Model Selection for Off-policy Evaluation: New Algorithms and Experimental Protocol

Anonymous Authors[1]

Abstract

Holdout validation and hyperparameter tuning from data is a long-standing problem in offline reinforcement learning (RL). A standard framework is to use off-policy evaluation (OPE) methods to evaluate and select between different policies, but OPE methods either incur exponential variance (e.g., importance sampling) or have hyperparameters of their own (e.g., FQE and model-based). We focus on model selection for OPE itself, which is even more under-investigated. Concretely, we select among candidate value functions ("model-free") or dynamics ("model-based") to best assess the performance of a target policy. Our contributions are two fold. We develop: (1) new model-free and model-based selectors with theoretical guarantees, and (2) a new experimental protocol for empirically evaluating them. Compared to the model-free protocol in prior works, our new protocol allows for more stable generation and better control of candidate value functions in an optimization-free manner, and evaluation of model-free and model-based methods alike. We exemplify the protocol on a Gym environment, and find that our new model-free selector, LSTD-Tournament, demonstrates promising empirical performance.

1. Introduction

Offline reinforcement learning (RL) is a promising paradigm for applying RL to important application domains where perfect simulators are not available and we must learn from data [LKTF20; JX24]. Despite the significant progress made in devising more performant *training* algorithms, how to perform holdout validation and model selection—an indispensable component of any practical machine learning pipeline—remains an open problem and has hindered the deployment of RL in real-life scenarios. Concretely, after multiple training algorithms (or instances of the same algorithm with different hyperparameter settings) have produced candidate policies, the *primary* task (which contrasts the *secondary* task which we focus on) is to select a good policy from these candidates, much like how we select a good classifier/regressor in supervised learning. To do so, we may estimate the performance (i.e., expected return) of each policy, and select the one with the highest estimated return.

Unfortunately, estimating the performance of a new *target* policy based on data collected from a different *behavior* policy is a highly challenging task, known as *off-policy evaluation* (OPE). Popular OPE algorithms can be roughly divided into two categories, each with their own critical weaknesses: the first is importance sampling [PSS00; JL16; TB16], which has elegant unbiasedness guarantees but suffers variance that is *exponential* in the horizon, limiting applicability beyond short-horizon settings such as contextual bandits [LCLW11]. The second category includes algorithms such as Fitted-Q Evaluation (FQE) [EGW05; LVY19; Pai+20], marginalized importance sampling [LLTZ18; NCDL19; UHJ20], and model-based approaches [VJY21], which avoid the exponential variance; unfortunately, this comes at the cost of introducing their own hyperparameters (choice of neural architecture, learning rates, etc.). While prior works have reported the effectiveness of these methods [Pai+20], they also leave a chicken-and-egg problem: **if these algorithms tune the hyperparameters of training, who tunes *their* hyperparameters?**

In this work, we make progress on this latter problem, namely model selection for OPE algorithms themselves, in multiple dimensions. Concretely, we consider two settings: in the **model-based** setting, evaluation algorithms build dynamics models to evaluate a target policy. Given the uncertainty of hyperparameters in model building, we assume that multiple candidate models are given, and the task is to select one that we believe evaluates the performance of the target policy most accurately. In the **model-free** setting, evaluation algorithms only output *value functions*. Similar to above, the task is to select a value function out of the candidate value functions.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

Our contributions are 4-fold:

1. **New selectors (model-free):** We propose a new selection algorithm (or simply *selector*), LSTD-Tournament, for selecting between candidate value functions by approximately checking whether the function satisfies the Bellman equation. The key technical difficulty here is the infamous ***double sampling*** problem [Bai95; SB18; CJ19]. Our derivation builds on BVFT [XJ21; ZJ21], which is the only existing selector that addresses double sampling in a theoretically rigorous manner without additional function-approximation assumptions. Our new selector relies on more plausible assumptions, enjoys better statistical rates ($1/\epsilon^2$ vs. $1/\epsilon^4$), and empirically outperforms BVFT and other baselines.
2. **New selectors (model-based):** When comparing candidate models, popular losses in model-based RL exhibit biases under stochastic transitions [Jia24]. Instead, we propose novel estimators with theoretical guarantees, including novel adaptation of previous model-free selectors that require additional assumptions to the model-based setting [ASM08; ZDMAK23].
3. **New experiment protocol**: To empirically evaluate the selection algorithms, prior works often use FQE to prepare candidate Q-functions [ZJ21; NFBJSB22], which suffers from unstable training[1] and lack of control in the quality of the candidate functions. We propose a new experiment protocol, where the candidate value functions are induced from variations of the groundtruth environment. This bypasses the caveats of FQE and allows for the computation of Q-values in an *optimization-free* and controllable manner. Moreover, the protocol can also be used to evaluate and compare estimators for the model-based setting. Implementation-wise, we use ***lazy evaluation*** and Monte-Carlo roll-outs to generate the needed Q-values. Combined with parallelization and caching, we reduce the computational cost and make the evaluation of new algorithms easier.
4. **Preliminary experiments**: We instantiate the protocol in Gym Hopper and demonstrate the various ways in which we can evaluate and understand different selectors.

2. Preliminaries

Markov Decision Process (MDP). An MDP is specified by $(\mathcal{S}, \mathcal{A}, P, R, \gamma, d_0)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition dynamics, $R : \mathcal{S} \times \mathcal{A} \to [0, R_{\max}]$ is the reward function, $\gamma \in [0, 1)$ is the discount factor, and d_0 is the initial state distribution. A policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ induces a distribution over random trajectories, generated as $s_0 \sim d_0$, $a_t \sim \pi(\cdot|s_t)$, $r_t = R(s_t, a_t)$, $s_{t+1} \sim P(\cdot|s_t, a_t)$, $\forall t \geq 1$. We use $\Pr_\pi[\cdot]$ and $\mathbb{E}_\pi[\cdot]$ to denote such a distribution and the expectation thereof. The performance of a policy is defined as $J(\pi) := \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t r_t]$, which is in the range of $[0, V_{\max}]$ where $V_{\max} := R_{\max}/(1-\gamma)$.

Value Function and Bellman Operator. The Q-function $Q^\pi \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ is the fixed point of $\mathcal{T}^\pi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$, i.e., $Q^\pi = \mathcal{T}^\pi Q^\pi$, where for any $f \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$, $(\mathcal{T}^\pi f)(s,a) := R(s,a) + \gamma \mathbb{E}_{s\sim P(\cdot|s,a)}[f(s', \pi)]$. We use the shorthand $f(s', \pi)$ for $\mathbb{E}_{a'\sim\pi(\cdot|s')}[f(s', a')]$.

Off-policy Evaluation (OPE). OPE is about estimating the performance of a given *target* policy π in the real environment denoted as $M^\star = (\mathcal{S}, \mathcal{A}, P^\star, R, \gamma, d_0)$, namely $J_{M^\star}(\pi)$, using an offline dataset $\mathcal{D}$ sampled from a behavior policy π_b. For simplicity, from now on we may drop the $M^\star$ in the subscript when referring to properties of $M^\star$, e.g., $J(\pi) \equiv J_{M^\star}(\pi)$, $Q^\pi \equiv Q^\pi_{M^\star}$, etc. As a standard simplification, our theoretical derivation assumes that the dataset $\mathcal{D}$ consists of n i.i.d. tuples (s, a, r, s') generated as $(s,a) \sim \mu$, $r = R(s,a)$, $s' \sim P^\star(\cdot|s,a)$. We use $\mathbb{E}_\mu[\cdot]$ to denote the true expectation under the data distribution, and $\mathbb{E}_\mathcal{D}[\cdot]$ denotes the empirical approximation from $\mathcal{D}$.

Model Selection. We assume that there are multiple OPE instances that estimate $J(\pi)$, and our goal is to choose among them based on the offline dataset $\mathcal{D}$. Our setup is agnostic w.r.t. the details of the OPE instances, and view them only through the intermediate objects (dynamics model or value function) they produce. Concretely, two settings are considered:

- **Model-based:** Each OPE instance produces an MDP M_i and uses $J_{M_i}(\pi)$ as an estimate of $J(\pi)$; w.l.o.g. we assume M_i only differs from $M^\star$ in the transition P_i. The task is to select $\hat{M}$ from $\mathcal{M} := \{M_i\}_{i\in[m]}$, such that $J_{\hat{M}}(\pi) \approx J(\pi)$. We assume that at least one model is close to $M^\star$, and in theoretical analyses we make the simplification that $M^\star \in \mathcal{M}$; extension to the misspecified case ($M^\star \notin \mathcal{M}$) is routine in RL theory [AJS23; ACK24] and orthogonal to the insights of this work.
- **Model-free:** Each OPE instance provides a Q-function that approximates Q^π. The validation task is to select $\hat{Q}$ from the candidate Q-functions $\mathcal{Q} := \{Q_i\}_{i\in[m]}$,[2] such that $\mathbb{E}_{s\sim d_0}[\hat{Q}(s, \pi)] \approx J(\pi)$. Similar to the model-based case, we will assume $Q^\pi \in \mathcal{Q}$ in the derivations.

The model-free setting is a more general protocol, as the model-based setting can be reduced to it: given candidate

[1]For example, our preliminary investigation has found that FQE often diverges with CQL-trained policies [KZTL20], which is echoed by [NFBJSB22] in personal communications.

[2]In practical scenarios, the candidate models and functions, $\{M_i\}_{i\in[m]}$ and $\{Q_i\}_{i\in[m]}$, may be learned from data, and we assume $\mathcal{D}$ is a holdout dataset independent of the data used for producing $\{M_i\}_{i\in[m]}$ and $\{Q_i\}_{i\in[m]}$.

models $\{M_1, \ldots, M_m\}$, we can induce a Q-function class $\{Q^\pi_{M_1}, \ldots, Q^\pi_{M_m}\}$ and run any model-free selection algorithm over them. The model-free setup also handles broader settings, especially when we lack prior knowledge of model dynamics. In either case, we treat the base algorithms as black-boxes and interact with them only through the models and the Q-functions they produce.

3. New Model-free Selector

In this section we introduce our new model-free selector, LSTD-Tournament. To start, we review the difficulties in model-free selection and the idea behind BVFT [XJ21; ZJ21] which we build on.

