Offline Imitation from Observation via Primal Wasserstein State Occupancy Matching

Kai Yan **Alexander G. Schwing** **Yu-Xiong Wang**
University of Illinois Urbana-Champaign
{kaiyan3, aschwing, yxw}@illinois.edu
https://github.com/KaiYan289/PW-DICE

Abstract

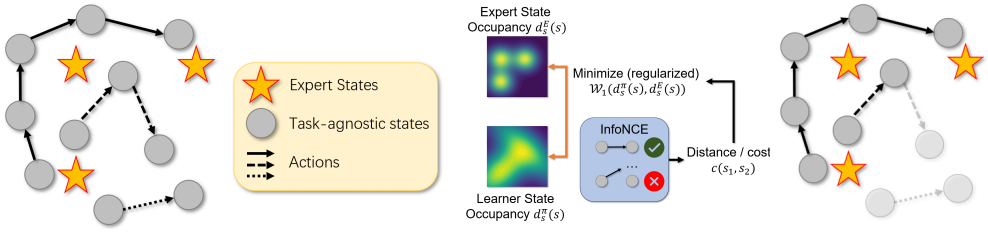
In real-world scenarios, arbitrary interactions with the environment can often be costly, and actions of expert demonstrations are not always available. To reduce the need for both, Offline Learning from Observations (LfO) is extensively studied, where the agent learns to solve a task with only expert states and *task-agnostic* non-expert state-action pairs. The state-of-the-art DIstribution Correction Estimation (DICE) methods minimize the state occupancy divergence between the learner and expert policies. However, they are limited to either f-divergences (KL and χ^2) or Wasserstein distance with Rubinstein duality, the latter of which constrains the underlying distance metric crucial to the performance of Wasserstein-based solutions. To address this problem, we propose Primal Wasserstein DICE (PW-DICE), which minimizes the primal Wasserstein distance between the expert and learner state occupancies with a pessimistic regularizer and leverages a contrastively learned distance as the underlying metric for the Wasserstein distance. Theoretically, we prove that our framework is a generalization of the state-of-the-art, SMODICE, and unifies f-divergence and Wasserstein minimization. Empirically, we find that PW-DICE improves upon several state-of-the-art methods on multiple testbeds.

1 Introduction

Recent years have witnessed remarkable advances in Offline Reinforcement Learning (RL) [8, 27, 26]: interaction data collected in the past is used to address sequential decision-making problems without online interaction, as it is often costly to conduct (e.g., autonomous driving [23]). Even without online interaction, methods achieve high sample efficiency. Such methods, however, require reward labels that are often missing when data is collected in the wild [6]. In addition, an informative reward is also expensive to obtain for many tasks, such as robotic manipulation, as it requires a carefully hand-crafted design [48]. To bypass the need for reward labels, offline Imitation Learning (IL) has prevailed recently [18, 19, 22]. It enables the agent to learn from existing demonstrations without reward labels. However, just like reward labels, expert demonstrations are also expensive and often in shortage, as they need to be recollected repeatedly for every task of interest. Among different types of expert data shortage, there is one widely studied type: *offline Learning from Observations (LfO).* In LfO, only the expert state, instead of both state and action, is recorded. This setting is useful when learning from experts with different embodiment [33] or from video demonstrations [7], where the expert action is either not applicable or not available.

Many methods have been proposed in the field of offline LfO, including inverse RL [52, 42, 24], similarity-based reward labeling [37, 7], and action pseudo-labeling[41, 28]. The state-of-the-art solution for LfO is the family of DIstribution Correction Estimation (DICE) methods, which are LobsDICE [20] and SMODICE [33]; both methods conduct convex optimization in the dual space to minimize the f-divergence of the state occupancy (visitation frequency) between the learner

Accepted to the Optimal Transport and Machine Learning Workshop at NeurIPS 2023.



(a) Problem Settings (b) Wasserstein Optimization (c) Weighted BC

Figure 1: An illustration of our method, PW-DICE. Different trajectories are illustrated by different styles of arrows. PW-DICE minimizes regularized 1-Wasserstein distance between learner's state occupancy $d_s^\pi(s)$ and expert state occupancy $d_s^E(s)$, and the underlying distance function is contrastively learned to represent the reachability between the states. With the matching result, Weights are calculated for downstream weighted Behavior Cloning (BC) to retrieve policy. High transparency indicates a small weight for the state and its corresponding action.

and the expert policies. However, DICE methods mostly focus on f-divergence [20, 33, 25, 22] (mainly KL-divergence and χ^2-divergence; see Appendix A for definition), a metric that ignores some underlying geometric properties of the distributions [39]. While there is a DICE work, SoftDICE [40], that introduces the Wasserstein distance into DICE methods, it adopts the Kantorovich-Rubinstein duality [3], which heavily limits the flexibility of the Wasserstein distances as duality requires the underlying metric to be Euclidean [39]. This limitation of the distance metric is not only theoretically infavorable, but also impacts practical performance. Concretely, we find the distance metric in Wasserstein-based methods to be crucial for performance (Sec. 3.1).

To solve the issue mentioned above, we propose Primal Wasserstein DICE (PW-DICE), a DICE method that optimizes the primal form of the Wasserstein distance, summarized in Fig. 1. With adequate regularizer for offline pessimism [21], the joint minimization of the Wasserstein matching variable and the learner policy can be eventually turned into a single-level convex optimization over the Lagrange space. The policy is then retrieved by weighted behavior cloning with weights determined by the Lagrange function. Different from SMODICE and LobsDICE, the underlying distance metric is arbitrary, and, different from all prior works, we explore the possibility of contrastively learning the metric from data. Our effort endows PW-DICE with much more flexibility; meanwhile, with specifically chosen hyperparameters, SMODICE can be seen as a special case of PW-DICE, which theoretically guarantees the performance of our solution.

We summarize our contributions as follows: 1) we propose a novel offline LfO method, PW-DICE, which uses the primal Wasserstein distance for LfO, gaining more flexibility regarding the distance metric than prior works, while removing the assumption for data coverage; 2) we theoretically prove that PW-DICE is a generalization of SMODICE, thus providing a unified framework for Wasserstein-based and f-divergence-based DICE methods; 3) we empirically show that our method achieves better results than the state of the art on multiple offline LfO testbeds.

2 Preliminaries

Markov Decision Process. The Markov Decision Process (MDP) is the widely adopted formulation for sequential decision-making problems. An MDP has five components: a state space S, an action space A, a transition function T, a reward r, and a discount factor γ. An MDP evolves in discrete steps; at step $t \in \{0, 1, 2, \dots\}$, state $s_t \in S$ is given, and the agent, according to its policy $\pi(a_t|s_t) \in \Delta(A)$ ($\Delta(A)$ is the probability simplex over A), choses an action $a_t \in A$. After receiving a_t, the MDP transits to a new state $s_{t+1} \in S$ with the transition probability function $T(s_{t+1}|s_t, a_t)$, and gives a reward $r(s_t, a_t) \in \mathbb{R}$ as feedback. The agent needs to maximize the discounted total reward $\sum_t \gamma^t r(s_t, a_t)$ with discount factor $\gamma \in [0, 1]$. A complete run of the MDP is defined as an episode, with the state(-action) pairs collected along the trajectory τ. The state occupancy, which is the visitation frequency of states given policy π, is $d_s^\pi(s) = (1-\gamma)\sum_t \gamma^t Pr(s_t = s)$. See Appendix A in the Appendix for more rigorous definitions of the state and other occupancies.

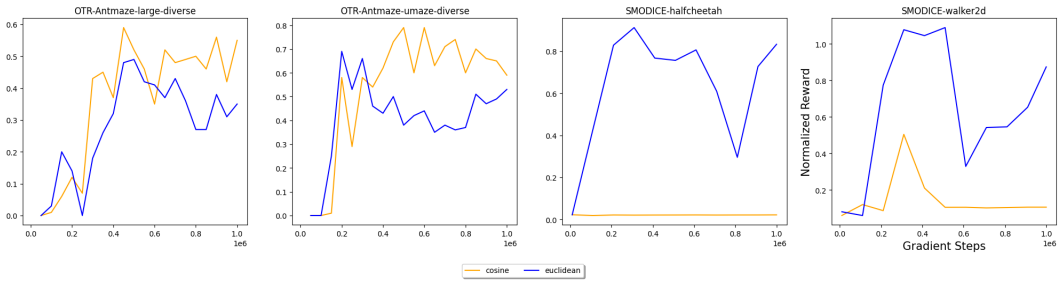


Figure 2: Performance comparison between OTR [32] with default distance metric and Euclidean distance metric on OTR (two leftmost) and SMODICE [33] (two rightmost) settings. The result shows that the underlying distance metric is crucial to the performance of Wasserstein-based method.

Offline Imitation Learning from Observations (LfO). In offline LfO, the agent needs to learn from two sources of data: the *expert* dataset E with state-only trajectories $\tau_E = \{s_1, s_2, \ldots, s_{n_1}\}$ that solves the exact target task, and the *task-agnostic* non-expert dataset I consisting of less relevant state-action trajectories $\tau_I = \{(s_1, a_1), (s_2, a_2), \ldots, (s_{n_2}, a_{n_2})\}$. Ideally, the agent learns the environment dynamics from I, and tries to follow the expert states in E with information about the MDP inferred from I. The state-of-the-art methods in offline LfO are SMODICE [33] and LobsDICE [20]. The two methods are in spirit similar, with the former minimizing state occupancy divergence and the latter optimizing adjacent *state-pair* occupancy divergence.

Wasserstein Distance. The Wasserstein distance, also known as Earth Mover's Distance (EMD) [3], is widely used as the distance between two probability distributions. It captures the geometry of the underlying space better and does not require any intersection between the support sets. For two distributions $p \in \Delta(S), q \in \Delta(S)$ over state space S, the Wasserstein[1] distance with underlying metric $c(x, y) : S \times S \to \mathbb{R}$ can be written as $\mathcal{W}(p, q) = \inf_{\Pi \in S \times S} \int_{x \in S} \int_{y \in S} \Pi(x, y) c(x, y)$, which is the *primal form* of the Wasserstein distance. Wasserstein also has an equivalent Kantorovich-Rubinstein dual form [3], which is $\mathcal{W}(p, q) = \max_{\|f\|_L \leq 1} \mathbb{E}_{x \sim p} f(x) - \mathbb{E}_{y \sim q} f(y)$, where $\|f\|_L \leq 1$ means that the function f is 1-Lipschitz. While this form is simpler and more often adopted by the machine learning community, the Lipschitz constraint is usually practically implemented by a gradient regularizer; and as the gradient is defined using a Euclidean distance, the underlying distance metric for Rubinstein duality is also restricted to Euclidean [39], which is often suboptimal.

3 Methodology

3.1 Motivation and Overview

As mentioned in Sec. 1, our goal is to improve the idea of divergence minimization between the expert's and the learner's policies by introducing the primal Wasserstein distance with arbitrary underlying distance metric. To better show the impotrance of distance metrics and the advantage of being able to select them, we study Optimal Transport Reward (OTR) [32], a current Wasserstein-based IL method that can be applied to our LfO setting. OTR optimizes the primal Wasserstein distance between every trajectory in the task-agnostic dataset and the expert trajectory, and uses the result to assign a reward to each state in the task-agnostic dataset; then, offline RL is applied to retrieve the optimal policy. Fig. 2 shows results of OTR on the D4RL mujoco dataset (see Sec. 4.2 for more) with testbeds appearing in both SMODICE [33] and OTR; we test both the cosine-similarity-based occupancy used in the paper and Euclidean distance as the underlying distance metric. The result illustrates that different distance metrics have a significant impact on results; thus, choosing a good metric is crucial for the performance of Wasserstein-based solutions.

