Oracle Inequalities for Model Selection in Offline Reinforcement Learning

Jonathan N. Lee
Stanford University
jnl@stanford.edu

George Tucker
Google Research
gjt@google.com

Ofir Nachum
Google Research
ofirnachum@google.com

Bo Dai
Google Research
bodai@google.com

Emma Brunskill
Stanford University
ebrun@cs.stanford.edu

Abstract

In offline reinforcement learning (RL), a learner leverages prior logged 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. The learner is given a nested sequence of model classes to minimize squared Bellman error and must select among these to achieve a balance between approximation and estimation error of the classes. We propose 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 a collection of candidate model classes and a generic base offline RL algorithm. By successively eliminating model classes using a novel one-sided generalization test, MODBE returns a policy with regret scaling with the complexity of the *minimally complete* model class. In addition to its theoretical guarantees, it is conceptually simple and computationally efficient, amounting to solving a series of square loss regression problems and then comparing relative square loss between classes. We conclude with several numerical simulations showing it is capable of reliably selecting a good model class.[1]

1 Introduction

Model selection is a fundamental task in supervised learning and statistical learning theory. Given a sequence of model classes, the goal is to optimally balance the approximation error (bias) and estimation error (variance) offered by the potential model class choices, even though the best model class is not known in advance. Model selection algorithms are extremely well-studied in learning theory (Massart, 2007; Lugosi and Nobel, 1999; Bartlett et al., 2002; Bartlett, 2008), and methods like cross-validation have become essential steps for practitioners.

In recent years, interest has turned to model selection in decision-making problems like bandits and reinforcement learning. A number of theoretical works have studied the *online* setting (Agarwal et al., 2017; Foster et al., 2019; Pacchiano et al., 2020; Lee et al., 2021a; Modi et al., 2020; Chatterji et al., 2020; Muthukumar and Krishnamurthy, 2021). Similar to the bias-variance balance in supervised learning, these algorithms typically aim to select the model class with smallest statistical complexity that contains the true model. Despite these recent efforts, the current understanding of model selection in *offline* (or batch) reinforcement learning (RL) is comparatively nascent. Offline RL is a paradigm where the

[1] Supplementary material is available at: https://sites.google.com/stanford.edu/offline-model-selection.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

learner leverages prior datasets of logged interactions with the environment (Lange et al., 2012; Levine et al., 2020). The learner is tasked with returning a good policy without further environment interaction. As has been acknowledged in several recent papers (Xie and Jiang, 2021; Mandlekar et al., 2021; Kumar et al., 2021), 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. However, such approaches are not feasible in many real world settings where the entire process of producing a single policy must be conducted only on the offline dataset, due to complications such as logistics, safety, or performance requirements.

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 (Kumar et al., 2021; Tang and Wiens, 2021; Paine et al., 2020). However, all have noted that solutions designed to evaluate or select models typically have their own hyperparameters and modeling choices. Consider, for example, applying off-the-shelf offline policy evaluation (OPE) methods (Precup, 2000; Thomas and Brunskill, 2016). These typically require some function approximation of their own. Thus, rather than solving the problem, naively using OPE just shifts the burden of model selection to the OPE estimator. Similarly, recent efforts to solve model selection in *online* bandits and RL are inapplicable as they almost universally require interaction with the environment (Foster et al., 2019; Pacchiano et al., 2020; Lee et al., 2021a). The solution to the *offline* problem seems to require new ideas.

On the theoretical side, there is also significant motivation for devising model selection algorithms as there is growing evidence suggesting that strong conditions on the function class[2] are necessary to achieve non-trivial guarantees in offline RL in the worst case (Foster et al., 2021; Zanette, 2021; Wang et al., 2020). Perhaps the most widely used and recognized condition is completeness (Munos and Szepesvári, 2008; Antos et al., 2008; Chen and Jiang, 2019) which essentially says that $\mathcal{T} f \in \mathcal{F}$ for any $f \in \mathcal{F}$, where $\mathcal{T}$ is the Bellman operator and $\mathcal{F}$ is the model class.[3] Unsurprisingly, completeness plays an important role in the proofs of many value-based offline RL algorithms since sample efficient results are provably impossible without it (in the absence of additional assumptions – see Xie and Jiang (2021); Zhang and Jiang (2021)). Despite the growing realization of the importance of these conditions, there seems to be comparatively little work addressing the problems of identifying complete model classes or certifying sufficient conditions for sample efficient offline RL.

Lee et al. (2021b) considered the problem of model selection in the offline setting with the intent of addressing some of the aforementioned issues. It was shown that full model selection (competitive with an oracle that has knowledge of the best model class) is impossible in general in offline reinforcement learning. They proposed several relaxations to achieve weaker oracle inequalities, but these were limited to contextual bandits with linear model classes where there is no issue of completeness. The question of whether any similar results are possible for full offline reinforcement learning with general function classes has remained open.

1.1 Contributions

Theoretical Guarantees In this paper, we give the first rate-optimal model selection algorithm for offline RL with value function approximation. We begin by summarizing known results for a single model class using value-based methods. For any individual model class $\mathcal{F}$ that satisfies completeness and an offline dataset of n samples with sufficient coverage, the gold-standard regret bound is $\tilde{\mathcal{O}}\left(\sqrt{\textsc{Comp}(\mathcal{F})/n}\right)$[4] where $\textsc{Comp}(\mathcal{F})$ denotes the statistical complexity of $\mathcal{F}$. This is achieved, for example, by Fitted Q-Iteration (FQI) (Chen and Jiang, 2019). Clearly, one would like $\textsc{Comp}(\mathcal{F})$ be as small as possible to achieve a tighter bound.

[2] That is, conditions sufficient for supervised learning, like realizability, tend not to be sufficient on their own for offline RL.

[3] $\mathcal{F}$ is a model class meant to estimate Q-functions. It consists of functions mapping state-action pairs to value predictions. The Bellman operator applied to $f \in \mathcal{F}$ pointwise is defined as $\mathcal{T} f(x,a) = r(x,a) + \max_{a'} \mathbb{E}_{x'|x,a} f(x',a')$.

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

We consider the model selection problem where we are given an offline dataset of n samples and a nested sequence of M model classes $\mathcal{F}_1 \subseteq ... \subseteq \mathcal{F}_M$. We investigate the following question: *Can we achieve a model selection guarantee for offline RL with regret scaling with the complexity of the smallest complete model class?*

We present a novel and conceptually simple algorithm, MODBE, that achieves regret scaling with the complexity of the smallest class satisfying completeness without knowledge of this class *a priori*.

Theorem 1. *(informal version of Corollary 1) Given an offline dataset of n samples and nested model classes $\mathcal{F}_1 \subseteq ... \subseteq \mathcal{F}_M$,* MODBE *outputs $\hat{\pi}$ such that* ***Reg***$(\hat{\pi}) = \tilde{\mathcal{O}}\left(\sqrt{\textsc{Comp}(\mathcal{F}_{k_*})/n}\right)$ *where* $k_* = \min\{k \in [M] : \mathcal{F}_k \text{ is complete}\}$.

A guarantee of this nature is typically known as an *oracle inequality* since an oracle with knowledge of the "best" model class ahead of time could simply choose it. We remark that this oracle inequality is rate-optimal in $\textsc{Comp}(\mathcal{F}_{k_*})$ and n, showing that we do not have to sacrifice efficiency for adaptivity. This is in contrast to some other works in model selection for decision-making where this unfortunate efficiency-adaptivity trade-off has been observed (Foster et al., 2019; Pacchiano et al., 2020; Xie and Jiang, 2021). In Appendix A, we discuss how the nestedness condition is necessary.

We also provide a robustness result for model selection (Theorem 3): if no models are Bellman complete (that is, k_* does not exist), MODBE obtains $\textbf{Reg}(\hat{\pi}) \leq \tilde{\mathcal{O}}\left(\min_{k\in[M]} \sqrt{\xi_k + \textsc{Comp}(\mathcal{F}_k)/n}\right)$ where ξ_k is a measure of the *global* completeness error of $\mathcal{F}_k$.[5] Our results show that, while some model selection problems remain elusive without further assumptions, strong rate-optimal oracle inequalities are still possible under standard offline RL assumptions even without knowledge of the best classes in advance.

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 of order $\tilde{\mathcal{O}}(\textsc{Comp}(\mathcal{F}_{k_*})/n)$). To do this, MODBE iteratively compares the relative effectiveness of two candidate model classes 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: using easier two-sided bounds (e.g. from uniform deviation bounds on risk estimators) leads to a squared Bellman error rate of $\tilde{\mathcal{O}}\left(\sqrt{\textsc{Comp}(\mathcal{F}_{k_*})/n}\right)$, which translates to a slow $\tilde{\mathcal{O}}((\textsc{Comp}(\mathcal{F}_{k_*})/n)^{1/4})$ regret rate. Instead the one-sided generalization error allows us to ultimately obtain the optimal $\tilde{\mathcal{O}}\left(\sqrt{\textsc{Comp}(\mathcal{F}_{k_*})/n}\right)$ 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 base 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 a good model class.

1.2 Additional Closely Related Work

Several prior works have specifically set out to address the model selection problem from a theoretical perspective, as we do here. Lee et al. (2021b) 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. Their positive results, however, were limited only to linear model classes for contextual bandits; ours apply to sequential settings. An earlier work by Farahmand and Szepesvári (2011) had partially addressed our problem but made several restrictive assumptions such as a known generalization bound that underestimates the approximation error (which is generally unknown); our algorithm only relies on commonly known quantities. Another notable work is the BVFT algorithm of Xie and Jiang (2021). While initially designed for general policy optimization, BVFT can be applied to model selection (Zhang and Jiang, 2021) but it incurs a slow $1/n^{1/4}$ regret rate in theory (compared to our $1/n^{1/2}$) and requires a stronger data coverage assumption. One advantage of BVFT 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

[5] See Section 3.1 for a precise definition.

the stronger guarantees. Thus, we view the two algorithms as complementary. Jiang et al. (2015) studied abstraction selection between nested state abstractions of increasing granularity; however, this eschews problems specific to value function approximation setting. Hallak et al. (2013) studied a similar abstraction problem, giving only asymptotic guarantees. In Section 3.2, we will discuss in more detail why several seemingly natural approaches to model selection do not produce satisfactory results.

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 $\Delta(\mathcal{A})$.[6] 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 densities of π respectively at step h.

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 (Sutton and Barto, 2018)) 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 $Q^* = Q^{\pi^*}$.

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 simplicity, we will assume that the learner uses the same $\mathcal{F}$ for each timestep $h \in [H]$ but this is trivially extended. We assume that $0 \in \mathcal{F}$ and we always write $f_{H+1} = 0$. For any function $f \in \mathcal{X} \times \mathcal{A} \to [0,H]$, we define the argmax policy $\pi_f(x) = \operatorname{argmax}_{a\in\mathcal{A}} f(x,a)$. We will also write $f(x) = \max_{a\in\mathcal{A}} f(x,a)$.

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. 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 $\mu_h \in \Delta(\mathcal{X} \times \mathcal{A})$ and the data are independent across timesteps h. 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 $\mu_h(x,a) = P_h^{\pi^b}(x,a) = \pi_h^b(a|x) P_h^{\pi_b}(x)$.

For $f, g \in (\mathcal{X} \times \mathcal{A} \to \mathbb{R})$, we use the notation $\|f - g\|_{\mu_h}^2 = \mathbb{E}_{\mu_h}\left[(f(x,a) - g(x,a))^2\right]$. 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 (Munos and Szepesvári, 2008; Duan et al., 2021), we make a concentrability 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* $\sup_{h,x,a,\pi} \frac{P_h^\pi(x,a)}{\mu_h(x,a)} \leq \mathcal{C}(\mu)$.

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

Concentrability is a structural assumption and it is widely regarded as perhaps the most standard assumption when studying offline RL problems (Foster et al., 2021). We remark that recent theoretical works have striven to weaken this condition via pessimistic methods (Liu et al., 2020; Jin et al., 2021; Xie et al., 2021; Uehara and Sun, 2021). However, Theorem 2 of Lee et al. (2021b) shows that model selection bounds of this type are not possible even in contextual bandits and even though the single model class bounds are possible. As a result, we will not consider this refinement in the present paper.