3.1. Challenges of Model-free Selection and BVFT

To select Q^π from $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$, perhaps the most natural idea is to check how much each candidate function Q_i violates the Bellman equation $Q^\pi = \mathcal{T}^\pi Q^\pi$, and choose the function that minimizes such a violation. This motivates the Bellman error (or residual) objective:

$$\mathbb{E}_\mu[(Q_i - \mathcal{T}^\pi Q_i)^2]. \tag{1}$$

Unfortunately, this loss cannot be estimated due to the infamous *double-sampling problem* [Bai95; SB18; CJ19], and the naïve estimation, which squares the TD error, is a biased estimation of the Bellman error (Eq.(1)) in stochastic environments:

$$\text{(TD-sq)} \qquad \mathbb{E}_\mu[(Q_i(s,a) - r - \gamma Q_i(s', \pi))^2]. \tag{2}$$

Common approaches to debiasing this objective involves additional "helper" classes, which we show can be naturally induced in the model-based setting; see Section 4 for details.

BVFT. The idea behind BVFT [XJ21] is to find an OPE algorithm for learning Q^π from a function class $\mathcal{F}$, such that to achieve polynomial sample-complexity guarantees, it suffices if $\mathcal{F}$ satisfies 2 assumptions:

1. Realizability, that $Q^\pi \in \mathcal{F}$.
2. Some *structural* (as opposed to *expressivity*) assumption on $\mathcal{F}$, e.g., smoothness, linearity, etc.

Standard learning results in RL typically require stronger expressivity assumption than realizability, such as the widely adopted *Bellman-completeness* assumption ($\mathcal{T}^\pi f \in \mathcal{F}, \forall f \in \mathcal{F}$). However, exceptions exist, and BVFT shows that *they can be converted into a pairwise-comparison subroutine* for selecting between two candidates $\{Q_i, Q_j\}$, and extension to multiple candidates can be done via a tournament procedure. Crucially, we can use $\{Q_i, Q_j\}$ to **automatically create an $\mathcal{F}$ needed by the algorithm** without additional side information or prior knowledge. We refer the readers to [XJ21] for further details, and we will also demonstrate such a process in the next subsection.

In short, BVFT provides a general recipe for converting a special kind of "base" OPE methods into selectors of favorable guarantees. Intuitively, the "base" method/analysis will determine the properties of the resulting selector. For BVFT, such a "base" is learning with Q^π-irrelevant abstractions [LWL06; Jia18], where the structural assumption on $\mathcal{F}$ is being piecewise-constant. Our novel insight is that for learning Q^π, there exists another algorithm, namely LSTDQ [LP03], which satisfies the needed criteria and has superior properties compared to Q^π-irrelevant abstractions, thus can induce better selectors than BVFT.

3.2. LSTD-Tournament

We now provide a theoretical analysis of LSTDQ (which is simplified from the literature [MPW23; PKBK23]), and show how to transform it into a selector via the BVFT recipe. In LSTDQ, we learn Q^π via linear function approximation, i.e., it is assumed that a feature map $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ is given, such that $Q^\pi(s,a) = \phi(s,a)^\top \theta^\star$, where $\theta^\star \in \mathbb{R}^d$ is the groundtruth linear coefficient. Equivalently, this asserts that the induced linear class, $\mathcal{F}_\phi := \{\phi^\top \theta : \theta \in \mathbb{R}^d\}$ satisfies realizability, $Q^\pi \in \mathcal{F}_\phi$.

LSTDQ provides a closed-form estimation of $\theta^\star$ by first estimating the following moment matrices:

$$\Sigma := \mathbb{E}_\mu[\phi(s,a)\phi(s,a)^\top], \quad \Sigma^{\text{cr}} := \mathbb{E}_\mu[\phi(s,a)\phi(s',\pi)^\top],$$
$$A := \Sigma - \gamma \Sigma^{\text{cr}}, \quad b := \mathbb{E}_\mu[\phi(s,a)r]. \tag{3}$$

As a simple algebraic fact, $A\theta^\star = b$. Therefore, when A is invertible, we immediately have that $\theta^\star = A^{-1}b$. The LSTDQ algorithm thus simply estimates A and b from data, denoted as $\widehat{A}$ and $\widehat{b}$, respectively, and estimate $\theta^\star$ as $\widehat{A}^{-1}\widehat{b}$. Alternatively, for any candidate θ, $\|A\theta - b\|_\infty$ can serve as a loss function that measures the violation of the equation $A\theta^\star = b$, which we can minimize over.[3] Its finite-sample guarantee is given below. All proofs of the paper can be found in Appendix B.

Theorem 1. *Let $\Theta \subset \mathbb{R}^d$ be a set of parameters such that $\theta^\star \in \Theta$. Assume $\max_{s,a}\|\phi(s,a)\|_2 \leq B_\phi$ and $\max_{\theta \in \Theta}\|\theta\|_2 \leq 1$. Let $\widehat{\theta} := \arg\min_{\theta \in \Theta} \|\widehat{A}\theta - \widehat{b}\|_\infty$. Then, with probability at least $1 - \delta$,*

$$\|Q^\pi - \hat{\phi}^\top \theta\|_\infty \leq \frac{6 \max\{R_{\max}, B_\phi\}^2}{\sigma_{\min}(A)} \sqrt{\frac{d \log(2d|\Theta|/\delta)}{n}},$$

where $\sigma_{\min}(\cdot)$ is the smallest singular value.

[3]When $\widehat{A}^{-1}\widehat{b} \in \Theta$, it will be a minimizer of the loss, so the loss-minimization version is a regularized generalization of LSTDQ.

Besides the realizability of Q^π, the guarantee also depends on the invertibility of A, which can be viewed as a coverage condition, since A changes with the data distribution μ [AJX20; AJS23; JX24]. In fact, in the on-policy setting (μ is an invariant distribution under π), $\sigma_{\min}(A)$ can be shown to be lower-bounded away from 0 [MPW23].

LSTD-Tournament. We are now ready to describe our new selector. Recall that we first deal with the case of two candidate functions, $\{Q_i, Q_j\}$, where $Q^\pi \in \{Q_i, Q_j\}$. To apply the LSTDQ algorithm and guarantee, all we need is to create the feature map ϕ such that Q^π is linearly realizable in ϕ. In the spirit of BVFT, we design the feature map as

$$\phi_{i,j}(s,a) := [Q_i(s,a), Q_j(s,a)]^\top. \tag{4}$$

The subscript "i, j" makes it clear that the feature is created based on Q_i and Q_j as candidates, and we will use similar conventions for all quantities induced from $\phi_{i,j}$, e.g., $A_{i,j}, b_{i,j}$, etc. Obviously, Q^π is linear in $\phi_{i,j}$ with $\theta^\star \in \{[1,0]^\top, [0,1]^\top\}$. Therefore, to choose between Q_i and Q_j, we can calculate the LSTDQ loss of $[1,0]^\top$ and $[0,1]^\top$ under feature $\phi_{i,j}$ and choose the one with smaller loss. For $\theta = [1,0]^\top$, we have $A_{i,j}\theta - b_{i,j} =$

$$\begin{aligned}
&\mathbb{E}_\mu\Big\{ \begin{bmatrix} Q_i(s,a) \\ Q_j(s,a) \end{bmatrix} ([Q_i(s,a), Q_j(s,a)] \\
&\quad - \gamma[Q_i(s',\pi), Q_j(s',\pi)])\Big\} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\
&\quad - \mathbb{E}_\mu\left[[Q_i(s,a), Q_j(s,a)] \cdot r\right] \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\
&= \mathbb{E}_\mu\left[\begin{bmatrix} Q_i(s,a) \\ Q_j(s,a) \end{bmatrix} (Q_i(s,a) - r - \gamma Q_i(s',\pi)) \right].
\end{aligned}$$

Taking the infinity-norm of the loss vector, we have $\|A_{i,j} \begin{bmatrix} 1 \\ 0 \end{bmatrix} - b_{i,j}\|_\infty$

$$= \max_{k\in\{i,j\}} |\mathbb{E}_\mu[Q_k(s,a)(Q_i(s,a) - r - \gamma Q_i(s',\pi))]|.$$

The loss for $\theta = [0,1]^\top$ is similar, where Q_i is replaced by Q_j. Following BVFT, we can generalize the procedure to m candidate functions $\{Q_1, \ldots, Q_m\}$ by pairwise comparison and recording the worst-case loss, this leads to our final loss function: $\mathcal{L}(Q_i; \{Q_j\}_{j\in[m]}, \pi) :=$

$$\max_{k\in[m]} |\mathbb{E}_\mu[Q_k(s,a)(Q_i(s,a) - r - \gamma Q_i(s',\pi))]|. \tag{5}$$

The actual algorithm replaces $\mathbb{E}_\mu$ with the empirical estimation from data, and chooses the Q_i that minimizes the loss. Building on Theorem 1, we have the following guarantee:

Theorem 2. *Given $Q^\pi := Q_{i^\star} \in \{Q_i\}_{i\in[m]}$, the $Q_{\hat{i}}$ that minimizes the empirical estimation of $\mathcal{L}(Q_i; \{Q_j\}_{j\in[m]}, \pi)$ (Eq.(5)) satisfies that w.p. $\geq 1-\delta$, $|J(\pi) - \mathbb{E}_{s\sim d_0}[Q_{\hat{i}}(s,\pi)]|$*

$$\leq \max_{i\in[m]\setminus\{i^\star\}} \frac{24V_{\max}^3}{\sigma_{\min}(A_{i,i^\star})} \sqrt{\frac{\log(8m/\delta)}{n}}.$$

Comparison to BVFT [XJ21]. BVFT's guarantee has a slow $1/\epsilon^4$ rate for OPE [ZJ21; JRSW24], whereas our method enjoys the standard $1/\epsilon^2$ rate. The additional $1/\epsilon^2$ is due to an adaptive discretization step in BVFT, which also makes its implementation somewhat complicated as the resolution needs to be heuristically chosen. By comparison, the implementation of LSTD-Tournament is simple and straightforward. Both methods inherit the coverage assumptions from their base algorithms and are not immediately comparable. We leave a detailed comparison of their differences for future work.

