GradientDICE: Rethinking Generalized Offline Estimation of Stationary Values

Shangtong Zhang [1] **Bo Liu** [2] **Shimon Whiteson** [1]

Abstract

We present GradientDICE for estimating the density ratio between the state distribution of the target policy and the sampling distribution in off-policy reinforcement learning. GradientDICE fixes several problems of GenDICE (Zhang et al., 2020a), the state-of-the-art for estimating such density ratios. Namely, the optimization problem in GenDICE is *not* a convex-concave saddle-point problem once nonlinearity in optimization variable parameterization is introduced to ensure positivity, so any primal-dual algorithm is *not* guaranteed to converge or find the desired solution. However, such nonlinearity is essential to ensure the consistency of GenDICE even with a tabular representation. This is a fundamental contradiction, resulting from GenDICE's original formulation of the optimization problem. In GradientDICE, we optimize a different objective from GenDICE by using the Perron-Frobenius theorem and eliminating GenDICE's use of divergence, such that nonlinearity in parameterization is not necessary for GradientDICE, which is provably convergent under linear function approximation.

1. Introduction

A key challenge in reinforcement learning (RL, Sutton & Barto 2018) is off-policy evaluation (Precup et al., 2001; Maei, 2011; Jiang & Li, 2015; Sutton & Barto, 2018; Liu et al., 2018; Nachum et al., 2019; Zhang et al., 2020a), where we want to estimate the performance of a target policy (average reward in the continuing setting or expected total discounted reward in the episodic setting (Puterman, 2014)), from data generated by one or more behavior policies. Compared with on-policy evaluation (Sutton, 1988), which requires data generated by the target policy, off-policy evaluation is more flexible. We can evaluate a new policy with existing data in a replay buffer (Lin, 1992) without interacting with the environment again. We can also evaluate multiple target policies simultaneously when following a single behavior policy (Sutton et al., 2011).

One major challenge in off-policy evaluation is dealing with distribution mismatch: the state distribution of the target policy is different from the sampling distribution. This mismatch leads to divergence of the off-policy linear temporal difference learning algorithm (Baird, 1995; Tsitsiklis & Van Roy, 1997). Precup et al. (2001) address this issue with products of importance sampling ratios, which, however, suffer from a large variance. To correct for distribution mismatch without incurring a large variance, Hallak & Mannor (2017); Liu et al. (2018) propose to directly learn the density ratio between the state distribution of the target policy and the sampling distribution using function approximation.[1] Intuitively, this learned density ratio is a "marginalization" of the products of importance sampling ratios.

The density ratio learning algorithms from Hallak & Mannor (2017); Liu et al. (2018) require data generated by a *single known* behavior policy. Nachum et al. (2019) relax this constraint in DualDICE, which is compatible with *multiple unknown* behavior policies and *offline* training. DualDICE, however, copes well with only the total discounted reward criterion and cannot be used under the average reward criterion. In particular, DualDICE becomes unstable as the discounting factor grows towards 1 (Zhang et al., 2020a). To address the limitation of DualDICE, Zhang et al. (2020a) propose GenDICE, which is compatible with multiple unknown behavior policies and offline training *under both criteria*. Zhang et al. (2020a) show empirically that GenDICE achieves a new state-of-the-art in off-policy evaluation.

In this paper, we point out key problems with GenDICE. In particular, the optimization problem in GenDICE is *not* a convex-concave saddle-point problem (CCSP) once nonlinearity in optimization variable parameterization is introduced to ensure positivity, so any primal-dual algorithm is not guaranteed to converge or find the desired solution

[1]University of Oxford [2]Auburn University. Correspondence to: Shangtong Zhang <shangtong.zhang@cs.ox.ac.uk>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

[1]Such density ratios are referred to as *stationary values* in Zhang et al. (2020a)

even with tabular representation. However, such positivity is essential to ensure the consistency of GenDICE. This is a fundamental contradiction, resulting from GenDICE's original formulation of the optimization problem.

Furthermore, we propose GradientDICE, which overcomes these problems. GradientDICE optimizes a different objective from GenDICE by using the Perron-Frobenius theorem (Horn & Johnson, 2012) and eliminating GenDICE's use of divergence. Consequently, nonlinearity in parameterization is not necessary for GradientDICE, which is provably convergent under linear function approximation. Finally, we provide empirical results demonstrating the advantages of GradientDICE over GenDICE and DualDICE.

2. Background

We use vectors and functions interchangeably when this does not confuse. For example, let $f : X \quad Y \; ! \; \mathbb{R}$ be a function; we also use f to denote the corresponding vector in $\mathbb{R}^{jXj \; jYj}$. All vectors are column vectors.

We consider an infinite-horizon MDP with a finite state space S, a finite action space A, a transition kernel $p : S \quad S \quad A \; ! \; [0; 1]$, a reward function $r : S \quad A \; ! \; \mathbb{R}$, a discount factor $\quad 2 \; [0; 1)$, and an initial state distribution $\sim_0$. The initial state S_0 is sampled from $\sim_0$. At time step t, an agent at S_t selects an action A_t according to $\quad (\; jS_t)$, where $\quad : A \quad S \; ! \; [0; 1]$ is the policy being followed by the agent. The agent then proceeds to the next state S_{t+1} according to $p(\; jS_t; A_t)$ and gets a reward R_{t+1} satisfying $\mathbb{E}[R_{t+1}] = r(S_t; A_t)$.

Similar to Zhang et al. (2020a), we consider two performance metrics for the policy $\quad$: the total discounted reward $\quad (\;) \doteq (1 \quad) \mathbb{E}[\sum_{t=0}^{1} \; {}^t R_{t+1}]$ and the average reward $\quad (\;) \doteq \lim_{T \; ! \; 1} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{T} R_t]$. When the Markov chain induced by $\quad$ is ergodic, $\quad (\;)$ is always well defined (Puterman, 2014). Throughout this paper, we implicitly assume this ergodicity whenever we consider $\quad (\;)$. When considering $\quad (\;)$, we are interested in the normalized discounted state-action occupation $d \; (s; a)$. Let $d_t \; (s; a) \doteq \Pr(S_t = s; A_t = a j \sim_0; p; \quad)$ be the probability of occupying the state-action pair $(s; a)$ at the time step t following $\quad$. Then, we have $d \; (s; a) \doteq (1 \quad) \sum_{t=0}^{1} \; {}^t d_t \; (s; a)$. When considering $\quad (\;)$, we are interested in the stationary state-action distribution $d \; (s; a) \doteq \lim_{t \; ! \; 1} d_t \; (s; a)$. To simplify notation, we extend the definition of $\quad (\;)$ and $d \; (s; a)$ from $\quad 2 \; [0; 1)$ to $\quad 2 \; [0; 1]$ by defining $\quad_1(\;) \doteq \quad (\;)$ and $d_1(s; a) \doteq d(s; a)$. It follows that for any $\quad 2 \; [0; 1]$, we have $\quad (\;) = \mathbb{E}_{(s;a) \quad d} \; [r(s; a)]$.

We are interested in estimating $\quad (\;)$ without executing the policy $\quad$. Similar to Zhang et al. (2020a), we assume access to a fixed dataset $D \doteq f(s_i; a_i; r_i; s_i^0)g_{i=1}^N$. Here the state-action pair $(s_i; a_i)$ is sampled from an unknown distribution $d \quad : S \quad A \; ! \; [0; 1]$, which may result from multiple unknown behavior policies. The reward r_i satisfies $\mathbb{E}[r_i] = r(s_i; a_i)$. The successor state s_i^0 is sampled from $p(\; js_i; a_i)$. As $\quad (\;) = \mathbb{E}_{(s;a) \quad d} \; [\frac{d \; (s;a)}{d \; (s;a)} r(s; a)]$, one possible approach for estimating $\quad (\;)$ is to learn the density ratio $\quad (s; a) \doteq \frac{d \; (s;a)}{d \; (s;a)}$ directly.