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 π^* and $\hat{\pi}$:

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

The following variant of the performance difference lemma will be used throughout the paper. It shows that it is sufficient to control the squared Bellman error to bound regret.

Lemma 1 (Duan et al. (2021)). *For any $f_1, \ldots, f_H$, let $\pi := (\pi_{f_h})_{h\in[H]}$. Then, $\textbf{Reg}(\pi) \le 2\sqrt{\mathcal{C}(\mu)\sum_{h\in[H]} \|f_h - T_h^* f_{h+1}\|_{\mu_h}^2}$.*

3 Model Selection Objectives

In this section, we state our primary model selection objectives and discuss their significance as well as challenges associated with solving them.

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

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

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 $\mathcal{F}$ (Chen and Jiang, 2019). 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[7] but require more data to minimize the estimation error. Small classes face the opposite problem.

Definition 1. *A class $\mathcal{F}$ is complete if $\textsc{Approx}(\mathcal{F}) := \max_{h\in[H], f'\in\mathcal{F}} \min_{f\in\mathcal{F}} \|f - T_h^* f'\|_\mu^2 = 0$.*

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 common practice in both supervised learning and offline RL. For example, one often starts with an extremely large class $\mathcal{F}$ and then considers restrictions of $\mathcal{F}$ to an increasing sequence $\mathcal{F}_1 \subseteq \ldots \subseteq \mathcal{F}_M = \mathcal{F}$. In a linear setting, this could correspond to trying to find a subset of candidate features that are sufficient to solve the problem.

Since the approximation error is typically unknown *a priori*, we aim to design an algorithm capable of selecting a good class in a data-dependent manner. In particular, we would like to achieve *oracle inequalities* reflecting that we can compete with the performance of an oracle that has this knowledge in advance.

Our primary objective is to compete with the *minimally complete* model class.

Problem 1. *Let $k_* = \min\{k \in [M] : \mathcal{F}_k \text{ is complete}\}$. Find $\hat{\pi}$ with $\textbf{Reg}(\hat{\pi}) = \tilde{\mathcal{O}}(\sqrt{\mathcal{C}(\mu)\log(|\mathcal{F}_{k_*}|)/n})$.*

Here, $\mathcal{F}_{k_*}$ is the smallest class that satisfies completeness on the data distribution. Such oracle inequalities are common in model selection for online bandits and RL (Foster et al., 2019) – albeit they are generally not rate-optimal in that literature. In particular, Problem 1 states the regret bound

[7] In contrast to realizability, this intuition of monotonicity of $\textsc{Approx}(\mathcal{F})$ is not universally true for completeness. Adding functions to the class $\mathcal{F}$ might actually *increase* $\textsc{Approx}(\mathcal{F})$. However, it remains a useful heuristic. In Appendix A, we discuss how model selection in this setting is not possible without nestedness.

should achieve the same dependence on $\log|\mathcal{F}_{k_*}|$ and n, as would an optimal offline algorithm using a single class with $k=k_*$. In other words, we do not tolerate any worse dependence on either quantity such as $\tilde{\mathcal{O}}(1/n^{1/4})$ rates and other lower order terms.

We are also interested in a *robustness* when k_* may not exist, i.e. all $\mathcal{F}_k$ have some approximation error.

Problem 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* $\textbf{\textit{Reg}}(\hat{\pi}) = \tilde{\mathcal{O}}\left(\min_{k\in[M]}\left\{\sqrt{\mathcal{C}(\mu)\xi_k}+\sqrt{\mathcal{C}(\mu)\log(|\mathcal{F}_k|)/n}\right\}\right)$

Note that $\xi_k \geq \textsc{Approx}(\mathcal{F}_k)$ by definition. For the estimation error, however, the guarantee remains rate-optimal. We remark that a solution to one of the above problems does not immediately imply a solution to the other. For example, a class $\mathcal{F}_k$ may be complete, but ξ_k can still be large. Perhaps surprisingly, our proposed algorithm will be able to handle both problems *simultaneously* without knowledge of whether k_* exists, thus achieving the $\min$ of both oracle inequalities.

3.2 Limitations of Prior Approaches

We now review some of the core challenges involved in solving the above problems. There are a number of seemingly natural approaches to model selection in RL that are surprisingly unable to produce satisfactory results, at least off-the-shelf.

Adaptive offline policy evaluation The most natural approach, to which we have alluded in the introduction, is to first compute $\hat{\pi}_k$ with a base algorithm using function class $\mathcal{F}_k$, for each $k\in[M]$. Then, one can estimate $v(\hat{\pi}_k)$ using an off-the-shelf offline policy evaluation approach such as fitted Q-evaluation (Munos and Szepesvári, 2008; Duan et al., 2020), DICE methods (Nachum et al., 2019; Dai et al., 2020; Zhan et al., 2022), marginalized importance estimators (Xie et al., 2019), or doubly robust estimators (Jiang and Li, 2016; Thomas and Brunskill, 2016). Then one simply picks the $\hat{\pi}_k$ with the best estimated value. The main drawback of this approach is that nearly all of the above methods require selecting a model class to perform the estimation,[8] 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 Su et al. (2020), 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 Lee et al. (2021b) 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 $\mathcal{C}(\mu)$. Since these bounds are required by the adaptive estimator as inputs, we are yet again left with unknown hyperparameters to tune.

Bellman error estimators Recall we are focusing on base offline RL algorithms that attempt to minimize the squared Bellman error of objective. Therefore, 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. Consider, for example, FQI which iteratively minimizes the squared Bellman error:

$$\hat{f}_h = \operatorname*{argmin}_{f\in\mathcal{F}_k}\hat{\mathbb{E}}_{D_h}\left[\left(f(x,a)-r-\max_{a'}\hat{f}_{h+1}(x',a')\right)^2\right],$$

where we use $\hat{\mathbb{E}}_{D_h}$ to denote the empirical mean calculated with samples from the dataset D_h. Presumably, we could simply choose the model class $\mathcal{F}_k$ that has the smallest cumulative squared error. The main issue with this approach is the classic double-sampling problem (Baird, 1995; Duan et al., 2021): the standard estimator of the Bellman error is biased, as a result of using an empirical version of the Bellman operator T^*. By selecting based on this error function alone, we will end up favoring model classes that also induce low variance of the *regression targets*, given by $r+\hat{f}_{h+1}(x')$ at step h. This is because the expectation is given by:

$$\mathbb{E}_{\mu_h}\left[\left(\hat{f}_h(x,a)-r-\hat{f}_{h+1}(x')\right)^2\right] = \|\hat{f}_h-T^*\hat{f}_{h+1}\|_{\mu_h}^2+\mathbb{E}_{\mu_h}\left[\operatorname*{var}_{x'\sim\mathcal{P}(\cdot|x,a)}\left(\hat{f}_{h+1}(x')\right)\right].$$

In reality, we want to choose a class $\mathcal{F}_k$ to minimize only the first term on the right-hand side, summed over $h\in[H]$, following Lemma 1. However, the second term is generally unknown. One could assume

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

there is a sufficiently powerful class $\mathcal{G}$ such that $T^* f \in \mathcal{G}$ for all $f \in \mathcal{F}$ (Chang et al., 2022). But there remains a question of how to select the class $\mathcal{G}$ to trade off approximation error and estimation error, creating another unsolved model selection problem.

In the same vein, another approach we might consider is recent BVFT algorithm of Xie and Jiang (2021) to select among the f^k learned by the base algorithm. This solves the model selection problem but the guarantee of BVFT has a slow $O(1/n^{1/4})$ dependence and thus does not achieve either oracle inequality. It also, in theory, requires that a discretization parameter is set based on a concentrability coefficient stronger than $C(\mu)$, which is typically unknown. Follow up work has shown this can be chosen adaptively in practice (Zhang and Jiang, 2021).

Perhaps most conceptually related to our approach is past work which compares Bellman errors of finer-grained state abstraction functions on the Q-function computed on coarser-grain state abstraction (Jiang et al., 2015). This work provided bounds on the resulting policy performance of the selected abstraction in discrete state and action setting, where models are varying levels of state abstractions. However, this work and analysis critically depends on the discrete state and action setting: our work shows how a similar idea can be used in the value function approximation setting, with substantially different tools and analysis techniques.

Representation Learning Readers familiar with work in representation learning for RL (Agarwal et al., 2020) might observe that the problem vaguely resembles objectives for selecting feature representations for low rank MDPs such as Modi et al. (2021). 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 or they require stronger realizability assumptions. It would be interesting future work to better understand the relationship between these two problems.

4 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 first give an intuitive sketch of the approach and present the full algorithm in subsequent subsection. As a thought experiment, we will consider the case when $M = 2$ and a minimally complete class $\mathcal{F}_{k_*}$ exists.[9] We will also ignore logarithmic factors and H dependence for now. A key algorithmic idea is that we will first start optimistically by guessing that $k_* = 1$. Running a base algorithm like FQI with $\mathcal{F}_1$ on training data returns the functions $f_1, \ldots, f_H$, which, with high probability, satisfy

$$\textstyle\sum_h \|f_h - T_h^* f_{h+1}\|_\mu^2 = \tilde{\mathcal{O}}\left(\frac{\mathcal{C}(\mu)\log(|\mathcal{F}_1|)}{n}\right)$$

if k_* actually equals 1. Given these functions, we can pose a square loss regression problem where the regression targets (i.e., the "y's" of the regression problem) are given by the empirical Bellman updates using training data:

$$\hat{L}_h(g, f_{h+1}) = \tfrac{1}{n}\textstyle\sum_{(x,a,r,x') \in D_h} (g(x,a) - r - f_{h+1}(x'))^2.$$

Let $L_h(f,g) := \mathbb{E}_{\mu_h}\left[\hat{L}_h(f,g)\right]$. Solving this regression problem for each h over the class $\mathcal{F}_2$ will generate $g_1, \ldots, g_H \subseteq \mathcal{F}_2$. The key insight is that the sequences $(f_h)_h$ and $(g_h)_h$ are both trying to minimize the same empirical square loss function with the same regression targets: $r + f_{h+1}(x')$. Unlike the Bellman error estimators from the previous section that incur biases, the losses $L_h(f_h, f_{h+1})$ and $L_h(g_h, f_{h+1})$ are comparable and estimable from a validation set. By nestedness of $\mathcal{F}_1 \subseteq \mathcal{F}_2$, $\mathcal{F}_2$ cannot have more approximation error on this regression problem. Provided we can get a good estimate of generalization errors $L(f_h, f_{h+1})$ and $L(g_h, f_{h+1})$ with validation data, this naturally brings forth the following *generalization test*: if

$$L_h(g_h, f_{h+1}) < L(f_h, f_{h+1}) - \tilde{\mathcal{O}}\left(\frac{\log(|\mathcal{F}_1|)}{n}\right) \tag{3}$$

[9] While the algorithm requires minimal changes to extend beyond these constraints, there are some notable analytic challenges in the proof. For general M, we cannot guarantee the class returned will be the correct one always – it may be substantially smaller but with controllable approximation error. When k_* does not exist, there is a chance to "skip" the best model class, so we must show that this is tolerable.

