# When is Offline Policy Selection Feasible for Reinforcement Learning?

**Vincent Liu**
University of Alberta
Edmonton, Canada
vliu1@ualberta.ca

**Prabhat Nagarajan**
University of Alberta
Edmonton, Canada
pmnagara@ualberta.ca

**Andrew Patterson**
University of Alberta
Edmonton, Canada
ap3@ualberta.ca

**Martha White**
University of Alberta
Edmonton, Canada
whitem@ualberta.ca

## Abstract

Offline reinforcement learning algorithms often require careful hyperparameter tuning. Consequently, before deployment, we need to select amongst a set of candidate policies. As yet, however, there is little understanding about the fundamental limits of this offline policy selection (OPS) problem. In this work we aim to provide clarity on when sample efficient OPS is possible, primarily by connecting OPS and off-policy policy evaluation (OPE). We first show a negative result, that in the worst case, OPS is just as hard as OPE, by proving a reduction of OPE to OPS. We show that a simple OPE method, importance sampling, achieves a nearly minimax sample complexity. As a result, no OPS method can be more sample efficient than OPE in the worst case. Then we propose a Bellman error estimation method for OPS, and theoretically analyze when this method is sample efficient. We highlight that using BE generally has more requirements, but if satisfied, has an easy method for selecting its own hyperparameters and may be more sample efficient than OPE. We conclude by showing the difficulty of OPS for an offline Atari benchmark, and an empirical study comparing OPE and BE estimation.

## 1 Introduction

Offline reinforcement learning (RL)—learning a policy from a dataset—is useful for many real-world applications, as learning from online interaction may be expensive or dangerous [Levine et al., 2020]. There have been significant advances in offline RL algorithms that have demonstrated the ability to learn purely from offline data. However, successfully use of these algorithms requires careful hyperparameter selection. Moreover, one may want to select amongst different offline algorithms as well, each with their own hyperparameters. Thus, we have a set of candidate policies to choose from where each candidate policy is generated from a specific algorithm-hyperparameter configuration.

The problem of finding the best-performing policy from a set of candidate policies is known as *policy selection*. Policy selection is a critical component in any practical offline RL system, since whether or not we can select effective hyperparameters for offline RL algorithms has a significant impact on the deployment performance [Wu et al., 2019, Gulcehre et al., 2020, Kumar et al., 2021]. With access to the environment or the simulator, we can find the best-performing policy by performing Monte Carlo rollouts for each candidate policy. However, in the offline setting, this Monte Carlo approach cannot be performed, and mechanisms to select policies with offline data are needed.

Preprint. Under review.

A common approach to policy selection in the offline setting, known as *offline policy selection (OPS)*, is to perform *off-policy policy evaluation* (OPE) to estimate the values of candidate policies. Typical OPE estimators include fitted Q evaluation (FQE) [Le et al., 2019] and importance sampling (IS) [Sutton and Barto, 2018]. Tang and Wiens [2021], Paine et al. [2020] provide experimental results on OPS using OPE. Doroudi et al. [2017], Yang et al. [2022] propose OPE estimators for OPS.

However, it is known that OPE is a hard problem that requires a large number of samples to evaluate any given policy in the worst case [Wang et al., 2021], so OPE can be unreliable for OPS. As a result, a natural follow-up question is: *do the hardness results from OPE also hold for OPS?* If the answer is yes, then we would need to consider additional assumptions to enable sample efficient OPS.

Moreover, OPS should be easier than OPE intuitively, since estimating each policy accurately might not be necessary for policy selection. There is mixed evidence that alternatives to OPE, which are based on variants of Bellman errors, might be effective. Tang and Wiens [2021] empirically show that selecting the candidate value function with the smallest TD errors perform poorly because TD errors provide overestimates; they conclude OPE is necessary. On the other hand, Kumar et al. [2021] propose using TD errors to select the number of training steps for an offline RL algorithm. Some works provide theoretical results. Zhang and Jiang [2021] propose a value-function selection algorithm called BVFT, which computes the (empirical) projected Bellman error for each pair of candidate value functions. Lee et al. [2022b] provide a method for selecting the best function class from a nested set of function classes for offline RL algorithm such as fitted Q iteration. However, these theoretically-sound methods rely on data coverage assumptions. It remains an open question about *when, or even if, alternative approaches can outperform OPE for OPS*.

In this work, we make contributions towards answering the question: *when can we perform offline policy selection efficiently—in other words with a polynomial sample complexity—for RL?* We first provide a clear connection between OPE and OPS, which has never been formally shown in the literature. We show that the sample complexity of the OPS problem is lower-bounded by the samples needed to perform OPE. As a consequence, OPS inherits the same hardness results as OPE. We further show that an OPS algorithm that simply chooses the policy with the highest IS estimate achieves a nearly minimax sample complexity, which is exponential in the horizon. This implies no OPS approach can be more sample efficient than IS in the worst case, and we must consider additional assumptions to circumvent exponential sample complexity.

Then we study alternative approaches to OPE, by using Bellman errors (BE) for OPS. Many OPE methods such as FQE introduce extra hyperparameters, which are not easy to tune. We propose a simple BE estimation algorithm with a simple way to select its own hyperparameters. We show that BE can provide improve sample efficiency than OPE methods like FQE, but under stricter requirements on data coverage and on the candidate set. We conclude with an empirical study, where we systematically compare different OPS methods with varying sample sizes and show the inherent difficulty of OPS in a bigger experiment in Atari. Notably, we show that all the OPS methods suffer relatively high regret compared to using the tuned hyperparameters (obtained through cheating by tuning for performance on the real environment), in some cases performing even worse than random hyperparameter selection.

## 2 Background

In RL, the agent-environment interaction can be formalized as a finite horizon finite Markov decision process (MDP) $M = (\mathcal{S}, \mathcal{A}, H, \nu, Q)$. $\mathcal{S}$ is a set of states with size $S = |\mathcal{S}|$, and $\mathcal{A}$ is a set of actions with size $A = |\mathcal{A}|$, $H \in \mathbb{Z}^+$ is the horizon, and $\nu \in \Delta(\mathcal{S})$ is the initial state distribution where $\Delta(\mathcal{S})$ is the set of probability distributions over $\mathcal{S}$. Without loss of generality, we assume that there is only one initial state $s_0$. The reward $R$ and next state $S'$ are sampled from $Q$, that is, $(R, S') \sim Q(\cdot|s, a)$. We assume the reward is bounded in $[0, r_{max}]$ almost surely so the total return of each episode is bounded in $[0, V_{max}]$ almost surely with $V_{max} = Hr_{max}$. The stochastic kernel $Q$ induces a transition probability $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, and a mean reward function $r(s, a)$ which gives the mean reward when taking action $a$ in state $s$.

A non-stationary policy is a sequence of memoryless policies $(\pi_0, \ldots, \pi_{H-1})$ where $\pi_h : \mathcal{S} \to \Delta(\mathcal{A})$. We assume that the set of states reachable at time step $h$, $\mathcal{S}_h \subset \mathcal{S}$, are disjoint, without loss of generality, because we can always define a new state space $\mathcal{S}' = \mathcal{S} \times [H]$, where $[H]$ denotes the set $[H] := \{0, 1, \ldots, H-1\}$. Then, it is sufficient to consider stationary policies $\pi : \mathcal{S} \to \Delta(\mathcal{A})$.

Given a policy $\pi$, for any $h \in [H]$ and $(s,a) \in \mathcal{S} \times \mathcal{A}$, we define the value function and the action-value function as $v_h^\pi(s) := \mathbb{E}^\pi[\sum_{t=h}^{H-1} r(S_t, A_t) | S_h = s]$ and $q_h^\pi(s,a) := \mathbb{E}^\pi[\sum_{t=h}^{H-1} r(S_t, A_t) | S_h = s, A_h = a]$, respectively. The expectation is with respect to $\mathbb{P}^\pi$, which is the probability measure on the random element $(S_0, A_0, R_0, \ldots, R_{H-1})$ induced by the policy $\pi$. The Bellman evaluation operator $\mathcal{T}^\pi$ is defined as

$$(\mathcal{T}^\pi q_h)(s,a) = r(s,a) + \sum_{s' \in \mathcal{S}} P(s,a,s') \sum_{a' \in \mathcal{A}} \pi(a'|s') q_h(s',a'),$$

with the Bellman optimality operator $(\mathcal{T} q_h)(s,a)$ obtained if a greedy policy in $q_h$ is used. We use $J(\pi)$ to denote the value of the policy $\pi$, that is, the expected return from the initial state $J(\pi) = v_0^\pi(s_0)$. The optimal value function is defined by $v_h^*(s) := \sup_\pi v_h^\pi(s)$. A policy $\pi$ is optimal if $J(\pi) = v_0^*(s_0)$.

In the offline setting, we are given a fixed set of transitions $D$ with samples drawn from a data distribution $\mu$. We consider the setting where the data is collected by a behavior policy $\pi_b$ since the data collection scheme is more practical [Xiao et al., 2022] and is used to collect data for benchmark datasets [Fu et al., 2020]. We use $d_h^\pi$ to denote the data distribution at the horizon $h$ by following the policy $\pi$, that is, $d_h^\pi(s,a) := \mathbb{P}^\pi(S_h = s, A_h = a)$, and $\mu_h(s,a) := \mathbb{P}^{\pi_b}(S_h = s, A_h = a)$.

## 3 Sample complexity of OPS

We consider the offline policy selection (OPS) problem and off-policy policy evaluation (OPE) problem. We follow a similar notation and formulation used in Xiao et al. [2022] to formally describe these problem settings. The OPS problem for a fixed number of episodes $n$ is given by the tuple $(\mathcal{S}, \mathcal{A}, H, \nu, n, \mathcal{I})$. $\mathcal{I}$ is a set of instances of the form $(M, d_b, \Pi)$ where $M \in \mathcal{M}(\mathcal{S}, \mathcal{A}, H, \nu)$ specifies an MDP with state space $\mathcal{S}$, action space $\mathcal{A}$, horizon $H$ and the initial state distributino $\nu$, $d_b$ is a distribution over a trajectory $(S_0, A_0, R_0, \ldots, R_{H-1})$ by running the behavior policy $\pi_b$ on $M$, and $\Pi$ is a finite set of candidate policies. We consider the setting where $\Pi$ has a small size and does not depend on $S$, $A$ or $H$.

An OPS algorithm takes as input a batch of data $D$, which contains $n$ trajectories, and a set of candidate policies $\Pi$, and outputs a policy $\pi \in \Pi$. We say an OPS algorithm $\mathcal{L}$ is $(\varepsilon, \delta)$-sound on instance $(M, d_b, \Pi)$ if

$$\Pr_{D \sim d_b}(J_M(\mathcal{L}(D, \Pi)) \geq J_M(\pi^\dagger) - \varepsilon) \geq 1 - \delta$$

where $\pi^\dagger$ is the best policy in $\Pi$. We say an OPS algorithm $\mathcal{L}$ is $(\varepsilon, \delta)$-sound on the problem $(\mathcal{S}, \mathcal{A}, H, \nu, n, \mathcal{I})$ if it is sound on any instance $(M, d_b, \Pi) \in \mathcal{I}$.