We assume $d \; (s; a) > 0$ for every $(s; a)$ and use $D \; 2 \; \mathbb{R}^{N_{sa} \quad N_{sa}} (N_{sa} \doteq jSj \quad jAj)$ to denote a diagonal matrix whose diagonal entry is d . Let $\quad_0(s; a) \doteq \sim_0(s) \; (a j s)$, Zhang et al. (2020a) show

$$D \quad = T \quad ;$$

where the operator T is defined as

$$T y \doteq (1 \quad) \; _0 + \quad P^{>} D y;$$

and $P \quad 2 \; \mathbb{R}^{N_{sa} \quad N_{sa}}$ is the state-action pair transition matrix, i.e., $P \; ((s; a); (s^0; a^0)) \doteq p(s^0 j s; a) \; (a^0 j s^0)$. The operator T is similar to the Bellman operator but in the reverse direction. Similar ideas have been explored by Wang et al. (2007; 2008); Hallak & Mannor (2017); Liu et al. (2018); Gelada & Bellemare (2019). As $D \quad$ is a probability measure, Zhang et al. (2020a) propose to compute $\quad$ by solving the following optimization problem:

$$\min_{\; 2 \mathbb{R}^{N_{sa}}; \quad \; _0} D \; (T \quad j j D \;) \quad \text{s.t.} \; d^{>} \quad = 1; \qquad (1)$$

where $\quad 0$ is elementwise greater or equal, $D \quad$ is an f-divergence (Nowozin et al., 2016) associated with a convex, lower semi-continuous generator function $\quad : \mathbb{R}_+ \; ! \; \mathbb{R}$ with $\quad (1) = 0$. Let $q_1; q_2$ be two probability measures; we have $D \; (q_1 j j q_2) \doteq \sum_x q_2(x) \; (\frac{q_1(x)}{q_2(x)})$. The f-divergence is used mainly for the ease of optimization but see Zhang et al. (2020a) for discussion of other possible divergences. Due to the difficulty in solving the constrained problem (1) directly, Zhang et al. (2020a) propose to solve the following problem instead:

$$\min_{\; 2 \mathbb{R}^{N_{sa}}; \quad \; _0} D \; (T \quad j j D \;) + \tfrac{}{2} (d^{>} \quad 1)^2; \qquad (2)$$

where $\quad > 0$ is a constant. We have

Lemma 1. *(Zhang et al., 2020a) For any constant $\quad > 0$, $\quad$ is optimal for the problem (2) iff $\quad = \quad$.*

To make the optimization tractable and address a double sampling issue, Zhang et al. (2020a) rewrite $\quad (x)$ as $\quad (x) = \max_{f \; 2 \mathbb{R}} x f \quad \frac{1}{2} \quad (f)$, where $\quad$ is the Fenchel conjugate of $\quad$, and use the interchangeability principle for interchanging maximization and expectation (Shapiro et al., 2014; Dai et al., 2016), yielding the following problem, which is equivalent to (2):

$$\min_{\; 2 \mathbb{R}^{N_{sa}}; \quad 0} \quad \max_{f \; 2 \mathbb{R}^{N_{sa}}; \; 2 \mathbb{R}} J(\; ; f; \quad); \qquad (3)$$

where

$$J(\ ; f;\ ) \doteq (1\quad)\mathsf{E}_{\ 0}[f(s;a)] + \quad \mathsf{E}_p[\ (s;a)f(s^0;a^0)] \quad \mathsf{E}_d\ [\ (s;a)\quad (f(s;a))] + \quad \mathsf{E}_d\ [\quad (s;a)\quad\ ]\quad \frac{^2}{2}\ :$$

Here $\mathsf{E}_{\ 0}; \mathsf{E}_d\ ; \mathsf{E}_p$ are shorthand for $\mathsf{E}_{(s;a)\quad 0}, \mathsf{E}_{(s;a)\quad d}$, $\mathsf{E}_{(s;a)\quad d\ ;s^0\quad p(\ js;a);a^0\quad (\ js^0)}$ respectively. Zhang et al. (2020a) show J is convex in $\quad$ and concave in $\ ; f$, i.e., (3) is a convex-concave saddle-point problem. Zhang et al. (2020a) therefore use a primal-dual algorithm (i.e, perform stochastic gradient ascent on $\ ; f$ and stochastic gradient descent on $\ $) to find the saddle-point, yielding GENeralized stationary DIstribution Correction Estimation (GenDICE).

3. Problems with GenDICE

3.1. Use of Divergences as the Objective

Zhang et al. (2020a) proposes to consider a family of divergences, the f-divergences. However, f-divergences are defined between probability measures. So $D\ $ in (2) implicitly requires its arguments to be valid probability measures. Consequently, (2) still has the implicit constraint that $d^{>}\ = 1$. The main motivation for Zhang et al. (2020a) to transform (1) into (2) is to get rid of this equality constraint. By using divergences, they do not really get rid of it. When this implicit constraint is considered, the problem (2) is still hard to optimize, as discussed in Zhang et al. (2020a).

We can, of course, just ignore this implicit constraint and interpret $D\ $ as a generic function instead of a divergence. Namely, we do not require its arguments to be valid probability measures. In this scenario, however, there is no guarantee that $D\ $ is always nonnegative, which plays a central role in proving Lemma 1's claim that GenDICE is consistent. Consider, for example, the KL-divergence, where $\ (x) = x\log(x)$. If $q_2(x) > q_1(x) > 0$ holds for all x (which is impossible when q_1 and q_2 are probability measures), clearly we have $D\ (q_1 jj q_2) < 0$. While Zhang et al. (2020a) propose not to use KL divergence due to numerical instability, here we provide a more principled explanation that if KL divergence is used, Lemma 1 does not necessarily hold. Zhang et al. (2020a) propose to use $\ ^2$-divergence instead. Fortunately, $\ ^2$-divergence has the property that $q_1(x) > 0 \wedge q_2(x) > 0$ implies $D\ (q_1 jj q_2)\quad 0$, even if q_1 and q_2 are not probability measures. This property ensures Lemma 1 holds even we just consider $D\ $ as a generic function instead of a divergence. But not all divergences have this property. Moreover, even if $\ ^2$-divergence is considered, $\quad 0$ is still necessary for Lemma 1 to hold. This requirement ($\quad 0$) is also problematic, as discussed in the next section.

To summarize, we argue that *f-divergence is not a good choice to form the optimization objective for density ratio learning*.

3.2. Use of Primal-Dual Algorithms as the Solver

We assume $\ , f$ are parameterized by $\ ^{(1)}; \ ^{(2)}$. As (3) requires $\ (s;a)\quad 0$, Zhang et al. (2020a) propose to add extra nonlinearity, e.g., $(\ )^2; \log(1 + \exp(\ ))$, or $\exp(\ )$, in the parameterization of $\ $. Plugging the approximation in (3) yields $\min_{\ ^{(1)}} \max_{\ ^{(2)};\ } J(\ _{\ ^{(1)}}; \ ; f_{\ ^{(2)}})$. Here $\ ^{(1)}$ and $f_{\ ^{(2)}}$ emphasize that $\ $ and f are parameterized functions.

There is now a contradiction. On the one hand, $J(\ _{\ ^{(1)}}; \ ; f_{\ ^{(2)}})$ is *not* necessarily CCSP when nonlinearity is introduced in the parameterization. In the definition of J in (3), the sign of $\ $ depends on f and $\ $. Unless $\ $ is linear in $\ ^{(1)}$, the convexity of J w.r.t. $\ ^{(1)}$ is in general hard to analyze (Boyd & Vandenberghe, 2004), even we just add $(\ )^2$ after a linear parameterization. Although Zhang et al. (2020a) demonstrate great empirical success from a primal-dual algorithm, this optimization procedure is *not* theoretically justified as $J(\ _{\ ^{(1)}}; \ ; f_{\ ^{(2)}})$ is not necessarily a convex-concave function. On the other hand, if we do not apply any nonlinearity in $\ ^{(1)}$, there is no guarantee that $_{\ ^{(1)}}(s;a) > 0$ even with a tabular representation. Then Lemma 1 does not necessarily hold, and GenDICE is not necessarily consistent.

To summarize, *applying the primitive primal-dual algorithm for the GenDICE objective is not theoretically justified, even with a tabular representation*.

3.3. Projection and Self-Normalization

Besides applying nonlinearity, one may also consider projection to account for the constraint $_{\ ^{(1)}}(s;a) > 0$, i.e., we project $\ ^{(1)}$ back to $\doteq f\ ^{(1)} j\ _{\ ^{(1)}}(s;a) > 0 g$. One direct instantiation of this idea is Projected Stochastic Gradient Descent (PSGD). With nonlinear function approximation, it is not clear how to achieve this. With tabular or linear representation, such projection reduces to inequality-constrained quadratic programming, solving which usually involves subroutine numerical optimization, making it very computationally expensive. Moreover, the number of constraints grows linearly w.r.t. the number of states (not state features), indicating such projection does not scale well. Stochastic Mirror Descent (SMD, Beck & Teboulle 2003) with generalized KL-divergence is also a possible way to to achieve such a projection. SMD, as well as PSGD, walks $\ ^{(1)}$ in the positive orthant. To ensure $_{\ ^{(1)}}(s;a) > 0$, they also require positive features. However, it is possible that the oracle lies in the feature space but the optimal $\ ^{(1)}$ does not lie in the positive orthant, indicating SMD and PSGD can yield arbitrarily large optimization errors.