Variants. A key step in the derivation is to design the linearly realizable feature of Eq.(4), but the design is not unique as any non-degenerate linear transformation would also suffice. For example, we can use $\phi_{i,j} = [Q_i/c_i, (Q_j - Q_i)/c_{j,i}]$; the "diff-of-value" term $Q_j - Q_i$ has shown improved numerical properties in practice [KZTL20; CXJA22], and c_i, $c_{j,i}$ can normalize the discriminators to unit variance for further numerical stability; this will also be the version we use in the main-text experiments. Preliminary empirical comparison across these variants can be found in Appendix E.2.

4. Model-based Selectors

We now turn to the model-based setting, i.e., choosing a model from $\{M_i\}_{i\in[m]}$. This is a practical scenario when we have structural knowledge of the system dynamics and can build reasonable simulators, but simulators of complex real-world systems will likely have many design choices and knobs that cannot be set from prior knowledge alone. In some sense, the task is not very different from system identification in control and model learning in model-based RL, except that (1) we focus on a finite and small number of plausible models, instead of a rich and continuous hypothesis class, and (2) the ultimate goal is to perform accurate OPE, and learning the model is only an intermediate step.

Existing Methods. Given the close relation to model learning, a natural approach is to simply minimize the model prediction loss [Jia24]: a candidate model M is scored by

$$\mathbb{E}_{(s,a,s')\sim\mu, \tilde{s}\sim P(\cdot|s,a)}[d(s', \tilde{s})], \tag{6}$$

where s' is in the data and generated according to the real dynamics $P^\star(\cdot|s,a)$, and $\tilde{s}$ is generated from the candidate model M's dynamics P. $d(\cdot,\cdot)$ is a distance metric that measures the difference between states.

Despite its wide use and simplicity [NKFL18], the method has major caveats: first, the distance metric $d(\cdot,\cdot)$ is a design choice. When the state is represented as a real-valued vector, it is natural to use the ℓ_2 distance as $d(\cdot,\cdot)$, which changes if we simply normalize/rescale the coordinates. Second, the metric is biased for stochastic environments as discussed in prior works [Jia24; VAAGF23], which we will also demonstrate in the experiment section (Section 6); essentially this is a version of the double-sampling issue but for the model-based setting [AFJKM24].

There are alternative methods that address these issues. For example, in the theoretical literature, MLE losses are commonly used, i.e., $\mathbb{E}_\mu[\log P(s'|s,a)]$ [AKKS20; UZS21; LNSJ23], which avoids $d(\cdot,\cdot)$ and works properly for stochastic MDPs by effectively measuring the KL divergence between $P^\star(\cdot|s,a)$ and $P(\cdot|s,a)$. Unfortunately, most complex simulators do not provide explicit probabilities $P(s'|s,a)$, making it difficult to use in practice. Moreover, when the support of $P^\star(\cdot|s,a)$ is not fully covered by $P(\cdot|s,a)$, the loss can become degenerate.

To address these issues, we propose to estimate the Bellman error $\mathbb{E}_\mu[(Q_i - \mathcal{T}^\pi Q_i)^2]$, where $Q_i := Q^\pi_{M_i}$. As discussed earlier, this objective suffers the double-sampling issue in the model-free setting, which we show can be addressed when we have access to candidate models $\{M_1,\ldots,M_m\}$ that contains the true dynamics $M^\star$. Moreover, the Bellman error $|Q^\pi_{M_i}(s,a) - (\mathcal{T}^\pi Q^\pi_{M_i})(s,a)| =$

$$\gamma|\mathbb{E}_{s'\sim P^\star(\cdot|s,a)}[Q_i(s',\pi)] - \mathbb{E}_{s'\sim P_i(\cdot|s,a)}[Q_i(s',\pi)]|,$$

which can be viewed as an IPM loss [Mül97] that measures the divergence between $P^\star(\cdot|s,a)$ and $P(\cdot|s,a)$ under $Q_i(\cdot,\pi)$ as a discriminator. IPM is also a popular choice of model learning objective in theory [SJKAL19; VJY21], and the Bellman error provides a natural discriminator relevant for the ultimate task of interest, namely OPE.

4.1. Regression-based Selector

Recall that the difficulty in estimating the Bellman error $\mathbb{E}_\mu[(Q_i - \mathcal{T}^\pi Q_i)^2]$ is the uncertainty in $\mathcal{T}^\pi$. To overcome this, we leverage the following observation from [ASM08], where for any $f : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$,

$$\mathcal{T}^\pi f \in \operatorname*{arg\,min}_{g:\mathcal{S}\times\mathcal{A}\to\mathbb{R}} \mathbb{E}_\mu[(g(s,a) - r - \gamma f(s',\pi))^2], \tag{7}$$

which shows that we can estimate $\mathcal{T}^\pi Q_i$ by solving a sample-based version of the above regression problem with $f = Q_i$. Statistically, however, we cannot afford to minimize the objective over all possible functions g; we can only search over a limited set $\mathcal{G}_i$ that ideally captures the target $\mathcal{T}^\pi Q_i$. Crucially, in the model-based setting we can generate such a set directly from the candidates $\{M_i\}_{i\in[m]}$:

Proposition 3. *Let $\mathcal{G}_i := \{\mathcal{T}^\pi_{M_j} Q_i : j \in [m]\}$. Then if $M^\star \in \{M_i\}_{i\in[m]}$, it follows that $\mathcal{T}^\pi Q_i = \mathcal{T}^\pi_{M^\star} Q_i \in \mathcal{G}_i$.*

The constructed $\mathcal{G}_i$ ensures that regression is statistically tractable given its small cardinality, $|\mathcal{G}_i| = m$. To select Q_i, we choose Q_i with the smallest loss defined as follows:

1. $\widehat{g}_i := \arg\min_{g\in\mathcal{G}_i} \mathbb{E}_\mathcal{D}[(g(s,a) - r - \gamma Q_i(s',\pi))^2]$.
2. The loss of Q_i is $\mathbb{E}_\mathcal{D}[(\widehat{g}_i(s,a) - Q_i(s,a))^2]$.

The 2nd step follows from [ZDMAK23]. Alternatively, we can also use the min value of Eq.(7) (instead of the argmin function) to correct for the bias in TD-squared (Eq.(2)) [ASM08]; see [LNPW23] for another related variant. These approaches share similar theoretical guarantees under standard analyses [XJ21; XCJMA21], and we only state the guarantee for the [ZDMAK23] version below, but will include both in the experiments.

Theorem 4. *Let $\mathcal{C}^\pi := \mathbb{E}_\pi\left[\frac{d^\pi(s,a)}{\mu(s,a)}\right]$. For $Q_{\widehat{i}}$ that minimizes $\mathbb{E}_\mathcal{D}[(\widehat{g}_i(s,a) - Q_i(s,a))^2]$ we have w.p. $\geq 1-\delta$,*

$$J(\pi) - \mathbb{E}_{d_0}\left[Q_{\widehat{i}}(s,\pi)\right] \leq \frac{V_{\max}}{1-\gamma}\sqrt{\frac{152\cdot\mathcal{C}^\pi\cdot\log\left(\frac{4m}{\delta}\right)}{n}}.$$

4.2. Sign-flip Average Bellman Error

We now present another selector that leverages the information of $\mathcal{G}_i = \{\mathcal{T}^\pi_{M_j} : j \in [m]\}$ in a different manner. Instead of measuring the squared Bellman error, we can also measure the absolute error, which can be written as (some (s,a) argument to functions are omitted):

$$\begin{aligned}
&\mathbb{E}_\mu[|Q_i - \mathcal{T}^\pi_{M^\star} Q_i|] \\
&= \mathbb{E}_\mu[\operatorname{sgn}(Q_i(s,a) - (\mathcal{T}^\pi Q_i)(s,a))(Q_i - \mathcal{T}^\pi Q_i)] \\
&= \mathbb{E}_\mu[\operatorname{sgn}(Q_i - \mathcal{T}^\pi Q_i)(Q_i(s,a) - r - \gamma Q_i(s',\pi))] \\
&\leq \max_{g\in\mathcal{G}_i} \mathbb{E}_\mu[\operatorname{sgn}(Q_i - g)(Q_i(s,a) - r - \gamma Q_i(s',\pi))]. \quad (8)
\end{aligned}$$

Here, the $\mathcal{G}_i$ from Proposition 3 induces a set of sign functions $\operatorname{sgn}(Q_i - g)$, which includes $Q_i - \mathcal{T}^\pi Q_i$, that will negate any negative TD errors. The guarantee is as follows:

Theorem 5. *Let $Q_{\widehat{i}}$ be the minimizer of the empirical estimate of Eq.(8), and $\mathcal{C}^\pi_\infty := \max_{s,a} \frac{d^\pi(s,a)}{\mu(s,a)}$. W.p. $\geq 1-\delta$,*

$$J(\pi) - \mathbb{E}_{d_0}\left[Q_{\widehat{i}}(s,\pi)\right] \leq 4\cdot\mathcal{C}^\pi_\infty\cdot V_{\max}\sqrt{\frac{\log(2m/\delta)}{n}}.$$

5. A Model-based Experiment Protocol

Given the new selectors, we would like to evaluate and compare them empirically. However, as alluded to in the introduction, current experiment protocols have various caveats and make it difficult to evaluate the estimators in well-controlled settings. In this section, we describe a novel model-based experiment protocol, which can be used to evaluate both model-based and model-free selectors.

5.1. The Protocol

Our protocol consists of experiment units defined by the following elements:

1. Groundtruth model $M^\star$.
2. Candidate model list $\mathcal{M} = \{M_i\}_{i\in[m]}$.
3. Behavior policy π_b and offline sample size n.
4. Target policies $\Pi = \{\pi_1, \ldots, \pi_l\}$.

Given the specification of a unit, we will draw a dataset of size n from $M^\star$ using behavior policy π_b. For each target policy $\pi \in \Pi$, we apply different selectors to choose a model $M \in \mathcal{M}$ to evaluate π. Model-free algorithms will access M only through its Q-function, Q^π_M, effectively choosing from the set $\mathcal{Q} = \{Q^\pi_M : M \in \mathcal{M}\}$. Finally, the prediction error $|J_M(\pi) - J_{M^\star}(\pi)|$ is recorded and averaged over the target policies in Π. Moreover, we may gather results from multiple units that share the same $M^\star$ but differ in $\mathcal{M}$ and/or the behavior policy to investigate issues such as robustness to misspecification and data coverage, as we will demonstrate in the next section.

