Marginalized Importance Sampling for Off-Environment Policy Evaluation

Pulkit Katdare
Department of Electrical and Computer Engineering
University of Illinois Urbana-Champaign Illinois, United States
`katdare2@illinois.edu`

Nan Jiang
Department of Computer Science
University of Illinois Urbana-Champaign Illinois, United States

Katherine Driggs-Campbell
Department of Electrical and Computer Engineering
University of Illinois Urbana-Champaign Illinois, United States

Abstract: Reinforcement Learning (RL) methods are typically sample-inefficient, making it challenging to train and deploy RL-policies in real world robots. Even a robust policy trained in simulation requires a real-world deployment to assess their performance. This paper proposes a new approach to evaluate the real-world performance of agent policies prior to deploying them in the real world. Our approach incorporates a simulator along with real-world offline data to evaluate the performance of any policy using the framework of Marginalized Importance Sampling (MIS). Existing MIS methods face two challenges: (1) large density ratios that deviate from a reasonable range and (2) indirect supervision, where the ratio needs to be inferred indirectly, thus exacerbating estimation error. Our approach addresses these challenges by introducing the target policy's occupancy in the simulator as an intermediate variable and learning the density ratio as the product of two terms that can be learned separately. The first term is learned with direct supervision and the second term has a small magnitude, thus making it computationally efficient. We analyze the sample complexity as well as error propagation of our two step-procedure. Furthermore, we empirically evaluate our approach on Sim2Sim environments such as Cartpole, Reacher, and Half-Cheetah. Our results show that our method generalizes well across a variety of Sim2Sim gap, target policies and offline data collection policies. We also demonstrate the performance of our algorithm on a Sim2Real task of validating the performance of a 7 DoF robotic arm using offline data along with the Gazebo simulator.

Keywords: Sim2Real, Policy Evaluation, Robot Validation

1 Introduction

Reinforcement Learning (RL) algorithms aim to select actions that maximize the cumulative returns over a finite time horizon. In recent years, RL has shown state-of-the-art performance over a range of complex tasks such as chatbots, [1], games [2], and robotics [3, 4, 5, 6]. However, RL algorithms still require a large number of samples, which can limit their practical use in robotics [7, 8].

A typical approach is to train robust robot policies in simulation and then deploy them on the robot [9, 10]. Such an approach, although useful, does not guarantee optimal performance on the real robot without significant fine tuning [11]. In this work, we propose an approach that evaluates

the real world performance of a policy using a robot simulator and offline data collected from the real-world [6]. To achieve this, we employ the framework of off-policy evaluation (OPE) [12].

OPE is the problem of using offline data collected from a possibly unknown *behavior* policy to estimate the performance of a different *target* policy. Classical OPE methods are based on the principle of importance sampling (IS) [13, 14, 15], which reweights each trajectory by its density ratio under the target versus the behavior policies. More recently, significant progress has been made on *marginalized importance sampling* (MIS), which reweights transition tuples using the density ratio (or importance weights) over states instead of trajectories to overcome the so-called curse of horizon [16, 17, 18]. The density ratio is often learned via optimizing minimax loss functions.

Most existing MIS methods are model-free, relying on data from the real environment to approximate the MIS weight function. However, a direct application of MIS methods to robotics carries two main disadvantages. (1) *Large ratios:* MIS method learns distribution mismatch between the behavior and the target policies. When the mismatch between the behavior and the target policy is large, which is often the case, MIS method tend be challenging to learn. (2) *Indirect supervision:* MIS methods requires no samples from target policies, which requires the weight being learned *indirectly* via the Bellman flow equation. In states where coverage of the offline dataset is scarce, MIS methods tend to perform poorly.

In robotics, it is reasonable to assume access to a good but imperfect simulator of the real environment [19, 20, 21]. In this work, we propose an improved MIS estimator that estimates the density ratio mismatch between the real world and the simulator. We further show that such a MIS estimator can be used to evaluate the real-world performance of a robot using just the simulator. As described in figure 1, we estimate the discrepancy between the real world and the simulator by using the target policy's occupancy in the simulator as an intermediate variable. This allows us to calculate the MIS weights as a combination of two factors, which can be learned independently. The first factor has direct supervision, while the second factor has a small magnitude (close to 1), thereby addressing both large ratios and indirect supervision issues mentioned above. We present a straightforward derivation of our method, examine its theoretical properties, and compare it to existing methods (including existing ways of incorporating a simulator in OPE) and baselines through empirical analysis.

We make the following contributions. (1) We derive an MIS estimator for off-environment evaluation (Section 4). (2) We explore the theoretical properties of our off-environment evaluation estimator by proposing a sample-complexity analysis (Section 4) and studying its special cases in linear and kernel settings. (3) We empirically evaluate our estimator on both Sim2Sim as well as Sim2Real environments (Section 5). For the Sim2Sim experiments, we perform a thorough ablation study over different Sim2Sim gap, data-collection policies and target policies, environments (Taxi, Cartpole, Reacher, and Half-Cheetah). Furthermore, we demonstrate practicality of our approach on a sim2real task by validating performance of a Kinova robotic arm over using offline data along with Gazebo based Kinova simulator.