Given a pair $(\varepsilon, \delta)$, the sample complexity of OPS is the smallest integer $n$ such that there exists a behavior policy $\pi_b$ and an OPS algorithm $\mathcal{L}$ such that $\mathcal{L}$ is $(\varepsilon, \delta)$-sound on the OPS problem $(\mathcal{S}, \mathcal{A}, H, \nu, n, \mathcal{I}(\pi_b))$ where $\mathcal{I}(\pi_b)$ denotes the set of instances with data distribution $d_b$. That is, if the sample complexity is lower-bounded by a number $\mathrm{N}_{OPS}$, then, for any behavior policy $\pi_b$, there exists an MDP $M$ and a set of candidate policies $\Pi$ such that any $(\varepsilon, \delta)$-sound OPS algorithm on $(M, d_b, \Pi)$ requires at least $\mathrm{N}_{OPS}$ episodes.

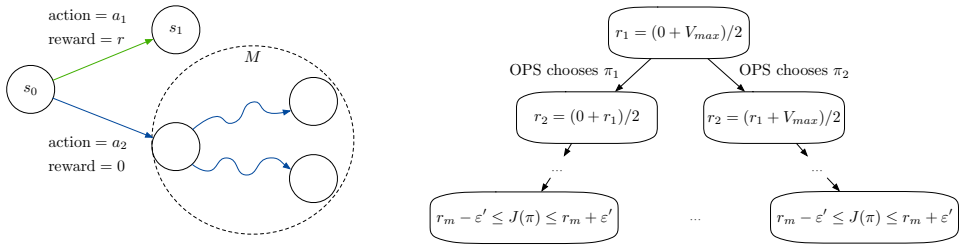
Similarly, the OPE problem for a fixed number of episodes $n$ is given by $(\mathcal{S}, \mathcal{A}, H, \nu, n, \mathcal{I})$. $\mathcal{I}$ is a set of instances of the form $(M, d_b, \pi)$ where $M$ and $d_b$ are defined as above, and $\pi$ is a target policy. An OPE algorithm takes as input a batch of data $D$ and a target policy $\pi$, and outputs an estimate of the policy value. We say an OPE algorithm $\mathcal{L}$ is $(\varepsilon, \delta)$-sound on instance $(M, d_b, \pi)$ if

$$\Pr_{D \sim d_b}(|\mathcal{L}(D, \pi) - J_M(\pi)| \leq \varepsilon) \geq 1 - \delta.$$

We say an OPE algorithm $\mathcal{L}$ is $(\varepsilon, \delta)$-sound on the problem $(\mathcal{S}, \mathcal{A}, H, \nu, n, \mathcal{I})$ if it is sound on any instance $(M, d_b, \pi) \in \mathcal{I}$. Note that $\varepsilon$ should be less than $V_{max}/2$ otherwise the bound is trivial.

### 3.1 OPE as subroutine for OPS

It is obvious that a sound OPE algorithm can be used for OPS, so the sample complexity of OPS is upper-bounded by the sample complexity of OPE up to a logarithmic factor. We state this formally in the theorem below.




(a) Constructing $M'$ and a candidate policy set (b) Searching for the true policy value

Figure 1: We can use an OPS algorithm for OPE. Given a MDP $M$ and a target policy $\pi$, we can construct a new MDP $M'$ and two candidate policies $\{\pi_1, \pi_2\}$ for OPS, as shown in (a). The MDP construction was first mentioned in Wang et al. [2021]. $\pi_1$ chooses $a_1$ in $s_0$ and is otherwise arbitrary, $\pi_2$ chooses $a_2$ and is otherwise identical to the target policy $\pi$. Figure (b) describes the search procedure to find the policy value. We can keep searching for the true policy value by setting $r$ for the OPS query, until the desired precision is reached.

**Theorem 1** (Upper bound on sample complexity of OPS)**.** Given an MDP $M$, a data distribution $d_b$, and a set of policies $\Pi$, suppose that, for any pair $(\varepsilon, \delta)$, there exists an $(\varepsilon, \delta)$-sound OPE algorithm $\mathcal{L}$ on any OPE instance $I \in \{(M, d_b, \pi) : \pi \in \Pi\}$ with a sample size at most $O(\mathrm{N}_{OPE}(S, A, H, \varepsilon, 1/\delta))$. Then there exists an $(\varepsilon, \delta)$-sound OPS algorithm for the OPS problem instance $(M, d_b, \Pi)$ which requires at most $O(\mathrm{N}_{OPE}(S, A, H, \varepsilon/2, |\Pi|/\delta))$ episodes.

In terms of the sample complexity, we have an extra $\sqrt{\log(|\Pi|)/n}$ term for OPS due to the union bound. For hyperparameter selection in practice, the size of the candidate set is often much smaller than $n$, so this extra term is negligible. However, if the set is too large, complexity regularization [Bartlett et al., 2002] may need to be considered. In this paper, we only consider a finite candidate set.

### 3.2 OPS is not easier than OPE

We have shown that OPS is sample efficient when OPE is sample efficient. However, it remains unclear whether OPS can be sample efficient when OPE is not. In the following theorem, we lower bound the sample complexity of OPS by the sample complexity of OPE. As a result, both OPS and OPE suffer from the same hardness result, and we cannot expect OPS to be sample efficient under conditions where OPE is not sample efficient.

**Theorem 2** (Lower bound on sample complexity of OPS)**.** Suppose for any data distribution $d_b$ and any pair $(\varepsilon, \delta)$ with $\varepsilon \in (0, V_{max}/2)$ and $\delta \in (0, 1)$, there exists an MDP $M$ and a policy $\pi$ such that any $(\varepsilon, \delta)$-sound OPE algorithm requires at least $\Omega(\mathrm{N}_{OPE}(S, A, H, \varepsilon, 1/\delta))$ episodes. Then there exists an MDP $M'$ with $S' = S + 2$, $H' = H + 1$, and a set of candidate policies such that for any pair $(\varepsilon, \delta)$ with $\varepsilon \in (0, V_{max}/3)$ and $\delta \in (0, 1/m)$ where $m := \lceil \log(V_{max}/\varepsilon) \rceil \geq 1$, any $(\varepsilon, \delta)$-sound OPS algorithm also requires at least $\Omega(\mathrm{N}_{OPE}(S, A, H, \varepsilon, 1/(m\delta)))$ episodes.

The proof sketch is to construct an OPE algorithm that queries OPS as a subroutine, as demonstrated in Figure 1. As a result, the sample complexity of OPS is lower bounded by the sample complexity of OPE. The proof can be found in Appendix A.

There exist several hardness results for OPE in tabular settings and with linear function approximation [Yin and Wang, 2021, Wang et al., 2021]. Theorem 2 implies that the same hardness results hold for OPS. We should not expect to have a sound OPS algorithm without additional assumptions. Theorem 2, however, does not imply that OPS and OPE are always equally hard. There are instances where OPS is easy but OPE is not. For example, when all policies in the candidate set all have the same value, any random policy selection is sound. However, OPE can still be difficult in such cases.

## 4 Minimax sample complexity for finite horizon finite MDPs

In this section, we prove a lower bound on the sample complexity of OPS for finite horizon finite MDPs, and show that an OPE method (nearly) matches the lower bound.

Using the construction from Xiao et al. [2022], we have an exponential lower bound on the sample complexity of OPE, which is presented in Theorem A.1 in the appendix for completeness. By the lower bound for OPE and Theorem 2, we have a lower bound for OPS.

**Corollary 1** (Exponential lower bound on the sample complexity of OPS)**.** For any positive integers $S, A, H$ with $S > 2H$ and a pair $(\varepsilon, \delta)$ with $0 < \varepsilon \leq \sqrt{1/8}$, $\delta \in (0, 1)$, any $(\varepsilon, \delta)$-sound OPS algorithm needs at least $\tilde{\Omega}(A^{H-1}/\varepsilon^2)$ episodes.

We show that importance sampling (IS) achieves the lower bound. Recall the IS estimator [Rubinstein, 1981] is given by $\hat{J}(\pi) = \frac{1}{n}\sum_{i=1}^{n}\prod_{h=0}^{H-1}\pi(A_h^{(i)}|S_h^{(i)})/\pi_b(A_h^{(i)}|S_h^{(i)})G_i$ where $G_i = \sum_{h=0}^{H-1} R_h^{(i)}$ and $n$ is the number of episodes in the dataset $D$.

**Corollary 2** (OPS using IS achieves nearly minimax sample complexity)**.** Suppose the data collection policy is uniformly random, that is, $\pi_b(a|s) = 1/A$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$, and $|G_i| \leq V_{max}$ almost surely. Then the selection algorithm $\mathcal{L}$ that selects the policy with the highest IS estimate is $(\varepsilon, \delta)$-sound with $O(A^H V_{max} \ln(|\Pi|/\delta)/\varepsilon^2)$ episodes.

These results suggest that IS achieves a nearly minimax optimal sample complexity for OPS up to a factor $A$ and logarithmic factors. There are other improved variants of IS, including per-decision IS and weighted IS [Precup et al., 2000, Sutton and Barto, 2018]. However, none of these variants can help reduce sample complexity in the worst case because the lower bound in Corollary 1 holds for any OPS algorithm. The result suggests that we need to consider additional assumptions on the environment, the data distribution, or the candidate set to perform sample efficient OPS.

Note that Wang et al. [2017] have shown that IS estimator achieved the minimax mean squared error for the OPE problem. Our result shows that IS also achieves a (nearly) minimax sample complexity for the OPS problem, which is different from their work.

## 5 Bellman error selection for OPS

We show that OPE is the most sample-efficient method for OPS in the worse case. In this section, we investigate whether and when BE can be useful for OPS.

Suppose we are given a set of candidate value functions $\mathcal{Q} := \{q_1, \ldots, q_K\}$ and let $\Pi = \{\pi_1, \ldots, \pi_K\}$ be the set of corresponding greedy policies, a common strategy is to select the action value function with the smallest BE [Farahmand and Szepesvári, 2011]. We define this as $(\varepsilon, \delta)$-sound *BE selection*, that is, the goal is to select a function $q \in \mathcal{Q}$ such that $\mathcal{E}(q) \leq \min_{i=1,\ldots,|\mathcal{Q}|}\mathcal{E}(q_i) + \varepsilon$ with probability at least $1 - \delta$, where we define $\mathcal{E}(q_i) := \frac{1}{H}\sum_h \|q_{i,h} - \mathcal{T}q_{i,h+1}\|_{2,\mu_h}^2$ to simplify the notation, and $\|q\|_{p,\mu} := (\sum_{(s,a)\in\mathcal{S}\times\mathcal{A}}\mu(s,a)|q(s,a)|^p)^{1/p}$. In this section, we show when BE selection can be used for OPS and propose a practical BE selection method for OPS.

### 5.1 When is BE selection useful for OPS?

In order to relate BE selection to OPS, we need data coverage for both the candidate policies $\Pi$ and an optimal policy. Data coverage is commonly measured by the concentration coefficient, first introduced in Munos [2007].[1] We present an error decomposition for OPS using BE selection. Given a candidate set $\mathcal{Q}$, we define the suboptimality of the candidate set as $\varepsilon_{sub} := \min_{q\in\mathcal{Q}}\mathcal{E}(q)$.

**Corollary 3** (Error decomposition for OPS using BE selection)**.** Assume there exists a constant $C$ such that, for any $\pi \in \Pi \cup \{\pi^*\}$, $\max_h \max_{s\in\mathcal{S}_h, a\in\mathcal{A}_h}\frac{d_h^\pi(s,a)}{\mu_h(s,a)} \leq C$, and we have an $(\varepsilon_{est}, \delta)$-sound BE selection algorithm, which outputs $q \in \mathcal{Q}$. Then, the OPS algorithm outputs the greedy policy with respect to $q$ is $(2H\sqrt{C(\varepsilon_{sub} + \varepsilon_{est})}, \delta)$-sound.