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 $\mathcal{F}_1 \subseteq ... \subseteq \mathcal{F}_M$, failure probability $\delta \leq 1/e$, and estimation error function ω for $\mathcal{B}$.
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 $\boldsymbol{\zeta} := \frac{96H^2 \log(16M^2H/\delta)}{n_{\text{valid}}}$
4: Initialize $k \leftarrow 1$.
5: **while** $k < M$ **do**
6: $(f_h)_{h\in[H]} \leftarrow \mathcal{B}(D_{\text{train}}, \mathcal{F}_k, \delta/4M)$
7: **for** $k' \leftarrow k+1, ..., M$ **do**
8: Set $\boldsymbol{\alpha} := \max\left\{\omega_{n_{\text{train}},\delta/4M}(\mathcal{F}_{k'}), \frac{200H^2\log(8M^2H|\mathcal{F}_{k'}|/\delta)}{n_{\text{train}}}\right\}$
9: Set TOL $:= 2\boldsymbol{\alpha} + 2\boldsymbol{\zeta} + \omega_{n_{\text{train}},\delta/4M}(\mathcal{F}_k)$
10: Minimize squared loss on training set for all $h \in [H]$ with regression targets from class k:

$$g_h \leftarrow \operatorname*{argmin}_{g\in\mathcal{F}_{k'}} \hat{L}_h(g, f_{h+1}) := \frac{1}{n_{\text{train}}} \sum_{(x,a,r,x')\in D_{\text{train},h}} \left(g(x,a) - r - f_{h+1}(x')\right)^2 \tag{4}$$

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

$$\tilde{L}_h(f, f_{h+1}) = \frac{1}{n_{\text{valid}}} \sum_{(x,a,r,x')\in D_{\text{valid},h}} \left(f(x_h,a_h) - r_h - f_{h+1}(x')\right)^2 \tag{5}$$

12: **if** $\tilde{L}_h(g_h, f_{h+1}) < \tilde{L}_h(f_h, f_{h+1}) - \text{TOL}$ for any $h \in [H]$ **then**
13: $k \leftarrow k+1$
14: goto Line 5.
15: **end if**
16: **end for**
17: goto Line 19
18: **end while**
19: **return** $\hat{\pi} = (\pi_{f_h})_{h\in[H]}$

reject $\mathcal{F}_1$ and pick $\mathcal{F}_2$. Otherwise pick $\mathcal{F}_1$. That is, a switch will occur not when $\mathcal{F}_2$ performs only marginally better than $\mathcal{F}_1$, but when it performs *substantially* better as measured by the generalization error that we see for both f_h and g_h on this regression problem. If (3) holds, then there is reason to believe that $\mathcal{F}_1$ is not complete, making $\mathcal{F}_2$ the right choice. Crucially, the test only checks for generalization error, so the tolerance term on the right side goes as $\tilde{\mathcal{O}}\left(\frac{\log(|\mathcal{F}_1|)}{n}\right)$, 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.

4.1 Full Algorithm

The full algorithm, MODBE (Model Selection via Bellman Error), is presented in Algorithm 1. While the underlying principle described just above is similar, MODBE must handle a number extensions that complicate the algorithm such as dealing with general M, accounting for proper estimation errors, and being robust to the case when k_* does not exist. Interestingly, the fundamental algorithmic idea remains the same – only the tolerances change and it loops over the model classes.

MODBE takes as input a base offline RL algorithm (such as FQI), the model classes $\mathcal{F}_1 \subseteq ... \subseteq \mathcal{F}_M$, and the offline dataset $D \in [H]$. 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 functions f. We retrain on the empirical square loss using a class $k' > k$ by regressing to target values $r + f_{h+1}(x')$. This amounts to solving a sequence of H least squares regression problems using class k', yielding the functions g_h.

Since f_h and g_h are attempting to solve the *same* regression problem (with the same target values), we can compare their performance on this shared squared loss objective $\tilde{L}$ with validation data. We use a *generalization error test* in Line 12 to decide whether to keep using class k. If the test fails and it is discovered that the larger model class $\mathcal{F}_{k'}$ is able to achieve substantially smaller loss than $\mathcal{F}_k$,

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

4.2 Rate-Optimal Oracle Inequalities

We show that this simple procedure is able to achieve both of the oracle inequalities of the previous section simultaneously. We start with a generic version of the theorem that is stated in terms of an assumed performance bound ω on the base algorithm. We will presently instantiate the base algorithm with FQI, showing that this version precisely achieves the desired oracle inequalities with the correct rates.

Definition 2. *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 $\beta > 0$ and a function ω, we say that $\mathcal{B}$ is (β,ω)-regular if (1) ω is a known real-valued function of $n \in \mathbb{N}$, $\delta \in \mathbb{R}$, and $\mathcal{F}_k$, and it 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 $(f_h)_{h\in[H]} \subseteq \mathcal{F}_k$ such that f_{h+1} is independent of D_h and*

$$P\left(\max_{h\in[H]} \|f_h - T_h^* f_{h+1}\|_{\mu_h}^2 \leq \beta \cdot \textsc{Approx}(\mathcal{F}_k) + \omega_{n,\delta}(\mathcal{F}_k)\right) \geq 1-\delta. \tag{6}$$

In this definition, β represents a multiplicative factor of error on the approximation error and ω represents the estimation error, which we expect to decrease in n and increase in the complexity of the class $\mathcal{F}$. Generally, we will have $\omega_{n,\delta}(\mathcal{F}) = \tilde{\mathcal{O}}(\log(|\mathcal{F}|/\delta)/n)$ (see Lemma 2 for FQI). For model selection, we thus hope to achieve a bound that matches what the base algorithm would achieve had k_* been known in advance, up to additive terms of $\sqrt{\frac{\log|\mathcal{F}_{k_*}|}{n}}$.

Our primary theorem addresses Problem 1 using an arbitrary base algorithm.

Theorem 2. *Let $\mathcal{B}$ be an (β,ω)-regular algorithm and suppose that k_* (defined in Problem 1) exists. Then Algorithm 1 with inputs D, $\mathcal{B}$, $\mathcal{F}_1 \subseteq \ldots \subseteq \mathcal{F}_M$, ω, and $\delta \leq 1/e$ outputs $\hat{\pi}$ such that, with probability at least $1-\delta$,*

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

for some absolute constant $C > 0$ and $\iota = \log(M^2 H/\delta)$.

The above theorem shows a regret bound scaling with the square root of the error term ω of the base algorithm $\mathcal{B}$ plus a $\tilde{\mathcal{O}}(\log(|\mathcal{F}_{k_*}|)/n)$ estimation error. Importantly, as stated in Problem 1, the statistical complexity depends only on $\mathcal{F}_{k_*}$ and not any of the larger classes.

For concreteness, we now instantiate Theorem 2 with a standard finite-horizon FQI (Duan et al., 2020) base algorithm, which satisfies Definition 2 with $\omega_n(\mathcal{F}) = \hat{\mathcal{O}}(\log|\mathcal{F}|/n)$. This in turn translates to the desired rate-optimal oracle inequalities.

Lemma 2. *Consider the FQI algorithm (stated in Appendix C for completeness). For a model class $\mathcal{F}$, FQI is a $(3,\omega)$-regular base algorithm with*

$$\omega_{n,\delta}(\mathcal{F}) = \mathcal{O}\left(\frac{H^2 \log(H|\mathcal{F}|/\delta)}{n}\right).$$

By plugging this classic result in Theorem 2 as the base algorithm, we arrive at a solution to Problem 1.

Corollary 1. *Let $\mathcal{B}$ be instantiated with FQI (Algorithm 3 in Appendix C). Define $\iota = \log(M^2 H/\delta)$ Then, under the same conditions as Theorem 2, there is an absolute constant $C > 0$ such that, with probability at least $1-\delta$, Algorithm 1 outputs $\hat{\pi}$ satisfying*

$$\textbf{\textit{Reg}}(\hat{\pi}) \leq C \cdot \sqrt{\frac{\mathcal{C}(\mu) H^3 (\log|\mathcal{F}_{k_*}|+\iota)}{n}}. \tag{8}$$

The proof of Theorem 2 (and by extension Corollary 1) follows a nearly identical intuition as outlined at the beginning of this section. In particular, the proof shows two parts: (1) MODBE will never return a value of k that exceeds k_* and (2) if MODBE returns $k < k_*$, then the approximation error must be small because it was undetectable by the test when comparing to k_*. however, a key novelty is recognizing that the generalization test in Line 12, which compares the errors of the two model classes on the same regression problem, can be used to prove both (1) and (2).

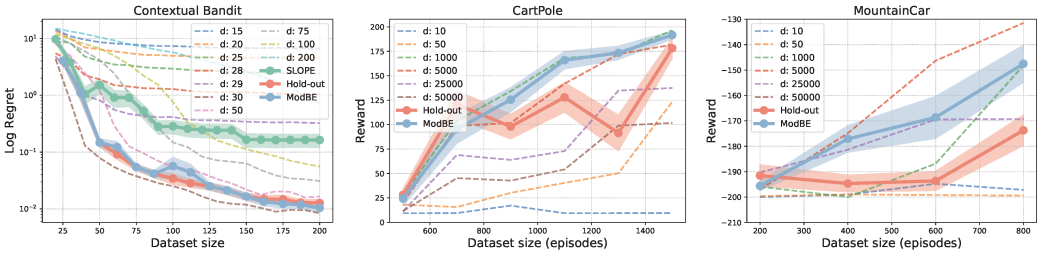

Figure 1: MODBE is evaluated on several simulated domains: a contextual bandit (left), CartPole (middle), and MountainCar (right). In CB, MODBE and Hold-out outperform SLOPE and match performance of the best model class in regret. In CartPole, both match the performance of the best model class. In MountainCar, both struggle to match the best model class, but MODBE maintains superior performance. In CB, error bands are standard error over 10 random trials. In RL, error bands are standard error over 20 random trials.

4.2.1 Robustness

We show that the same Algorithm 1 simultaneously achieves the desired robustness result of Problem 2 when k_* does not exist without any modification.

Theorem 3. *Under the same conditions as Theorem 2, if k_* does not exist, there exists an absolute constant $C>0$ such that, with probability at least $1-\delta$, Algorithm 1 outputs $\hat{\pi}$ satisfying*

$$\textbf{\textit{Reg}}(\hat{\pi}) \leq C \cdot \min_{k\in[M]} \left\{ \sqrt{\mathcal{C}(\mu) H \left(\beta \cdot \xi_k + \omega_{n_{train},\delta/4M}(\mathcal{F}_k) + \frac{H^2(\log|\mathcal{F}_k|+\iota)}{n} \right)} \right\}. \tag{9}$$

We can use Lemma 2 to see a solution to Problem 2 with an instantiation of FQI.

Corollary 2. *Under the same conditions as Corollary 1, there is an absolute constant $C>0$ such that, with probability at least $1-\delta$, Algorithm 1 outputs $\hat{\pi}$ satisfying*

$$\textbf{\textit{Reg}}(\hat{\pi}) \leq C \cdot \min_{k\in[M]} \left\{ \sqrt{\mathcal{C}(\mu) H \xi_k} + \sqrt{\frac{\mathcal{C}(\mu) H^3(\log|\mathcal{F}_k|+\iota)}{n}} \right\} \tag{10}$$

Crucially, the guarantees that solve Problems 1 and 2 are achieved simultaneously, meaning that we do not require knowledge of whether k_* exists and we can automatically get the best of both guarantees.

The proof of Theorem 3 (Corollary 2) is more involved than that of Theorem 2. Rather than showing that the k returned by MODBE never exceeds the index attaining the minimum, we allow k to exceed it sometimes. To ensure that the error can still be bounded, we use the fact that class $(k-1)$ must have failed the generalization test against some larger class in $[k,M]$. Using this fact, we can argue that the estimation error of the larger class can be bounded in terms of the unknown ξ_{k-1}, which we know is small since k exceeds the index of the minimal class. Like before, these arguments are made possible by the generalization test in Line 12.

Computational Complexity MODBE is computationally efficient given a squared loss regression oracle. Within inner and outer loops over the model classes, a squared loss minimizer is computed on the training dataset and then functions are evaluated on the validation set. MODBE requires only $\mathcal{O}(Hk_*M)$ calls to the computational oracle when k_* exists (a consequence of Theorem 2) or $\mathcal{O}(HM^2)$ in the worst case. Note that algorithms for optimizing squared loss regression problems are ubiquitous in machine learning (Simchi-Levi and Xu, 2021).

5 Empirical Results

The previous sections outlined the strong theoretical properties of MODBE. In this section, we ask: what practical insights can be gleaned from MODBE and its theoretical guarantees? 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 D 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 Lee et al. (2021b) 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 (Lee et al., 2021b) ends up being fooled by nearby classes.