[1] Unless otherwise specified, we only consider 1-Wasserstein distance in this paper.

Optimizing the primal Wasserstein distance between the state occupancies, the objective of our PW-DICE can be written as

$$\min_{\Pi,\pi} \sum_{s_i \in S} \sum_{s_j \in S} \Pi(s_i, s_j) c(s_i, s_j), \text{ s.t. } \forall s \in S, d_s^\pi(s) = (1-\gamma)p_0(s) + \gamma \sum_{\bar{s},\bar{a}} d_{sa}^\pi(\bar{s},\bar{a}) p(s|\bar{s},\bar{a}); d_{sa}^\pi \geq 0;$$
$$\forall s_j \in S, \sum_i \Pi(s_i, s_j) = d_s^E(s_j); \forall s_i \in S, \sum_j \Pi(s_i, s_j) = d_s^\pi(s_i); \Pi \geq 0. \tag{1}$$

In Eq. 1, we use $\Pi(s_i, s_j)$ as the matching variable between two state distributions, and $c(s_i, s_j)$ is the distance between s_i and s_j. d_s^E is the state occupancy of the expert policy, d_s^π is the state occupancy induced by policy π, and the state-action occupancy is d_{sa}^π. $p_0 \in \Delta(S)$ is the initial state distribution. There are two types of constraints in Eq. 1: the first row is the marginal constraint for the matching variable Π, and the second row is the *Bellman flow constraints* [33] that ensures correspondence between occupancy d_s^π and a feasible policy π.

For a tabular MDP, Eq. 1 can be solved by any Linear Programming (LP) solver, as both the objective and the constraints are linear; however, such solution is impractical for any MDP with continuous state or action space. Thus, we will add a pessimistic regularizer in Sec. 3.2 to Eq. 1, with which the Lagrange dual of the problem is unconstrained. We derive the closed-form solution and retrieve policy in Sec. 3.3, and discuss the distance metric selection in Sec. 3.4. See Tab. 2 in Appendix F for reference of notations.

3.2 Lagrange Dual of the Regularized Objective

For simplicity of derivation, we rewrite our main objective in Eq. 1 as a LP problem over a single vector $x = \begin{bmatrix} \Pi \\ d_{sa}^\pi \end{bmatrix} \in \mathbb{R}^{|S|\times(|S|+|A|)}$, where $\Pi \in \mathbb{R}^{|S|\times|S|}$ and $d_{sa}^\pi \in \mathbb{R}^{|S|\times|A|}$ are flattened by row-first manner. Correspondingly, we extend the cost function from c to $c' : (|S| \times (|S| + |A|)) \times (|S| \times (|S| + |A|)) \to \mathbb{R}$, such that $c' = c$ on the original domain of c and $c' = 0$ otherwise. Further, we summarize all linear equality constraints as $Ax = b$. Then, we get the simplified version of Eq. 1:

$$\min_{x \geq 0} (c')^T x, \text{ s.t. } Ax = b. \tag{2}$$

It is easy to see that the Lagrange dual form of Eq. 2 is also a constrained optimization. In order to remove the constraints in the dual, we modify the objective as follows:

$$\min_x (c')^T x + \epsilon_1 D_f(\Pi \| U) + \epsilon_2 D_f(d_{sa}^\pi \| d_{sa}^I), \text{ s.t. } Ax = b, x \geq 0, \tag{3}$$

where $U(s, s') = d_s^E(s) d_s^I(s')$, i.e., U is the product of two independent distributions d_s^E and d_s^I. $\epsilon_1 > 0, \epsilon_2 > 0$ are hyperparameters, and D_f can be any f-divergence. Note though f-divergence is used, unlike SMODICE [33] or LobsDICE [20], such formulation does not require data coverage of the task-agnostic data over expert data. The two regularizers we add are "pessimistic" and encourages the agents to stay within the support set of the dataset, which is common in offline IL/RL [21].

With the regularized objective in Eq. 3, we now consider its Lagrange dual form:

$$\max_\lambda \min_{x \geq 0} L(\lambda, x) = (c')^T x + \epsilon_1 D_f(\Pi \| U) + \epsilon_2 D_f(d_{sa}^\pi \| d_{sa}^I) - \lambda^T (Ax - b). \tag{4}$$

3.3 Convertion Into Single-Level Convex Optimization

While Eq. 4 is unconstrained, it is a bi-level optimization; to obtain a practical and stable solution, a single-level optimization is preferred. To do so, one could consider using the KKT condition [4], and set the derivative of the inner-level optimization to 0; however, such approach will lead to an exp function in the objective [36, 20], and thus is numerically unstable. To avoid this, we first rewrite Eq. 4 *with negated* $L(\lambda, x)$ to separate Π and d_{sa}^π in x:

$$\min_{\lambda}\{\epsilon_1 \max_{\Pi\in\Delta(S^2)}[\frac{(A_1^T\lambda - c)}{\epsilon_1}^T \Pi - D_f(\pi\|U)] + \epsilon_2 \max_{d^\pi_{sa}\in\Delta(S\cdot A)}[\frac{(A_2^T\lambda)}{\epsilon_2}^T d^\pi_{sa} - D_f(d^\pi_{sa}\|d^I_{sa})] - b^T\lambda\}. \quad (5)$$

In Eq. 5, we have $A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$, where $A_1 \in \mathbb{R}^{(|S|\times|S|)\times M}$, $A_2 \in \mathbb{R}^{(|S|\times|A|)\times M}$, and $M = 3|S|$ is the number of equality constraints in the primal form. There are two things worth noting in Eq. 5. First, we append two extra constraints, which are $\Pi \in \Delta, d^\pi_{sa} \in \Delta$. Such appended constraints does not affect final results because of the following fact:

Lemma 1. *For any MDP and feasible expert policy π^E, the inequality constraints in Eq. 1 with $\Pi \geq 0, d^\pi_{sa} \geq 0$ and $\Pi \in \Delta, d^\pi_{sa} \in \Delta$ are equivalent.*

The detailed proof of Lemma 1 is given in the Appendix B.3; in a word, the optimal solution of Eq. 4, as long as it satisfies all constraints in the primal form, must have $\Pi \in \Delta, d^\pi_{sa} \in \Delta$. Second, we decompose the max operator into two independent maximizations, as the equality constraints that correlate Π and d^π_{sa} are all relaxed in the dual, and $b^T\lambda$ is independent from the maximization.

With Eq. 5, we now apply the following theorem from SMODICE [33]:

Theorem 1. *With mild assumptions [12], for any f-divergence D_f, probability distribution p, q on domain $\mathcal{X}$ and function $y : \mathcal{X} \to \mathbb{R}$, we have*

$$\max_{p\in\Delta(\mathcal{X})} \mathbb{E}_{x\sim p}[y(x)] - D_f(p\|q) = \mathbb{E}_{x\sim q}[f_*(y(x))]. \quad (6)$$

Also, for maximizer $p^*(x) = \arg\max_{p\in\Delta(\mathcal{X})} \mathbb{E}_{x\sim q}[f_*(y(x))]$, we have $p^*(x) = q(x)f'_*(y(x))$, where $f_*(\cdot)$ is the Fenchel conjugate of f, and f'_* is its derivative.

The proof is out of scope of this work, and is discussed in the Appendix B.4. The rigorous notion of f-divergence and Fenchel conjugate are in the Appendix A. For this work, we mainly consider KL-divergence as D_f, which corresponds to $f(x) = x\log x$, and $f_*(x) = \text{logsumexp}(x)$ to be the Fenchel dual function with $x \in \Delta$ [4]. [2] With Thm. 1, we set $p = \Pi, x = \lambda, y(x) = \frac{A_1^T\lambda - c}{\epsilon_1}$ for the first max operator, and set $p = d^\pi_{sa}, x = \lambda, y(x) = \frac{A_2^T\lambda}{\epsilon_2}$ for the second max operator. Then, we get the following single-level convex objective:

$$\min_{\lambda} \epsilon_1 \log \mathbb{E}_{s_i\sim I, s_j\sim E} \exp(\frac{(A_1^T\lambda - c)^T}{\epsilon_1}) + \epsilon_2 \log \mathbb{E}_{(s_i,a_j)\sim I} \exp(\frac{A_2^T\lambda}{\epsilon_2}) - b^T\lambda. \quad (7)$$

Finally, by considering the elements in A (see Appendix B.2), we get our final objective

$$\begin{aligned} &\min_{\lambda} \epsilon_1 \log \mathbb{E}_{s_i\sim I, s_j\sim E} \exp(\frac{\lambda_{i+|S|} + \lambda_{j+2|S|} - c(s_i, s_j)}{\epsilon_1}) + \\ &\epsilon_2 \log \mathbb{E}_{(s_i,a_j)\sim I} \exp(\frac{-\gamma\mathbb{E}_{s_k\sim p(\cdot|s_i,a_j)}\lambda_k + \lambda_i - \lambda_{i+|S|}}{\epsilon_2}) - [(1-\gamma)\mathbb{E}_{s\sim p_0}\lambda_{:|S|} + \mathbb{E}_{s\sim E}\lambda_{2|S|:3|S|}], \end{aligned} \quad (8)$$

with the maximizer $d^\pi_{sa} = d^I_{sa} \cdot \text{softmax}(\frac{-\gamma\mathbb{E}_{s_k\sim p(\cdot|s_i,a_j)}\lambda_k + \lambda_i - \lambda_{i+|S|}}{\epsilon_2})$, and the denominator of the softmax is summing over all state-action pairs. Thus, we can now retrieve the desired policy π by weighted behavior cloning:

$$\begin{aligned} \mathbb{E}_{(s_i,a_j)\sim d^\pi_{sa}} \log p(a|s) &= \mathbb{E}_{(s_i,a_j)\sim I} \frac{d^\pi_{sa}(s_i, a_j)}{d^I_{sa}(s_i, a_j)} \log p(a_j|s_i) \\ &\propto \mathbb{E}_{(s_i,a_j)\sim I} \exp(\frac{-\gamma\mathbb{E}_{s_k}\lambda_k + \lambda_i - \lambda_{i+|S|}}{\epsilon_2}) \log p(a_j|s_i). \end{aligned} \quad (9)$$

In practice, we use 1-sample estimation for $p(\cdot|s_i, a_j)$, which is found in prior works to be simple and effective [33, 20]. That is, we sample $(s_i, a_j, s_k) \sim I$ from the dataset instead of (s_i, a_j), and

[2] χ^2-divergence does not work as well as KL-divergence in mujoco environment; see Appendix D.3 for details.

use λ_k corresponding to s_k as an estimation for $\mathbb{E}_{s_k \sim p(\cdot|s_i,a_j)}\lambda_k$. Since the number of states can be infinite in practice, we use a 3-head neural network to estimate λ_s, $\lambda_{s+|S|}$ and $\lambda_{s+2|S|}$ given state s.