Note that this bound on OPS can be very bad, this is because the term $\varepsilon_{sub}$ does not go to zero even when we can collect more samples. Only $\varepsilon_{est}$ goes to zero as $n$ goes to infinity, which will be discussed in the next subsection.

To see how poor the guarantee can be, suppose we have two action values $q_1 = 100q^*$ and $q_2 = q^* +$ some random noise, then $q_1$ has a large Bellman error but $\pi_1$ is actually optimal. We would

[1] There are other measures of data coverage with function approximation [Xie et al., 2021]. However, we focus on the concentration coefficient to clearly compare assumptions between different methods.

choose $q_2$ since it has a lower Bellman error, even when we can collect an infinite number of samples. To obtain a meaningful guarantee, we need to include another value function, for example, $q_3 = q^*$ to make $\varepsilon_{sub} = 0$. As we collect more samples, we can estimate the Bellman error more accurately and eventually choose $q_3$.

At the same time, as we get more offline data, then we might actually change our candidate set. If we have more samples for training offline algorithms and generating the candidate set, it is more likely that there is one action value function in the candidate set that is close to optimal. That is, for a fixed candidate set, $\varepsilon_{sub}$ does not get smaller as we collect more samples for OPS, but if we use those offline samples to train the candidates policies, then $\varepsilon_{sub}$ might get smaller.

### 5.2 A sample-efficient and hyperparameter-free BE selection method

In deterministic environments, the BE can be easily estimated using TD errors. Given a state-action value $q$ with $v_q(s) := \max_a q(s,a)$ the corresponding state value, the BE estimate is $\text{TDE}(q) := \frac{1}{|D|}\sum_{(s,a,s',r)\in D}(q(s,a) - r - v_q(s'))^2$. Using standard concentration inequality, we can show that using TD error for BE selection requires a sample size $n = O(H^4 \log(|\Pi|H/\delta)/\varepsilon_{est}^2)$.

In stochastic environments, estimating BE typically involves an additional regression problem [Antos et al., 2008]. Antos et al. [2008] propose to estimate the BE by introducing an auxiliary function $g$:

$$\hat{\mathcal{E}}(q) = \text{TDE}(q) - \min_{g\in\mathcal{G}} \frac{1}{|D|} \sum_{(s,a,s',r)\in D} (g(s,a) - r - v_q(s'))^2 \tag{1}$$

The sample complexity of estimating the BE depends on the complexity of the function class $\mathcal{G}$, as shown in the theorem below.

**Theorem 3.** Suppose all $q \in \mathcal{Q}$ and $g \in \mathcal{G}$ take value in $[0, V_{max}]$. Let $q^\dagger$ be the action value function with the smallest estimated Bellman error $\hat{\mathcal{E}}(q)$. Then with probability at least $1-\delta$, for some constant $c_0$ (ignoring approximation and optimization error),

$$\mathcal{E}(q^\dagger) \leq \min_{i=1,\dots,|\mathcal{Q}|} \mathcal{E}(q_i) + c_0 V_{max}^2 \sqrt{\log(2|\Pi|H/\delta)/n} + c_0 V_{max} \mathcal{R}_n^\mu(\mathcal{G})$$

where $\mathcal{R}_n^\mu(\mathcal{G})$ is the Rademacher complexity of the function class $\mathcal{G}$.

Assume $\mathcal{G}$ has finite elements (for simplicity only), the sample complexity of finding a good action value function is $O(H^4 \log(|\mathcal{G}||\Pi|H/\delta)/\varepsilon_{est}^2)$. The proof can be found in Appendix A.

Compared to deterministic environments, we have an additional term that depends on the size of the function class. Fortunately, we can perform model selection to choose the function approximation $\mathcal{G}$. This is because we are running regression with fixed targets in Eq (1), and model selection for regression is well-studied. For example, we can use a holdout validation set to select the function class and hyperparameters. Therefore, we can choose a function class such that it has a low approximation error and a small complexity measure, which can potentially result in improved sample efficiency.

Moreover, we can also use the auxiliary function to predict $\mathcal{T}q - q$, instead of $\mathcal{T}q$ [Dai et al., 2018, Patterson et al., 2022]. The benefit is that the Bellman errors are more likely to be predictable. Under the conditions for Corollary 3, if there is an action-value function $q \in \mathcal{Q}$ with a small BE, the Bellman errors are nearly zero everywhere and any reasonable function class are able to learn it. We describe the practical BE selection with holdout validation in Algorithm 1.

As far we know, no prior work has studied this BE selection method for OPS. Some works consider selecting a value function that has the smallest BE or is the closest to the optimal value function. Farahmand and Szepesvári [2011] consider selecting a value function such that with high probability, the output value functions has the smallest BE. They propose to fit a regression model $\tilde{q}_i$ to predict $\mathcal{T}q_i$ and bound the BE by $\|q_i - \tilde{q}_i\|_{2,\mu}^2 + b_i$ where the first term can be viewed as the (empirical) projected Bellman error, and the second term $b_i$ is a high-probability upper bound on the excess risk of the regression model, which is assumed to be given.

Zhang and Jiang [2021] propose the (empirical) projected Bellman error with piecewise constant function classes. Their selection algorithm chooses the value function with the smallest BVFT loss, assuming $q^*$ is in the candidate set (approximately) and a stronger data assumption is satisfied. Interestingly, this condition on having $q^*$ is similar to our condition requiring small $\varepsilon_{sub}$, since

**Algorithm 1** BE selection with holdout validation

Input: Candidate set $\mathcal{Q}$, training data $D$, validation data $D_{val}$, a set of function classes $\mathcal{G}_1, \ldots, \mathcal{G}_M$
**for** $q \in \mathcal{Q}$ **do**
  Let $\delta(s, a, r, s') := r + v_q(s') - q(s, a)$
  **for** $m = 1, \ldots, M$ **do**
    Perform regression: $\hat{g}_m \leftarrow \min_{g \in \mathcal{G}_m} \frac{1}{|D|} \sum_D (g(s, a) - \delta(s, a, r, s'))^2$
    Compute validation error: $l(\hat{g}_m) \leftarrow \frac{1}{|D_{val}|} \sum_{D_{val}} (\hat{g}_m(s, a) - \delta(s, a, r, s'))^2$
  Find the best function class: $k \leftarrow \arg\min_{m=1,\ldots,M} l(\hat{g}_m)$
  Estimate the Bellman error for $q$: $\mathrm{BE}(q) \leftarrow \frac{1}{|D|} \sum_D 2\hat{g}_k(s, a)\delta(s, a, r, s') - \hat{g}_k(s, a)^2$
Output: $q^\dagger \leftarrow \arg\min_{q \in \mathcal{Q}} \mathrm{BE}(q)$

$q^*$ has exactly zero BE. The algorithms, though, are quite different from our work. Their method is computationally expensive since it scales with $O(|\Pi|^2)$ instead of $O(|\Pi|)$, making the method impractical when the candidate set is large.

Compare to Jiang and Li [2016], our method does not require the stronger data coverage assumption and the computation cost scales with $O(|\Pi|)$. Compared to Farahmand and Szepesvári [2011], our method does not assume the access to a high-probability upper bound on the excess risk.

## 6 Should we use BE selection or OPE?

We can leverage Theorem 1 to inherit the sample complexity result from OPE to OPS. There is a wealth of literature on OPE. However, estimators that use IS have exponentially high variance, which result in exponential sample complexity. There are more complex estimators such as the MAGIC estimator [Thomas and Brunskill, 2016] and the clipped IS estimator [Bottou et al., 2013], however, they are sensitive to their hyperparameters and it is hard to pick the hyperparameters offline. Model-based methods [Mannor et al., 2007, Liu et al., 2018] require learning a model, which makes it impractical in large state and action spaces. Some OPE methods are designed for specific MDPs such as Exogenous MDPs [Liu et al., 2023]. Finally, FQE and DICE methods have been shown to be effective for OPS [Paine et al., 2020, Yang et al., 2022]. We provide a short summary of the known OPE results in the appendix.

FQE and DICE methods require a standard data coverage assumption. That is, we need data coverage for all the candidate policies. For FQE, we also need a function class $\mathcal{F}$ which is closed under $\mathcal{T}^\pi$ for all $\pi \in \Pi$. Assume $\mathcal{F}$ has finite elements (for simplicity only), the result from Duan et al. [2021] can be extended to show that the sample complexity of using FQE for OPS is $O(H^4 \log(|\mathcal{F}||\Pi|H/\delta)/\varepsilon^2)$. There are corresponding results for the DICE methods [Nachum et al., 2019, Uehara et al., 2020]. If we use the same function approximation, the sample complexity of FQE has a similar order as BE. However, model selection for FQE is still an open problem.

In summary, OPE is a more general method for OPS. OPS using BE selection requires a stronger data coverage assumption since it needs coverage not only for the candidates policies, but also $\pi^*$. Moreover, the guarantee for OPS using BE selection can be poor due to the additional term $\varepsilon_{sub}$, which does not decrease as we collect more samples for OPS. We need at least one of the action value functions to be close to the optimal value. For OPE, we can perform OPS even if none of the candidate policies are close to optimal.

On the other hand, if we satisfy this stronger data coverage condition and have small $\varepsilon_{sub}$, then BE selection has several advantages. BE selection can be much more sample efficient in deterministic environments, or even in stochastic environments by choosing an appropriate function approximation. More importantly, for BE, we are not plagued by the issue of having hard-to-specify hyperparameters. This is critical for the offline setting, where we cannot test different hyperparameter choices in the environment.

Table 1: Normalized top-1 regret. The numbers are averaged over 5 datasets.

| Method | Breakout-early | Breakout-medium | Seaquest-early | Seaquest-final |
|---|---|---|---|---|
| FQE | **0.2874** $\pm$ 0.1847 | **0.3763** $\pm$ 0.3548 | 0.8819 $\pm$ 0.1436 | 0.5714 $\pm$ 0.3370 |
| BE | 0.4420 $\pm$ 0.1083 | 0.6917 $\pm$ 0.2311 | 0.7573 $\pm$ 0.2178 | **0.4851** $\pm$ 0.2721 |
| BVFT | 0.4071 $\pm$ 0.1368 | 0.6917 $\pm$ 0.2311 | 0.7573 $\pm$ 0.2178 | 0.5202 $\pm$ 0.1847 |
| random | 0.3748 $\pm$ 0.0766 | 0.4916 $\pm$ 0.0806 | **0.6604** $\pm$ 0.0752 | 0.5324 $\pm$ 0.0724 |

# 7 Experimental results

In this section, we provide experimental results to support our theoretical findings. We first show the difficulty of OPS for a standard offline Atari dataset. We then more systematically investigate the differences between several OPS methods, in a more controlled setting. All hyperparameters for OPS methods are selected using only the datasets, not by peaking at performance on the real environment; for more details, see Appendix C.

To evaluate the performance of OPS, we consider the normalized top-$k$ regret used in Zhang and Jiang [2021]. Top-$k$ regret is the gap between the best policy within the top-$k$ policies, and the best policy among all candidate policies. We then normalized the regret by the difference between the best and the worst policy, so the normalized regret is between 0 and 1. OPS corresponds to $k = 1$; for most results we use $k = 1$, but include some results for $k > 1$ in the appendix.