Lazy Evaluation of Q-values via Monte Carlo. While the pipeline is conceptually straightforward, practically accessing the Q-function Q^π_M is nontrivial: we could run TD-style algorithms in M to learn Q^π_M, but that invokes a separate RL algorithm that may require additional tuning and verification, and it can be difficult to control the quality of the learned function.

Our innovation here is to note that, for all the model-free algorithms we are interested in evaluating, **they all access Q^π_M exclusively through the value of $Q^\pi_M(s,a)$ and $Q^\pi_M(s',\pi)$ for (s,a,r,s') in the offline dataset $\mathcal{D}$.** That is, given n data points in $\mathcal{D}$, we only need to know $2n$ scalar values about Q^π_M. Therefore, we propose to directly compute these values without explicitly representing Q^π_M, and each value can be easily estimated by averaging over multiple Monte-Carlo rollouts, i.e., $Q^\pi_M(s,a) = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t r_t | s_0 = s, a_0 = a]$ can be approximated by rolling out multiple trajectories starting from (s,a) and taking actions according to π.

Moreover, for the model-based estimators proposed in Section 4, we need access to quantities in the form of $(\mathcal{T}^\pi_{M_j} Q^\pi_{M_i})(s,a)$. This value can also be obtained by Monte-Carlo simulation: (1) start in (s,a) and simulate one step in M_j, then (2) switch to M_i, simulate from step 2 onwards and rollout the rest of the trajectory.

5.2. Computational Efficiency

Despite not involving neural-net optimization, the experiment can still be computationally intensive due to rolling out a large number of trajectories. In our code, we incorporate the following measures to reduce the computational cost:

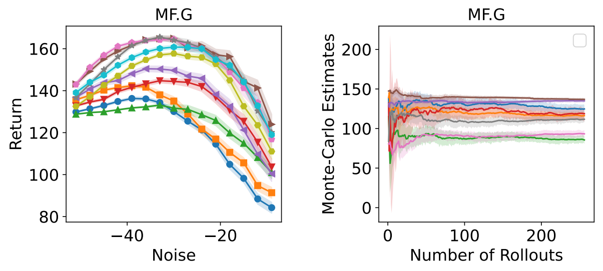

Figure 1. **Left:** $J_M(\pi)$ in $M \in \mathcal{M}_\mathbf{g}$ for different target policies. **Right:** Convergence of Monte-Carlo estimates of $J(\pi)$. Each curve corresponds to a target policy.

Q-caching. The most intensive part of the pipeline is to roll-out Monte-Carlo trajectories for Q-value estimation. In contrast, the cost of running the actual selection algorithms is often much lower and negligible. Therefore, we generate these Monte-Carlo Q-estimates and save them to files, and retrieve them during the selection period. This makes it efficient to experiment with new selection algorithms or add extra baselines, and also enables fast experiment that involves a subset of the candidate models (see Section 6.2).

Bootstrapping. To account for the randomness due to $\mathcal{D}$, we use bootstrapping to sample (with replacement) multiple datasets and run the algorithms on each dataset, and report the mean performance across these bootstrapped samples with 95% confidence intervals. Using bootstrapping maximally reuses the cached Q-values and avoids the high computational costs of sampling multiple datasets and performing Q-caching in each of them, which is unavoidable if we were to repeat each experiment verbatim multiple times.

6. Exemplification of the Protocol

In this section we instantiate our protocol in the Gym Hopper environment to demonstrate its utility, while also providing preliminary empirical results for our algorithms.

6.1. Experiment Setup and Main Results

Our experiments will be based on the *Hopper-v4* environment. To create a variety of environments, we add different levels of stochastic noise in the transitions and change the gravity constant (see Appendix C.1). Each environment is then parameterized by the gravity constant $\mathbf{g}$ and noise level $\mathbf{n}$. We consider arrays of such environments as the set of candidate simulator $\mathcal{M}$: in most of our results, we consider a "gravity grid" (denoted using **MF.G** in the figures) $\mathcal{M}_\mathbf{g} := \{M^0_\mathbf{g} \ldots, M^{14}_\mathbf{g}\}$ (fixed noise level, varying gravity constants from -51 to -9) and a "noise grid" (**MF.N**) $\mathcal{M}_\mathbf{n} := \{M^0_\mathbf{n} \ldots, M^{14}_\mathbf{n}\}$ (fixed gravity constant, varying noise level from 10 to 100). Each array contains 15 environments, though some subsequent results may only involve

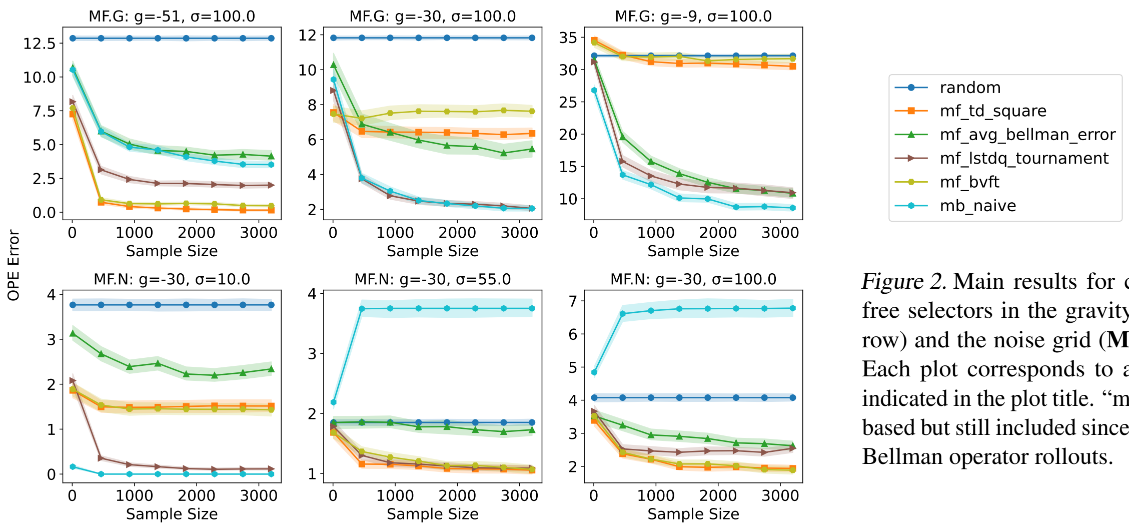

Figure 2. Main results for comparing model-free selectors in the gravity grid (**MF.G**; top row) and the noise grid (**MF.N**; bottom row). Each plot corresponds to a different $M^\star$ as indicated in the plot title. "mb_naive" is model-based but still included since it does not require Bellman operator rollouts.

a subset of them (Section 6.2). Some of these simulators will also be treated as groundtruth environment $M^\star$, which determines the groundtruth performance of target policies and produces the offline dataset $\mathcal{D}$.

Behavior and target policies. We create 15 target policies by running DDPG [Lil+15] in one of the environments and take checkpoints. For each $M^\star$, the behavior policy is the randomized version of one of the target policies; see Appendix C.2 for details. A dataset is collected by sampling trajectories until $n = 3200$ transition tuples are obtained. As a sanity check, we plot $J_M(\pi)$ for $\pi \in \Pi_{\mathbf{g}}$ and $M \in \mathcal{M}_{\mathbf{g}}$ in Figure 1. As can be shown in the figure, the target policies have different performances, and also vary in a nontrivial manner w.r.t. the gravity constant $\mathbf{g}$. It is important to perform such a sanity check to avoid degenerate settings, such as $J_M(\pi)$ varies little across $M \in \mathcal{M}$ (then even a random selection will be accurate) or across $\pi \in \Pi$.

Number of Rollouts. We then decide the two important parameters for estimating the Q-value, the number of Monte-Carlo rollouts l and the horizon (i.e., trajectory length) H. For horizon, we set $H = 1024$ which is substantially longer than typically observed trajectories from the target policies. For l, we plot the convergence of $J_M(\pi)$ estimation and choose $l = 128$ accordingly (see Figure 1R).

Compared Methods. We compare our methods with baselines, including TD-square (Eq.(2)), naïve model-based (Eq.(6)), BVFT [ZJ21], and "average Bellman error" $|E_{\mathcal{D}}[Q_i(s,a) - r - \gamma Q_i(s',\pi)]|$ [JKALS17], which can be viewed as our LSTD-Tournament but with a trivial constant discriminator. The model-based methods in Section 4 require MC rollouts for $\{T^\pi_{M_j} Q^\pi_{M_i} : i,j \in [m]\}$, which requires $O(m^2)$ computational complexity. Therefore, we first compare other selectors (mostly model-free) in Figure 2 with $m = 15$; the relatively large number of candidate simulators will also enable the later subgrid studies in Section 6.2. We then perform a separate experiment with $m = 5$ for the model-based selectors (Figure 3).

Main Results. Figure 2 shows the main model-free results. Our LSTD-Tournament method demonstrates strong and reliable performance. Note that while some methods sometimes outperform it, they suffer catastrophic performances when the true environment changes. For example, the naïve model-based method performs poorly in high-noise environment, as predicted by theory (Section 4). BVFT's performance mostly coincides with TD-sq, which is a possible degeneration predicted by [ZJ21]. This is particularly plausible when the number of data points n is not large enough to allow for meaningful discretization and partition of the state space required by the method.

Figure 3 shows the result on smaller candidate model sets (**MB.G** and **MB.N**; see Appendix C.2), where we implement the 3 model-based selectors in Section 4 whose computational complexities grow quadratically with $|\mathcal{M}|$. Our expectation was that (1) these algorithms should address the double-sampling issue and will outperform naïve model-based when the latter fails catastrophically, and (2) by having access to more information (the model, and in particular, the Bellman operators), model-based should outperform model-free algorithms under realizability. While the first prediction is largely verified, we are surprised to find that the second prediction went wrong, and our LSTD-Tournament method is more robust and generally outperforms the more complicated model-based selectors.