Note, the formulation can be seen as a generalization of SMODICE [33]. More specifically, we have the following theorem (see Appendix B for proof):

Theorem 2. *If $c(s_i, s_j) = -\log \frac{d_s^E(s_i)}{d_s^I(s_i)}, \epsilon_2 = 1$, then as $\epsilon_1 \to 0$, Eq. 8 is equivalent to SMODICE objective with KL divergence.*

For different choice of D_f, similarly we have the following corollary:

Corollary 1. *If $c(s_i, s_j) = -\log \frac{d_s^E(s_i)}{d_s^I(s_i)}, \epsilon_2 = 1$, then as $\epsilon_1 \to 0$, Eq. 5 is equivalent to SMODICE with any f-divergence.*

Thus, our PW-DICE work is a unification of f-divergence and Wasserstein distance minimization.

3.4 Underlying Distance Metric

With Eq. 8 and 9, the only problem remaining is to choose the distance metric $c(s_i, s_j)$. For tabular cases, one could use the simplest distance, i.e., $c(s_i, s_j) = 1$ if $s_i \neq s_j$, and 0 otherwise. However, such design would lead to gradient disappearance in continuous case; to address this, prior works have explored many heuristic choices, such as cosine similarity [32] or Euclidean [40]. However, such heuristic choice is prone to different representations over the same state.

In this work, inspired by both CURL [29] and SMODICE [33], we propose a weighted sum of $R(s) = \log \frac{d_s^E(s)}{(1-\alpha)d_s^I(s)+\alpha d_s^E(s)}$ and the Euclidean distance between an embedding learned by the InfoNCE [43] loss. To be more specifically, we have

$$c(s_i, s_j) = R(s_i) + \beta \|f(s_i) - f(s_j)\|_2^2, \tag{10}$$

where $f(s_i), f(s_j)$ are embeddings for the states s_i, s_j, α is a positive constant close to 0, and $\beta \geq 0$ is a hyperparameter.

The distance function consists of two parts. The first part, $R(s_i)$, is an improved version of reward function $\log \frac{d_s^E(s)}{d_s^I(s)}$ in SMODICE [33]; intuitively, high $\log \frac{d_s^E(s)}{d_s^I(s)}$ indicates that the state s is more frequently visited by the expert than agents generating the task-agnostic data, which is probably desirable. Such reward can be obtained by training a discriminator $c(s)$ that takes expert states from E as label 1 and non-expert ones as label 0. If c is optimal, i.e., $c(s) = c^*(s) = \frac{d_s^E(s)}{d_s^E(s)+d_s^I(s)}$, then we have $\frac{d_s^E(s)}{d_s^I(s)} = \log \frac{c^*(s)}{1-c^*(s)}$. In our implementation, we change the denominator $d_s^E(s)$ to $(1-\alpha)d_s^I(s) + \alpha d_s^E(s)$ to avoid the theoretical assumption that the task-agnostic dataset I covers the expert dataset E, i.e., $d_s^I(s)$ must be positive wherever $d_s^E(s) > 0$.

The second part uses embedding $f(s)$ learned with infoNCE [43] following CURL [29], such that $f(s)$ and $f(s')$ are similar if and only if they can be reachable along trajectories in the task-agnostic dataset. More specifically, we use the following loss function:

$$L_c = \log \frac{\exp(q^T W k_+)}{\exp(q^T W k_+) + \sum_{k_-} \exp(q^T W k_-)}, \tag{11}$$

where q is the query (anchor), W is a learned, semi positive-definite weight matrix, k_+ is positive key, and k_- are negative keys. To train the embedding function f, for every gradient step, we sample a batch of adjacent state pairs $\{(s_i, s_i')|i \in \{1, 2, \ldots, K\}\}$; then, for $q = f(s_i)$, we set $k_+ = f(s_i')$ and the set of k_- to be $\{f(s_j')|j \neq i\}$; this essentially amounts to a K-way classification task, where for the i-th sample the correct label is i. Intuitively, the idea is to learn a good embedding space where the vicinity of state can be evaluated by the Euclidean distance between the embedding vectors. We define the vicinity as the "reachability" between states; that is, if one state can reach the other through a trajectory in the task-agnostic data, then they should be close; otherwise, they are far away. Such definition clusters states that lead to success together in the embedding space, while being robust to actual numerical values of the state.

4 Experiments

We evaluate PW-DICE in this section across multiple environments. There are two problems that we care about: 1) can the Wasserstein objective indeed leads to closer match between the learner's and expert's policy? (Sec. 4.1) 2) can PW-DICE work better than f-divergence based methods on more complicated environments, and does a flexible underlying distance metric indeed benefit (Sec. 4.2)?

4.1 Tabular Environments

Baselines. We compare to the two baselines closely discussed in the paper, which are SMODICE [33] and LobsDICE [20]. We test two variants of our method: Linear Programming (LP) that directly solves Eq. 1, and Regularizer (Reg) that solves Eq. 3 . As the environment is tabular, all methods are implemented with CVXPY [2] to get optimal numerical solutions. The mean and standard deviation data are from 10 independent runs with different seeds. We evaluate all methods with the **regret**, i.e., the gap between reward gained by learner policy and expert policy (*lower is better*). To be consistent with LobsDICE, We also compare the Total Variation (TV) distance between the state and state-pair occupancies, i.e., $\text{TV}(d_s^\pi \| d_s^E)$ and $\text{TV}(d_{ss}^\pi \| d_{ss}^E)$, in the Appendix D.1.

Environment Setup. Following random MDP experiment in LobsDICE [20], we randomly generate a MDP with $|S| = 20$ states, $|A| = 4$ actions and $\gamma = 0.95$. The stochasticity of the MDP is controlled by $\beta \in [0, 1]$, where $\beta = 0$ is deterministic and 1 is highly stochastic. Agent always start from one particular state, and tries to reach another particular state with reward $+1$, which is the only source of reward. We report the regret with different β, expert dataset size and task-agnostic dataset size. The only difference from LobsDICE's experiment is that the expert policy is deterministic instead of being softmax, as we found that due to the high connectivity of the MDP states, the value function for each state are close; thus, the softmax expert policy is highly suboptimal and near-uniform. See Appendix C for the reason and Appendix D.1 for the corresponding results.

Experimental Setup. As the environment is tabular, we use CVXPY [2] to solve the optimal policy for each method using the primal formulation; for example, we directly solve Eq. 1 to get the learner's policy π. Following SMODICE [33], for the estimation of transition function and task-agnostic average policy π^I, we simply count from the task-agnostic dataset I, i.e., the transition probability $p(s'|s,a) = \frac{\#[(s,a,s') \in I]}{\#[(s,a) \in I]}$, and $\pi^I(a|s) = \frac{\#[(s,a) \in I]}{\#[s \in I]}$ ($\#$ stands for "the number of"). Similarly, Expert state occupancy d_s^E is estimated by $d_s^E(s) = \frac{\#[s \in E]}{|E|}$, where $|E|$ is the size of the expert dataset E. Specially, if the denominator is 0, the distribution will be estimated as uniform.

Main Results. Fig. 3 shows the regret of each method. It is clearly shown that our method, with or without regularizer, performs similarly well and achieves the lowest regret across different expert dataset size, task-agnostic (non-expert) dataset size, and noise level. The gap increases with the task-agnostic dataset size, which shows that our method works better when the MDP dynamics are more accurately estimated. LobsDICE performs poorly in this scenario, albeit being the best in minimizing divergence with softmax expert (see Appendix D.1), as consistent with LobsDICE.

4.2 Mujoco Environments

Baselines. We adopt seven baselines in our comparisons: state-of-the-art DICE methods SMODICE [33], LobsDICE [20] and ReCOIL [38], non-DICE method ORIL [52], Wasserstein-based method OTR [32], DWBC [47] with extra access to the expert action, and the plain Behavior Cloning (BC). As we have no access to the ReCOIL code, we directly report the final numbers in their paper. The mean and standard deviation data are from 3 independent runs with different seeds. We measure the performance using the average reward (the higher the better).

Environment and Environmental Setup. Following SMODICE [33], we test PW-DICE on four standard OpenAI gym mujoco environments: hopper, halfcheetah, ant, and walker2d environment (see Appendix C for details). The metric we use is the normalized average reward [3], where higher reward indicates better performance; if the final reward is similar, the algorithm with fewer gradient step update is better. We plot the reward curve, which illustrates the change of the mean and standard deviation of the reward with the number of gradient steps. See Appendix C for hyperparameters.

[3] We use the same normalization standard as that of D4RL [15] and SMODICE [33].

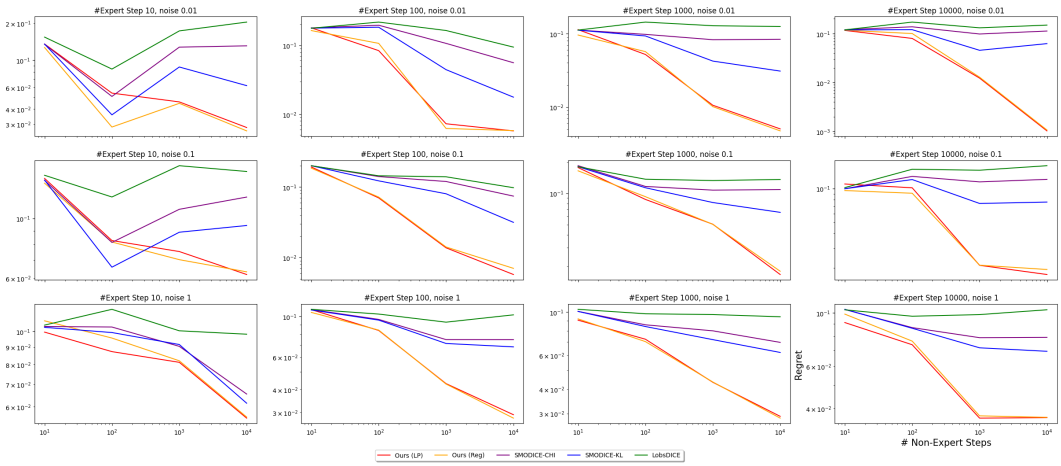


Figure 3: The regret (reward gap between learner and expert) of each method on tabular environment. It is clearly shown that our method, regardless of the presence of regularizer, works the best.

Main Results. Fig. 4 shows the result on the mujoco testbed, where our method achieves comparable or the best result on all four testbeds with the baselines. SMODICE with KL divergence and LobsDICE works decently well, while the other methods struggles under our setting.

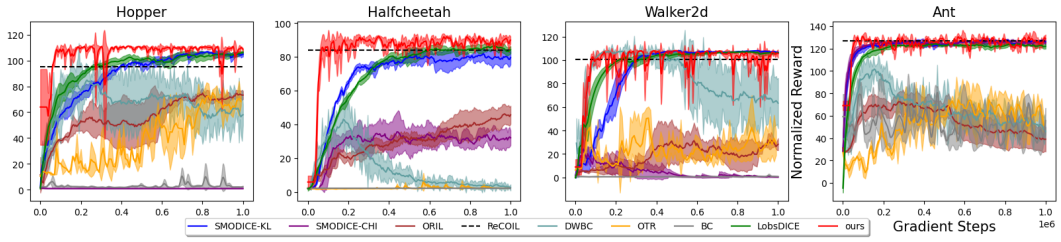


Figure 4: The performance comparison on the mujoco testbed; SMODICE-KL and SMODICE-CHI stands for variants of SMODICE using different f-divergences (KL or χ^2). Our method works the best among all methods.