2   Preliminaries

Robot learning problems are often modelled as an infinite-horizon discounted Markov Decision Process (MDP). MDP is specified by $(\mathcal{S}, \mathcal{A}, P, R, \gamma, d_0)$. Here, $\mathcal{S}$ and $\mathcal{A}$ are the state and the action spaces, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition function ($\Delta(\cdot)$ is the probability simplex). We also define the reward function $R : \mathcal{S} \times \mathcal{A} \to \Delta([0, R_{max}])$, $\gamma \in [0, 1)$ is the discount factor, and $d_0 \in \Delta(\mathcal{S})$ is the initial state distribution. A policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ induces a distribution of trajectory: $\forall t \geq 0$, $s_0 \sim d_0, a_t \sim \pi(\cdot|s_t), r_t \sim R(\cdot|s, a), s_{t+1} \sim P(\cdot|s_t, a_t)$. The performance of π is measured by its expected discounted return under the initial state distribution, defined as $J_P(\pi) = (1 - \gamma)\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | \pi, d_0]$; here, we use the subscript P in $J_P(\pi)$ to emphasize the dynamics w.r.t. which the return is defined, since we will consider both the true environment and the simulator in the rest of this paper and the subscript will help distinguish between them. $J_P(\pi)$ also
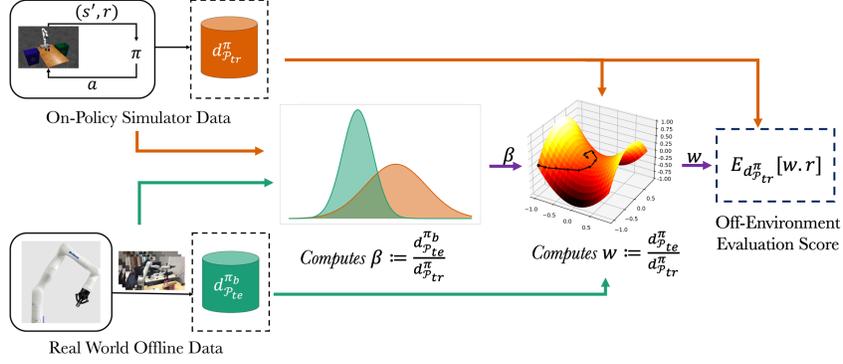
Figure 1: For a given policy π, we first collect on-policy data $d^\pi_{\mathcal{P}_{tr}}$ from a simulator environment. Using $d^\pi_{\mathcal{P}_{tr}}$ and offline data $d^{\mathcal{D}}_{\mathcal{P}_{te}}$ on the real world, we first calculate the importance sampling factor $\beta = d^{\pi_b}_{\mathcal{P}_{te}}/d^\pi_{\mathcal{P}_{tr}}$. This importance sampling factor essentially allows us to re-weight existing off-policy evaluation algorithm to estimate $w = d^\pi_{\mathcal{P}_{te}}/d^\pi_{\mathcal{P}_{tr}}$ which helps us estimate real-world performance of the agent $J_{\mathcal{P}_{te}}(\pi)$ using on-policy simulator data.

has an alternative expression $J_P(\pi) := \mathbb{E}_{(s,a)\sim d^\pi_P, r\sim R(s,a)}[r]$, where

$$d^\pi_P(s,a) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t\mathbb{P}[s_t = s, a_t = a|\pi, d_0] \tag{1}$$

is the discounted state-action occupancy induced by π from d_0. An important quantity associated with a policy is its Q-function Q^π_P, which satisfies the Bellman equation $Q^\pi_P(s,a) = \mathbb{E}_{r\sim R(s,a), s'\sim P(s,a)}[r + \gamma Q^\pi_P(s',\pi)]$, where $f(s',\pi) := \mathbb{E}_{a'\sim\pi(\cdot|s')}[f(s',a')]$. We can also define the state-value function $V^\pi_P(s) = Q^\pi_P(s,\pi)$, and $J(\pi) = (1-\gamma)\mathbb{E}_{s\sim d_0}[V^\pi_P(s)]$.

OPE and Marginalized Importance Sampling: In off-policy evaluation (OPE), we want to evaluate a target policy π using data collected from a different policy in the real environment, denoted by its dynamics P. As a standard simplification, we assume data is generated i.i.d. as $(s,a)\sim\mu, r\sim R(s,a), s'\sim P(s,a)$, and the sample size is n. When the data is generated from some behavior policy π_b, μ can correspond to its occupancy $d^{\pi_b}_P$. We will use $\mathbb{E}_\mu[\cdot]$ as a shorthand for taking expectation over (s,a,r,s') generated from such a data distribution in the real environment.

The key idea in marginalized importance sampling (MIS) is to learn the weight function $w^{\pi/\mu}_P(s,a) := \frac{d^\pi_P(s,a)}{\mu(s,a)}$. Once this function is known, $J(\pi)$ can be estimated as $J(\pi) = \mathbb{E}_{(s,a)\sim d^\pi_P, r\sim R(s,a)}[r] = \mathbb{E}_\mu[w^{\pi/\mu}_P(s,a)\cdot r]$. Note that $\mathbb{E}_\mu[\cdot]$ can be empirically approximated by the dataset sampled from the real environment. The real challenge in MIS is how to learn $w^{\pi/\mu}_P$. Existing works often do so by optimizing minimax loss functions using Q-functions as discriminators, and is subject to both difficulties (large ratios and indirect supervision) mentioned in the introduction. We refer the readers to [22] for a summary of typical MIS methods.

