Oracle Inequalities for Model Selection in Offline Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Offline reinforcement learning (RL) is a promising paradigm where a learner leverages prior data to learn a good policy without interacting with the environment. A major challenge in applying such methods in practice is the lack of both theoretically principled and practical tools for model selection and evaluation. To address this, we study the problem of model selection in offline RL with value function approximation where the learner is given a nested sequence of model classes to minimize squared Bellman error and must select among these to achieve the optimal balance of approximation and estimation error of the classes. We propose, to our knowledge, the first model selection algorithm for offline RL that achieves minimax rate-optimal oracle inequalities up to logarithmic factors. The algorithm, MODBE, takes as input the model classes and a base offline RL algorithm designed to minimize squared Bellman error. It successively eliminates model classes using a novel one-sided generalization test, finally returning a policy that competes with the performance of the best model class. In addition to its theoretical guarantees, it is conceptually simple and computationally efficient, amounting to calculating and comparing relative squared errors between classes. Finally, we demonstrate it is capable of reliably selecting a good model class in small simulated experiments.

1 Introduction

Model selection is a fundamental task in supervised learning and statistical learning theory. Given a sequence of model class, the goal is to optimally balance the approximation error (bias) and estimation error (variance) offered by the potential model class choices. Model selection algorithms are extremely well-studied in learning theory [Mas07, LN99, BBL02, Bar08], and methods like cross-validation are essential steps for practitioners. In recent years, interest has turned to model selection in *online* bandits and reinforcement learning [ALNS17, FKL19, PPAY+20, LPM+21, MJTS20, CMB20, MK21].

Despite this, the current understanding of model selection in the context of *offline* (or batch) reinforcement learning (RL) is comparatively nascent. The offline RL setting describes the paradigm where the learner leverages prior datasets of interactions with the environment [LGR12, LKTF20]. The learner is tasked with returning a good policy without further environment interaction. As has been acknowledged in several recent papers [XJ21, MXW+21, KST+21], one of the major challenges preventing widespread deployment of offline RL algorithms in the real world is the lack of algorithmic tools for model selection, evaluation, and hyperparameter tuning. In experimental settings, researchers typically evaluate candidate learned models by using online rollouts of the policies after learning with offline data. Such approaches are not feasible in many real world settings, where due to logistics, safety or performance requirements, the entire process of offline RL through to producing a single policy, must be conducted only on the offline dataset.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

In recent years, this problem has been recognized as a major deficiency in the field and a number of efforts have been made to remedy it. On the empirical side, several researchers have proposed workflows and general heuristics specifically addressing this problem [KST+21, TW21, PPM+20]. However, all have noted that solutions designed to evaluate or select models typically have their own hyperparameters and modeling choices. Thus, applying off-the-shelf methods, such as those from the offline policy evaluation (OPE) literature [Pre00, TB16], does not solve the model selection problem. Similarly, recent efforts to solve model selection in *online* bandits and RL are inapplicable as they almost universally require interaction with the environment [FKL19, PPAY+20, LPM+21]. The solution to the *offline* problem seems to require new ideas.

On the theoretical side, [LTND21] formalized the problem of model selection in the offline setting with the intent of addressing the above recursive issue. It was shown that full model selection (competitive with an oracle that has knowledge of the best model class) is surprisingly impossible in general in offline reinforcement learning. However, they proposed several relaxations to accomplish partial model selection. Namely, if one is only concerned with optimally balancing approximation error with estimation error (which is our focus), model selection is indeed possible, but they were only able to show this for contextual bandits (single-step horizon).

1.1 Contributions

Theoretical Guarantees In this paper, we build on the theoretical foundations of [LTND21] and give, to our knowledge, the first rate-optimal model selection algorithm for offline reinforcement learning. We consider the model selection problem where we are given an offline dataset of n samples and a nested sequence of M model classes $\mathcal{F}_1 \subseteq \ldots \subseteq \mathcal{F}_M$. For any individual class $\mathcal{F}_k$, the gold-standard regret bound is $\tilde{\mathcal{O}}\left(\sqrt{\textsc{Approx}(\mathcal{F}_k)} + \sqrt{\textsc{Comp}(\mathcal{F}_k)/n}\right)$[1] [CJ19] where $\textsc{Approx}(\mathcal{F}_k)$ denotes the approximation error (i.e., completeness error) of $\mathcal{F}_k$, $\textsc{Comp}(\mathcal{F}_k)$ denotes the statistical complexity of $\mathcal{F}_k$, and n is the number of offline samples.This is achieved, for example, by Fitted Q-Iteration (FQI) [CJ19].

We propose a novel and conceptually simple algorithm, MODel selection via Bellman Error (MODBE), for model selection and prove that it achieves the following oracle inequalities simultaneously: (1) If k_* is the index of the class of minimal complexity satisfying completeness (that is, $\textsc{Approx}(\mathcal{F}_{k_*}) = 0$), it achieves regret $\tilde{\mathcal{O}}\left(\sqrt{\textsc{Comp}(\mathcal{F}_{k_*})/n}\right)$; (2) If no such class exists, it achieves regret $\tilde{\mathcal{O}}\left(\min_{k\in[M]}\left\{\sqrt{\xi_k} + \sqrt{\textsc{Comp}(\mathcal{F}_k)/n}\right\}\right)$, where $\xi_k \geq \textsc{Approx}(\mathcal{F}_k)$ is a global completeness error defined in Section 3.1. Crucially, both guarantees are rate-optimal in $\textsc{Comp}(\cdot)$ and n, and this is achieved without prior knowledge of the optimal model class, the approximation errors, or online interaction with the environment.

Technical Highlights The key to achieving the near optimal regret rate is to achieve the near optimal excess risk rate of the squared Bellman error (which is on the order of $\tilde{\mathcal{O}}(1/n)$). To do this, MODBE iteratively compares the relative effectiveness of a two candidate model class by employing a hypothesis test that compares the difference of their estimated risks to a *one-sided* generalization bound. The fact that the test leverages only the one-sided generalization bound is crucial: deferring to the easier case of two-sided bounds (e.g. from uniform deviation bounds on risk estimators) leads to a squared Bellman error rate of $\tilde{\mathcal{O}}(1/\sqrt{n})$, which translates to a slow $\tilde{\mathcal{O}}(1/n^{1/4})$ regret rate. We use the one-sided generalization error to create a conceptually simple hypothesis test that enables a risk rate of $\tilde{\mathcal{O}}(1/n)$, which translates to the optimal $\tilde{\mathcal{O}}(1/\sqrt{n})$ regret rate.