RL Discrete Control Our experimental setup for the RL problems in Gym (Brockman et al., 2016) builds on top of the d3rlpy framework (Seno and Imai, 2021), which contains open-source implementations of offline RL algorithms. We used DQN (Mnih et al., 2015) (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 (Bartlett et al., 2019). For simplicity, we modified MODBE to work in the discounted infinite horizon setting, which can trivially be done (see Appendix D 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 it is the first to achieve rate-optimal oracle inequalities in n and $\textsc{Comp}(\mathcal{F}_{k_*})$. A number of interesting open questions remain. (1) Are there rate-optimal procedures that can be used to select hyperparameters beyond model complexity such as learning rates, batch sizes, *et cetera*? (2) Can the ideas of MODBE be extended to more general algorithms that do not rely on Bellman error minimization? (3) For the robustness guarantee, 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 $\mathcal{O}(\min_k \sqrt{\textsc{Approx}(\mathcal{F}_k)+\log|\mathcal{F}_k|/n})$ when k_* does not exist? We believe these questions are of great practical and theoretical importance for understanding how to effectively evaluate and select models in offline RL.

Acknowledgments and Disclosure of Funding

We thank Annie Xie and Yannis Flet-Berliac for help and advice with experiments and anonymous reviewers for their valuable feedback. JNL is supported by the NSF GRFP. This work was also supported in part by NSF Grant #2112926.

References

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.

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.

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

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

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

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

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.

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

Jonathan Chang, Kaiwen Wang, Nathan Kallus, and Wen Sun. Learning bellman complete representations for offline policy evaluation. In *International Conference on Machine Learning*, pages 2938–2971. PMLR, 2022.

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.

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

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.

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.

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.

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

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

Dylan J Foster, Akshay Krishnamurthy, David Simchi-Levi, and Yunzong Xu. Offline reinforcement learning: Fundamental barriers for value function approximation. *arXiv preprint arXiv:2111.10919*, 2021.

Assaf Hallak, Dotan Di-Castro, and Shie Mannor. Model selection in markovian processes. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 374–382, 2013.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 652–661. PMLR, 2016.

Nan Jiang, Alex Kulesza, and Satinder Singh. Abstraction selection in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 179–188. PMLR, 2015.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pages 5084–5096. PMLR, 2021.

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.

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

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, 2021a.

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

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.

Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. Provably good batch off-policy reinforcement learning without great exploration. *Advances in neural information processing systems*, 33:1264–1274, 2020.

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

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.

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

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.

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.

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.

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

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

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.

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.

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.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

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

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

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.

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.

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*, pages 2–35. PMLR, 2021.

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.

Masatoshi Uehara and Wen Sun. Pessimistic model-based offline reinforcement learning under partial coverage. *arXiv preprint arXiv:2107.06226*, 2021.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Ruosong Wang, Dean P Foster, and Sham M Kakade. What are the statistical limits of offline rl with linear function approximation? *arXiv preprint arXiv:2010.11895*, 2020.

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

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.

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*, 34:6683–6694, 2021.

Andrea Zanette. Exponential lower bounds for batch reinforcement learning: Batch rl can be exponentially harder than online rl. In *International Conference on Machine Learning*, pages 12287–12297. PMLR, 2021.

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.

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 D.

(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 D.

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]

A On the Nestedness of Model Classes

Throughout this work, we assume that the given model classes are nested in the sense that $\mathcal{F}_1 \subseteq ... \subseteq \mathcal{F}_M$. While this is a very common problem setting in both supervised learning and reinforcement learning, it can cause theoretical issues with completeness since adding functions to a model class can actually increase in the completeness error unlike realizability-based assumptions. Despite this, the following proposition shows that no type of model selection bound is possible without this nestedness in general.

Proposition 1. *There exists a family of MDPs, a collection of model classes $\mathcal{F}_1$ and $\mathcal{F}_2$, and a data distribution $\mu = \mu_1, ..., \mu_H$ with $\mathcal{C}(\mu) = \Theta(1)$ such that (1) either $\mathcal{F}_1$ or $\mathcal{F}_2$ complete for any MDP in the family and $|\mathcal{F}_i| = \Theta(1)$ for $i \in \{1,2\}$, and (2) any algorithm that outputs $\hat{\pi}$ with $v(\pi^*) - \mathbb{E}[v(\hat{\pi})] \leq C_1$ must use at least $C_2 S^{1/3}$ samples where $0 < C_2$ and $0 < C_1 < 1$ are constants.*

Proof. The proof is a simple consequence of a recent impossibility result from Foster et al. (2021) for problems with concentrability but without completeness. Theirs is summarized as follows:

Let $H = 3$. Let $\mathcal{F}_1 = (\mathcal{F}_1^h)_{h \in [H]}$ and $\mathcal{F}_2 = (\mathcal{F}_2^h)_{h \in [H]}$ be time-varying model classes such that each contains a single function, f_1^h and f_2^h respectively for each h. The result of Foster et al. (2021) shows that, defining $\mathcal{F}^h = \mathcal{F}_1^h \cup \mathcal{F}_2^h$, there are a family of MDP models $\mathcal{M}$ and functions f_1^h and f_2^h such that (1) the value function Q_h^π of any policy π is realized in $\mathcal{F}^h$ for all h and any algorithm that outputs $\hat{\pi}$ with $v(\pi^*) - \mathbb{E}[v(\hat{\pi})] \leq C_1$ must use at least $C_2 S^{1/3}$ samples for constants C_1, C_2.

Thus, for any MDP model in $\mathcal{M}$, either $\mathcal{F}_1$ or $\mathcal{F}_2$ satisfies completeness by the realizability condition above. A model selection oracle inequality should then ideally yield $v(\pi^*) - \mathbb{E}[v(\hat{\pi})] = \tilde{\mathcal{O}}(\sqrt{1/n})$ since $|\mathcal{F}_1^h| = 1$ and $|\mathcal{F}_2^h| = 1$. However, this would contradiction the lower bound result of Foster et al. (2021), which requires at least $\Omega(S^{1/3})$ samples to achieve constant error. □

Though the argument is simple, we remark on its significance. Our model selection objectives outlined in Section 3.1 suggest that we should aim to achieve $v(\pi^*) - \mathbb{E}[v(\hat{\pi})] = \tilde{\mathcal{O}}\left(\sqrt{\frac{\mathcal{C}(\mu) \log \mathcal{F}_{k_*}}{n}}\right)$ for any of the MDPs in the family, where n is the number of samples in the dataset and k_* is the index of the class that is complete for the given MDP. This is because it is guaranteed in the first condition that at least one of the classes is complete and realizes Q^*.

However, the proposition shows that we will need at least poly(S) samples to achieve any non-trivial regret bound, precluding the model selection objective since S can be much larger than $\log \mathcal{F}_{k_*}$. The proposition also ensures that $|\mathcal{F}_i| = \Theta(1)$ (in fact the size of both in the proof is simply $|\mathcal{F}_i| = 1$), which is to say that the hardness is not due to the inherent complexity of the model classes.

Note that BVFT (Xie and Jiang, 2021) does not contradict this hardness result for the same reason that it does not contradict the result of Foster et al. (2021): BVFT leverages a stronger coverage assumption.

B Proof of Theorem 2

B.1 Proof Sketch

Having outlined the intuition behind the algorithm and generalization error test in Section 4, we will now sketch the proof in a simplified setting so that the primary mechanism can be seen in a slightly more formal way. We restrict the sketch to the setting where $M = 2$ and $k_* \in \{1,2\}$ exists. For such a setting the definition of $\textsc{Tol}_n(\mathcal{F}_1, \mathcal{F}_2)$ will be excessively large, but sufficient nonetheless to prove our desired oracle inequality. We define the following quantities:

$$L_h(f,g) := \mathbb{E}_{\mu_h}(f(x,a) - r - g(x'))^2$$
$$L_h^*(g) := \inf_f L_h(f,g)$$

where the inf in the second line is over all measurable functions. Note that this makes $L_h^*(g)$ the *irreducible* error of the regression problem which is actually achieved by $f = T^* g$. For the purposes of exposition, we will take $n_{\text{train}} = n$ and assume that the validation error $\tilde{L}_h(f,g)$ exactly equals its expectation $L_h(f,g) := \mathbb{E}_{\mu_h}(f(x,a) - r - g(x'))^2$. For the sketch only, we will ignore dependence

on H and we will also assume that all necessary concentration inequalities hold with high probability[10]. That is, the base algorithm returns functions $f^k = (f_h^k)$ such that

$$\|f_h^k - T_h^* f_{h+1}^k\|_{\mu_h}^2 \le L \times \textsc{Approx}(\mathcal{F}_k) + \omega_{n,\delta}(\mathcal{F}_k) \tag{11}$$

and the empirical minimizers $g^2 = (g_h^2)$ in Algorithm 1 satisfy

$$L_h(g_h^2, f_{h+1}^1) - L_h^*(f_{h+1}^1) \le \tilde{\mathcal{O}}\left(\textsc{Approx}(\mathcal{F}_2) + \frac{\log|\mathcal{F}_2|}{n}\right)$$

We will now break the analysis down into cases:

- If it happens that $k_* = 1$ then we will show that the test will not fail and the correct class $k_* = 1$ will always be returned by Algorithm 1. Note that in this case $\textsc{Approx}(\mathcal{F}_1) = 0$ by definition of k_*. Thus, (11) implies that for all $h \in [H]$:

  $$L_h(f_h^1, f_{h+1}^1) - L_h^*(f_{h+1}^1) = \|f_h^1 - T_h^* f_{h+1}^1\|_{\mu_h}^2 \le \omega_{n,\delta}(\mathcal{F}_1)$$

  Then, an algorithm that reliably picks $k_* = 1$ should be able to *tolerate* generalization error on the order of at least $\omega_{n,\delta}(\mathcal{F}_1)$. This motivates our definition of $\textsc{Tol}_n(\mathcal{F}_1, \mathcal{F}_2)$ in Line 9 of Algorithm 1, which ensures that $\textsc{Tol}_n(\mathcal{F}_1, \mathcal{F}_2) \ge \omega_{n,\delta}(\mathcal{F}_1)$[11]. Then, when the algorithm reaches the generalization test in Line 12, it will compare the error in $L_h(f_h^1, f_{h+1}^1) - L_h^*(f_{h+1}^1)$ to the error in $L_h(g_h^2, f_{h+1}^1) - L_h^*(f_{h+1}^1)$. Since $\textsc{Tol}_n(\mathcal{F}_1, \mathcal{F}_2) \ge \omega_{n,\delta}(\mathcal{F}_1)$ and $L_h(f_h^1, f_{h+1}^1) - L_h^*(f_{h+1}^1) \le \omega_{n,\delta}(\mathcal{F}_1)$ (because $k_* = 1$), we will always have that

  $$L_h(g_h^2, f_{h+1}^1) - L_h^*(f_{h+1}^1) \ge L_h(f_h^1, f_{h+1}^1) - L_h^*(f_{h+1}^1) - \textsc{Tol}_n(\mathcal{F}_1, \mathcal{F}_2)$$

  And, by adding $L_h^*(f_{h+1}^1)$ to both sides of the above display, we see that the generalization test in Line 12 will never fail:

  $$L_h(g_h^2, f_{h+1}^1) \ge L_h(f_h^1, f_{h+1}^1) - \textsc{Tol}_n(\mathcal{F}_1, \mathcal{F}_2)$$

  meaning that Algorithm 1 will never make the switch from $k = 1$ to $k = 2$ when $k_* = 1$, so the correct model class is returned and the error is then trivially bounded as

  $$\|f_h^1 - T_h^* f_{h+1}^1\|_{\mu_h}^2 \le \omega_{n,\delta}(\mathcal{F}_1) \tag{12}$$

- If $k_* = 2$ and the switch to $k = 2$ is made, then the correct model class is returned and we immediately have the error bound $\|f_h^2 - T_h^* f_{h+1}^2\|_{\mu_h}^2 \le \omega_{n,\delta}(\mathcal{F}_2)$.
- However, if $k_* = 2$ and the switch is not made (meaning Algorithm 1 returns $k = 1$), we can show that the error cannot be much worse than the error of k_*. To do this, we will use the fact that the generalization test in Line 12 has (wrongly) succeeded in order to bound the error of $L_h(f_h^1, f_{h+1}^1) - L_h^*(f_{h+1}^1)$ in terms of the error of $L_h(g_h^2, f_{h+1}^1) - L_h^*(f_{h+1}^1)$ plus additional terms due to the tolerance: That is, for any $h \in [H]$

  $$\begin{aligned}\|f_h^1 - T_h^* f_{h+1}^1\|_{\mu_h}^2 &= L_h(f_h^1, f_{h+1}^1) - L_h^*(f_{h+1}^1) \\ &\le L_h(g_h^2, f_{h+1}^1) - L_h^*(f_{h+1}^1) + \textsc{Tol}_n(\mathcal{F}_1, \mathcal{F}_2)\end{aligned} \tag{13}$$

  Recall that since $k_* = 2$, there is no approximation error for this class so $L_h(g_h^2, f_{h+1}^1) - L_h^*(f_{h+1}^1) \le \tilde{\mathcal{O}}\left(\frac{\log|\mathcal{F}_2|}{n}\right)$. Finally, we can apply the definition of $\textsc{Tol}_n(\mathcal{F}_1, \mathcal{F}_2)$ as well as monotonicity of $\omega_n(\mathcal{F}_1) \le \omega_n(\mathcal{F}_2)$ so that (13) can further be bounded as

  $$\|f_h^1 - T_h^* f_{h+1}^1\|_{\mu_h}^2 \le \tilde{\mathcal{O}}\left(\frac{\log|\mathcal{F}_2|}{n} + \omega_{n,\delta}(\mathcal{F}_2)\right) \tag{14}$$

  Therefore, since $\mathcal{F}_1$ did not fail the generalization test, we can actually use this to our advantage to say that its error is not much worse than $\mathcal{F}_2$ even though $k_* = 2$.