Learning Density Ratios from Direct Supervision: Given two distributions p and q over the same space $\mathcal{X}$, the density ratio $p(x)/q(x)$ can be learned *directly* if we have access to samples from both p and q, using the method proposed by [23]:

$$\frac{p(x)}{q(x)} = \arg\max_{f:\mathcal{X}\to\mathbb{R}_{>0}}\mathbb{E}_{x\sim p}[\ln f(x)] - \mathbb{E}_{x\sim q}[f(x)] + 1. \tag{2}$$

To guarantee generalization over a finite sample when we approximate the expectations empirically, we will need to restrict the space of f that we search over to function classes of limited capacities, such as RKHS or neural nets, and $p(x)/q(x)$ can still be well approximated as long as it can be represented in the chosen function class (i.e., *realizable*). More concretely, if we have n samples $x_1,\ldots,x_n$ from p and m samples $\tilde{x}_1,\ldots,\tilde{x}_m$ from q, and use $\mathcal{F}$ to approximate $p(x)/q(x)$, the learned density ratio can be made generalizable by adding a regularization term $I(f)$ to improve the statistical and computational stability of learning:

$$\arg\max_{f\in\mathcal{F}}\frac{1}{n}\sum_i\ln f(x_i) - \frac{1}{m}\sum_j f(\tilde{x}_j) + \frac{\lambda}{2}I(f)^2. \tag{3}$$

3

3 Related Work

3.1 Reinforcement Learning applications in Robotics

There are three different themes that arise in reinforcement learning for robotics [24, 25]. (1) **Sim2Real:** algorithms are primarily concerned with learning robust policy in simulation by training the algorithms over a variety of simulation configurations [26, 9, 27]. Sim2Real algorithms, although successful, still require a thorough real-world deployment in order to gauge the policy's performance. (2) **Imitation learning** algorithms learn an optimal policy by trying to mimic offline expert demonstrations [28, 29, 30]. Many successful imitation learning algorithms minimize some form of density matching between the expert demonstrations and on-policy data to learn optimal policy. A key problem with imitation learning is the fact that it requires constant interaction with the real-world environment in order to learn an optimal policy. (3) **Offline reinforcement learning** is a relatively new area. Here the idea is to learn an optimal policy using offline data without any interaction with the environment [31, 32, 33, 34]. Offline reinforcement learning has recently demonstrated performance at-par with classical reinforcement learning in a few tasks. However, offline reinforcement learning algorithms tend to overfit on the offline data. Thus, even offline learning modules too require an actual deployment in-order to assess performance.

3.2 Off-Policy Evaluation

We review related works in this section, focusing on comparing to existing OPE methods that can leverage the side information provided by an imperfect simulator. (1) **Marginalized Importance Sampling (MIS):** MIS methods tend to assume the framework of data collection and policy evaluation being on the same environment. To that end, there are both model-free [35, 36, 37] and model-based variants of MIS methods and face the aforementioned two challenges (large magnitude of weights and indirection supervision) simultaneously. Model based variants of MIS sometimes tend to be doubly robust (DR) in nature [18, 38] and can benefit from Q-functions as control variates, which can be supplied by the simulator. However, the DR version of MIS is a meta estimator, and the weight $d_{P_{te}}^{\pi}/\mu$ still needs to be estimated via a "base" MIS procedure. Therefore, the incorporation of the simulator information does not directly address the challenges we are concerned with, and there is also opportunity to further combine our estimator into the DR form of MIS. (2) **Model-based methods:** Model-based estimators first approximate the transition dynamics of the target environment [39], which is further used to generate rollouts and evaluate performance for any target policy. One way of incorporating a given imperfect simulator in this approach is to use the simulator as "base predictions," and only learn an additive correction term, often known as residual dynamics [40]. This approach combines the two sources of information (simulator and data from target environment) in a very different way compared to ours, and are more vulnerable to misspecification errors than model-free methods.

4 Weight Estimator

Recall that our goal is to incorporate a given simulator into MIS. We will assume that the simulator shares the same $\mathcal{S}, \mathcal{A}, \gamma, d_0$ with the real environment, but has its own transition function P_{tr} which can be different from P_{te}. As we will see, extension to the case where the reward function is unknown and must be inferred from sample rewards in the data is straightforward, and for simplicity we will only consider difference in dynamics for most of the paper.