## 7.1 The hardness of OPS for benchmark datasets

We first conduct experiments on Atari datasets to show the hardness of OPS for offline RL. We use 1 million transitions on Breakout and Seaquest from the DQN replay dataset[2] and choose transitions from early learning (first 1 million steps), medium learning (between 19 to 20 millions steps) and final learning (between 49 to 50 millions steps). Note that the data coverage might be poor due to the absent of explicit exploration to cover all candidate policies. We use $50\%$ of the data to generate a set of candidate policy-value pairs by running CQL with different number of gradient steps and different value of regularization, as specified in Kumar et al. [2020]. We use the other $50\%$ data to perform OPS using FQE or BE. We also included BVFT [Zhang and Jiang, 2021], and a random selection baseline where we randomly choose a policy from the candidate set.

Table 1 shows the top-1 regret. The results for top-2 and top-3 regret can be found in Appendix B. We can see that these OPS methods suffer high regret and are often comparable to or worse than the random baseline. For Breakout, we found that BE tends to pick the candidate value function with a small number of gradient steps. This is likely due to that none of the candidate value functions are close to optimal (large $\varepsilon_{sub}$) and the value function with a small number of training steps has a small magnitude and hence a small BE. FQE is more reliable in general, but it still suffers high regret in Seaquest where we do not have good data coverage for the candidate policies.

## 7.2 Empirical comparison of OPE and BE selection

In the second set of experiments, we aim to answer the following questions: (1) How do OPE and BE selection perform for OPS under different data distribution (2) Can BE selection be more sample efficient than OPE for OPS? Again we focus on a particular OPE method, FQE.

We design three different datasets: (a) well-covered data is generated such that all candidate policies are well-covered, (b) diverse data includes more diverse trajectories collected by an $\varepsilon$-greedy expert policy, and (c) train data is collected by the behavior policy that is used to collect the training data to generate the candidate set. We conduct experiments on two standard RL environments: Acrobot and Cartpole. We also include the stochastic versions of these environments with sticky actions [Machado et al., 2018], which we call Stochastic Acrobot and Stochastic Cartpole. We generate a set of candidate policy-value pairs by running CQL with different hyperparameters on a batch of data collected by an optimal policy with random actions taken $40\%$ of the time. We then use either FQE, TDE, BE or BVFT for OPS. We do not include IS since the IS estimates are mostly zero.

[2] `https://research.google/resources/datasets/dqn-replay/`

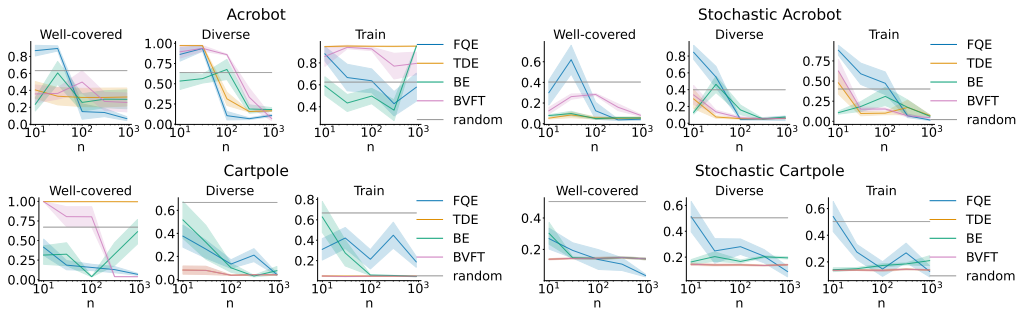



Figure 2: Normalized top-1 regret with varying sample size. The results are averaged over 10 runs with one standard error.

Figure 2 shows the results for top-1 regret with varying numbers of episodes. We first focus on the asymptotic performance ($n \approx 10^3$). FQE performs very well with a small regret on well-covered and diverse data. TDE and BE perform better with diverse data, compared to well-covered data. BVFT performs similarly as TDE, which is likely due to the fact that BVFT with small discretization is equivalent to TDE. For train data, these methods perform well in some environments even though there is not theoretical guarantee. We hypothesize the reason might be that we use a conservative algorithm to generate candidate policies so these candidate policies are covered to the train data.

Next we focus on the sample efficiency on the diverse data since FQE, BE and TDE should all work well with such data. TDE and BE perform better than FQE with a small sample size ($\leq 100$ episodes) in most cases. Theoretically, FQE and BE have the same magnitude of sample complexity, which depends on the complexity of the function class. In practice, we perform model selection for BE, so, for problems where a small function class is sufficient, we can use a small function class and obtain better sample efficiency. This might explain the observation that BE is more sample efficient than FQE in some of the environments. Additionally, we show that the model selection procedure is important for BE, by comparing to BE with a fixed hidden size, in Figure 4 in Appendix B.

## 8 Related work

In this section we provide a more comprehensive survey of prior work on model selection for RL. In the online setting, model selection has been studied extensively across contextual bandits [Foster et al., 2019] to RL [Lee et al., 2021]. In the online setting, the goal is to select model classes while balancing exploration and exploitation to achieve low regret, which is very different from the offline setting where no exploration is performed.

In the offline setting, besides using OPE and BE selection, other work on model selection in RL is in other settings: selecting models and selecting amongst OPE estimators. Hallak et al. [2013] consider model selection for model-based RL algorithms with batch data. They focus on selecting the most suitable model that generates the observed data, based on the maximum likelihood framework. Su et al. [2020] consider estimator selection for OPE when the estimators can be ordered with monotonically increasing biases and decreasing confidence intervals.

In offline RL pipelines, we often split the offline data into training data for generating multiple candidate policies and validation data for selecting the best candidate policy. Nie et al. [2022] highlight the utility of performing multiple random data splits for OPS. They do not study the hardness or sample complexity of this procedure.

To the best of our knowledge, there is no previous work on understanding the fundamental limits for OPS in RL. There is one related work in the batch contextual bandit setting, studying the selection of a linear model [Lee et al., 2022a]. They provide a hardness result suggesting it is impossible to achieve an oracle inequality that balances the approximation error, the complexity of the function class, and data coverage. Our work considers the more general problem, selecting a policy from a set of policies, in the RL setting.

## 9 Conclusion

In this paper, we made contributions towards understanding when OPS is feasible for RL. One of our main results—that the sample complexity of OPS is lower-bounded by the sample complexity of OPE—is perhaps expected. However, to our knowledge, this has never been formally shown. This result implies that without conditions to make OPE feasible, we cannot do policy selection efficiently.

We expect an active research topic will be to identify suitable conditions on the policies, environments and data, to make OPS feasible, or to design offline RL algorithms that have sound methods to select their own hyperparameters. In offline RL, we cannot select hyperparameters by testing in the real-world, and instead are limited to using the offline data. OPS is arguably one of the most critical steps towards bringing RL into the real-world, and there is much more to understand.

## References


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*, 2008.

Peter L Bartlett, Stéphane Boucheron, and Gábor Lugosi. Model selection and error estimation. *Machine Learning*, 2002.

Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 14 (11), 2013.

Bo Dai, Albert Shaw, Lihong Li, Lin Xiao, Niao He, Zhen Liu, Jianshu Chen, and Le Song. SBEED: Convergent reinforcement learning with nonlinear function approximation. In *International Conference on Machine Learning*, 2018.

Shayan Doroudi, Philip S Thomas, and Emma Brunskill. Importance sampling for fair policy selection. In *Conference on Uncertainty in Artificial Intelligence*, 2017.

Yaqi Duan, Chi Jin, and Zhiyuan Li. Risk bounds and Rademacher complexity in batch reinforcement learning. In *International Conference on Machine Learning*, 2021.

Amir-massoud Farahmand and Csaba Szepesvári. Model selection in reinforcement learning. *Machine Learning*, 2011.

Dylan J Foster, Akshay Krishnamurthy, and Haipeng Luo. Model selection for contextual bandits. *Advances in Neural Information Processing Systems*, 2019.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Thomas Paine, Sergio Gómez, Konrad Zolna, Rishabh Agarwal, Josh S Merel, Daniel J Mankowitz, Cosmin Paduraru, et al. RL unplugged: A suite of benchmarks for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2020.

Assaf Hallak, Dotan Di-Castro, and Shie Mannor. Model selection in markovian processes. In *International Conference on Knowledge Discovery and Data Mining*, 2013.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, 2016.

Akshay Krishnamurthy, Alekh Agarwal, and John Langford. PAC reinforcement learning with rich observations. *Advances in Neural Information Processing Systems*, 2016.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2020.

Aviral Kumar, Anikait Singh, Stephen Tian, Chelsea Finn, and Sergey Levine. A workflow for offline model-free robotic reinforcement learning. In *Conference on Robot Learning*, 2021.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*, 2019.

Jonathan Lee, Aldo Pacchiano, Vidya Muthukumar, Weihao Kong, and Emma Brunskill. Online model selection for reinforcement learning with function approximation. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Jonathan Lee, George Tucker, Ofir Nachum, and Bo Dai. Model selection in batch policy optimization. In *International Conference on Machine Learning*, 2022a.

Jonathan N Lee, George Tucker, Ofir Nachum, Bo Dai, and Emma Brunskill. Oracle inequalities for model selection in offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2022b.

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.

Vincent Liu, James R Wright, and Martha White. Exploiting action impact regularity and exogenous state variables for offline reinforcement learning. *Journal of Artificial Intelligence Research*, 77: 71–101, 2023.

Yao Liu, Omer Gottesman, Aniruddh Raghu, Matthieu Komorowski, Aldo A Faisal, Finale Doshi-Velez, and Emma Brunskill. Representation balancing MDPs for off-policy policy evaluation. *Advances in Neural Information Processing Systems*, 2018.

Marlos C Machado, Marc G Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018.

Shie Mannor, Duncan Simester, Peng Sun, and John N Tsitsiklis. Bias and variance approximation in value function estimates. *Management Science*, 2007.

Rémi Munos. Performance bounds in $l_p$-norm for approximate value iteration. *SIAM journal on control and optimization*, 46(2):541–561, 2007.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. DualDICE: Behavior-agnostic estimation of discounted stationary distribution corrections. *Advances in Neural Information Processing Systems*, 2019.

Allen Nie, Yannis Flet-Berliac, Deon R Jordan, William Steenbergen, and Emma Brunskill. Data-efficient pipeline for offline reinforcement learning with limited data. *Advances in Neural Information Processing Systems*, 2022.

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.

Andrew Patterson, Adam White, and Martha White. A generalized projected Bellman error for off-policy value estimation in reinforcement learning. *Journal of Machine Learning Research*, 2022.

Doina Precup, Richard S Sutton, and Satinder P Singh. Eligibility traces for off-policy policy evaluation. In *International Conference on Machine Learning*, 2000.

Reuven Y Rubinstein. *Simulation and the Monte Carlo method*. John Wiley & Sons, 1981.

Yi Su, Pavithra Srinath, and Akshay Krishnamurthy. Adaptive estimator selection for off-policy evaluation. In *International Conference on Machine Learning*, 2020.

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

Shengpu Tang and Jenna Wiens. Model selection for offline reinforcement learning: Practical considerations for healthcare settings. In *Machine Learning for Healthcare Conference*, 2021.

Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, 2016.

Masatoshi Uehara, Jiawei Huang, and Nan Jiang. Minimax weight and q-function learning for off-policy evaluation. In *International Conference on Machine Learning*, 2020.