Practical Results In practice, MODBE can be instantiated with *any* base offline RL algorithm that attempts to minimize squared Bellman error, including but not limited to FQI. MODBE is also computationally efficient, requiring $\mathcal{O}(Hk_*M)$ calls to an empirical squared loss minimization oracle and $\mathcal{O}(k_*)$ calls to the instantiated offline RL algorithm. In Section 5, we demonstrate the effectiveness of MODBE on several simulated experimental domains. We use neural network-based offline RL algorithms as baselines and show that MODBE is able to reliably select the good model class compared to baselines.

[1] For clarity, $\tilde{\mathcal{O}}$ omits dependence on extraneous parameters such as the horizon H, number of classes M, failure probability δ, log factors, and constants.

1.2 Additional Related Work

Several prior works have specifically set out to address the model selection problem from a theoretical perspective, as we do here. [LTND21] formalized the end-to-end model selection problem for offline RL where, given nested model classes, the goal is to produce a regret bound competitive with an oracle that has knowledge of the optimal model class. While the focus was a negative result, their positive results were limited only to linear model classes for contextual bandits. Model selection guarantees of this type for reinforcement learning with general model classes has since remained an open problem. An earlier work by [FS11] had partially addressed this problem but made several restrictive assumptions that are incompatible with that of [LTND21], such as a known generalization bound that underestimates the approximation error (which is generally unknown). Another notable work is the BVFT algorithm of [XJ21]. While initially designed for general policy optimization, BVFT can be applied to model selection [ZJ21] but it incurs a slow $1/n^{1/4}$ regret rate in theory and and requires a stronger data coverage assumption. One advantage of BVFT over our algorithm is that it can be used more generally to tune hyperparameters beyond the selection of model classes. However, the specialization of our algorithm to model selection enables the stronger guarantees. As a result, we view the two algorithms as complementary. On the empirical side, several authors have also proposed general workflows and best practices to make model selection and hyperparameter tuning useful in practice [KST+21, TW21, PPM+20], but they did not address the theoretical aspects of the problem, which still present a significant technical challenge.

2 Preliminaries

Notation For any $n \in \mathbb{N}$, we let $[n] = \{1,\dots,n\}$. The notation $a \lesssim b$ implies that $a \leq Cb$ for some absolute constant $C > 0$. We will use $C, C_1, C_2 \dots > 0$ to denote absolute constants (independent of problem parameters). For a set A, $\Delta(A)$ denotes the set of distributions over A.

We consider the finite-horizon Markov decision process $\mathcal{M}(\mathcal{X},\mathcal{A},H,\mathbb{P},r,\rho)$ where $\mathcal{X}$ is the (potentially infinite) state-space, $\mathcal{A}$ is the action space, H is the length of the horizon, $\mathbb{P}:\mathcal{X}\times\mathcal{A}\to\Delta(\mathcal{X})$ is the transition kernel, $r:\mathcal{X}\times\mathcal{A}\to[0,1]$ is a deterministic reward function, and $\rho\in\Delta(\mathcal{X})$ is an initial state distribution. A learner interacts with the MDP by proposing an H-step policy $\pi=(\pi_h)_{h\in[H]}$ where each $\pi_h : x \mapsto \pi_h(\cdot|x)$ maps $x\in\mathcal{X}$ to a distribution over actions in $\Delta(\mathcal{A})$[2]. At step $h=1$, x_1 is drawn according to ρ. Then at step $h\in[H]$, the agent observes x_h, draws a_h according to $\pi_h(\cdot|x_h)$ observes reward $r(x_h,a_h)$ and the MDP transitions to x_{h+1} according to $\mathbb{P}(\cdot|x_h,a_h)$. For a policy π, we let $P_h^\pi(x,a)$ and $P_h^\pi(x)$ denote the marginal state-action and state of π densities respectively at step h. Note that $P_1^\pi(x)=\rho(x)$ for all π.

Following standard definitions, we let $V_h^\pi : \mathcal{X}\to\mathbb{R}$ denote the value function of π at step $h\in[H]$ which is given by $V_h^\pi(x) = \mathbb{E}_\pi\left[\sum_{s\geq h} r(x_s,a_s)\middle| x_s = x\right]$. Here, the expectation $\mathbb{E}_\pi$ is over trajectories under π with $a_h \sim \pi_h(\cdot|x_h)$. Similarly, the action-value function $Q_h^\pi : \mathcal{X}\times\mathcal{A}\to\mathbb{R}$ is defined as $Q_h^\pi(x,a) = \mathbb{E}_\pi\left[\sum_{s\geq h} r(x_s,a_s)\middle| x_s=x, a_s=a\right]$.The optimal policy (which exists under mild conditions when H is finite [SB18]) is denoted by π^* and this maximizes $V_h^\pi(x)$ for all x and h. The average value of a policy π is given by $v(\pi) := \mathbb{E}_{x\sim\rho}[V_1^\pi(x)]$. Finally, we define the Bellman operators: $T_h^\pi Q(x,a) = r(x,a) + \mathbb{E}_{x'\sim P(\cdot|x,a), a'\sim\pi_{h+1}(\cdot|x')}[Q(x',a')]$ and $T_h^* Q(x,a) = r(x,a) + \mathbb{E}_{x'\sim P(\cdot|x,a)}[\max_{a'\in\mathcal{A}} Q(x',a')]$. Note that the values of $v(\pi)$, V_h^π, and Q_h^π are always in $[0,H]$ due to the constraint on r. For convenience, we denote the Q function of the optimal policy as $Q^* = Q^{\pi^*}$.

