Journal of Machine Learning Research 23 (2022) 1-40 Submitted 6/21; Revised 11/22; Published 11/22

On Instrumental Variable Regression for Deep Offline Policy Evaluation

Yutian Chen YUTIANC@GOOGLE.COM
DeepMind
R7, 14-18 Handyside Street
King's Cross London
N1C 4DN

Liyuan Xu LIYUAN.JO.19@UCL.AC.UK
Gatsby Unit

Caglar Gulcehre CAGLARG@GOOGLE.COM
DeepMind

Tom Le Paine TPAINE@GOOGLE.COM
DeepMind

Arthur Gretton ARTHUR.GRETTON@GMAIL.COM
Gatsby Unit

Nando de Freitas NANDODEFREITAS@GOOGLE.COM
DeepMind

Arnaud Doucet ARNAUDDOUCET@GOOGLE.COM
DeepMind

Editor: Francis Bach

Abstract

We show that the popular reinforcement learning (RL) strategy of estimating the state-action value (Q-function) by minimizing the mean squared Bellman error leads to a regression problem with confounding, the inputs and output noise being correlated. Hence, direct minimization of the Bellman error can result in significantly biased Q-function estimates. We explain why fixing the target Q-network in Deep Q-Networks and Fitted Q Evaluation provides a way of overcoming this confounding, thus shedding new light on this popular but not well understood trick in the deep RL literature. An alternative approach to address confounding is to leverage techniques developed in the causality literature, notably instrumental variables (IV). We bring together here the literature on IV and RL by investigating whether IV approaches can lead to improved Q-function estimates. This paper analyzes and compares a wide range of recent IV methods in the context of offline policy evaluation (OPE), where the goal is to estimate the value of a policy using logged data only. By applying different IV techniques to OPE, we are not only able to recover previously proposed OPE methods such as model-based techniques but also to obtain competitive new techniques. We find empirically that state-of-the-art OPE methods are closely matched in performance by some IV methods such as AGMM, which were not developed for OPE[1].

Keywords: Instrumental variable regression; Generalized method of moments; Reinforcement learning; Two Stage Least Squares; Offline policy evaluation

1. We open-source all our code and datasets at `https://github.com/liyuan9988/IVOPEwithACME`

1. Introduction

Deep neural networks have made it possible for reinforcement learning (RL) to attain superhuman performance in challenging domains such as ATARI from raw sensory data (Mnih et al., 2015) and Go (Silver et al., 2016). While RL is starting to be used for real-world applications (Bellemare et al., 2020), its adoption remains fairly limited. Standard RL techniques require repeated interaction with the environment. This is often too costly to implement practically, and running a poor policy could lead to disastrous outcomes (e.g., in power plants or healthcare decision-making systems). While controlling a large and/or complex real-world system can be costly and risky, data acquisition is often comparatively cheap. The goal of offline RL is to evaluate and learn new policies based only on logged data, without any interaction with the environment.

In this paper, we focus on the problem of Offline Policy Evaluation (OPE), also known in the literature as Off-policy Policy Evaluation. OPE involves estimating the value of a new policy using logged data produced by possibly many different policies. This is a problem of great significance because individuals and organizations often need to choose a single policy among a wide set of proposed policies for deployment. Choosing which policy to deploy well can result in improved user satisfaction, or better medical treatments.

A plethora of methods have been proposed to address this problem; see e.g. Precup (2000); Precup et al. (2001); Dudík et al. (2011); Thomas and Brunskill (2016); Jiang and Li (2016); Liu et al. (2018); Farajtabar et al. (2018); see also Levine et al. (2020); Fu et al. (2021) for recent reviews. We focus here on methods that are relying on an estimate of the state-action value function, known as the Q-function. It is well-known that the Q-function can be estimated by minimizing the mean squared Bellman error. However, the resulting regression problem is not standard as the inputs and the output noise are correlated, leading to some confounding. We show here that fixing the target Q-network in the popular Deep Q-Networks (DQN) (Mnih et al., 2013) and Fitted Q Evaluation (FQE) (Le et al., 2019) can be re-interpreted as a strategy addressing this confounding. We then investigate a different class of approaches to address the same problem. In causal inference, Instrumental Variables (IV) regression is a standard strategy for learning causal relationships between confounded treatment and outcome variables from observational data by utilizing an instrumental variable, which affects the outcome only through the treatment (Stock and Trebbi, 2003). The connection between RL and IV ideas was made early on by Bradtke and Barto (1996) when introducing Least Square Temporal Differences (LSTD), a method to estimate on-policy linearly parameterized value functions. Their derivation made use of the *two-stage least squares* (2SLS) algorithm, the most standard IV regression technique. However, the connection between RL and IV seemed to have been largely ignored ever since in the literature.

Here we build on this connection. We exploit the fact, shown by Xu et al. (2021), that we can estimate a Q function parameterized by a neural networks using non-linear IV regression techniques. We can thus use the non-linear IV techniques recently developed in machine learning (Hartford et al., 2017a; Lewis and Syrgkanis, 2018; Singh et al., 2019a; Muandet et al., 2019; Bennett et al., 2019a; Dikkala et al., 2020; Luofeng et al., 2020; Xu et al., 2021) to perform Q function estimation.

Our contributions in this paper are four-fold.

- We show that estimating the state-action value (Q) by minimizing the mean squared Bellman error leads to a regression problem with confounding, the inputs and output noise being correlated. We provide a re-intepretation of the popular strategy consisting of fixing the target Q-network in Deep Q-Networks (DQN) (Mnih et al., 2013) and Fitted Q Evaluation (FQE) (Le et al., 2019) as a way to overcome confounding.
- We extend the IV interpretation of the on-policy state value (V) linear estimation problem to off-policy state-action value (Q) linear estimation. As shown recently by Xu et al. (2021), we can further recast the problem of non-linear Q-function evaluation and OPE as a non-linear IV regression problem, bringing together the literature on IV and RL.
- We review recent IV methods developed in machine learning, including Deep IV (Hartford et al., 2017a), Kernel IV (Singh et al., 2019b), Deep Generalized Method of Moments (Deep GMM) (Bennett et al., 2019b), adversarial GMM (AGMM) (Dikkala et al., 2020) and Deep Feature IV (DFIV) (Xu et al., 2021) and specialize them to the OPE problem. By doing so, not only do we recover some OPE techniques already available, but also obtain novel methods and insights.
- We evaluate the performance of these techniques empirically on a variety of tasks and environments, including Behaviour Suite (BSuite) (Osband et al., 2019) and DeepMind Control Suite (DM Control) (Tassa et al., 2020). We found experimentally that some of the recent IV techniques such as AGMM display performance on par with state-of-the-art FQE methods. We open-source the implementation of all methods and datasets at `https://github.com/liyuan9988/IVOPEwithACME`.

Our main findings are that when doing OPE for a policy near to that which generated the available data, the confounding effect can be very pronounced, and ignoring it — as in Deterministic Bellman Residual Minimization (DBRM) (Saleh and Jiang, 2019) — is problematic. In this scenario, we find that the best IV method - AGMM - performs on par with FQE, and is only outperformed by distributional FQE. On more difficult scenarios where the evaluation policy is far from the behavioral policy, additional effects due to a combination of distribution shift and model mismatch come into play. In this context, while AGMM performs on par with FQE and DFQE, DBRM is also competitive, while being more stable than competing methods.

Note that there have been recent papers combining IV techniques to RL; see e.g. Bennett et al. (2021); Li et al. (2021); Liao et al. (2021). These papers use IV methods for non-standard RL models with unobserved confounders, whereas we focus here on the standard RL model.

The rest of this paper is organized as follows. In Section 2, we define the RL model of interest, and provide overviews of the offline policy evaluation problem and of instrumental variable regression. In Section 3, we review the LSTD method of Bradtke and Barto (1996), introduced to estimate linearly parameterized value functions and show how it is related to 2SLS, the most popular IV method. In Section 4, we recast the problem of non-linear Q function estimation as a non-linear IV problem, and then review some of the promising recent techniques that have been developed in this context. In Section 5, we propose two sets of benchmarking problems with stochastic environments to assess the performance of those methods in their application to OPE, compared with a state-of-the-art OPE baseline. Section 6 concludes with a discussion.

2. Background

2.1 Reinforcement learning and offline policy evaluation

Reinforcement learning considers a Markov decision process $\langle \mathcal{S}, \mathcal{A}, P, R, \mu_0, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, and $P(s'|a,s)$ is the probability or probability density of making a transition to state s' when taking action a in state s. $R(r|s,a)$ denotes the probability density of observing reward r after having taken action a from s, and $\mu_0(s)$ is the initial state distribution. Let π be a policy of an agent, and denote $\pi(a|s)$ as the probability or probability density of selecting action a in state $s \in \mathcal{S}$. With a discount factor $\gamma \in (0,1]$, the state-action value function - i.e. Q function - is defined by

$$Q(s,a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a\right], \tag{1}$$

with $a_t \sim \pi(\cdot \mid s_t), s_{t+1} \sim P(\cdot|s_t,a_t), r_t \sim R(\cdot|s_t,a_t)$ for $t \geq 0$.

Common tasks in reinforcement learning including estimating the *value* of a given target policy π or optimizing the policy value with respect to π, where the policy value is defined by the expected sum of discounted rewards from the initial state distribution

$$\rho(\pi) = \mathbb{E}_{s_0 \sim \mu_0}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] = \mathbb{E}_{s \sim \mu_0, a|s \sim \pi}[Q(s,a)], \tag{2}$$

When an agent is prohibited to interact with the environment directly, one has to rely on an existing dataset of trajectories or transition tuples (s,a,r,s'), to estimate the policy value or learn the optimal policy. The dataset could have been collected by one or a mixture of potentially unknown policies of potentially unknown analytical form, denoted by $\pi_b(\cdot|s)$, and the corresponding state action distribution is denoted by μ_b.

The goal of offline policy evaluation (OPE) is to evaluate the value of a target policy, π, based on the offline behavior dataset. This problem has been extensively studied in the literature. The readers are referred to Levine et al. (2020) for a review and Voloshin et al. (2019b); Fu et al. (2021) for benchmarks of recent OPE algorithms.

One family of OPE approaches is to estimate the value function based on the Bellman equation,

$$Q(s,a) = \mathbb{E}_{r \sim R(\cdot|s,a)}\left[r|s,a\right] + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a), a' \sim \pi(\cdot|s')}\left[Q(s',a')|s,a\right], \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{3}$$

We can solve the Bellman equation as a least square regression problem for the reward r on the state-action pair (s,a)

$$\mathbb{E}\left[r|s,a\right] = Q(s,a) - \gamma \mathbb{E}\left[Q(s',a')|s,a\right], \tag{4}$$

and find the function Q to minimize the mean squared Bellman error (MSBE) (Sutton and Barto, 2018, p. 268) with respect to the behavior distribution

$$Q = \arg\min_Q \mathbb{E}_{(s,a)\sim\mu_b, r\sim R}\left[\left(r - Q(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a), a' \sim \pi(\cdot|s')}\left[Q(s',a')\right]\right)^2\right]. \tag{5}$$

We then use Equation (2) for the evaluation policy π.

2.1.1 A Simple Biased Estimator

A simple algorithm to approximate the objective of Equation (5) using transition samples (s, a, r, s', a') from the dataset is known as Deterministic Bellman Residual Minimization (DBRM) (Saleh and Jiang, 2019). We consider here a simple variant of DBRM as a baseline with two independent action samples from the given target policy,

$$Q_{\text{DBRM}} = \arg\min_{Q} \mathbb{E}\left[\left(r - Q(s,a) + \gamma Q(s', a'^{(1)})\right)\left(r - Q(s,a) + \gamma Q(s', a'^{(2)})\right)\right], \tag{6}$$

where $(s, a) \sim \mu_b, r \sim R, s' \sim P(\cdot|s, a), a'^{(1)} \sim \pi(\cdot|s'), a'^{(2)} \sim \pi(\cdot|s')$. The argument within the expectation of Equation (6) is an unbiased estimate of MSBE only if the MDP's transition dynamics P and the target policy π are deterministic. If this is not the case, we would require two independent samples of s' starting from the same (s, a) to obtain an unbiased estimate of the MSBE objective in Equation (5) (Baird, 1995). This is usually not possible. More practical and sophisticated methods have been proposed to mitigate the bias (Antos et al., 2008; Munos and Szepesvári, 2008).

2.1.2 Fitted Q Evaluation

Alternatively, one can move the troublesome expectation in Equation (4) from inside the regression function to the target as follows,

$$\mathbb{E}\left[r|s,a\right] = Q(s,a) - \gamma \mathbb{E}\left[Q(s',a')|s,a\right] \Longrightarrow \mathbb{E}\left[r + \gamma Q(s',a')|s,a\right] = Q(s,a), \tag{7}$$

and minimize the least squared temporal difference (TD) error iteratively, with the Q function on the left hand side being fixed at every iteration,

$$Q_k = \arg\min_{Q} \mathbb{E}_{(s,a)\sim\mu_b, s'\sim P, a'\sim\pi, r\sim R}\left[\left(Q(s,a) - \left(r + \gamma Q_{k-1}(s',a')\right)\right)^2\right]. \tag{8}$$

This method is known as fitted Q evaluation (FQE) (Le et al., 2019), a variant of the fitted Q iteration (Ernst et al., 2005) algorithm. Fu et al. (2021) show that FQE outperforms other OPE algorithms in a deep OPE benchmark. The same idea was used in other approximate dynamic programming approaches such as the Deep Q Network (DQN) (Mnih et al., 2015) where the parameters of the target Q network (corresponding to Q_{k-1} in Equation (8)) were fixed when updating the online Q network.

2.2 Instrumental variable regression

Instrumental variable (IV) regression methods (Stock and Trebbi, 2003) are standard techniques developed in the causal inference and econometrics literature, which are used to predict the effect of actions X (called *treatment*) on the world when the treatment affects the distribution of the variable of interest Y, which is called the *outcome*. IV regression provides a framework to assess this effect (called the *structural function*) by using an instrumental variable Z, which only affects the treatment directly, but has no direct effect on the outcome.