Self-normalization (Liu et al., 2018; Zhang et al., 2020a) is also an approach to ensure nonlinearity, where we normalize the prediction over all possible state-action pairs. Recently, Mousavi et al. (2020) propose an objective with Maximum Mean Discrepancy and use self-normalization. However, self-normalization usually generates biased solutions and is computationally prohibitive for large datasets (see Appendix E.3 in Zhang et al. (2020a)). Moreover, self-normalization in general yields non-convex objectives even with tabular or linear representation (e.g., the objective in Mousavi et al. (2020) is non-convex), rendering difficulties in optimization.

To summarize, we argue that *neither projection nor self-normalization is a theoretically justified solution for the problems of GenDICE.*

3.4. A Hard Example for GenDICE

Figure 1. A single-state MDP.

We now provide a concrete example to demonstrate the defects of GenDICE. We consider a single state MDP (Figure 1) with two actions, both of which lead to that single state. We set $= 1; d(s_0; a_1) = d(s_0; a_2) = (a_1 j s_0) = (a_2 j s_0) = 0.5$. Therefore, we have $_0(s_0; a_1) = _0(s_0; a_2) = 0.5$. Under this setting, it is easy to verify that $= [1; 1]^>$. We now instantiate (3) with a ^{2}-divergence and $= 1$ as recommended by Zhang et al. (2020a), where $(x) = x + \frac{x^2}{4}$. To solve (3), we need 0. As suggested by Zhang et al. (2020a), we use $(\ )^2$ nonlinearity. Namely, we define $(s_0; a_1) = _1^2; (s_0; a_2) = _2^2; f(s_0; a_1) = f_1; f(s_0; a_2) = f_2$. Now our optimization variables are $_1; _2; f_1; f_2; $. It is easy to verify that at the point $(_1; _2; f_1; f_2;) = (0; 0; 0; 0; 1)$, we have $\frac{@J(\ ;f;\ )}{@\ _1} = \frac{@J(\ ;f;\ )}{@\ _2} = \frac{@J(\ ;f;\ )}{@ f_1} = \frac{@J(\ ;f;\ )}{@ f_2} = \frac{@J(\ ;f;\ )}{@} = 0$, indicating GenDICE stops at this point if the true gradient is used. However, $[0; 0]^>$ is obviously not the optimum. Details of the computation are provided in the appendix. This suboptimality results from the fact that once nonlinearity is introduced in $J(\ ; f; \ )$, it is not convex-concave even with a tabular representation. Although this example is trivial to solve, it numerically verifies the problems of GenDICE.

4. GradientDICE

As discussed above, the problems with GenDICE come mainly from the formulation of (2), namely the constraint 0 and the use of the divergence D . We eliminate both by considering the following problem instead:

$$\min_{2R^{Nsa}} L(\ ) \doteq \frac{1}{2} jjT \quad D\ jj^2_{D^{\ 1}} + \frac{}{2}(d^> \quad 1)^2; \tag{4}$$

where > 0 is a constant and $jyjj^2$ stands for $y^>\ y$. Readers familiar with Gradient TD methods (Sutton et al., 2009a;b) or residual gradients (Baird, 1995) may find the first term of this objective similar to the Mean Squared Bellman Error (MSBE). However, while in MSBE the norm is induced by D, we consider a norm induced by $D^{\ 1}$. This norm is carefully designed and provides expectations that we can sample from, which will be clear once $L(\ )$ is expanded (see Eq (5) below). Remarkably, we have:

Theorem 1. *is optimal for (4) iff* $=$.

Proof. Sufficiency: Obviously is optimal.
Necessity: (a) < 1: In this case $I \quad P^>$ is nonsingular. The linear system $D \quad T = 0$ has only one solution, we must have $=$. (b) $= 1$: If is optimal, we have $d^> = 1$ and $D = P^> D$, i.e., D is a left eigenvector of P associated with the Perron-Frobenius eigenvalue 1. Note d is also a left eigenvector of P associated with the eigenvalue 1. According to the Perron-Frobenius theorem for nonnegative irreducible matrices (Horn & Johnson, 2012), the left eigenspace of the Perron-Frobenius eigenvalue is 1-dimensional. Consequently, there exists a scalar such that $D = d$. On the other hand, $= 1^> d = 1^> D = d^> = 1$, implying $D = d$, i.e., $=$. □

Remark 1. *Unlike the problem formulation in Zhang et al. (2020a) (see (1) and (2)), we do not use* 0 *as a constraint and can still guarantee there is no degenerate solution. Eliminating this constraint is key to eliminating nonlinearity. Although the Perron-Frobenius theorem can also be used in the formulation of Zhang et al. (2020a), their use of the divergence* D *still requires* 0.

With $= T \quad D$, we have

$$\begin{aligned} L(\ ) &\doteq \frac{1}{2}\mathsf{E}_d\left[\frac{(s;a)}{d\ (s;a)}^{\ 2}\right] + \frac{}{2}(d^> \quad 1)^2 \qquad (5)\\ &= \max_{f\ 2R^{Nsa}} \mathsf{E}_d\left[\frac{(s;a)}{d\ (s;a)} f(s;a) \quad \frac{1}{2} f(s;a)^2\right] \\ &\quad + \max_{2R} \mathsf{E}_d\left[\quad (s;a) \quad\right] \quad \frac{^2}{2}\ ; \end{aligned}$$

where the equality comes also from the Fenchel conjugate and the interchangeability principle as in Zhang et al.

(2020a). We, therefore, consider the following problem

$$\begin{aligned}
&\min_{\in \mathbb{R}^{N_{sa}}} \max_{f \in \mathbb{R}^{N_{sa}};\ \in \mathbb{R}} L(; ; f) \\
&\doteq \mathbb{E}_d\left[\tfrac{(s;a)}{d\ (s;a)} f(s;a) \quad \tfrac{1}{2} f(s;a)^2\right] \\
&\quad + \quad (\mathbb{E}_d[\ (s;a) \quad 1]) \quad \tfrac{^2}{2} \\
&= (1 \quad)\mathbb{E}_{\ 0}[f(s;a)] + \quad \mathbb{E}_p[\ (s;a) f(s^0;a^0)] \\
&\quad \mathbb{E}_d[\ (s;a) f(s;a)] \quad \frac{1}{2}\mathbb{E}_d[f(s;a)^2] \\
&\quad + \quad \mathbb{E}_d[\quad (s;a) \quad] \quad \tfrac{^2}{2} \quad :
\end{aligned}$$

Here the equality comes from the fact that

$$\mathbb{E}_p[\ (s;a) f(s^0;a^0)] = \sum_{s^0;a^0} (P^{\top} D \quad)(s^0;a^0) f(s^0;a^0):$$

This problem is an *unconstrained* optimization problem and L is convex (linear) in $\quad$ and concave in $f;\ \ .$ Assuming $\ ; f$ is parameterized by $w;\quad$ respectively and including ridge regularization for w for reasons that will soon be clear, we consider the following problem

$$\min_w \max_{\ ;\ } L(\ _w;\ ;f\ ) + \tfrac{}{2}\|w\|^2; \tag{6}$$

where $\quad 0$ is a constant. When a linear architecture is considered for $\ _w$ and $f\ ,$ the problem (6) is CCSP. Namely, it is convex in w and concave in $\ ;\ .$ We use $X \in \mathbb{R}^{N_{sa} \quad K}$ to denote the feature matrix, each row of which is $x(s;a)$, where $x : S \quad A \to \mathbb{R}^K$ is the feature function. Assuming $_w \doteq Xw; f \doteq X\ ,$ we perform gradient descent on w and gradient ascent on $\ ;\ .$ As we use techniques similar to Gradient TD methods to prove the convergence of our new algorithm, we term it Gradient stationary DIstribution Correction Estimation (GradientDICE):

$$\begin{aligned}
&\ _t \quad (1 \quad)x_{0;t} + \quad x_t^{\top} w_t x_t^0 \quad x_t^{\top} w_t x_t \\
&\ _{t+1} \quad \ _t + \ _t(\ _t \quad x_t^{\top} \ _t x_t) \\
&\ _{t+1} \quad \ _t + \ _t(x_t^{\top} w_t \quad 1 \quad \ _t) \\
&w_{t+1} \quad w_t \quad \ _t \quad x_t^{0\top} \ _t x_t \quad x_t^{\top} \ _t x_t + \quad \ _t x_t + \quad w_t \quad :
\end{aligned}$$

Here $x_{0;t} \doteq x(s_{0;t};a_{0;t}); x_t \doteq x(s_t;a_t); x_t^0 \doteq (s_t^0;a_t^0),$ $(s_{0;t};a_{0;t}) \quad _0; (s_t;a_t;s_t^0;a_t^0) \quad p$ (c.f. $\mathbb{E}_p$ in (3)), $f\ _t g$ is a sequence of deterministic nonnegative nonincreasing learning rates satisfying the Robin-Monro's condition (Robbins & Monro, 1951), i.e., $\sum_t \ _t = 1; \sum_t \ _t^2 < 1$.