[10] Eventually, in the main proof, care will have to be taken to ensure these events to occur with high probability at the expense of logarithmic factors.

[11] Factors on δ (due to union bounds to handle the high probability events) are omitted in the sketch for clarity.

Algorithm 2 Model Selection via Bellman Error (MODBE) with indexing notation

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 $\delta\leq 1/e$.
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 $h\in[H]$.
Set $\zeta:=\frac{96H^2\log(16M^2H/\delta)}{n_{\text{valid}}}$.
Initialize $k\leftarrow 1$.
while $k<M$ **do**
  $f^k:=(f_h^k)_{h\in[H]}\leftarrow\mathcal{B}(D_{\text{train}},\mathcal{F}_k,\delta/4M)$
  for $k'\leftarrow k+1,\ldots,M$ **do**
    Set $\boldsymbol{\alpha}_k:=\max\left\{\omega_{n_{\text{train}},\delta/4M}(\mathcal{F}_k),\frac{200H^2\log(8M^2H|\mathcal{F}_k|/\delta)}{n_{\text{train}}}\right\}$ for all $k\in[M]$
    Set $\text{TOL}_{n_{\text{train}}}(\mathcal{F}_k,\mathcal{F}_{k'}):=2\delta_{k'}+2\zeta+\omega_{n_{\text{train}},\delta/4M}(\mathcal{F}_k)$ for all $k<k'$.
    Minimize squared loss on training set for all $h\in[H]$ with regression targets from class k:

$$g_h^{k'}\leftarrow\operatorname*{argmin}_{g\in\mathcal{F}_{k'}}\ \hat{L}_h(g,f_{h+1}^k):=\frac{1}{n_{\text{train}}}\sum_{(x,a,r,x')\in D_{\text{train},h}}\left(g(x,a)-r-f_{h+1}^k(x')\right)^2 \tag{15}$$

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

$$\tilde{L}_h(f,f_{h+1}^k)=\frac{1}{n_{\text{valid}}}\sum_{(x,a,r,x')\in D_{\text{valid},h}}\left(f(x_h,a_h)-r_h-f_{h+1}^k(x')\right)^2 \tag{16}$$

    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**
      $k\leftarrow k+1$
      goto Line 5.
    end if
  end for
  goto Line 19
end while
return $\hat{\pi}=\left(\pi_{f_h^k}\right)_{h\in[H]}$

By combining the results of the case when $k_*=1$ (where we showed (12) holds) and the case when $k_*=2$ (where we showed that (14) holds), we have managed to show that the index k returned by Algorithm 1 will satisfy

$$\|f_h^k-T^*f_{h+1}^k\|_{\mu_h}^2=\tilde{\mathcal{O}}\left(\frac{\log|\mathcal{F}_{k_*}|}{n}+\omega_{n,\delta}(\mathcal{F}_{k_*})\right)$$

Appealing to the performance difference lemma (Lemma 1), we are able to guarantee that

$$\textbf{Reg}(\hat{\pi})=\tilde{\mathcal{O}}\left(\sqrt{\mathcal{C}(\mu)\left(\frac{\log|\mathcal{F}_{k_*}|}{n}+\omega_{n,\delta}(\mathcal{F}_{k_*})\right)}\right)$$

B.2 Concentration Inequalities

We now turn to the formal proof of Theorem 2. In order to make the analysis easier, we state another version of the algorithm, which is more notation-heavy but also more precise so that we can easily refer objects in the analysis at different indices. To be clear, the algorithms are identical – the notation has just been augmented to include indices and other modifiers for clarity.

We require several basic components in order for the final model selection bound to hold. The first few are concentration results concerning the datasets. These will allow us to prove generalization error bounds for each of the classes as well as to obtain good estimates of the regression error via the validation set.

Recall some useful shorthand notation to represent the true and empirical loss functions. For any measurable functions $f,g \in (\mathcal{X} \times \mathcal{A} \to \mathbb{R})$ and a training dataset $D = (D_h)_{h \in [H]}$ of n samples for each h and a validation dataset $D' = (D'_h)_{h \in [H]}$ of m samples for each h, we define

$$L_h(f,g) = \mathbb{E}_{\mu_h}(f(x,a) - r - g(x'))^2 \tag{17}$$

$$L_h^*(g) = \inf_f L_h(f,g) \tag{18}$$

$$\hat{L}_h(f,g) = \frac{1}{n} \sum_{(x,a,r,x') \in D_h} \left(f(x_{i,h},a_{i,h}) - r(x_{i,h},a_{i,h}) - g(x'_{i,h})\right)^2 \tag{19}$$

$$\tilde{L}_h(f,g) = \frac{1}{m} \sum_{(x,a,r,x',) \in D'_h} (f(x,a) - r - g(x'))^2 \tag{20}$$

where, as in the proof sketch, the $\inf$ in the second line is also over all measurable functions. Finally, recall that, for any functions $f,g \in (\mathcal{X} \times \mathcal{A} \to \mathbb{R})$, we have defined

$$\|f - T^* g\|_{\mu_h}^2 := \mathbb{E}_{\mu_h}(f(x,a) - T^* g(x,a))^2$$

It is easy to see that this is equal to $L_h(f,g)$ without the irreducible erorr: $\|f - T^* g\|_{\mu_h}^2 = L_h(f,g) - L_h^*(g)$.

To proceed with the concentration analysis, we will show that all the necessary events will hold simultaneously with high probability. This requires defining some additional notation.

We let $f^k = (f_h^k)_{h \in [H]} \leftarrow \mathcal{B}(D, \mathcal{F}_k)$ for each $k \in [M]$. Then, for each $h \in [H]$, we define the empirical minimizers for a larger class k' with the same regression target as follows:

$$g_h^{k' \to k} = \operatorname*{argmin}_{g \in \mathcal{F}_{k'}} \hat{L}_h(g, f_{h+1}^k)$$

for all $k' \geq k$. Not all of these need be computed in the execution of Algorithm 1, but we will analyze them all for the sake of simplicity in the concentration analysis. We define the following event that guarantees all of these value function approximators achieve their desired errors simultaneously up to log factors.

$$\mathcal{E}_1 = \bigcap_{k \in [M], k' > k, h \in [H]} \left\{ \|g_h^{k' \to k} - T_h^* f_{h+1}^k\|_{\mu_h}^2 \leq 3\textsc{Approx}(\mathcal{F}_{k'}) + \frac{C_1 H^2 \log(8M^2 H |\mathcal{F}_{k'}|/\delta)}{n} \right\}$$

where $C_1 > 0$ is a constant to be determined. The next event ensures that the base algorithm actually achieves its guarantees from Definition 2.

$$\mathcal{E}_2 = \bigcap_{k \in [M], h \in [H]} \left\{ \|f_h^k - T_h^* f_{h+1}^k\|_\mu^2 \leq \beta \cdot \textsc{Approx}(\mathcal{F}_k) + w_{n,\delta/4M}(\mathcal{F}_k) \right\}$$

The above $\mathcal{E}_2$ occurs with probability at least $1 - \frac{\delta}{4}$ essentially by definition of the base algorithm. The last event that we are interested in relates the true loss L_h to the validation loss $\tilde{L}_h$ on the independent dataset D'. We define $\tilde{L}_h^*(f_{h+1}^k) := \tilde{L}_h(T_h^* f_{h+1}^k, f_{h+1}^k)$. The events are given by

$$\mathcal{E}_3 = \bigcap_{k \in [M], k' > k, h \in [H]} \left\{ \tilde{L}_h(g_h^{k' \to k}, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k) \leq 2\left(L_h(g_h^{k' \to k}, f_{h+1}^k) - L_h^*(f_{h+1}^k)\right) + \frac{C_3 H^2 \log(16HM^2/\delta)}{m} \right\}$$

$$\bigcap \bigcap_{k \in [M], h \in [H]} \left\{ \tilde{L}_h(f_h^k, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k) \leq 2\left(L_h(f_h^k, f_{h+1}^k) - L_h^*(f_{h+1}^k)\right) + \frac{C_3 H^2 \log(16HM^2/\delta)}{m} \right\}$$

$$\mathcal{E}_4 = \bigcap_{k \in [M], k' > k, h \in [H]} \left\{ L_h(g_h^{k' \to k}, f_{h+1}^k) - L_h^*(f_{h+1}^k) \leq 2\left(\tilde{L}_h(g_h^{k' \to k}, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k)\right) + \frac{C_4 H^2 \log(16HM^2/\delta)}{m} \right\}$$

$$\bigcap \bigcap_{k \in [M], h \in [H]} \left\{ L_h(f_h^k, f_{h+1}^k) - L_h^*(f_{h+1}^k) \leq 2\left(\tilde{L}_h(f_h^k, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k)\right) + \frac{C_4 H^2 \log(16HM^2/\delta)}{m} \right\}$$

where $C_3, C_4 > 0$ are constants to be determined. We will prove the following guarantee.

Theorem 4. *Let $\mathcal{E} = \bigcap_{i=1}^{4} \mathcal{E}_i$. Then, $P(\mathcal{E}) \geq 1-\delta$.*

To prove this result, we will show that these events occur with high probability. We require several intermediate results, starting simply with Bernstein's inequality.

Lemma 3 (Bernstein's Inequality)**.** *Let $Z_1, \ldots, Z_n$ be a sequence of independent random variables with $\mathbb{E}[Z_i] = 0$, $\sigma^2 = \mathrm{var}(Z_i)$ and $|Z_i| \leq B$. Then, with probability at least $1-\delta$, for any $\eta > 0$*

$$\begin{aligned}|\sum_{i\in[n]} Z_i| &\leq \sqrt{2n\sigma^2 \log(2/\delta)} + B\log(2/\delta) \\ &\leq \frac{n\sigma^2}{2\eta} + B\log(2/\delta) + \eta\log(2/\delta)\end{aligned}$$

Proof. The first inequality is a standard Bernstein inequality found in, for example, Vershynin (2018). The second inequality follows by applying the AM-GM inequality to the first. □

The next lemma is a generalization error bound showing that the Bellman error of a function $f \in \mathcal{F}$ can be bounded in terms of the excess training loss, the approximation error of $\mathcal{F}$, and the estimation error which is $\tilde{\mathcal{O}}(\log|\mathcal{F}|/n)$. The next lemma shows that the minimizer of the empirical squared loss achieves good generalization error with respect to the optimal function in its class.

