provides practical insights in Section 4.4.

4.1 Problem Formulation

Formally, given a target task distribution $\nu(c) \in \Delta(\mathcal{C})$, we aim to automatically generate a curriculum of task distributions, $\rho_0(c), \rho_1(c), \ldots, \rho_K(c)$ with K stages, that enables the agent to gradually adapt from the auxiliary task distribution $\mu(c)$ to the target $\nu(c)$, i.e., $\rho_0(c) \to \mu(c)$ and $\rho_K(c) \to \nu(c)$. If the context space $\mathcal{C}$ is discrete (sometimes called fixed-support in OT [28]) with cardinality $|\mathcal{C}|$, the task distribution is represented by a categorical distribution, e.g., $\mu(c) = [p(c_1), p(c_2), ..., p(c_{|\mathcal{C}|})]$, where $p(c_i) \geq 0, \sum_i p(c_i) = 1$. Whereas if the context space $\mathcal{C}$ is continuous (sometimes called

free-support in OT), the task distribution is then approximated by a set of particles sampled from the distribution, e.g., $\mu(c) \approx \hat{\mu}(c) = \frac{1}{n_s}\sum_{i=1}^{n_s} \delta_{c_{si}}(c)$, where $\delta_{c_{si}}(c)$ is a Dirac delta at c_{si}. This highlights the capability of dealing with both parametric and non-parametric distributions in our formulation and algorithms, in contrast to existing algorithms [6, 11, 6, 7, 8, 9].

4.2 Contextual Distance Metrics

To define the distance between contexts, we start by defining the distance between states. **Bisimulation metrics** [35, 36, 37] are metrics to measure the "behaviorally equivalence" between states by defining a recursive relationship between states. Recent literature in representation learning for RL control [38, 39] also uses the bisimulation concept to train state encoders that facilitate multi-tasking. [40] also uses this concept to enforce the policy to visit state-action pairs close to the support of logged transitions in offline RL.

However, the bisimulation metric is inherently "pessimistic" because they consider equivalence under all actions, even including the actions that never lead to positive outcomes for the agent [37]. Even if $V^*(s_i) = V^*(s_j)$, the bisimulation metric could be arbitrarily large. To address this issue, [37] proposes on-policy π-bisimulation that considers the dynamics induced by the policy π. Similarly, we extend this notion to the CMDP settings and define the **contextual $\boldsymbol{\pi}$-bisimulation metric**:

$$d^{\pi}_{c_i,c_j}(s_i, s_j) = |R^{\pi}_{s_i,c_i} - R^{\pi}_{s_j,c_j}| + \gamma \mathcal{W}_d(\mathcal{P}^{\pi}_{s_i,c_i}, \mathcal{P}^{\pi}_{s_j,c_j}) \tag{3}$$

where $R^{\pi}_{s,c} := \sum_a \pi(a \mid s) R(s, a, c)$ and $P^{\pi}_{s,c} := \sum_a P(\cdot \mid s, a, c)\pi(a \mid s)$. With the definition of contextual π-bisimulation metric, we are ready to propose the **$\boldsymbol{\pi}$-contextual-distance** to measure the distance between two contexts:

Definition 4.1 ($\boldsymbol{\pi}$-contextual-distance) *Given a CMDP $M = (\mathcal{S}, \mathcal{C}, \mathcal{A}, R, P, p_0, \rho, \gamma,)$, the distance between two context $d^{\pi}(c_i, c_j)$ under the policy π is defined as*

$$d^{\pi}(c_i, c_j) = \mathbb{E}_{s_i \sim p_0(\cdot|c_i), s_j \sim p_0(\cdot|c_j)} \left[d^{\pi}_{c_i,c_j}(s_i, s_j) \right] \tag{4}$$

Essentially, the **$\boldsymbol{\pi}$-contextual-distance** measures the performance difference between two contexts c_i and c_j under π. The computation is generally applied in a dynamic-programming fashion, which requires considering all possible states and actions. However, it is not always possible to have access to that especially when the data is collected by an agent. Fortunately, we can compute the π-contextual-distance via samples in a similar fashion to Deep Q-Learning [2]. The algorithm is detailed in the Appendix C. There are situations where it is reasonable to use l_2 as a surrogate metric when the context space resembles the Euclidean space, such as in goal-conditioned continuous environments. With the concepts of a contextual distance metric, we now introduce the algorithms to generate intermediate task distribution to enable the agent to gradually transfer from the source to the target tasks distribution.

4.3 Algorithms

We present our main algorithm in Algorithm 1. We introduce an interpolation factor $\Delta\alpha$ to decide the "difference" between two subsequent task distributions in the curriculum. Smaller $\Delta\alpha$ means a smaller difference between subsequent task distributions. In this work, we treat $\Delta\alpha$ as a constant for simplicity and show empirically that constant $\Delta\alpha$ already performs well in our experiments. We leave the adaptive $\Delta\alpha$ to future works. Note that the $\Delta\alpha$ is analogical to the KL divergence constraint in [7, 8, 6]. The main difference is that we use Wasserstein distance instead of KL divergence.

At the beginning of each stage in the curriculum, we add a $\Delta\alpha$ to the previous α (starting from 0). Then we pass the α into the function `ComputeBarycenter` (Algorithm 2) to generate the intermediate task distribution. Note that when $\alpha = 0$ and 1, the generated distribution are the source and target distribution respectively. In the *ComputeBarycenter* function, the computation method differs depending on whether the context is discrete or continuous. After generating the intermediate task distribution, we optimize the agent in this task distribution until the accumulative return G reaches the threshold $\bar{G}$. Then the curriculum enters the next stage and repeats the process until

$\alpha = 1$. In other words, the path of the intermediate task distributions is the OT geodesic on the manifold defined by Wasserstein distance.

Algorithm 1: GRAdual **D**omain adaptation for curriculum re**I**nforc**E**ment lear**N**ing via optimal **T**ransport (**GRADIENT**)

Input: Source task distribution $\mu(c)$, target task distribution $\nu(c)$, interpolation factor $\Delta\alpha$, distance metric d, reward threshold $\bar{G}$
Initialize the agent policy π
for k *in* $0, 1, 2, ..., K$ **do**
  $\alpha \leftarrow \min(k * \Delta\alpha, 1)$
  $\rho(c) \leftarrow$ `ComputeBarycenter`(μ, ν, α, d)
  do
    $G \leftarrow$ Optimize π in the task distribution $\rho(c)$ and return training reward
  until $G > \bar{G}$;
Output: Agent policy π

Algorithm 2: ComputeBarycenter

Input: Source and target μ, ν, interpolation constant α, distance metric d
$w_1 \leftarrow \alpha, w_2 \leftarrow 1 - \alpha, (b_1, b_2), l \leftarrow \mathbf{1}$
if $\mathcal{C}$ *is discrete* **then**
  $\lambda_1 \leftarrow \mu(c)$, $\lambda_2 \leftarrow \nu(c)$
  Cost matrix: $\mathbf{C} := d(\cdot, \cdot) : \mathcal{C} \times \mathcal{C} \mapsto \mathbb{R}$
else
  $\lambda_1 = \frac{1}{n_s} \sum_i^{n_s} c_{si}$, $\lambda_2 = \frac{1}{n_t} \sum_j^{n_t} c_{tj}$
  Cost matrix $\mathbf{C} : C_{ij} = d(c_{si}, c_{tj})$
$\mathbf{K} = exp\left(-\mathbf{C}/\epsilon\right)$ `// Sinkhorn param` ϵ
do
  $a_1 \leftarrow (\lambda_1/\mathbf{K}b_1)\,, a_2 \leftarrow (\lambda_2/\mathbf{K}b_2)$
  $\rho \leftarrow l \odot \prod_{m=1}^{2} (\mathbf{K}^\top a_m)^{w_m}$
  $b_1 \leftarrow \left(\rho/\mathbf{K}^\top a_1\right), b_2 \leftarrow \left(\rho/\mathbf{K}^\top a_2\right)$
  $l \leftarrow \sqrt{l \odot (\rho/\mathbf{K}l)}$
until *convergence*;
Output: ρ

4.4 Theoretical Analysis

The proposed algorithm benefits from breaking the difficulty of learning in the target domain into multiple small challenges by designing a sequence of K stages that gradually morph towards the target. This motivates us to theoretically answer the following questions:

Can we learn the optimal policy rapidly in a new stage based on the output of the previous one?

Towards this, motivated by the maze navigation example shown in Figure 1, throughout this section, we focus on the collection of CMDPs which obeys the following assumption:

Assumption 4.1 (Consistency and homogeneity) *The initial state distribution is consistent with the associated context distribution, namely $p_0(s \,|\, c) = \mathbb{1}(s = c)$, which indicates $\mathcal{S} \subseteq \mathcal{C}$. In addition, the transition kernels and rewards of the CMDP are homogeneous with respect to the context. Specifically,*

$$P(s' \,|\, s, a, c_i) = P(s' \,|\, s, a, c_j), \quad R(s, a, c_i) = R(s, a, c_j), \qquad \forall c_i, c_j \in \mathcal{C}, s \in \mathcal{S}, a \in \mathcal{A}.$$

To continue, for k-th stage associated with the context distribution $\rho_k \in \Delta(\mathcal{C})$, we denote the optimal policy in k-th stage as $\pi_k^\star := \arg\max_\pi V^\pi(\rho_k)$. Armed with the above notations and assumptions, we are positioned to introduce our main theorem.

Theorem 4.1 *Let m be a problem-dependent positive constant. Consider any MDP under the assumption 4.1 and the situations further specified in Appendix A.1. For any two subsequent stages $k, k+1 \in [K]$, the optimal policies $\pi_k^\star, \pi_{k+1}^\star$ obey*

$$V^{\pi_{k+1}^\star}(\rho_{k+1}) - V^{\pi_k^\star}(\rho_{k+1}) \leq m\mathcal{W}_{d^\star}(\rho_k, \rho_{k+1}). \tag{5}$$

Here, $d^\star$ is the distance metric defined on the metric space $(\mathcal{C}, d^\star)$ which obeys ($\mathcal{U}(\mathcal{C})$ denote the uniform distribution over $\mathcal{C}$)

$$\pi^\star := \max_\pi V^\pi\big(\mathcal{U}(\mathcal{C})\big), \qquad d^\star(\cdot, \cdot) := d^{\pi^\star}(\cdot, \cdot). \tag{6}$$

The proof can be found in Appendix A.4. In addition, the following corollary can be directly verified by Theorem 4.1.