6.2. Subgrid Studies: Gaps and Misspecifications

We now demonstrate how to extract additional insights from the Q-values cached earlier. Due to space limit we are only able to show representative results in Figure 4, and more

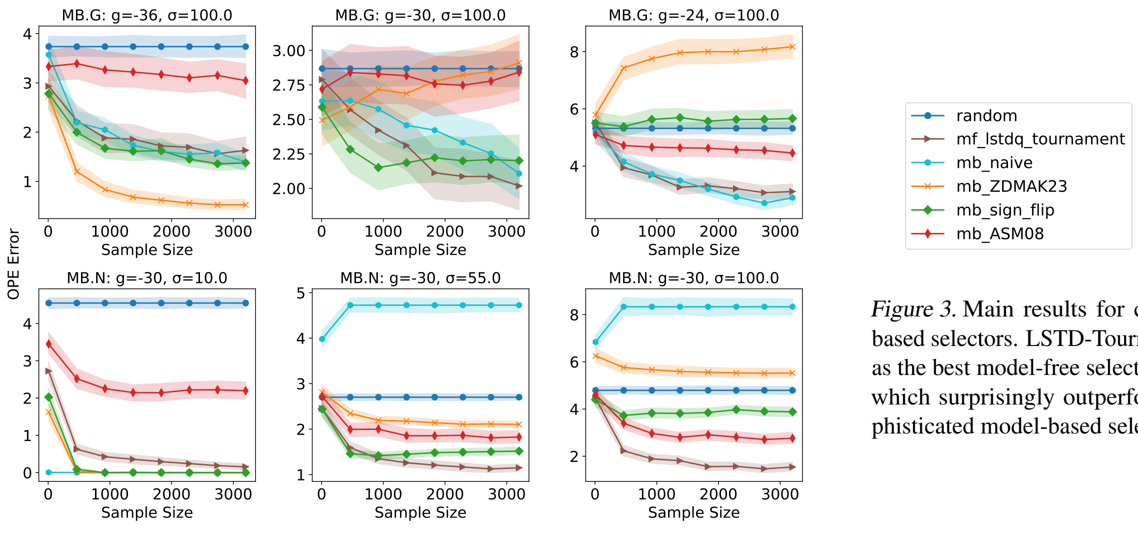

Figure 3. Main results for comparing model-based selectors. LSTD-Tournament is included as the best model-free selector for comparison, which surprisingly outperforms the more sophisticated model-based selectors in Section 4.

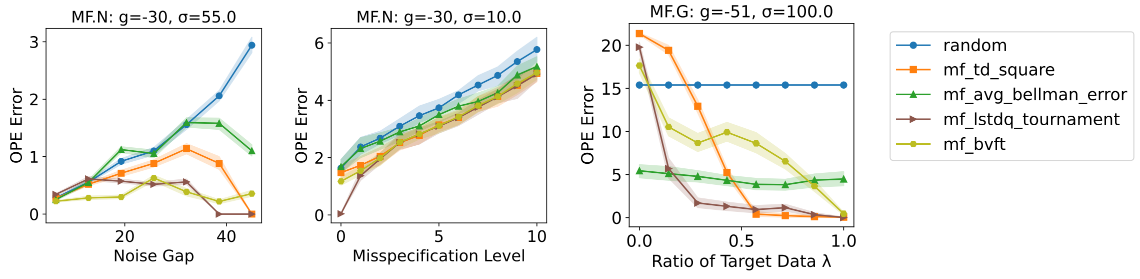

Figure 4. **Left:** OPE error vs. simulator gaps. **Middle:** OPE error vs. misspecification. **Right:** OPE error vs. data coverage.

comprehensive results can be found in Appendix D.

Gaps. We investigate an intellectually interesting question: is the selection problem easier if the candidate simulators are very similar to each other, or when they are very different? We argue that the answer is **neither**, and an intermediate difference (or *gap*) is the most challenging: if the simulators are too similar, their $J_M(\pi)$ predictions will all be close to $J_{M^\star}(\pi)$ since $M \approx M^\star$, and any selection algorithm will perform well; if the simulators are too dissimilar, it should be easy to tell them apart, which also makes the task easy.

We show how we can empirically test this. We let $M^\star = M^7_{\mathbf{n}}$, and run the experiments with different 3-subsets of $\mathcal{M}_{\mathbf{n}}$, including $\{6,7,8\}$ (least gap), $\{5,7,9\}$, $\ldots$, $\{0,7,14\}$ (largest gap). Since the needed Q-values have already been cached in the main experiments, we can skip caching and directly run the selection algorithms. We plot the prediction error as a function of gap size in Figure 4L, and observe the down-U curves (except for trivial methods such as random) as predicted by theory.

Misspecification. Similarly, we can study the effect of misspecification, that is, $M^\star \in \mathcal{M}$. For example, we can take $M^\star = M^0_\sigma$, and consider different subsets of $\mathcal{M}_{\mathbf{n}}$: 0–4 (realizable), 1–5 (low misspecification), $\ldots$, 10–14 (high misspecification). Figure 4M plots prediction error vs. misspecification level for different methods, where we expect to observe potential difference in the sensitivity to misspecification. The actual result is not that interesting given similar increasing trends for all methods.

6.3. Data Coverage

In the previous subsection, we have seen how multiple experiment units that only differ in $\mathcal{M}$ can provide useful insights. Here we show that we can also probe the methods' sensitivity to data coverage by looking at experiment units that only differ in the dataset $\mathcal{D}$. In Figure 4R, we take a previous experiment setting ($\mathcal{M}_{\mathbf{g}}$) and isolate a particular target policy π; then, we create two datasets: (1) $\mathcal{D}_\pi$ sampled using π; (2) $\mathcal{D}_{\text{off}}$ sampled using a policy that is created to be very different from the target policies and offer very little coverage (see Appendix C.2). Then, we run the algorithm with λ fraction of data from $\mathcal{D}_\pi$ combined with $(1-\lambda)$ from $\mathcal{D}_{\text{off}}$; as predicted by theory, most methods perform better with better coverage (large λ), and performance degrades as λ goes to 0.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

[ACK24] Philip Amortila, Tongyi Cao, and Akshay Krishnamurthy. “Mitigating covariate shift in misspecified regression with applications to reinforcement learning”. In: *arXiv preprint arXiv:2401.12216* (2024).

[AFJKM24] Philip Amortila, Dylan J Foster, Nan Jiang, Akshay Krishnamurthy, and Zakaria Mhammedi. “Reinforcement Learning under Latent Dynamics: Toward Statistical and Algorithmic Modularity”. In: *arXiv preprint arXiv:2410.17904* (2024).

[AJS23] Philip Amortila, Nan Jiang, and Csaba Szepesvári. “The optimal approximation factors in misspecified off-policy value function estimation”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 768–790.

[AJX20] Philip Amortila, Nan Jiang, and Tengyang Xie. “A Variant of the Wang-Foster-Kakade Lower Bound for the Discounted Setting”. In: *arXiv preprint arXiv:2011.01075* (2020).

[AKKS20] Alekh Agarwal, Sham Kakade, Akshay Krishnamurthy, and Wen Sun. “FLAMBE: Structural Complexity and Representation Learning of Low Rank MDPs”. In: *arXiv preprint arXiv:2006.10814* (2020).

[ASM08] András Antos, Csaba Szepesvári, and Rémi Munos. “Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path”. In: *Machine Learning* (2008).

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

[CJ19] Jinglin Chen and Nan Jiang. “Information-Theoretic Considerations in Batch Reinforcement Learning”. In: *Proceedings of the 36th International Conference on Machine Learning*. 2019, pp. 1042–1051.

[CXJA22] Ching-An Cheng, Tengyang Xie, Nan Jiang, and Alekh Agarwal. “Adversarially trained actor critic for offline reinforcement learning”. In: *International Conference on Machine Learning* (2022).

[Dee] Google Deepmind. *MuJoCo Documentation*. URL: https://mujoco.readthedocs.io/en/stable/computation/index.html.

[EGW05] Damien Ernst, Pierre Geurts, and Louis Wehenkel. “Tree-based batch mode reinforcement learning”. In: *Journal of Machine Learning Research* 6 (2005), pp. 503–556.

[FMPNG22] Scott Fujimoto, David Meger, Doina Precup, Ofir Nachum, and Shixiang Shane Gu. “Why should i trust you, bellman? the bellman error is a poor replacement for value error”. In: *arXiv preprint arXiv:2201.12417* (2022).

[Jia18] Nan Jiang. *CS 598: Notes on State Abstractions*. http://nanjiang.cs.illinois.edu/files/cs598/note4.pdf. University of Illinois at Urbana-Champaign. 2018.

[Jia24] Nan Jiang. “A Note on Loss Functions and Error Compounding in Model-based Reinforcement Learning”. In: *arXiv preprint arXiv:2404.09946* (2024).

[JKALS17] Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. “Contextual decision processes with low Bellman rank are PAC-learnable”. In: *International Conference on Machine Learning*. 2017.

[JL16] Nan Jiang and Lihong Li. “Doubly Robust Off-policy Value Evaluation for Reinforcement Learning”. In: *Proceedings of the 33rd International Conference on Machine Learning*. Vol. 48. 2016, pp. 652–661.

[JRSW24] Zeyu Jia, Alexander Rakhlin, Ayush Sekhari, and Chen-Yu Wei. “Offline Reinforcement Learning: Role of State Aggregation and Trajectory Data”. In: *arXiv preprint arXiv:2403.17091* (2024).

[JX24] Nan Jiang and Tengyang Xie. “Offline reinforcement learning in large state spaces: Algorithms and guarantees”. In: (2024). https://nanjiang.cs.illinois.edu/files/STS_Special_Issue_Offline_RL.pdf.

[KKKKNS23] Haruka Kiyohara, Ren Kishimoto, Kosuke Kawakami, Ken Kobayashi, Kazuhide Nakata, and Yuta Saito. “SCOPE-RL: A

Python Library for Offline Reinforcement Learning and Off-Policy Evaluation". In: *arXiv preprint arXiv:2311.18206* (2023).

[KZTL20] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. "Conservative q-learning for offline reinforcement learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 1179–1191.

[LCLW11] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. "Unbiased Offline Evaluation of Contextual-bandit-based News Article Recommendation Algorithms". In: *Proceedings of the 4th International Conference on Web Search and Data Mining*. 2011, pp. 297–306.

[Lil+15] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Manfred Otto Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. "Continuous control with deep reinforcement learning". In: *CoRR* abs/1509.02971 (2015). URL: https://api.semanticscholar.org/CorpusID:16326763.