Lemma 4. *Fix $\mathcal{F} \subseteq (\mathcal{X} \times \mathcal{A} \to [0, H])$ and h. Let $g \in \mathcal{G}$ be fixed where $\mathcal{G} \subseteq \mathcal{F}$. For $i \in [n]$ and $(x_i, a_i) \sim \mu_h$ and $x_i' \sim \mathbb{P}(\cdot|x, a)$, define $y_i = r(x_i, a_i) + g(x_i')$. Define $Z_i^f = (f(x_i,a_i) - y_i)^2 - (f^*(x_i,a_i) - y_i)^2$ where $f^* = \mathrm{argmin}_{f\in\mathcal{F}} \|f - T_h^* g\|_{\mu_h}^2$. Then, with probability at least $1-\delta$, for all $f \in \mathcal{F}$ simultaneously,*

$$\|f - T_h^* g\|_{\mu_h}^2 \leq \frac{2}{n}\sum_i Z_i^f + 3\|f^* - T_h^* g\|_{\mu_h}^2 + \frac{40(H+1)^2 \log(2|\mathcal{F}|/\delta)}{n}$$

Proof. We will drop some of the sub- and super-script notation with the understanding that $\mathbb{E}$ means $\mathbb{E}_{\mu_h}$ and Z_i means Z_i^f. It is easy to see that $\mathbb{E}[Z_i] = L(f,g) - L(f^*,g) = \|f - T^* g\|_{\mu_h}^2 - \|f^* - T^* g\|_{\mu_h}^2$. Furthermore, we can bound the variance as

$$\begin{aligned}\mathrm{var}(Z_i) &= \mathbb{E}[Z_i^2] - \mathbb{E}[Z_i]^2 \\ &\leq \mathbb{E}Z_i^2 \\ &= \mathbb{E}\Big((f(x_i,a_i) - y_i)^2 - (f^*(x_i,a_i) - y_i)^2\Big)^2 \\ &= \mathbb{E}\Big((f(x_i,a_i) - f^*(x_i,a_i))^2 + 2(f(x_i,a_i) - f^*(x_i,a_i))(f^*(x_i,a_i) - y_i)\Big)^2 \\ &= \mathbb{E}(f(x_i,a_i) - f^*(x_i,a_i))^2 (f(x_i,a_i) + f^*(x_i,a_i) - 2y_i)^2 \\ &\leq 4(H+1)^2 \|f - f^*\|_{\mu_h}^2 \\ &\leq 8(H+1)^2 \big(\|f - T^* g\|_{\mu_h}^2 + \|f^* - T^* g\|_{\mu_h}^2\big)\end{aligned}$$

where we have used the fact that $f(\cdot,\cdot), f(\cdot,\cdot) \in [0, H]$, $y_i \in [0, H+1]$, and $(a+b)^2 \leq 2a^2 + 2b^2$ for $a, b \in \mathbb{R}$. Using Lemma 3, we have that with probability at least $1-\delta$,

$$\begin{aligned}\|f - T^* g\|_{\mu_h}^2 - \|f^* - T^* g\|_{\mu_h}^2 &= \mathbb{E}[Z_i] \\ &\leq \frac{1}{n}\sum_i Z_i + \frac{\mathrm{var}(Z_1)}{\eta} + \frac{4(H+1)^2\log(2/\delta)}{n} + \frac{\eta\log(2/\delta)}{n} \\ &\leq \frac{1}{n}\sum_i Z_i + \frac{8(H+1)^2\big(\|f - T^* g\|_{\mu_h}^2 + \|f^* - T^* g\|_{\mu_h}^2\big)}{\eta} \\ &\quad + \frac{4(H+1)^2\log(2/\delta)}{n} + \frac{\eta\log(2/\delta)}{n} \\ &\leq \frac{1}{n}\sum_i Z_i + \frac{\|f - T^* g\|_{\mu_h}^2 + \|f^* - T^* g\|_{\mu_h}^2}{2} + \frac{20(H+1)^2\log(2/\delta)}{n}\end{aligned}$$

where in the last equality we have chosen $\eta = 16(H+1)^2$. Rearranging and then taking the union bound over all $f \in \mathcal{F}$ gives the result. □

Note that if we take $f = \hat{f}_h$ to be the empirical minimizer of $\hat{L}_h(\cdot, g)$, then the bound in Lemma 4 becomes

$$\begin{aligned}\|\hat{f}_h - T_h^* g\|_{\mu_h}^2 &\leq 3\|f^* - T_h^* g\|_{\mu_h}^2 + \frac{40(H+1)^2 \log(2|\mathcal{F}|/\delta)}{n} \\ &\leq 3\textsc{Approx}(\mathcal{F}) + \frac{40(H+1)^2 \log(2|\mathcal{F}|/\delta)}{n}\end{aligned}$$

where the last inequality follows because $\mathcal{G} \subseteq \mathcal{F}$. Equipped with these bounds, we are now ready to prove that event $\mathcal{E}_1$ holds with good probability.

Proposition 2. *$P(\mathcal{E}_1) \geq 1 - \frac{\delta}{4}$ with the constant $C = 200$.*

Proof. The proof follows by repeatedly applying Lemma 4. Note that $\hat{f}_{h+1}^k$ is independent of the data D_h. Therefore, for any h we may condition on f_{h+1}^k and see that

$$\|g_h^{k' \to k} - T^* f_{h+1}^k\|_{\mu_h}^2 \leq 3\textsc{Approx}(\mathcal{F}_{k'}) + \frac{40(H+1)^2 \log(2|\mathcal{F}|/\delta)}{n} \tag{21}$$

with probability at least $1 - \delta$. By this independence, integrating ensures that the above holds regardless of f_{h+1}^k. Taking the union bound over all $h \in [H]$, all $k \in [M]$ and all $k' > k$, we get that (21) holds for all with probability at least $1 - M^2 H \delta$. Changing variables to $\delta' = 4M^2 H \delta$ completes the proof. □

Proposition 3. *$P(\mathcal{E}_2) \geq 1 - \frac{\delta}{4}$.*

Proof. This follows immediately from Definition 2 and a union bound and changing variables $\delta' = 4M\delta$. □

Proposition 4. *$P(\mathcal{E}_3 \cap \mathcal{E}_4) \geq 1 - \frac{\delta}{2}$ with $C_3 = C_4 = 96$.*

Proof. Fix a single tuple (k, k', h). For shorthand, let us define $f := f_{h+1}^k$, $g := g_h^{k' \to k}$ and $g^* := T_h^* f_{h+1}^k$. Then, similar to the proof of Lemma 4, we define $y_i = r_i + f(x_i')$ and $Z_i = (g(x_i, a_i) - y_i)^2 - (g^*(x_i, a_i) - y_i)^2$.

Note that $\mathbb{E}_{\mu_h}[Z_i] = L(g, f) - L^*(f) = \|g - T_h^* f\|_{\mu_h}^2$. Similarly, $\mathrm{var}(Z_i) \leq \mathbb{E}[Z_i^2]$ where

$$\begin{aligned}\mathbb{E}[Z_i^2] &= \mathbb{E}(g(x_i, a_i) - g^*(x_i, a_i))^2 (g(x_i, a_i) + g^*(x_i, a_i) - 2y_i)^2 \\ &= 4(H+1)^2 \mathbb{E}(g(x_i, a_i) - g^*(x_i, a_i))^2 \\ &\leq 4(H+1)^2 \|g - g^*\|_{\mu_h}^2 \\ &= 4(H+1)^2 (L_h(g, f) - L_h^*(f))\end{aligned}$$

By Lemma 3, we can guarantee that

$$\begin{aligned}|\left(\tilde{L}_h(g, f) - \tilde{L}_h(g^*, f)\right) - (L_h(g, f) - L_h^*(f))| &\leq \frac{4(H+1)^2 (L_h(g, f) - L_h^*(f))}{\eta} \\ &\quad + \frac{4(H+1)^2 \log(2/\delta)}{m} + \frac{\eta \log(2/\delta)}{m} \\ &= \frac{(L_h(g, f) - L_h^*(f))}{2} + \frac{12(H+1)^2 \log(2/\delta)}{m}\end{aligned}$$

with probability at least $1 - \delta$. Rearranging terms, we are able to conclude that

$$\tilde{L}_h(g, f) - \tilde{L}_h(g^*, f) \leq \frac{3(L_h(g, f) - L_h^*(f))}{2} + \frac{12(H+1)^2 \log(2/\delta)}{m}$$

and, simultaneously,

$$L_h(g, f) - L_h^*(f) \leq 2\left(\tilde{L}_h(g, f) - \tilde{L}_h(g^*, f)\right) + \frac{24(H+1)^2 \log(2/\delta)}{m}$$

We may repeat the same calculation when setting $g = f_h^k$ for all $k \in [M]$ and $h \in [H]$. Taking the union bound over all (k, k', h) and changing variables to $\delta' = 4(M^2 H + MH)\delta$ gives the result. □

Proof of Theorem 4. The result follows immediately by a union bound combining the events $\mathcal{E}_1$, $\mathcal{E}_2$, and $\mathcal{E}_3$ and $\mathcal{E}_4$, where it was shown that $P(\mathcal{E}_1) \geq 1-\frac{\delta}{4}$, $P(\mathcal{E}_2) \geq 1-\frac{\delta}{4}$ and $P(\mathcal{E}_3 \cap \mathcal{E}_4) \geq 1-\frac{\delta}{2}$. □

B.3 Proof of Theorem 2

Armed with the concentration results of the previous section, we are ready to prove Theorem 2, which is restated here for clarity.

Theorem 2. *Let $\mathcal{B}$ be an (β,ω)-regular algorithm and suppose that k_* (defined in Problem 1) exists. Then Algorithm 1 with inputs D, $\mathcal{B}$, $\mathcal{F}_1 \subseteq \ldots \subseteq \mathcal{F}_M$, ω, and $\delta \leq 1/e$ outputs $\hat{\pi}$ such that, with probability at least $1-\delta$,*

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

for some absolute constant $C > 0$ and $\iota = \log(M^2 H/\delta)$.

Proof. Let us assume the event $\mathcal{E}$ holds using the training dataset D_{train} of n_{train} samples and validation dataset D_{valid} of n_{valid} samples. Theorem 4 shows that $P(\mathcal{E}) \geq 1-\delta$. Recall that the training set size is n_{train} and the validation set size is n_{valid}. As shorthand, Algorithm 2 also defines the following quantities:

$$\boldsymbol{\alpha}_k = \max\left\{ \omega_{n_{\text{train}},\delta/4M}(\mathcal{F}_k), \frac{C_1 H^2 \log(8M^2H|\mathcal{F}_k|/\delta)}{n_{\text{train}}} \right\}$$
$$\boldsymbol{\zeta} = \frac{C_3 H^2 \log(16M^2H/\delta)}{n_{\text{valid}}}$$

where C_1 and C_3 are the constants from Propositions 2 and 4.

Note that $\boldsymbol{\alpha}_k$ is still monotonically non-decreasing in k as both sequences that comprise it are monotonically non-decreasing. Recall the definition of $\textsc{Tol}_{n_{\text{train}}}(\mathcal{F}_k,\mathcal{F}_{k'})$:

$$\textsc{Tol}_{n_{\text{train}}}(\mathcal{F}_k,\mathcal{F}_{k'}) := 2\omega_{n_{\text{train}},\delta/4M}(\mathcal{F}_k) + 2\boldsymbol{\zeta} + \boldsymbol{\alpha}_{k'}$$

We will drop the subscript notation on ω and $\textsc{Tol}$ with the implicit understanding that $\omega(\cdot) = \omega_{n_{\text{train}},\delta/4M}(\cdot)$ and $\textsc{Tol}(\cdot,\cdot) = \textsc{Tol}_{n_{\text{train}}}(\cdot,\cdot)$.

We will prove the oracle inequality of Theorem 2 when k_* exists (second claim of Theorem 2). Consider the following cases.