To deal with potentially large state and action spaces and enable generalization, we consider the setting where the learner is provided with a model class $\mathcal{F}\subseteq(\mathcal{X}\times\mathcal{A}\to[0,H])$ to estimate action value functions at each step. For exposition, we assume this model class is *finite*, however, it is straightforward to extend to infinite settings with appropriate complexity measures. For notational simplicity, we will assume that the learner uses the same $\mathcal{F}$ for each time-step $h\in[H]$ but this can be trivially extended to the time-varying case. We assume that $0\in\mathcal{F}$, and by convention, we will always write $f_{H+1}=0$. For any function $f\in\mathcal{X}\times\mathcal{A}\to[0,H]$, we define the argmax policy at h as $\pi_f(x) = \mathrm{argmax}_{a\in\mathcal{A}} f(x,a)$. We will also write $f(x) = \max_{a\in\mathcal{A}} f(x,a)$.

[2] With some abuse of notation, for deterministic π_h we write $a=\pi_h(x)$ to denote its highest-probability action.

2.1 Offline Reinforcement Learning

The distinguishing feature of the offline (or batch) RL is that we assume that the learner is provided with a dataset D of example transitions in the MDP. This prior data might be random interactions with the environment, data collected with an existing policy, or even demonstrations from an expert. Furthermore, the learner itself is not permitted to interact in the environment. The objective is to produce a good policy $\hat{\pi}$ using only data from the dataset D.

Formally, the dataset decomposes as $D = (D_h)_{h\in[H]}$ for each timestep where $D_h = \{(x,a,r,x')\}$ consists of tuples of transitions and incurred rewards. We assume D_h contains n datapoints that are sampled i.i.d from a fixed marginal distribution $\mu_h \in \Delta(\mathcal{X}\times\mathcal{A})$. That is, there are Hn datapoints total. For example, the data could be generated from h-step state-action distribution of a behavior policy π^b so that $\mu_h(x,a) = P_h^{\pi^b}(x,a) = \pi_h^b(a|x)P_h^{\pi_b}(x)$ where $\nu_h^{\pi^b}$ denotes the h-step marginal state distribution of policy π^b. However, in general, we do not assume the data was collected by rolling out a policy.

For $f,g \in (\mathcal{X}\times\mathcal{A}\to\mathbb{R})$, we use the notation $\|f-g\|_{\mu_h}^2 = \mathbb{E}_{\mu_h}[f(x,a)-g(x,a)]^2$. The average squared Bellman error under μ at state h with respect to f,g is $\|f-T_h^* g\|_{\mu_h}^2$.

Following classical conventions [MS08, DJL21], we make the assumption that the data distribution μ has good coverage over the MDP for all reachable state-actions.

Assumption 1. *There exists a constant $\mathcal{C}(\mu) > 0$ such that* $\sup_{h,x,a,\pi} \frac{P_h^\pi(x,a)}{\mu_h(x,a)} \le \mathcal{C}(\mu)$.

The constant $\mathcal{C}(\mu)$ is typically unknown. Such assumptions are common in the offline literature and can sometimes be weakened depending on the function class, e.g., minimum eigenvalues for linear classes. We note that recent work has endeavored to relax this assumption so that, for example, only good coverage for a comparison policy is required as opposed to every policy via pessimistic methods. Despite this, Theorem 2 of [LTND21] shows that in general model selection bounds of this type are not possible (even though the single model class bounds are possible), so we will not be concerned with this refinement.

In this offline setting, the learner aims to use D and $\mathcal{F}$ to produce a policy $\hat{\pi}$ so as to minimize the regret, which measures the difference in average value between the optimal policy and $\hat{\pi}$:

$$\mathbf{Reg}(\hat{\pi}) := v(\pi^*) - v(\hat{\pi}). \tag{1}$$

Throughout the paper, we will make use of the following variant of the performance difference lemma, which shows that for value function approximation, it is sufficient to control the squared Bellman error to bound regret.

Lemma 1 ([DJL21]). *For any $f = (f_h)_{h\in[H]}$, let $\pi = (\pi_{f_h})_{h\in[H]}$. Then,*

$$\textbf{\textit{Reg}}(\pi) \le 2\sqrt{\mathcal{C}(\mu)\sum_{h\in[H]}\|f_h - T_h^* f_{h+1}\|_{\mu_h}^2}.$$

3 Model Selection Objectives and Main Result

In this section, we state our primary model selection objectives and introduce our main contributions, namely the model selection algorithm, MODBE, and its theoretical guarantee showing that it is able to compete in a rate-optimal manner with an oracle that knows the optimal model class.

3.1 The Model Selection Problem

For a finite function class $\mathcal{F}$ that we consider here, the gold-standard regret guarantee for offline algorithms with value function approximation is

$$\mathbf{Reg}(\hat{\pi}) = \tilde{\mathcal{O}}\left(\sqrt{\mathcal{C}(\mu)\textsc{Approx}(\mathcal{F})} + \sqrt{\frac{\mathcal{C}(\mu)\log|\mathcal{F}|}{n}}\right), \tag{2}$$

where $\textsc{Approx}(\mathcal{F}) = \max_{h\in[H],f'\in\mathcal{F}} \min_{f\in\mathcal{F}} \|f - T_h^* f'\|_\mu^2$ is the completeness error of the class $\mathcal{F}$ [CJ19]. This is achieved, for example, by the Fitted Q-Iteration (FQI) algorithm. If we were using infinite classes, we would replace $\log|\mathcal{F}|$ with another suitable notion of complexity such as pseudodimension. Such bounds naturally exhibit a trade-off: larger function classes may have a better

chance of keeping $\textsc{Approx}(\mathcal{F})$ close to zero[3] but require more data to minimize the estimation error. Meanwhile small classes face the opposite problem.

The objective of model selection is to achieve refined regret bounds that balance approximation error and estimation error. To this end, we assume that the learner is presented with, not just a single model class $\mathcal{F}$, but rather a nested sequence of M classes $\mathcal{F}_1 \subseteq \ldots \subseteq \mathcal{F}_M$. Solving a problem with nested model classes is an extremely common practice in both supervised learning and RL. For example, one often starts with an extremely large class $\mathcal{F}$ and the considers restrictions of $\mathcal{F}$ to an increasing sequence $\mathcal{F}_1, \ldots, \mathcal{F}_M = \mathcal{F}$. In a linear setting, this could correspond to trying to find a subset of candidate features that are sufficient to solve the problem.