[LKTF20] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. "Offline reinforcement learning: Tutorial, review, and perspectives on open problems". In: *arXiv preprint arXiv:2005.01643* (2020).

[LLTZ18] Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. "Breaking the curse of horizon: Infinite-horizon off-policy estimation". In: *Advances in Neural Information Processing Systems*. 2018, pp. 5356–5366.

[LNPW23] Vincent Liu, Prabhat Nagarajan, Andrew Patterson, and Martha White. "When is Offline Policy Selection Sample Efficient for Reinforcement Learning?" In: *arXiv preprint arXiv:2312.02355* (2023).

[LNSJ23] Qinghua Liu, Praneeth Netrapalli, Csaba Szepesvari, and Chi Jin. "Optimistic mle: A generic model-based algorithm for partially observable sequential decision making". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. 2023, pp. 363–376.

[LP03] Michail G Lagoudakis and Ronald Parr. "Least-squares policy iteration". In: *The Journal of Machine Learning Research* 4 (2003), pp. 1107–1149.

[LTND22] Jonathan Lee, George Tucker, Ofir Nachum, and Bo Dai. "Model selection in batch policy optimization". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 12542–12569.

[LVY19] Hoang Le, Cameron Voloshin, and Yisong Yue. "Batch Policy Learning under Constraints". In: *International Conference on Machine Learning*. 2019, pp. 3703–3712.

[LWL06] Lihong Li, Thomas J Walsh, and Michael L Littman. "Towards a unified theory of state abstraction for MDPs". In: *Proceedings of the 9th International Symposium on Artificial Intelligence and Mathematics*. 2006, pp. 531–539.

[MPW23] Wenlong Mou, Ashwin Pananjady, and Martin J Wainwright. "Optimal oracle inequalities for projected fixed-point equations, with applications to policy evaluation". In: *Mathematics of Operations Research* 48.4 (2023), pp. 2308–2336.

[Mül97] Alfred Müller. "Integral probability metrics and their generating classes of functions". In: *Advances in Applied Probability* (1997).

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

[NFBJSB22] Allen Nie, Yannis Flet-Berliac, Deon Jordan, William Steenbergen, and Emma Brunskill. "Data-efficient pipeline for offline reinforcement learning with limited data". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 14810–14823.

[NKFL18] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning". In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 7559–7566.

[Pai+20] Tom Le Paine, Cosmin Paduraru, Andrea Michi, Caglar Gulcehre, Konrad Zolna, Alexander Novikov, Ziyu Wang, and Nando de Freitas. "Hyperparameter Selection for Offline Reinforcement Learning". In: *arXiv preprint arXiv:2007.09055* (2020).

[PKBK23] Juan C Perdomo, Akshay Krishnamurthy, Peter Bartlett, and Sham Kakade. "A Complete Characterization of Linear Estimators for Offline Policy Evaluation”. In: *Journal of Machine Learning Research* 24.284 (2023), pp. 1–50.

[PSS00] Doina Precup, Richard S Sutton, and Satinder P Singh. “Eligibility Traces for Off-Policy Policy Evaluation”. In: *Proceedings of the Seventeenth International Conference on Machine Learning*. 2000, pp. 759–766.

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

[SJKAL19] Wen Sun, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. “Model-based RL in Contextual Decision Processes: PAC bounds and Exponential Improvements over Model-free Approaches”. In: *Conference on Learning Theory*. 2019.

[TB16] Philip Thomas and Emma Brunskill. “Data-Efficient Off-Policy Policy Evaluation for Reinforcement Learning”. In: *Proceedings of the 33rd International Conference on Machine Learning*. 2016.

[UHJ20] Masatoshi Uehara, Jiawei Huang, and Nan Jiang. “Minimax Weight and Q-Function Learning for Off-Policy Evaluation”. In: *Proceedings of the 37th International Conference on Machine Learning*. 2020, pp. 1023–1032.

[UKNST23] Takuma Udagawa, Haruka Kiyohara, Yusuke Narita, Yuta Saito, and Kei Tateno. “Policy-adaptive estimator selection for off-policy evaluation”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 8. 2023, pp. 10025–10033.

[UZS21] Masatoshi Uehara, Xuezhou Zhang, and Wen Sun. “Representation learning for online and offline rl in low-rank mdps”. In: *arXiv preprint arXiv:2110.04652* (2021).

[VAAGF23] Claas A Voelcker, Arash Ahmadian, Romina Abachi, Igor Gilitschenski, and Amir-massoud Farahmand. “λ-AC: Learning latent decision-aware models for reinforcement learning in continuous state-spaces”. In: *arXiv preprint arXiv:2306.17366* (2023).

[VJY21] Cameron Voloshin, Nan Jiang, and Yisong Yue. “Minimax Model Learning”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 1612–1620.

[VLJY19] Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. “Empirical Study of Off-Policy Policy Evaluation for Reinforcement Learning”. In: *arXiv preprint arXiv:1911.06854* (2019).

[XCJMA21] Tengyang Xie, Ching-An Cheng, Nan Jiang, Paul Mineiro, and Alekh Agarwal. “Bellman-consistent Pessimism for Offline Reinforcement Learning”. In: *arXiv preprint arXiv:2106.06926* (2021).

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

[ZDMAK23] Joshua P Zitovsky, Daniel De Marchi, Rishabh Agarwal, and Michael Rene Kosorok. “Revisiting bellman errors for offline model selection”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 43369–43406.

[ZJ21] Siyuan Zhang and Nan Jiang. “Towards hyperparameter-free policy selection for offline reinforcement learning”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 12864–12875.

A. Other Related Works

Here we review some existing works on model selection in offline RL. Most of them are not concerned about new selection algorithms with theoretical guarantees (apart from [XJ21; ZJ21; ZDMAK23; LNPW23] which are already discussed in the main text) or experiment protocol for OPE model selection (see [VLJY19; KKKKNS23] for experiment protocol and benchmarks of OPE itself), so their focus is different and often provides insights complementary to our work. For example, [NFBJSB22] discuss data splitting in offline model selection; this is a question we avoid by assuming a fixed holdout dataset for OPE model selection. An exception is [UKNST23] who studies the model selection problem for OPE itself, but focuses on the bandit case and makes heavy use of the importance sampling estimator, which we do not consider due to the focus on long-horizon tasks.

[FMPNG22] challenge the idea of using Bellman errors for model selection due to their surrogacy and poor correlation with actual objective; despite the valid criticisms, there are no clear alternatives that address the pain points of Bellman errors, and the poor performance is often due to lack of data coverage, which makes the task fundamentally difficult for any algorithms. We still believe that Bellman-error-like objectives (defined in a broad sense, which includes our LSTD-Tournament) are promising for model selection, and the improvement on OPE error is the right goal to pursue instead of correlation (which we know could be poor due to the surrogacy).

As mentioned above and demonstrated in our experiments, the lack of data coverage is a key factor that determines the difficulty of the selection tasks. [LTND22] propose feature selection algorithms for offline contextual bandits that account for the different coverage effects of candidate features. On a related note, ideas from offline RL training, such as version-space-based pessimism [XCJMA21], can also be incorporated in our method. This will unlikely improve the accuracy of OPE itself, but may be helpful if we measure performance by how OPE can eventually lead to successful selection of performant policies, which we leave for future investigation.

B. Proofs

B.1. Proof of Theorem 1

Proof. Define the following loss vectors,

$$\begin{aligned} \ell(\theta) &:= A\theta - b \in \mathbb{R}^d, \\ \widehat{\ell}(\theta) &:= \widehat{A}\theta - \widehat{b} \in \mathbb{R}^d \end{aligned}$$

and recall that we select as the estimator

$$\widehat{\theta} := \arg\min_{\theta\in\Theta} \|\widehat{\ell}(\theta)\|_\infty.$$

Since $\theta^\star = A^{-1}b$, we can write the desired bound as a function of $\ell(\theta)$ as follows,

$$\begin{aligned} \|Q^\pi(\cdot) - \phi^\top(\cdot)\widehat{\theta}\|_\infty &= \|\phi^\top(\cdot)(\widehat{\theta} - \theta^\star)\|_\infty \\ &= \|\phi^\top(\cdot)A^{-1}(A\widehat{\theta} - b)\|_\infty \\ &= \|\phi^\top(\cdot)A^{-1}\ell(\widehat{\theta})\|_\infty \\ &= \max_{s,a}|\phi^\top(s,a)A^{-1}\ell(\widehat{\theta})| \\ &\le \left(\max_{s,a}\left\|\phi^\top(s,a)\right\|_2\right)\cdot\left\|A^{-1}\right\|_2\cdot\|\ell(\widehat{\theta})\|_2 \\ &\le \sqrt{d}B_\phi\cdot\left\|A^{-1}\right\|_2\cdot\|\ell(\widehat{\theta})\|_\infty \end{aligned}$$

Next, we control the $\ell(\widehat{\theta})$ term. In the sequel we will establish via concentration that

$$\|\ell(\theta) - \widehat{\ell}(\theta)\|_\infty \le \varepsilon_{\mathsf{stat}} := 3\cdot\max\{R_{\max}, B_\phi\}^2\cdot\sqrt{\frac{\log(2d|\Theta|\delta^{-1})}{n}},\ \forall\theta\in\Theta. \tag{9}$$

Then, we have that

$$\begin{aligned}\|\ell(\widehat{\theta})\|_\infty &\leq \|\widehat{\ell}(\widehat{\theta})\|_\infty + \varepsilon_{\mathsf{stat}} \\ &\leq \|\widehat{\ell}(\theta^\star)\|_\infty + \varepsilon_{\mathsf{stat}} \\ &\leq \|\ell(\theta^\star)\|_\infty + 2\cdot\varepsilon_{\mathsf{stat}} \\ &= 2\cdot\varepsilon_{\mathsf{stat}},\end{aligned}$$

where we recall that $\widehat{\theta} = \arg\min_{\theta\in\Theta}\|\widehat{\ell}(\theta)\|_\infty$ in the second inequality, and that $A\theta^\star = b$ in the last line. Combining the above, we obtain

$$\begin{aligned}\|Q^\pi - \phi^\top\widehat{\theta}\|_\infty &\leq 2\sqrt{d}B_\phi\cdot\left\|A^{-1}\right\|_2\cdot\varepsilon_{\mathsf{stat}} \\ &= 6\sqrt{d}\cdot\left\|A^{-1}\right\|_2\cdot\max\{R_{\max}, B_\phi\}^2\cdot\sqrt{\frac{\log(2d|\Theta|\delta^{-1})}{n}},\end{aligned}$$

as desired. We now establish the concentration result of Equation (9).