1. Suppose that algorithm has currently reached $k = k_*$. We can guarantee that the generalization test in Line 12 will never fail in this situation, and, therefore, the algorithm will return $k = k_*$ which achieves the desired oracle inequality by definition. Note that by $\mathcal{E}$, for all $h \in [H]$ and $k' > k$,

$$\begin{aligned}
\tilde{L}_h(f_h^k, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k) - \textsc{Tol}(\mathcal{F}_k,\mathcal{F}_{k'}) &\leq 2L_h(f_h^k, f_{h+1}^k) - 2L_h^*(f_{h+1}^k) + \boldsymbol{\zeta} - \textsc{Tol}(\mathcal{F}_k,\mathcal{F}_{k'}) \\
&= 2\|f_h^k - f_{h+1}^k\|_{\mu_h}^2 + \boldsymbol{\zeta} - \textsc{Tol}(\mathcal{F}_k,\mathcal{F}_{k'}) \\
&\leq 2\omega(\mathcal{F}_k) + \boldsymbol{\zeta} - \textsc{Tol}(\mathcal{F}_k,\mathcal{F}_{k'}) \\
&= -\boldsymbol{\alpha}_{k'} - \boldsymbol{\zeta} \\
&\leq -\boldsymbol{\zeta}
\end{aligned}$$

where the second inequality has used $\mathcal{E}_2$ along with the fact that $\textsc{Approx}(\mathcal{F}_k) = 0$ in this case. Similarly, we have that

$$\begin{aligned}
0 &\leq \frac{1}{2}\left(L_h(g_h^{k' \to k}, f_{h+1}^k) - L_h^*(f_{h+1}^k) \right) \\
&\leq \tilde{L}_h(g_h^{k' \to k}, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k) + \boldsymbol{\zeta}
\end{aligned}$$

The above inequalities imply that we will always find that

$$\begin{aligned}
\tilde{L}_h(f_h^k, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k) - \textsc{Tol}(\mathcal{F}_k,\mathcal{F}_{k'}) &\leq -\boldsymbol{\zeta} \\
&\leq \tilde{L}_h(g_h^{k' \to k}, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k)
\end{aligned}$$

and therefore

$$\tilde{L}_h(f_h^k, f_{h+1}^k) - \textsc{Tol}(\mathcal{F}_k, \mathcal{F}_{k'}) \le \tilde{L}_h(g_h^{k' \to k}, f_{h+1}^k)$$

Therefore, the test will never fail when $k = k_*$ while $\mathcal{E}$ holds.

2. Now let us consider the case where Algorithm 1 returns $k < k_*$. In this case, the test succeeded for all $k' > k$ even though class $\mathcal{F}_k$ has $\textsc{Approx}(\mathcal{F}_K) \neq 0$. It remains to show that little is lost in this case even though there is approximation error in the returned class. Note that this implies that the test succeeded for $k' = k_*$. Therefore, we have

$$\tilde{L}_h(f_h^k, f_{h+1}^k) - \textsc{Tol}(\mathcal{F}_k, \mathcal{F}_{k'}) \le \tilde{L}_h(g_h^{k_* \to k}, f_{h+1}^k)$$

for all $h \in [H]$. Then, event $\mathcal{E}$ implies that

$$\begin{aligned}
\|f_h^k - T_h^* f_{h+1}^k\|_{\mu_h}^2 &= L_h(f_h^k, f_{h+1}^k) - L_h^*(f_{h+1}^k) \\
&\le 2\Big(\tilde{L}_h(f_h^k, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k)\Big) + \boldsymbol{\zeta} \\
&\le 2\Big(\tilde{L}_h(g_h^{k_* \to k}, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k)\Big) + \boldsymbol{\zeta} + 2\textsc{Tol}(\mathcal{F}_k, \mathcal{F}_{k_*}) \\
&\le 4\Big(L_h(g_h^{k_* \to k}, f_{h+1}^k) - L_h^*(f_{h+1}^k)\Big) + 2\boldsymbol{\zeta} + \boldsymbol{\zeta} + 2\textsc{Tol}(\mathcal{F}_k, \mathcal{F}_{k_*}) \\
&\le 8\boldsymbol{\alpha}_{k_*} + 7\boldsymbol{\zeta} + 2\omega(\mathcal{F}_k) \\
&\le 8\boldsymbol{\alpha}_{k_*} + 7\boldsymbol{\zeta} + 2\omega(\mathcal{F}_{k_*})
\end{aligned}$$

where the second to last line follows from applying $\mathcal{E}_1$ along with the fact that $f_{h+1}^k \in \mathcal{F}_k \subseteq \mathcal{F}_{k_*}$ and the last line uses the monotonicity property $\omega(\mathcal{F}_k) \le \omega(\mathcal{F}_{k_*})$ since $k < k_*$ by assumption.

Since all the cases have been handled, we see that we are able to guarantee that, for all $h \in [H]$

$$\|f_h^k - T_h^* f_{h+1}^k\|_{\mu_h}^2 \le 8\boldsymbol{\alpha}_{k_*} + 7\boldsymbol{\zeta} + 2\omega(\mathcal{F}_{k_*})$$

Appealing to the performance difference lemma, the regret can be bounded as

$$\textbf{Reg}(\hat{\pi}) \le 2\sqrt{\mathcal{C}(\mu) H (8\boldsymbol{\alpha}_{k_*} + 7\boldsymbol{\zeta} + 2\omega(\mathcal{F}_{k_*}))}$$

This completes the proof of the second claim of Theorem 2 when k_* exists. □

B.4 Proof of Theorem 3

Theorem 3. *Under the same conditions as Theorem 2, if k_* does not exist, there exists an absolute constant $C > 0$ such that, with probability at least $1 - \delta$, Algorithm 1 outputs $\hat{\pi}$ satisfying*

$$\textbf{\textit{Reg}}(\hat{\pi}) \le C \cdot \min_{k \in [M]} \left\{ \sqrt{\mathcal{C}(\mu) H \left(\beta \cdot \xi_k + \omega_{n_{\textit{train}}, \delta/4M}(\mathcal{F}_k) + \frac{H^2(\log|\mathcal{F}_k| + \iota)}{n} \right)} \right\}. \tag{9}$$

Proof. Now consider the case where k_* does not necessarily exist. This setting is slightly more challenging as we must tolerate the case where Algorithm 1 outputs k that is too large; whereas, in the previous case, we showed that such an event could never occur. Let us denote $k^\dagger = \operatorname{argmin}_{k \in [M]} \{\xi_k + \omega_{n_{\text{train}}, \delta/4M}(\mathcal{F}_k)\}$.

1. If the algorithm returns $k = k^\dagger$, then we are done.

2. Consider the case where $\mathcal{F}_k$ is returned with $k < k^\dagger$. Then, since the test has succeeded with $k^\dagger$, we have that for all $h \in [H]$

$$\begin{aligned}
\tilde{L}_h(f_h^k, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k) &\le \tilde{L}_h(g_h^{k^\dagger \to k}, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k) + \textsc{Tol}(\mathcal{F}_k, \mathcal{F}_{k^\dagger}) \\
&\le 2\Big(L_h(g_h^{k^\dagger \to k}, f_{h+1}^k) - L_h^*(f_{h+1}^k)\Big) + \boldsymbol{\zeta} + \textsc{Tol}(\mathcal{F}_k, \mathcal{F}_{k^\dagger}) \\
&\le 2(3\textsc{Approx}(\mathcal{F}_{k^\dagger}) + \delta_{k^\dagger}) + \boldsymbol{\zeta} + \textsc{Tol}(\mathcal{F}_k, \mathcal{F}_{k^\dagger})
\end{aligned}$$

Furthermore,

$$\begin{aligned} L_h(f_h^k, f_{h+1}^k) - L_h^*(f_{h+1}^k) &\le 2\Big(\tilde{L}_h(f_h^k, f_{h+1}^k) - \tilde{L}_h^*(f_{h+1}^k)\Big) + \boldsymbol{\zeta} \\ &\le 12\textsc{Approx}(\mathcal{F}_{k^\dagger}) + 4\boldsymbol{\alpha}_{k^\dagger} + 3\boldsymbol{\zeta} + 2\textsc{Tol}(\mathcal{F}_k, \mathcal{F}_{k^\dagger}) \\ &\le 12\textsc{Approx}(\mathcal{F}_{k^\dagger}) + 8\boldsymbol{\alpha}_{k^\dagger} + 7\boldsymbol{\zeta} + 2\omega(\mathcal{F}_k) \\ &\le 12\textsc{Approx}(\mathcal{F}_{k^\dagger}) + 8\boldsymbol{\alpha}_{k^\dagger} + 7\boldsymbol{\zeta} + 2\omega(\mathcal{F}_{k^\dagger}) \end{aligned}$$

3. Finally, we consider the last case where $\mathcal{F}_k$ is returned for $k > k^\dagger$. This implies that for $i = k-1$ there is some $j \in [k, M]$ and $h \in [H]$ such that the test failed. That is,

$$\begin{aligned} \tilde{L}_h(g_h^{j\to i}, f_{h+1}^i) - \tilde{L}_h^*(f_{h+1}^i) &\le \tilde{L}_h(f_h^i, f_{h+1}^i) - \tilde{L}_h^*(f_{h+1}^i) - \textsc{Tol}(\mathcal{F}_i, \mathcal{F}_j) \\ &\le 2\big(L_h(f_h^i, f_{h+1}^i) - L_h^*(f_{h+1}^i)\big) + \boldsymbol{\zeta} - \textsc{Tol}(\mathcal{F}_i, \mathcal{F}_j) \\ &\le 2(\beta \cdot \textsc{Approx}(\mathcal{F}_i) + \omega(\mathcal{F}_i)) + \boldsymbol{\zeta} - \textsc{Tol}(\mathcal{F}_i, \mathcal{F}_j) \end{aligned}$$

where the last line uses event $\mathcal{E}_2$ from the base algorithm guarantee. Further lower bounding the left side, we get that

$$\begin{aligned} 0 &\le \frac{1}{2}\Big(L_h(g_h^{j\to i}, f_{h+1}^i) - L_h^*(f_{h+1}^i)\Big) \\ &\le \tilde{L}_h(g_h^{j\to i}, f_{h+1}^i) - \tilde{L}_h^*(f_{h+1}^i) + \boldsymbol{\zeta} \\ &\le 2(\beta \cdot \textsc{Approx}(\mathcal{F}_i) + \omega(\mathcal{F}_i)) + 2\boldsymbol{\zeta} - \textsc{Tol}(\mathcal{F}_i, \mathcal{F}_j) \end{aligned} \tag{22}$$

Plugging in our value for $\textsc{Tol}_{n_{\text{train}}}(\mathcal{F}_i, \mathcal{F}_j)$ and rearranging, we are able to conclude from (22) that

$$\delta_j \le 2\beta\textsc{Approx}(\mathcal{F}_i) \le 2\beta\xi_i$$

Therefore, $\boldsymbol{\alpha}_k \le \boldsymbol{\alpha}_j \le 2L\xi_i \le 2L\xi_{k^\dagger}$ by the monotone property of both sequences $(\boldsymbol{\alpha}_k)$ and (ξ_k). Finally using $\mathcal{E}_2$ again, this implies that for all $h \in [H]$

$$\begin{aligned} \|f_h^k - T_h^* f_{h+1}^k\|_{\mu_h}^2 &\le L\textsc{Approx}(\mathcal{F}_k) + \omega(\mathcal{F}_k) \\ &\le \beta \cdot \textsc{Approx}(\mathcal{F}_k) + \delta_k \\ &\le \beta \cdot \textsc{Approx}(\mathcal{F}_k) + 2\beta \cdot \xi_{k^\dagger} \\ &\le 3\beta \cdot \xi_{k^\dagger} \end{aligned}$$

Observing the bounds from both cases, we are then able to conclude that for whatever k is returned by Algorithm 1, we have the bound

$$\max_{h\in[H]} \|f_h^k - T_h^* f_{h+1}^k\|_{\mu_h}^2 \le 12(\beta\xi_{k^\dagger} + \boldsymbol{\alpha}_{k^\dagger} + \boldsymbol{\zeta} + \omega(\mathcal{F}_{k^\dagger}))$$

Again, the performance difference lemma ensures that

$$\textbf{Reg}(\hat{\pi}) \le 2\sqrt{12\mathcal{C}(\mu)H(\beta\xi_{k^\dagger} + \boldsymbol{\alpha}_{k^\dagger} + \boldsymbol{\zeta} + \omega(\mathcal{F}_{k^\dagger}))}.$$

We finally conclude by using the fact that $\boldsymbol{\alpha}_k \lesssim \omega(\mathcal{F}_k) + \frac{H^2\log(M^2H|\mathcal{F}_k|/\delta)}{n_{\text{train}}}$ for all $k \in [M]$ and that n_{train} and n_{valid} are constant fractions of n.