Is our design of distance metric useful? To better show the importance and effectiveness of distance metric design, we conduct an ablation study on the distance metric used in PW-DICE; specifically, we test the result of PW-DICE with $c(s, s') = R(s)$, $c(s, s') = \|s - s'\|_2^2$ (Euclidean), $c(s, s') = 1 - \frac{s^T s'}{|s||s'|}$ (cosine similarity), $c(s, s')$ from contrastive learning and their combinations; the result is illustrated in Fig. 5. The result shows that both our design of distance and the combination of cosine similarity and $R(s)$ works well, while distance metrics with single component fails (including Euclidean distance implied by Rubinstein duality). We also conduct an ablation on the choice of ϵ_1 and ϵ_2 in Appendix D.2, showing that our method is generally robust to the hyperparameters.

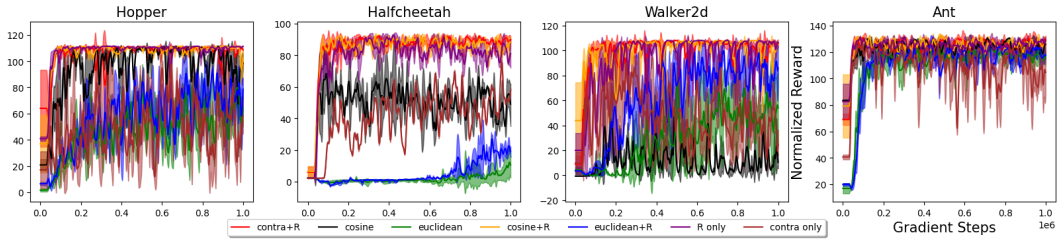


Figure 5: Ablations on the choice of distance metric. Our choice of $c(s, s')$, which combines contrastively learned distance and R is the best. Euclidean distance fails in our scenario, which further proves the importance of using the primal form instead of Rubinstein dual form.

5 Related Work

Wasserstein Distance for Imitation Learning. As a metric known to be capable of leveraging geometric properties of the distributions and give gradients to distributions with different support sets, Wasserstein distance (also known as *Optimal Transport*) [3] is a popular choice of distribution divergence minimization in recent years, and is widely used in IL/RL [1, 13, 45, 11, 16]. Among them, SoftDICE [40] is the most similar work to PW-DICE, which also optimizes Wasserstein distance under the DICE framework. However, SoftDICE and most Wasserstein-based IL algorithms use Rubinstein-Kantorovich duality [40, 45, 49, 31], which limits the underlying distance metric to Euclidean. There are a few methods optimizing primal Wasserstein distance: for example, OTR [32] computes primal Wasserstein distance between two trajectories and assigns reward accordingly for offline RL, and PWIL [11] uses greedy coupling to simplify the computation of Wasserstein distance. However, the former struggles in our experiment settings, and the latter only optimizes an upper bound of the Wasserstein distance. Our PW-DICE fixes both problems instead.

Offline Imitation Learning from Observation. Offline Learning from Observation (LfO) aims to learn from expert observations with no labeled action, which is useful in robotics where the expert action is either not available (e.g. video [35]) or not applicable (e.g. from a different embodiment [37]). Three major directions present in this area: 1) offline planning or RL with assigned, similarity-based reward [41, 28]; 2) minimization of occupancy divergence, which includes iterative inverse-RL methods [52, 46, 42] and DICE [33, 20, 22, 30, 51]; 3) action pseudolabeling, where the missing actions are predicted with an inverse dynamic model [37, 10, 44]. Our method, PW-DICE, falls in the second category but is a generalization and improvement over the existing methods.

Contrastive Learning for State Representations. Contrastive learning, such as InfoNCE [43] and SIMCLR [9], aims to find a good representation that satisfies similarity and dissimilarity constraints between particular pairs of data points. Such method is widely used in reinforcement learning, especially with visual input [29, 35, 37] and for meta RL [14] to improve the generalizability of the agent and mitigate the curse of dimensionality; in such works, similarity constraints can come from different augmentations of the same state [29, 35], multiview alignment [37], consistency after reconstruction [50], or task contexts [14]. PW-DICE tries to use contrastive learning to find a good distance metric considering state reachability, while still adopting the reward from the DICE works.

6 Conclusion

In this paper, we propose PW-DICE, a DICE method that uses the primal form of the Wasserstein distance with contrastively learned objective. By adding adequate pessimistic regularizer, we conduct an unconstrained convex optimization in the Lagrange dual space, and retrieve policy using weighted behavior cloning with weights determined by the Lagrange function. Our method is a generalization of SMODICE, unifies f-divergence and Wasserstein minimization, and gets better performance than multiple baselines, such as SMODICE [33] and LobsDICE [20] in multiple environments.

References

[1] R. Agarwal, M. C. Machado, P. S. Castro, and M. G. Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. In *ICLR*, 2021.

[2] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter. Differentiable convex optimization layers. In *NeurIPS*, 2019.

[3] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *ICML*, 2017.

[4] S. Boyd and L. Vandenberghe. *Convex optimization*. 2004.

[5] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.

[6] M. Chang, A. Gupta, and S. Gupta. Semantic visual navigation by watching youtube videos. In *NeurIPS*, 2020.

[7] A. S. Chen, S. Nair, and C. Finn. Learning generalizable robotic reward functions from "in-the-wild" human videos. *ArXiv:2103.16817*, 2021.

[8] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *ArXiv:2106.01345*, 2021.

[9] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. *NeurIPS*, 2020.

[10] X. Chen, S. Li, H. Li, S. Jiang, Y. Qi, and L. Song. Generative adversarial user model for reinforcement learning based recommendation system. In *ICML*, 2019.

[11] R. Dadashi, L. Hussenot, M. Geist, and O. Pietquin. Primal wasserstein imitation learning. In *ICLR*, 2021.

[12] B. Dai, N. He, Y. Pan, B. Boots, and L. Song. Learning from Conditional Distributions via Dual Embeddings. In *AISTATS*, 2017.

[13] A. Fickinger, S. Cohen, S. Russell, and B. Amos. Cross-domain imitation learning via optimal transport. In *10th International Conference on Learning Representations, ICLR*, 2022.

[14] H. Fu, H. Tang, J. Hao, C. Chen, X. Feng, D. Li, and W. Liu. Towards effective context for meta-reinforcement learning: an approach based on contrastive learning. In *AAAI*, 2020.

[15] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4rl: Datasets for deep data-driven reinforcement learning. *ArXiv:2004.07219*, 2020.

[16] D. Garg, S. Chakraborty, C. Cundy, J. Song, and S. Ermon. Iq-learn: Inverse soft-q learning for imitation. *NeurIPS*, 2021.

[17] S. Ghasemipour, R. Zemel, and S. Gu. A divergence minimization perspective on imitation learning methods. In *CoRL*, 2019.

[18] K. Hakhamaneshi, R. Zhao, A. Zhan, P. Abbeel, and M. Laskin. Hierarchical few-shot imitation with skill transition models. In *ICLR*, 2022.

[19] J. Ho and S. Ermon. Generative adversarial imitation learning. In *NIPS*, 2016.

[20] G. hyeong Kim, J. Lee, Y. Jang, H. Yang, and K. Kim. Lobsdice: Offline learning from observation via stationary distribution correction estimation. In *NeurIPS*, 2022.

[21] Y. Jin, Z. Yang, and Z. Wang. Is pessimism provably efficient for offline rl? In *ICML*, 2021.

[22] G. Kim, S. Seo, J. Lee, W. Jeon, H. Hwang, H. Yang, and K. Kim. Demodice: Offline imitation learning with supplementary imperfect demonstrations. In *ICLR*, 2022.

[23] B. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. Sallab, S. Yogamani, and P. Perez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[24] I. Kostrikov, K. K. Agrawal, D. Dwibedi, S. Levine, and J. Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. In *ICLR*, 2019.

[25] I. Kostrikov, O. Nachum, and J. Tompson. Imitation learning via off-policy distribution matching. In *ICLR*, 2020.

[26] I. Kostrikov, A. Nair, and S. Levine. Conservative q-learning for offline reinforcement learning. In *ICLR*, 2022.

[27] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit q-learning. In *ICLR*, 2022.

[28] A. Kumar, S. Gupta, and J. Malik. Learning navigation subroutines from egocentric videos. In *CoRL*, 2019.

[29] M. Laskin, A. Srinivas, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *ICML*. PMLR, 2020.

[30] J. Lee, W. Jeon, B.-J. Lee, J. Pineau, and K.-E. Kim. Optidice: Offline policy optimization via stationary distribution correction estimation. In *ICML*, 2021.

[31] F. Liu, Z. Ling, T. Mu, and H. Su. State alignment-based imitation learning. In *ICLR*, 2020.

[32] Y. Luo, Z. Jiang, S. Cohen, E. Grefenstette, and M. P. Deisenroth. Optimal transport for offline imitation learning. In *ICLR*, 2023.

[33] Y. J. Ma, A. Shen, D. Jayaraman, and O. Bastani. Smodice: Versatile offline imitation learning via state occupancy matching. In *ICML*, 2022.

[34] O. Nachum, Y. Chow, B. Dai, and L. Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. In *NeurIPS*, 2019.

[35] J. Pari, N. M. M. Shafiullah, S. P. Arunachalam, and L. Pinto. The surprising effectiveness of representation learning for visual imitation. *ArXiv:2112.01511*, 2021.

[36] Y. Polyanskiy. f-divergences, 2020. URL https://people.lids.mit.edu/yp/homepage/data/LN_fdiv.pdf.

[37] P. Sermanet, C. Lynch, J. Hsu, and S. Levine. Time-contrastive networks: Self-supervised learning from multi-view observation. *ArXiv:1704.06888*, 2017.

[38] H. S. Sikchi, A. Zhang, and S. Niekum. Imitation from arbitrary experience: A dual unification of reinforcement and imitation learning methods. *ArXiv:2302.08560*, 2023.

[39] J. Stanczuk, C. Etmann, L. Kreusser, and C.-B. Schonlieb. Wasserstein gans work because they fail (to approximate the wasserstein distance). *ArXiv:2103.01678*, 2021.

[40] M. Sun, A. Mahajan, K. Hofmann, and S. Whiteson. Softdice for imitation learning: Rethinking off-policy distribution matching. *ArXiv:2106.03155*, 2021.

[41] F. Torabi, G. Warnell, and P. Stone. Behavioral cloning from observation. In *IJCAI*, 2018.

[42] F. Torabi, G. Warnell, and P. Stone. Generative adversarial imitation from observation. In *ICML Workshop on Imitation, Intent, and Interaction*, 2019.

[43] C. Wan, T. Zhang, Z. Xiong, and H. Ye. Representation learning for fault diagnosis with contrastive predictive coding. In *CAA Symposium on Fault Detection, Supervision, and Safety for Technical Processes (SAFEPROCESS)*, 2021.

[44] A. Wu, A. Piergiovanni, and M. S. Ryoo. Model-based behavioral cloning with future image similarity learning. In *CoRL*, 2019.

[45] H. Xiao, M. Herman, J. Wagner, S. Ziesche, J. Etesami, and T. H. Linh. Wasserstein adversarial imitation learning. *arXiv preprint arXiv:1906.08113*, 2019.