Corollary 4.1.1 *By constant speed geodesics (7.2 of [41]), the stages generated by GRADIENT obey*

$$V^{\pi_{k+1}^\star}(\rho_{k+1}) - V^{\pi_k^\star}(\rho_{k+1}) \leq m\Delta\alpha\mathcal{W}_{d^\star}(\mu, \nu). \tag{7}$$

Remark 4.1 *Theorem 4.1 combined with Corollary 4.1.1 indicates that with tailored stages by GRADIENT (sufficiently small $\Delta\alpha$), it is easy to adapt the learned optimal policy $\pi_k^\star$ in the current stage to the optimal policy $\pi_{k+1}^\star$ in the next stage defined by ρ_{k+1}, since the performance gap is small. In particular, after we obtain the optimal policy $\pi_k^\star$ associated with the k-th stage in the curriculum and transfer this policy to the next stage, the performance gap between it and the optimal policy in the $(k+1)$-th stage can be controlled by the Wasserstein distance between two subsequent context distributions and therefore a fraction of the Wasserstein distance between the source and the target (which can be seen as a constant). By properly controlling the Wasserstein distance using GRADIENT, we can achieve gradual domain adaptation efficiently from the source to the target task distribution.*

5 Experiments

Evaluation Metric To quantify the benefit of CRL, we compare the learning progress of an agent evaluated on the target task distribution. More concretely, the metric we use is *time-to-threshold* [5], which shows how fast an agent can learn a policy that achieves a satisfying return on the target task if it transfers knowledge. We also compare the *asymptotic performance* of the agent. For the learner, we use the SAC [42] and PPO [43] implementation provided in the `StableBaseline3` library [44].

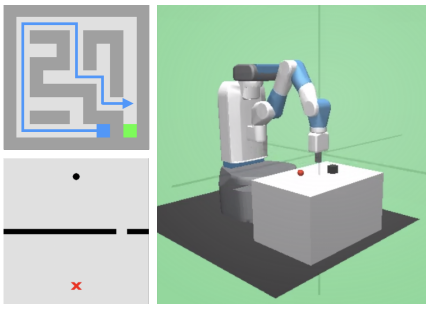


Figure 2: Environments used for experiments. For the **Maze** (upper-left), the context is the initial state. For the **PointMass** (lower-left), the context is the position and width of the gap on the wall. For the **FetchPush**, the context is the goal position on a 2d surface.

Environments The first **Maze** environment is to investigate the performance of GRADIENT and the effect of $\Delta\alpha$. Maze has a discrete context space and therefore the exact context distance metric is available. The next two environments are to investigate the benefits of GRADIENT when the context space is continuous and we use l_2 distance as a surrogate pseudometric. In the **PointMass** environment, the source and the target distribution are both Gaussian distributions, which are in line with the settings of existing algorithms. In the **FetchPush** environment, we want to investigate the case where the source and the target are not Gaussian distributions.

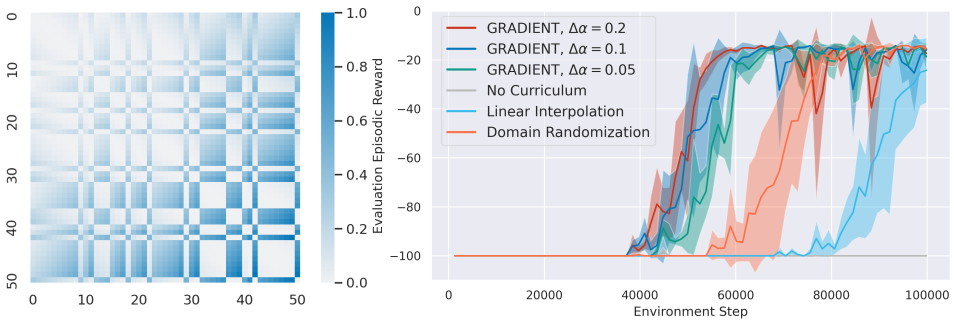
We benchmark our algorithm with six baselines including four CRL algorithms:

1. `No-Curriculum`: sampling contexts according to the target task distribution.
2. `Domain-Randomization`: uniformly sampling the contexts from the context space.
3. `Linear-Interpolation`: sampling contexts based on a weighted average between two distributions, i.e., $(1-\alpha)\mu + \alpha\nu$.
4. `ALP-GMM` [11]: the state-of-the-art intrinsically motivated CRL algorithm that maximizes the absolute learning progress of the agent.
5. `Goal-GAN` [10]: using GAN to generate contexts of intermediate difficulty and automatically explore the goal space.
6. `SPDL` [6] and `WB-SPDL` [19]: interpreting the curriculum generation as a variational inference problem to progressively approach the target task distribution. `WB-SPDL` extends SPDL by using Wasserstein distance instead of KL divergence.

5.1 Can GRADIENT Handle Discrete Context and What is the Effect of $\Delta\alpha$?

In the **Maze** task, we fix the layout of the maze and the goal. The state and action space are discrete. The action space includes moving in four directions. The context is the initial position, and therefore the context space overlaps with the state space. The observation is the flattened value representation of the maze, including the goal, the current position, and the layout.

We visualize the context distance metric in Figure 3a (computed using the optimal A* policy and normalized to make the maximum value be 1). The distance metric can be represented by a symmetrical matrix. We then generate curricula using `GRADIENT` with $\Delta\alpha = 0.2, 0.1, 0.05$. From Figure 3b, our



(a) π-contextual-distance (b) Evaluation reward in the target

Figure 3: The π-contextual-distance and evaluation results in Maze. The shaded area represents the standard deviation. We use a PPO learner and show that GRADIENT converges faster than baselines.

proposed method outperforms the baselines significantly in terms of the time-to-threshold evaluated in the target distribution. With a smaller $\Delta\alpha$, GRADIENT learns slightly slower; nevertheless, all choices achieve good performance. In this simple environment, Domain-Randomization also improves the learning efficiency compared with No-Curriculum (which cannot learn at all). Interestingly, Domain-Randomization even outperforms the Linear-Interpolation, which demonstrates the caveat of bad intermediate task distribution.

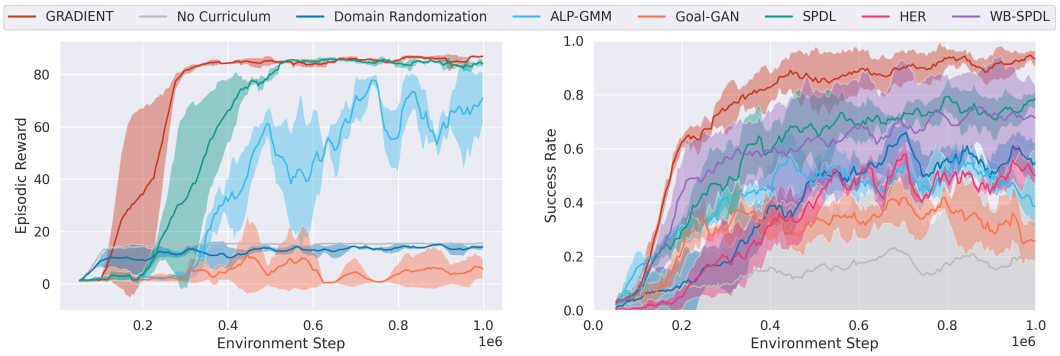
5.2 How Does GRADIENT Perform When the Distributions Are Gaussian?

In PointMass [6], the agent needs to navigate a pointmass through a wall with a small gap at an off-center position to reach the goal. The context is a 2-dimension vector representing the width and position of the gap on the wall. Following the setting in the original SPDL paper [6], the target distribution is an isotropic Gaussian distribution centered at $[2.5, 0.5]$ with a negligible variance $[4 \times 10^{-3}, 3.75 \times 10^{-3}]$ (which is effectively a point as shown in Figure 5a). The source distribution is an isotropic Gaussian distribution centered at $[0, 4.25]$ with a variance of $[2, 1.875]$.

From Figure 4a, we observe that both GRADIENT and SPDL significantly outperform other baselines in terms of the time-to-threshold and achieve the same asymptotic performance. It is reasonable since these two methods are capable of specifying the target distribution to guide curriculum generation. We present the curricula visualization of both methods in Figure 5a and show that the intermediate task distributions gradually move from the source to the target distribution. However, the other three baselines lack the flexibility to leverage the target distribution. This experiment highlights the importance of being able to specify the target distribution.

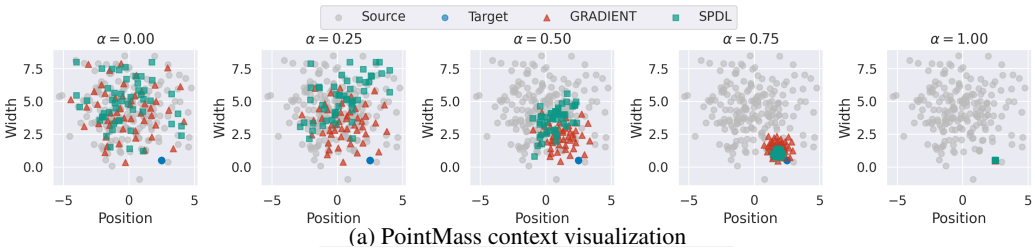
5.3 How Does GRADIENT Perform When the Distributions Are NOT Gaussian?

In FetchPush [45], the objective is to use the gripper to push the box to a goal position. The observation space is a 28-dimension vector, including information about the goal. The context is a

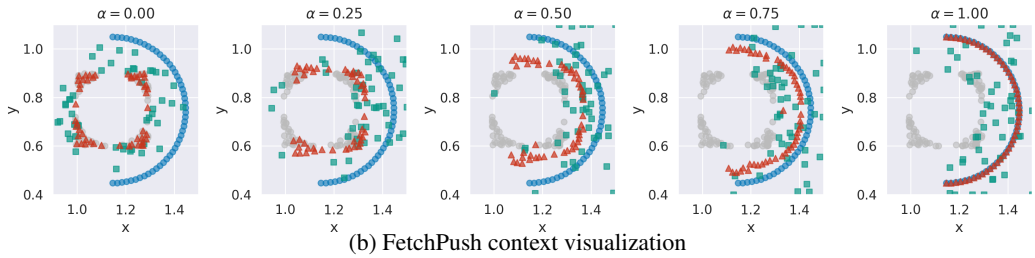


(a) PointMass (b) FetchPush

Figure 4: Evaluation performance in the target task distribution (with a rolling average of 10) for PointMass and FetchPush. In PointMass, where both the source and target distributions are defined by a parametric Gaussian, GRADIENT converges faster than baselines. In FetchPush where the target task distribution can NOT be accurately approximated by a Gaussian, GRADIENT converges faster and has a better asymptotic performance. We use SAC as learners for both environments.



(a) PointMass context visualization



(b) FetchPush context visualization

Figure 5: Visualizations of context distributions and curricula in PointMass and FetchPush. We present GRADIENT's curricula corresponding to $\alpha = 0.00, 0.25, 0.50, 0.75, 1.00$ together with SPDL's contexts from 5 representative stages. For PointMass, the contexts of SPDL are taken from environment steps at 10k, 50k, 100k, 200k, and 300k. For FetchPush, the contexts of SPDL are taken from environment steps at 5k, 50k, 100k, 250k, and 500k.

2-dimension vector representing the goal position on a surface. The target distribution is a uniform distribution over the circumference of a half-circle (Figure 5b). The source distribution is a uniform distribution over a square region centered at the box position, excluding the region within a certain radius of the object. We use this experiment to highlight the importance of the capability to handle arbitrary distributions rather than only the parametric Gaussian distributions. Since SPDL can only deal with parametric Gaussian distribution, we first fit the target and the source distribution with two Gaussian distributions and feed them into the baseline algorithms.