Instrumental variables can be found in many contexts, and IV regression is extensively used by economists and epidemiologists. For example, (Wright, 1928; Blundell et al., 2012) used supply cost shifters as instrumental variables, and estimate the effect of price on the demand to correct for confounders such as the time of the year. IV regression has also been used to measure the effect of a drug in the scenario of imperfect compliance (Angrist et al., 1996), or the influence of military service on lifetime earnings (Angrist, 1990).

Formally, we aim to learn a relationship between X and Y, which is generated from

$$Y = f_{\text{struct}}(X) + \varepsilon, \quad \mathbb{E}[\varepsilon] = 0, \quad \mathbb{E}[\varepsilon|X] \neq 0, \tag{9}$$

where f_{struct} is the structural function, which we assume to be continuous, and ε is an additive noise term. The challenge is that $\mathbb{E}[\varepsilon|X] \neq 0$, which reflects the existence of a latent confounder. Hence, we cannot use ordinary supervised learning techniques since $f_{\text{struct}}(x) \neq \mathbb{E}[Y|X=x]$.

To deal with the confounder ε, we assume to have access to an instrumental variable $Z \in \mathcal{Z}$ which satisfies the following assumption.

Assumption 1 *The conditional distribution $P(X|Z)$ is not constant in Z and one has $\mathbb{E}[\varepsilon|Z] = 0$.*

Intuitively, Assumption 1 means that the instrument Z induces variation in the treatment X but is uncorrelated with the hidden confounder ε. The causal graph describing these relationships is shown in Figure 1.[2] Note that the instrument Z cannot have an incoming edge from the latent confounder that is also a parent of the outcome. It follows directly from Assumption 1 that

$$\mathbb{E}[Y|Z] = \mathbb{E}[f(X)|Z] + \mathbb{E}[\varepsilon|Z] = \int_X f(X)P(X|Z)\mathrm{d}X. \tag{10}$$

Classically, IV regression is solved by the *two-stage least squares* (2SLS) algorithm; we learn a mapping from the instrument to the treatment in the first stage, and learn the structural function in the second stage as the mapping from the conditional expectation of the treatment given the instrument (obtained from stage 1) to the outcome. Originally, 2SLS assumes linear relationships in both stages, i.e.,

$$X = Z\omega + \delta, \quad f_{\text{struct}}(X) = X\theta\,,$$

where ω and θ are unknown regression coefficients and δ is a random variable satisfying $\mathbb{E}[\delta|Z] = 0$. In the case when the dimension of X equals that of Z, the 2SLS estimator has the following simple form

$$\hat{\theta} = \left(Z^\top X\right)^{-1}\left(Z^\top Y\right)\,, \tag{11}$$

where we somewhat abuse notation and denote by X, Z the matrices of observed treatment and instrumental variables, by Y the vector of outcomes, and each row corresponds to one observation.

2. We show the simplest causal graph in Figure 1 It entails $Z \perp\!\!\!\perp \varepsilon$, but we only require Z and ε to be uncorrelated in Assumption 1. Of course, this graph also says that Z is not independent of ε when conditioned on observations X.

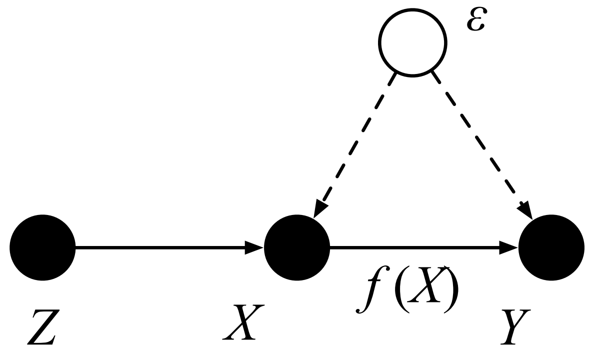


Figure 1: Causal graphical model for instrumental variable methods.

Instrumental variable methods have been extended to non-linear settings first in economics (Newey and Powell, 2003; Carrasco et al., 2007; Darolles et al., 2011; Blundell et al., 2012; Chen and Christensen, 2018; Voloshin et al., 2019a) and more recently in machine learning (Hartford et al., 2017a; Lewis and Syrgkanis, 2018; Singh et al., 2019a; Muandet et al., 2019; Bennett et al., 2019a; Dikkala et al., 2020; Luofeng et al., 2020; Xu et al., 2021). One approach has been to use non-linear feature maps. Sieve IV (Newey and Powell, 2003; Chen and Christensen, 2018) uses a finite number of explicitly specified basis functions. Kernel IV (KIV) (Singh et al., 2019a) and Dual IV regression (Muandet et al., 2019) extend sieve IV to allow for an infinite number of basis functions using reproducing kernel Hibert spaces (RKHS). Although these methods enjoy desirable theoretical properties, the flexibility of the model is limited due to the prespecified features. To mitigate this limitation, Xu et al. (2021) proposes Deep Feature IV (DFIV) method, in which one learns features adaptively using neural networks while preserving the two-stage nature of 2SLS.

Another non-linear approach to the stage 1 regression is to estimate the conditional distribution $P(X|Z)$ (Carrasco et al., 2007; Darolles et al., 2011; Hartford et al., 2017a). This allows flexible models, including deep neural nets, as proposed in the DeepIV algorithm of (Hartford et al., 2017a). However, the conditional density estimation can be costly, and can suffer from high variance when the treatment is high-dimensional.

As a further alternative, several recent works (Lewis and Syrgkanis, 2018; Bennett et al., 2019a; Dikkala et al., 2020; Luofeng et al., 2020) have been inspired by another instrumental variable technique, the Generalized Method of Moments (GMM) (Hansen, 1982), and find non-linear structural functions to ensure that the regression residual and the instrument are uncorrelated. These works do not require two stage regression, and the resulting methods are often formulated as solving a minimax optimization problem.

3. Relationship between LSTD and Linear IV

Solving Equation (5) requires computing the conditional expectation $\mathbb{E}\left[Q(s', a')|s, a\right]$. This is usually infeasible in practice because it would require being able to reset the environment to state s and draw multiple samples of the next state s'. Bradtke and Barto (1996) proposed the Least Square Temporal Difference (LSTD) algorithm to solve this problem with a single sample of s'. Specifically, they consider estimating the *state value function* $V(s) = \mathbb{E}_{a\sim\pi}[Q(s,a)]$ in the *on-policy* RL setting, i.e., $\pi_b = \pi$, and assume the value function can be parameterized as a linear function of a fixed set of features. Bradtke and Barto (1996) pointed out originally that Equation (5) could be reformulated and solved with a linear IV method.

Lagoudakis and Parr (2003) proposed to model the *state-action value function* Q instead of V using a similar linear combination of features, and extended LSTD to LSTD-Q so that it could be applied to off-policy RL, $\pi_b \neq \pi$. Their derivation does not rely on IV ideas. We propose here an alternative derivation of LSTD-Q which is a natural extension of the IV approach pioneered in Bradtke and Barto (1996) to the Q function.

We consider a linear approximation of the form

$$Q(s,a) = \phi(s,a)^\top \theta, \tag{12}$$

where $\phi(s,a)$ is a set of features evaluated at (s,a) and θ is the parameter vector to estimate. *If Equation* (12) *were exact and not an approximation*, then we could rewrite Equation (4) as

$$\underbrace{r}_{Y} = Q(s,a) - \gamma Q(s',a') + \left(\gamma Q(s',a') - \gamma \mathbb{E}\left[Q(s',a')|s,a\right]\right) + \left(r - \mathbb{E}\left[r|s,a\right]\right) \tag{13}$$

$$= \underbrace{\left(\phi(s,a) - \gamma\phi(s',a')\right)^\top}_{X} \theta + \underbrace{\gamma\left(\phi(s',a') - \mathbb{E}\left[\phi(s',a')|s,a\right]\right)^\top \theta + r - \mathbb{E}\left[r|s,a\right]}_{\varepsilon}, \tag{14}$$

where $(s,a) \sim \mu_b,\ \ s' \sim P(s'|s,a),\ \ a' \sim \pi(a'|s')$. The decomposition in Equation (13) was discussed in Xu et al. (2021). Equation (14) matches the regression formulation in Equation (9) (see Figure 2 for the causal graphical model).

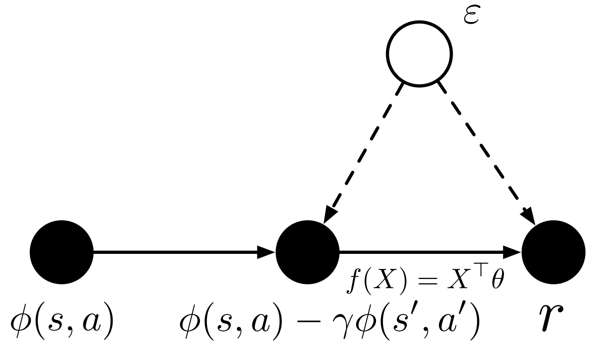


Figure 2: Causal graphical model of LSTD

Because the observation noise ε and input X are correlated through the sample of s', a', we have a confounded regression problem, that is

$$\begin{aligned} &\mathbb{E}\left[\varepsilon\right] = 0, \text{ but} \\ &\mathbb{E}\left[\varepsilon|X\right] \neq 0, \text{ because } \mathbb{E}\left[\phi(s',a') - \mathbb{E}\left[\phi(s',a')|s,a\right]|s,a,s',a'\right] \neq 0\,. \end{aligned} \tag{15}$$

Solving the least squared minimization problem of Equation (14) directly will lead to an inconsistent estimate of θ. By choosing the feature of (s,a) as the instrumental variable,

$Z = \phi(s, a)$, we can show that Z is uncorrelated with both terms in ε

$$\operatorname{corr}(\phi(s,a), \phi(s',a') - \mathbb{E}\left[\phi(s',a')|s,a\right]) = 0, \quad \operatorname{corr}(\phi(s,a), r - \mathbb{E}\left[r|s,a\right]) = 0\,, \tag{16}$$

which follows directly from Lemma 4 in Bradtke and Barto (1996) by swapping s for (s, a). Therefore, we have $\mathbb{E}\left[\varepsilon|Z\right] = 0$. In this scenario Assumption 1 for applying IV regression is satisfied. We can thus derive the same LSTD-Q estimator using 2SLS as

$$\hat{\theta} = \left(\Phi^\top \left(\Phi - \gamma\Phi'\right)\right)^{-1} \Phi^\top R\,, \tag{17}$$

where Φ, Φ', and R are matrices where every row corresponds respectively to the transpose of $\phi(s, a), \phi(s', a')$ and r from the offline dataset, except for $a' \sim \pi(a'|s')$.

We note that the derivation by Bradtke and Barto (1996) requires that the value function lives in the linear subspace of the features, Equation (12). Then they show that as the number of data increases, the solution converges under regularity conditions to the least-square fixed-point approximation (without mentioning it). The instrumental variable interpretation also requires the structural function in Equation (14) to hold, which is derived from Equation (12). Convergence of the LSTD algorithm does not require Equation (12) to be valid. Lagoudakis and Parr (2003) show that there is indeed no need to make such an assumption, and obtain a direct derivation by least-square fixed-point approximation. We refer the readers to Lagoudakis and Parr (2003) for this alternative interpretation.

Remark 1 *In the formulation of FQE in Section 2.1.2, because $Q(s', a')$ is part of the output, the resulting regression problem becomes*

$$\underbrace{r + \gamma\phi(s',a')^\top\,\theta}_{Y} = \underbrace{\phi(s,a)}_{X}{}^\top\,\theta + \underbrace{\gamma\left(\phi(s',a') - \mathbb{E}\left[\phi(s',a')|s,a\right]\right)^\top\,\theta + r - \mathbb{E}\left[r|s,a\right]}_{\varepsilon}\,. \tag{18}$$

One can see that this regression problem is not confounded as ε is uncorrelated with X (see Equation (16)*), but θ in the target Y has to be fixed when estimating the parameter.*

4. Policy Evaluation with Non-linear Functions and Non-linear IV

In this section, we will first extend the LSTD algorithm to the scenario with a non-linear value function and formulate it as a non-linear IV problem. We then introduce a few recent representative non-linear IV methods as OPE algorithms under that setting, using our notations for consistency whenever possible.

4.1 Extension to Non-linear Value Functions

When the value function Q is a non-linear function of (s, a), it was shown recently by Xu et al. (2021) that we can estimate it by solving a non-linear IV regression problem following Equation (13) with the corresponding causal graphical model in Figure 3,

$$\underbrace{r}_{Y} = \underbrace{Q(s,a) - \gamma Q(s',a')}_{f(X)} + \underbrace{\left(\gamma Q(s',a') - \gamma\mathbb{E}\left[Q(s',a')|s,a\right]\right) + (r - \mathbb{E}\left[r|s,a\right])}_{\varepsilon}, \tag{19}$$

where $(s,a)\sim\mu_b,\ \ s'\sim P(s'|s,a),\ \ a'\sim\pi(a'|s')$ and the structural function is $f(s,a,s',a')=Q(s,a)-\gamma Q(s',a')$ with $X=(s,a,s',a')$. Choosing the instrument $Z=(s,a)$, it is easy to show that Z satisfies Assumption 1 as $P(X|Z)$ is not constant in Z, where Z is a subset of X and $\mathbb{E}(\varepsilon|Z)=0$.

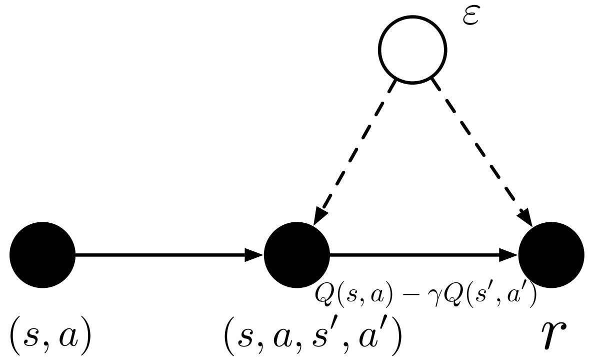


Figure 3: Causal graphical model for policy evaluation with non-linear Q function.

In practice, we need to parameterize the structural function $f(s,a,s',a')$. It is sensible to use a parameterization of the form

$$f_\theta(s,a,s',a')=Q_\theta(s,a)-\gamma Q_\theta(s',a') \tag{20}$$

instead of parameterizing f directly. We should also keep in mind that when we use approximate Q functions it will not be true that $\mathbb{E}(\varepsilon|Z)=0$ exactly. The induced bias will be illustrated later in the experiments with an under-fitted function approximator.