[46] D. Xu and M. Denil. Positive-unlabeled reward learning. In *CoRL*, 2019.

[47] H. Xu, X. Zhan, H. Yin, and H. Qin. Discriminator-weighted offline imitation learning from suboptimal demonstrations. In *NeurIPS*, 2022.

[48] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *CoRL*, 2019.

[49] M. Zhang, Y. Wang, X. Ma, L. Xia, J. Yang, Z. Li, and X. Li. Wasserstein distance guided adversarial imitation learning with reward shape exploration. In *DDCLS*, 2020.

[50] J. Zhu, Y. Xia, L. Wu, J. Deng, W. Zhou, T. Qin, T.-Y. Liu, and H. Li. Masked contrastive representation learning for reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[51] Z. Zhu, K. Lin, B. Dai, and J. Zhou. Off-policy imitation learning from observations. In *NeurIPS*, 2020.

[52] K. Zolna, A. Novikov, K. Konyushkova, C. Gulcehre, Z. Wang, Y. Aytar, M. Denil, N. de Freitas, and S. E. Reed. Offline learning from demonstrations and unlabeled experience. In *Offline Reinforcement Learning Workshop at NeurIPS*, 2020.

Appendix: Offline Imitation from Observation via Primal Wasserstein State Occupancy Matching

The appendix is organized as follows. We first give rigorous introductions on the most important mathematical concepts in our work in Sec. A, which includes state, state-action and state-pair occupancy, as well as f-divergence and Fenchel conjugate; then, in Sec. B, we give detailed math derivations omitted in the main paper, as well as the proofs of the theorems and corollaries appearing in the main paper; in Sec. C, we give detailed description of our experiments; in Sec. D, we give additional experimental results, including auxiliary metrics and identical settings as LobsDICE [20] in the tabular experiment, and ablations in mujoco environments; in Sec. E, we discuss the limitation of the work; finally, in Sec. F, we give a notation list containing all notations in the paper as a reference.

A Mathematical Concepts

In this section, we introduce three important mathematical concepts in our paper, which are state/state-action/state-pair occupancy, f-divergence, and Fenchel conjugate. The first one is the key concept throughout the work, the second is used in our motivation and Thm. 1, and the last is used in Sec. 3.3.

A.1 State, State-Action, and State-Pair Occupancy

Consider a MDP (S, A, T, r, γ) with initial state distribution p_0 and infinite horizon; at t-th timestep, we denote the current state as s_t and the action as a_t. Then, with a fixed policy π, the probability of $\Pr(s_t = s)$ and $\Pr(a_t = a)$ for any s, a are determined. Based on this, the *state occupancy*, which is the state visitation frequency under policy π, is defined as $d_s^\pi(s) = (1-\gamma)\sum_{t=1}^{\infty}\gamma^t \Pr(s_t = s)$. Similarly, we define *state-action occupancy* as $d_{sa}^\pi(s,a) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \Pr(s_t = s, a_t = a)$. Some works such as LobsDICE also use *state-pair occupancy*, which is defined as $d_{ss}^\pi(s,s') = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \Pr(s_t = s, s_{t+1} = s')$. In this work, we denote the average policy that generates the task-agnostic dataset I as π^I with state occupancy d_s^I and state-action occupancy d_{sa}^I, and the expert policy that generates the expert dataset E as π^E with state occupancy d_s^E.

A.2 f-divergences

The f-divergence is a measure of distance between probability distributions p, q and is widely used in the machine learning community [17]. For two probability distributions p, q on domain $\mathcal{X}$ based on any continuous and convex function f, the f-divergence between p and q is defined as

$$D_f(p\|q) = \mathbb{E}_{x\sim q}[f(\frac{p(x)}{q(x)})]. \tag{12}$$

For instance, when $f(x) = x\log x$, we have $D_f(p\|q) = \mathbb{E}_{x\sim q}\frac{p(x)}{q(x)}\log\frac{p(x)}{q(x)} = \mathbb{E}_{x\sim p}\log\frac{p(x)}{q(x)}$, which induces KL-divergence; and when $f(x) = (x-1)^2$, we have $D_f(p\|q) = \mathbb{E}_{x\sim q}((\frac{p(x)-q(x)}{q(x)})^2)$, which induces χ^2-divergence.

A.3 Fenchel Conjugate

Fenchel conjugate is widely used in DICE methods for either debiasing estimations [34] or solving formulations with stronger constraints to get numerically more stable objectives [33]; PW-DICE uses Fenchel conjugate for the latter. For vector space Ω and convex, differentiable function $f : \Omega \to \mathbb{R}$, the Fenchel conjugate of $f(x)$ is defined as

$$f_*(y) = \max_{x\in\Omega}\langle x, y\rangle - f(x), \tag{13}$$

where $\langle\cdot,\cdot\rangle$ is the inner product over γ.

B Mathematical Derivations

In this section, we give the detailed mathematical derivations omitted in the main paper due to page limit. In Sec. B.1, we briefly introduce SMODICE to clarify the motivation of using Wasserstein distance and preliminary for Thm. 2; in Sec. B.2, we give a detailed derivation on the elements of A and b from Eq. 2 to Eq. 7 in Sec. 3; in Sec. B.3, we explain why additional constraints are applied from Eq. 4 to Eq. 5 while the optimal solution remains the same; in Eq. B.4, we give the source of the proof for Thm. 1; finally in Eq. B.5, we give a detailed proof for Thm. 2 and Corollary 1.

B.1 SMODICE

SMODICE [33] is a state-of-the-art offline LfO method. It minimizes the f-divergence between the state occupancy of the learner's policy π and the expert policy π^E, i.e., the objective is

$$\min_{\pi} D_f(d_s^{\pi}(s) \| d_s^E(s)), \text{s.t.} \pi \text{ is feasible.} \tag{14}$$

where the feasibility of π is the same as the Bellman flow constraint (the second row of constraints in Eq. 1) in the main paper. To take the only information source of environment dynamics, which is the task-agnostic dataset I into account, the objective is relaxed to

$$\max_{\pi} \mathbb{E}_{s\sim d^{\pi}} \log \frac{d_s^E(s)}{d_s^I(s)} - D_f(d_{sa}^{\pi}(s,a) \| d_{sa}^I(s,a)), \text{s.t. } \pi \text{ is a feasible policy,} \tag{15}$$

where D_f can be any divergence not smaller than KL-divergence (SMODICE mainly studies χ^2-divergence). The first term, $\log \frac{d_s^E(s)}{d_s^I(s)}$ indicates the relative importance of the state; the more often the expert visit a particular state s than non-expert policies, the more possible that s is a desirable state. Reliance on such ratio introduces a theoretical limitation: the assumption that $d_s^I(s) > 0$ wherever $d_s^E(s) > 0$ must be made, which does not necessarily hold in high-dimensional space. Thus, we introduce a hyperpamater of α to mix the distribution in the denominator in our reward design.

By converging the constrained problem into unconstrained problem in the Lagrange dual space, SMODICE optimizes the following objective (assuming using KL-divergence):

$$\min_{V} (1-\gamma)\mathbb{E}_{s\sim p_0}[V(s)] + \log \mathbb{E}_{(s,a,s')\sim I} \exp[\log \frac{d_s^E(s)}{d_s^I(s)} + \gamma V(s') - V(s)], \tag{16}$$

where p_0 is the initial state distribution and γ is the discount factor. As stated in Thm. 2, such objective is a special case of PW-DICE with $c(s,s') = \log \frac{d_s^E(s)}{d_s^I(s)}$, $\epsilon_2 = 1, \epsilon_1 \to 0$. LobsDICE [20] is similar in spirit; however, it minimizes state pair divergence $\mathrm{KL}(d_{ss}^{\pi} \| d_{ss}^E)$ instead.

B.2 Components of A, b in Eq. 2

In Eq. 2, we summarize all equality constraints in Eq. 1 as $Ax = b$, $x = \begin{bmatrix} \Pi \\ d_{sa}^{\pi} \end{bmatrix}$, where Π, d_{sa}^{π} are row-firstly expanded. Thus, we have $x_{:i|S|+j} = \Pi(s_i, s_j)$, and $x_{|S|^2+i|A|+j} = d_{sa}^{\pi}(s_i, a_j)$.

We further assume that in A and b, the first $|S|$ rows are the Bellman flow constraint

$$\forall s, \sum_a d_{sa}^{\pi}(s,a) - \gamma \sum_{\bar{s},\bar{a}} p(s|\bar{s},\bar{a}) d_{sa}^{\pi}(\bar{s},\bar{a}) = (1-\gamma)p_0(s), \tag{17}$$

the second $|S|$ rows are the $\sum_j \Pi(s_i, s_j) = d_s^{\pi}(s_i)$ marginal constraint

$$\forall s, \sum_{s'} \Pi(s,s') = \sum_a d_{sa}^{\pi}(s,a), \tag{18}$$

and the third $|S|$ rows are the $\sum_i \Pi(s_i, s_j) = d_s^E(s_j)$ constraint

$$\forall s, \sum_{s'} \Pi(s', s) = \sum_a d_{sa}^E(s, a). \tag{19}$$

Thus, we have $A_{i,|S|^2+j|A|+k} = -\gamma p(s_i|s_j, a_k)$ for $i \in \{1, 2, \ldots, |S|\}$, $A_{i,|S|^2+i|A|:|S|^2+(i+1)|A|} = 1$ for $i \in \{1, 2, \ldots, |S|\}$ (Eq. 17), $A_{i+|S|,i|S|+j} = 1$ for $i \in \{1, 2, \ldots, |S|\}$, $A_{i+|S|,|S|^2+i|A|+j} = -1$ (Eq. 18), and $A_{i+2|S|,j|S|+i} = 1$ (Eq. 19). Other entries of A are 0. For vector b, we have

$$b = \begin{bmatrix} (1-\gamma)p_0 \\ 0 \\ d_s^E \end{bmatrix}. \tag{20}$$

B.3 Lemma 1

Lemma. 1 is stated as follows:

Lemma 1. *For any MDP and feasible expert policy π^E, the inequality constraints in Eq. 1 with $\Pi \geq 0, d_{sa}^\pi \geq 0$ and $\Pi \in \Delta, d_{sa}^\pi \in \Delta$ are equivalent.*

Proof. according to the equality constraint, $\sum_s \Pi(s, s') = d_s^E(s')$ for any s'. Thus, we have $\sum_{s'} \sum_s \Pi(s, s') = \sum_{s'} d_s^E(s') = 1$ by the definition of state occupancy, and thus $\Pi \geq 0$ is equivalent to $\Pi \geq \Delta$. Similarly, by summing over both sides of the Bellman flow equality constraint, we have

$$\begin{aligned}
\sum_s d_s^\pi(s) &= \sum_s (1-\gamma)p_0(s) + \sum_s \gamma \sum_{\bar{s},\bar{a}} d_{sa}^\pi(\bar{s}, \bar{a}) p(s|\bar{s}, \bar{a}) \\
\sum_{s,a} d_{sa}^\pi(s, a) &= (1-\gamma) + \gamma \sum_s \sum_{\bar{s},\bar{a}} d_{sa}^\pi(\bar{s}, \bar{a}) p(s|\bar{s}, \bar{a}) \\
\sum_{s,a} d_{sa}^\pi(s, a) &= (1-\gamma) + \gamma \sum_{s'} \sum_{s,a} d_{sa}^\pi(s, a) p(s'|s, a) \\
\sum_{s,a} d_{sa}^\pi(s, a)(1 - \gamma \sum_{s'} p(s'|s, a)) &= 1 - \gamma \\
\sum_{s,a} d_{sa}^\pi(s, a) &= 1
\end{aligned} \tag{21}$$

given that p_0 and transition function are legal. Thus, $d_{sa}^\pi \geq 0$ is equivalent to $d_{sa}^\pi \in \Delta$.