From Figure 4b, our proposed algorithm outperforms all the baseline methods in terms of the time-to-threshold in the target distribution. For ALP-GMM and Goal-GAN, since these two methods do not consider the target distribution, they achieve lower performance than the target-aware baseline such as SPDL and GRADIENT. We further compare these two target-aware methods and find that GRADIENT outperforms the SPDL in terms of learning efficiency and asymptotic performance. The reason can be explained by visualizing the context distribution in Figure 5b. SPDL indeed shows the behavior of moving the context distribution towards the target. However, since the target distribution cannot be fit with a Gaussian well, many sampled contexts are in fact far away from the target distribution, which may not improve the learning efficiency in the target domain. Although Domain-Randomization learns slightly lower than the CRL methods, it achieves even higher asymptotic performance than ALP-GMM and Goal-GAN. The potential reason could be that it does not bias towards a certain area while ALP-GMM and Goal-GAN are possible.

6 Limitations

In our experiments, we examine the settings that are either discrete or low-dimensional. Such experiment design choice is common in the related works and baselines [11, 10, 6]. It comes from both the fact of many tasks can be represented by a low-dimensional vector, such as goal positioned or physical parameters of robots, and the consideration of easier visualization. We believe how to deal with high-dimensional contexts, such as images, is a very promising future direction. In addition, our analysis is conducted based on a restrictive subset of contextual MDPs (Assumption 4.1)where the context controls only the initial state distribution. It would be beneficial to relax this assumption by imposing assumptions on the specific manner in which the context controls the dynamics or reward. Nevertheless, we believe that our main theorem is a significant first step toward a theoretical understanding of GDA in CRL domain.

References

[1] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[2] 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.

[3] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[4] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[5] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of machine learning research: JMLR*, 21(181):1–50, 2020.

[6] Pascal Klink, Hany Abdulsamad, Boris Belousov, Carlo D'Eramo, Jan Peters, and Joni Pajarinen. A probabilistic interpretation of self-paced learning with applications to reinforcement learning. *Journal of Machine Learning Research*, 22(182):1–52, 2021.

[7] Pascal Klink, Carlo D'Eramo, Jan R Peters, and Joni Pajarinen. Self-paced deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33:9216–9227, 2020.

[8] Pascal Klink, Hany Abdulsamad, Boris Belousov, and Jan Peters. Self-paced contextual reinforcement learning. In *Conference on Robot Learning*, pages 513–529. PMLR, 2020.

[9] Jiayu Chen, Yuanxin Zhang, Yuanfan Xu, Huimin Ma, Huazhong Yang, Jiaming Song, Yu Wang, and Yi Wu. Variational automatic curriculum learning for sparse-reward cooperative multi-agent problems. *Advances in Neural Information Processing Systems*, 34, 2021.

[10] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*, pages 1515–1528. PMLR, 2018.

[11] Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In *Conference on Robot Learning*, pages 835–853. PMLR, 2020.

[12] Clément Romac, Rémy Portelas, Katja Hofmann, and Pierre-Yves Oudeyer. Teachmyagent: a benchmark for automatic curriculum learning in deep rl. In *International Conference on Machine Learning*, pages 9052–9063. PMLR, 2021.

[13] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J Pal, and Liam Paull. Active domain randomization. In *Conference on Robot Learning*, pages 1162–1176. PMLR, 2020.

[14] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *International Conference on Machine Learning*, pages 2817–2826. PMLR, 2017.

[15] Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. *Advances in Neural Information Processing Systems*, 33:13049–13061, 2020.

[16] Peide Huang, Mengdi Xu, Fei Fang, and Ding Zhao. Robust reinforcement learning as a stackelberg game via adaptively-regularized adversarial training. *arXiv preprint arXiv:2202.09514*, 2022.

[17] Adrien Baranes and Pierre-Yves Oudeyer. Intrinsically motivated goal exploration for active motor learning in robots: A case study. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1766–1773. IEEE, 2010.

[18] Pierre Fournier, Olivier Sigaud, Mohamed Chetouani, and Pierre-Yves Oudeyer. Accuracy-based curriculum learning in deep reinforcement learning. *arXiv preprint arXiv:1806.09614*, 2018.

[19] Pascal Klink, Haoyi Yang, Jan Peters, and Joni Pajarinen. Metrics matter: A closer look on self-paced reinforcement learning. 2021.

[20] Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. 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 5468–5479. PMLR, 13–18 Jul 2020.

[21] Haoxiang Wang, Bo Li, and Han Zhao. Understanding gradual domain adaptation: Improved analysis, optimal path and beyond. *arXiv preprint arXiv:2204.08200*, 2022.

[22] Judy Hoffman, Trevor Darrell, and Kate Saenko. Continuous manifold based adaptation for evolving visual domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 867–874, 2014.

[23] Michael Gadermayr, Dennis Eschweiler, Barbara Mara Klinkhammer, Peter Boor, and Dorit Merhof. Gradual domain adaptation for segmenting whole slide images showing pathological variability. In *International Conference on Image and Signal Processing*, pages 461–469. Springer, 2018.

[24] Markus Wulfmeier, Alex Bewley, and Ingmar Posner. Incremental adversarial domain adaptation for continually changing environments. In *2018 IEEE International conference on robotics and automation (ICRA)*, pages 4489–4495. IEEE, 2018.

[25] Hao Wang, Hao He, and Dina Katabi. Continuously indexed domain adaptation. *arXiv preprint arXiv:2007.01807*, 2020.

[26] Samira Abnar, Rianne van den Berg, Golnaz Ghiasi, Mostafa Dehghani, Nal Kalchbrenner, and Hanie Sedghi. Gradual domain adaptation in the wild: When intermediate distributions are absent. *arXiv preprint arXiv:2106.06080*, 2021.

[27] Hong-You Chen and Wei-Lun Chao. Gradual domain adaptation without indexed intermediate domains. *Advances in Neural Information Processing Systems*, 34, 2021.

[28] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

[29] Leonid V Kantorovich. On the translocation of masses. *Journal of mathematical sciences*, 133(4):1381–1382, 2006.

[30] Vivien Seguy and Marco Cuturi. Principal geodesic analysis for probability measures under the optimal transport metric. *Advances in Neural Information Processing Systems*, 28, 2015.

[31] Martial Agueh and Guillaume Carlier. Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*, 43(2):904–924, 2011.

[32] Robert J McCann. A convexity principle for interacting gases. *Advances in mathematics*, 128(1):153–179, 1997.

[33] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

[34] Hicham Janati, Marco Cuturi, and Alexandre Gramfort. Debiased sinkhorn barycenters. In *International Conference on Machine Learning*, pages 4692–4701. PMLR, 2020.

[35] Norm Ferns, Prakash Panangaden, and Doina Precup. Bisimulation metrics for continuous markov decision processes. *SIAM Journal on Computing*, 40(6):1662–1714, 2011.

[36] Norman Ferns and Doina Precup. Bisimulation metrics are optimal value functions. In *UAI*, pages 210–219. Citeseer, 2014.

[37] Pablo Samuel Castro. Scalable methods for computing state similarity in deterministic markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10069–10076, 2020.

[38] Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020.

[39] Philippe Hansen-Estruch, Amy Zhang, Ashvin Nair, Patrick Yin, and Sergey Levine. Bisimulation makes analogies in goal-conditioned reinforcement learning. *arXiv preprint arXiv:2204.13060*, 2022.

[40] Robert Dadashi, Shideh Rezaeifar, Nino Vieillard, Léonard Hussenot, Olivier Pietquin, and Matthieu Geist. Offline reinforcement learning with pseudometric learning. In *International Conference on Machine Learning*, pages 2307–2318. PMLR, 2021.

[41] Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient flows: in metric spaces and in the space of probability measures*. Springer Science & Business Media, 2005.

[42] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[43] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[44] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[45] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[46] Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite markov decision processes. In *UAI*, volume 4, pages 162–169, 2004.

[47] Yuxin Deng and Wenjie Du. The kantorovich metric in computer science: A brief survey. *Electronic Notes in Theoretical Computer Science*, 253(3):73–82, 2009.

[48] Xinfeng Zhou. *A Practical Guide to Quantitative Finance Interviews*. Lulu. com, 2008.

[49] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Proof of Theorem 4.1

In this section, we shall provide the proof for Theorem 4.1. The rest of this section is organized as follows. Appendix A.1 provides additional useful notations and definitions including but not limited to CMDPs, value function, and distance metrics. Appendix A.2 introduces further assumptions, and A.3 introduces the preliminary of optimal transport. Then, we provide the proof pipeline of Theorem 4.1 in Appendix A.4 and postpone the auxiliary proof to Appendix A.5. Finally, we summarize all the useful notations in Table 1.

A.1 Additional notation and definitions used in the proof

Before starting, let's introduce some additional notations useful throughout the theoretical analysis. For any discrete set X, we will denote the set of probability measures on X by $\Delta(X)$. For any vector $x \in \mathbb{R}^{SA}$ (resp. $x \in \mathbb{R}^{S}$ or $x \in \mathbb{R}^{C}$) that constitutes certain values for each state-action pair (resp. state or context), we use $x(s,a)$ (resp. $x(s)$ or $x(c)$) to denote the entry associated with the (s,a) pair (resp state s or context c). We denote $\mathsf{supp}(\rho)$ as the support of any distribution ρ.

Target CMDPs. Throughout the proof, we shall focus on the set of CMDPs introduced in Assumption 4.1. Moreover, without loss of generality, let the state space $\mathcal{S} = \{1, 2, \cdots, S\}$ (resp. action space $\mathcal{A} = \{1, 2, \cdots, A\}$) to be with finite cardinality S (resp. A). Without loss of generality, let the immediate reward $R(s,a,s) \in [0, 1-\gamma]$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$.

First, we introduce the following *occupancy distributions* associated with policy π and any CMDP with the initial state distribution $c \sim \rho$:

$$o^{\pi}(s;\rho) \coloneqq (1-\gamma)\sum_{t=0}^{\infty}\gamma^{t}\mathbb{P}\big(s_t = s \,|\, c \sim \rho; \pi\big), \tag{8a}$$

$$o^{\pi}(s,a;\rho) \coloneqq (1-\gamma)\sum_{t=0}^{\infty}\gamma^{t}\mathbb{P}\big(s_t = s, a_t = a \,|\, c \sim \rho; \pi\big) = o^{\pi}(s;\rho)\pi(a\,|\,s). \tag{8b}$$

With this in mind, we introduce two important definitions in the proof,: the covering state-action (resp. state) space by executing π with the initial state distribution $c \sim \rho$ defined as

$$\mathcal{C}(\rho,\pi) = \{(s,a) : o^{\pi}(s,a;\rho) > 0\}, \qquad \mathcal{C}_s(\rho,\pi) = \{s : o^{\pi}(s;\rho) > 0\}. \tag{9}$$

The covering space represent the area that is possible to be visited by the policy π given the initial state distribution.