The dependency of the output noise ε on the next state s' bears resemblance to the "colored" noise in the GPTD formulation (Engel et al., 2005). However, we emphasize here that the GPTD formulation fails to reflect the confounding issue inherent to the Bellman residual minimization problem because the noise $N(s,s')$ in GPTD is still assumed to be uncorrelated with s' even though dependent. Therefore, their solution may still lead to a biased estimator.

Depending on the function family of Q_θ and how to apply the instrumental variables, we will present a few representative non-linear IV methods in the setting of offline policy evaluation in the following sections.

4.2 Deep IV

The Deep IV method in Hartford et al. (2017b) is based on the identity Equation (10)

$$\mathbb{E}\left[Y|Z\right]=\int_X f(X)P(X|Z)\mathrm{d}X. \tag{21}$$

It is a two-stage regression approach that estimates the nonlinear relationships between Z and X, and X and Y, using a neural network function approximator in each stage. In the first stage, Deep IV trains a treatment network to estimate the conditional distribution of treatment $P_\phi(X|Z)$ by maximum likelihood estimation. The network outputs a categorical distribution if the treatment variable X is discrete and a mixture of Gaussian distributions if it is continuous. In the second stage, it estimates the structural function using an outcome network $f_\theta(X)$ by regressing Y on the conditional expectation $\mathbb{E}_{P_\phi(X|Z)}\left[f_\theta(X)|Z\right]$ where the estimate of the expectation is obtained by Monte Carlo samples from $P_\phi(X|Z)$.

In our RL context, recall that $X = (s, a, s', a')$ while $Z = (s, a)$, so the non-degenerate part of the conditional distribution of the treatment is given by

$$P(s', a'|s, a) = P(s'|s, a)\pi(a'|s'). \tag{22}$$

Thus, applying Deep IV in the RL context consists first of estimating the transition distribution $P(s'|s, a)$ using a generative model (as we know the target policy $\pi(a'|s')$). We then compute a Monte Carlo estimate of $\mathbb{E}(f_\theta(X)|Z)$ with multiple samples, and estimate Q by minimizing the approximate MSBE in Equation (5).

Hartford et al. (2017b) also allow for observable confounders, but this extension is unnecessary in our scenario. Deep IV is closely related to the model-based reinforcement learning algorithm Dyna-Q (Sutton, 1990), which also learns the transition distribution and then Q by minimizing the TD error. The difference is that Dyna-Q uses Q-learning to minimize the TD error in the Bellman optimal equation in order to improve the policy instead of estimation, and it requires only a single sample of s' to provide an unbiased estimate of the gradient, similar to the iterative FQE algorithm.

4.3 KIV and DFIV

KIV (Singh et al., 2019a) and DFIV (Xu et al., 2021) introduce nonlinear feature maps in the IV formulation. Similar to Deep IV, this approach is also based on Equation (10), and solves for f by minimizing

$$\mathcal{L}(f) = \mathbb{E}_{YZ}\left[(Y - \mathbb{E}_{X|Z}\left[f(X)\right])^2\right] + R(f),$$

where $R(f)$ is the regularization for f. KIV and DFIV regress to the expected features of X on Z, however, in contrast to Deep IV, which estimates the conditional distribution of $P(Z|X)$; density estimation is not necessary for estimating $\mathbb{E}_{X|Z}\left[f(X)\right]$, and can be more difficult in practice. KIV and DFIV employ the following models:

$$f(X) = w^\top \phi(X), \quad \mathbb{E}\left[\phi(X)|Z\right] = V\psi(Z),$$

where w, V are the learnable linear weights and $\phi(X), \psi(Z)$ are the nonlinear feature maps for the treatment and the instrument, respectively.

KIV considers static feature maps from a Reproducing Kernel Hilbert Space (RKHS) for $\phi(X), \psi(Z)$ and learns weights w, V by a two-stage regression: stage 1 performs the regression from the instrument Z to the treatment features $\phi(X)$ to learn weight V; then in stage 2, weights w are learned by minimizing the loss $\mathcal{L}(f)$ using the predicted treatment features $V\psi(Z)$. DFIV additionally learns feature maps $\phi(X), \psi(Z)$ using neural networks in the same two-stage regression.

Note that in the RL context where $X = (s, a, s', a')$ and $Z = (s, a)$, the loss $\mathcal{L}$ is identical to MSBE defined in eq. (5) apart from the regularization term. This two-stage regression proceeds as follows. As in eq. (12), KIV and DFIV model $Q(s, a) = \phi(s, a)^\top\theta$ where ϕ is a feature map and θ are the parameters. Furthermore, they model the conditional expectation as $\mathbb{E}_{s',a'|s,a}\left[\phi(s',a')\right] = V\psi(s,a)$, where $\psi(s,a)$ is another feature map and V is the parameter matrix to be learned.

In stage 1, V is learned by minimizing the following loss,

$$\hat{V} = \arg\min_{V} \mathcal{L}_1(V), \quad \mathcal{L}_1(V) = \mathbb{E}_{s,a,s',a'}\left[\|\phi(s',a') - V\psi(s,a)\|^2\right] + \lambda_1\|V\|^2, \tag{23}$$

where $\lambda_1 > 0$ is a regularization parameter. This is a linear ridge regression problem with multiple targets, which can be solved analytically. In stage 2, given $\hat{V}$, θ is obtained by minimizing the loss

$$\hat{\theta} = \arg\min_{\theta} \mathcal{L}_2(\theta), \quad \mathcal{L}_2(\theta) = \mathbb{E}_{r,s,a}\left[\|r - \theta^\top(\phi(s,a) - \hat{V}\psi(s,a))\|^2\right] + \lambda_2\|\theta\|^2, \tag{24}$$

where $\lambda_2 > 0$ is another regularization parameter. Stage 2 corresponds to a ridge linear regression from $\phi(s,a) - \hat{V}\psi(s,a)$ to r, and also has a closed-form solution.

In KIV, from the characteristics of RKHS functions one can learn a non-linear Q function while retaining the closed-form solution of the two-stage regressions. In DFIV, because of the use of adaptive features for $\phi(s,a)$ and $\psi(s,a)$ parameterized with neural networks they learn those features by alternating the two regression stages, which enables one to learn a more flexible Q function compared to KIV. In this paper, we consider a variant of DFIV that regresses $\phi(s,a) - \gamma\phi(s',a')$ instead of $\phi(s',a')$ on instrumental variables in stage 1, with details explained in Appendix A. We found this approach is more stable than the original version in the experiments. While we can derive a similar variant for KIV, we did not notice an improvement in performance.

4.4 Generalized Method of Moments

A family of non-linear IV methods is based on the moment restrictions derived from Assumption 1 of instrumental variables,

$$\mathbb{E}(\varepsilon|Z) = \mathbb{E}(Y - f(X)|Z) = 0, \forall Z. \tag{25}$$

In the linear setting, we require the following unconditional moment to be zero, whose solution is Equation (11),

$$\mathbb{E}(\varepsilon Z) = \mathbb{E}[(Y - f(X))Z] = 0. \tag{26}$$

In the non-linear setting, by defining a set of potentially infinitely many test functions of Z, $g \in \mathcal{G}$, we require all the unconditional moments to be zero,

$$\Psi(f,g) = \mathbb{E}_{X,Y,Z}\left[(Y - f(X))g(Z)\right] = 0, \forall g. \tag{27}$$

The unified solution in the family of Generalized Method of Moments (GMM) (Hansen, 1982) is a saddle-point of the following minimax objective function,

$$f^* = \arg\inf_{f\in\mathcal{F}} \sup_{g\in\mathcal{G}} \Psi_n(f,g) + R_f(f) - R_g(g)\,, \tag{28}$$

where g is optimized to find the largest violation of the moment condition for the current estimate of f and the expectation in Ψ is the empirical estimate from a dataset of size n. $R_f(f)$ and $R_g(g)$ are regularization terms for f and g, respectively, for identifiability.

In the context of RL, this objective is thus given by

$$
\begin{aligned}
&Q^* = \arg\inf_{Q\in\mathcal{Q}} \sup_{g\in\mathcal{G}} \Psi_n(Q,g) + R_f(Q) - R_g(g)\,, \\
&\text{with } \Psi_n(Q,g) = \mathbb{E}_{s,a\sim\mu_b,s'\sim P,r\sim R,a'\sim\pi}\left[(r - Q(s,a) + \gamma Q(s',a'))g(s,a)\right]. \qquad (29)
\end{aligned}
$$

Here g acts to find the largest moment of the TD error. Variants of GMM method differ in the choice of the function space $\mathcal{Q}$, $\mathcal{G}$ and the regularization functions.

4.4.1 Deep GMM

Inspired by the optimally weighted GMM method in linear IV problems, Bennett et al. (2019b) propose the Deep GMM method. In the linear setting with a fixed set of feature bases, $f_1, ..., f_m$, the test function g is defined as a linear combination of the bases $g(Z) = v^T f(Z)$. The optimally weighted GMM (OWGMM) yields the minimum variance estimate of the linear weights θ by setting $R_f(f) = 0$ and $R_g(v) = \frac{1}{4}v^T C v$ (see definition of the matrix C in Bennett et al. (2019b) which is a function of the optimal weights $\tilde{\theta}$.

Bennett et al. (2019b) extend OWGMM to the non-linear setting and parameterizes both f (Q in RL) and g with neural networks, Q_θ and g_ϕ, and uses the following regularization,

$$
\begin{aligned}
R_Q(Q_\theta) &= 0\,, \\
R_g(g_\tau) &= \frac{1}{4}\mathbb{E}_{s,a\sim\mu_b,s'\sim P,r\sim R,a'\sim\pi}\left[g_\tau^2(s,a)(r - Q_{\tilde{\theta}}(s,a) + \gamma Q_{\tilde{\theta}}(s',a'))^2\right], \qquad (30)
\end{aligned}
$$

where $\tilde{\theta}$ should be a consistent estimator of the true parameter value. In practice, Bennett et al. (2019b) suggests to set $\tilde{\theta}$ to the latest estimate of θ in the iterative optimization process.

Deep GMM reduces to OWGMM in the linear function setting and results in the most efficient estimator in that case. The efficiency of this particular regularization scheme is not discussed in the non-linear case in Bennett et al. (2019b). However, Dikkala et al. (2020, Sec. 6) argues that such re-weighting is not required if one simply wants a fast convergence rate in the projected RMSE, defined next in Equation (31).

4.4.2 Adversarial GMM Networks (AGMM)

Dikkala et al. (2020) consider the general minimax objective function in Equation (28) and focus on the generalization performance of the projected residual mean squared error, defined as

$$
\sqrt{\mathbb{E}_Z\left[\left(\mathbb{E}_X\left[\hat{f}(X) - f_0(X)|Z\right]\right)^2\right]}\,, \qquad (31)
$$

where $\hat{f}$ is the optimal solution of Equation (28) on a dataset, and f_0 is the optimal solution of $\arg\inf_{f\in\mathcal{F}}\sup_{g\in\mathcal{G}}\Psi(f,g)$.

The authors discuss the choice of the function spaces and regularization constants in order to derive a bound on the estimation error rate. They also instantiate the objective in different function spaces, including Reproducing Kernel Hilbert Spaces, High-dimensional Sparse Linear Function Spaces, Neural Networks, etc. When applying their theoretical findings to neural networks, the regularizers on the function $Q_\theta(s,a)$ and $g_\tau(s,a)$ in the RL context are as follows[3],

$$\begin{aligned} R_Q(Q_\theta) &= a\|\theta\|_2^2\,, \\ R_g(g_\tau) &= b\|\tau\|_2^2 + \mathbb{E}_{s,a\sim\mu_b}\left[g_\tau^2(s,a)\right]\,, \end{aligned} \tag{32}$$

where a and b are hyper-parameters for the L_2 regularization on the network parameters.

4.4.3 Adversarial Structural Equation Models (ASEM)

Luofeng et al. (2020) introduce the generalized structural equation model (SEM) problem with IV being a special instance,

$$Af = b, \tag{33}$$

where $A : \mathcal{H} \to \mathcal{E}$ is a conditional expectation operator between two separable Hilbert spaces of square integrable functions, $f \in \mathcal{H}$ is the structural function of interest and $b \in \mathcal{E}$ is known or can be estimated. When applied to IV regression, this reduces to the same conditional moment restriction in Equation (25), under conditions $f \in L^2(\mathcal{X})$, $Af = \mathbb{E}\left[f(X)|Z\right] \in L^2(\mathcal{Z})$, $b = \mathbb{E}\left[Y|Z\right] \in L^2(\mathcal{Z})$.

With a similar dual formulation as other GMM methods introduced in preceding sections, Luofeng et al. (2020) propose a general solution to the SEM problem using an adversarial training approach with a minimax objective function, and both functions are parameterized with neural networks:

$$f^* = \operatorname*{arg\,min}_{f\in L^2(\mathcal{X})} \max_{g\in L^2(\mathcal{Z})} \mathbb{E}[(f(X) - b(Z))g(Z)] + \frac{\alpha}{2}\mathbb{E}[f(X)^2] - \frac{1}{2}\mathbb{E}[g(Z)^2]\,. \tag{34}$$

They establish the consistency of the estimator theoretically under regularity conditions. In the RL context, the two networks should satisfy $Q_\theta \in L^2(\mathcal{S}\times\mathcal{A})$, $g_\tau \in L^2(\mathcal{S}\times\mathcal{A})$, and the corresponding regularizers are

$$R_Q(Q_\theta) = \frac{\alpha}{2}\mathbb{E}_{s,a\sim\mu_b}\left[Q_\theta^2(s,a)\right]\,, \qquad R_g(g_\tau) = \frac{1}{2}\mathbb{E}_{s,a\sim\mu_b}\left[g_\tau^2(s,a)\right]\,, \tag{35}$$

where α is a hyper-parameter. In this paper, we consider an additional L_2 regularization on the network parameters θ and τ as in AGMM, and tune the associated hyper-parameters together with α. It is easy to see that AGMM described in the previous subsection is a special case of ASEM with $\alpha = 0$ (the different multiplier in R_g does not change the solution of Q_θ after rescaling g_τ accordingly).