Since it is unknown a priori which class has the best balance (as the approximation error is unknown), we aim to design an algorithm capable of selecting a good class in data-dependent manner. In particular, we would like to achieve *oracle inequalities* that reflect we can compete with the performance of an oracle that has this knowledge. Model selection asks whether it is possible to achieve $\mathbf{Reg}(\hat{\pi}) = \tilde{\mathcal{O}}\Big(\min_{k\in[M]}\Big\{\sqrt{\mathcal{C}(\mu)\textsc{Approx}(\mathcal{F}_k)} + \sqrt{\mathcal{C}(\mu)\log|\mathcal{F}_k|/n}\Big\}\Big)$.

A slightly weaker objective is to achieve the following two types of oracle inequalities, which we consider in this paper:

1. Let $k_* = \min\{k : \textsc{Approx}(\mathcal{F}_k) = 0\}$. Find $\hat{\pi}$ so that $\mathbf{Reg}(\hat{\pi}) = \tilde{\mathcal{O}}\Big(\sqrt{\mathcal{C}(\mu)\log|\mathcal{F}_{k_*}|/n}\Big)$.
2. Define the global completeness error as $\xi_k := \max_{h\in[H], f'\in\mathcal{F}_M} \min_{f\in\mathcal{F}_k} \|f - T_h^* f'\|_{\mu_h}^2$. Find $\hat{\pi}$ so that $\mathbf{Reg}(\hat{\pi}) = \tilde{\mathcal{O}}\Big(\min_{k\in[M]}\Big\{\sqrt{\mathcal{C}(\mu)\xi_k} + \sqrt{\mathcal{C}(\mu)\log|\mathcal{F}_k|/n}\Big\}\Big)$.

The first oracle inequality asks whether it is possible to compete against the minimally complex class that has no approximation error. That is $\mathcal{F}_{k_*}$ is smallest class that satisfies completeness on the data distribution. Such oracle inequalities are extremely common in model selection for *online* bandits and RL – albeit they are generally not rate-optimal in that literature unlike the stronger objective we consider here [ALNS17]. We would also like to be robust to the case where k_* does not exist and not have to rely on the assumption that some (albeit unknown) model class satisfies completeness. The second oracle inequality handles this by defining a global completeness error ξ_k to replace the approximation error. Note that $\xi_k \geq \textsc{Approx}(\mathcal{F}_k)$ by definition. Our proposed algorithm will be able to handle both *simultaneously*.

A crucial aspect of both oracle inequalities is that their estimation errors are rate-optimal in n and $\log(\mathcal{F}_k)$, matching the rate of the single model class case. That is, we do not tolerate any worse dependence on either quantity such as $\mathcal{O}(1/n^{1/4})$ rates and other lower order terms.

3.2 MODBE Algorithm

Having introduced the model selection objectives, we now present our main result, a novel model selection algorithm for offline RL that provably achieves the aforementioned oracle inequalities. We will focus specifically on the implications of the result and defer the intuition of the method to Section 4.

The algorithm, MODBE (Model Selection via Bellman Error), is presented in Algorithm 1. It takes as input the base offline RL algorithm, the model classes $\mathcal{F}_1 \subseteq \ldots \subseteq \mathcal{F}_M$, and the offline dataset D of n i.i.d samples for each time-step $h \in [H]$. The dataset is split randomly into a training set D_{train} and a validation set D_{valid}. The algorithm begins optimistically, starting with the candidate model class $k = 1$ and running the base algorithm with $\mathcal{F}_k$ on the training dataset to generate the candidate action-value functions f^k. We then use the larger model classes $k' > k$ to evaluate this function, by using the target values $r + f_{h+1}^k(x')$ as regression targets for model class $\mathcal{F}_{k'}$. Varying over h, this amounts to solving a sequence of H least squares regression problems using class k', yielding the functions $(f_h^{k'})_{h\in[H]}$.

Since f_h^k and $f_h^{k'}$ are attempting to solve the *same* regression problem (with the same target values), we can use the validation set to compare their performance on this shared squared loss objective $\tilde{L}$. This comparison takes the form of a *generalization error test* in Line 11. If the test fails and it is discovered that the larger model class $\mathcal{F}_{k'}$ is able to achieve substantially smaller loss than f^k (relative to a tolerance), then we have reason to believe performance can be much better by moving to a larger

[3] In contrast to realizability, this intuition of monotonicity of $\textsc{Approx}(\mathcal{F})$ is not universally true for completeness – by adding functions to the class $\mathcal{F}$, we might actually *increase* $\textsc{Approx}(\mathcal{F})$. However, it remains a useful heuristic.

Algorithm 1 Model Selection via Bellman Error (MODBE)

1: **Input**: Offline dataset $D=(D_h)$ of n samples for each $h\in[H]$, Base algorithm $\mathcal{B}$, function classes $\mathcal{F}_1\subseteq\ldots\subseteq\mathcal{F}_M$, failure probability $\delta\leq 1/e$.
2: Let $n_{\text{train}}=\lceil 0.8\cdot n\rceil$ and $n_{\text{valid}}=\lfloor 0.2\cdot n\rfloor$ and split the dataset D randomly into $D_{\text{train}}=(D_{\text{train},h})$ of n_{train} samples and $D_{\text{valid}}=(D_{\text{valid},h})$ of n_{valid} samples for each $h\in[H]$.
3: Set $\delta_k := \max\left\{\omega_{n_{\text{train}},\delta/4M}(\mathcal{F}_k), \frac{200H^2\log(4M^2H|\mathcal{F}_k|/\delta)}{n_{\text{train}}}\right\}$ for all $k\in[M]$ and $\zeta := \frac{96H^2\log(16M^2H/\delta)}{n_{\text{valid}}}$.
4: Calculate tolerance $\text{TOL}_{n_{\text{train}}}(\mathcal{F}_k,\mathcal{F}_{k'}):=2\delta_{k'}+2\zeta+\omega_{n_{\text{train}},\delta/4M}(\mathcal{F}_k)$ for all $k<k'$.
5: Initialize $k\rightarrow 1$.
6: **while** $k<M$ **do**
7: $f^k:=(f_h^k)_{h\in[H]}\leftarrow\mathcal{B}(D_{\text{train}},\mathcal{F}_k,\delta/4M)$
8: **for** $k'\leftarrow k+1,\ldots,M$ **do**
9: Minimize squared loss on training set for each $h\in[H]$ with regression targets from class k:

$$g_h^{k'}\leftarrow\operatorname*{argmin}_{g\in\mathcal{F}_{k'}}\quad\hat{L}_h(g,f_{h+1}^k):=\frac{1}{n_{\text{train}}}\sum_{(x,a,r,x')\in D_{\text{train},h}}\left(g(x,a)-r-f_{h+1}^k(x')\right)^2 \tag{3}$$