Split the weight: The key idea in our approach is to split the weight $w_P^{\pi/\mu}$ into two parts by introducing $d_{P_{tr}}^{\pi}$ as an intermediate variable:

$$w_P^{\pi/\mu}(s,a) = \frac{d_{P_{te}}^{\pi}(s,a)}{\mu(s,a)} = \underbrace{\frac{d_{P_{tr}}^{\pi}(s,a)}{\mu(s,a)}}_{\substack{:=\beta \\ \text{(direct supervision)}}} \cdot \underbrace{\frac{d_{P_{te}}^{\pi}(s,a)}{d_{P_{tr}}^{\pi}(s,a)}}_{:=w_{P_{te}/P_{tr}}^{\pi} \text{(magnitude} \simeq 1)}$$

Note that $d_{P_{tr}}^{\pi}$ is the occupancy of π in the simulator, which we have free access to. The advantage of our approach is that by estimating β and $w_{P_{te}/P_{tr}}^{\pi}$ separately, we avoid the situation of running

into the two challenges mentioned before simultaneously, and instead address one at each time: $\beta = d_{P_{tr}}^\pi / \mu$ has large magnitude but can be learned directly via [23] without the difficult minimax optimization typically required by MIS, and we expect $w_{P_{te}/P_{tr}}^\pi = d_{P_{te}}^\pi / d_{P_{tr}}^\pi$ to be close to 1 when $P_{te} \approx P_{tr}$ (and thus easier to learn).

Estimate $w_{P_{te}/P_{tr}}^\pi$: Since β is handled by the method of [23], the key remaining challenge is how to estimate $w_{P_{te}/P_{tr}}^\pi$. (Interestingly, β also plays a key role in estimating $w_{P_{te}/P_{tr}}^\pi$, as will be shown below.) Note that once we have approximated $w_{P_{te}/P_{tr}}^\pi$ with some w, we can directly reweight the state-action pairs from the simulator (i.e., $d_{P_{tr}}^\pi$) if the reward function is known (this is only assumed for the purpose of derivation), i.e.,

$$J_{P_{te}}(\pi) \approx \mathbb{E}_{(s,a)\sim d_{P_{tr}}^\pi, r\sim R(s,a)}[w \cdot r],$$

and this becomes an identity if $w = w_{P_{te}/P_{tr}}^\pi$. Following the derivation in [18, 22], we now reason about the error of the above estimator for an arbitrary w to derive an upper bound as our loss for learning w:

$$|\mathbb{E}_{(s,a)\sim d_{P_{tr}}^\pi, r\sim R(s,a)}[w \cdot r] - J_{P_{te}}(\pi)|$$
$$= |\mathbb{E}_{(s,a)\sim d_{P_{tr}}^\pi, s'\sim P(s,a)}[w \cdot (Q_{P_{te}}^\pi(s,a) - \gamma Q_{P_{te}}^\pi(s',\pi))] - (1-\gamma)\mathbb{E}_{s\sim d_0}[Q_{P_{te}}^\pi(s,\pi)]|$$
$$\leq \sup_{q\in\mathcal{Q}} |\mathbb{E}_{d_{P_{tr}}^\pi \times P_{te}}[w \cdot (q(s,a) - \gamma q(s',\pi))] - (1-\gamma)\mathbb{E}_{s\sim d_0}[q(s,\pi)]|. \quad (4)$$

Here $d_{P_{tr}}^\pi \times P_{te}$ is a shorthand for $(s,a) \sim d_{P_{tr}}^\pi, s' \sim P(s,a)$. In the last step, we handle the unknown Q_P^π by a relaxation similar to [16, 18, 22], which takes an upper bound of the error over $q \in \mathcal{Q}$ for some function class $\mathcal{Q} \subset \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$, and the inequality holds as long as $Q_P^\pi \in conv(\mathcal{Q})$ with $conv(\cdot)$ being the convex hull.

Approximate $d_{P_{tr}}^\pi \times P_{te}$: The remaining difficulty is that we will need samples from $d_{P_{tr}}^\pi \times P_{te}$, i.e., (s,a) sampled from π's occupancy in the P_{tr} *simulator*, and the next s' generated in the P_{te} *real environment*. While there is no natural dataset for such a distribution, we can take the data from the real environment, $(s,a,s') \sim \mu \times P_{te}$, and reweight it using $\beta = d_{P_{tr}}^\pi / \mu$ to approximate expectation w.r.t. $d_{P_{tr}}^\pi \times P_{te}$, i.e.,

$$(s,a,s') \sim d_{P_{tr}}^\pi \times P_{te} \iff (s,a,s') \sim \mu \times P_{te} \text{ reweighted with } \beta := d_{P_{tr}}^\pi / \mu.$$

Based on such an observation, we can further upper-bound $|\mathbb{E}_{(s,a)\sim d_{P_{tr}}^\pi, r\sim R(s,a)}[w \cdot r] - J_{P_{te}}(\pi)|$ from end of Equation 4 with:

$$\sup_{q\in\mathcal{Q}} L_w(w,\beta,q) := |\mathbb{E}_\mu[w \cdot \beta \cdot (q(s,a) - \gamma q(s',\pi))] - (1-\gamma)\mathbb{E}_{s\sim d_0}[q(s,\pi)]|. \quad (5)$$