□

Intuitively, by adding the extra constraints, we can assume that redundant equality constraints exist in Eq. 1, and they are not relaxed in the Lagrange dual. By imposing more strict constraints over the dual form, Fenchel conjugate yields numerically more stable formulation.

B.4 Theorem 1

Thm. 1 is stated as follows:

Theorem 1. *With mild assumptions [12], for any f-divergence D_f, probability distribution p, q on domain $\mathcal{X}$ and function $y : \mathcal{X} \to \mathbb{R}$, we have*

$$\max_{p \in \Delta(\mathcal{X})} \mathbb{E}_{x \sim p}[y(x)] - D_f(p\|q) = \mathbb{E}_{x \sim q}[f_*(y(x))] \tag{22}$$

also, for $p^* = \arg\max_{p \in \Delta(\mathcal{X})} \mathbb{E}_{x \sim q}[f_*(y(x))]$, we have

$$p^*(x) = q(x) f_*'(y(x)), \tag{23}$$

where $f_*(\cdot)$ is the Fenchel conjugate of f, and f_*' is its derivative.

This theorem is utilized in SMODICE [33] and our work to get a more robust optimization objective. The proof of the theorem is out of scope of this work; see Sec. 7.14* [36] for the detailed proof of the theorem.

B.5 Theorem 2 and Corollary 1

Thm. 2 and Corollary 1 are stated as follows:

Theorem 2. *If* $c(s_i, s_j) = -\log \frac{d_s^E(s_i)}{d_s^I(s_i)}, \epsilon_2 = 1$*, then as* $\epsilon_1 \to 0$*, Eq. 8 is equivalent to SMODICE objective with KL divergence.*

Corollary 1. *If* $c(s_i, s_j) = -\log \frac{d_s^E(s_i)}{d_s^I(s_i)}, \epsilon_2 = 1$*, then as* $\epsilon_1 \to 0$*, Eq. 5 is equivalent to SMODICE with any* f*-divergence.*

We first give a simple proof from the primal persepctive:

Proof. (Primal Perspective) According to Eq. 15 and Eq. 1, the SMODICE and PW-DICE primal objectives are as follows:

$$\begin{aligned} &\min_x (c')^T x + \epsilon_1 D_f(\Pi \| U) + \epsilon_2 D_f(d_{sa}^\pi \| d_{sa}^I), \text{s.t. } Ax = b, x \geq 0; \text{ (PW-DICE)} \\ &\max_\pi \mathbb{E}_{s\sim d^\pi} \log \frac{d_s^E(s)}{d_s^I(s)} - D_f(d_{sa}^\pi(s,a) \| d_{sa}^I(s,a)), \text{s.t. } \pi \text{ is a feasible policy. (SMODICE)} \end{aligned} \tag{24}$$

where $x = \begin{bmatrix} d_s^\pi \\ \Pi \end{bmatrix}$. Note: 1) $Ax = b, x \geq 0$ contains three equality constraints: Bellman flow equation (which is the same as "π is a feasible policy"), $\sum_{s'} \Pi(s, s') = d_s^\pi(s)$, and $\sum_s \Pi(s, s') = d^E(s')$; 2) $(c')^T x = \sum_{s,s'} c(s, s')\Pi(s, s')$. Thus, we have

$$\sum_s \sum_{s'} c(s, s')\Pi(s, s') = \sum_s \log \frac{d_s^E(s)}{d_s^I(s)} \sum_{s'} \Pi(s, s') = -\mathbb{E}_{s\sim d_s^\pi} \log \frac{d_s^E(s)}{d_s^I(s)}. \tag{25}$$

Therefore, when $\epsilon_1 = 0, \epsilon_2 = 1$, the objective between PW-DICE and SMODICE is exactly the opposite (with one maximization and the other minimization), and the constraints on d_{sa}^π are identical. Since Π is also solvable (one apparent solution is $\Pi = d_s^\pi \otimes d_s^E$), the two objectives are identical, and thus Eq. 1 and Eq. 15 are equivalent. Since Eq. 1, Eq. 5 and Eq. 8 are equivalent due to strong duality, both the Theorem and the Corollary are proved. □

However, such theorem is unintuitive in its dual form: as we always have $\epsilon_1 > 0, \epsilon_2 > 0$ in the dual form, the behavior of $\lim_{\epsilon_1 \to 0} \epsilon_1 \log \mathbb{E}_{s_i \sim I, s_j \sim E} \exp(\frac{\lambda_{i+|S|} + \lambda_{j+2|S|} - c(s_i, s_j)}{\epsilon_1})$ in Eq. 8 is non-trivial. Thus, here we give another proof directly from the dual perspective for KL-divergence as D_f in the continuous space:

Proof. (Dual Perspective, KL-divergence, continuous space) First, we prove by contradiction that

$$\lim_{\epsilon_1 \to 0} \epsilon_1 \log \mathbb{E}_{s\sim I, s'\sim E} \exp\left(\frac{\lambda_{s+|S|} + \lambda_{s'+2|S|} - c(s, s')}{\epsilon_1}\right) \tag{26}$$

is not $\max$ operator, because at optimal we have $\lambda_{s+|S|} + \lambda_{s'+2|S|} - c(s, s')$ to be equal for every $d_s^I(s) > 0, d_s^E(s') > 0$. Otherwise, assume the state pair (s, s') has the largest $\lambda_{s+|S|} + \lambda_{s'+2|S|} - c(s_0, s_0')$; because ϵ_1 can be arbitrarily close to 0, there exists ϵ_1 small enough such that there exists $s \neq s_0$ or $s' \neq s_0'$ that makes the infinitesimal increment of λ_s or λ_s' worthy (i.e., partial derivative with respect to λ_s or λ_s' greater than 0).

Then, we have

$$\begin{aligned} &\lim_{\epsilon_1 \to 0} \epsilon_1 \log \mathbb{E}_{s\sim I, s'\sim E} \exp\left(\frac{\lambda_{s+|S|} + \lambda_{s'+2|S|} - c(s, s')}{\epsilon_1}\right) \\ =&\mathbb{E}_{s\sim I, s'\sim E}(\lambda_{s+|S|} + \lambda_{s'+2|S|} - c(s, s')) \\ =&\mathbb{E}_{s\sim I}[\lambda_{s+|S|} + \log \frac{d_s^E(s)}{d_s^I(s)}] + \mathbb{E}_{s'\sim E}\lambda_{s'+2|S|}. \end{aligned} \tag{27}$$

Note that $\lambda_{s'+2|S|}$ canceled out with the term later, so the value of $\lambda_{s'+2|S|}$ does not matter anymore. That means, for any $\lambda_{s'+2|S|}$, there exists an optimal solution (In fact, different optimal solution can be converted by the formula in the next subsection). Therefore, without loss of generality, we let $\lambda_{s'+2|S|} = 0$. The objective then becomes

$$\begin{aligned} &\epsilon_1 \log \mathbb{E}_{s\sim I} \exp(\frac{\lambda_{s+|S|} + \log \frac{d_s^E(s)}{d_s^I(s)}}{\epsilon_1}) + \\ &\epsilon_2 \log \mathbb{E}_{(s,a,s')\sim I} \exp\left(\frac{-\gamma\lambda_{s'} + \lambda_s - \lambda_{s+|S|}}{\epsilon_2}\right) - (1-\gamma)\mathbb{E}_{s\sim p_0}\lambda_s. \end{aligned} \tag{28}$$

Then, we can use the same trick on $\epsilon_1 \to 0$ and infer that $\lambda_{s+|S|} = -\log \frac{d_s^E(s)}{d_s^I(s)} + Q$, where Q is some constant. Then, we have

$$L(\lambda) = Q + \epsilon_2 \log \mathbb{E}_{(s,a,s')\sim I} \exp\left(\frac{-\gamma\lambda_{s'} + \lambda_s + \log \frac{d_s^E(s)}{d_s^I(s)} - Q}{\epsilon_2}\right) - (1-\gamma)\mathbb{E}_{s\sim p_0}\lambda_s. \tag{29}$$

Note that Q is cancelled out again, which means the value of Q does not matter. Without loss of generality, we set $Q = 0$, and then we get SMODICE objective with KL-divergence. □

C Experiment Details

C.1 Tabular MDP

Experimental Settings. We adopt the tabular MDP experiment from LobsDICE [20]. For the tabular experiment, there are 20 states in the MDP and 4 actions for each state s; each action a leads to four uniformly chosen state s'_1, s'_2, s'_3, s'_4. The possibility vector for each possibility is determined by the formula $(p(s'_1|s,a), p(s'_2|s,a)), p(s'_3|s,a), p(s'_4|s,a) = (1-\beta)X + \beta Y$, where $X \sim$ Categorical$(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$, and $Y \sim$ Dirichlet$(1,1,1,1)$. $\beta \in [0,1]$ controls the randomness of the transition: $\beta = 0$ means deterministic, and $\beta = 1$ means highly stochastic. The agent always starts from state s_0, and can only get a reward of $+1$ by reaching a particular state s_x. x is chosen such that value function at optimal $V^*(s_0)$ is minimized. Discount factor γ is set to 0.95.

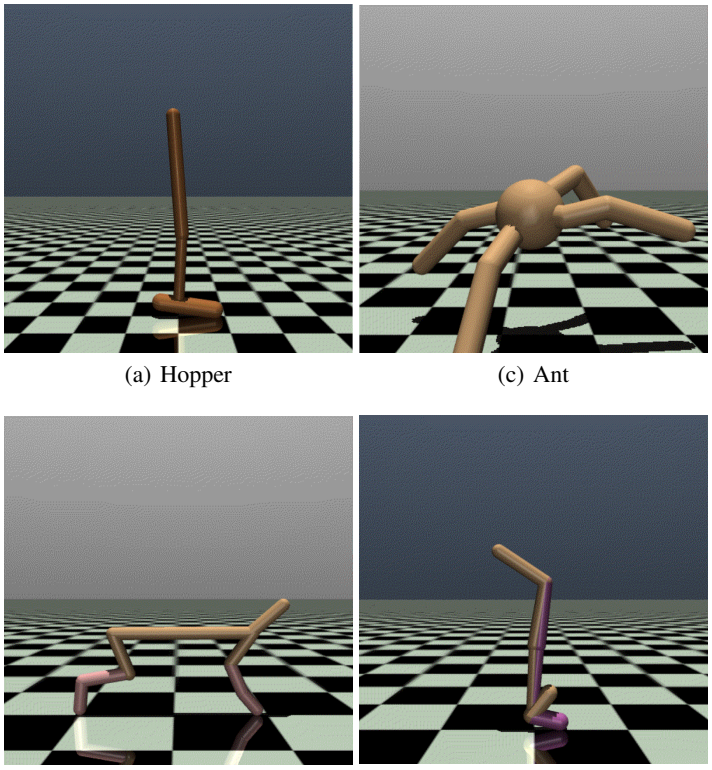
Dataset Settings. For each MDP, The expert dataset is generated using a deterministic optimal policy with infinite horizon, and the task-agnostic dataset is generated similarly but with a uniform policy. Note we use a different expert policy from the softmax policy of LobsDICE, because we found that due to the high connectivity of the MDP, the value function for each state are quite close to each other; thus, the "expert" softmax policy is actually near-uniform and severely sub-optimal.