□

C FQI Algorithm and Guarantees

Here we state and then prove a more detailed version of the FQI guarantee that was originally stated in Lemma 2.

Lemma 5. *Consider the FQI algorithm (stated in Appendix C for completeness). For a model class $\mathcal{F}$, FQI is a $(3,\omega)$-regular base algorithm with $\omega_{n,\delta}(\mathcal{F}) = \frac{200H^2\log(16H|\mathcal{F}|/\delta)}{n}$.*

Algorithm 3 Fitted Q-Iteration

Input: Offline dataset $D = (D_h)$ of n samples for each $h \in [H]$ and model class $\mathcal{F}$
Initialize $f_{H+1} = 0 \in \mathcal{F}$
for $h = H, \ldots, 1$ **do**

$$f_h \leftarrow \operatorname*{argmin}_{f \in \mathcal{F}} \frac{1}{n} \sum_{(x,a,r,x') \in D_h} \left(f(x,a) - r - f_{h+1}(x')\right)^2$$

end for
return $(f_h)_{h \in [H]}$

Proof. This result can be obtained almost immediately from Lemma 4 in the case where the model classes are the same. Observe that D_h is independent of f_{h+1}. Therefore, conditioned on f_{h+1}, we have that

$$\|f_h - T_h^* f_{h+1}\|_{\mu_h}^2 \le 3\textsc{Approx}(\mathcal{F}) + \frac{40(H+1)^2 \log(2|\mathcal{F}|/\delta)}{n}$$

with probability at least $1-\delta$ since f_h is the empirical minimizer. Integrating out the conditioning, taking the union bound over $h \in [H]$, and changing variables to $\delta' = H\delta$ yields the result. □

We may now apply this result to immediately Corollaries 1 and 2.

Corollary 1. *Let $\mathcal{B}$ be instantiated with FQI (Algorithm 3 in Appendix C). Define $\iota = \log(M^2 H/\delta)$ Then, under the same conditions as Theorem 2, there is an absolute constant $C > 0$ such that, with probability at least $1-\delta$, Algorithm 1 outputs $\hat{\pi}$ satisfying*

$$\textbf{\textit{Reg}}(\hat{\pi}) \le C \cdot \sqrt{\frac{\mathcal{C}(\mu) H^3 (\log|\mathcal{F}_{k_*}| + \iota)}{n}}. \tag{8}$$

Proof of Corollaries 1 and 2. We start with Corollary 2 Recall that Theorem 3 ensures that for an (β, ω)-regular algorithm in the case where k_* does not exist, we have

$$\textbf{Reg}(\hat{\pi}) \le C_0 \cdot \min_{k \in [M]} \left\{ \sqrt{\mathcal{C}(\mu) H \left(\beta \xi_k + \omega_{n_{\text{train}}, \delta/4M}(\mathcal{F}_k) + \frac{H^2(\log|\mathcal{F}_k| + \iota)}{n} \right)} \right\} \tag{23}$$

with probability at least $1-\delta$ for some absolute constant $C_0 > 0$ and $\iota = \log(M^2 H/\delta)$.

Using Lemma 2, we may substitute in the values of ω and $L = 3$ to achieve

$$\begin{aligned}
\textbf{Reg}(\hat{\pi}) &\le C_0 \cdot \min_{k \in [M]} \left\{ \sqrt{\mathcal{C}(\mu) H \left(3\xi_k + \frac{200 H^2 \log(64M|\mathcal{F}_k|/\delta)}{n_{\text{train}}} + \frac{H^2(\log|\mathcal{F}_k| + \iota)}{n} \right)} \right\} \\
&\le C_0' \cdot \min_{k \in [M]} \left\{ \sqrt{\mathcal{C}(\mu) H \left(\xi_k + \frac{H^2 \log(M|\mathcal{F}_k|/\delta)}{n} + \frac{H^2(\log|\mathcal{F}_k| + \iota)}{n} \right)} \right\} \\
&\le C_0'' \cdot \min_{k \in [M]} \left\{ \sqrt{\mathcal{C}(\mu) H \xi_k} + \sqrt{\frac{\mathcal{C}(\mu) H^3 (\log|\mathcal{F}_k| + \iota)}{n}} \right\}
\end{aligned}$$

where $C_0', C_0'' > 0$ are absolute constants. In the second line, we have used the fact n_{train} and n_{valid} are constant fractions of n. In the third line, we have used $\sqrt{a+b} \le \sqrt{a} + \sqrt{b}$ for $a, b \ge 0$.

For Corollary 1, when k_* exists, the proof is essentially identical, except that we use Theorem 2:

$$\begin{aligned}\textbf{Reg}(\hat{\pi}) &\leq C_1 \cdot \sqrt{\mathcal{C}(\mu) H \left(\omega_{n_{\text{train}},\delta/4M}(\mathcal{F}_{k_*}) + \frac{H^2(\log|\mathcal{F}_{k_*}|+\iota)}{n} \right)} \\ &\leq C_1' \cdot \sqrt{\mathcal{C}(\mu) H \left(\frac{200H^2\log(64HM|\mathcal{F}_{k_*}|/\delta)}{n_{\text{train}}} + \frac{H^2(\log|\mathcal{F}_{k_*}|+\iota)}{n} \right)} \\ &\leq C_1'' \cdot \sqrt{\frac{\mathcal{C}(\mu) H^3(\log|\mathcal{F}_{k_*}|+\iota)}{n}}\end{aligned}$$

where $C_1, C_1', C_1'' > 0$ are all absolute constants. □

D Experiment Details

D.1 Practical Implementation of MODBE for the RL Setting

MODBE, as stated in Algorithm 1, is originally designed for the finite horizon case in which there are H functions comprising the value function approximators. For the contextual bandit setting (where $H = 1$), we make no modifications. In an effort to further increase the computational and statistical efficiency of MODBE in the RL setting (as well as to demonstrate that its primary principles are fairly robust), we opted for a discounted infinite horizon implementation with discount factor $\gamma = 0.99$ (default for d3rlpy).

We use a single fixed dataset D (not split into timesteps) and fed this to the Deep Q-Network (DQN) implementation of Seno and Imai (2021) using all the default hyperparameters except for the network architecture, which was specific to each model class as described in Section 5. For consistency, we set the number of epochs to 20 across all model classes and experiments for DQN. This generates value function approximators $f^1, \ldots, f^M$. To implement a close approximation of Algorithm 1 in discounted case, considered the following procedure. While the algorithm is on model class k, we compute empirical risk minimizers for $k' \geq k$ so that

$$g^{k'} \leftarrow \operatorname*{argmin}_{g \in \mathcal{F}_{k'}} \frac{1}{n_{\text{train}}} \sum_{(x,a,r,x') \in D_{\text{train}}} (g(x,a) - r - \gamma f^k(x'))^2$$

We then decide whether to switch to $k+1$ by using the generalization test:

$$\tilde{L}(g^{k'}, f^k) \geq \tilde{L}(g^k, f^k) - \text{TOL}_{n_{\text{train}}}(\mathcal{F}_k, \mathcal{F}_{k'})$$

where the functional $\tilde{L}(\cdot, f^k)$ is the estimated loss on the validation data, as before:

$$\tilde{L}(g, f^k) = \frac{1}{n_{\text{valid}}} \sum_{(x,a,r,x') \in D_{\text{valid}}} (g(x,a) - r - \gamma f^k(x'))^2$$

As noted in Section 5, we did not find it necessary to tune the any parameters related to $\text{TOL}_{n_{\text{train}}}(\mathcal{F}_k, \mathcal{F}_{k'})$ and simply set it to $\frac{d_{k'}}{n}$ where d_k is the dimension of the linar model (for the contextual bandit setting) or the number of hidden nodes in the neural network (for the RL settings), which roughly (up to constants and logarithmic factors) matches known bounds on the pseudo-dimension (Bartlett et al., 2019). The lack of necessity to actually make TOL theoretically valid is actually a positive of the algorithm: it shows it is fairly robust in practice and simply matching the order appears to be good enough to generate the current results. To fit the empirical risk minimizers in the CB setting, we simply used ridge regression as in Lee et al. (2021b). To do the same in the RL setting, we trained neural networks of with the same architectures as the DQNs in d3rlpy (state inputs and one output per action to predict the value). We used an Adam optimizer with on 10 epochs with a learning rate of 4e-3 and a batch size of 64. This was implemented through PyTorch (Paszke et al., 2019).

One might ask whether it is possible to extend this beyond neural networks with one hidden layer. In practice one can easily use any model, but, in theory, some care may need to be taken in order to set the value of $\text{TOL}_{n_{\text{train}}}$. For example, to handle more hidden layers, we can appeal to generalized pseudo-dimension bounds (Bartlett et al., 2019). As observed in the current experiments, the setting of $\text{TOL}_{n_{\text{train}}}$ to rough estimates does not seem to make a huge impact on the results.

Hold-out baseline The hold-out method as a model selection baseline was implemented by choosing k that minimizes $\tilde{L}(f^k, f^k)$ in the RL setting. In the contextual bandit setting it is equivalent to selecting k to minimize $\tilde{L}(f^k,0)$, since there is only one step.

D.2 Experimental Setups

We now describe the specific experimental setup so that it may be reproduced. In order to generate the plots which vary based on the sample size of D, we simply curtailed the dataset to the given amount of samples shown on the x-axis. Generation of the datasets varied in each domain. It would be interesting in the future to evaluate performance on more stochastic RL environments (the CB evnironment is stochastic) as these are ones we expect toe Hold-out method to do very poorly on. Despite this, our current experiments show it is already sub-optimal even in deterministic settings.

Contextual Bandit We replicated almost exactly the study of Lee et al. (2021b). To recap their study, there is a linear contextual bandit with $|\mathcal{A}| = 10$ and an infinite state space where the linear feature vectors of ambient dimension $d = 200$ for each action are generated by sampling from normal distributions with different covariance matrices. The reward function is generated by taking the inner product of θ_* with feature vector for action $a \in \mathcal{A}$. To make this an interesting model selection problem, only the first $d_* = 30$ coordinates are non-zero (although this is not known to the learner) and thus a model class using only the first d_* coordinates is sufficient to solve the problem without any approximation error. The individual model classes were generated by simply truncating the coordinates of the feature vectors to the following sizes $\{15,20,25,28,29,30,50,75,100,200\}$. The base algorithm was Algorithm 1 of Lee et al. (2021b).

One difference is that we included several additional model classes to the $d_* = 30$ model that are close enough to *fool* the SLOPE algorithm used in Lee et al. (2021b). This also involved increasing the ambient dimension from $d = 100$ to $d = 200$, but we kept $d_* = 30$. We suspect that this poor performance of SLOPE is due to the fact that SLOPE is heavily dependent on the known deviation bounds whereas MODBE seems to be comparatively robust. The results of SLOPE seem to be poor whenever the deviation bounds are invalid or too conservative.

CartPole We used the default dataset from d3rlpy (Seno and Imai, 2021) which contains approximately 1500 episodes of a good (but not optimal) behavior policy on the CartPole domain. Everything else remains the same as the standard CartPole environment in Gym (Brockman et al., 2016).

MountainCar Since no default dataset for MountainCar is provided in d3rlpy, we generated our own through the following procedure. First, we trained a policy online via SARSA on the discretized environment to achieve good performance on the task. We then collected the offline policy by executing 1000 episodes under the good policy which also took a random action at any time step with probability 0.3 to induce some coverage on the dataset. To simplify the problem for the base DQN algorithm, we also replaced the sparse reward in the offline dataset with a more dense and informative reward function, giving bonuses for high speeds, proximity to the goal, and achieving the goal. We note that this change is done only to simplify the problem and help the base algorithm solve the task with limited computational resources and tuning so as to increase reproducibility. Everything else remains the same as the standard MountainCar environment in Gym (Brockman et al., 2016).

D.3 Hardware

Contextual bandit experiments were run on a standard personal laptop with 16 GB of memory and an Intel Core i7 processor. RL experiments were run on an internal cluster with 16 GB of memory and an NVIDIA GTX 1080 Ti GPU for PyTorch (Paszke et al., 2019).