4.4.4 Other adversarial IV methods

Lewis and Syrgkanis (2018) propose another AGMM algorithm that minimizes the L_2 norm of the vector

$$(\Phi_n^{(1)}, \Phi_n^{(2)}, \ldots, \Phi_n^{(m)})$$

consisting of a finite set of moments (test functions), and solve it with a no-regret on line learning algorithm in an adversarial training fashion. This corresponds to a finite set of $\mathcal{G}$ and no regularizations in the objective of Equation (28).

3. Personal communication with the authors

Muandet et al. (2019) consider a slightly different conditional moment objective,

$$f^* = \arg\min_{f\in\mathcal{F}} \mathbb{E}_{Y,Z}\left[\left(Y - \mathbb{E}_{X|Z}\left[X|Z\right]\right)^2\right] . \tag{36}$$

Note that the expectation with respect to Y is outside of the square operator compared to Equation (25). This objective leads to a different dual formulation,

$$f^* = \arg\inf_{f\in\mathcal{F}} \sup_{g\in\mathcal{G}} \Psi_n(f,g) - \frac{1}{2}\mathbb{E}_{Y,Z}\left[g(Y,Z)\right] , \tag{37}$$

where the adversarial function g is defined in the joint domain of Y and Z. Muandet et al. (2019) assume both f and g lie in reproducing kernel Hilbert spaces, which allows to obtain an analytical expression for the solution.

4.4.5 Related OPE Methods

Dual formulations have also been considered in the OPE literature, see e.g. (Nachum et al., 2019; Yang et al., 2020; Mousavi et al., 2020; Uehara et al., 2020). Most of these works apply a saddle-point optimization method to estimate the density ratio between the distributions of state-action pairs under the evaluation policy π and the behavioural policy π_b, each such distribution corresponding to the probability of encountering a state-action pair and averaging over time using the discount factor γ. This is an alternative approach to OPE based on importance sampling rather than the value function. Yang et al. (2020) point out that estimating the density ratio function is the dual formulation of estimating the Q function with their particular objective. The closest work to our IV interpretation is Uehara et al. (2020). The objective of their MQL algorithm is the squared Ψ_n without regularization,

$$Q^* = \arg\inf_{Q\in\mathcal{Q}} \sup_{g\in\mathcal{G}} \Psi_n^2(f,g) .$$

5. Experiments

In this section, we first demonstrate the advantage of IV methods over the biased DBRM and iterative Fitted Q Evaluation (FQE) algorithm in the OPE problem using a simple MDP environment, and conduct an ablation study to investigate the influence of the model and algorithm parameters. We then propose a set of OPE benchmark problems with a varying level of randomness in the system dynamics. We evaluate the performance of all the non-linear IV methods in the paper on those problems, and compare to DBRM, FQE and distributional FQE which are state-of-the-art OPE method (Fu et al., 2021).

5.1 Simple MDP problem

Let us consider a simple MDP with 100 discrete states allocated uniformly along the interval $[-2, 2]$: $s_i = -2 + \frac{4}{100}i, i \in \{0, 1, \ldots, 99\}$. The agent always starts at the first state s_0 in every episode and terminates at the last state s_{99}. There is only a single action, $a = \text{right}$, in every state to move to right, and therefore the policy is always fixed. The state is transitioned to the right neighboring state with a probability of $p > 0$ or stays in the same location. It is easy to show that the resulting state distribution $\mu(s)$ pooled from different steps across the trajectory, i.e. the offline data distribution, is uniform among all the non-terminating states $i \in \{0, 1, \ldots, 98\}$, whatever the value of p and $\mu(s_{99}) = p\mu(s_{98})$. The reward function is defined with a Gaussian kernel as $R = \exp(-\frac{s^2}{0.2^2})$, and is illustrated in Figure 4 together with $Q(s, a = \text{right})$.

As there is only a single policy in this environment, the target policy is the same as the behavior policy. We sample a dataset of $N = 10^5$ transitions with $p = 0.5$, and estimate the state value using a fixed set of $D = 90$ Gaussian kernel features $\phi_j(s) = \exp\left(\frac{(s-(-2+(4/D)j))^2}{0.1^2}\right), j = \{0, 1, \ldots, 89\}$. For this linear instrumental variable regression problem, we compare the LSTD-Q method in Section 3 with DBRM, which reduces to a naive least square minimization algorithm in this case, and FQE, which alternates solving a least squared minimization in Equation (18) with the linear weights θ in Y being fixed and replacing those weights with the solution.

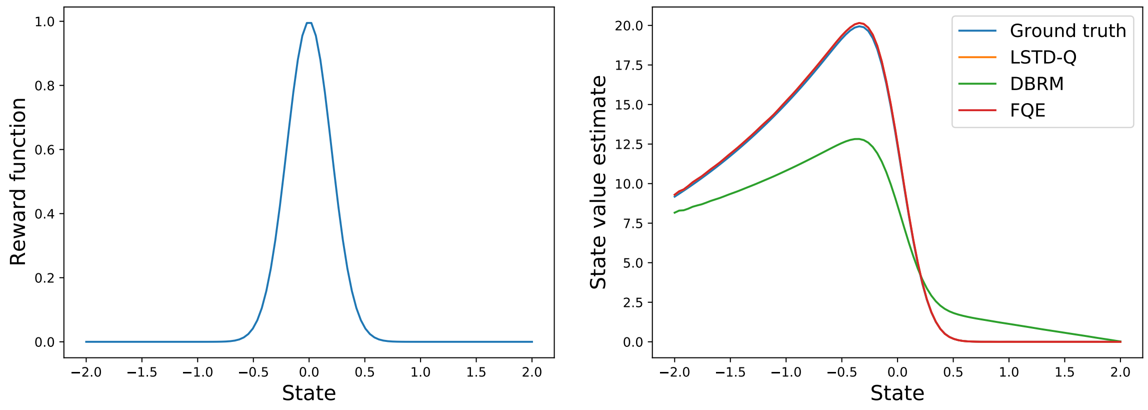


Figure 4: Simple MDP. Left: reward function. Right: The ground truth and estimated $Q(s, a = \text{right})$ by LSTD-Q, DBRM and FQE with $p = 0.5$. LSTD-Q overlaps FQE.

The estimates of the Q function for all methods are shown in Figure 4. The estimate of LSTD-Q matches the ground truth value very well, while DBRM - which ignores the confounding problem - leads to a heavily biased estimate as expected. The estimate of FQE also matches the ground truth, but requires multiple iterations to converge to the solution as shown in Figure 6 (right). This is because the TD formulation of FQE can only propagate the state value information along the reverse order of state dynamics $s' \to s$ by one step at every iteration.

Next, we conduct an ablation study to investigate how the advantage of IV method depends on the following four variables: dataset size N, feature dimensions D, transition randomness $(1-p)$, and the extent of off-policyness. While the target policy always matches the behavior policy in this problem, we create an offline dataset with a shifted distribution by sampling the states with the following distribution $\mu(s) \propto \exp(\alpha s)$, where $\alpha = 0$ corresponds to the original uniform distribution, and a larger value of α leads to a shifted distribution towards the right end of the state space.

We display the absolute error of the estimated state value at the initial state $Q(s_0, a = \textit{right})$ by LSTD-Q and DBRM in Figure 5. The error of FQE is not shown because it

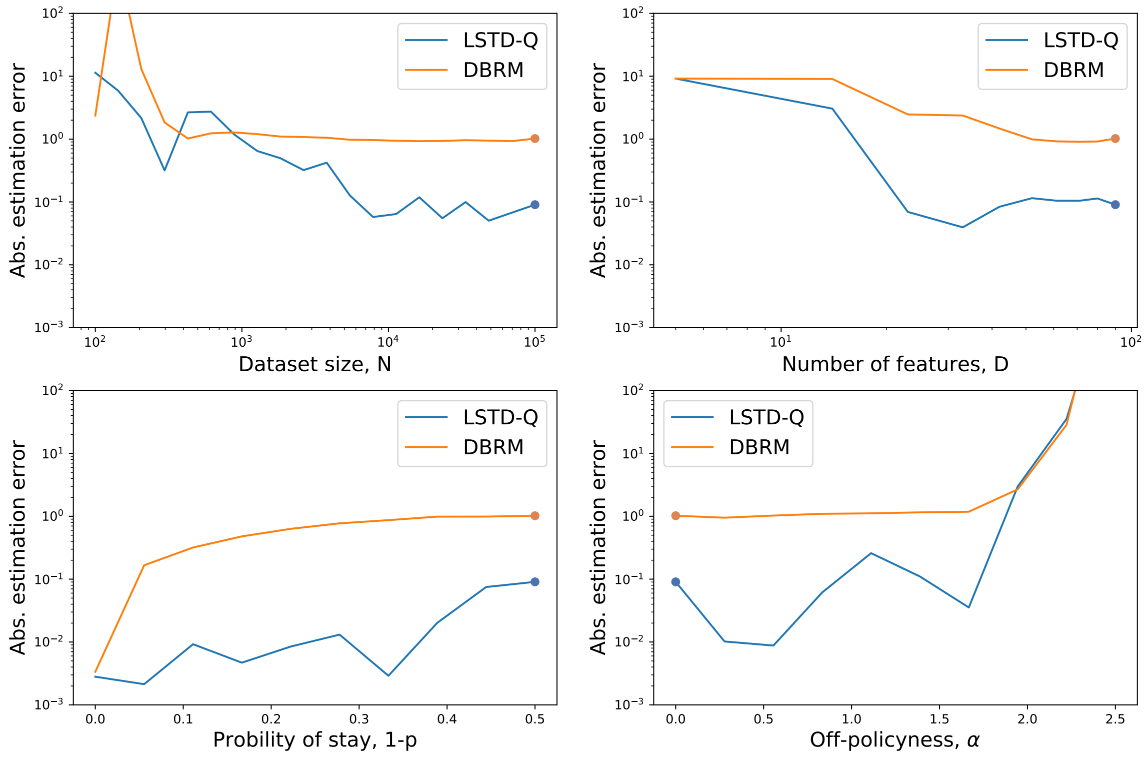


Figure 5: Simple MDP ablation study. Each plot shows the absolute error of $Q(s_0, a = \text{right})$ as a function of dataset size, number of features, stochasticity of the dynamics, and the distribution shift between the dataset and that generated by the target policy. The dots represent the default setting in Figure 4.

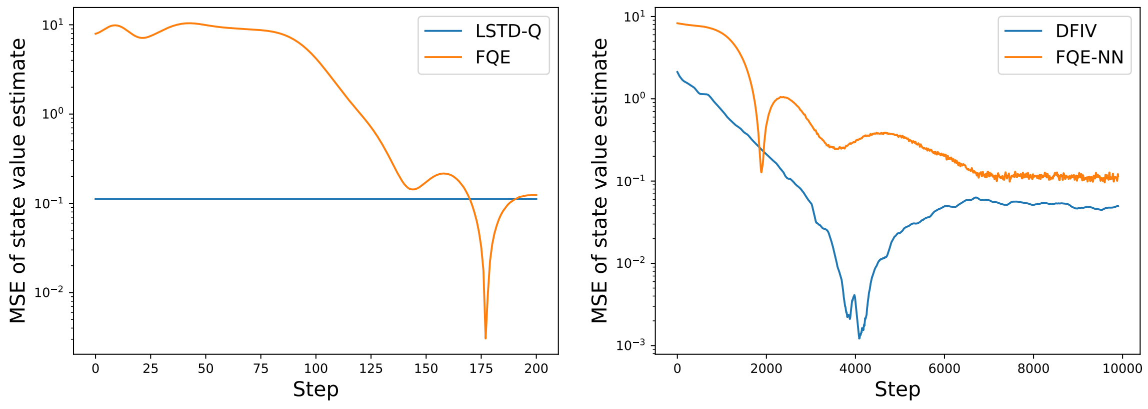


Figure 6: The absolute error of $Q(s_0, a = \text{right})$ as a function of optimization iteration in the linear (left) and non-linear setting (right). LSTD-Q is solved analytically in the linear setting.

converges to the same solution as LSTD-Q in the linear case. We find the error of LSTD-Q increases and eventually becomes on par with DBRM when we reduce the size of the dataset, or increase the data distribution shift, which also decreases the effective dataset size. The error of LSTD-Q also increases when we reduce the number of features, which will lead to model misspecification, and could violate the IV requirement that the residual needs to have zero mean. Lastly, we see that DBRM is unbiased when the dynamics are deterministic, $p = 1$, but the bias increases quickly when p decreases. In contrast, the error of LSTD-Q increases but much more slowly, which we suspect is due to an increasingly diverse distribution of transitions (s, s') that requires more data to learn the estimate accurately.

When we use a non-linear value function, the solutions given from IV techniques do not coincides with FQE, and both methods require iterative optimization. Nonetheless, the policy evaluation algorithms based on non-linear IV (e.g. DFIV) preserve the relatively faster convergence rate than FQE in this example, as shown in Figure 6 (right). Here we estimate the value function from the raw state s and estimate an multi-layer perceptron (MLP) with two hidden layers, each with 50 units and ReLU activation function. Both methods use a learning rate of 10^{-4}.

From this ablation study, we demonstrate that the advantages of using an IV method over a simple method like DBRM and FQE can be significant, but that the magnitude of this benefit depends on multiple variables. On a more complex and comprehensive OPE benchmark, we study in the following section whether recent non-linear IV methods are competitive with state-of-the-art OPE methods such as FQE and Distributional FQE (Fu et al., 2021).