10: Compute squared loss estimator using the validation set for each $h\in[H]$ as a function of f:

$$\tilde{L}_h(f,f_{h+1}^k)=\frac{1}{n_{\text{valid}}}\sum_{(x,a,r,x')\in D_{\text{valid},h}}\left(f(x_h,a_h)-r_h-f_{h+1}^k(x')\right)^2 \tag{4}$$

11: **if** $\tilde{L}(g_h^{k'},f_{h+1}^k)<\tilde{L}(f_h^k,f_{h+1}^k)-\text{TOL}_{n_{\text{train}}}(\mathcal{F}_k,\mathcal{F}_{k'})$ for any $h\in[H]$ **then**
12: $k\leftarrow k+1$
13: goto Line 6.
14: **end if**
15: **end for**
16: goto Line 18
17: **end while**
18: **return** $\hat{\pi}=\left(\pi_{f_h^k}\right)_{h\in[H]}$

model class $k\leftarrow k+1$. The process is repeated until we exhaust all our model classes or find that no larger model class k' offers a big enough improvement over k to cause the test to fail.

3.3 Rate-Optimal Oracle Inequalities

We show that this simple procedure is able to achieve the oracle inequalities of the previous section. We start with a generic version of the theorem that is stated in terms of an assumed performance bound on the base algorithm for value function approximation. We then instantiate the base algorithm with FQI, showing that this version precisely achieves the desired oracle inequalities with the correct rate.

Definition 1. *Let $\mathcal{B}$ be a base offline RL algorithm for value function approximation that takes as input a model class $\mathcal{F}$, an offline dataset D of n samples for each $h\in[H]$, and a failure probability δ. For $L>0$ and a function ω, we say that $\mathcal{B}$ is (L,ω)-regular if (1) ω is a known function and satisfies $\omega_{n,\delta}(\mathcal{F}_k)\leq\omega_{n,\delta}(\mathcal{F}_k')$ for all $k'\geq k$; (2) $\mathcal{B}(D,\mathcal{F}_k,\delta)$ returns $f=(f_h)_{h\in[H]}\subseteq\mathcal{F}_k$ such that f_{h+1} is independent of D_h and*

$$P\left(\max_{h\in[H]}\|f_h-T_h^*f_{h+1}\|_{\mu_h}^2\leq L\cdot\text{APPROX}(\mathcal{F}_k)+\omega_{n,\delta}(\mathcal{F}_k)\right)\geq 1-\delta. \tag{5}$$

Theorem 1. *Let $\mathcal{B}$ be an (L,ω)-regular algorithm. Then Algorithm 1 with inputs $\mathcal{B}$, $\mathcal{F}_1\subseteq\ldots\subseteq\mathcal{F}_M$, and $\delta\leq 1/e$ outputs $\hat{\pi}$ such that, with probability at least $1-\delta$,*

$$\textbf{\textit{Reg}}(\hat{\pi})\leq C_0\cdot\min_{k\in[M]}\left\{\sqrt{\mathcal{C}(\mu)H\left(L\xi_k+\omega_{n_{train},\delta/4M}(\mathcal{F}_k)+\frac{H^2(\log|\mathcal{F}_k|+\iota)}{n}\right)}\right\} \tag{6}$$

for some absolute constant $C_0 > 0$ *and* $\iota = \log(M^2 H/\delta)$. *Furthermore if* $k_* = \min\{k : \textsc{Approx}(\mathcal{F}_k) = 0\}$ *exists, then, under the same event,* $\hat{\pi}$ *also satisfies*

$$\textbf{\textit{Reg}}(\hat{\pi}) \leq C_1 \cdot \sqrt{\mathcal{C}(\mu) H \left(\omega_{n_{train}, \delta/4M}(\mathcal{F}_{k_*}) + \frac{H^2(\log|\mathcal{F}_{k_*}| + \iota)}{n} \right)} \tag{7}$$

for some absolute constant $C_1 > 0$.

For concreteness, consider standard finite-horizon FQI [DJW20], which is a base algorithm satisfying Definition 1 with $\omega_n(\mathcal{F}) = \mathcal{O}(\log|\mathcal{F}|/n)$. This in turn translates to the desired rate-optimal oracle inequalities.

Lemma 2. *Consider the FQI algorithm (stated in Appendix B for completeness). For a model class* $\mathcal{F}$, *FQI is a* $(3,\omega)$*-regular base algorithm with* $\omega_{n,\delta}(\mathcal{F}) = \frac{200 H^2 \log(16H|\mathcal{F}|/\delta)}{n}$.

Armed with this guarantee, it is now immediate that Algorithm 1 instantiated with FQI achieves the desired oracle inequalities.

Corollary 1. *Let* $\mathcal{B}$ *be instantiated with FQI (Algorithm 2 in Appendix B). Define* $\iota = \log(M^2 H/\delta)$ *Then, under the same conditions as Theorem 1, there are absolute constants* $C_0, C_1 > 0$ *such that, with probability at least* $1-\delta$, *Algorithm 1 outputs* $\hat{\pi}$ *satisfying*

$$\textbf{\textit{Reg}}(\hat{\pi}) \leq C_0 \cdot \min_{k \in [M]} \left\{ \sqrt{\mathcal{C}(\mu) H \xi_k} + \sqrt{\frac{\mathcal{C}(\mu) H^3 (\log|\mathcal{F}_k| + \iota)}{n}} \right\} \tag{8}$$

and, if k_* *exists,*

$$\textbf{\textit{Reg}}(\hat{\pi}) \leq C_1 \cdot \sqrt{\frac{\mathcal{C}(\mu) H^3 (\log|\mathcal{F}_{k_*}| + \iota)}{n}}. \tag{9}$$

Computational Complexity MODBE is computationally efficient given a squared loss regression oracle. The procedure itself requires an inner and outer loop over the model classes. Within these loops, a squared loss minimizer is computed on the training dataset and then functions are evaluated on the validation set. Given access to squared loss regression oracle [SLX21], MODBE requires only $\mathcal{O}(H k_* M)$ calls to the computational oracle when k_* exists (a consequence of Theorem 1) or $\mathcal{O}(H M^2)$ in the worst case. Note that algorithms for optimizing squared loss regression problems are ubiquitous in machine learning.