As our derivation has shown, this is a valid upper bound of the error as long as $conv(\mathcal{Q})$ can represent $Q_{P_{te}}^\pi$. We also need to show that the upper bound is non-trivial, i.e., when $w = w_{P_{te}/P_{tr}}^\pi$, the upper bound should be 0. This is actually easy to see, as for any q:

$$L(w_{P_{te}/P_{tr}}^\pi, \beta, q) := |\mathbb{E}_{d_{P_{tr}}^\pi \times P_{te}}[w_{P_{te}/P_{tr}}^\pi \cdot (q(s,a) - \gamma q(s',\pi))] - (1-\gamma)\mathbb{E}_{s\sim d_0}[q(s,\pi)]| \quad (6)$$
$$= |\mathbb{E}_{d_{P_{te}}^\pi \times P}[q(s,a) - \gamma q(s',\pi)] - (1-\gamma)\mathbb{E}_{s\sim d_0}[q(s,\pi)]| = 0. \quad (7)$$

The last step directly follows from the fact that $d_{P_{te}}^\pi$ is a valid discounted occupancy and obeys the Bellman flow equation. Therefore, it makes sense to search for w over a function class $\mathcal{W} \subset \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ to minimize the loss $\sup_{q\in\mathcal{Q}} L(w,\beta,q)$.

Final estimator: To summarize our estimation procedure, we will first use [23] to estimate $\hat{\beta} \approx d_{P'}^\pi / \mu$ with a function class $\mathcal{F}$, and plug the solution into our loss for estimating $w_{P_{te}/P_{tr}}^\pi$, i.e.,

$$\hat{w} = \arg\min_{w\in\mathcal{W}} \sup_{q\in\mathcal{Q}} L_w(w,\hat{\beta},q). \quad (8)$$

As mentioned above, if the reward function is known, we can use $\mathbb{E}_{(s,a)\sim d_{P_{tr}}^\pi, r\sim R(s,a)}[\hat{w} \cdot r]$ as our estimation of $J_{P_{te}}(\pi)$. We can also demonstrate interesting properties of our optimization like the effect of a linear function class and RKHS function class. This discussion is deferred to the supplementary materials section 9.1.

Sample Complexity Guarantee: We can further provide an upper-bound on the performance of our final estimator under the following two assumptions.

5

Assumption 1 (Boundedness). *We assume $\forall f \in \mathcal{F}$, $0 < C_{\mathcal{F},\min} \leq f \leq C_{\mathcal{F},\max}$. Define $C_{\mathcal{F}} := C_{\mathcal{F},\max} + \max(\log C_{\mathcal{F},\max}, -\log C_{\mathcal{F},\min})$. Similarly, $\forall w \in \mathcal{W}$, $w \in [0, C_{\mathcal{W}}]$, and $\forall q \in \mathcal{Q}$, $q \in [0, C_{\mathcal{Q}}]$.*

Assumption 2 (Realizability of $\mathcal{F}$). *$d_{P'}^{\pi}/\mu \in \mathcal{F}$.*

Theorem 4.1. *Let $\hat{\beta}$ be our approximation of $d_{P'}^{\pi}/\mu$ which we found using [23]. We utilize this $\hat{\beta}$ to further optimize for $\hat{w}_n$ (equation 8) using n samples. In both cases, $\mathbb{E}_{(s,a)\sim d_{P_{tr}}^{\pi}}[\cdot]$ is also approximated with n samples from the simulator P_{tr}. Then, under Assumptions 1 and 2 along with the additional assumption that $Q_{P_{te}}^{\pi} \in C(\mathcal{Q})$ with probability at least $1 - \delta$, the Off Environment Evaluation error can be bounded as*

$$\left| \mathbb{E}_{(s,a)\sim d_{P_{tr}}^{\pi}, r\sim R(s,a)}[\hat{w}_n \cdot r] - J_{P_{te}}(\pi) \right| \leq \min_{w\in\mathcal{W}} \max_{q\in\mathcal{Q}} |L_w(w, \beta, q)|$$

$$+ 2C_{\mathcal{W}} \cdot C_{\mathcal{Q}} \cdot \tilde{O}\left(\sqrt{ \left\| \frac{d_{P'}^{\pi}}{\mu} \right\|_{\infty} \cdot \left(4\mathbb{E}\mathcal{R}_n(\mathcal{F}) + C_{\mathcal{F}}\sqrt{\frac{2\log(\frac{2}{\delta})}{n}} \right) } \right) \tag{9}$$

$$+ 2\mathcal{R}_n(\mathcal{W}, \mathcal{Q}) + C_{\mathcal{Q}}C_{\mathcal{W}}\sqrt{\frac{\log(\frac{2}{\delta})}{2n}}$$

where $\mathcal{R}_n(\mathcal{F}), \mathcal{R}_n(\mathcal{W}, \mathcal{Q})$ are the Radamacher complexities of function classes $\{(x,y) \to f(x) - \log(f(y)) : f \in \mathcal{F}\}$ and $\{(s,a,s') \to (w(s,a) \cdot \frac{d_{P'}^{\pi}(s,a)}{\mu(s,a)} \cdot (q(s,a) - \gamma q(s', \pi)) : w \in \mathcal{W}, q \in \mathcal{Q}\}$, respectively, $\|d_{P'}^{\pi}/\mu\|_{\infty} := \max_{s,a} d_{P'}^{\pi}(s,a)/\mu(s,a)$ measures the distribution shift between $d_{P'}^{\pi}$ and μ, and $\tilde{O}(\cdot)$ is the big-Oh notation suppressing logarithmic factors.