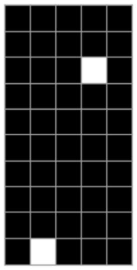
5.2 OPE benchmark problems

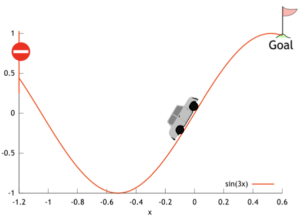
5.2.1 Environments

We consider a list reinforcement learning environments from two widely used task collections: Behaviour Suite (BSuite) (Osband et al., 2019) and DeepMind Control Suite (DM Control) (Tassa et al., 2020). BSuite is a collection of traditional RL environments with a discrete action space. We choose three environments that can be solved by a standard DQN agent (Mnih et al., 2015): Catch, Cartpole, and Mountain Car. DM Control is a collection of physics-based simulation environments, using MuJoCo physics, for studying continuous control problems with a continuous action space. We choose four environments that can be solved by a standard D4PG agent (Barth-Maron et al., 2018): Cartpole Swingup, Cheetah Run, Walker Walk, Humanoid run. A brief description of each of the seven environments is provided as follows with illustrations in Figures 7 and 8:

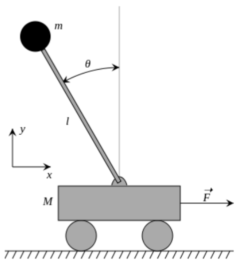
- BSuite
  - Catch: A 10x5 Tetris-grid with single block falling per column. The agent can move left/right in the bottom row to ‘catch’ the block.
  - Mountain Car: The agent drives an underpowered car up a hill (Moore, 1990).
  - Cartpole: The agent can move a cart left/right on a plane to keep a balanced pole upright (Barto et al., 1983).
- DM Control

– Cartpole Swingup: Swing up and balance an unactuated pole by applying forces to a cart at its base. The physical model conforms to Barto et al. (1983).
– Cheetah Run: A running planar biped based on Wawrzyński (2009).
– Humanoid Run: A simplified humanoid with 21 joints, based on the model in Tassa et al. (2012).
– Walker Walk: An improved planar walker based on the one introduced in Lillicrap et al. (2015).


(a) Catch

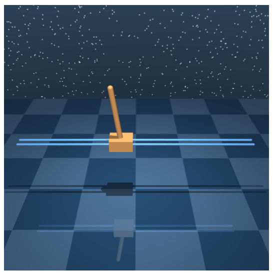


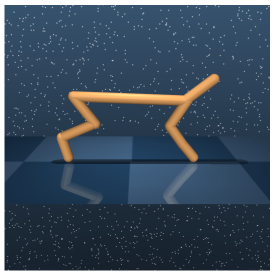
(b) Mountain Car



(c) Cartpole

Figure 7: Three BSuite tasks


(a) Cartpole Swingup


(b) Cheetah Run

(c) Humanoid Run

(d) Walker Walk

Figure 8: Four DM Control tasks

Every environment has a real-valued state space $\mathcal{S} \subseteq \mathcal{R}^{D_S}$, and a discrete action space $\mathcal{A} = \{0, 1, 2, \ldots, D_A - 1\}$ for BSuite tasks and continuous action space $\mathcal{A} \subseteq \mathcal{R}^{D_A}$ for DM Control tasks, respectively.

All environments have deterministic system dynamics, which would be hard to find in real-world applications. In order to study how IV methods can address the bias introduced through the confounding variable of the next state s', we modify the original environments with additional randomness in the dynamics. Specifically, for BSuite environments, we randomly replace the agent action by a uniformly sampled action with a probability of $p \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$, resulting in a stochastic transition distribution,

$$\tilde{P}(s'|s,a) = (1-p)P(s'|s,a) + \frac{p}{D_A}\sum_{\tilde{a}=0}^{D_A} P(s'|s,\tilde{a}) \,. \quad (38)$$

For DM Control environments, we insert some Gaussian noise to the agent action, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ with $\sigma \in \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$, resulting in a stochastic transition distribution,

$$\tilde{P}(s'|s, a) = \int_{\varepsilon} P(s'|s, a + \varepsilon)\mathcal{N}(\varepsilon|0, \sigma^2)\mathrm{d}\varepsilon \,. \tag{39}$$

Note that the randomly perturbed action is transparent to the agent. When $p = 0$ or $\sigma = 0$, it reduces to the original deterministic environment.

Overall, we create 42 environments for our experiments with 7 different tasks and 6 levels of dynamics randomness per task. The state and action dimensions are provided in Table 1 and 2.

		Catch	**Mountain Car**	**Cartpole**
Dimensions	D	50	3	6
	A	3	3	3
Target policy	Train Episodes	2K	500	1K
Near-policy Dataset	Train Episodes	20K	5K	1K
	Train Transitions	180K	759K∼1.59M	325K∼833K
	Valid Episodes	2K	500	100
	Valid Transitions	19K	75.1K∼158K	33K∼83K
Pure Offline Dataset	Train Episodes	1.8K	450	900
	Train Transitions	16.2K	75K∼160K	512K∼671K
	Valid Episodes	200	50	100
	Valid Transitions	1.8K	8K∼14K	71K∼80K

Table 1: BSuite tasks. Every Catch episode has 9 transitions. The average length of an episode in the Mountain Car/Cartpole increases/decreases as the level of environment randomness p increases. The training and validation data ratio is 9:1.

		Cartpole Swingup	**Cheetah Run**	**Humanoid Run**	**Walker Walk**
Dimensions	D	5	17	67	24
	A	1	6	21	6
Target policy	Train Episodes	300	4K	100K	1.5K
Pure Offline Dataset	Train Episodes	270	3.6K	9K	1.35K
	Train Transitions	270K	3.6M	9M	1.35M
	Valid Episodes	30	400	1K	150
	Valid Transitions	30K	400K	1M	150K

Table 2: DM Control Suite tasks. Every episode has 1000 transitions. The training and validation data ratio is 9:1. The offline dataset of the Humanoid Run task is subsampled with by 10% from the 100K episodes generated from the training process.

5.2.2 Target policies for evaluation and offline datasets

We run the default DQN agent for every random level of the three BSuite tasks and the default D4PG agent for the four DM Control tasks from the ACME library (Hoffman et al., 2020) until the episodic return does not increase noticeably any more. The number of training episodes is provided in Table 1 and 2. We use the learned policy with a small amount of action noise as the target policy for evaluation. For the DQN agent, we use an ε-greedy policy, that is, taking the greedy action of $\arg\max_a Q(s,a)$ with a probability $1-\varepsilon$ and a random action otherwise where $\varepsilon = 0.1$. For the D4PG agent, we use the learned policy network with an additive Gaussian noise with a standard deviation of 0.2.

We consider two types of offline datasets with a different level of difficulty for OPE: an easy near-policy dataset and a hard pure offline dataset. The size of the dataset for each environment is given in Table 1 and 2.

For the easy dataset, we consider the three BSuite tasks only. For every task we define the behavior policy in the same way as the target policy except with a slightly larger exploration probability of $\varepsilon = 0.3$ instead of 0.1. Therefore the behavior policy is close to the target policy. We then play the behavior policy in the corresponding environment repeatedly and collect a sufficiently large off-policy dataset for each task.

For the hard dataset, we restart the agent training process with a different random seed and collect the episodes along the training. The resulting dataset consists of episodes generated from various partially trained policies, some of which are close to the initial random policies while others are close to a well-optimized policy. The dataset is then split randomly into training and validation subsets with a ratio of 9:1. This is akin to the data generation protocol in the RL Unplugged dataset (Gulcehre et al., 2020) with two differences: (1) we modify environments with random dynamics from the original deterministic environment, (2) the dataset is collected from the training process of a different random seed, therefore the policies used to generate the dataset could be substantially different from the target policy to be evaluated.

5.3 Experiment setup and hyper-parameter selection

We compare a list of representative non-linear IV methods, including Kernel IV (KIV), Deep IV, Deep Feature IV (DFIV) and three adversarial IV methods: Deep GMM, Adversarial GMM Networks (AGMM), Adversarial approach to structural equation models (ASEM). We also include as baselines the deterministic Bellman residual minimization (DBRM) and two variants of the fitted Q evaluation methods with a deterministic (FQE) and distributional (DFQE) Q representation respectively. (D)FQE was shown to be the best performing OPE algorithm in a recent benchmark paper (Fu et al., 2021) under a similar task setting as in this paper.

All algorithms except KIV use the same network architecture to estimate the Q function as in the trained agent for a fair comparison. For BSuite tasks, the Q network is an MLP with layer size 50-50-1 and ReLU activation. The input is a concatenation of the flattened observation and one-hot encoding of the discrete action variable. For DM Control tasks, it is an MLP with layer size 512-512-256-1, ELU activation and a layer normalization after the first hidden layer. The input is the concatenation of the flattened observation and action variables. The architecture of additional networks in each algorithm, such as the generative model in Deep IV and the adversarial function network in AGMM, ASEM and DeepGMM are selected as part of the hyper-parameter search procedure that will be discussed later. We use OAdam for adversarial methods as suggested by Bennett et al. (2019b); Dikkala et al. (2020) and Adam for other methods.

We compare all the algorithms with respect to the accuracy of estimating the target policy value $\rho(\pi)$ (Eq. 2), i.e., the expected cumulative discounted reward from the initial state distribution. The estimate is computed as

$$\hat{\rho}(\pi) = \mathbb{E}_{s \sim \mu_0, a|s \sim \pi}[\hat{Q}^\pi(s, a)], \tag{40}$$

where $\hat{Q}^\pi$ is given by each algorithm under comparison. We normalize the policy value into a range of $[0, 1]$ for ease of comparison across environments:

$$\rho_{\text{Norm}} = \frac{\rho - \rho_{\min}}{\rho - \rho_{\max}} \tag{41}$$

where $\rho_{\min/\max} := -1/1$ for BSuite Catch and $\rho_{\min/\max} := R_{\min/\max} \sum_{t=0}^{1000} \gamma^t$ for other environments. We measure the accuracy in terms of the absolute error $|\hat{\rho}_{\text{Norm}} - \rho_{\text{Norm}}|$ in this paper. Other metrics such as the policy ranking correlation and regret have been considered in the literature (Paine et al., 2020; Fu et al., 2021) when multiple target policies are available in the same environment. Our experiment setup does not meet that condition and those metrics are hence not included.

Some of the algorithms implemented are sensitive to the choice of hyper-parameters. In order to ensure a fair comparison, we run a thorough hyper-parameter search for every algorithm in every environment. We randomly sample up to 100 hyper-parameter settings for every algorithm and choose the setting with the best metric on a held-out validation dataset. Due to the large number of tasks (environment and dataset combinations), we search for the best hyper-parameter at one environment random level in every dataset ($p = 0.2$ for BSuite and $\sigma = 0.4$ for DM Control tasks) and apply the same setting to other levels. Once the hyper-parameter is selected, we run each algorithm with 5 random seeds for every task to measure the mean and variance of the estimate.

Note that due to the state distribution shift between the behavior policy and target policy, the best hyper-parameter setting on the validation dataset from the behavior distribution does not guarantee a good performance when evaluating the target policy value. It remains an open research problem how to select the hyper-parameter for OPE given one does not have access to the ground truth value (Paine et al., 2020). We explain the metric adopted for selecting the hyper-parameters of each algorithm in details in Appendix B.

5.4 Results

5.4.1 Near-policy dataset

We first study the performance of all the algorithms on the easy offline dataset with a near-policy data distribution and sufficiently larget data size. Figure 9 shows the scatter plot of the estimated policy value versus the ground-truth value. Each dot represents the mean and 1-standard deviation of the estimate from 5 random runs for every environment and every random level. Additionally, we show the absolute error of the estimates for each task