Ruosong Wang, Dean P Foster, and Sham M Kakade. What are the statistical limits of offline RL with linear function approximation? In *International Conference on Learning Representations*, 2021.

Yu-Xiang Wang, Alekh Agarwal, and Miroslav Dudık. Optimal and adaptive off-policy evaluation in contextual bandits. In *International Conference on Machine Learning*, 2017.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Chenjun Xiao, Ilbin Lee, Bo Dai, Dale Schuurmans, and Csaba Szepesvari. The curse of passive data collection in batch reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Tengyang Xie and Nan Jiang. Q* approximation schemes for batch reinforcement learning: A theoretical comparison. In *Conference on Uncertainty in Artificial Intelligence*, 2020.

Tengyang Xie, Ching-An Cheng, Nan Jiang, Paul Mineiro, and Alekh Agarwal. Bellman-consistent pessimism for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2021.

Mengjiao Yang, Bo Dai, Ofir Nachum, George Tucker, and Dale Schuurmans. Offline policy selection under uncertainty. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Ming Yin and Yu-Xiang Wang. Optimal uniform OPE and model-based offline reinforcement learning in time-homogeneous, reward-free and task-agnostic settings. *Advances in Neural Information Processing Systems*, 2021.

Siyuan Zhang and Nan Jiang. Towards hyperparameter-free policy selection for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2021.

# A Technical details

## A.1 Proof of Theorem 1

**Theorem 1** (Upper bound on sample complexity of OPS)**.** Given an MDP $M$, a data distribution $d_b$, and a set of policies $\Pi$, suppose that, for any pair $(\varepsilon, \delta)$, there exists an $(\varepsilon, \delta)$-sound OPE algorithm $\mathcal{L}$ on any OPE instance $I \in \{(M, d_b, \pi) : \pi \in \Pi\}$ with a sample size at most $O(\mathrm{N}_{OPE}(S, A, H, \varepsilon, 1/\delta))$. Then there exists an $(\varepsilon, \delta)$-sound OPS algorithm for the OPS problem instance $(M, d_b, \Pi)$ which requires at most $O(\mathrm{N}_{OPE}(S, A, H, \varepsilon/2, |\Pi|/\delta))$ episodes.

*Proof.* The OPS algorithm $\mathcal{L}(D, \Pi)$ for a given $(\varepsilon, \delta)$ works as follows: we query an $(\varepsilon', \delta')$-sound OPE algorithm for each policy in $\Pi$ and select the policy with the highest estimated value. That is, $\mathcal{L}(D, \Pi)$ outputs the policy $\bar{\pi} := \arg\max_{\pi \in \Pi} \hat{J}(\pi)$, where $\hat{J}(\pi)$ is the value estimate for policy $\pi$ by the $(\varepsilon', \delta')$-sound OPE algorithm with data $D$.

By definition of an $(\varepsilon', \delta')$-sound OPE algorithm we have

$$\Pr_{D \sim d_b}(|\hat{J}(\pi) - J(\pi)| \leq \varepsilon') \geq 1 - \delta', \forall \pi \in \Pi.$$

Applying the union bound, we have

$$\Pr_{D \sim d_b}(\forall \pi \in \Pi, |\hat{J}(\pi) - J(\pi)| \leq \varepsilon') \geq 1 - \delta'|\Pi|.$$

Let $\pi^\dagger$ denote the best policy in the candidate set $\Pi$, that is, $\pi^\dagger := \arg\max_{\pi \in \Pi} J(\pi)$. With probability $1 - \delta'|\Pi|$, we have

$$J(\bar{\pi}) \geq \hat{J}(\bar{\pi}) - \epsilon' \geq \hat{J}(\pi^\dagger) - \varepsilon' \geq J(\pi^\dagger) - 2\varepsilon'.$$

The second inequality follows from the definition of $\bar{\pi}$.

Finally, by setting $\delta' = \delta/|\Pi|$ and $\varepsilon' = \varepsilon/2$, we get

$$\Pr_{D \sim d_b}(J(\bar{\pi}) \geq J(\pi^\dagger) - \varepsilon) \geq 1 - \delta.$$

That is, $\mathcal{L}$ is an $(\varepsilon, \delta)$-sound OPS algorithm. This requires at most $O(\mathrm{N}_{OPE}(S, A, H, \varepsilon/2, |\Pi|/\delta))$ samples. □

## A.2 Proof of Theorem 2

**Theorem 2** (Lower bound on sample complexity of OPS)**.** Suppose for any data distribution $d_b$ and any pair $(\varepsilon, \delta)$ with $\varepsilon \in (0, V_{max}/2)$ and $\delta \in (0, 1)$, there exists an MDP $M$ and a policy $\pi$ such that any $(\varepsilon, \delta)$-sound OPE algorithm requires at least $\Omega(\mathrm{N}_{OPE}(S, A, H, \varepsilon, 1/\delta))$ episodes. Then there exists an MDP $M'$ with $S' = S + 2$, $H' = H + 1$, and a set of candidate policies such that for any pair $(\varepsilon, \delta)$ with $\varepsilon \in (0, V_{max}/3)$ and $\delta \in (0, 1/m)$ where $m := \lceil \log(V_{max}/\varepsilon) \rceil \geq 1$, any $(\varepsilon, \delta)$-sound OPS algorithm also requires at least $\Omega(\mathrm{N}_{OPE}(S, A, H, \varepsilon, 1/(m\delta)))$ episodes.

*Proof.* Our goal is to construct an $(\varepsilon, \delta)$-sound OPE algorithm with $\delta \in (0, 1)$ and $\varepsilon \in [0, V_{max}/2]$. To evaluate any policy $\pi$ in $M$ with dataset $D$ sampled from $d_b$, we first construct a new MDP $M_r$ with two additional states: an initial state $s_0$ and a terminal state $s_1$. Taking $a_1$ at $s_0$ transitions to $s_1$ with reward $r$. Taking $a_2$ at $s_0$ transitions to the initial state in the *original* MDP $M$. See Figure 1 in the main paper for visualization.

Let $\Pi = \{\pi_1, \pi_2\}$ be the candidate set in $M_r$ where $\pi_1(s_0) = a_1$ and $\pi_2(s_0) = a_2$ and $\pi_2(a|s) = \pi(a|s)$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$. Since $\pi_1$ always transitions to $s_1$, it never transitions to states in MDP $M$. Therefore, $\pi_1$ can be arbitrary for all $(s, a) \in \mathcal{S} \times \mathcal{A}$. We can add any number of transitions $(s_0, a_1, r, s)$ and $(s_0, a_2, 0, s)$ in $D$ to construct the dataset $D_r$ with distribution $d_{b,r}$ arbitrarily.

Suppose we have an $(\varepsilon', \delta')$-sound OPS algorithm, where we set $\varepsilon' = 2\varepsilon/3$, $\delta' = \delta/m$ and $m := \lceil \log(V_{max}/\varepsilon') \rceil$. Note that if this assumption does not hold, then it directly implies that the sample complexity of OPS is larger than $\Omega(\mathrm{N}_{OPE}(S, A, H, \varepsilon, 1/\delta))$. Our strategy will be to iteratively set the reward $r$ of MDP $M_r$ and run our sound OPS algorithm on $\Pi$ and using bisection search to estimate a precise interval for $J(\pi)$.

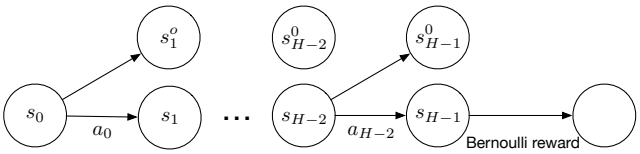



Figure 3: Lower bound construction.

The process is as follows. By construction, our OPS algorithm will output either $\pi_1$, which has value $J_{M_r}(\pi_1) = r$, or output $\pi_2$, which has value $J_{M_r}(\pi_2) = J_M(\pi)$. That is, it has the same value as $\pi$ in the original MDP. Let us consider the following two cases. Let $\pi^\dagger$ be the best policy in $\Pi$ for MDP $M_r$.

**Case 1: the OPS algorithm selects $\pi_1$.** We know, by definition of a sound OPS algorithm, that

$$\begin{aligned} & \Pr(J_{M_r}(\pi_1) \geq J_{M_r}(\pi^\dagger) - \varepsilon') \geq 1 - \delta' \\ \implies & \Pr(r \geq \max(r, J_{M_r}(\pi_2)) - \varepsilon') \geq 1 - \delta' \\ \implies & \Pr(J_{M_r}(\pi_2) \leq r + \varepsilon') \geq 1 - \delta'. \end{aligned}$$

**Case 2: the OPS algorithm selects $\pi_2$.**

$$\begin{aligned} & \Pr(J_{M_r}(\pi_2) \geq J_{M_r}(\pi^\dagger) - \varepsilon') \geq 1 - \delta' \\ \implies & \Pr(J_{M_r}(\pi_2) \geq \max(r, J_{M_r}(\pi_2)) - \varepsilon') \geq 1 - \delta' \\ \implies & \Pr(J_{M_r}(\pi_2) \geq r - \varepsilon') \geq 1 - \delta'. \end{aligned}$$

Given this information, we describe the iterative process by which we produce the estimate $\hat{J}(\pi)$. We first set $U = V_{max}$, $L = 0$ and $r = \frac{U+L}{2}$ and run the sound OPS algorithm with input $D_r$ of sample size $n_r$ and the candidate set $\Pi$. Then if the selected policy is $\pi_1$, then we conclude the desired event $J(\pi) \leq r + \varepsilon'$ occurs with probability at least $1 - \delta'$, and set $U$ equal to $r$. If the selected policy is $\pi_2$, then we know the desired event $J(\pi) \geq r - \varepsilon'$ occurs with probability at least $1 - \delta'$, and set $L$ equal to $r$. We can continue the bisection search until the accuracy is less than $\varepsilon'$, that is, $U - L \leq \varepsilon'$, and the output value estimate is $\hat{J}(\pi) = \frac{U+L}{2}$.

If all desired events at each call occur, then we conclude that $L - \varepsilon' \leq J(\pi) \leq U + \varepsilon'$ and thus $|J(\pi) - \hat{J}(\pi)| \leq \varepsilon$. The total number of OPS calls is at most $m$. Setting $\delta' = \delta/m$ and applying a union bound, we can conclude that with probability at least $1 - \delta$, $|J(\pi) - \hat{J}(\pi)| \leq \varepsilon$.

Finally, since any $(\varepsilon, \delta)$-sound OPE algorithm on the instance $(M, d_b, \pi)$ needs at least $\Omega(N_{OPE}(S, A, H, \varepsilon, 1/\delta))$ samples, the $(\varepsilon', \delta')$-sound OPS algorithm needs at least $\Omega(N_{OPE}(S, A, H, \varepsilon, 1/\delta))$, or equivalently $\Omega(N_{OPE}(S, A, H, 3\varepsilon'/2, 1/(m\delta')))$ samples for at least one of the instances $(M_r, d_{b,r}, \Pi)$. □

### A.3 Proof of Corollary 2

**Theorem A.1** (Exponential lower bound on the sample complexity of OPE)**.** For any positive integers $S, A, H$ with $S > 2H$ and a pair $(\varepsilon, \delta)$ with $0 < \varepsilon \leq \sqrt{1/8}$, $\delta \in (0, 1)$, any $(\varepsilon, \delta)$-sound OPE algorithm needs at least $\Omega(A^H \ln(1/\delta)/\varepsilon^2)$ episodes.