4 Technical Challenges and Overview of the Method

In this section, we discuss the intuition for Algorithm 1. We first provide a description of challenges faced by seemingly natural approaches to model selection that are actually unable to resolve the problem satisfactorily. We then provide further intuition for the algorithm and proof.

4.1 Challenges

Adaptive offline policy evaluation The most natural approach, to which we have alluded in the introduction, is to apply an off-the-shelf offline policy evaluation approach such as fitted Q-evaluation [MS08, DJW20], DICE methods [NCDL19, DNC+20, ZHH+22], marginalized importance estimators [XMW19], or doubly robust estimators [TB16]. One could then attempt to estimate $v(\hat{\pi}_k)$ for each $k \in [M]$ generated by the base algorithm using function class $\mathcal{F}_k$. The main drawback of this approach is that nearly all of the above methods require selecting a model class to perform the estimation[4], and it is unclear how to balance the estimation and approximation error optimally to compete with the oracle.

One possible solution is to employ the adaptive estimator of [SSK20], which takes as inputs a sequence of offline estimators and known upper bounds on their deviations and returns an estimator that competes with the best one. This is precisely the approach taken by [LTND21] for linear contextual bandits. However, for general function classes in RL, there is no obvious way to compute the analogous deviation bounds, which oftentimes depend on the unknown quantity $\mathcal{C}(\mu)$ in the worst case. Since these bounds are required by the adaptive estimator as inputs, we are yet again left with unknown hyperparameters to tune.

[4] In the case of marginalize importance sampling, the guarantee is not strong enough to compete with the oracle.

Bellman error estimators Another very natural approach arises from the fact that we are considering base algorithms that attempt to minimize the squared Bellman error of objective. One might ask whether it is possible to estimate the Bellman errors (e.g. with the validation dataset) and compare the model classes using the Bellman error as a proxy. The main issue with this approach is the classic double-sampling problem [Bai95, DJL21] where, as a result of using an empirical version of the Bellman operator T^*. By selecting based on the validation error alone, we will end up favoring model classes that also induce low variance since:

$$\mathbb{E}_{\mu_h}\left[\tilde{L}_h(f,g)\right] = \|f - T^* g\|^2_{\mu_h} + \mathbb{E}_{\mu_h}\left[\operatorname*{var}_{x' \sim P(\cdot|x,a)}\left(\max_{a' \in \mathcal{A}} g(x',a')\right)\right]. \tag{10}$$

In the same vein, another approach we might consider is recent BVFT algorithm of [XJ21] to select among the f^k learned by the base algorithm. However the guarantee of BVFT unfortunately exhibits a slow $O(1/n^{1/4})$ and thus does not achieve either oracle inequality. It also, in theory, requires that a discretization parameter is set based concentrability coefficient stronger than $\mathcal{C}(\mu)$, which is typically unknown.

Representation Learning Readers familiar with work in representation learning for RL [AKKS20] might observe that our selection procedure vaguely resembles algorithms designed to select feature representations for linear MDPs such as [MCK+21]. They are similar in the sense that they both use one model class (or feature representation) to identify a gap in another model class. Unfortunately, the problem settings are quite different, and we cannot simply adapt such representation learning algorithms to the model selection problem since they are either insensitive to the model class complexities, thus making them incapable of satisfying an oracle inequality, or they require strong realizability assumptions. However, it would be interesting to better understand the relationship between these two problems and their methods in the future.

4.2 A Test for Generalization Error

We now outline the intuition of the algorithm and the proof of Theorem 1. For more technical depth, we include both a proof sketch and the full proof in Appendix A.

MODBE can be viewed as a refinement of the Bellman error estimator mentioned in the previous section. Recall that the problem with the naive method is that $L_h(f^k_{h+1}, f^k_{h+1})$ produces a biased estimate of the Bellman error due to the double-sampling problem. Inherently, the issue with comparing two classes $\mathcal{F}_k$ and $\mathcal{F}_{k'}$ by using $L_h(f^k_{h+1}, f^k_{h+1})$ and $L_h(f^{k'}_{h+1}, f^{k'}_{h+1})$ is that f^k_h and $f^{k'}_h$ are solving two different regression problems. For class k, the regression target is $T^* f^k_{h+1}$. Meanwhile, for class k', the target is $T^* f^{k'}_{h+1}$.

However, we can compare model classes based on their relative performance on the *same* regression problem. For any $h \in [H]$, Definition 1 provides us with a bound on the Bellman error; i.e., this is the generalization error that we should expect when the regression target is f^k_{h+1}. We can also use class $\mathcal{F}_{k'}$ to try to solve the same regression problem, generating $g^{k'}_h$ which minimizes the training loss $\hat{L}_h(\cdot, f^k_{h+1})$.

On the validation set, if we find that $\tilde{L}_h(f^k_h, f^k_{h+1}) \leq \tilde{L}_h(g^{k'}_h, f^k_{h+1})$, then we have no evidence to abandon class $\mathcal{F}_k$ because it was able to solve its own regression problem better than $\mathcal{F}_{k'}$. On the other hand, if $\tilde{L}_h(g^{k'}_h, f^k_{h+1}) \leq \tilde{L}_h(f^k_h, f^k_{h+1})$ it is plausible that we might benefit from moving to a larger model class since $\mathcal{F}_{k'}$ was able to achieve better generalization error. However, while $\mathcal{F}_{k'}$ may have good generalization error when $T^*_h f^k_{h+1}$ is the target, its true performance will still be measured by its generalization error when $T^*_h f^{k'}_{h+1}$ is the target, which could end up being much larger. To deal with this, we propose a *one-sided* generalization error test: a switch will occur if

$$\tilde{L}_h(g^{k'}_h, f^k_{h+1}) < \tilde{L}_h(f^k_h, f^k_{h+1}) - \textsc{Tol}_{n_{\text{train}}}(\mathcal{F}_k, \mathcal{F}_{k'}). \tag{11}$$