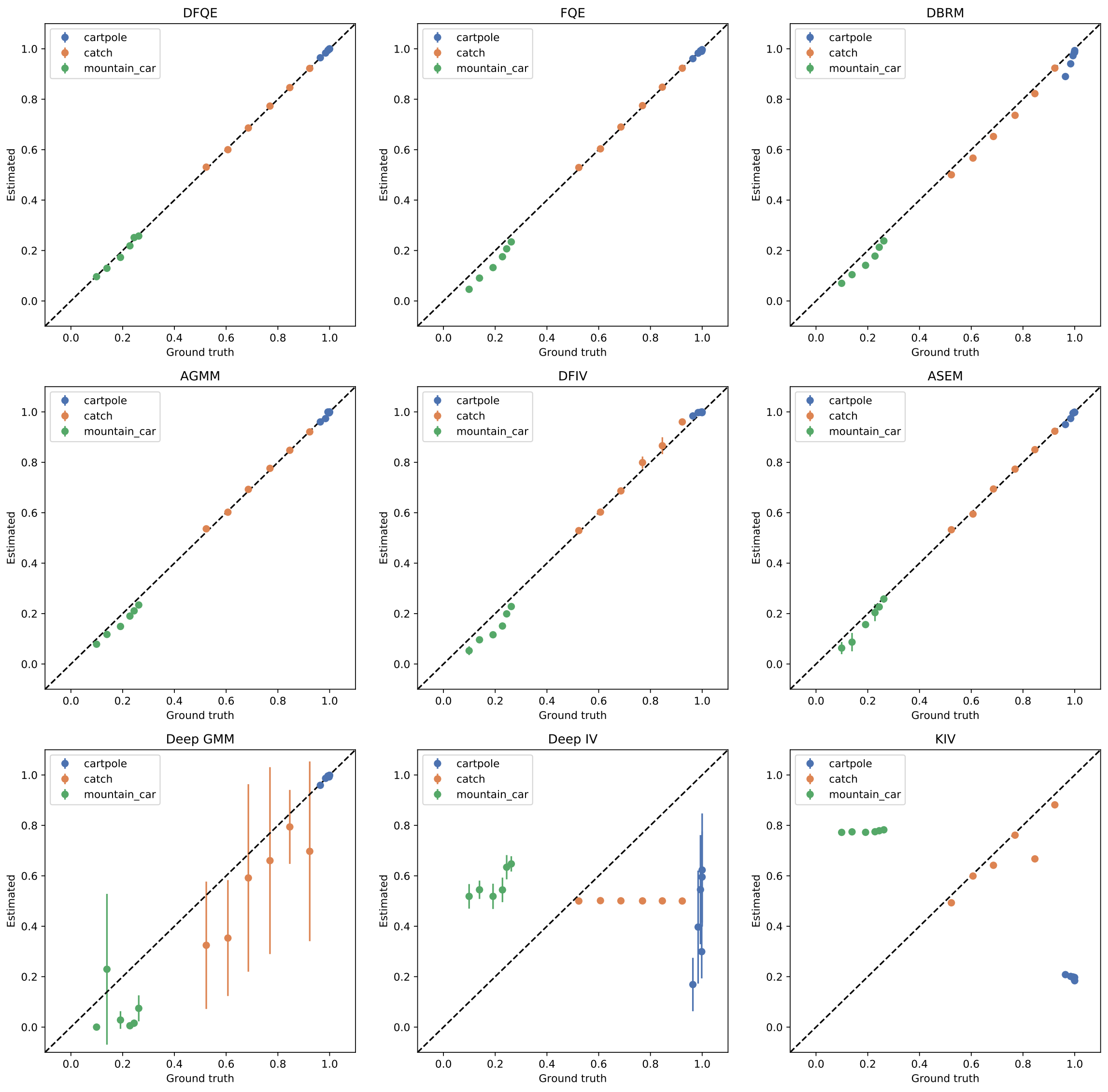


Figure 9: Estimated policy value vs groundtruth with the near-policy dataset

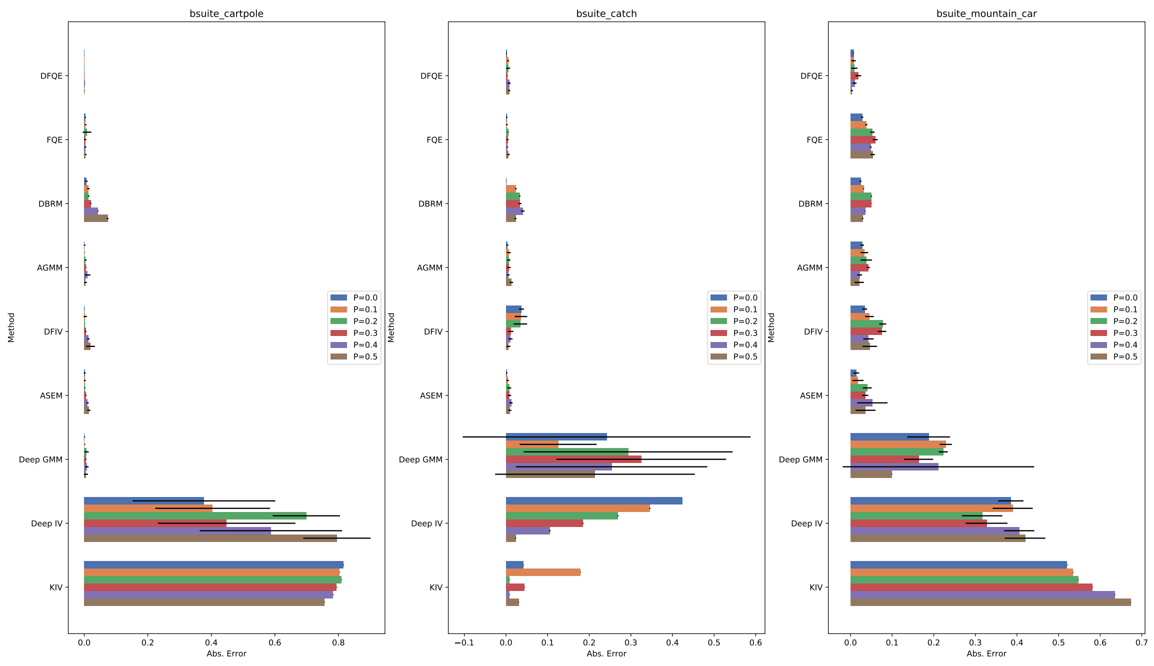


Figure 10: Absolute error of policy value estimation with the near-policy dataset

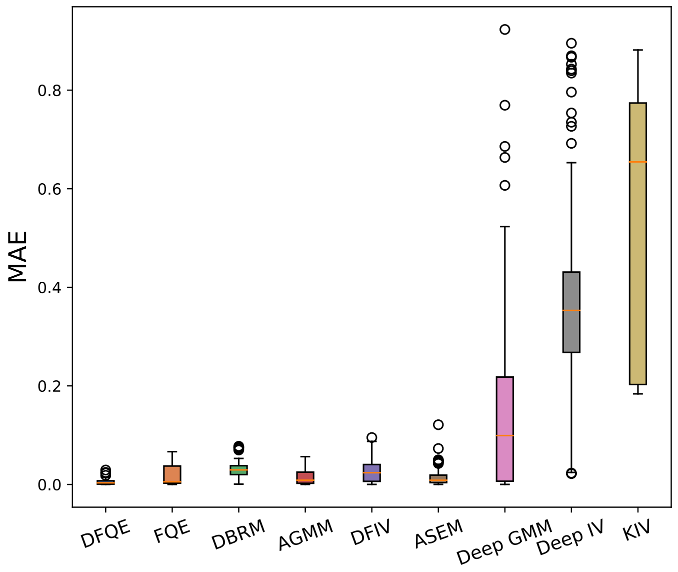


Figure 11: Distribution of the absolute error across all tasks with the near-policy dataset

in Figure 10 and a box-plot of the distribution of errors pooled from all tasks as a summary in Figure 11.

Most algorithms provide an accurate estimate of the policy value on this dataset, except Deep GMM, Deep IV, and KIV. We observe experimentally that the training process of Deep GMM is very unstable compared the other two adversarial approaches (AGMM and ASEM). Figure 12 shows a typical trajectory of the training loss and the estimated policy value along the training process. We suspect it is due to the use of the optimal weighting $\mathcal{C}_{\tilde{\theta}}$ in the regularization term. Deep IV fails with both a large mean absolute error and variance. In particular, in the Catch environment, the generative model completely fails to predict the next state. This illustrates the challenge of modeling a moderately-high dimensional state space (50-dimensions) using a simple feed-forward network to predict the parameters of a mixture of Gaussian generative model as proposed in Hartford et al. (2017b). We expect the performance to improve with a more sophisticated generative model as evidenced by recent model-based OPE work in Zhang et al. (2021). KIV fails in the Cartpole and Mountain Car environments with a large error too. This is in agreement with the observation by Xu et al. (2021) that shallow features are not capable of modeling complex structural functions.

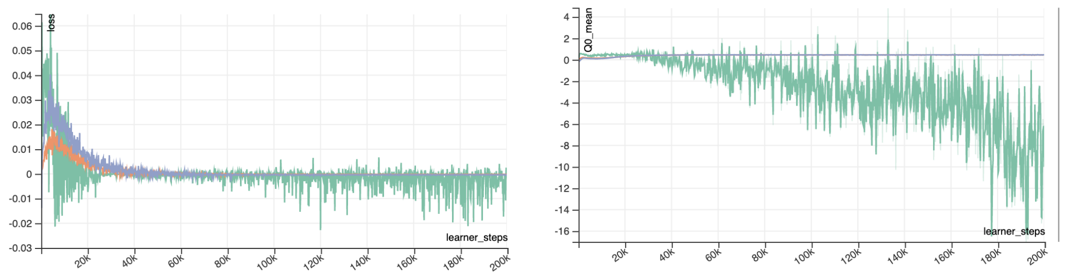


Figure 12: A typical trajectory of the training loss (left) and the estimated unnormalized policy value (right) of DeepGMM (green), AGMM (orange), and ASEM (blue) as a function of training steps. The plots are obtained from the task of BSuite Catch with environment random level $p = 0.2$. A valid policy value should be in the range of $[-1, 1]$. The curves of AGMM and ASEM overlap on even other in the right plot.

Figure 11 shows that DFQE is the most accurate OPE method on this dataset. Among the IV methods, AGMM, ASEM and DFIV all give fairly small estimation errors across all tasks while AGMM performs the best.

5.4.2 Bias in DBRM

The bias in DBRM due to ignoring the confounding variable s' is clearly observed in the Cartpole and Catch environments where the error increases with the level of randomness.

We further inspect its bias in the Q estimation in the following ablation study using the Catch environment with an environment random level $p = 0.4$ as an example. After AGMM converges, we take the last hidden layer of the value network as a set of deep features, and re-fit the weights of the output linear layer. Given the fixed features, it reduces to a standard IV problem with linear functions. We fit the linear weights with two methods, DBRM and LSTD-Q, and compare the new estimated Q value of the 15 unique initial state-action pairs of Catch together with those obtained by AGMM in Figure 13. The estimates of LSTD-Q and AGMM match the ground truth very well within a 95% confidence interval while DBRM estimate shows a significantly large error. This suggests that DBRM suffers, as expected, from a bias in the presence of environment randomness even when using a good set of deep features.

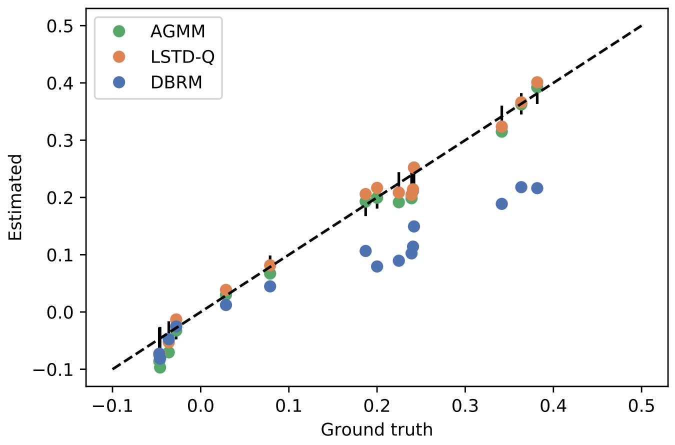


Figure 13: Estimated policy value in the DBRM ablation study. Every dot represents one initial state-action pair. The error bar shows the 95% confidence interval of the ground-truth value estimation.

5.4.3 Pure offline dataset

We then evaluate all the algorithms on the hard dataset in both BSuite and DM Control environments. The episodes are generated from a mixture of partially trained policies from a different run, and the distribution of states is likely to have a quite different coverage from the distribution generated by the target distribution.

We show the scatter plot of the estimated policy value versus ground truth in Figure 14, the absolute error of the estimation per task and pooled together in Figure 15 and Figure 16 respectively.

Apparently, none of the algorithms considered here outperform all the other algorithms in all tasks. For example, DFQE and FQE perform relatively well in most tasks but fail on the Humanoid Run and Walker Walk control tasks. In contrast, AGMM estimates the policy value most accurately among all methods on the challenging Humanoid Run tasks but has higher error in BSuite catch, DM Control Cheetah Run and Walker Walk tasks. This observation is consistent with the results in (Xu et al., 2021) that benchmarks other OPE algorithms in deterministic environments, where they observe that "no evaluated algorithm attains near-maximum performance under any metric".

Nonetheless, by inspecting the estimation error in Figure 15 and Figure 16 we notice that the relative performance of IV methods is roughly in agreement with the results observed in the near-policy dataset. AGMM is the most robust IV method implemented here. ASEM and DFIV has a similar median error in all tasks but have a few large estimation errors in some tasks such as Walker Walk and BSuite Cartpole. Given the similarity between AGMM and ASEM, we conjecture that the relatively worse performance of ASEM is due to a bad choice of the hyper-parameters as it has more regularization hyper-parameters to tune. KIV and Deep IV obtain considerably larger error than all the other methods, followed by Deep GMM, which has large variance in a few tasks such as Mountain Car and Walker Walk.

Surprisingly, DBRM that simply ignores the stochasticity of the dynamics displays the most robust performance among all the algorithms. It converges quickly and almost always provides estimates of lowest variance among different runs compared to the other more complicated algorithms, at the price of a somewhat higher median error.

Nonetheless, this observation is indeed understandable. The estimation error of the policy value comes from multiple sources including: (1) the approximation error due to lack of model capacity (2) generalization error due to the limited dataset size (3) the off-policyness, or the divergence between the offline data distribution and the target policy distribution, which affects the effective size of the dataset (4) optimization error, affected by the optimization algorithm, and (5) the bias of ignoring the stochasticity of the dynamics. As demonstrated in Section 5.1, the benefits of IV methods depend on multiple conditions. Compared to the large near-policy dataset on BSuite, the pure offline dataset, including the continuous control tasks has higher modeling challenges, less data, and a larger divergence between the behavior and target distributions. All the other sources of error may dominate the bias present in DBRM, and due to the simplicity of DBRM as a single minimization problem, it is simpler to optimize than the loss of the other algorithms.

6. Conclusion

The regression problem one needs to solve when estimating a Q-function suffers from confounding as a result of the inputs and the output noise being correlated. If this confounding is ignored, one can obtain significantly biased Q-function estimates. We have shown here that fixing the target Q-network in DQN and FQE can be thought of as a way to overcome this confounding problem. As first suggested in Xu et al. (2021), another approach overcoming this problem consists of using IV regression techniques.

By exploiting this observation and bringing together the literature on IV and RL, we have presented here various general nonlinear IV techniques developed recently in machine learning to estimate Q-functions parameterized by neural networks in the OPE context. This has allowed us not only to recover previously proposed OPE methods such as model-based techniques but also to obtain novel techniques. We have assessed the performance of the resulting algorithms on a simple MDP model and two sets of benchmarking problems.