*Proof.* We provide a proof which uses the construction from Xiao et al. [2022]. They provide the result for the offline RL problem with Gaussian rewards. Here we provide the result for OPE problem with Bernoulli rewards since we assume rewards are bounded to match Theorem 2.

We can construct an MDP with $S$ states, $A$ actions and $2H$ states. See Figure 3 for the construction. Given any behavior policy $\pi_b$, let $a_h = \arg\min_a \pi_b(a|s_h)$ be the action that leads to the next state $s_{h+1}$ from state $s_h$, and all other actions lead to an absorbing state $s_h^o$. Once we reach an absorbing state, the agent gets zero reward for all actions for the remainder of the episode. The only nonzero reward is in the last state $s_{H-1}$. Consider a target policy that chooses $a_h$ for state $s_h$ for all $h = 0, \ldots, H-1$, and two MDPs where the only difference between them is the reward distribution

in $s_{H-1}$. MDP 1 has Bernoulli distribution with mean $1/2$ and MDP 2 has Bernoulli distribution with mean $1/2 - 2\varepsilon$. Let $\mathbb{P}_1$ denote the probability measure with respect to MDP 1 and $\mathbb{P}_2$ denote the probability measure with respect to MDP 2.

Let $\hat{r}$ denote the OPE estimate by an algorithm $\mathcal{L}$. Define an event $E = \{\hat{r} < \frac{1}{2} - \varepsilon\}$. Then $\mathcal{L}$ is not $(\varepsilon, \delta)$-sound if $(\mathbb{P}_1(E) + \mathbb{P}_2(E^c))/2 \geq \delta$. This is because $\mathcal{L}$ is not $(\varepsilon, \delta)$-sound if either $\mathbb{P}_1(\hat{r} < \frac{1}{2} - \varepsilon) \geq \delta$ or $\mathbb{P}_2(\hat{r} > \frac{1}{2} - \varepsilon) \geq \delta$.

Using the Bretagnolle–Huber inequality (see Theorem 14.2 of Lattimore and Szepesvári [2020]), we know

$$\frac{\mathbb{P}_1(E) + \mathbb{P}_2(E^c)}{2} \geq \frac{1}{4} \exp\left(-D_{KL}(\mathbb{P}_1, \mathbb{P}_2)\right).$$

By the chain rule for KL-divergence and the fact that $\mathbb{P}_1$ and $\mathbb{P}_2$ only differ in the reward for $(s_{H-1}, a_{H-1})$, we have

$$\begin{aligned}
D_{KL}(\mathbb{P}_1, \mathbb{P}_2) &= \mathbb{E}_1 \left[ \sum_{i=1}^{n} \mathbb{I}\{S_{H-1}^{(i)} = s_{H-1}, A_{H-1}^{(i)} = a_{H-1}\} \left( \frac{1}{2} \log\left(\frac{1/2}{1/2 - \varepsilon}\right) + \frac{1}{2} \log\left(\frac{1/2}{1/2 + \varepsilon}\right) \right) \right] \\
&= \sum_{i=1}^{n} \mathbb{P}_1(S_{H-1}^{(i)} = s_{H-1}, A_{H-1}^{(i)} = a_{H-1}) \left( -\frac{1}{2} \log\left(1 - 4\varepsilon^2\right) \right) \\
&\leq \frac{n 8 \varepsilon^2}{A^H}
\end{aligned}$$

The last inequality follows from $-\log(1 - 4\varepsilon^2) \leq 8\varepsilon^2$ if $4\varepsilon^2 \leq 1/2$ [Krishnamurthy et al., 2016] and $\mathbb{P}_1(S_{H-1}^{(i)} = s_{H-1}, A_{H-1}^{(i)} = a_{H-1}) < 1/A^H$ from the construction of the MDPs.

Finally,

$$\frac{\mathbb{P}_1(E) + \mathbb{P}_2(E^c)}{2} \geq \frac{1}{4} \exp\left(-\frac{n 8 \varepsilon^2}{A^H}\right)$$

which is larger than $\delta$ if $n \leq A^H \ln(1/4\delta)/8\varepsilon^2$. As a result, we need at least $\Omega(A^H \ln(1/\delta)/\varepsilon^2)$ episodes. □

**Corollary 2** (OPS using IS achieves nearly minimax sample complexity)**.** Suppose the data collection policy is uniformly random, that is, $\pi_b(a|s) = 1/A$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$, and $|G_i| \leq V_{max}$ almost surely. Then the selection algorithm $\mathcal{L}$ that selects the policy with the highest IS estimate is $(\varepsilon, \delta)$-sound with $O(A^H V_{max} \ln(|\Pi|/\delta)/\varepsilon^2)$ episodes.

*Proof.* Since the policy is uniform random, we know $|W_i G_i| < A^H V_{max}$ almost surely. Moreover, the importance sampling estimator is unbiased, that is, $\mathbb{E}[W_i G_i] = J(\pi)$. Using the Bernstein's inequality, we can show that the IS estimator satisfies

$$\Pr\left( \left| \hat{J}(\pi_k) - J(\pi_k) \right| \leq \frac{2 A^H V_{max} \ln(2/\delta)}{3n} + \sqrt{\frac{2 \mathrm{Var}(W_i G_i) \ln(2/\delta)}{n}} \right) \geq 1 - \delta$$

for one candidate policy $\pi_k$. Using the union bound over all candidate policies, we have

$$\Pr\left( \left| \hat{J}(\pi_k) - J(\pi_k) \right| \leq \frac{2 A^H V_{max} \ln(2|\Pi|/\delta)}{3n} + \sqrt{\frac{2 \mathbb{V}(W_i G_i) \ln(2|\Pi|/\delta)}{n}}, \forall k \right) \geq 1 - \delta.$$

That is,

$$\Pr\left( J(\mathcal{L}(D, \Pi)) \geq J(\pi^\dagger) - \frac{4 A^H V_{max} \ln(2|\Pi|/\delta)}{3n} + \sqrt{\frac{8 \mathbb{V}(W_i G_i) \ln(2|\Pi|/\delta)}{n}} \right) \geq 1 - \delta.$$

For the variance term,

$$\mathbb{V}(W_i G_i) = \mathbb{E}[W_i^2 G_i^2] - \mathbb{E}[W_i G_i]^2 \leq \mathbb{E}[W_i^2 G_i^2] \leq \sqrt{\mathbb{E}[W_i^2] \mathbb{E}[G_i^2]} \leq A^H V_{max}.$$

The second inequality follows from the Cauchy-Schwarz inequality. Therefore, if $n > 32 A^H V_{max} \ln(2|\Pi|/\delta)/\varepsilon^2$, $\mathcal{L}$ is $(\varepsilon, \delta)$-sound. □

### A.4 Proof of Corollary 3

**Lemma A.2** (Theorem 2 of Xie and Jiang [2020] for discounting setting and Lemma 3.2 of Duan et al. [2021] for finite horizon setting)**.** Assume there exists a constant $C$ such that, for any $\pi \in \Pi \cup \{\pi^*\}$, $\max_h \max_{s\in\mathcal{S}_h, a\in\mathcal{A}_h} \frac{d_h^\pi(s,a)}{\mu_h(s,a)} \leq C$. For any $q \in \mathcal{Q}$, let $\pi$ denotes the greedy policy with respect to $q = (q_0, \ldots, q_{H-1})$, then

$$J(\pi) \geq J(\pi^*) - 2H\sqrt{C}\sqrt{\frac{1}{H}\sum_{h=0}^{H-1} \|q_h - \mathcal{T}q_{h+1}\|_{2,\mu_h}}\,.$$

**Corollary 3** (Error decomposition for OPS using BE selection)**.** Assume there exists a constant $C$ such that, for any $\pi \in \Pi \cup \{\pi^*\}$, $\max_h \max_{s\in\mathcal{S}_h, a\in\mathcal{A}_h} \frac{d_h^\pi(s,a)}{\mu_h(s,a)} \leq C$, and we have an $(\varepsilon_{est}, \delta)$-sound BE selection algorithm, which outputs $q \in \mathcal{Q}$. Then, the OPS algorithm outputs the greedy policy with respect to $q$ is $(2H\sqrt{C(\varepsilon_{sub} + \varepsilon_{est})}, \delta)$-sound.

*Proof.* Since $\mathcal{E}(q_k) = \frac{1}{H}\sum_h \|q_{k,h} - \mathcal{T}q_{k,h+1}\|_\mu^2 \leq \varepsilon_{sub} + \varepsilon_{est}$ with probability at least $1-\delta$, Lemma A.2 implies

$$J(\pi_k) \geq J(\pi^*) - 2H\sqrt{C(\varepsilon_{sub} + \varepsilon_{est})} \geq J(\pi^\dagger) - 2H\sqrt{C(\varepsilon_{sub} + \varepsilon_{est})}$$

where $\pi^\dagger$ is the best performing policy in $\Pi$. By the definition, the OPS algorithm is $(2H\sqrt{C(\varepsilon_{sub} + \varepsilon_{est})}, \delta)$-sound for this instance. □

### A.5 Proof of Theorem 3

**Lemma A.3** (Bellman error selection in deterministic environments)**.** Consider a deterministic environment, let $k$ be the index with the lowest empirical TD error error, that is,

$$k = \operatorname*{arg\,min}_{i=1,\ldots,|\mathcal{Q}|} \frac{1}{H}\sum_{h=0}^{H-1} \frac{1}{n} \sum_{(s,a,r,s')\in D_h} \left| q_{i,h}(s,a) - r - \max_{a'} q_{i,h+1}(s',a') \right|^2,$$

then the following holds with probability at least $1-\delta$ and some constant $c_0$

$$\mathcal{E}(q_k) \leq \min_{i=1,\ldots,|\mathcal{Q}|} \mathcal{E}(q_i) + c_0 V_{max}^2 \sqrt{\frac{\log(|\Pi|H/\delta)}{n}}.$$

*Proof.* Using Hoeffding's inequality and the union bound, with probability at least $1-\delta$, we have

$$\left| \frac{1}{n} \sum_{(s,a,r,s')\in D_h} |q_h(s,a) - r - \max_{a'} q_{h+1}(s',a')|^2 - \|q_h - \mathcal{T}q_{h+1}\|_{2,\mu_h}^2 \right| \leq c_0 V_{max}^2 \sqrt{\frac{\log(2H|\Pi|/\delta)}{n}}$$

for all candidate value function $q = (q_1, \ldots, q_{H-1}) \in \mathcal{Q}$ and $h \in [H]$ for some constant $c_0 > 0$. Then

$$\mathcal{E}(q_k) \leq \min_{i=1,\ldots,|\mathcal{Q}|} \mathcal{E}(q_i) + c_0 V_{max}^2 \sqrt{\frac{\log(|\Pi|H/\delta)}{n}}.$$

for some constant $c_0 > 0$. □

**Assumption A.4** (Approximation error)**.** For any $h \in [H]$ and any candidate function $f \in \mathcal{Q}$ (here we use $f$ instead of $g$ to avoid confusion between $q$ and $g$), we assume the approximation error is bounded by $\varepsilon_{apx}$, that is,

$$\inf_{g\in\mathcal{G}} \|g - \mathcal{T}f\|_{2,\mu_h}^2 \leq \varepsilon_{apx}.$$

**Definition A.5** (Rademacher complexity)**.** Given a function class $\mathcal{F}$, let $X = \{x_1, \ldots, x_n\}$ denotes $n$ fixed data points at horizon $h$ following the distribution $\mu_h$, the empirical Rademacher complexity is defined as

$$R_X(\mathcal{F}) = \mathbb{E}\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i f(x_i) \mid X\right]$$

where the expectation is with respect to the Rademacher random variables $\sigma_i$. The Rademacher complexity is defined as $R_n^{\mu_h}(\mathcal{F}) = \mathbb{E}[R_X(\mathcal{F})]$ where the expectation is with respect to the $n$ data points. Finally, to simply the notation, we define $R_n^{\mu}(\mathcal{F}) = \max_{h\in[H]} R_n^{\mu_h}(\mathcal{F})$ as the maximum Rademacher complexity over all horizons.