Concentration results. For $j\in[d]$, let $\phi_j(s,a)\in\mathbb{R}$ refer to the j'th entry of the vector. For any (s,a,s') and θ, define

$$B^\pi(s,a,s';\theta) := \phi^\top(s,a)\theta - \gamma\phi^\top(s',\pi)\theta - r(s,a)$$

Recall that $\|\phi(s,a)\|_2 \leq B_\phi$ for all (s,a) and that $\|\theta\|_2 \leq B_\Theta$ for all $\theta\in\Theta$. We have that, for all $j\in[d]$, $\theta\in\Theta$, and $s,a\in\mathcal{S}\times\mathcal{A}$ we have that $\phi_j(s,a)B^\pi(s,a,s';\theta)$ is bounded, since:

$$\begin{aligned}\phi_j(s,a)\big(\phi^\top(s,a)\theta - \gamma\phi^\top(s',\pi)\theta - r(s,a)\big) &\leq \|\phi(s,a)\|_\infty(\|\phi(s,a)\|_2\|\theta\|_2 + \gamma\|\phi(s',\pi)\|_2\|\theta\|_2 + R_{\max}) \\ &\leq \max_{s,a}\|\phi(s,a)\|_2\left(\max_{s,a}\|\phi(s,a)\|_2\|\theta\|_2 + \gamma\max_{s,a}\|\phi(s,a)\|_2\|\theta\|_2 + R_{\max}\right) \\ &\leq (1+\gamma)B_\phi^2 + R_{\max}B_\phi \\ &\leq 3\max\{B_\phi, R_{\max}\}^2.\end{aligned}$$

Thus, from Hoeffding's inequality and a union bound, we have that for all $j\in[d]$ and $\theta\in\Theta$:

$$\left|\mathbb{E}_\mu\left[\phi^j(s,a)B^\pi(s,a,s';\theta)\right] - \widehat{\mathbb{E}}_\mu\left[\phi^j(s,a)B^\pi(s,a,s';\theta)\right]\right| \leq 3\max\{B_\phi, R_{\max}\}^2\sqrt{\frac{2\log(d|\Theta|\delta^{-1})}{n}} = \varepsilon_{\mathsf{stat}},$$

with probability at least $1-\delta$. As a result, we can write

$$\begin{aligned}\left\|\ell(\theta) - \widehat{\ell}(\theta)\right\|_\infty &= \left\|\mathbb{E}_\mu\left[\phi(s,a)\big(\phi^\top(s,a)\theta - \gamma\phi^\top(s',\pi)\theta - r(s,a)\big)\right] - \widehat{\mathbb{E}}_\mu\left[\phi(s,a)\big(\phi^\top(s,a)\theta - \gamma\phi^\top(s',\pi)\theta - r(s,a)\big)\right]\right\|_\infty \\ &= \left\|\mathbb{E}_\mu[\phi(s,a)B^\pi(s,a,s';\theta)] - \widehat{\mathbb{E}}_\mu[\phi(s,a)B^\pi(s,a,s';\theta)]\right\|_\infty \\ &\leq \|\mathbf{1}\cdot\varepsilon_{\mathsf{stat}}\|_\infty \\ &\leq \varepsilon_{\mathsf{stat}}.\end{aligned}$$

This concludes the proof. □

B.2. Proof of Theorem 2

Proof. We first note that the proposed algorithm is equivalent to the following tournament procedure:

- $\forall i\in[m], j\neq i$:
  - Define $\phi_{i,j}(s,a) := [Q_i(s,a), Q_j(s,a)]^\top$ and associated $\hat{A}_{i,j}$ matrix and $\hat{b}_{i,j}$ vector (Eq. 3)
  - Define $\hat{\ell}_{i,j} = \hat{A}_{i,j}e_1 - \hat{b}_{i,j}\in\mathbb{R}^2$
- Pick $\arg\min_{i\in[m]}\max_{j\neq i}\|\hat{\ell}_{i,j}\|_\infty$

Let $i^\star \in [m]$ denote the index of Q^π in the enumeration of $\mathcal{Q}$. We start with the upper bound

$$|J_{M^\star}(\pi) - \mathbb{E}_{s\sim d_0}[Q_{\hat{i}}(s,\pi)]| = |\mathbb{E}_{s\sim d_0}[Q^\pi(s,\pi)] - \mathbb{E}_{s\sim d_0}[Q_{\hat{i}}(s,\pi)]| \le \|Q^\pi(\cdot) - Q_{\hat{i}}(\cdot)\|_\infty.$$

Let $\ell_{i,j} \coloneqq A_{i,j}e_1 - b_{i,j}$ denote the population loss. We recall the concentration result from Equation (9), which, for any fixed i and j, implies:

$$\|\ell_{i,j} - \widehat{\ell}_{i,j}\|_\infty \le \varepsilon_{\mathsf{stat}} = 3 \cdot \max\{B_\phi, R_{\max}\}^2 \cdot \sqrt{\frac{\log(2d\delta^{-1})}{n}},$$

with probability at least $1-\delta$. This further implies $|\|\ell_{i,j}\|_\infty - \|\hat{\ell}_{i,j}\|_\infty| \le \varepsilon_{\mathsf{stat}}$. Taking a union bound over all (i,j) where either i or j equal $i^\star$, this implies that

$$\|\ell_{i,j} - \widehat{\ell}_{i,j}\|_\infty \le \varepsilon_{\mathsf{stat}} = 3 \cdot \max\{B_\phi, R_{\max}\}^2 \cdot \sqrt{\frac{\log(4dm\delta^{-1})}{n}} \quad \forall (i,j) \in ([m]\times\{i^\star\}) \cup (\{i^\star\}\times[m])$$

If $\hat{i} = i^\star$ then we are done. If not, then there exists a comparison in the tournament where $i = \hat{i}$ and $j = i^\star$. For these features $\phi_{\hat{i},i^\star}(s,a) = [Q_{\hat{i}}(s,a), Q_{i^\star}(s,a)]^\top$, we have:

$$\begin{aligned}
\|Q^\pi(\cdot) - Q_{\hat{i}}(\cdot)\|_\infty &= \|\phi_{\hat{i},i^\star}^\top(e_2 - e_1)\|_\infty \\
&= \|\phi_{\hat{i},i^\star}^\top A_{\hat{i},i^\star}^{-1} A_{\hat{i},i^\star}(e_2 - e_1)\|_\infty \\
&= \max_{s,a} |\phi_{\hat{i},i^\star}^\top(s,a) A_{\hat{i},i^\star}^{-1} A_{\hat{i},i^\star}(e_2 - e_1)| \\
&\le \left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \|A_{\hat{i},i^\star}^{-1}\|_2 \|A_{\hat{i},i^\star}(e_2 - e_1)\|_2 \\
&\le \sqrt{d}\left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \|A_{\hat{i},i^\star}^{-1}\|_2 \|A_{\hat{i},i^\star}(e_2 - e_1)\|_\infty \\
&= \sqrt{d}\left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \|A_{\hat{i},i^\star}^{-1}\|_2 \|A_{\hat{i},i^\star}e_1 - b_{\hat{i},i^\star}\|_\infty \\
&= \sqrt{d}\left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \|A_{\hat{i},i^\star}^{-1}\|_2 \|\ell_{\hat{i},i^\star}\|_\infty \\
&\le \sqrt{d}\left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \|A_{\hat{i},i^\star}^{-1}\|_2 \left(\|\hat{\ell}_{\hat{i},i^\star}\|_\infty + \varepsilon_{\mathsf{stat}}\right) \\
&\le \sqrt{d}\left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \|A_{\hat{i},i^\star}^{-1}\|_2 \left(\max_{j\in[m]\setminus\{i^\star\}}\|\hat{\ell}_{\hat{i},j}\|_\infty + \varepsilon_{\mathsf{stat}}\right) \\
&\le \sqrt{d}\left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \|A_{\hat{i},i^\star}^{-1}\|_2 \left(\max_{j\in[m]\setminus\{i^\star\}}\|\hat{\ell}_{i^\star,j}\|_\infty + \varepsilon_{\mathsf{stat}}\right) \\
&\le \sqrt{d}\left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \|A_{\hat{i},i^\star}^{-1}\|_2 \left(\max_{j\in[m]\setminus\{i^\star\}}\|\ell_{i^\star,j}\|_\infty + 2\varepsilon_{\mathsf{stat}}\right) \\
&\le 2\sqrt{d}\left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \|A_{\hat{i},i^\star}^{-1}\|_2 \varepsilon_{\mathsf{stat}}. \\
&\le 2\sqrt{d}\left(\max_{s,a}\|\phi_{\hat{i},i^\star}(s,a)\|_2\right) \max_{i\in[m]\setminus\{i^\star\}} \frac{1}{\sigma_{\min}(A_{i,i^\star})} \varepsilon_{\mathsf{stat}}.
\end{aligned}$$

To conclude, we note that $d=2$ in our application and that $\max_{s,a}\|\phi_{i,j}(s,a)\|_2^2 = Q_i^2(s,a) + Q_j^2(s,a) \le 2V_{\max}^2$. Plugging in the value for $\varepsilon_{\mathsf{stat}}$, this gives a final bound of

$$\begin{aligned}
|J_{M^\star}(\pi) - \mathbb{E}_{s\sim d_0}[Q_{\hat{i}}(s,\pi)]| &\le 4V_{\max} \max_{i\in[m]\setminus\{i^\star\}} \frac{1}{\sigma_{\min}(A_{i,i^\star})} \varepsilon_{\mathsf{stat}} \\
&= 24V_{\max}^3 \max_{i\in[m]\setminus\{i^\star\}} \frac{1}{\sigma_{\min}(A_{i,i^\star})} \sqrt{\frac{\log(4dm\delta^{-1})}{n}}.
\end{aligned}$$