Value function. Different from the value function defined by context distribution ρ, with abuse of notation, we denote the value function V_{s} associated with a state s, a policy π and the target CMDPs as

$$V_{\mathsf{s}}^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^{t}R(s_t, a_t, c') \;\Big|\; s_0 = s, c = s; \pi\right]. \tag{10}$$

For convenience, for any $s \in \mathcal{S}$, we define $\rho_s(c) \coloneqq \mathbb{1}(c = s)$. Recalling that the target MDPs are defined by $p_0(s\,|\,c) = \mathbb{1}(s = c)$, it is easily verified that for any policy π, the value function satisfies the following properties:

$$\forall s \in \mathcal{S}: \qquad 0 \le V_{\mathsf{s}}^{\pi}(s) = V^{\pi}(\rho_s) \le 1, \tag{11a}$$

$$\forall \rho \in \Delta(\mathcal{C}) \in \Delta(\mathcal{S}): \qquad V_{\mathsf{s}}^{\pi}(\rho) = V^{\pi}(\rho). \tag{11b}$$

Distance metrics. Furthermore, for the specified target CMDPs, plugging in $p_0(s\,|\,c) = \mathbb{1}(s = c)$, for any two contexts $c_i, c_j \in \mathcal{C}$, the π**-contextual-distance** metric defined in context space $\mathcal{C}$ obeys

$$d^{\pi}(c_i, c_j) = \mathbb{E}_{s_i \sim p_0(\cdot|c_i), s_j \sim p_0(\cdot|c_j)}\left[d^{\pi}_{c_i,c_j}(s_i, s_j)\right] = d^{\pi}_{s_i,s_j}(s_i, s_j), \tag{12}$$

where s_i (resp. s_j) is the corresponding initial state when the context are c_i (resp. c_j). With this observation, we highlight that for the target CMDPs, the **contextual π-bisimulation** metric degrades to **π-bisimulation** $d_{\mathsf{s}}^{\pi}(\cdot,\cdot)$ [37] such that:

$$\forall s_i, s_j \in \mathcal{S}: \qquad d_{\mathsf{s}}^{\pi}(s_i, s_j) \coloneqq d_{s_i,s_j}^{\pi}(s_i, s_j) = d^{\pi}(s_i, s_j). \tag{13}$$

The above result directly leads to that for any $\rho, \rho' \in \Delta(\mathcal{C}) \subseteq \Delta(\mathcal{S})$,

$$\mathcal{W}_{d^{\pi}}(\rho, \rho') = \mathcal{W}_{d_{\mathsf{s}}^{\pi}}(\rho, \rho'). \tag{14}$$

A.2 Additional Assumptions in the proof

Besides the key properties of the target CMDPs introduced in Assumption 4.1, we shall introduce some auxiliary assumptions for convenience as follows:

- **Bounded environment.** For the optimal policy $\pi^{\star}$ and any state $s \in \mathcal{S}$, without loss of generality, the minimum and maximum **π-bisimulation** distance between s and other states satisfies

$$\min_{s' \in \mathcal{S}} d_{\mathsf{s}}^{\star}(s, s')\, \mathbb{1}(d_{\mathsf{s}}^{\star}(s, s') > 0) \geq 1, \qquad \frac{D_{\mathsf{max}}}{2} \leq \max_{s' \in \mathcal{S}} d_{\mathsf{s}}^{\star}(s, s') \leq D_{\mathsf{max}}, \tag{15}$$

  for some universal positive constant D_{max}.
- **Deterministic environment.** Without loss of generality, we suppose the transition kernel is deterministic. Moreover, at any state $s \in \mathsf{supp}(\rho_{k+1}) \setminus \mathsf{supp}(\rho_k)$, there are a fixed portion of actions lead to the decreasing of the distance $\min_{s' \in \rho_k} d_{\mathsf{s}}^{\star}(s', s)$, i.e., for any transition sample (s_t, a_t, s_{t+1}),

$$\min_{s' \in \rho_k} d_{\mathsf{s}}^{\star}(s', s_{t+1}) = \begin{cases} \min_{s' \in \rho_k} d_{\mathsf{s}}^{\star}(s', s_t) - 1 & \text{with probability p} \\ \min_{s' \in \rho_k} d_{\mathsf{s}}^{\star}(s', s_t) + 1 & \text{otherwise.} \end{cases} \tag{16}$$

- **Closeness of the subsequent stages.** Starting from any state $s_0 = t \in \mathsf{supp}(\rho_{k+1}) \setminus \mathsf{supp}(\rho_k)$, if the sample trajectory shall enter $\mathcal{C}_{\mathsf{s}}(\rho_k, \pi_k^{\star})$ before arriving at somewhere s obeying $d_{\mathsf{s}}^{\star}(s, t) = \max_{s \in \mathcal{S}} d_{\mathsf{s}}^{\star}(s, t)$. Then we suppose that the hitting point s is not far from the initial distribution ρ_k of the previous stage k, namely

$$\mathbb{E}_s[d_{\mathsf{s}}^{\star}(s, t)] \leq C_1 \min_{s' \in \rho_k} d_{\mathsf{s}}^{\star}(s', t), \tag{17}$$

  for some universal constant $C_1 > 0$.

We would like to note that these additional assumptions are useful for a concise proof to show the main intuition and idea of our proposed algorithm **GRADIENT**. These are some general assumptions without specific limitations. It is interesting to extend our main Theorem 4.1 to more general cases which shall be the further work.

A.3 Preliminaries: optimal transport and random walk

In this subsection, we introduce several key properties/facts of optimal transport, value function, and random walk, which play a crucial role in the proof of Theorem 4.1. The proofs for this subsection are deferred to Appendix A.5.

Preliminaries of optimal transport. We first introduce an important fact of the optimization problem in optimal transport. Suppose there exists two discrete sets $\mathcal{X}, \mathcal{Y}$ (possibly different). As it is well-known, for any two distributions $\mu \in \Delta(\mathcal{X}), \nu \in \Delta(\mathcal{Y})$, 1-Wasserstein distance with any metric d between μ and ν can be expressed as the optimal solution of the following linear programming (Kantorovich duality) problem:

$$\begin{aligned} &\max_{u,v} \sum_{x \in \mathcal{X}} u(x)\mu(x) - \sum_{y \in \mathcal{Y}} v(y)\nu(y), \\ &\text{subject to} \quad \forall x \in \mathcal{X}, y \in \mathcal{Y}: \; u(x) - v(y) \leq d(x, y), \\ &\qquad\qquad\quad 0 \leq u \leq 1, 0 \leq v \leq 1. \end{aligned} \tag{18}$$

Note that, when $\mathcal{X} = \mathcal{Y} = \mathcal{S}$, the resulting claim in (18) is equivalent to [37, 46, 47].

Controlling performance gap. Inspired by the above problem formulation, the difference of the value function conditioned on two different states/state distribution can be controlled in the following lemma:

Lemma A.1 *For any $s, t \in \mathcal{S}$ and any policy π, we have*

$$|V_{\mathsf{s}}^{\pi}(s) - V_{\mathsf{s}}^{\pi}(t)| \leq d_{\mathsf{s}}^{\pi}(s,t). \tag{19}$$

In addition, for any two context distribution $\rho_1, \rho_2 \in \Delta(\mathcal{C}) \in \Delta(\mathcal{S})$, the performance gap associated with any policy π obeys

$$|V_{\mathsf{s}}^{\pi}(\rho_1) - V_{\mathsf{s}}^{\pi}(\rho_2)| \leq \mathcal{W}_{d^{\pi}}(\rho_1, \rho_2). \tag{20}$$

This fact connects the performance gap in RL and the Wasserstein distance between different state distribution/contextual distribution, taking advantage of the Kantorovich duality form of the optimal transport problem.

Definition and properties of random walk. Finally, we describe the following useful lemma which is essential in proving the main part of Theorem 4.1 when the starting state is not in the covering area of the optimal policy of the previous training stage

Lemma A.2 *The process $\{S_n : n \geq 1\}$ is called a random walk if $\{X_i : i \geq 1\}$ are iid Bernoulli distribution with $p < 1$ and $S_n = \sum_{i=1}^{n} X_i$. Starting from 0, the expectation of the* **stopping time** *N of hitting any $d > 0$ or $-D_{\mathsf{max}}$ and the probability p_α of hitting $d > 0$ obeys:*

$$\mathbb{E}[N] \leq \max\left\{ D_{\mathsf{max}} d \,,\, \sqrt{\frac{D_{\mathsf{max}}}{(1-2p)\wedge 1}} d \right\}, \quad p_\alpha \geq 1 - \frac{2d\sqrt{D_{\mathsf{max}}}}{d + D_{\mathsf{max}}}. \tag{21}$$

A.4 Proof of Theorem 4.1

With above preliminaries in hand, we are ready to embark on the proof for Theorem 4.1, which is divided into multiple steps as follows.

Step 1: decomposing the performance gap of interest. To begin with, we decompose the term of interest as follows:

$$\begin{aligned} V^{\pi_{k+1}^\star}(\rho_{k+1}) - V^{\pi_k^\star}(\rho_{k+1}) &\overset{\text{(i)}}{=} V_{\mathsf{s}}^{\pi_{k+1}^\star}(\rho_{k+1}) - V_{\mathsf{s}}^{\pi_k^\star}(\rho_{k+1}) \\ &\overset{\text{(ii)}}{=} V_{\mathsf{s}}^{\pi^\star}(\rho_{k+1}) - V_{\mathsf{s}}^{\pi_k^\star}(\rho_{k+1}) \\ &= V_{\mathsf{s}}^{\pi^\star}(\rho_{k+1}) - V_{\mathsf{s}}^{\pi^\star}(\rho_k) + V_{\mathsf{s}}^{\pi^\star}(\rho_k) - V_{\mathsf{s}}^{\pi_k^\star}(\rho_{k+1}) \\ &\overset{\text{(iii)}}{\leq} \mathcal{W}_{d^\star}(\rho_{k+1}, \rho_k) + V_{\mathsf{s}}^{\pi_k^\star}(\rho_k) - V_{\mathsf{s}}^{\pi_k^\star}(\rho_{k+1}), \end{aligned} \tag{22}$$

where (i) holds by the equivalence verified in (11b), (ii) follows from the value function of the optimal policy $\pi^\star$ (covering the entire state-action space) is the same as that of $\pi_{k+1}^\star$ in the region $\mathcal{C}(\rho_{k+1}, \pi_{k+1}^\star)$, and (iii) comes from applying Lemma A.1 and the fact that $\pi^\star$ achieves the same value function as $\pi_k^\star$ in the region $\mathcal{C}(\rho_k, \pi_k^\star)$.