4.1. Convergence Analysis

Let $d_t^{\top} \doteq [\ _t^{\top}; w_t^{\top};\ _t]$, we rewrite the GradientDICE updates as $d_{t+1} \doteq d_t + \ _t(G_{t+1} d_t + g_{t+1})$, where

$$\begin{aligned}
G_{t+1} &\doteq \begin{bmatrix} 2 & & & 3 \\ & x_t^{\top} x_t & (x_t \quad x_t^0) x_t^{\top} & 0 \\ & x_t(x_t^{\top} \quad x_t^{0\top}) & I & x_t 5 \\ & 0 & x_t^{\top} & \end{bmatrix}; \\
g_{t+1} &\doteq \begin{bmatrix} 2 & 3 \\ (1 \quad)x_{0;t} & \\ 4 \quad 0 & 5 \end{bmatrix}:
\end{aligned}$$

Defining $A \doteq X^{\top}(I \quad P^{\top})DX; C \doteq X^{\top} DX$, the limiting behavior of GradientDICE is governed by

$$\begin{aligned}
G &\doteq \mathbb{E}_p[G_t] = \begin{bmatrix} 2 & C & A & 0 & 3 \\ 4 A^{\top} & & I & X^{\top} d & 5 \\ 0 & & d^{\top} X & & \end{bmatrix}; \\
g &\doteq \mathbb{E}_{\ 0}[g_t] = \begin{bmatrix} 2 & 3 \\ (1 \quad)X^{\top} & _0 \\ 4 \quad 0 & 5 \end{bmatrix}:
\end{aligned} \tag{7}$$

Assumption 1. *X has linearly independent columns.*

Assumption 2. *A is nonsingular or $\quad > 0$.*

Assumption 3. *The features $\{x_{0;t}; x_t; x_t^0\}$ have uniformly bounded second moments.*

Remark 2. *Assumption 1 ensures C is strictly positive definite. When $\quad < 1$, it is common to assume A is nonsingular (Maei, 2011), the ridge regularization (i.e., $\quad > 0$) is then optional. When $\quad = 1$, A can easily be singular (e.g., in a tabular setting). We, therefore, impose the extra ridge regularization. Assumption 3 is commonly used in Gradient TD methods (Maei, 2011). Assumptions (1 - 3) are also used in previous density ratio learning literature for analyzing the optimization error, either explicitly (e.g., the Assumption 8 in Appendix D.4 in Nachum et al. (2019)) or implicitly (e.g., Appendix C.3 in Zhang et al. (2020a)).*

Theorem 2. *Under Assumptions (1-3), we have*

$$\lim_{t \to \ 1} d_t = \quad G^{\ 1} g \quad \textit{almost surely:}$$

We provide a detailed proof of Theorem 2 in the appendix, which is inspired by Sutton et al. (2009a). One key step in the proof is to show that the real parts of all eigenvalues of G are strictly negative. The G in Sutton et al. (2009a) satisfies this condition easily. However, for our G to satisfy this condition when $\quad = 1$, we must have $\quad > 0$, which motivates the use of ridge regularization.

With simple block matrix inversion expanding $G^{\ 1}$, we

have $\lim_{t \to 1} w_t = w_{1;}$, where

$$\begin{aligned} w_{1;} &\doteq (1 \quad) A^\top C^{-1} X^\top {}_0 \\ &\quad + \quad {}^1 z[1 \quad (1 \quad) z^\top A^\top C^{-1} X^\top {}_0] \\ &\doteq (I + A^\top C^{-1} A)^{-1}; \\ z &\doteq X^\top d; \quad \doteq 1 + d^\top X X^\top d : \end{aligned}$$

The maximization step in (6) is quadratic (with linear function approximation) and thus can be solved analytically. Simple algebraic manipulation together with Assumption 1 shows that this quadratic problem has a unique optimizer for all $2 [0; 1]$. Plugging the analytical solution for the maximization step in (6), the KKT conditions then state that the optimizer $w_{;}$ for the minimization step must satisfy $A_{;} w_{;} = b$, where

$$\begin{aligned} A_{;} &\doteq A^\top C^{-1} A + X^\top d \, d^\top X + I; \\ b &\doteq (1 \quad) A^\top C^{-1} X^\top {}_0 + X^\top d : \end{aligned}$$

Assumption 2 ensures $A_{;}$ is nonsingular. Using the Sherman-Morrison formula (Sherman & Morrison, 1950), it is easy to verify $w_{;} = w_{1;}$. For a quick sanity check, it is easy to verify that $w_{1;0} =$ holds when < 1, $X = I$, and $= 0$, using the fact $(1 \quad)1^\top (I \quad P^\top)_0 = 1$.

4.2. Consistency Analysis

To ensure convergence, we require ridge regularization in (6) for the setting $= 1$. The asymptotic solution $w_{1;}$ is therefore biased. We now study the regularization path consistency for the setting $= 1$, i.e., we study the behavior of $w_{1;}$ when approaches 0.

Case 1: $2 col(X)$. Here $col(\,)$ indicates the column space. As X has linearly independent columns (Assumption 1), we use w to denote the unique w satisfying $Xw =$. As $= 1$, A can be singular. Hence both $w_{1;0}$ and $A^{\dagger}_{;0}$ can be ill-defined. We now show under some regularization, we still have the desired consistency. As $A^\top C^{-1} A$ is always positive semidefinite, we consider its eigendecomposition $A^\top C^{-1} A = Q^\top Q$, where Q is an orthogonal matrix, $\doteq diag([\,{}_1; \quad ; \,{}_r; 0; \quad ; 0])$, r is the rank of $A^\top C^{-1} A$, ${}_i > 0$ are eigenvalues. Let $u \doteq Q X^\top d$, we have

Proposition 1. *Assuming $XC^{-1}X^\top$ is positive definite, $\|u_{r+1:N_{sa}}\| \neq 0$, then $\lim_{\to 0} w_{1;} = w$; where $u_{i:j}$ denotes the vector consisting of the elements indexed by $i; i+1; \ldots; j$ in the vector u.*

Proof. According to the Perron-Frobenius theorem (c.f. the proof of Theorem 1), it suffices to show

$$\begin{aligned} &\lim_{\to 0} L_1(w_{1;}) = \lim_{\to 0} L_2(w_{1;}) = 0; \\ &L_1(w_{1;}) \doteq d^\top X w_{1;} \quad 1; \\ &L_2(w_{1;}) \doteq \|D X w_{1;} \quad P^\top X D w_{1;}\|^2_{X C^{-1} X^\top}; \end{aligned}$$

as w is the only w satisfying $L_1(w) = L_2(w) = 0$. With the eigendecomposition of $A^\top C^{-1} A$, we can compute explicitly. Simple algebraic manipulation then yields

$$\begin{aligned} L_1(w_{1;}) &= \frac{u^\top \quad u}{1 + \quad u^\top \quad u} \quad 1; \\ L_2(w_{1;}) &= \frac{{}^2 u^\top \quad u}{(1 + \quad u^\top \quad u)^2} + \frac{{}^2 \quad u^\top \quad {}^2 u}{(1 + \quad u^\top \quad u)^2}; \end{aligned}$$

where $\doteq diag([\frac{1}{+\,{}_1}; \ldots; \frac{1}{+\,{}_r}; \frac{1}{}; \quad ; \frac{1}{}])$. The desired limits then follow from the L'Hopital's rule. □

Remark 3. *The assumption $\|u_{r+1:N_{sa}}\| \neq 0$ is not restrictive as it is independent of learnable parameters and mainly controlled by features. Requiring $XC^{-1}X^\top$ to be positive definite is more restrictive, but it holds at least for the tabular setting (i.e., $X = I$). The difficulty of the setting $= 1$ comes mainly from the fact that the objective of the minimization step in the problem (6) is no longer strictly convex when $= 0$ (i.e., $A_{;0}$ can be singular). Thus there may be multiple optima for this minimization step, only one of which is w. Extra domain knowledge (e.g., assumptions in the proposition statement) is necessary to ensure the regularization path converges to the desired optimum. We provide a sufficient condition here and leave the analysis of necessary conditions for future work.*

Case 2: $2 col(X)$*:* In this scenario, it is not clear how to define w. The minimization step in (6) can have multiple optima, and it is not clear which one is the best. To analyze this scenario, we need to explicitly define projection in the optimization objective like Mean Squared Projected Bellman Error (Sutton et al., 2009a), instead of using an MSBE-like objective. We leave this for future work.