Selection of Hyperparameters. There is no hyperparameter selection for SMODICE; for LobsDICE, we follow the settings in their paper, which is $\alpha = 0.1$. For our method, we use $\epsilon_1 = \epsilon_2 = 0.01$ for our method with regularizer, and $\epsilon_1 = \epsilon_2 = 0$ for our method with Linear Programming (LP).

C.2 Mujoco Environment

Experimental Settings. We test four widely adopted mujoco locomotion environments, which are hopper, halfcheetah, ant and walker2d. Below is the detailed description for each environment; see Fig. 6 for illustration.

1. **Hopper.** Hopper is a 2D environment where the agent controls a single-legged robot to jump forward. The state is 11-dimensional, which includes the angle and velocity for each joint of the robot; the action is 3-dimensional, each of which controls the torque applied on a particular joint.
2. **Halfcheetah**. In Halfcheetah, the agent controls a cheetah-like robot to run forward. Similar to Hopper, the environment is also 2D, with 17-dimensional state space describing the coordinate and velocity and 6-dimensional action space controlling torques on its joints.



(a) Hopper (c) Ant

(b) Halfcheetah (d) Walker2d

Figure 6: Illustration of environments tested in Sec. 4.2 based on OpenAI Gym [5] and D4RL [15].

3. **Ant.** Ant is a 3D environment where the agent controls a quadrupedal robotic ant to move forward with 111-dimensional state space including the coordinate and velocity of each joint. The action space is 8-dimensional.
4. **Walker2d.** Walker2d, as its name suggests, is a 2D environment where the agent controls a two-legged robot to walk forward. The state space is 27-dimensional and the action space is 8-dimensional.

Dataset Settings. We adopt the same settings as SMODICE [33]. SMODICE uses a single trajectory (1000 states) from the "expert-v2" dataset in D4RL [15] as the expert dataset E. For the task-agnostic dataset I, SMODICE uses the concatenation of 200 trajectories ($200K$ state-action pairs) from "expert-v2" and the whole "random-v2" dataset ($1M$ state-action pairs).

Selection of Hyperparameters. Tab. 1 summarizes our hyperparameters, which is also the hyperpameters of plain Behavior Cloning if applicable. For baselines (SMODICE, LObsDICE, ORIL, OTR and DWBC), we use the hyperparameters reported in their paper (unless the hyperparameter values in the paper and the code are different; in that case, we record the values from the code).

D Additional Experimental Results

D.1 Supplementary Results for the Tabular Environment

D.1.1 State and State-pair Total Variation (TV) distance

In this section, we show the Total Variation (TV) divergence between the state and state-pair occupancies of the learner and expert, i.e., $\text{TV}(d_s^\pi \| d_s^E)$ and $\text{TV}(d_{ss}^\pi \| d_{ss}^E)$. Fig. 7 shows the result of state occupancy distance between learner and expert policy, and Fig. 8 shows the distance between state-pair occupancies. It is clearly shown that our method works better than SMODICE and LobsDICE.

Type	Hyperparameter	Value	Note
Disc.	Network Size	[256, 256]	
	Activation Function	Tanh	
	Learning Rate	0.0003	
	Training Length	40K steps	
	Batch Size	512	
	Optimizer	Adam	
Actor	Network Size	[256, 256]	
	Activation Function	ReLU	
	Learning Rate	0.001	
	Weight Decay	10^{-5}	
	Training length	1M steps	
	Batch Size	1024	
	Optimizer	Adam	
	Tanh-Squashed	Yes	
Critic	Network Size	[256, 256]	
	Activation Function	ReLU	
	Learning Rate	0.0003	
	Training Length	1M steps	
	Batch Size	1024	
	Optimizer	Adam	
	ϵ_1	0.5	coefficient for the KL regularizer
	ϵ_2	0.5	coefficient for the KL regularizer
	α	0.01	mixing coefficient to the denominator of $R(s)$
	β	5	coefficient for combination of distance metric
	γ	0.998	discount factor in our formulation

Table 1: Our selection of hyperparameter. We use the same network architecture and optimizer as SMODICE [33].

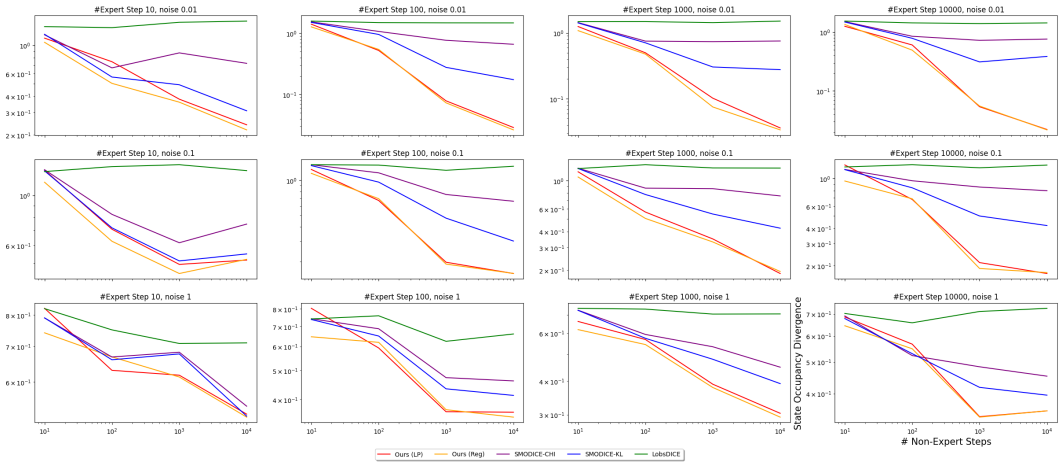


Figure 7: The TV distance $\text{TV}(d_s^\pi \| d_s^E)$ of each method on tabular environment. Our method, both with and without regularizer, works comparably well with the baselines on small task-agnostic dataset, and prevails with larger the task-agnostic dataset (more accurate estimated dynamics).

D.1.2 Tabular Experiment with Softmax Expert

To be consistent with LobsDICE [20], we also test the experiment result under exactly the same settings of LobsDICE, which uses an expert highly sub-optimal. Fig. 9, Fig. 10 and Fig. 11 shows the regret, state occupancy divergence $\text{TV}(d_s^\pi \| d_s^E)$ and state-pair occupancy divergence $\text{TV}(d_{ss}^\pi \| d_{ss}^E)$ of each method under such settings. The result shows that our method does not performs well in minimizing occupancy divergence, as the coefficient of f-divergence regularizer in our method is much smaller or 0, which means our obtained policy is more deterministic and thus different from the highly stochastic "expert" policy. It is worth noting that our method, with accurate estimation of MDP dynamics (i.e. large size of task-agnostic/non-expert dataset), is the only method that achieves negative regret, i.e., our method is even better than the "expert" policy; also, our method with regularizer generally achieves lower regret.

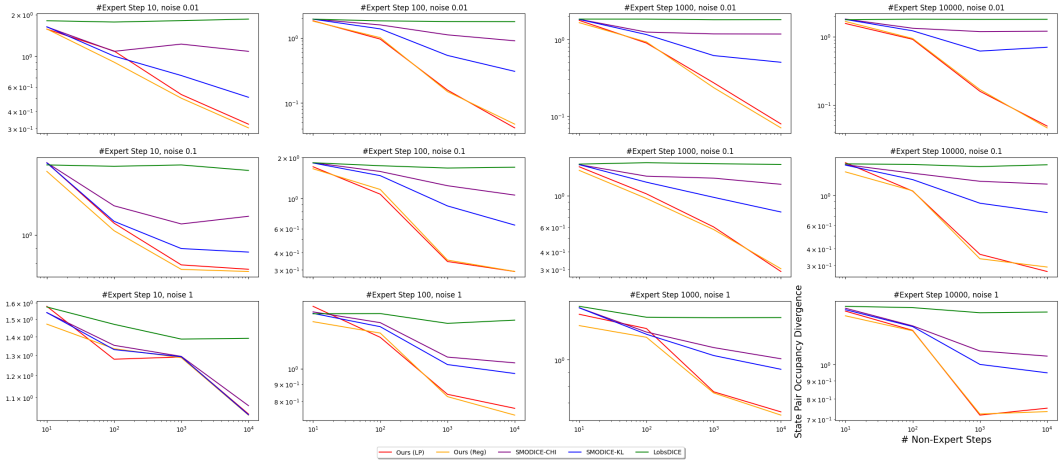


Figure 8: The state-pair occupancy TV distance between the learner and expert ($\text{TV}(d^{\pi}_{ss} \| d^{E}_{ss})$) on tabular environment. LobsDICE works the best, but this is because LobsDICE maximizes state-pair occupancy instead of state occupancy.

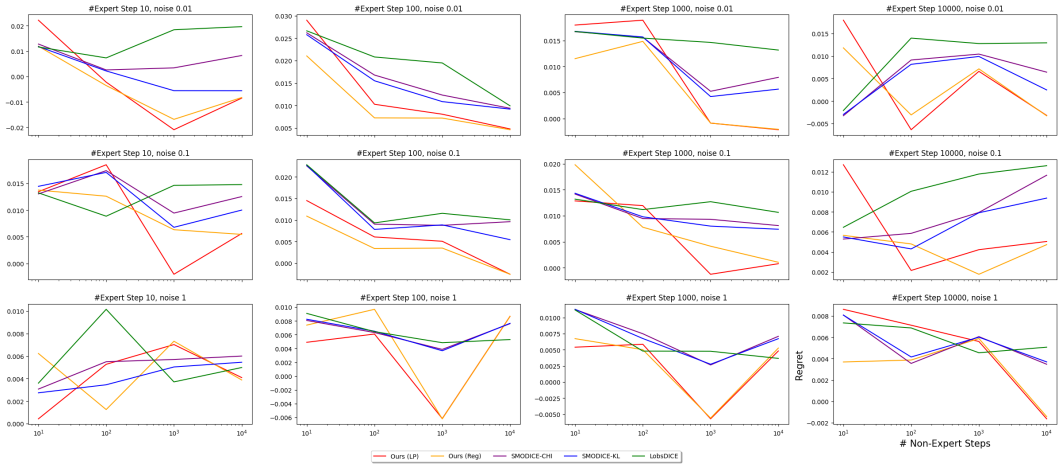


Figure 9: The regret of each method for tabular experiment with softmax expert. Our method with regularizer generally achieves the lower regret; also, our method is the only one that achieves negative regret (i.e. better than the highly sub-optimal "expert").