That is, a switch will occur not when $\mathcal{F}_{k'}$ performs better than $\mathcal{F}_k$ when the target is $T^*_h f^k_{h+1}$, but when it performs *substantially* better as measured by the amount of estimation error we expect to observe both model classes exhibiting on this regression problem. If (11) holds, then there is reason to believe that the approximation error of $\mathcal{F}_k$ is large enough to make $\mathcal{F}_{k'}$ a better class. Crucially, the test only checks for generalization error, so the tolerance is in terms of $\omega_{n_{\text{train}}}(\mathcal{F}_k)$ and $\delta_{n_{\text{train}}}(\mathcal{F}_{k'}) = \tilde{\mathcal{O}}\left(\frac{\log \mathcal{F}_{k'}}{n_{\text{train}}}\right)$, which is the correct rate for this problem. Thus, if the test turns out to be wrong, we will only lose additive factors of the correct rate.

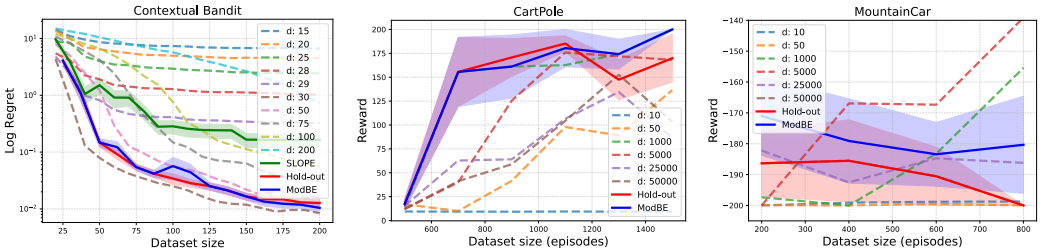

Figure 1: MODBE is evaluated on several simulated domains: a contextual bandit (left), CartPole (middle), and MountainCar (right). In CB, MODBE and Hold-out outperform SLOPE and match performance of the best model class in regret. In CartPole, both match the performance of the best model class. In MountainCar, both struggle to match the best model class, but MODBE maintains marginally superior performance. In CB, error bands are standard error over 10 random trials. In RL, error bands are standard error over 5 random trials.

5 Empirical Results

The previous sections outlined the strong theoretical properties of MODBE. In this section, we ask: what practical insights can we gleaned from MODBE and its theoretical guarantees? In particular, we would like to understand if the core selection method of MODBE can be applied out-of-the-box on existing offline RL algorithms with minimal effort. We evaluated MODBE in three simulated environments with discrete actions: (1) synthetic contextual bandits (CB), (2) Gym CartPole, (3) Gym MountainCar. See Appendix C for specific details about the setups. All training and validation sets were split 80/20.

Contextual Bandit As a basic validation experiment, we started with the CB setting of [LTND21] which considers a nested sequence of linear model classes with increasing dimension d. Without any tuning, we simply set the tolerance of MODBE to $\textsc{Tol}(\mathcal{F}_k, \mathcal{F}_{k'}) = \frac{d_{k'}}{n}$. Figure 1 shows the results in terms of the log-regret as a function of the dataset size. We observe that both MODBE and Hold-Out (choosing the model class with the smallest error) are able to easily match the performance of the best model class while SLOPE [LTND21] ends up being fooled by nearby classes.

RL Discrete Control Our experimental setup for the RL problems in Gym [BCP+16] builds on top of the d3rlpy framework [SI21], which contains open-source implementations of offline RL algorithms. We used DQN [MKS+15] (which is closest to FQI). In both CartPole and MountainCar, we considered model classes that were two-layer neural networks with ReLU activations and d nodes in the hidden layer and varied the parameter d. Again, we simply set the tolerance of MODBE to d_k/n motivated by pseudodimension bounds [BHLM19]. For simplicity, we modified MODBE to work in the discounted infinite horizon setting, which can trivially be done (see Appendix C for details on this modification). The neural network classes considered had $d \in \{10, 50, 1000, 5000, 25000, 50000\}$. In both settings, we compared MODBE to Hold-Out, which is a seemingly sensible baseline that chooses the model class with lowest estimated Bellman error on a validation set. For deterministic settings only, this is theoretically justified. Figure 1 shows the reward as a function of the dataset size (in episodes). On CartPole, MODBE and Hold-Out are both able to compete with the best classes and are roughly at parity. However, on MountainCar, we find that Hold-Out does surprisingly poorly while MODBE is successfully able to reject the poor model classes. We conjecture that the empirical failure of Hold-Out (which is not predicted in theory since the environment is deterministic) is possibly due to sensitivity to optimization error that makes the inherent Bellman error misleading. In contrast, the generalization test of MODBE seems to be more robust to this.

6 Discussion

In this paper, we introduced a new algorithm, MODBE, for model selection in offline RL. To our knowledge, this is the first model selection algorithm in this setting to achieve rate-optimal oracle inequalities in n and $\textsc{Comp}(\mathcal{F}_{k_*})$. A number of interesting open questions remain. (1) The global completeness ξ is potentially much worse than $\textsc{Approx}(\mathcal{F})$. Is it possible to achieve a robust oracle inequality of the form $\mathcal{O}(\min_k \sqrt{\textsc{Approx}(\mathcal{F}_k)} + \sqrt{\log|\mathcal{F}_k|/n})$ when k_* does not exist? (2) Are there rate-optimal procedures that can be used to select hyperparameters beyond model complexity such as learning rates, batch sizes, *et cetera*? (3) Can the ideas of MODBE be extended to more general algorithms that do not rely on Bellman error minimization? We believe these questions are of great practical and theoretical importance for understanding how to effectively evaluate and select models in offline RL.

References

[AKKS20] Alekh Agarwal, Sham Kakade, Akshay Krishnamurthy, and Wen Sun. Flambe: Structural complexity and representation learning of low rank mdps. *Advances in neural information processing systems*, 33:20095–20107, 2020.