Step 2: controlling the second term in (22) **in two cases.** To proceed, it is observed that

$$\forall (s,t) \in \mathcal{C}_s(\rho_k, \pi_k^\star) \times \rho_{k+1} : \qquad V_{\mathsf{s}}^{\pi_k^\star}(s) - V_{\mathsf{s}}^{\pi_k^\star}(t) \leq d_{\mathsf{s}}^\star(s,t) \tag{23}$$

can directly leads to that $(u, v) = \left(V_{\mathsf{s}}^{\pi_k^\star}, V_{\mathsf{s}}^{\pi_k^\star}\right)$ is a feasible solution to the Wasserstein distance $\mathcal{W}_{d^\star}(\rho_k, \rho_{k+1})$ dual formulation in (18), which yields

$$\begin{aligned} V_{\mathsf{s}}^{\pi_k^\star}(\rho_k) - V_{\mathsf{s}}^{\pi_k^\star}(\rho_{k+1}) &= \sum_{s \in \mathcal{C}_s(\rho_k, \pi_k^\star)} V_{\mathsf{s}}^{\pi_k^\star}(s)\rho_k(s) - \sum_{s \in \mathcal{C}_s(\rho_{k+1}, \pi_k^\star)} V_{\mathsf{s}}^{\pi_k^\star}(s)\rho_{k+1}(s) \\ &\leq \mathcal{W}_{d_{\mathsf{s}}^\star}(\rho_k, \rho_{k+1}). \end{aligned} \tag{24}$$

As a result, we turn to show (23) instead of controlling $V_{\mathsf{s}}^{\pi_k^\star}(\rho_k) - V_{\mathsf{s}}^{\pi_k^\star}(\rho_{k+1})$. Towards this, we start from considering $V_{\mathsf{s}}^{\pi_k^\star}(s) - V_{\mathsf{s}}^{\pi_k^\star}(t)$ for any $s \in \rho_k$ and $t \in \rho_{k+1}$ in two different cases: (i) $t \in \mathcal{C}_{\mathsf{s}}(\rho_k, \pi_k^\star)$; (ii) otherwise.

In the first case when $t \in \mathcal{C}_{\mathsf{s}}(\rho_k, \pi_k^\star)$, since $\pi_k^\star$ is the same as the optimal policy $\pi^\star$ in $\mathcal{C}_{\mathsf{s}}(\rho_k, \pi_k^\star)$, invoking Lemma A.1 directly yields

$$V_{\mathsf{s}}^{\pi_k^\star}(s) - V_{\mathsf{s}}^{\pi_k^\star}(t) = V_{\mathsf{s}}^{\pi^\star}(s) - V_{\mathsf{s}}^{\pi^\star}(t) \leq d_{\mathsf{s}}^\star(s,t). \tag{25}$$

Step 3: focusing on case (ii). Then, we shall focus on the other case when

$$t \in \mathsf{supp}(\rho_{k+1}) \setminus \mathcal{C}_{\mathsf{s}}(\rho_k, \pi_k^\star). \tag{26}$$

Invoking Lemma A.2, without loss of generality, since in the stage k, we can't visit the region outside of $\mathcal{C}_{\mathsf{s}}(\rho_k, \pi_k^\star)$, we can define $\pi_k^\star(\cdot \mid s)$ to be uniformly random in the unseen region, i.e.,

$$\forall a \in \mathcal{A}, s \in \mathsf{supp}(\rho_{k+1}) \setminus \mathcal{C}_{\mathsf{s}}(\rho_k, \pi_k^\star): \qquad \pi_k^\star(a \mid s) = \frac{1}{A}. \tag{27}$$

Then we define p_{g} as the probability of the benign events B when we can hit some point s_{b} at the boundary of the region $\mathcal{C}_{\mathsf{s}}(\rho_k, \pi_k^\star)$ visited by the previous stage k before arriving at some limit point $s_{\mathsf{limit}} := \arg\max_{s' \in \mathcal{S}} d_{\mathsf{s}}^\star(t, s')$. Invoking the assumption of the target CMDPs in (16), combind with the policy defined in (27), we observe that applying Lemma A.2 leads to

$$p_{\mathsf{g}} \geq 1 - \frac{2d_{\mathsf{s}}^\star(t, s_{\mathsf{b}})\sqrt{D_{\mathsf{max}}}}{d_{\mathsf{s}}^\star(t, s_{\mathsf{b}}) + D_{\mathsf{max}}} \geq 1 - \frac{2d_{\mathsf{s}}^\star(t, s_{\mathsf{b}})}{\sqrt{D_{\mathsf{max}}}} \geq 1 - \frac{2C_1 d_{\mathsf{s}}^\star(t, s)}{\sqrt{D_{\mathsf{max}}}}, \tag{28}$$

where the last inequality holds by the assumption in (15).

To continue, we express the value function at state t as

$$V_{\mathsf{s}}^{\pi_k^\star}(t) \geq p_{\mathsf{g}} \left(\mathbb{E}_B[\sum_{t=0}^{N} 0 \cdot \gamma^t] + \gamma^N V_{\mathsf{s}}^{\pi_k^\star}(s_{\mathsf{b}}) \right) - (1 - p_{\mathsf{g}}) = \gamma^N p_{\mathsf{g}} V_{\mathsf{s}}^{\pi_k^\star}(s_{\mathsf{b}}) - (1 - p_{\mathsf{g}}). \tag{29}$$

Similarly, we have

$$V_{\mathsf{s}}^{\pi_k^\star}(s) \leq (1 - p_{\mathsf{g}}) + p_{\mathsf{g}} \mathbb{E}[\sum_{t=0}^{N} (1-\gamma)\gamma^t + \gamma^N V_{\mathsf{s}}^{\pi_k^\star}(s_N)]. \tag{30}$$

We shall control the performance gap in two cases separately:

- When $V_{\mathsf{s}}^{\pi_k^\star}(s_{\mathsf{b}}) \geq V_{\mathsf{s}}^{\pi_k^\star}(s_N)$. we have

$$\begin{aligned} V_{\mathsf{s}}^{\pi_k^\star}(s) - V_{\mathsf{s}}^{\pi_k^\star}(t) &\leq V_{\mathsf{s}}^{\pi_k^\star}(s) - \gamma^N p_{\mathsf{g}} V_{\mathsf{s}}^{\pi_k^\star}(s') + (1 - p_{\mathsf{g}}) \\ &\leq p_{\mathsf{g}} \mathbb{E}[\sum_{t=0}^{N} (1-\gamma)\gamma^t + \gamma^N V_{\mathsf{s}}^{\pi_k^\star}(s_N)] - \gamma^N p_{\mathsf{g}} V_{\mathsf{s}}^{\pi_k^\star}(s_{\mathsf{b}}) + 2(1 - p_{\mathsf{g}}) \\ &\leq p_{\mathsf{g}} \mathbb{E}[\sum_{t=0}^{N} (1-\gamma)\gamma^t] + 2(1 - p_{\mathsf{g}}). \end{aligned} \tag{31}$$

- When $V_{\mathsf{s}}^{\pi_k^\star}(s_{\mathsf{b}}) < V_{\mathsf{s}}^{\pi_k^\star}(s_N)$, we have

$$
\begin{aligned}
V_{\mathsf{s}}^{\pi_k^\star}(s) - V_{\mathsf{s}}^{\pi_k^\star}(t) &\le p_{\mathsf{g}}\mathbb{E}[\sum_{t=0}^{N}(1-\gamma)\gamma^t + \gamma^N V_{\mathsf{s}}^{\pi_k^\star}(s_N)] - \gamma^N p_{\mathsf{g}} V_{\mathsf{s}}^{\pi_k^\star}(s_{\mathsf{b}}) + 2(1-p_{\mathsf{g}}) \\
&\le p_{\mathsf{g}}\mathbb{E}[\sum_{t=0}^{N}(1-\gamma)\gamma^t + \gamma^N (N + V_{\mathsf{s}}^{\pi_k^\star}(s))] - \gamma^N p_{\mathsf{g}} V_{\mathsf{s}}^{\pi_k^\star}(s_{\mathsf{b}}) + 2(1-p_{\mathsf{g}}) \\
&\le p_{\mathsf{g}}\mathbb{E}\left[\sum_{t=0}^{N}(1-\gamma)\gamma^t + \gamma^N\big(N + V_{\mathsf{s}}^{\pi_k^\star}(s_{\mathsf{b}}) + (C_1+1)d^\star(s,t)\big)\right] \\
&\qquad - \gamma^N p_{\mathsf{g}} V_{\mathsf{s}}^{\pi_k^\star}(s_{\mathsf{b}}) + 2(1-p_{\mathsf{g}}) \\
&\le p_{\mathsf{g}}\left[\sum_{t=0}^{N}(1-\gamma)\gamma^t + \gamma^N\big(N + (C_1+1)d^\star(s,t)\big)\right] + 2(1-p_{\mathsf{g}}). \qquad (32)
\end{aligned}
$$

Summing up the above two cases yields, for all $s \in \mathcal{C}(\rho_k, \pi_k^\star), t \in \mathsf{supp}(\rho_{k+1}) \setminus \mathcal{C}(\rho_k, \pi_k^\star)$,

$$
\begin{aligned}
&V_{\mathsf{s}}^{\pi_k^\star}(s) - V_{\mathsf{s}}^{\pi_k^\star}(t) \\
&\le p_{\mathsf{g}}\left[\sum_{t=0}^{N}(1-\gamma)\gamma^t + \gamma^N\big(N + (C_1+1)d^\star(s,t)\big)\right] + 2(1-p_{\mathsf{g}}) \\
&\le p_{\mathsf{g}}\mathbb{E}_B[N] + p_{\mathsf{g}}(C_1+1)d^\star(s,t) + 2(1-p_{\mathsf{g}}) \\
&\overset{\text{(i)}}{\le} \max\left\{D_{\mathsf{max}} d^\star(s,t)\ ,\ \sqrt{\frac{D_{\mathsf{max}}}{(1-2p)\wedge 1}}d^\star(s,t)\right\} + (C_1+1)d^\star(s,t) + \frac{4d^\star(s,t)}{\sqrt{D_{\mathsf{max}}}} \\
&\le \left(\max\left\{D_{\mathsf{max}}\ ,\ \sqrt{\frac{D_{\mathsf{max}}}{(1-2p)\wedge 1}}\right\} + (C_1+1) + \frac{4}{\sqrt{D_{\mathsf{max}}}}\right) d^\star(s,t) \\
&\le c_{\mathsf{model}} d^\star(s,t) \qquad (33)
\end{aligned}
$$

where (i) follows from Lemma A.2, and the last inequality holds by choosing

$$
c_{\mathsf{model}} := \left(\max\left\{D_{\mathsf{max}}\ ,\ \sqrt{\frac{D_{\mathsf{max}}}{(1-2p)\wedge 1}}\right\} + (C_1+1) + \frac{4}{\sqrt{D_{\mathsf{max}}}}\right)
$$

as a large enough problem-dependent parameter.