On the simple MDP problem, we find that the confounding effect can be very pronounced, and ignoring it as in Deterministic Bellman Residual Minimization (DBRM) (Saleh and Jiang, 2019) is problematic. In this example, an IV method like LSTD proves very useful. On more realistic examples from BSuite and DM Control, we have investigated scenarios where the evaluation policy is close to the behavioral one and where it is far from it. When doing OPE for a policy near to the one having generated data, we find that the confounding effect could be very pronounced and that techniques like DBRM are performing poorly. We also find that the best IV method - AGMM - displays performance on par with FQE and is only outperformed by distributional FQE. However, when evaluating a policy far from the

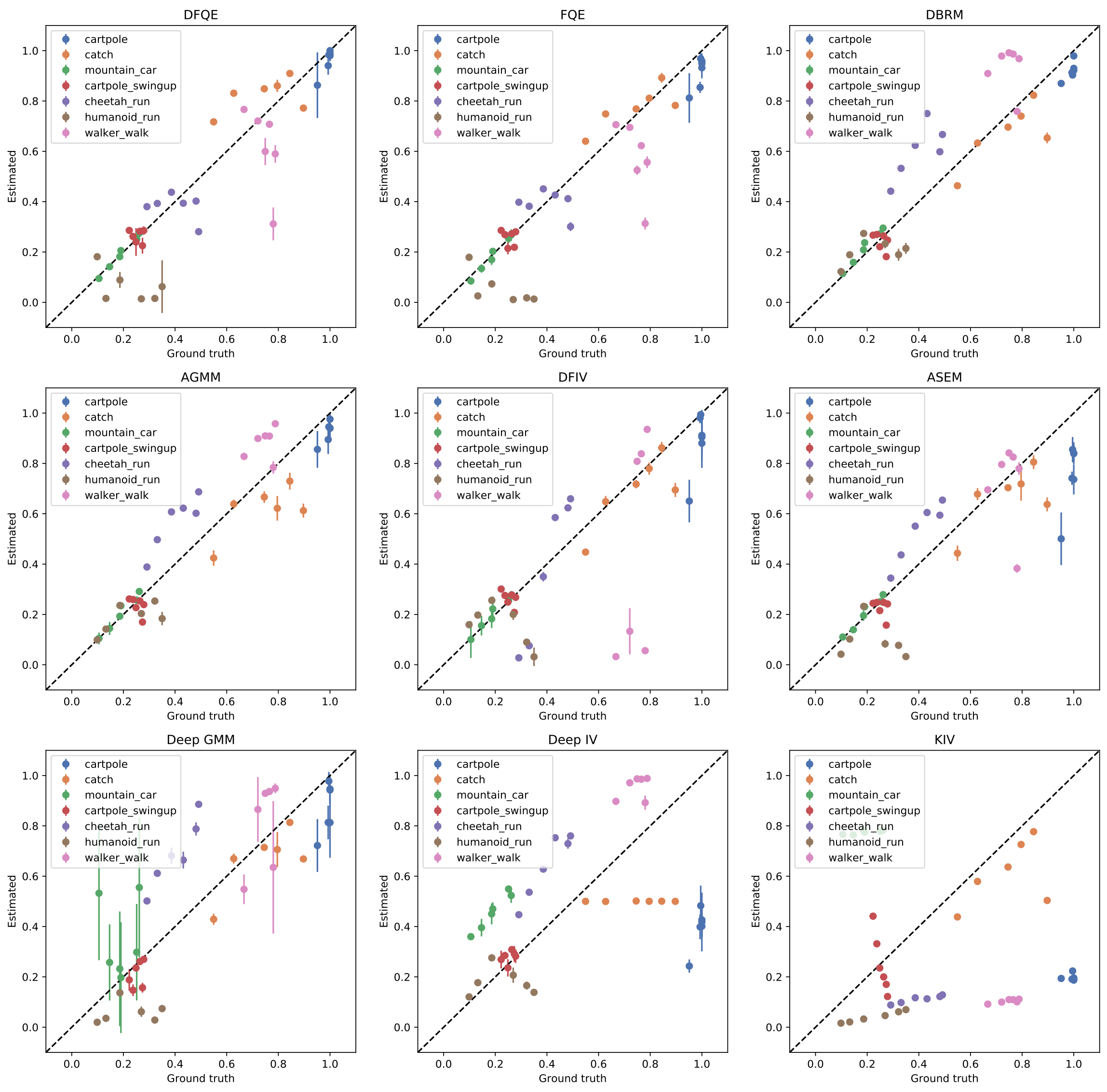


Figure 14: Estimated policy value vs groundtruth with the offline dataset

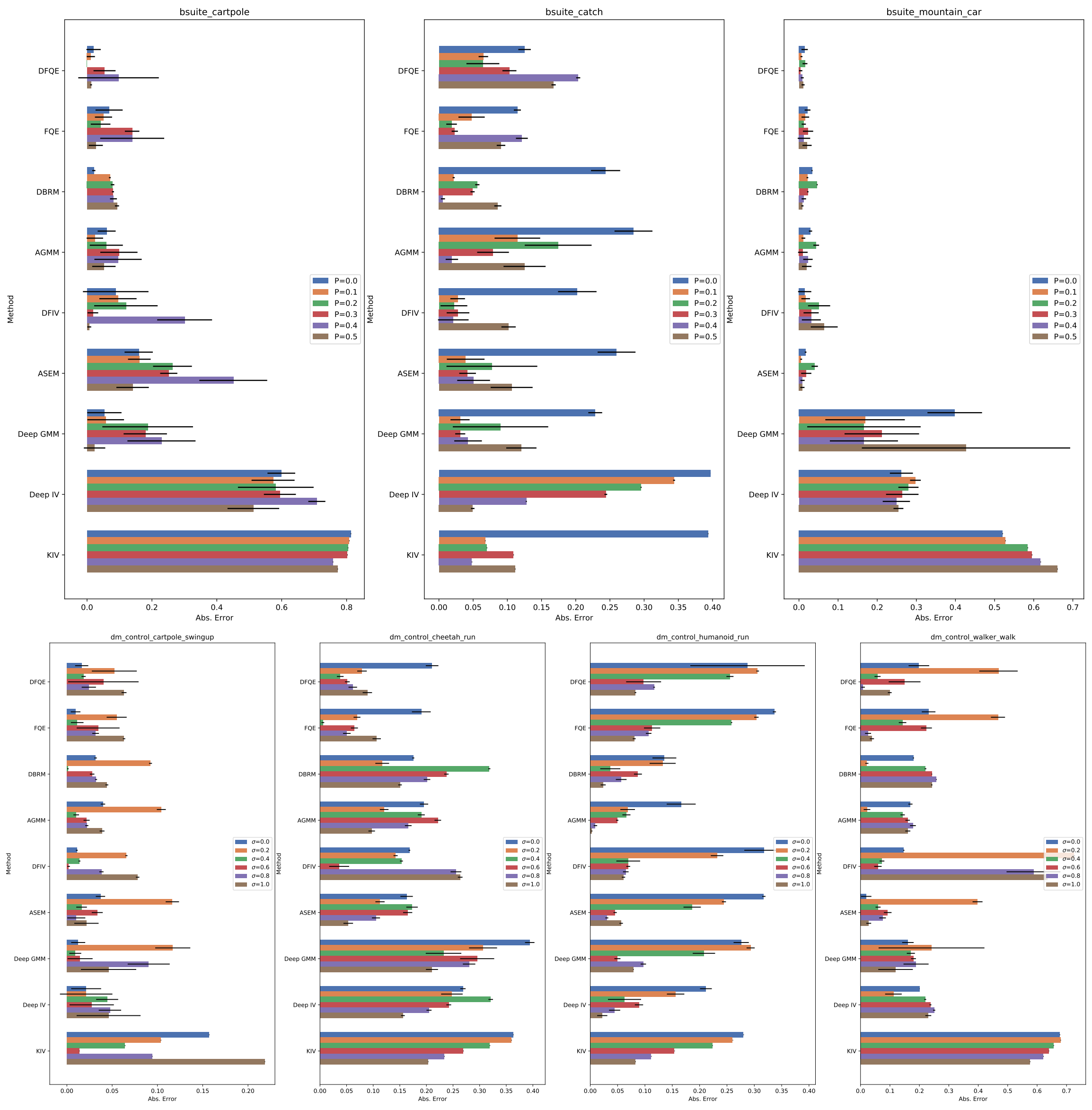


Figure 15: Absolute error of policy value estimation with the offline dataset

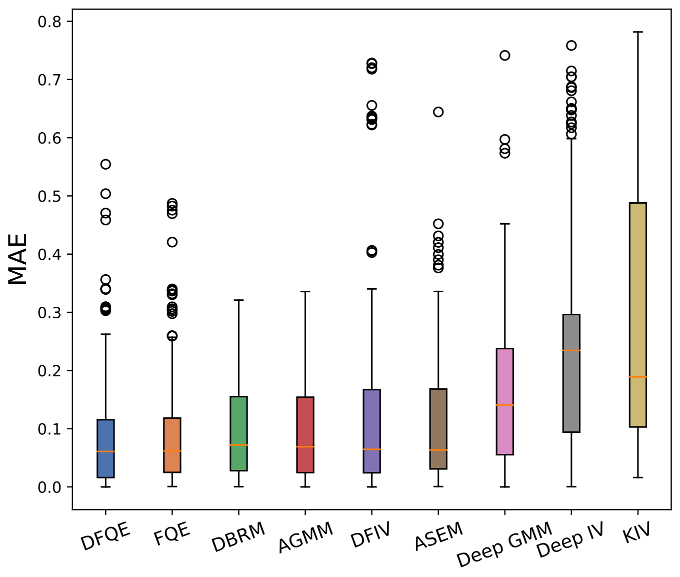


Figure 16: Distribution of the absolute error across all tasks with the offline dataset

one(s) having generated the observations, the combination of distribution shift and model mismatch has a non-negligible impact on performance. While we observe that AGMM also performed on par with FQE and DFQE, DBRM surprisingly also performs very well, and in general appears more stable (suffers from fewer outliers with poor performance) than FQE and DFQE.

Acknowledgements
This work was supported by the Gatsby Charitable Foundation.

References

Joshua D. Angrist. Lifetime earnings and the Vietnam era draft lottery: Evidence from social security administrative records. *The American Economic Review*, 80(3):313–336, 1990.

Joshua D. Angrist, Guido W. Imbens, and Donald B. Rubin. Identification of causal effects using instrumental variables. *Journal of the American Statistical Association*, 91(434): 444–455, 1996.

András Antos, Csaba Szepesvári, and Rémi Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1):89–129, 2008.

Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier, 1995.

Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, TB Dhruva, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. In *International Conference on Learning Representations*, 2018.

Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on Systems, Man, and Cybernetics*, (5):834–846, 1983.

Marc G Bellemare, Salvatore Candido, Pablo Samuel Castro, Jun Gong, Marlos C Machado, Subhodeep Moitra, Sameera S Ponda, and Ziyu Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, 2020.

Andrew Bennett, Nathan Kallus, and Tobias Schnabel. Deep generalized method of moments for instrumental variable analysis. In *Advances in Neural Information Processing Systems 32*, pages 3564–3574. 2019a.

Andrew Bennett, Nathan Kallus, and Tobias Schnabel. Deep generalized method of moments for instrumental variable analysis. In *Advances in Neural Information Processing Systems*, pages 3559–3569, 2019b.

Andrew Bennett, Nathan Kallus, Lihong Li, and Ali Mousavi. Off-policy evaluation in infinite-horizon reinforcement learning with latent confounders. In *International Conference on Artificial Intelligence and Statistics*, pages 1999–2007. PMLR, 2021.

Richard Blundell, Joel Horowitz, and Matthias Parey. Measuring the price responsiveness of gasoline demand: Economic shape restrictions and nonparametric demand estimation. *Quantitative Economics*, 3:29–51, 2012.

Steven J Bradtke and Andrew G Barto. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22(1-3):33–57, 1996.

Marine Carrasco, Jean-Pierre Florens, and Eric Renault. Linear inverse problems in structural econometrics estimation based on spectral decomposition and regularization. In *Handbook of Econometrics*, volume 6B, chapter 77. 2007.

Xiaohong Chen and Timothy M. Christensen. Optimal sup-norm rates and uniform inference on nonlinear functionals of nonparametric IV regression: Nonlinear functionals of nonparametric IV. *Quantitative Economics*, 9:39–84, 2018.

S. Darolles, Y. Fan, J. P. Florens, and E. Renault. Nonparametric instrumental regression. *Econometrica*, 79(5):1541–1565, 2011.

Nishanth Dikkala, Greg Lewis, Lester Mackey, and Vasilis Syrgkanis. Minimax estimation of conditional moment models. *arXiv preprint arXiv:2006.07201*, 2020.

Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. *CoRR*, abs/1103.4601, 2011.

Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, pages 201–208, 2005.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(Apr):503–556, 2005.

Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning*, pages 1447–1456, 2018.

Justin Fu, Mohammad Norouzi, Ofir Nachum, George Tucker, ziyu wang, Alexander Novikov, Mengjiao Yang, Michael R Zhang, Yutian Chen, Aviral Kumar, Cosmin Paduraru, Sergey Levine, and Thomas Paine. Benchmarks for deep off-policy evaluation. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=kWSeGEeHvF8`.

Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Tom Le Paine, Sergio Gomez Colmenarejo, Konrad Zolna, Rishabh Agarwal, Josh Merel, Daniel Mankowitz, Cosmin Paduraru, et al. Rl unplugged: A suite of benchmarks for offline reinforcement learning. *arXiv preprint arXiv:2006.13888*, 2020.

Lars Peter Hansen. Large sample properties of generalized method of moments estimators. *Econometrica*, 50(4):1029–1054, 1982.

Jason Hartford, Greg Lewis, Kevin Leyton-Brown, and Matt Taddy. Deep IV: A flexible approach for counterfactual prediction. In *Proceedings of the 34th International Conference on Machine Learning*, 2017a.

Jason Hartford, Greg Lewis, Kevin Leyton-Brown, and Matt Taddy. Deep IV: A flexible approach for counterfactual prediction. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1414–1423. JMLR. org, 2017b.

Matt Hoffman, Bobak Shahriari, John Aslanides, Gabriel Barth-Maron, Feryal Behbahani, Tamara Norman, Abbas Abdolmaleki, Albin Cassirer, Fan Yang, Kate Baumli, Sarah Henderson, Alex Novikov, Sergio Gómez Colmenarejo, Serkan Cabi, Caglar Gulcehre, Tom Le Paine, Andrew Cowie, Ziyu Wang, Bilal Piot, and Nando de Freitas. Acme: A research framework for distributed reinforcement learning. *arXiv preprint arXiv:2006.00979*, 2020.

Nan Jiang and Lihong 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.

Michail G Lagoudakis and Ronald Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4(Dec):1107–1149, 2003.

Hoang M Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. *arXiv preprint arXiv:1903.08738*, 2019.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Greg Lewis and Vasilis Syrgkanis. Adversarial generalized method of moments. *arXiv preprint arXiv:1803.07164*, 2018.

Jin Li, Ye Luo, and Xiaowei Zhang. Causal reinforcement learning: An instrumental variable approach. *arXiv preprint arXiv:2103.04021*, 2021.

Luofeng Liao, Zuyue Fu, Zhuoran Yang, Mladen Kolar, and Zhaoran Wang. Instrumental variable value iteration for causal offline reinforcement learning. *arXiv preprint arXiv:2102.09907*, 2021.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Conference on Neural Information Processing Systems*, 2018.