Note that we do not make realizability assumption for $\mathcal{W}$ in the theorem above. Realizability assumption is reflected in the $\inf_{w\in\mathcal{W}} \sup_{q\in\mathcal{Q}} |L(w, \beta, q)|$, which equals 0 when $d_{P_{te}}^{\pi}/d_{P_{tr}}^{\pi} \in \mathcal{W}$. The remaining terms vanishes to 0 at an $O(1/\sqrt{n})$ rate when $n \to \infty$.

Generalizing Off-Environment Policy Evaluation: A key advantage of our two-step approach is that we can improve many existing off-policy evaluation algorithm with a similar two-step process. In this work, we use our two-step procedure with GradientDICE—which is an empirically state-of-the-art estimator in the DICE family—can also be similarly adapted as below, which we use in our experiments. Detailed derivation for the same can be found in the supplementary materials 9.3.

5 Experiments

5.1 Sim2Sim Validation of β-DICE

Experimental Setting: In the Sim2Sim experiments, we aim to show the effectiveness of our approach across different target policies, offline dataset as well as changing sim2sim gap. We further show the effectiveness of our approach over different types of Sim2Sim environments like Taxi (Tabular), Cartpole (discrete-control), Reacher (continuous control), and HalfCheetah (continuous control) environments. For each of these environments, we refer to the default configurations of

Table 1: Log mean squared error between the performance predicted by our β-DICE algorithm and the real world performance of the robot. We observe that our method is able to outperform DICE based baselines by a comfortable margin.

	Log$_{10}$ Mean Squared Error ($\downarrow$)				
Algorithm	Kinova (Sim2Real)	Taxi	Cartpole	Reacher	Half-Cheetah
Simulator	-3.96	-0.19	-2.58	-1.09	1.18
β-DICE (Ours)	**-4.38**	**-1.60**	**-4.19**	**-4.08**	**-3.42**
GenDICE	-3.48	-0.13	-2.84	-2.61	-2.96
GradientDICE	-3.49	-0.59	-1.45	-3.17	-2.16
DualDICE	-3.48	-0.48	-0.99	-2.88	-2.12
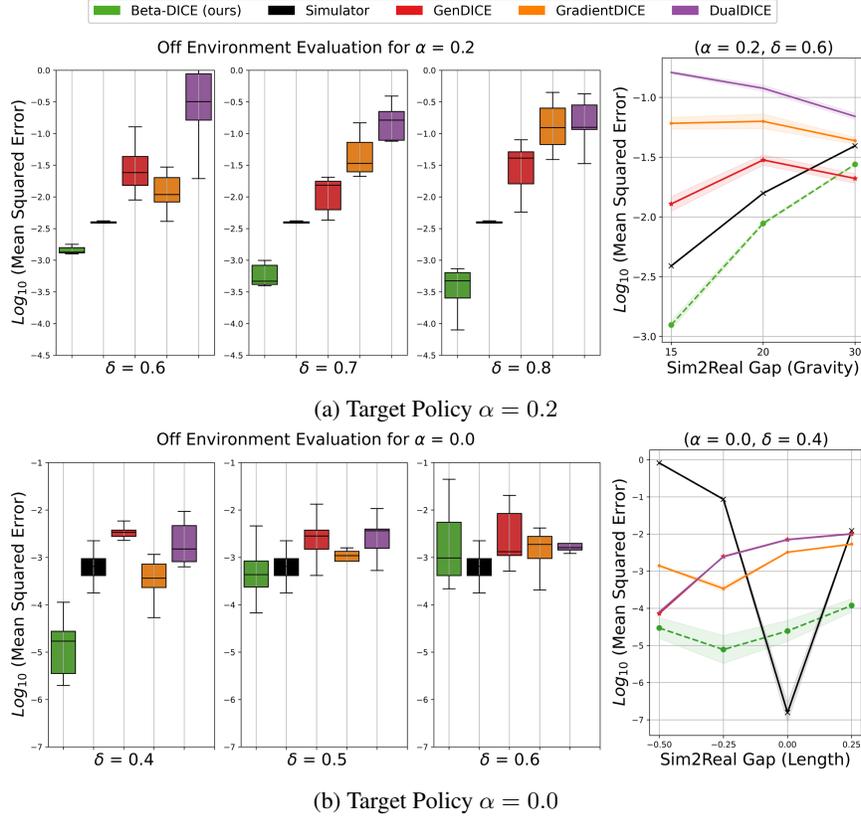
Figure 2: We demonstrate the effectiveness of β-DICE on Cartpole (a) and Reacher (b) Sim2Sim environment. For Cartpole environment we demonstrate the performance of β-DICE over for a Sim2Sim pair of $\{10, 15\}m/s^2$. Similarly for Reacher the Sim2Sim pair is (0.1m, 0.075m) for the length of the link. On left hand side, we demonstrate the effect of β-DICE with different data collection policies while keeping the target policies fixed. On the right hand side, we demonstrate the impact of β-DICE with increasing Sim2Sim gap keeping the offline data collection policy the same. We observe that our β-DICE algorithm comfortably outperforms closest DICE based baselines.