4.3. Finite Sample Analysis

We now provide a finite sample analysis for a variant of GradientDICE, *Projected GradientDICE* (Algorithm 1), where we introduce projection and iterates average, akin to Nemirovski et al. (2009); Liu et al. (2015). Intuitively, Projected GradientDICE groups the $;$ in GradientDICE into $y^\top = [\,^\top; \,]$. Precisely, we have $y_t \in \mathbb{R}^{K+1}; w_t \in \mathbb{R}^K$,

$$\begin{aligned} G_{1;t} &\doteq \begin{matrix} x_t^\top x_t & 0 \\ 0 & \end{matrix}; G_{2;t} \doteq \begin{matrix} (x_t \quad x_t^{\prime})x_t^\top \\ x_t^\top \end{matrix}; \\ G_{3;t} &\doteq x_t(x_t^\top \quad x_t^{\prime\top}) \quad x_t \; ; G_{4;t} \doteq I; \\ G_{5;t} &\doteq (1 \quad) x_{0;t} \; ; \end{aligned}$$

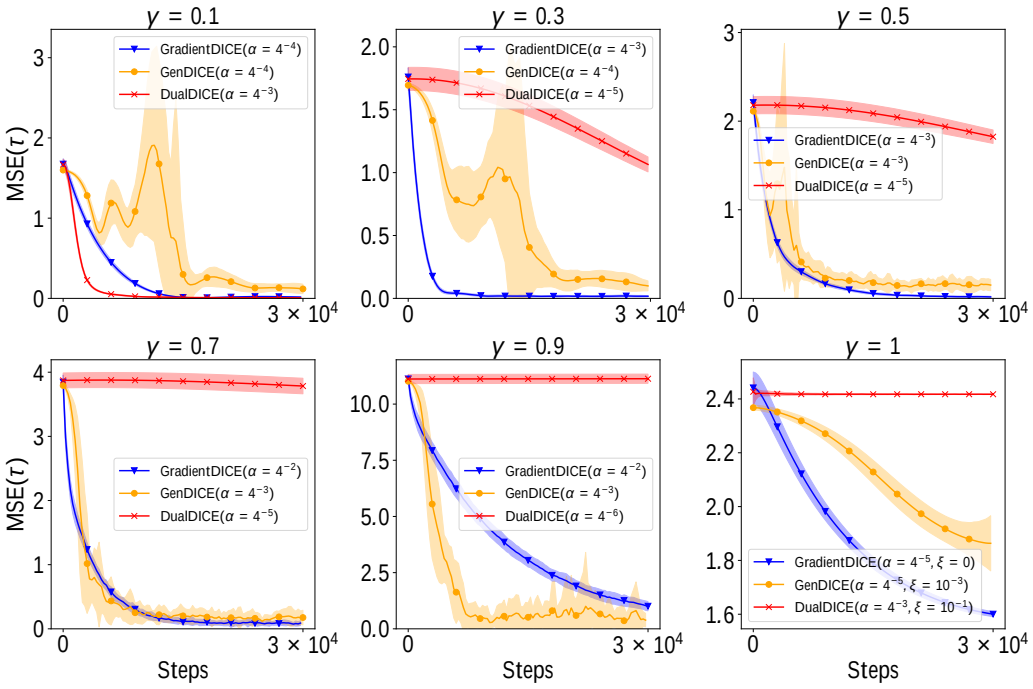

Figure 2. Density ratio learning in Boyan's Chain with a tabular representation.

Algorithm 1 Projected GradientDICE

for $t = 0;\dots;n-1$ **do**
$y_{t+1} \quad \Pi_Y[y_t + \alpha_t(G_{1;t}y_t + G_{2;t}w_t + G_{5;t})]$
$w_{t+1} \quad \Pi_W[w_t + \alpha_t(G_{3;t}y_t + G_{4;t}w_t)]$
end for
Output:
$y_n \doteq \frac{\sum_{t=1}^{n} \alpha_t y_t}{\sum_{t=1}^{n} \alpha_t}; w_n \doteq \frac{\sum_{t=1}^{n} \alpha_t w_t}{\sum_{t=1}^{n} \alpha_t}$

$Y \quad \mathbb{R}^{K+1}, W \quad \mathbb{R}^K$. Π_Y and Π_W are projections onto Y and W w.r.t. ℓ_2 norm (such projection is trivial if Y and W are balls), n is the number of iterations, and α_t is a learning rate, detailed below. We consider the following problem

$$\min_{w \in W} \max_{y \in Y} L(w;y) \doteq L(\Pi_W; \Pi_Y f) + \frac{1}{2}\|w\|^2 \quad (8)$$

It is easy to see $L(w;y)$ is a convex-concave function and its saddle point $(w^*; y^*)$ is unique. We assume

Assumption 4. *Y and W are bounded, closed, and convex, $w \in W; y \in Y$.*

For the CCSP problem (8), we define the optimization error

$$\epsilon_{\text{opt}}(w;y) \doteq \max_{y' \in Y} L(w;y') - \min_{w' \in W} L(w';y).$$

It is easy to see $L(w;y) = 0$ iff $(w;y) = (w^*;y^*)$.

Proposition 2. *Under Assumptions (1-4), for the $(w_n;y_n)$ from the Projected GradientDICE algorithm after n iterations, we have at least with probability $1-\delta$*

$$\epsilon_{\text{opt}}(w_n;y_n) \quad \frac{2}{n}(8 + 2\ln\frac{2}{\delta})C_0;$$

where $C_0 > 0$ is a constant.

Both C_0 and the learning rates α_t are detailed in the proof in the appendix. Note it is possible to conduct a finite sample analysis without introducing projection using arguments from Lakshminarayanan & Szepesvari (2018), which we leave for future work.

5. Experiments

In this section, we present experiments comparing GradientDICE to GenDICE and DualDICE. All curves are averaged over 30 independent runs and shaded regions indicate one standard derivation. The implementations are made publicly available for future research.[2]

5.1. Density Ratio Learning

We consider two variants of Boyan's Chain (Boyan, 1999) as shown in Figure 4. In particular, we use `Episodic Boyan's Chain` when < 1 and `Continuing Boyan's Chain` when $= 1$. We consider a uniform sampling distribution, i.e., $d(s;a) = \frac{1}{26}\delta(s;a)$, and a target policy satisfying $(a_0|s) = 0.1\delta s$. We design