Liao Luofeng, Chen You-Lin, Yang Zhuoran, Dai Bo, Wang Zhaoran, and Kolar Mladen. Provably efficient neural estimation of structural equation model: An adversarial approach. *arXiv preprint arXiv:2007.01290*, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Andrew William Moore. *Efficient Memory-Based Learning for Robot Control.* PhD thesis, Cambridge University, 1990.

Ali Mousavi, Lihong Li, Qiang Liu, and Denny Zhou. Black-box off-policy estimation for infinite-horizon reinforcement learning. *arXiv preprint arXiv:2003.11126*, 2020.

Krikamol Muandet, Arash Mehrjou, Si Kai Lee, and Anant Raj. Dual IV: A single stage instrumental variable regression. *arXiv preprint arXiv:1910.12358*, 2019.

Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(5), 2008.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *arXiv preprint arXiv:1906.04733*, 2019.

Whitney K. Newey and James L. Powell. Instrumental variable estimation of nonparametric models. *Econometrica*, 71(5):1565–1578, 2003.

Ian Osband, Yotam Doron, Matteo Hessel, John Aslanides, Eren Sezener, Andre Saraiva, Katrina McKinney, Tor Lattimore, Csaba Szepesvari, Satinder Singh, et al. Behaviour suite for reinforcement learning. In *International Conference on Learning Representations*, 2019.

Tom Le Paine, Cosmin Paduraru, Andrea Michi, Caglar Gulcehre, Konrad Zolna, Alexander Novikov, Ziyu Wang, and Nando de Freitas. Hyperparameter selection for offline reinforcement learning. *arXiv preprint arXiv:2007.09055*, 2020.

Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

Doina Precup, Richard S Sutton, and Sanjoy Dasgupta. Off-policy temporal-difference learning with function approximation. In *International Conference on Machine Learning*, pages 417–424, 2001.

Ehsan Saleh and Nan Jiang. Deterministic Bellman residual minimization. In *Advances in Neural Information Processing Systems*. 2019.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Rahul Singh, Maneesh Sahani, and Arthur Gretton. Kernel instrumental variable regression. In *Advances in Neural Information Processing Systems 32*, pages 4593–4605. 2019a.

Rahul Singh, Maneesh Sahani, and Arthur Gretton. Kernel instrumental variable regression. In *Advances in Neural Information Processing Systems*, pages 4595–4607, 2019b.

James H. Stock and Francesco Trebbi. Retrospectives: Who invented instrumental variable regression? *Journal of Economic Perspectives*, 17(3):177–194, 2003.

Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pages 216–224. Elsevier, 1990.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction.* The MIT Press, 2018.

Yuval Tassa, Tom Erez, and Emanuel Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4906–4913. IEEE, 2012.

Yuval Tassa, Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, and Nicolas Heess. dm_control: Software and tasks for continuous control, 2020.

Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 2139–2148, 2016.

Masatoshi Uehara, Jiawei Huang, and Nan Jiang. Minimax weight and Q-function learning for off-policy evaluation. In *International Conference on Machine Learning*, pages 9659–9668. PMLR, 2020.

Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *Workshop at NeurIPS*, 2019a.

Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*, 2019b.

Paweł Wawrzyński. Real-time reinforcement learning by sequential actor–critics and experience replay. *Neural Networks*, 22(10):1484–1497, 2009.

P.G. Wright. *The Tariff on Animal and Vegetable Oils*. Investigations in International Commercial Policies. Macmillan Company, 1928.

Liyuan Xu, Yutian Chen, Siddarth Srinivasan, Nando de Freitas, Arnaud Doucet, and Arthur Gretton. Learning deep features in instrumental variable regression. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=sy4Kg_ZQmS7`.

Mengjiao Yang, Ofir Nachum, Bo Dai, Lihong Li, and Dale Schuurmans. Off-policy evaluation via the regularized Lagrangian. *arXiv preprint arXiv:2007.03438*, 2020.

Michael R Zhang, Thomas Paine, Ofir Nachum, Cosmin Paduraru, George Tucker, ziyu wang, and Mohammad Norouzi. Autoregressive dynamics models for offline policy evaluation and optimization. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=kmqjgSNXby`.

Appendix A. Variant of DFIV

With the linear assumption of the Q function, $Q_\theta(s,a) = \phi(s,a)^\top \theta$, we can follow the derivation of LSTD and write the structural function as

$$f(s,a,s',a') = \left(\phi(s,a) - \gamma\phi(s',a')\right)^\top \theta\,.$$

The original DFIV method exploits the decomposition of the structural function and regresses the expected deep features depending on (s',a') only, $\mathbb{E}\left[\phi(s',a')|s,a\right]$, in the first stage. We consider a variant where we regress the entire feature map directly $\mathbb{E}\left[\phi(s,a) - \gamma\phi(s',a')|s,a\right] = V\psi(s,a)$, and obtain the following stage 1 regression,

$$\hat{V} = \arg\min_V \mathcal{L}_1(V), \quad \mathcal{L}_1(V) = \mathbb{E}_{s,a,s',a'}\left[\|\phi(s,a) - \gamma\phi(s',a') - V\psi(s,a)\|^2\right] + \lambda_1\|V\|^2. \tag{42}$$

Stage 2 regression is then simplified as

$$\hat{\theta} = \arg\min_\theta \mathcal{L}_2(\theta), \quad \mathcal{L}_2(\theta) = \mathbb{E}_{r,s,a}\left[\|r - \theta^\top \hat{V}\psi(s,a)\|^2\right] + \lambda_2\|\theta\|^2. \tag{43}$$

While both versions make use of instrumental variables, we find in the experiments that the second variant is more stable during training. A potential explanation is that the stage 1 target, $\phi(s,a) - \gamma\phi(s',a')$, has a smaller variance than $\phi(s,a)$ when the state-action pair (s,a) changes smoothly in consecutive steps, and is easier to model by a least squares regression. Further investigation is yet to be done to verify this hypothesis.

Appendix B. Hyper-parameter selection

We choose the hyper-parameters of each IV algorithm using the recommended method in their original work. To be self-contained, we give a brief description for each method in this section. We split the offline dataset randomly into a training $\mathcal{D}_{\text{train}}$ and validation $\mathcal{D}_{\text{valid}}$ dataset with a ratio of 9:1, train each algorithm on $\mathcal{D}_{\text{train}}$ and choose the best hyper-parameters using $\mathcal{D}_{\text{valid}}$. The mini-batch size is 1024 unless explicitly specified.

Please note, however, that choosing the best hyper-parameters on the validation set does not guarantee a high accuracy in evaluating the target policy value, because the behavior distribution of states where the algorithm learns from is different from the evaluation distribution where the target policy induces and the learned OPE algorithm should be tested in. There is currently no commonly accepted way of choosing hyper-parameters in the offline reinforcement learning setting (see, e.g. Paine et al. (2020), for an attempt in choosing hyper-parameters for offline RL algorithms).

B.1 Kernel Instrumental Variable

We use random Fourier features to approximate the squared exponential kernel. The hyper-parameters of KIV are provided in Table 3. The validation metric for choosing the hyper-parameters is the stage-2 loss, eq. (24), without regularization.

Hyper-parameter	Value range
Stage-1 regularization	$\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$
Stage-2 regularization	$\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$
Number of random features	$\{128, 256, 512, 1024\}$

Table 3: Hyper-parameter of KIV. Values in braces are candidates to search over.

Hyper-parameter	Value range
Training steps	10^5
Stage-1 regularization	$\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$
Stage-2 regularization	$\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$
Value net regularization	$\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$
Instrument net regularization	$\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$
Value net learning rate	$\{10^{-5}, 3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}, 10^{-3}\}$
Instrument net learning rate	$\{10^{-5}, 3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}, 10^{-3}\}$
Instrument net hidden units (BSuite)	$\{(50, 50), (100, 100), (150, 150)\}$
Instrument net hidden units (DM Control)	$\{(512, 512, 256), (768, 768, 384), (1024, 1024, 512)\}$

Table 4: Hyper-parameter of DFIV. Values in braces are candidates to search over.

B.2 Deep Feature Instrumental Variables

The hyper-parameters of DFIV include the L_2 regularization strength in the regression of both stages, L_2 regularization strength for the value and instrumental network parameters, the learning rate of the Adam optimizer, and the network architecture of the instrument network. The mini-batch size is fixed at 2048. Details are shown in Table 4. The validation metric for choosing the hyper-parameters is the stage-2 loss in Equation (43), without regularization.

B.3 Deep IV

The hyper-parameters of Deep IV include the hidden layer sizes and the number of mixing components in the treatment network, learning rate of treatment and value networks, and the number of Monte Carlo samples in estimating the integral in Equation (10). The treatment network outputs both a mixture of Gaussian distribution to predict the next state s' and a Bernoulli distribution to predict if the current state is a terminating state. Details are shown in Table 5.

As suggested in Hartford et al. (2017b) we choose the hyper-parameters associated with training the treatment network according to the log-likelihood on the validation dataset, and those associated with training the value network according to the regression loss on the validation dataset.

Network	Hyper-parameter	Value range
Treatment net	Training steps	10^5
	Hidden units (BSuite)	$\{(32,32),(64,64),(128,128)\}$
	Hidden units (DM Control)	$\{(128,128,128),(256,256,256),(512,512,256),(768,768,384)\}$
	# mixing components	$\{1,3,10\}$
	Learning rate	$\{10^{-5}, 3\times10^{-5}, 10^{-4}, 3\times10^{-4}, 10^{-3}, 3\times10^{-3}\}$
Value net	Training steps	10^5
	# samples	$\{1,3,10\}$
	Learning rate	$\{10^{-5}, 3\times10^{-5}, 10^{-4}, 3\times10^{-4}, 10^{-3}, 3\times10^{-3}\}$

Table 5: Hyper-parameter of DFIV. Values in braces are candidates to search over.

B.4 Generalized Method of Moments

B.4.1 DeepGMM

The hyper-parameters of DeepGMM include the hidden layer sizes of the adversarial network, learning rate of the value network η_v and a learning rate multiplier for the adversarial network $\eta_a = \lambda\eta_v$, and the parameters β_1 and β_2 for the OAdam optimizer. Details are shown in Table 6.

We follow the hyper-parameter selection method in Bennett et al. (2019b). For every candidate of hyper-parameter setting and every checkpoint during the training i, we evaluate the value function Q_{θ_i} and adversarial function g_{τ_i} on a fixed set of validation data points. The metric for choosing the hyper-parameters is the objective in Equation (29) except that it is evaluated on the validation set (both Ψ_n and the expectation in R_g), the function g is in the finite set $\{g_{\tau_i}\}$, and $Q_{\tilde{\theta}}$ in the regularization R_g is averaged over all Q_{θ_i}.

$$\theta^* = \operatorname*{arg\,min}_{\theta\in\{\theta_i\}} \max_{\tau\in\{\tau_i\}} \Psi_n(Q_\theta, g_\tau) - \frac{1}{4}\mathbb{E}\left[g_\tau^2(s,a)(r - Q_{\tilde{\theta}}(s,a) + \gamma Q_{\tilde{\theta}}(s',a'))^2\right], \tag{44}$$

The hyper-parameters used to train θ^* is selected.

B.4.2 Adversarial GMM and SEM

The hyper-parameters of Adversarial GMM and SEM are similar to DeepGMM with additional hyper-parameters in the regularization terms. Details are shown in Table 7 and Table 8.

We choose the best hyper-parameter based on the early stopping method in the open-sourced implementation[4] of AGMM. Particularly, we keep a set of candidate parameters Q_{θ_i} and g_{τ_i} and the validation set as in the previous section, and then find parameter θ_i with minimum moment violation

$$\theta^* = \operatorname*{arg\,min}_{\theta\in\{\theta_i\}} \max_{\tau\in\{\tau_i\}} \Psi_n(Q_\theta, g_\tau)\,. \tag{45}$$

4. `https://github.com/microsoft/AdversarialGMM/tree/main/mliv/neuralnet`

Hyper-parameter	Value range
Training steps	20^5
Adversarial net hidden units (BSuite)	$\{(50, 50), (100, 100), (150, 150)\}$
Adversarial net hidden units (DM Control)	$\{(512, 512, 256), (768, 768, 384), (1024, 1024, 512)\}$
Value net learning rate	$\{10^{-5}, 3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}, 10^{-3}\}$
Instrument net learning rate multiplier	$\{1, 5, 10, 50\}$
OAdam (β_1, β_2)	$\{(0, 0.01), (0.5, 0.9)\}$

Table 6: Hyper-parameter of DeepGMM. Values in braces are candidates to search over.

Hyper-parameter	Value range
Training steps	20^5
Adversarial net hidden units (BSuite)	$\{(50, 50), (100, 100), (150, 150)\}$
Adversarial net hidden units (DM Control)	$\{(512, 512, 256), (768, 768, 384), (1024, 1024, 512)\}$
Value net learning rate	$\{10^{-5}, 3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}, 10^{-3}\}$
Instrument net learning rate multiplier	$\{1, 5, 10, 50\}$
OAdam (β_1, β_2)	$\{(0, 0.01), (0.5, 0.9)\}$
Value net parameter regularization a	$\{10^{-10}, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$
Adversarial net parameter regularization b	$\{10^{-10}, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$

Table 7: Hyper-parameter of Adversarial GMM. Values in braces are candidates to search over.

In order to avoid the max operator to be dominated by a test function with a large magnitude, we normalize all the g_{τ_i} functions on the validation set, that is,

$$\tilde{g}(s, a) = \frac{1}{|\mathcal{D}_{\mathrm{valid}}|}\mathbb{E}_{\mathcal{D}_{\mathrm{valid}}}\left[g(s, a)\right].$$

Hyper-parameter	Value range
Training steps	20^5
Adversarial net hidden units (BSuite)	$\{(50, 50), (100, 100), (150, 150)\}$
Adversarial net hidden units (DM Control)	$\{(512, 512, 256), (768, 768, 384), (1024, 1024, 512)\}$
Value net learning rate	$\{10^{-5}, 3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}, 10^{-3}\}$
Instrument net learning rate multiplier	$\{1, 5, 10, 50\}$
OAdam (β_1, β_2)	$\{(0, 0.01), (0.5, 0.9)\}$
Value net parameter regularization a	$\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$
Adversarial net parameter regularization b	$\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$
Value net parameter regularization α	$\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$

Table 8: Hyper-parameter of Adversarial SEM. Values in braces are candidates to search over.