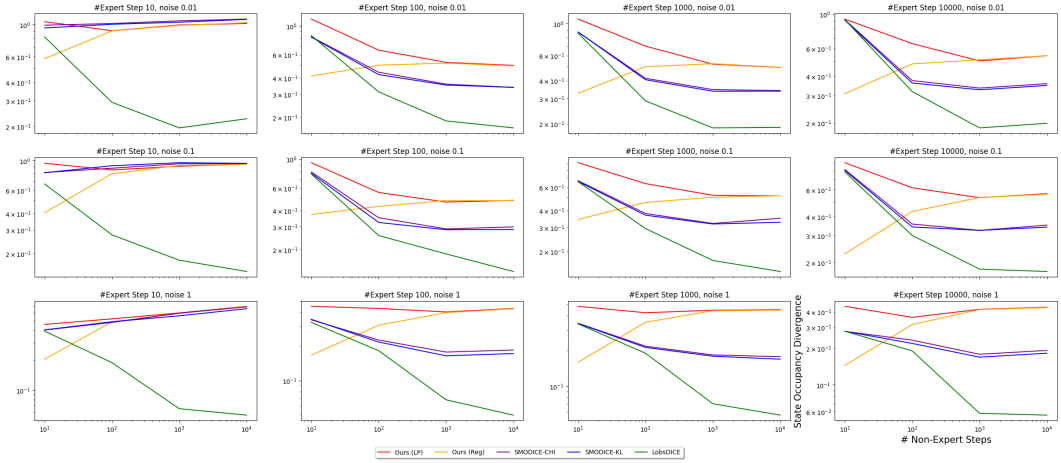


Figure 10: The state occupancy TV distance $\text{TV}(d^{\pi}_{s} \| d^{E}_{s})$ of each method for tabular experiment with softmax expert. Our method does not work well because the expert policy is highly stochastic.

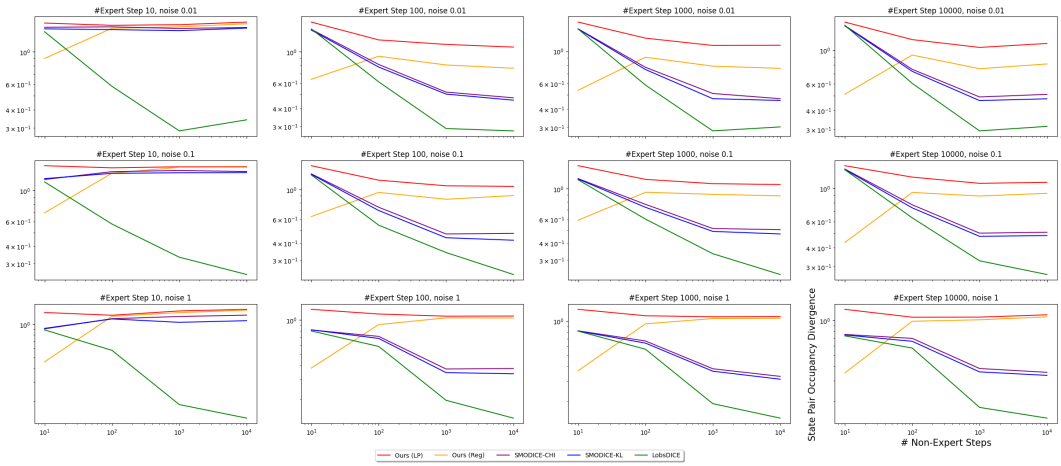


Figure 11: The state-pair occupancy TV distance $\text{TV}(d^{\pi}_{ss} \| d^{E}_{ss})$ of each method for tabular experiment with softmax expert. Our method does not work well because the expert policy is highly stochastic.

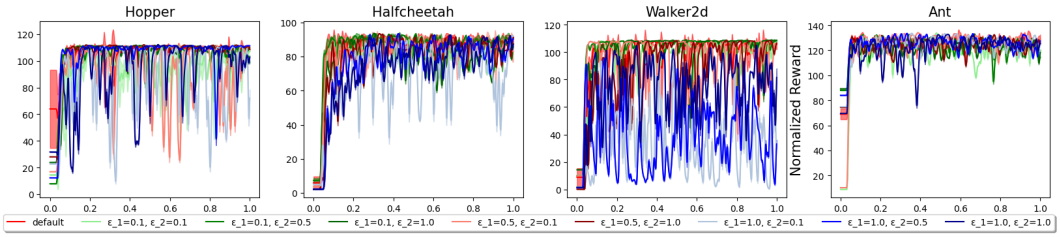


Figure 12: The ablation of ϵ_1 and ϵ_2 on mujoco testbed; $\epsilon_1 = 0.1$ are marked as green, $\epsilon_1 = 0.5$ are marked as red and $\epsilon_1 = 1.0$ are marked as blue. The deeper the color, the larger ϵ_2 is. Our method is generally robust to hyperparameter changes, though some choice leads to failure. Generally, large ϵ_1 leads to worse performance.

D.2 The Effect of ϵ_1 and ϵ_2

In order to show the robustness of PW-DICE with the choice of ϵ_1 and ϵ_2, we conduct an ablation study on the choice of ϵ_1 and ϵ_2 on the mujoco environment; specifically, we test $\epsilon_1 \in \{0.1, 0.5, 1\} \times \epsilon_2 \in \{0.1, 0.5, 1\}$. The result is shown in Fig. 12. While some choice of hyperparameter leads to failure, PW-DICE is generally robust to the choice of ϵ_1 and ϵ_2; generally, ϵ_1 should be small to maintain good performance.

D.3 PW-DICE with χ^2-divergence on Mujoco Environment

In the main paper, we mainly considered PW-DICE with KL-divergence; however, as Corollary 1 suggests, the D_f regularizer in PW-DICE can also be χ^2-divergence. Suppose we use half χ^2-divergence as SMODICE [33] does, i.e., $f(x) = \frac{1}{2}(x-1)^2$, $f_*(x) = \frac{1}{2}(x+1)^2$ and $f'(x) = x + 1$; With such divergence, the final optimization objective of PW-DICE becomes

$$\begin{aligned} &\min_{\lambda} \frac{\epsilon_1}{2} \mathbb{E}_{s_i \sim I, s_j \sim E}\Big(\frac{\lambda_{i+|S|} + \lambda_{j+2|S|} - c(s_i, s_j)}{\epsilon_1} + 1\Big)^2 \\ &+ \frac{\epsilon_2}{2} \mathbb{E}_{(s_i, a_j, s_k) \sim I}\Big(\frac{-\gamma \lambda_k + \lambda_i - \lambda_{i+|S|}}{\epsilon_2} + 1\Big)^2 - \big[(1-\gamma)\mathbb{E}_{s \sim p_0} \lambda_{:|S|} + \mathbb{E}_{s \sim E} \lambda_{2|S|:3|S|}\big] \end{aligned} \tag{30}$$

and the policy loss is

$$E_{(s,a) \sim I} \max\Big(0, \frac{-\gamma \mathbb{E}_{s_k \sim p(\cdot|s_i, a_j)} \lambda_k + \lambda_i - \lambda_{i+|S|}}{\epsilon_2}\Big). \tag{31}$$

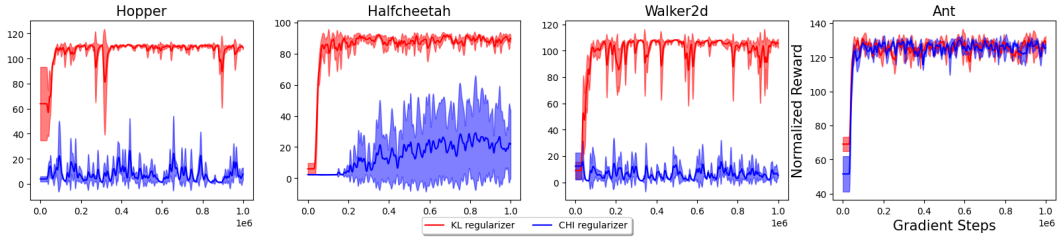


Figure 13: Performance comparison between χ^2-divergence (blue) and KL-divergence (red) in PW-DICE. χ^2-divergence does not work as well as KL-divergence.

However, similar to SMODICE, we found that χ^2-divergence regularizer does not work well under mujoco environments, as the weight ratio between good and bad actions in the task-agnostic dataset is only proportional (instead of exponential) to $-\gamma\lambda_k + \lambda_i - \lambda_{i+|S|}$, and thus is not discriminative enough. As a result, the retrieved policy is highly stochastic. Fig. 13 shows the result of χ^2-divergence, which is much worse than KL-divergence.

E Limitation

In order to get unconstrained optimization formulation, we add KL terms to the objective, which introduces logsumexp into the final objective. Some works argue that logsumexp brings instability to optimization [40], which may be a potential shortcoming of our paper on more complicated environments. Thus, one of the future directions is to find a more robust formulation while maintaining the good properties of PW-DICE.

F Notations Table

Tab. 2 shows the notations appear in the paper.

Name	Meaning	Note
S	State space	$\|S\|$ is the size of state space for tabular MDP
A	Action space	$\|A\|$ is the size of state space for tabular MDP
γ	Discount factor	$\gamma \in (0, 1)$
r	Reward function	$r(s, a)$ for single state-action pair
T	Transition function	
p	Transition (single entry)	$p(s'\|s, a) \in \Delta(S)$
p_0	Initial distribution	$p_0 \in \Delta(S)$
s	State	$s \in S$
a	Action	$a \in A$
$\bar{s}$	Past state	
$\bar{a}$	Past action	
τ	Trajectories	State-only or state-action; depend on context
E	Expert dataset	state-only expert trajectories
I	Task-agnostic dataset	state-action trajectories of unknown optimality
π	Learner policy	
π^E	Expert policy abstracted from E	
π^I	Task-agnostic policy abstracted from I	
d^π_{sa}	State-action occupancy of π	
d^π_s	State occupancy of π	1) $\forall s \in \mathcal{S}, \sum_a d^\pi_{sa}(s, a) = d^\pi_s(s)$. This equation also applies similarly between d^E_{sa} and d^E_s, as well as d^I_{sa} and d^I_s. 2) $d^\pi_s(s) = \sum_{i=0}^{\infty} \gamma^i \text{Pr}(s_i = s)$, where s_i is the i-th state in a trajectory. This holds similarly for $d^I(s)$ and $d^E(s)$. 3) $d^\pi_{sa}(s, a) = d^\pi_s(s)\pi(a\|s)$. This holds similarly for d^E_{sa}, π_E and d^I_{sa}, π_I.
d^π_{ss}	State-pair occupancy of π	
d^E_s	State occupancy of π^E	
d^E_{ss}	State-pair occupancy of π^E	
d^I_{sa}	State-action occupancy of π^I	
d^I_s	State occupancy of π^I	
λ	Dual variable	
D_f	f-divergence	
f_*	Fenchel conjugate of f	
c	Matching cost for Wasserstein distance	
c'	Matching cost for Wasserstein distance	With extended domain
Π	Wasserstein matching variable	$\sum_{s \in S} \Pi(s, s') = d^E_s(s'), \sum_{s' \in S} \Pi(s, s') = d^\pi_s(s)$
A	Equality constraint matrix	
x	unified self-variable	concatenation of flattened Π and d^π_{sa} (row first)
b	Equality Constraint vector	$Ax = b$
U	Distribution as regularizer	product of d^I_s and d^E_s
$\mathcal{W}$	Wasserstein distance	

Table 2: Complete list of notations used in the project. The first part is for offline LfO settings and the second part is notations specific to PW-DICE.