**Definition A.6.** Given an action value function $f$, define the state value function $v_f(s) = \arg\max_{a\in\mathcal{A}} f(s,a)$, and four loss functions:

$$\begin{aligned}
L_{D_h}(g,f) &:= \frac{1}{n}\sum_{(s,a,r,s')\in D_h} (g(s,a) - r - v_f(s'))^2 \\
L_{D}(g,f) &:= \frac{1}{H}\sum_{h}\frac{1}{n}\sum_{(s,a,r,s')\in D_h} (g(s,a) - r - v_f(s'))^2 \\
L_{\mu_h}(g,f) &:= \mathbb{E}_{(s,a,r,s')\sim\mu_h}[(g(s,a) - r - v_f(s'))^2] \\
L_{\mu}(g,f) &:= \frac{1}{H}\sum_{h}\mathbb{E}_{(s,a,r,s')\sim\mu}[(g(s,a) - r - v_f(s'))^2].
\end{aligned}$$

**Theorem 3** (Bellman error selection in stochastic environments)**.** Suppose all $f \in \mathcal{Q}$ and $g \in \mathcal{G}$ take value in $[0, V_{max}]$. Let $k = \arg\min_i L_D(f_i, f_i) - L_D(\hat{g}_i, f_i)$ where $\hat{g}_i = \arg\min_{g\in\mathcal{G}} L_D(g, f_i)$. Then the following holds with probability at least $1-\delta$,

$$\mathcal{E}(q_k) \le \min_{i=1,\ldots,|\mathcal{Q}|} \mathcal{E}(q_i) + c_0\varepsilon_{apx} + c_0 V_{max}\mathcal{R}_n^{\mu}(\mathcal{G}) + c_0 V_{max}^2\sqrt{\frac{2\log(2|\Pi|H/\delta)}{n}}$$

for some constant $c_0 > 0$.

*Proof.* Fix a horizon $h$ and a target function $f$. By concentration with Rademacher complexity (e.g., Lemma G.1 of Duan et al. [2021]), with probability at least $1-\delta/2$, we know, for any $g \in \mathcal{G}$,

$$|L_{D_h}(g,f) - L_{\mu_h}(g,f)| \le 2\mathcal{R}_n^{\mu_h}(L_{D_h}\circ\mathcal{G}) + V_{max}^2\sqrt{\frac{\log(2/\delta)}{2n}}.$$

We note that the loss function $L_{D_h}$ is $(2V_{max})$-Lipschitz in its first argument, that is,

$$\begin{aligned}
|L_{D_h}(g,f)(s,a,r,s') - L_{D_h}(g',f)(s,a,r,s')| &= |(g(s,a) - r - V_f(s'))^2 - (g'(s,a) - r - V_f(s'))^2| \\
&= |(g(s,a) - g'(s,a))(g(s,a) + g'(s,a) - 2r - 2V_f(s'))| \\
&\le |g(s,a) - g'(s,a)||g(s,a) + g'(s,a) - 2r - 2V_f(s')| \\
&\le |g(s,a) - g'(s,a)|2V_{max}.
\end{aligned}$$

Therefore,

$$|L_{D_h}(g,f) - L_{\mu_h}(g,f)| \le 4V_{max}\mathcal{R}_n^{\mu_h}(\mathcal{G}) + V_{max}^2\sqrt{\frac{\log(2/\delta)}{2n}}.$$

By Hoeffding's inequality, we also know, with probability at least $1-\delta/2$,

$$|L_{D_h}(f,f) - L_{\mu_h}(f,f)| \le V_{max}^2\sqrt{\frac{\log(2/\delta)}{2n}}.$$

By the union bound over all $h \in [H]$, we have, with probability at least $1-\delta$,

$$\begin{aligned}
&|L_D(g,f) - L_\mu(g,f)| \le 4V_{max}\mathcal{R}_n^{\mu}(\mathcal{G}) + V_{max}^2\sqrt{\frac{\log(2H/\delta)}{2n}}, \forall g \in \mathcal{G}, \text{ and} \\
&|L_D(f,f) - L_\mu(f,f)| \le V_{max}^2\sqrt{\frac{\log(2H/\delta)}{2n}}.
\end{aligned}$$

Recall that we define $\hat{g}$ as the empirical minimizer, that is, $\hat{g} = \arg\min_{g \in \mathcal{G}} L_D(g, f)$, and let $g^\dagger$ be the population minimizer, that is, $g^\dagger = \arg\min_{g \in \mathcal{G}} L_\mu(g, f)$. It follows that with probability at least $1 - \delta$,

$$
\begin{aligned}
&|L_D(f,f) - L_D(\hat{g},f) - (L_\mu(f,f) - L_\mu(\mathcal{T}f,f))| \\
&\leq |L_D(f,f) - L_\mu(f,f)| + |L_D(\hat{g},f) - L_\mu(\mathcal{T}f,f)| \\
&= |L_D(f,f) - L_\mu(f,f)| + |\underbrace{L_D(\hat{g},f) - L_D(g^\dagger,f)}_{\leq 0}| + |L_D(g^\dagger,f) - L_\mu(g^\dagger,f)| + |L_\mu(g^\dagger,f) - L_\mu(\mathcal{T}f,f)| \\
&= |L_D(f,f) - L_\mu(f,f)| + |L_D(g^\dagger,f) - L_\mu(g^\dagger,f)| + \underbrace{|L_\mu(g^\dagger,f) - L_\mu(\mathcal{T}f,f)|}_{=\|g^\dagger - \mathcal{T}f\|_{2,\mu}^2} \\
&\leq \varepsilon_{apx} + 4V_{max}\mathcal{R}_n^\mu(\mathcal{G}) + V_{max}^2\sqrt{\frac{2\log(2H/\delta)}{n}}.
\end{aligned}
$$

Note that $L_\mu(f,f) - L_\mu(\mathcal{T}f,f) = \mathcal{E}(f)$. Now apply the union bound for all $f$ in the candidate set by setting $\delta = \delta/|\Pi|$, then the following holds for all $f$

$$
|L_D(f,f) - L_D(\hat{g},f) - \mathcal{E}(f)| \leq \varepsilon_{apx} + 4V_{max}\mathcal{R}_n^\mu(\mathcal{G}) + V_{max}^2\sqrt{\frac{2\log(2|\Pi|H/\delta)}{n}}.
$$

Let $k = \arg\min_i L_D(f_i,f_i) - L_D(\hat{g}_i,f_i)$, the following holds with probability at least $1-\delta$,

$$
\mathcal{E}(f) \leq \min_{i=1,\ldots,|\mathcal{Q}|} \mathcal{E}(f_i) + c_0\varepsilon_{apx} + c_0 V_{max}\mathcal{R}_n^\mu(\mathcal{G}) + c_0 V_{max}^2\sqrt{\frac{2\log(2|\Pi|H/\delta)}{n}}
$$

for some constant $c_0 > 0$.

Assume $\mathcal{F}$ has finite elements, and $\varepsilon_{apx} = 0$, then we need a sample size of $n = O(H^4 \log(|\mathcal{F}||\Pi|H/\delta)/\varepsilon_{est}^2)$. □

### A.6 Sample complexity of OPS using FQE

Consider a function class $\mathcal{F} = \mathcal{F}_1 \times \cdots \times \mathcal{F}_{H-1}$ such that $\mathcal{F}$ is closed under $\mathcal{T}^\pi$ for all $\pi \in \Pi$ and $|\mathcal{F}_h|$ is finite for all $h$. Assume $r_{max} = 1$ for simplicity. Given a policy $\pi \in \Pi$, we can show that

$$
\begin{aligned}
\|q_0^\pi - q_0\|_{1,d_0^\pi} &= \sum_a \pi(a|s_0)|q_0^\pi(s_0,a) - q_0(s_0,a)| \\
&= \sum_a \pi(a|s_0)|(\mathcal{T}^\pi q_1^\pi)(s_0,a) - (\mathcal{T}^\pi q_1)(s_0,a) + (\mathcal{T}^\pi q_1)(s_0,a) - q_0(s_0,a)| \\
&\leq \sum_{a,s',a'} \pi(a|s_0)p(s'|s,a)\pi(a'|s')|q_1^\pi(s,a) - q_1(s,a)| + \sum_a \pi(a|s_0)|(\mathcal{T}^\pi q_1)(s_0,a) - q_0(s_0,a)| \\
&= \|q_1^\pi - q_1\|_{1,d_1^\pi} + \|\mathcal{T}^\pi q_1 - q_0\|_{1,d_0^\pi}.
\end{aligned}
$$

Apply the same inequality recursively, we have

$$
\begin{aligned}
\|q_0^\pi - q_0\|_{1,d_0^\pi} &\leq \sum_{h=0}^{H-1} \|\mathcal{T}^\pi q_{h+1} - q_h\|_{1,d_h^\pi} \\
&\leq H\sqrt{\frac{1}{H}\sum_{h=0}^{H-1} \|\mathcal{T}^\pi q_{h+1} - q_h\|_{2,d_h^\pi}^2} \\
&\leq H\sqrt{C\frac{1}{H}\sum_{h=0}^{H-1} \|\mathcal{T}^\pi q_{h+1} - q_h\|_{2,\mu_h}^2}.
\end{aligned}
$$

The second inequality follows from the Cauchy-Schwarz inequality, and the last inequality follows from data coverage assumption.

| Method | Assumptions | Sample Complexity |
|---|---|---|
| IS | $\pi_b(a\|s) > 0$ if $\pi(a\|s) > 0$ for some $\pi \in \Pi$ | $O(A^H H \ln(\|\Pi\|/\delta)/\varepsilon^2)$ (Corollary 2) |
| FQE | (1) Data coverage for $\Pi$<br>(2) $\mathcal{F}$ is closed under $\mathcal{T}^\pi$ for all $\pi \in \Pi$ | Complexity of $\mathcal{F}$ |
| DICE | (1) Data coverage for $\Pi$<br>(2) $\mathcal{F}$ realizes $d^\pi/\mu$ for all $\pi \in \Pi$ | Complexity of $\mathcal{F}$<br>[Nachum et al., 2019, Uehara et al., 2020] |
| TDE | (1) Data coverage for $\Pi \cup \{\pi^*\}$<br>(2) Small $\varepsilon_{sub}$<br>(3) Deterministic environments | $O(H^4 \log(\|\Pi\|H/\delta)/\varepsilon_{est}^2)$<br>(see Section 5.2) |
| IBE | (1) Data coverage for $\Pi \cup \{\pi^*\}$<br>(2) Small $\varepsilon_{sub}$<br>(3) $\mathcal{G}$ realizes $(\mathcal{T}q)$ or $(\mathcal{T}q - q)$ for $q \in \mathcal{Q}$ | $O(H^4 \log(\|\mathcal{F}\|\|\Pi\|H/\delta)/\varepsilon_{est}^2)$<br>(see Section 5.2) |
| BVFT | (1) Stronger data coverage for the underlying MDP<br>(2) $q^* \in \mathcal{Q}$ approximately (which implies small $\varepsilon_{sub}$) | Number of partitions<br>(see Zhang and Jiang [2021]) |

Table 2: A summary of OPS methods. The first three methods are OPE methods and the last three methods uses action value functions. The assumptions for function approximation can be relaxed to hold approximately.