Step 4: summing up the results. Combining the results (25) and (33) in two cases, we directly arrive at, for all $s \in \mathcal{C}(\rho_k, \pi_k^\star), t \in \mathsf{supp}(\rho_{k+1})$, the following fact is satisfied

$$
\begin{aligned}
&V_{\mathsf{s}}^{\pi_k^\star}(s) - V_{\mathsf{s}}^{\pi_k^\star}(t) \le c_{\mathsf{model}} d^\star(s,t), \\
&\to \frac{1}{c_{\mathsf{model}}} V_{\mathsf{s}}^{\pi_k^\star}(s) - \frac{1}{c_{\mathsf{model}}} V_{\mathsf{s}}^{\pi_k^\star}(t) \le d^\star(s,t). \qquad (34)
\end{aligned}
$$

We observe that $(u, v) = \left(\frac{1}{c_{\mathsf{model}}} V_{\mathsf{s}}^{\pi_k^\star}, \frac{1}{c_{\mathsf{model}}} V_{\mathsf{s}}^{\pi_k^\star}\right)$ is a feasible solution to the Wasserstein distance $\mathcal{W}_{d_{\mathsf{s}}^\star}(\rho_k, \rho_{k+1})$ dual formulation in (18), we achieve

$$
V_{\mathsf{s}}^{\pi_k^\star}(\rho_k) - V_{\mathsf{s}}^{\pi_k^\star}(\rho_{k+1}) \le c_{\mathsf{model}} \mathcal{W}_{d_{\mathsf{s}}^\star}(\rho_k, \rho_{k+1}). \qquad (35)
$$

Finally, plugging (35) into (22) complete the proof by

$$
V^{\pi_{k+1}^\star}(\rho_{k+1}) - V^{\pi_k^\star}(\rho_{k+1}) \le (1 + c_{\mathsf{model}}) \mathcal{W}_{d^\star}(\rho_k, \rho_{k+1}). \qquad (36)
$$

A.5 Proof of auxiliary lemmas

Proof of Lemma A.1. As it is well known that $V_{\mathsf{s}}^{\pi}(s)$ is the fixed point of the Bellman operator $\mathcal{T}(V, \pi) := \mathbb{E}_{a\sim\pi(\cdot\,|\,s)}[R(s,a,s) + \gamma \sum_{s'} P(s' \,|\, s,a,s)V]$. In addition, d_{s}^{π} is the fixed point of the

operator [37]

$$\mathcal{F}(d)(s,t) \coloneqq \left|\mathbb{E}_{a\sim\pi(\cdot\,|\,s)}[R(s,a,s) - \mathbb{E}_{a\sim\pi(\cdot\,|\,t)}[R(t,a,t)\right| \\ + \gamma \mathcal{W}_d\left(\mathbb{E}_{a\sim\pi(\cdot\,|\,t)}[P(\cdot\,|\,s,a,s)], \mathbb{E}_{a\sim\pi(\cdot\,|\,t)}[P(\cdot\,|\,t,a)]\right). \tag{37}$$

Armed with above facts, initializing $V^\pi_{\mathsf{s},0} = 0$ and $d^\pi_{\mathsf{s},0} = 0$, the update rules of $V^\pi_{\mathsf{s},n+1}$ and $d^\pi_{\mathsf{s},n+1}$ at the $(n+1)$-th iteration are defined as

$$\begin{aligned} V^\pi_{\mathsf{s},n+1} &= \mathcal{T}(V^\pi_{\mathsf{s},n}, \pi), \\ d^\pi_{\mathsf{s},n+1} &= \mathcal{F}(d^\pi_{\mathsf{s},n})(s,t). \end{aligned} \tag{38}$$

With this in mind, for any $s, t \in \mathcal{S}$, we arrive at,

$$\begin{aligned} &\left|V^\pi_{\mathsf{s},n+1}(s) - V^\pi_{\mathsf{s},n+1}(t)\right| \\ &= \Bigg|\mathbb{E}_{a\sim\pi(\cdot\,|\,s)}\left[R(s,a,s) + \gamma\sum_{s'} P(s'\,|\,s,a,s)V^\pi_{\mathsf{s},n}\right] \\ &\qquad - \mathbb{E}_{a\sim\pi(\cdot\,|\,t)}\left[R(t,a,t) + \gamma\sum_{s'} P(s'\,|\,t,a,t)V^\pi_{\mathsf{s},n}\right]\Bigg| \\ &\le \left|\mathbb{E}_{a\sim\pi(\cdot\,|\,s)}R(s,a,s) - \mathbb{E}_{a\sim\pi(\cdot\,|\,t)}R(t,a,t)\right| + \gamma\left|\sum_{s'} P(s'\,|\,s,a,s)V^\pi_{\mathsf{s},n} - \sum_{s'} P(s'\,|\,t,a,t)V^\pi_{\mathsf{s},n}\right| \\ &\overset{\text{(i)}}{\le} \left|\mathbb{E}_{a\sim\pi(\cdot\,|\,s)}R(s,a,s) - \mathbb{E}_{a\sim\pi(\cdot\,|\,t)}R(t,a,t)\right| \\ &\qquad + \gamma\mathcal{W}_{d^\pi_{\mathsf{s},n}}\left(\mathbb{E}_{a\sim\pi(\cdot\,|\,s)}[P(s'\,|\,s,a,s)], \mathbb{E}_{a\sim\pi(\cdot\,|\,t)}[P(s'\,|\,t,a,t)]\right) \\ &= \mathcal{F}(d^\pi_{\mathsf{s},n})(s,t) = d^\pi_{\mathsf{s},n+1}, \end{aligned} \tag{39}$$

where (i) holds by $(u,v) = \left(V^\pi_{\mathsf{s},n}, V^\pi_{\mathsf{s},n}\right)$ is a feasible solution to the Wasserstein distance $\mathcal{W}_{d^\pi_{\mathsf{s},n}}\left(\mathbb{E}_{a\sim\pi(\cdot\,|\,s)}[P(s'\,|\,s,a,s)], \mathbb{E}_{a\sim\pi(\cdot\,|\,t)}[P(s'\,|\,t,a,t)]\right)$ dual formulation in (18).

As a result, we have

$$\forall n = 1, 2, \cdots: \qquad \left|V^\pi_{\mathsf{s},n+1}(s) - V^\pi_{\mathsf{s},n+1}(t)\right| \le d^\pi_{\mathsf{s},n+1}, \tag{40}$$

which directly yields

$$\forall s, t \in \mathcal{S}: \qquad |V^\pi_{\mathsf{s}}(s) - V^\pi_{\mathsf{s}}(t)| \le d^\pi_{\mathsf{s}}(s,t). \tag{41}$$

The above fact indicates that for any two distribution $\rho, \rho' \in \Delta(\mathcal{S})$,

$$V^\pi_{\mathsf{s}}(\rho_k) - V^\pi_{\mathsf{s}}(\rho_{k+1}) = \sum_{s\in\mathcal{S}} V^\pi_{\mathsf{s}}(s)\rho(s) - \sum_{s\in\mathcal{S}} V^\pi_{\mathsf{s}}(s)\rho'(s) \le \mathcal{W}_{d^\pi_{\mathsf{s}}}(\rho,\rho') = \mathcal{W}_{d^\pi}(\rho,\rho'), \tag{42}$$

where the penultimate inequality arises from $(u,v) = \left(V^\pi_{\mathsf{s}}, V^\pi_{\mathsf{s}}\right)$ is a feasible solution to the Wasserstein distance $\mathcal{W}_{d^\pi_{\mathsf{s}}}(\rho,\rho')$ dual formulation in (18), and the last equality holds by (14).

Proof of Lemma A.2 Before starting, we denote p_α as the probability that stopping at d and N as the stopping time. Then we consider the terms of interest in two cases. In the first case when $p = \frac{1}{2}$, invoking the facts in [48] directly leads to

$$N = D_{\mathsf{max}} d, \qquad p_\alpha = 1 - \frac{d}{d + D_{\mathsf{max}}}. \tag{43}$$

So the remainder of the proof will focus on the other case when $p \neq \frac{1}{2}$. Following the proof in [48], by basic calculus, it is easily verified that the following two centered terms are all martingales

$$S_n - (2p-1)n, \qquad S_n^2 + 2(1-2p)S_n + (2p-1)^2 n^2 + 4p(p-1)n. \tag{44}$$

In continue, it can also be verified that

$$\mathbb{E}[S_N - (2p-1)N] = p_\alpha d - (1-p_\alpha)D_{\mathsf{max}} = 0, \tag{45}$$

which leads to

$$p_\alpha = \frac{D_{\mathsf{max}} + (2p-1)N}{d + D_{\mathsf{max}}}. \tag{46}$$

Observing that

$$\mathbb{E}[S_n^2 + 2(1-2p)S_n + (2p-1)^2 n^2 + 4p(p-1)n] = 0 \tag{47}$$

yields

$$(2p-1)^2 N^2 + [(D_{\mathsf{max}} - d)(1-2p) - (sp-1)^2 - 1]N + D_{\mathsf{max}} d = 0, \tag{48}$$

and then implies

$$N \leq \sqrt{\frac{D_{\mathsf{max}} d}{1-2p}} \leq \sqrt{\frac{D_{\mathsf{max}}}{1-2p}} d. \tag{49}$$

Plugging in (49) into (46) leads to

$$p_\alpha \geq \frac{D_{\mathsf{max}} - \sqrt{D_{\mathsf{max}} d}}{d + D_{\mathsf{max}}} = 1 - \frac{\sqrt{D_{\mathsf{max}} d} + d}{d + D_{\mathsf{max}}} \geq 1 - \frac{2d\sqrt{D_{\mathsf{max}}}}{d + D_{\mathsf{max}}}, \tag{50}$$

which follows from (15) in Appendix A.2.

Finally, summing up the two cases, we arrive at

$$N \leq \max\left\{ D_{\mathsf{max}} d \,,\, \sqrt{\frac{D_{\mathsf{max}}}{(1-2p) \wedge 1}} d \right\}, p_\alpha \geq 1 - \frac{2d\sqrt{D_{\mathsf{max}}}}{d + D_{\mathsf{max}}}. \tag{51}$$

A.6 Table of notations

We summarize useful notations in the proof here.

Table 1: Notations

Symbol	Definition
$\Delta(X)$	the set of probability measures on X
$\mathsf{supp}(\rho)$	support of distribution ρ
$o^\pi(s;\rho)$	$(1-\gamma)\sum_{t=0}^{\infty} \gamma^t \mathbb{P}\big(s_t = s \,\vert\, c \sim \rho; \pi\big)$
$o^\pi(s,a;\rho)$	$o^\pi(s;\rho)\pi(a \,\vert\, s)$
$\mathcal{C}_s(\rho,\pi)$	$\{s : o^\pi(s;\rho) > 0\}$
$\mathcal{C}(\rho,\pi)$	$\{(s,a) : o^\pi(s,a;\rho) > 0\}$
$V_{\mathsf{s}}^\pi(s)$	$\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, c') \,\middle\vert\, s_0 = s, c = s; \pi\right]$
$d_{\mathsf{s}}^\star$	bisimulation distance $d_{\mathsf{s}}^{\pi^*}$ under the optimal policy π^*
$a \wedge b$	$\min(a,b)$

B Experiment Details

In this section, we discuss details that could not be included in the main paper due to space limitations. This includes hyperaparameters of the algorithms, additional details about the environment, and visualizations that assist the qualitative analysis. The experiments were conducted on a desktop computer with an Intel Core i7-8700K CPU @ 3.70GHz 12-Core Processor, an Nvidia RTX 2080Ti graphics card and 64GB of RAM.

Table 2: Observation, action, and context space in Maze

Dim.	Discrete Observation Space	range
0-120	Cell Type	$\{0 : \text{free}, 1 : \text{wall}, 2 : \text{agent}, 3 : \text{goal}\}$
Index	Discrete Action Space	
0	Go north	
1	Go south	
2	Go west	
3	Go east	
Index	Discrete Context Space	
0-50	Initial position	

B.1 Maze

We show the observation, action and context space in Table 2. The state, action and context space are discrete. For this toy-like example, we fix the maze layout throughout. The context is the initial position. The observation is the flattened value representation of the maze, including the goal, the current position of the agent, and the layout.