[ALNS17] Alekh Agarwal, Haipeng Luo, Behnam Neyshabur, and Robert E Schapire. Corralling a band of bandit algorithms. In *Conference on Learning Theory*, pages 12–38. PMLR, 2017.

[Bai95] Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier, 1995.

[Bar08] Peter L Bartlett. Fast rates for estimation error and oracle inequalities for model selection. *Econometric Theory*, pages 545–552, 2008.

[BBL02] Peter L Bartlett, Stéphane Boucheron, and Gábor Lugosi. Model selection and error estimation. *Machine Learning*, 48(1):85–113, 2002.

[BCP+16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym (2016). *arXiv preprint arXiv:1606.01540*, 2016.

[BHLM19] Peter L Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *The Journal of Machine Learning Research*, 20(1):2285–2301, 2019.

[CJ19] Jinglin Chen and Nan Jiang. Information-theoretic considerations in batch reinforcement learning. In *International Conference on Machine Learning*, pages 1042–1051. PMLR, 2019.

[CMB20] Niladri Chatterji, Vidya Muthukumar, and Peter Bartlett. Osom: A simultaneously optimal algorithm for multi-armed and linear contextual bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 1844–1854, 2020.

[DJL21] Yaqi Duan, Chi Jin, and Zhiyuan Li. Risk bounds and rademacher complexity in batch reinforcement learning. In *International Conference on Machine Learning*, pages 2892–2902. PMLR, 2021.

[DJW20] Yaqi Duan, Zeyu Jia, and Mengdi Wang. Minimax-optimal off-policy evaluation with linear function approximation. In *International Conference on Machine Learning*, pages 2701–2709. PMLR, 2020.

[DNC+20] Bo Dai, Ofir Nachum, Yinlam Chow, Lihong Li, Csaba Szepesvári, and Dale Schuurmans. Coindice: Off-policy confidence interval estimation. *Advances in neural information processing systems*, 33:9398–9411, 2020.

[FKL19] Dylan Foster, Akshay Krishnamurthy, and Haipeng Luo. Model selection for contextual bandits. *arXiv preprint arXiv:1906.00531*, 2019.

[FS11] Amir-massoud Farahmand and Csaba Szepesvári. Model selection in reinforcement learning. *Machine learning*, 85(3):299–332, 2011.

[KST+21] Aviral Kumar, Anikait Singh, Stephen Tian, Chelsea Finn, and Sergey Levine. A workflow for offline model-free robotic reinforcement learning. *arXiv preprint arXiv:2109.10813*, 2021.

[LGR12] Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer, 2012.

[LKTF20] 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.

[LN99] Gábor Lugosi and Andrew B Nobel. Adaptive model selection using empirical complexities. *The Annals of Statistics*, 27(6):1830–1864, 1999.

[LPM+21] 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*, pages 3340–3348. PMLR, 2021.

[LTND21] Jonathan N Lee, George Tucker, Ofir Nachum, and Bo Dai. Model selection in batch policy optimization. *arXiv preprint arXiv:2112.12320*, 2021.

[Mas07] Pascal Massart. *Concentration inequalities and model selection: Ecole d'Eté de Probabilités de Saint-Flour XXXIII-2003*. Springer, 2007.

[MCK+21] Aditya Modi, Jinglin Chen, Akshay Krishnamurthy, Nan Jiang, and Alekh Agarwal. Model-free representation learning and exploration in low-rank mdps. *arXiv preprint arXiv:2102.07035*, 2021.

[MJTS20] Aditya Modi, Nan Jiang, Ambuj Tewari, and Satinder Singh. Sample complexity of reinforcement learning using linearly combined model ensembles. In *International Conference on Artificial Intelligence and Statistics*, pages 2010–2020. PMLR, 2020.

[MK21] Vidya Muthukumar and Akshay Krishnamurthy. Universal and data-adaptive algorithms for model selection in linear contextual bandits. *arXiv preprint arXiv:2111.04688*, 2021.

[MKS+15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[MS08] Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(5), 2008.

[MXW+21] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[NCDL19] Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *Advances in Neural Information Processing Systems*, 32, 2019.

[PPAY+20] Aldo Pacchiano, My Phan, Yasin Abbasi Yadkori, Anup Rao, Julian Zimmert, Tor Lattimore, and Csaba Szepesvari. Model selection in contextual stochastic bandit problems. *Advances in Neural Information Processing Systems*, 33:10328–10337, 2020.

[PPM+20] 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.

[Pre00] Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[SI21] Takuma Seno and Michita Imai. d3rlpy: An offline deep reinforcement learning library. *arXiv preprint arXiv:2111.03788*, 2021.

[SLX21] David Simchi-Levi and Yunzong Xu. Bypassing the monster: A faster and simpler optimal algorithm for contextual bandits under realizability. *Mathematics of Operations Research*, 2021.

[SSK20] Yi Su, Pavithra Srinath, and Akshay Krishnamurthy. Adaptive estimator selection for off-policy evaluation. In *International Conference on Machine Learning*, pages 9196–9205. PMLR, 2020.

[TB16] Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148. PMLR, 2016.

[TW21] Shengpu Tang and Jenna Wiens. Model selection for offline reinforcement learning: Practical considerations for healthcare settings. In *Machine Learning for Healthcare Conference*, pages 2–35. PMLR, 2021.

[XJ21] Tengyang Xie and Nan Jiang. Batch value-function approximation with only realizability. In *International Conference on Machine Learning*, pages 11404–11413. PMLR, 2021.

[XMW19] Tengyang Xie, Yifei Ma, and Yu-Xiang Wang. Towards optimal off-policy evaluation for reinforcement learning with marginalized importance sampling. *Advances in Neural Information Processing Systems*, 32, 2019.

[ZHH+22] Wenhao Zhan, Baihe Huang, Audrey Huang, Nan Jiang, and Jason D Lee. Offline reinforcement learning with realizability and single-policy concentrability. *arXiv preprint arXiv:2202.04634*, 2022.

[ZJ21] Siyuan Zhang and Nan Jiang. Towards hyperparameter-free policy selection for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Introduction, Discussion, and discussions of assumptions and after every theorem statement.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] The work is theoretical in nature.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See preliminaries and before theorem statements.
   (b) Did you include complete proofs of all theoretical results? [Yes] See appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Empirical Results section and Appendix C.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix C.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]