[2] https://github.com/ShangtongZhang/DeepRL

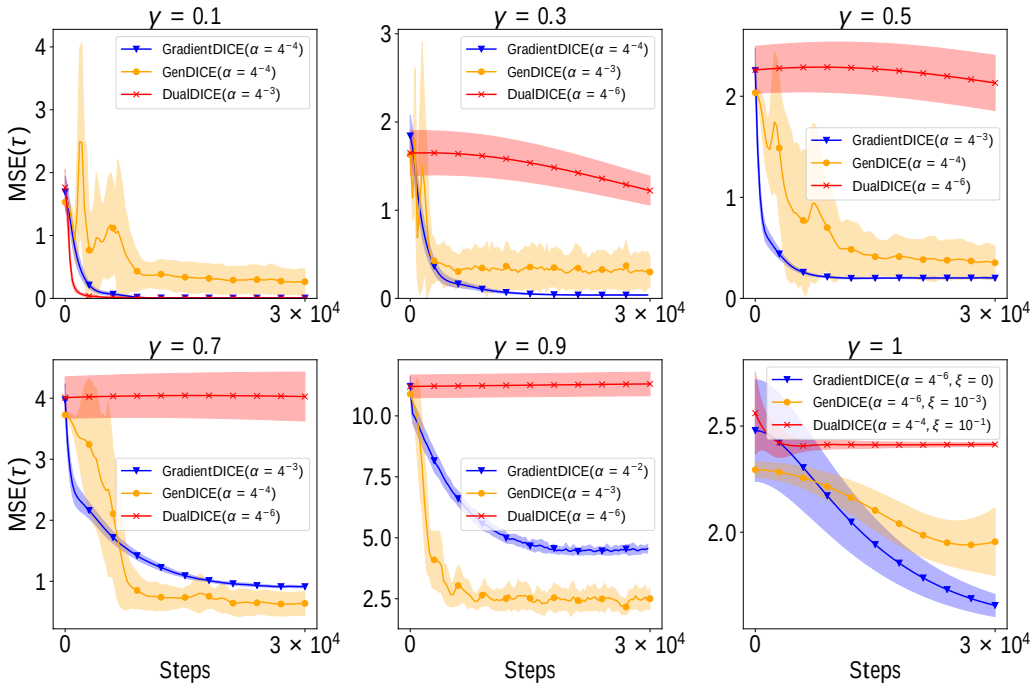

Figure 3. Density ratio learning in Boyan's Chain with a linear architecture.

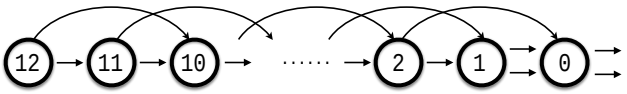

Figure 4. Two variants of Boyan's Chain. There are 13 states in total with two actions $\{a_0, a_1\}$ available at each state. The initial distribution $\tilde{\mu}_0$ is uniform over $\{s_0, \ldots, s_{12}\}$. At a state $s_i (i \geq 2)$, a_0 leads to s_{i-1} and a_1 leads to s_{i-2}. At s_1, both actions leads to s_0. At s_0, there are two variants. (1) `Episodic Boyan's Chain`: both actions at s_0 lead to s_0 itself, i.e., s_0 is an absorbing state. (2) `Continuing Boyan's Chain`: both actions at s_0 lead to a random state among $\{s_0, \ldots, s_{12}\}$ with equal probability.

a sequence of tasks by varying the discount factor in $\{0.1, 0.3, 0.5, 0.7, 0.9, 1\}$.

We train all compared algorithms for $3 \quad 10^4$ steps. We evaluate the Mean Squared Error (MSE) for the predicted every 300 steps, computed as MSE( ) $\doteq \frac{1}{26}$ $_{s,a}$((s,a) $(s,a))^2$, where the ground truth is computed analytically. We use fixed learning rates for all algorithms, which is tuned from $\{4^{-6}, 4^{-5}, \ldots, 4^{-1}\}$ to minimize the MSE( ) at the end of training. For the setting $= 1$, we additionally tune from $\{0, 10^{-3}, 10^{-2}, 10^{-1}\}$ (for a fair comparison, we also add this ridge regularization for GenDICE and DualDICE). For the penalty coefficient, we set $= 1$ as recommended by Zhang et al. (2020a). We find has little influence on the learning process in this domain.

We report the results in both tabular (Figure 2) and linear (Figure 3) settings. In the tabular setting, we use lookup tables to store ; f and . In the linear setting, we use two independent sets of weights for the two actions. As GenDICE requires 0, we use the nonlinearity $(\ )^2$ for its prediction as suggested by Zhang et al. (2020a). We do not apply any nonlinearity for GradientDICE and DualDICE. Our results show that GradientDICE reaches a lower prediction error at the end of training than GenDICE in 4 (5) out 6 tasks in the tabular (linear) setting. Moreover, the learning curves of GradientDICE are more stable than those of GenDICE in all the 6 tasks in both tabular and linear settings. Although DualDICE performs the best for the task $= 0.1$, it becomes unstable as increases, which is also observed in Zhang et al. (2020a).

5.2. Off-Policy Evaluation

We now benchmark DualDICE, GenDICE, and GradientDICE in an off-policy evaluation problem. We consider `Reacher-v2` from OpenAI Gym (Brockman et al., 2016). We consider policies in the form of $_d(s,a) + N(0, \ ^2)$, where $_d$ is a deterministic policy trained via TD3 (Fujimoto et al., 2018) for 10^6 steps and N is Gaussian noise. For the behavior policy, we set $= 0.1$ and run the policy for 10^5 steps to collect transitions, which form the dataset used across all the experiments. For the target policy, we set $= 0.05$.

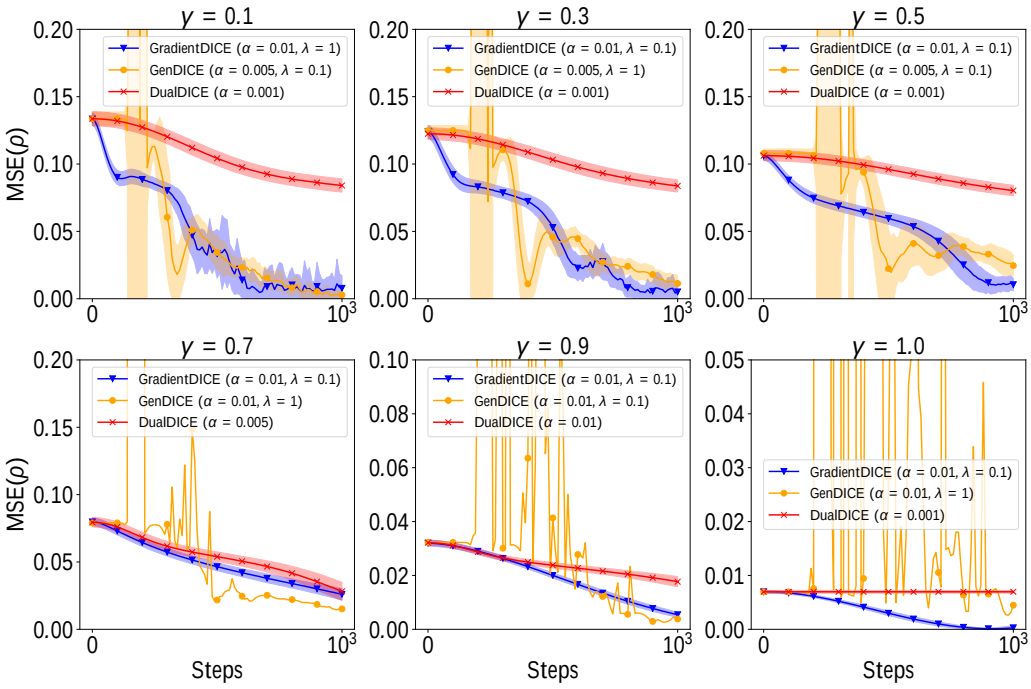

Figure 5. Off-policy evaluation in `Reacher-v2` with neural network function approximators.

We use neural networks to parameterize and f, each of which is represented by a two-hidden-layer network with 64 hidden units and ReLU (Nair & Hinton, 2010) activation function. For GenDICE, we add $(\cdot)^2$ nonlinearity for the prediction by the network. For GradientDICE and DualDICE, we do not have such nonlinearity in their prediction. Given the learned , the performance $(\cdot)$ is approximated by $\hat{(\cdot)} \doteq \frac{1}{N}\sum_{i=1}^{N}(s_i;a_i)r_i$. We train each algorithm for 10^3 steps and examine $\text{MSE}(\cdot) \doteq \frac{1}{2}(\cdot - \hat{(\cdot)})^2$ every 10 steps, where the ground truth $(\cdot)$ is computed from Monte Carlo methods via executing the target policy multiple times. We use SGD to train the neural networks with batch size 128. The learning rate and the penalty coefficient are tuned from $\{0.01; 0.005; 0.001\}$ and $\{0.1; 1\}$ with grid search to minimize MVE( ) at the end of training.

The results are reported in Figure 5. Although policy evaluation errors of GradientDICE and GenDICE tend to be similar at the end of training, the learning curves of GradientDICE are more stable than those of GenDICE, which matches the results in the tabular and linear settings. Although DualDICE tends to be more stable than both GradientDICE and GenDICE, it learns slower and does not work for the setting $= 1$, which also matches the results in Zhang et al. (2020a). To summarize, GradientDICE combines the advantages of both DualDICE (stability under the total discounted reward criterion) and GenDICE (compatibility with the average reward criterion).