these environments as the simulator environment. We further create a "real" world environment by changing key configurations from each of these environments. For example, we modify the transition probability in taxi, gravity in cartpole, and link lengths in reacher. These kinds of configuration changes help us assess the performance limits of our algorithm across a variety of sim2sim gap. In table 2, we list all the different sim2sim environments configurations over which we experimented our algorithm. Typically these configurations are such that the real-world performance predicted by the simulator alone is off by 9-45%

We first collect our offline data by using a noisy pre-trained policy which is parameterised by δ. Higher the δ, noisier the data-collection policy. For Taxi and Cartpole environment, we using a uniform random policy for the noise, while we choose zero-mean gaussian policy for continuous environments like reacher and halfcheetah. Using this offline data as well as our simulator, we now evaluate the performance of any target policy, which we parameterise by α. Target policy is further defined by a mixture of another pre-trained policy with noise. More the α, more the randomness in the policy. Detailed experimental details along with the setup has been detailed in Appendix 9.11

Results and Observations: We present the detailed results for all the four environments in figures 3 (Taxi), figures 2a, 4 (Cartpole), figures 2b and 6 (Reacher) and figure 7 (HalfCheetah). For the boxplot, we fix target policy (α) and demonstrate the evaluation error for our algorithm across a range of offline dataset (δ) while keeping Sim2Sim gap fixed. For the line plots, we demonstrate the effectiveness of our algorithm across a changing Sim2Sim gap, while keeping the offline

data (δ) and target policy α fixed. We also compare our algorithm against DICE baselines GradientDICE [36], GenDICE [37], DualDICE [35]. DICE baselines are currently the state-of-the-art algorithm in off-policy evaluation and are known to outperform even hybrid off-policy evaluation algorithms. In figure 5, we also compare our algorithm against hybrid off-policy evaluation baselines for the cartpole environment (further details in section 9.11). We observe that our method is able to comfortably outperform closest DICE baselines with the help of extra simulator. These results not only empirically validate the effectiveness of our algorithm, but also point out that we can learn important information from imperfect simulated environments to help in improving RL policies. We also observe that as the Sim2Sim gap increases the performance of our algorithm tends to decline. This means that with increasing Sim2Sim gap the amount of relevant information that can be learned from the simulator diminishes. We observe that this decline actually becomes significant when the Sim2Sim gap breaches the 60% threshold.

5.2 Real-world performance validation on Kinova Robotic Arm

Experimental Setting: We demonstrate the effectiveness of our β-DICE algorithm for a sim2real validation task on a Kinova robotic arm. We first collect offline data by asking users to move the arm from one-position to another via RC controllers. Our data collection ensures sufficient coverage of the robotic arm's task space. We then use this offline data along with our in-house gazebo based simulator to experimentally validate the real-world performance of a PID controller using β-DICE that moves our robot from a given initial location to any desired location.

Results and Observations: Our results along with different baselines are averaged over 10 different locations are tabulated in Table 1. We observe that β-DICE is able to outperform state-of-the-art showing an improvement of 60% over the nearest baseline. There are two key conclusions from all of our experiments. One, although β-DICE outperforms state-of-the art baselines in off-policy evaluation. We observe that the performance drops when the gap between the target policy and behavior policy increases. Two, prediction error decreases as the gap between the training and test environments increases, as the transferable information between the two environments decreases.

6 Limitations

We present the limitations of our work that we wish to address in future work. (1) Our algorithm expects high quality data with sufficient coverage of the state-action space. Identifying the confidence interval of our estimator w will not only ensure sample efficient evaluation, but also help us in designing robust offline reinforcement learning algorithm. (2) Similar to DICE class of min-max optimization, our algorithm also suffers from high variance in their performance. Efforts are required to reduce this variance.

7 Conclusion and Future Work

We derive a novel MIS estimator that is able to evaluate real world performance of a robot using offline data and an imperfect robot simulator. We then develop sample complexity bounds, and empirically validate our approach on diverse Sim2Sim environments and Sim2Real environment like KinovaGen3 robot. For future work, we wish to utilize this framework of off-environment evaluation to learn optimal robot policies using simulation and a limited amount of real-world offline data.

8 Acknowledgements

References

[1] C. K. John Schulman, Barret Zoph et al. Chatgpt: Optimizing language models for dialogue, 2023.

[2] D. Silver, J. Schrittwieser, K. Simonyan, and thers. Mastering the game of go without human knowledge. *Nat.*, 2017.

[3] S. Liu, P. Chang, Z. Huang, N. Chakraborty, W. Liang, J. Geng, and K. R. Driggs-Campbell. Socially aware robot crowd navigation with interaction graphs and human trajectory prediction. 2022.

[4] OpenAI, I. Akkaya, M. Andrychowicz, et al. Solving rubik's cube with a robot hand. 2019.

[5] W. Yu, V. C. V. Kumar, G. Turk, and C. K. Liu. Sim-to-real transfer for biped locomotion. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 2019.