For this task, we use the PPO implementation in the `StableBaseline3` library [44]. We list the hyperparameters in the Table 3 (set to the default value if not mentioned in the table). We conduct hyperparameter grid search mainly on the following hyperparameters: $(learning_rate, ent_coef, n_steps) \in \{0.0001, 0.0003, 0.001\} \times \{0.001, 0.01, 0.1\} \times \{10, 100, 200\}$

Table 3: Hyperparameters in Maze

PPO Learner	value
gamma	0.99
learning_rate	0.0001
n_steps	100
ent_coef	0.1
total_timesteps	100000
GRADIENT hyperparameters	value
reward threshold $\bar{G}$	-15

We visualize the intermediate task distributions generated by GRADIENT in Figure 6 with a $\Delta\alpha = 0.05$. For $\Delta\alpha = 0.1$ and 0.2 are in the same figure as well (just with different gaps between two consecutive α). The π-contextual-distance is computed using the A* path finding algorithm. Although it is not always realistic to have access to the optimal policy, this toy example is just to show the effectiveness of an appropriate contextual distance metric. More choices of the contextual distance could be explored in the future work.

B.2 PointMass

In PointMass, the agent needs to navigate a pointmass through a wall with a small gap at an off-center position to reach the goal. The context is a 2-dimension vector representing the width and position of the gap on the wall. Following the setting in the original SPDL paper [6], the target distribution is an isotropic Gaussian distribution centered at $[2.5, 0.5]$ with a negligible variance $[4\times10^{-3}, 3.75\times10^{-3}]$ (which is effectively a point as shown in Figure 5a). The source distribution is an isotropic Gaussian distribution centered at $[0, 4.25]$ with a variance of $[2, 1.875]$. We show the observation, action and context space in Table 4.

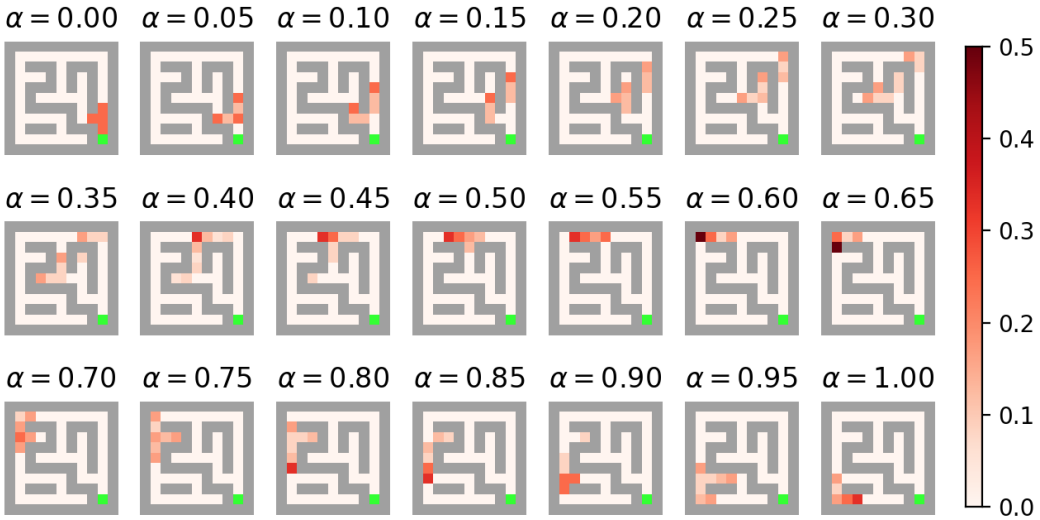


Figure 6: Intermediate task distributions generated by GRADIENT.

For the baseline implantation, we use and modify the code base provided in [6]. We also use the best hyperparameters found by [6]. For ALP-GMM, they conduct hyperparameter grid search over $(p_{\text{RAND}}, n_{\text{ROLLOUT}}, s_{\text{BUFFER}}) \in \{0.1, 0.2, 0.3\} \times \{50, 100, 200\} \times \{500, 1000, 2000\}$. For Goal-GAN, they conduct grid search over $(\delta_{\text{NOISE}}, n_{\text{ROLLOUT}}, p_{\text{SUCCESS}}) \in \{0.025, 0.05, 0.1\} \times \{50, 100, 200\} \times \{0.1, 0.2, 0.3\}$. Due to unknown issue, the scale of episodic reward we get (about 0-100) is different from what is shown in [6] (0-10), nevertheless the trend of the training curves is very similar. We visualize the context distribution of ALP-GMM and Goal-GAN in Figure 7. From this figure, we can reason why they underform GRADIENT and SPDL: since they cannot specify the target distribution, it is very difficult for them to learn to navigate through a narrow door at a specific position.

Table 4: Observation, action, and context space in the PointMass

Dim.	Continuous Observation Space	range
0	x position	$[-4, 4]$
1	x velocity	$[-\text{inf}, \text{inf}]$
2	y position	$[-4, 4]$
3	y velocity	$[-\text{inf}, \text{inf}]$
Dim.	Continuous Action Space	
0	x force	$[-10, 10]$
1	y force	$[-10, 10]$
Dim.	Continuous Context Space	
0	gate position	$[-4, 4]$
1	gate width	$[0.5, 8]$

B.3 FetchPush

In FetchPush [45], the objective is to use the gripper to push the box to a goal position. The observation space is a 28-dimension vector, including information about the goal. The context is a 2-dimension vector representing the goal position on a surface. The target distribution is a uniform distribution over the circumference of a half-circle (Figure 5b). The source distribution is a uniform distribution over a square region centered at the box position, excluding the region within a certain radius of the object. We use this experiment to highlight the importance of the capability to handle arbitrary distributions

Table 5: Hyperparameters in PointMass

SAC Learner	value
train_freq	5
buffer_size	10000
gamma	0.95
learning_rate	0.0003
learning_starts	500
batch_size	64
net architecture	[64, 64]
activation_fn	Tanh
SPDL hyperparameter	**value**
α offset	25 iterations
ζ	1.1
KL threshold	8000
maximum KL	0.05
steps per iteration	2048
ALP-GMM hyperparameter	**value**
random task ratio	0.2
fit rate (number of episodes between two fit of GMM)	200
max size (maximal number of episodes for computing ALP)	1000
GOAL-GAN hyperparameter	**value**
state noise level	0.05
fit rate (number of episodes between two fit of GAN)	25
probability to sample state with noise	0.05
GRADIENT hyperparameter	**value**
$\Delta\alpha$	0.1
Reward threshold $\bar{G}$	40

rather than only the parametric Gaussian distributions. Since SPDL can only deal with parametric Gaussian distribution, we first fit the target and the source distribution with two Gaussian distributions and feed them into the baseline algorithms. The source Gaussian is $[1.14655655, 0.74819359]$ with a variance array of $[[0.0141083, 0.00055327], [0.00055327, 0.0149638]]$, and the target Gaussian is $[1.33561676, 0.74819359]$ with a variance array of $[[8.89519060e-03, -1.34467507e-18], [-1.34467507e-18, 4.59090909e-02]]$. We visualize the context distribution of ALP-GMM and Goal-GAN in Figure 8

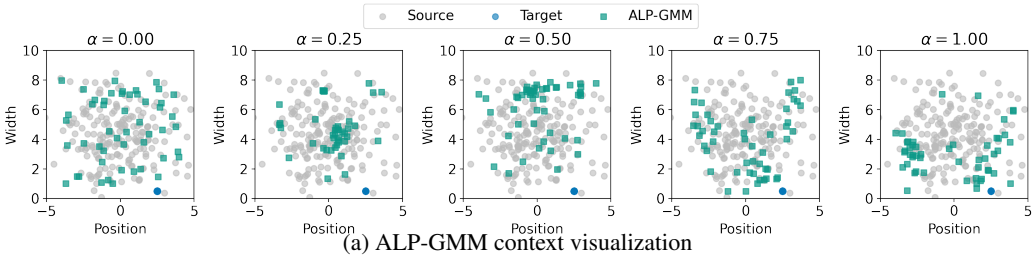
C Algorithm Details

C.1 Computation of π-contextual-distance from Samples

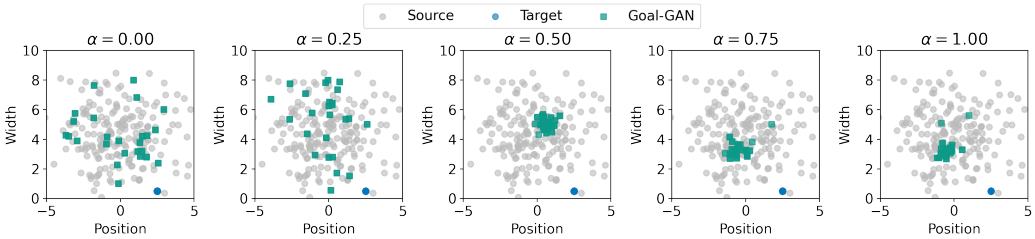
Computing contextual π-bisimulation from samples. We start from estimating the **contextual π-bisimulation metric**. We first define an iterative procedure for computing it from a data buffer $\mathcal{D}$. Let $d^{\pi}_{c_i,c_j,0} \equiv 0$ be the everywhere-zero metric. At step n, let $\tau = \{s_i, s_j, a_i, a_j, s'_i, s'_j, R_i, R_j, c_i, c_j\} \in \mathcal{T}$ be a sample from $\mathcal{D}$, define $d^{\pi}_{c_i,c_j,n}$ as:

$$
\begin{aligned}
d^{\pi}_{c_i,c_j,n}(s,t) &= d^{\pi}_{c_i,c_j,n-1}(s,t), \quad \forall s \neq s_i, t \neq s_j \\
d^{\pi}_{c_i,c_j,n}(s_i,s_j) &= \max \begin{bmatrix} d^{\pi}_{c_i,c_j,n-1}(s_i,s_j) \\ |R_i - R_j| + \gamma d^{\pi}_{c_i,c_j,n-1}\left(s'_i, s'_j\right) \end{bmatrix}
\end{aligned} \tag{52}
$$

Learning an approximation for contextual π-bisimulation. Let $\phi : \mathcal{S} \times C \mapsto \mathbb{R}^k$ be a k-dimensional represetation of the state and context space and let $\psi_\omega : \mathbb{R}^{2k} \mapsto \mathbb{R}$ be a neural