6. Related Work

Inspired by Hallak & Mannor (2017); Liu et al. (2018), Gelada & Bellemare (2019); Uehara & Jiang (2019) propose different estimators for learning density ratios, either with semi-gradient updates (Sutton, 1988) or by restricting the function class to Reproducing Kernel Hilbert Space. Zhang et al. (2020b) show that some density ratios can be interpreted as special emphasis (Sutton et al., 2016). Hence the Gradient Emphasis Learning algorithm in Zhang et al. (2020b) can also be used to learn certain density ratios. All these methods, however, require a single known behavior policy. By contrast, DualDICE, GenDICE, and GradientDICE cope well with multiple unknown behavior policies.

7. Conclusion

In this paper, we point out two problems with GenDICE and fix them with GradientDICE. We provide a comprehensive theoretical analysis for GradientDICE. Our experiments confirm that the theoretical advantages of GradientDICE over GenDICE translate into an empirical performance boost in the tested domains. Overall, our work provides a new theoretical justification for the field of density-ratio-learning-based off-policy evaluation.

Acknowledgments

SZ is generously funded by the Engineering and Physical Sciences Research Council (EPSRC). This project has received funding from the European Research Council under the European Union's Horizon 2020 research and innovation programme (grant agreement number 637713). The experiments were made possible by a generous equipment grant from NVIDIA. BL's research is funded by the National Science Foundation (NSF) under grant NSF IIS1910794 and Amazon Research Award. The authors thank Lihong Li and Bo Dai for the useful discussion.

References

Baird, L. Residual algorithms: Reinforcement learning with function approximation. *Machine Learning*, 1995.

Beck, A. and Teboulle, M. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

Boyan, J. A. Least-squares temporal difference learning. In *Proceedings of the 16th International Conference on Machine Learning*, 1999.

Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

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

Dai, B., He, N., Pan, Y., Boots, B., and Song, L. Learning from conditional distributions via dual embeddings. *arXiv preprint arXiv:1607.04579*, 2016.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

Gelada, C. and Bellemare, M. G. Off-policy deep reinforcement learning by bootstrapping the covariate shift. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 2019.

Hallak, A. and Mannor, S. Consistent on-line off-policy evaluation. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Horn, R. A. and Johnson, C. R. *Matrix analysis (2nd Edition)*. Cambridge university press, 2012.

Jiang, N. and Li, L. Doubly robust off-policy value evaluation for reinforcement learning. *arXiv preprint arXiv:1511.03722*, 2015.

Lakshminarayanan, C. and Szepesvari, C. Linear stochastic approximation: How far does constant step-size and iterate averaging go? In *The 21st International Conference on Artificial Intelligence and Statistics*, 2018.

Lin, L.-J. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 1992.

Liu, B., Liu, J., Ghavamzadeh, M., Mahadevan, S., and Petrik, M. Finite-sample analysis of proximal gradient td algorithms. In *UAI*, 2015.

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

Maei, H. R. *Gradient temporal-difference learning algorithms*. PhD thesis, University of Alberta, 2011.

Mousavi, A., Li, L., Liu, Q., and Zhou, D. Black-box off-policy estimation for infinite-horizon reinforcement learning. In *International Conference on Learning Representations*, 2020.

Nachum, O., Chow, Y., Dai, B., and Li, L. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *arXiv preprint arXiv:1906.04733*, 2019.

Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.

Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 2009.

Nowozin, S., Cseke, B., and Tomioka, R. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in neural information processing systems*, pp. 271–279, 2016.

Precup, D., Sutton, R. S., and Dasgupta, S. Off-policy temporal-difference learning with function approximation. In *Proceedings of the 18th International Conference on Machine Learning*, 2001.

Puterman, M. L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Robbins, H. and Monro, S. A stochastic approximation method. *The Annals of Mathematical Statistics*, 1951.

Shapiro, A., Dentcheva, D., and Ruszczyński, A. *Lectures on stochastic programming: modeling and theory*. SIAM, 2014.

Sherman, J. and Morrison, W. J. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, 1950.

Sutton, R. S. Learning to predict by the methods of temporal differences. *Machine Learning*, 1988.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction (2nd Edition)*. MIT press, 2018.

Sutton, R. S., Maei, H. R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, C., and Wiewiora, E. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th International Conference on Machine Learning*, 2009a.

Sutton, R. S., Maei, H. R., and Szepesvári, C. A convergent $o(n)$ temporal-difference algorithm for off-policy learning with linear function approximation. In *Advances in Neural Information Processing Systems*, 2009b.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, 2011.

Sutton, R. S., Mahmood, A. R., and White, M. An emphatic approach to the problem of off-policy temporal-difference learning. *The Journal of Machine Learning Research*, 2016.

Tsitsiklis, J. N. and Van Roy, B. Analysis of temporal-diffference learning with function approximation. In *Advances in Neural Information Processing Systems*, 1997.

Uehara, M. and Jiang, N. Minimax weight and q-function learning for off-policy evaluation. *arXiv preprint arXiv:1910.12809*, 2019.

Wang, T., Bowling, M., and Schuurmans, D. Dual representations for dynamic programming and reinforcement learning. In *2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, 2007.

Wang, T., Bowling, M., Schuurmans, D., and Lizotte, D. J. Stable dual dynamic programming. In *Advances in neural information processing systems*, 2008.

Zhang, R., Dai, B., Li, L., and Schuurmans, D. Gendice: Generalized offline estimation of stationary values. In *International Conference on Learning Representations*, 2020a.

Zhang, S., Liu, B., Yao, H., and Whiteson, S. Provably convergent two-timescale off-policy actor-critic with function approximation. In *Proceedings of the 37th International Conference on Machine Learning*, 2020b.

A. Proofs

A.1. Proof of Theorem 2

Proof. We first rewrite the update for d_t as

$$d_{t+1} \doteq d_t + \ _t(G_{t+1}d_t + g_{t+1}) = d_t + \ _t(h(d_t) + M_{t+1});$$

where

$$h(d) \doteq g + Gd;\ M_{t+1} \doteq (G_{t+1} \quad G)d_t + (g_{t+1} \quad g)$$

The rest is the same as the proof of Theorem 2 in page 44 of Maei (2011) without changing notations except for that we need to verify that the real parts of all eigenvalues of G are negative. Although our G is more involved than the G used in Maei (2011), a similar routine can be carried out to verify this condition.

We first show that the real part of any nonzero eigenvalue is strictly negative. Let $\ 2\ \mathbb{C};\ \ \not= 0$ be a nonzero eigenvalue of G with normalized eigenvector x, i.e., $x\ x = 1$, where $x\ $ is the complex conjugate of x. Hence $x\ Gx = \ ;x \not= 0$. Let $x^> = (x_1^>; x_2^>; x_3)$, where $x_1\ 2\ \mathbb{C}^K; x_2\ 2\ \mathbb{C}^K; x_3\ 2\ \mathbb{C}$, it is easy to verify

$$= x\ Gx = \quad x_1Cx_1 \quad x_2x_2 \quad x_3x_3:$$

According to Assumption 1, $<(x_1Cx_1) \quad 0$, where $<(\ )$ denotes the real part and the equality holds iif $x_1 = 0$. When < 1, we have $\ = 0$. Then $\not= 0$ implies at least one of $\{x_1; x_3\}$ is nonzero. Consequently, we have $<(\ ) < 0$. When $= 1$, we have $\ > 0$. Then $\not= 0$ implies at least one of $\{x_1; x_2; x_3\}$ is nonzero. Consequently, we have $<(\ ) < 0$.