[6] P. Katdare, S. Liu, and K. D. Campbell. Off environment evaluation using convex risk minimization. In *International Conference on Robotics and Automation, ICRA*, 2022.

[7] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, 2018.

[8] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. 2017.

[9] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 2017.

[10] Y. Du, O. Watkins, T. Darrell, P. Abbeel, and D. Pathak. Auto-tuned sim-to-real transfer. In *IEEE International Conference on Robotics and Automation, ICRA*, 2021.

[11] G. Leibovich, G. Jacob, S. Endrawis, G. Novik, and A. Tamar. Validate on sim, detect on real - model selection for domain randomization. In *International Conference on Robotics and Automation, (ICRA)*, 2022.

[12] C. Voloshin, H. M. Le, N. Jiang, and Y. Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*, 2019.

[13] D. Precup, R. S. Sutton, and S. P. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000.

[14] N. Jiang and L. Li. Doubly Robust Off-policy Value Evaluation for Reinforcement Learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48, pages 652–661, 2016.

[15] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[16] Q. Liu, L. Li, Z. Tang, and D. Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Advances in Neural Information Processing Systems*, 2018.

[17] O. Nachum and B. Dai. Reinforcement learning via fenchel-rockafellar duality. *arXiv preprint arXiv:2001.01866*, 2020.

[18] M. Uehara, J. Huang, and N. Jiang. Minimax Weight and Q-Function Learning for Off-Policy Evaluation. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.

[19] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI gym. *arXiv preprint arXiv:1606.01540*, 2016.

[20] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021.

[21] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[22] N. Jiang and J. Huang. Minimax value interval for off-policy evaluation and policy optimization. *Advances in Neural Information Processing Systems*, 33, 2020.

[23] X. Nguyen, M. J. Wainwright, and M. I. Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *CoRR*, abs/0809.0853, 2008. URL http://arxiv.org/abs/0809.0853.

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

[25] O. Nachum and B. Dai. Reinforcement learning via fenchel-rockafellar duality. *CoRR*, 2020.

[26] S. Höfer, K. E. Bekris, et al. Sim2real in robotics and automation: Applications and challenges. *IEEE Trans Autom. Sci. Eng.*, 2021.

[27] Q. Vuong, S. Vikram, H. Su, S. Gao, and H. I. Christensen. How to pick the domain randomization parameters for sim-to-real transfer of reinforcement learning policies? 2019.

[28] S. Ross, G. J. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS*, JMLR Proceedings, 2011.

[29] J. Ho and S. Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems 29*, 2016.

[30] Y. Wu, N. Charoenphakdee, H. Bao, V. Tangkaratt, and M. Sugiyama. Imitation learning from imperfect demonstration. *CoRR*, 2019.

[31] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

[32] J. Lee, W. Jeon, B. Lee, J. Pineau, and K. Kim. Optidice: Offline policy optimization via stationary distribution correction estimation. 2021.

[33] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn. COMBO: conservative offline model-based policy optimization. In *Advances in Neural Information Processing Systems 34*, 2021.

[34] L. Yu, T. Yu, J. Song, W. Neiswanger, and S. Ermon. Offline imitation learning with suboptimal demonstrations via relaxed distribution matching. In *Thirty-Seventh AAAI Conference on Artificial Intelligence*, 2023.

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

[36] S. Zhang, B. Liu, and S. Whiteson. Gradientdice: Rethinking generalized offline estimation of stationary values. 2020.

[37] R. Zhang, B. Dai, L. Li, and D. Schuurmans. Gendice: Generalized offline estimation of stationary values. In *International Conference on Learning Representations (ICLR)*, 2019.

[38] Z. Tang, Y. Feng, L. Li, D. Zhou, and Q. Liu. Doubly robust bias reduction in infinite horizon off-policy estimation. *arXiv preprint arXiv:1910.07186*, 2019.

[39] J. Fu, M. Norouzi, O. Nachum, G. Tucker, Z. Wang, A. Novikov, M. Yang, M. R. Zhang, Y. Chen, A. Kumar, et al. Benchmarks for deep off-policy evaluation. *arXiv preprint arXiv:2103.16596*, 2021.

[40] C. Voloshin, N. Jiang, and Y. Yue. Minimax model learning. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021.

[41] M. R. Bruce Hajek. Ece 543: Statistical learning theory, 2018.

[42] P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. In *14th Annual Conference on Computational Learning Theory, COLT*, 2001.

[43] M. Wulfmeier, I. Posner, and P. Abbeel. Mutual alignment transfer learning. *CoRR*, 2017.

[44] B. Eysenbach, S. Chaudhari, S. Asawa, S. Levine, and R. Salakhutdinov. Off-dynamics reinforcement learning: Training for transfer with domain classifiers. In *9th International Conference on Learning Representations, ICLR*, 2021.

[45] Z. Zhu, K. Lin, and J. Zhou. Transfer learning in deep reinforcement learning: A survey. *CoRR*, 2020.

[46] C. Packer, K. Gao, J. Kos, P. Krähenbühl, V. Koltun, and D. Song. Assessing generalization in deep reinforcement learning, 2018.

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