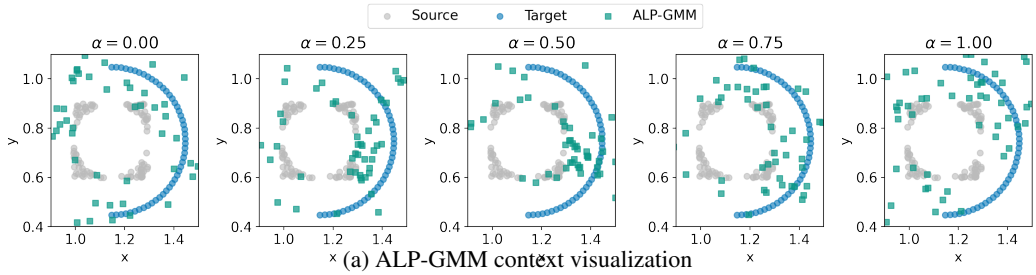


(a) ALP-GMM context visualization

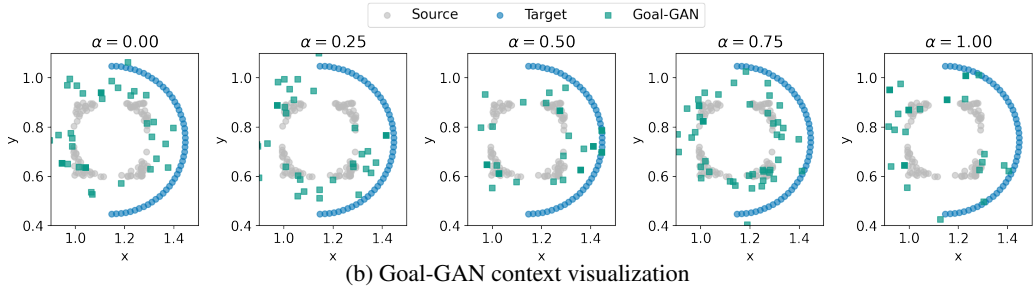


(b) Goal-GAN context visualization

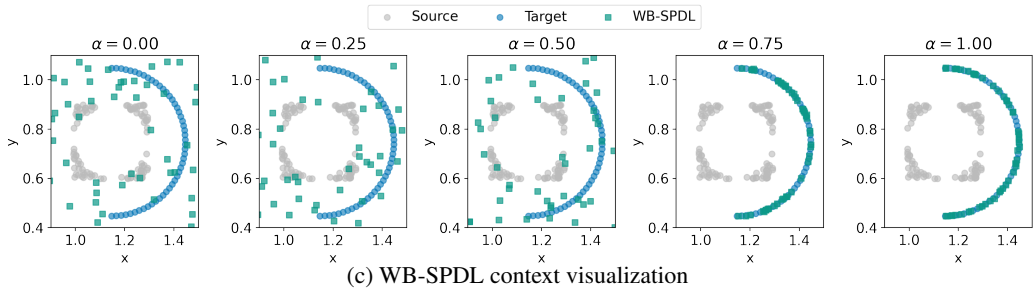
Figure 7: Visualizations of context distributions and curricula in PointMass. The contexts are taken from environment steps at 10k, 50k, 100k, 200k, and 300k.



(a) ALP-GMM context visualization



(b) Goal-GAN context visualization



(c) WB-SPDL context visualization

Figure 8: Visualizations of context distributions and curricula in FetchPush. The contexts are taken from environment steps at 5k, 50k, 100k, 250k, and 500k for ALP-GMM and Goal-GAN, 100, 500, 1000, 1200, 1500 for WB-SPDL.

Table 6: Observation, action, and context space in the FetchPush

Dim.	Continuous Observation Space	range
0-2	grip_pos	$[-\text{inf}, \text{inf}]$
3-5	object_pos	$[-\text{inf}, \text{inf}]$
6-8	object_rel_pos	$[-\text{inf}, \text{inf}]$
9-10	gripper_state	$[-\text{inf}, \text{inf}]$
11-13	object_rot	$[-\text{inf}, \text{inf}]$
14-16	object_velp	$[-\text{inf}, \text{inf}]$
17-19	object_velr	$[-\text{inf}, \text{inf}]$
20-22	grip_velp	$[-\text{inf}, \text{inf}]$
23-24	gripper_vel	$[-\text{inf}, \text{inf}]$
25-27	goal_pos	$[-\text{inf}, \text{inf}]$
Index	**Continuous Action Space**	
0-2	pos_ctrl	$[-1, 1]$
3	gripper_ctrl	$[-1, 1]$
Dim.	**Continuous Context Space**	
0	goal_x_pos	$[0.8, 1.5]$
1	goal_y_pos	$[0.4, 1.1]$

network parameterized by ω, that receives a concatenation of two state representations such that $\psi_\omega(\phi(s_i, c_i), \phi(s_j, c_j)) \approx d^\pi_{c_i,c_j}(s_i, s_j)$. Following the conventional practice in [2], we use online parameters ω and target parameters ω^-, where the online parameters ω are updated at each iteration while the target parameters are updated very multiple iterations. We define the target objective $\mathbf{T}_{\omega^-}(\tau)$ as:

$$\max \left[\begin{array}{c} \psi_{\omega^-}([\phi(s_i, c_i), \phi(s_j, c_j)]), \\ |\mathcal{R}_i - \mathcal{R}_j| + \gamma \psi_{\omega^-}([\phi(s'_i, c_i), \phi(s'_i, c_i)]) \end{array} \right] \tag{53}$$

Then we can define our loss $\mathcal{L}_{\omega^-}(\tau)$ as $\mathbb{E}\left(\mathbf{T}_{\omega^-}(\tau) - \psi_\omega(\phi((s_i, c_i), \phi(s_j, c_j))\right)^2$. This loss is defined for a single pair of transitions from the buffer, but the mini-batch version will be very similar. The computation can be carried out efficiently on GPUs.

To obtain π-contextual-distance, given a set of $s_i \in \mathbf{s}$ sampled from $p_0(\cdot \mid c_i)$ and $s_j \in \mathbf{t}$ from $p_0(\cdot \mid c_j)$,

$$d^\pi(c_i, c_j) = \sum_{s_i \in \mathbf{s}} \sum_{s_j \in \mathbf{t}} [\psi_\omega(\phi(s_i, c_i), \phi(s_j, c_j))] \tag{54}$$

Note that with approximation, these distance metric are not guaranteed to be a proper pseudometric: it can violate the identity of indiscernibles, symmetry, and subadditivity conditions.

C.2 Exact Computation of π-contextual-distance for Maze

For the maze, we compute the exact π-contextual-distance using dynamic programming. We iteratively compute the π-contextual-distance (represented by a matrix) until the maximum difference in metric estimate between successive iterations is smaller than a tolerance. Due the Assumption 4.1, the computation of π-contextual-distance is largely simplified (since the context space is essentially overlapped with the state space). We present the pseudocode as follows:

```
def compute_exact_metric(tolerance=0.001):
    """Compute the exact pi-contextual-distance metric up to
        the specified tolerance.
    Args:
    tolerance: float, maximum difference in metric estimate between
        successive iterations. Once this threshold is reached,
```

Table 7: Hyperparameters in FetchPush

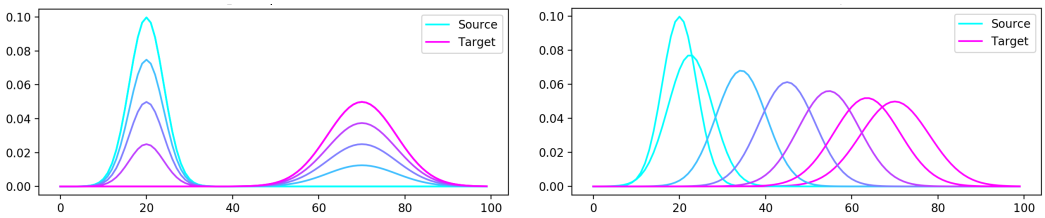
SAC Learner	value
train_freq	1
buffer_size	100000
gamma	0.99
learning_rate	0.001
learning_starts	1000
batch_size	256
net architecture	[64, 64, 64]
activation_fn	Tanh
SPDL hyperparameter	**value**
α offset	0 iterations
ζ	1.0
KL threshold	20
maximum KL	0.05
steps per iteration	5000
ALP-GMM hyperparameter	**value**
random task ratio	0.3
fit rate (number of episodes between two fit of GMM)	200
max size (maximal number of episodes for computing ALP)	1000
GOAL-GAN hyperparameter	**value**
state noise level	0.1
fit rate (number of episodes between two fit of GAN)	200
probability to sample state with noise	0.3
GRADIENT hyperparameter	**value**
$\Delta\alpha$	0.2
Reward threshold $\bar{G}$	-25

```
    computation stops.
"""
gamma = 0.99
policy = compute_optimal_policy()

n = get_context_size()
self.contextual_distance = np.zeros((n, n))
metric_difference = tolerance * 2.
i = 1
exact_metric_differences = []
while metric_difference > tolerance:
    new_metric = np.zeros((n, n))
    for s1 in range(n):
        for s2 in range(n):
            # get the action from policy
            action1 = policy[s1]
            action2 = policy[s2]
            # environment step
            next_state1, rew1 = step(s1, action1)
            next_state2, rew2 = step(s2, action2)
            # apply iterative operator
            new_metric[s1, s2] = (
```

```
                    abs(rew1 - rew2) +
                    gamma*self.contextual_distance[next_state1, next_state2])
        metric_difference = np.max(abs(new_metric - self.contextual_distance))
        exact_metric_differences.append(metric_difference)
        self.contextual_distance = np.copy(new_metric)
        i += 1

    # adding very small offset to the off-diagonal entries
    # to avoid computation issue
    offset = 0.1*np.ones(self.contextual_distance.shape)
    np.fill_diagonal(offset, 0)
    self.contextual_distance += offset

    # normalize to maximum value of 1
    self.contextual_distance =
            self.contextual_distance/np.max(self.contextual_distance)

    return self.contextual_distance
```

D Miscellaneous

1D Wasserstein Barycenter. We show an example comparing the linear interpolation with Wasserstein barycenter using Euclidean distance between two Gaussian distribution in Figure 9. We can observe that for the linear interpolation, the intermediate distributions assign very little probability mass on the region where there is low probability for the source and target. However, the Wasserstein barycentes assign appropriate mass to the middle region and forms a smooth transition. As shown in Figure 1, the Wasserstein barycenters generate much more reasonable intermediate task than the linear interpolation.



(a) Linear interpolation (b) Wasserstein interpolation

Figure 9: Comparison of interpolations between two Gaussian distributions. The source distribution is a Gaussian centered at 20, and the target distribution is a Gaussian centered at 70.

Feature summary of baseline methods. We compare four important features of the baseline algorithms with GRADIENT: whether the method can deal with continuous contexts and discrete contexts, whether the method considers a target task distribution, and whether the method can handle non-parametric distributions. Unsurprisingly, all the methods can handle continuous contexts, while only GRADIENT and Domain Randomization (DR) can handle discrete contexts. It is resulted from either the formulation or the computation feasibility. In addition, only SPDL and GRADIENT can specify the target task distribution, which is a important feature as illustrated by the PointMass experiment. The capability of handling both parametric and non-parametric distribution is also extremely important because it gives extra flexibility when the task task distribution cannot be accurately approximated by parametric distribution, namely, Gaussian, as shown in the FetchPush experiment.

Table 8: Comparison with existing methods.

	Continuous context	Discrete context	Target Task	Arbitrary distribution
DR [49]	✓	✓	✗	✓
ALP-GMM [11]	✓	✗	✗	✗
GoalGAN [10]	✓	✗	✗	✗
SPDL [6]	✓	✗	✓	✗
GRADIENT	✓	✓	✓	✓