Theorem 5.3 and Proposition 6.1 and of Duan et al. [2021] imply that for some constant $c_0$, with probability at least $1-\delta$,

$$\left|J(\pi) - \sum_{a\in\mathcal{A}} \pi(a|s_0) q_0(s_0,a)\right| \le c_0 H \sqrt{CH\left(\sum_h \frac{H \log|\mathcal{F}_h|}{n} + H^2 \frac{\log(H/\delta)}{n}\right)}$$
$$\le c_0 H \sqrt{CH^2 \frac{\log(|\mathcal{F}|H/\delta)}{n}}.$$

Apply the union bound, we know the following holds for all $\pi \in \Pi$ with probability at least $1-\delta$

$$\left|J(\pi) - \sum_{a\in\mathcal{A}} \pi(a|s_0) q_0(s_0,a)\right| \le c_0 H \sqrt{CH^2 \frac{\log(|\mathcal{F}||\Pi|H/\delta)}{n}}.$$

To get an accuracy of $\varepsilon/2$, we need $n \ge c_1 H^4 C \log(|\mathcal{F}||\Pi|H/\delta)/\varepsilon^2$ for some constant $c_1$.

# B Additional experiments

**Top-$k$ regret for Atari experiments.** We provide the top-$k$ regret for Atari experiments in Table 3 and 4.

Table 3: Normalized top-2 regret. The numbers are averaged over 5 replay datasets.

| Method | Breakout-early | Breakout-medium | Seaquest-early | Seaquest-final |
|---|---|---|---|---|
| FQE | $0.2863 \pm 0.1016$ | **0.2174** $\pm 0.2462$ | $0.7612 \pm 0.2465$ | $0.2496 \pm 0.3431$ |
| BE | $0.3598 \pm 0.1364$ | $0.5564 \pm 0.2016$ | $0.6064 \pm 0.2323$ | $0.1879 \pm 0.2047$ |
| BVFT | $0.3197 \pm 0.1443$ | $0.5564 \pm 0.2016$ | $0.6064 \pm 0.2323$ | **0.1693** $\pm 0.1558$ |
| random | **0.2559** $\pm 0.0617$ | $0.3162 \pm 0.0505$ | **0.5079** $\pm 0.0644$ | $0.3835 \pm 0.0803$ |

Table 4: Normalized top-3 regret. The numbers are averaged over 5 replay datasets.

| Method | Breakout-early | Breakout-medium | Seaquest-early | Seaquest-final |
|---|---|---|---|---|
| FQE | $0.2755 \pm 0.1055$ | **0.0584** $\pm 0.0785$ | $0.6999 \pm 0.2793$ | $0.2309 \pm 0.3224$ |
| BE | $0.3219 \pm 0.1433$ | $0.4763 \pm 0.1424$ | $0.5351 \pm 0.2106$ | **0.0554** $\pm 0.0955$ |
| BVFT | $0.2479 \pm 0.2081$ | $0.4763 \pm 0.1424$ | $0.5351 \pm 0.2106$ | $0.1193 \pm 0.1406$ |
| random | **0.2023** $\pm 0.0596$ | $0.2323 \pm 0.0258$ | **0.4183** $\pm 0.0612$ | $0.3025 \pm 0.0733$ |

**Model selection for BE selection.** In the classic RL experiments, for BE, we use a two layer neural network model as the function approximation, and perform model selection to find the hidden size from the set $\{32, 64, 128, 256\}$. In Figure 4, we show that the model selection procedure is important for BE selection. BE-fixed is the baseline where we fix the hidden size to 256. We can see that BE is better than or equal to BE-fixed and their performance convergence as sample size gets larger.

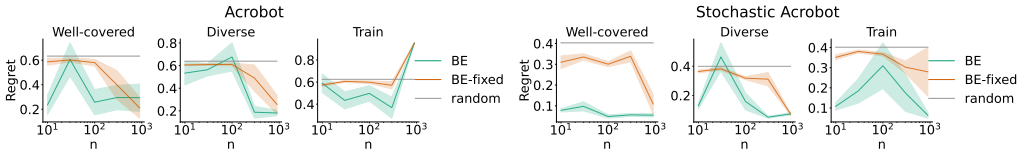



Figure 4: Top-1 regret with varying sample size. The results are averaged over 10 runs with one standard error.

# C Experimental details

In this section, we provide the experimental details for classic RL environments and Atari environments.

## C.1 Classic RL experiments

**FQE Implementation.** Since it is unclear how to perform model selection for FQE, we fix the function approximation as a two layer neural network model with hidden size 256. We still need to choose the learning rate, so we use a simple validation approach. We use the Adam optimizer with learning rate selected from the set $\{0.001, 0.0003, 0.0001, 0.00003\}$ and fix the training epochs to 200. We selected the hyperparameter configuration that resulted in a value function with the smallest RMSTD error evaluated on a separate validation dataset after 200 epochs.

Note that this validation approach for FQE has a big issue. Unlike the validation approach for BE selection, a smaller validation error for FQE does not mean that the FQE estimate is more accurate.

It is known that FQE can diverge, due to the fact that it combines off-policy learning with bootstrapping and function approximation, known as the deadly triad [Sutton and Barto, 2018]. If one of the candidate policies is not well-covered, then the FQE estimate may overestimate the value of the uncovered policy (or even diverge to a very large value) and resulting in poor OPS. To circumvent the issue of uncovered policies, we need assign low value estimates for uncovered policies. In our FQE implementation, we assign low value estimates to policies for which FQE diverges so the OPS algorithm would not choose these policies.

We provide a pseudocode for OPS using FQE in Algorithm 2. In our experiment, we set $U = V_{max} + 100$ (we assume we know $V_{max}$).

**Algorithm 2** OPS using FQE

Input: Candidate set $\Pi$, training data $D$, function class $\mathcal{F}$, threshold $U$
**for** $\pi \in \Pi$ **do**
  Initialize $q_{H-1} = 0$
  for $h = H-2, \ldots, 0$, $q_h \leftarrow \arg\min_{f \in \mathcal{F}} \hat{l}_h(f, q_{h+1})$ where

$$\hat{l}_h(f, q_{h+1}) := \frac{1}{|D_h|} \sum_{(s,a,r,s') \in D_h} (f(s,a) - r - q_{h+1}(s', \pi(s')))^2$$

  Estimate the policy value $\hat{J}(\pi) \leftarrow \mathbb{E}_{a \sim \pi(\cdot|s_0)}[q_0(s_0, a)]$
  **if** $\hat{J}(\pi) > U$ **then**
    $\hat{J}(\pi) \leftarrow -\infty$
Output: $\pi^\dagger \leftarrow \arg\max_{\pi \in \Pi} \hat{J}(\pi)$

**Stochastic environments.** We implement stochastic environments by sticky actions. That is, when the agent selects an action to the environment, the action might be repeated up to $4$ times, with probability $25\%$ repeating the action again.

**Generating candidate policies.** To generate a set of candidate policies, we run CQL with different hyperparameter configurations on a batch of data with $300$ episodes collected with an $\varepsilon$-greedy policy with respect to the optimal policy where $\varepsilon = 0.4$. The hyperparameter configuration includes:

- Learning rate $\in \{0.001, 0.0003, 0.0001\}$
- Network hidden layer size $\in \{128, 256, 512\}$
- Regularization coefficient $\in \{1.0, 0.1, 0.01, 0.001, 0.0\}$
- Iterations of CQL $\in \{100, 200\}$

As a result, we have $90$ candidate policies. For deterministic Cartpole, CQL with many hyperparameter configurations can generate an optimal policy (reaching a return of $200$) so the selection is sufficiently easy that OPS algorithms using both FQE and BE achieve zero regret. In order to make the result more informative, we remove all the policies that are optimal from the candidate set. That results in $67$ candidate policies for Cartpole.

**Generating data for OPS.** To generate data for offline policy selection, we use three different data distributions: (a) a data distribution collected by running the mixture of all candidate policies. As a result, the data distribution covers all candidate policies well; (b) a data distribution collected by running the mixture of all candidate policies and an $\varepsilon$-greedy optimal policy that provides more diverse trajectories than (a); and (d) a data distribution which is the same distribution used to generate training data for CQL.

**Randomness across multiple runs.** To perform experiments with multiple runs, we fix the offline data and the candidate policies and only resample the offline data for OPS. This better reflects the theoretical result that the randomness is from resampling the data for an OPS algorithm. In our experiments, we use $10$ runs and report the average regret with one standard error. Since the variability across runs is not large, we find using $10$ runs is enough.

**Random selection baseline.** We include a random selection baseline that randomly chooses $k$ policies given $k$. Since the random selection algorithm has very high variance, we compute the expected regret of random selection by performing $10000$ random selection, and report the average regret.

**BVFT.** We modify the BVFT implementation from the author of Zhang and Jiang [2021] (`https://github.com/jasonzhang929/BVFT_empirical_experiments/`). BVFT has a hyperparameter to tune, that is, the discretization resolution. We follow the method described in the original paper to search for the best resolution from a set of predefined values. Note that in the author's implementation, they use different sets for different environments.

### C.2 Atari experiments

For the Atari experiments, we use the CQL and FQE implemntation from the d3rlpy package (`https://github.com/takuseno/d3rlpy/`) and the DQN replay dataset (`https://research.google/resources/datasets/dqn-replay/`).

**Generating candidate policies.** To generate a set of candidate policies, we run CQL with the hyperparameters used in the original paper:

- Regularization coefficient $\in \{0.5, 4.0, 5.0\}$
- Number of gradient steps $\in \{50000, 100000, 150000, \ldots, 500000\}$

As a result, we have $30$ candidate policies.

**Randomness across multiple runs.** To perform experiments with multiple runs, we use different runs in DQN replay dataset. We split the the data for each run into a training set to generate the candidate policies and a validation set to perform OPS. That is, the candidate set and the offline data for OPS are random. This reflects the practical experimental situation where the randomness is from different dataset. In our experiments, we use $5$ runs and report the average regret with one standard error. The variability across runs is large, but the DQN replay dataset only provides $5$ runs of data.

# D Paper Checklist

**Broader Impacts.** Our work is foundational research and not tied to particular applications. Therefore, we do not see a direct negative societal impacts.

**Limitations.** In this work, we investigate some conditions to enable sample efficient OPS, with a focus on data coverage condition. We do not discuss other conditions to make OPS efficient, such as the structure of the candidate policy set. For example, if we know that performance is smooth in a hyperparameter for an algorithm, such as a regularization parameter, then it might be feasible to exploit curve fitting to get better estimates of performance.

For the experiments, due to computational constraints, we provided results only on two standard datasets and two Atari datasets. We do not expect the results to be too different for more datasets.

**Compute.** For Atari experiments, we used NVIDIA Tesla V100 GPUs. Each run for generating candidate policies, running FQE, running BE selection took less than 3 hours. For classic RL experiments, we used CPUs only. Each run for generating candidate policies, running FQE, running BE selection also took less than 3 hours.