We then show 0 is not an eigenvalue of G, which completes the proof. It suffices to show $\det(G) \not= 0$. For a block matrix

$$Y = \begin{matrix} Y_1 & Y_2 \\ Y_3 & Y_4 \end{matrix};$$

we have $\det(Y) = \det(Y_4)\det(Y_1 \quad Y_2Y_4^{\ 1}Y_3) = \det(Y_1)\det(Y_4 \quad Y_3Y_1^{\ 1}Y_2)$. Applying this rule to G yields

$$\begin{aligned}
\det(G) = \quad & \det\begin{pmatrix} C & A \\ A^> & I \end{pmatrix} + \begin{matrix} {}_1 & 0 \\ 0 & \end{matrix} \quad \begin{matrix} 0 & \\ {}^2X^>d\ d^>X & \end{matrix} \\
= (\ 1)^{2K+1} & \det\begin{pmatrix} C & A \\ A^> & I + X^>d\ d^>X \end{pmatrix} \\
= (\ 1)^{2K+1} & \det(C)\det(\ I + \ X^>d\ d^>X + A^>C^{\ 1}A):
\end{aligned}$$

Note $\ X^>d\ d^>X$ is always positive semidefinite. Assumption 2 ensures at least one of $\{\ I; A^>C^{\ 1}A\}$ is strictly positive definite, which ensures $\det(G) \not= 0$. □

A.2. Proof of Proposition 2

Proof. This proof employs results from Proposition 3.2 in Nemirovski et al. (2009) by mapping Projected GradientDICE to the general mirror saddle-point stochastic approximation algorithm in Nemirovski et al. (2009). Given the similarity between Projected GradientDICE and Revised GTD (Liu et al., 2015), most of our proof is a verbatim repetition (thus omitted) of the finite sample analysis for Revised GTD in the proof of Proposition 3 in Liu et al. (2015) except for that we need to bound different terms. Namely, the *stochastic sub-gradient vector* for Projected GradientDICE is

$$G(w; y) \doteq \begin{matrix} G_w(w; y) \\ G_y(w; y) \end{matrix} \doteq \begin{matrix} G_{3;t}y + G_{4;t}w \\ (G_{1;t}y + G_{2;t}w + G_{5;t}) \end{matrix}:$$

We need only to bound $\mathbb{E}[\|G_w(w; y)\|^2]$ and $\mathbb{E}[\|G_y(w; y)\|^2]$. Assumption 4 ensures the following quantities are well-defined:

$$\begin{aligned}
D_W &\doteq (\max_{w2W} \|w\|^2 \quad \min_{w2W} \|w\|^2)^{\frac{1}{2}}; \\
D_Y &\doteq (\max_{y2Y} \|y\|^2 \quad \min_{y2Y} \|y\|^2)^{\frac{1}{2}}
\end{aligned}$$

We define shorthand $G_i \doteq \mathsf{E}[G_{i;t}]$ for $i = 1;:::;5$, all of which are constant (c.f. Eq (7)). Under Assumption 3, it is easy to verify that there are constants $_i > 0$ such that

$$\mathsf{E}[jjG_{i;t} \quad G_i]jj^2] \quad {}^2_i:$$

Using the equality

$$\mathsf{E}[jjxjj^2] = \mathsf{E}[jjx \quad \mathsf{E}[x]jj^2] + jj\mathsf{E}[x]jj^2;$$

we have

$$\begin{aligned}
\mathsf{E}[jjG_w(w;y)jj^2] \quad & \mathsf{E}[jjG_{3;t}jj^2]D_Y^2 + \mathsf{E}[jjG_{4;t}jj^2]D_W^2 \\
& {}^2_3D_Y^2 + {}^2_3jjG_3jj^2 + {}^2D_W^2 \\
& \doteq C_1 \\
\mathsf{E}[jjG_y(w;y)jj^2] \quad & {}^2_1D_Y^2jjG_1jj^2 + {}^2_2D_W^2jjG_2jj^2 + {}^2_5jjG_5jj^2 \\
& \doteq C_2
\end{aligned}$$

We now have all gradients to invoke Proposition 3.2 in Nemirovski et al. (2009). Particularly, the M^2 in Eq 3.5 in Nemirovski et al. (2009) is now

$$M^2 \doteq 2D_W^2C_1^2 + 2D_Y^2C_2^2:$$

According to Eq 3.12 in Nemirovski et al. (2009), we set the learning rates $_t$ in Projected GradientDICE as

$$_t \doteq \frac{2c}{M\sqrt{n}};$$

where $c > 0$ is a constant. Note this is a constant learning rate. Now Proposition 3.2 in Nemirovski et al. (2009) states for any > 1,

$$\Pr\left\{ {}_{\text{opt}}(w_n;y_n) > \sqrt{\frac{5}{n}}(8 + 2\ )\max\{fc;c\ {}^1gM\right\} \quad 2e \quad :$$

For a small > 0, setting $= \ln\frac{}{2}$ completes the proof. □

B. Computational Details of the Hard Example for GenDICE

In our instantiation, we have

$$\begin{aligned}
J(\ ;f;\ ) &= \mathsf{E}_p[\ (s;a)f(s^0;a^0)] \quad \mathsf{E}_d\ [\ (s;a)(f(s;a) + \frac{1}{4}f(s;a)^2)] + \mathsf{E}_d\ [\quad (s;a) \quad]\ \ \frac{^2}{2} \\
&= \frac{1}{4}\ {}^2_1f_1 + \frac{1}{4}\ {}^2_1f_2 + \frac{1}{4}\ {}^2_2f_1 + \frac{1}{4}\ {}^2_2f_2 \quad \frac{1}{2}\ {}^2_1(f_1 + \frac{1}{4}f_1^2) \quad \frac{1}{2}\ {}^2_2(f_2 + \frac{1}{4}f_2^2) + \frac{1}{2}\ {}^2_1 + \frac{1}{2}\ {}^2_2 \qquad \frac{1}{2}\ {}^2; \\
\frac{@J}{@\ _1} &= \frac{1}{2}\ {}_1f_1 + \frac{1}{2}\ {}_1f_2 \quad {}_1(f_1 + \frac{1}{4}f_1^2) + \quad {}_1; \\
\frac{@J}{@\ _2} &= \frac{1}{2}\ {}_2f_1 + \frac{1}{2}\ {}_2f_2 \quad {}_2(f_2 + \frac{1}{4}f_2^2) + \quad {}_2; \\
\frac{@J}{@f_1} &= \frac{1}{4}\ {}^2_1 + \frac{1}{4}\ {}^2_2 \quad \frac{1}{2}\ {}^2_1(1 + \frac{1}{2}f_1); \\
\frac{@J}{@f_1} &= \frac{1}{4}\ {}^2_1 + \frac{1}{4}\ {}^2_2 \quad \frac{1}{2}\ {}^2_1(1 + \frac{1}{2}f_2); \\
\frac{@J}{@} &= \frac{1}{2}\ {}^2_1 + \frac{1}{2}\ {}^2_2 \quad 1 \quad :
\end{aligned}$$

It is clear that the gradient vanishes at $(\ _1;\ _2;f_1;f_2;\ ) = (0;0;0;0;\quad 1)$.

C. Details of Experiments

We implemented `Boyan's Chain` (Boyan, 1999) by ourselves. `Reacher-v2` is from Open AI gym (Brockman et al., 2016) [3], which is used as a benchmark in both Nachum et al. (2019) and Zhang et al. (2020a). For `Reacher-v2` experiments with neural networks, we use a batch size 128 for training. For `Boyan's Chain` experiments, the batch size is 1. For DualDICE, the convex function to compose the loss is $f(x) = \frac{2}{3}|x|^{\frac{3}{2}}$ as recommended by Nachum et al. (2019). We conducted our experiments in a server with an Intel ® Xeon ® Gold 6152 CPU. Our implementation is based on PyTorch.

C.1. State Features of Boyan's Chain

We use exactly the same features as Boyan (1999).

$$
\begin{aligned}
x(s_{12}) &\doteq [1; 0; 0; 0]^\top \\
x(s_{11}) &\doteq [0.75; 0.25; 0; 0]^\top \\
x(s_{10}) &\doteq [0.5; 0.5; 0; 0]^\top \\
x(s_9) &\doteq [0.25; 0.75; 0; 0]^\top \\
x(s_8) &\doteq [0; 1; 0; 0]^\top \\
x(s_7) &\doteq [0; 0.75; 0.25; 0]^\top \\
x(s_6) &\doteq [0; 0.5; 0.5; 0]^\top \\
x(s_5) &\doteq [0; 0.25; 0.75; 0]^\top \\
x(s_4) &\doteq [0; 0; 1; 0]^\top \\
x(s_3) &\doteq [0; 0; 0.75; 0.25]^\top \\
x(s_2) &\doteq [0; 0; 0.5; 0.5]^\top \\
x(s_1) &\doteq [0; 0; 0.25; 0.75]^\top \\
x(s_0) &\doteq [0; 0; 0; 1]^\top
\end{aligned}
$$

[3] `https://gym.openai.com/`