□

B.3. Proof of Theorem 4

We bound

$$\begin{aligned}
J(\pi) - \mathbb{E}_{d_0}\left[\widehat{Q}(s,\pi)\right] &= \mathbb{E}_{d_0,\pi}\left[Q^\pi(s,a) - \widehat{Q}(s,a)\right] \\
&= \frac{1}{1-\gamma}\,\mathbb{E}_{d^\pi}\left[Q^\pi(s,a) - \gamma Q^\pi(s',\pi) - \widehat{Q}(s,a) - \gamma\widehat{Q}(s',\pi)\right] \\
&= \frac{1}{1-\gamma}\,\mathbb{E}_{d^\pi}\left[Q^\pi(s,a) - [\mathcal{T}^\pi Q^\pi](s,a) - \widehat{Q}(s,a) + \left[\mathcal{T}^\pi \widehat{Q}\right](s,a)\right] \\
&= \frac{1}{1-\gamma}\,\mathbb{E}_{d^\pi}\left[\left[\mathcal{T}^\pi \widehat{Q}\right](s,a) - \widehat{Q}(s,a)\right] \\
&\le \frac{1}{1-\gamma}\sqrt{\mathcal{C}^\pi \cdot \mathbb{E}_\mu\left[\left(\left[\mathcal{T}^\pi \widehat{Q}\right](s,a) - \widehat{Q}(s,a)\right)^2\right]}
\end{aligned}$$

where the second line follows from Bellman flow. Now we consider the term under the square root, and let $\widehat{g}_{\widehat{Q}} = \arg\min_{g\in\mathcal{G}_{\widehat{Q}}} \widehat{\ell}(g,\widehat{Q})$.

$$\mathbb{E}_\mu\left[\left(\left[\mathcal{T}^\pi \widehat{Q}\right](s,a) - \widehat{Q}(s,a)\right)^2\right] \le 2\cdot \underbrace{\mathbb{E}_\mu\left[\left(\left[\mathcal{T}^\pi \widehat{Q}\right](s,a) - \widehat{g}_{\widehat{Q}}(s,a)\right)\right]^2}_{\text{(T1)}} + 2\cdot \underbrace{\mathbb{E}_\mu\left[\left(\widehat{g}_{\widehat{Q}}(s,a) - \widehat{Q}(s,a)\right)^2\right]}_{\text{(T2)}}$$

We consider each term above individually. $(T1)$ is the regression error between $\widehat{g}_Q$ and the population regression solution $\mathcal{T}^\pi Q$, which we can control using well-established bounds. The second term $(T2)$ measure how close the Q-value is to its estimated Bellman backup. To bound these two terms we utilize the following results. The first controls the error between the squared-loss minimizer $\widehat{g}_Q$ and the population solution $\mathcal{T}^\pi Q$, and is adapted from [XJ21].

Lemma 6 (Lemma 9 from [XJ21])**.** *Suppose that we have $|g|_\infty \le V_{\max}$ for all $g\in\mathcal{G}_Q$ and $Q\in\mathcal{Q}$, and define*

$$\widehat{g}_Q := \operatorname*{arg\,min}_{g\in\mathcal{G}_Q} \mathbb{E}_{\mathcal{D}}\left[(g(s,a) - r - \gamma Q(s',\pi))^2\right].$$

Then with probability at least $1-\delta$, for all $i\in[m]$ we have

$$\mathbb{E}_\mu\left[(\widehat{g}_Q(s,a) - [\mathcal{T}^\pi Q](s,a))^2\right] \le \frac{16V_{\max}^2\log\left(\frac{2m}{\delta}\right)}{n} := \varepsilon_{\mathsf{reg}}^2.$$

The second controls the error of estimating the objective for choosing $\widehat{i}$ from finite samples, and a proof is included at the end of this section.

Lemma 7 (Objective estimation error)**.** *Suppose that we have $\|g\|_\infty \le V_{\max}$ for all $g\in\mathcal{G}_Q$ and $Q\in\mathcal{Q}$. Then with probability at least $1-\delta$, for all $g\in\mathcal{G}_Q$ and $Q\in\mathcal{Q}$ we have*

$$\max\left\{\frac{1}{2}\cdot\mathbb{E}_\mu\left[(g(s,a)-Q(s,a))^2\right] - \mathbb{E}_{\mathcal{D}}\left[(g(s,a)-Q(s,a))^2\right],\right.$$
$$\left.\mathbb{E}_{\mathcal{D}}\left[(g(s,a)-Q(s,a))^2\right] - \frac{3}{2}\cdot\mathbb{E}_\mu\left[(g(s,a)-Q(s,a))^2\right]\right\} \le \frac{3V_{\max}^2\log\left(\frac{2m}{\delta}\right)}{n} := \varepsilon_{\mathsf{obj}}.$$

Using lem:model-reg-concentration, we directly obtain that with probability at $1-\delta$,

$$\text{(T1)} \le \varepsilon_{\mathsf{reg}}^2.$$

By leveraging lem:model-obj-concentration, we have that with probability at least $1-\delta$,

$$\begin{aligned}
(\mathrm{T2}) &= \mathbb{E}_\mu\left[\left(\widehat{g}_{\widehat{Q}}(s,a) - \widehat{Q}(s,a)\right)^2\right] \\
&\le 2\cdot\varepsilon_{\mathsf{obj}} + 2\cdot\mathbb{E}_{\mathcal{D}}\left[\left(\widehat{g}_{\widehat{Q}}(s,a) - \widehat{Q}(s,a)\right)^2\right] \\
&\le 2\cdot\varepsilon_{\mathsf{obj}} + 2\cdot\mathbb{E}_{\mathcal{D}}\left[\left(\widehat{g}_{Q^\pi}(s,a) - Q^\pi(s,a)\right)^2\right] \\
&\le 4\cdot\varepsilon_{\mathsf{obj}} + 3\cdot\mathbb{E}_{\mu}\left[\left(\widehat{g}_{Q^\pi}(s,a) - Q^\pi(s,a)\right)^2\right] \\
&= 4\cdot\varepsilon_{\mathsf{obj}} + 3\cdot\mathbb{E}_{\mu}\left[\left(\widehat{g}_{Q^\pi}(s,a) - [\mathcal{T}^\pi Q^\pi](s,a)\right)^2\right] \\
&\le 4\cdot\varepsilon_{\mathsf{obj}} + 3\cdot\varepsilon_{\mathsf{reg}}^2
\end{aligned}$$

where in the first inequality we apply Lemma 7 (by lower bounding the LHS with the first expression in the $\max$); in the second we use the Q-value realizability assumption $Q^\pi\in\mathcal{Q}$ with the fact that $\widehat{Q}$ is the minimizer of the empirical objective; and in the third we again apply Lemma 7 (now lower bounding the LHS with the second expression in the $\max$). Then we use the identity that $Q^\pi = \mathcal{T}^\pi Q^\pi$, and apply the squared-loss regression guarantee. The bounds for (T1) and (T2) mean that

$$\mathbb{E}_\mu\left[\left(\left[\mathcal{T}^\pi\widehat{Q}\right](s,a) - \widehat{Q}(s,a)\right)^2\right] \le 8\left(\varepsilon_{\mathsf{obj}} + \varepsilon_{\mathsf{reg}}^2\right),$$

resulting in the final estimation bound of

$$\begin{aligned}
J(\pi) - \mathbb{E}_{d_0}\left[\widehat{Q}(s,\pi)\right] &\le \frac{1}{1-\gamma}\sqrt{\mathcal{C}^\pi\cdot\mathbb{E}_\mu\left[\left(\left[\mathcal{T}^\pi\widehat{Q}\right](s,a) - \widehat{Q}(s,a)\right)^2\right]} \\
&\le \frac{1}{1-\gamma}\sqrt{8\cdot\mathcal{C}^\pi\cdot\left(\varepsilon_{\mathsf{obj}} + \varepsilon_{\mathsf{reg}}^2\right)}, \\
&= \frac{V_{\max}}{1-\gamma}\sqrt{\frac{152\cdot\mathcal{C}^\pi\cdot\log\left(\frac{2m}{\delta}\right)}{n}},
\end{aligned}$$

which holds with probability at least $1-2\delta$.

Proof of Lemma 7. Observe that the random variable $(g(s,a)-Q(s,a))^2 \in [-V_{\max}^2, V_{\max}^2]$, and

$$\begin{aligned}
\mathbb{V}_\mu\left[(g(s,a)-Q(s,a))^2\right] &\le \mathbb{E}_\mu\left[(g(s,a)-Q(s,a))^4\right] \\
&\le V_{\max}^2\cdot\mathbb{E}_\mu\left[(g(s,a)-Q(s,a))^2\right].
\end{aligned}$$

Then, applying Bernstein's inequality with union bound, we have that, for any $g\in\mathcal{G}_Q$ and $Q\in\mathcal{Q}$ with probability at least $1-\delta$,

$$\begin{aligned}
&\left|\mathbb{E}_\mu\left[(g(s,a)-Q(s,a))^2\right] - \mathbb{E}_{\mathcal{D}}\left[(g(s,a)-Q(s,a))^2\right]\right| \\
&\le \sqrt{\frac{4\mathbb{V}_\mu\left[(g(s,a)-Q(s,a))^2\right]\log\left(\frac{2m}{\delta}\right)}{n}} + \frac{V_{\max}^2\log\left(\frac{2m}{\delta}\right)}{n} \\
&\le \sqrt{\frac{4V_{\max}^2\mathbb{E}_\mu\left[(g(s,a)-Q(s,a))^2\right]\log\left(\frac{2m}{\delta}\right)}{n}} + \frac{V_{\max}^2\log\left(\frac{2m}{\delta}\right)}{n} \\
&\le \frac{\mathbb{E}_\mu\left[(g(s,a)-Q(s,a))^2\right]}{2} + \frac{3V_{\max}^2\log\left(\frac{2m}{\delta}\right)}{n}.
\end{aligned}$$

Expanding the absolute value on the LHS and rearranging, this then implies that

$$\frac{1}{2}\cdot \mathbb{E}_\mu\left[(g(s,a)-Q(s,a))^2\right] \le \mathbb{E}_\mathcal{D}\left[(g(s,a)-Q(s,a))^2\right] + \frac{3V_{\max}^2 \log\left(\frac{2m}{\delta}\right)}{n},$$
$$\mathbb{E}_\mathcal{D}\left[(g(s,a)-Q(s,a))^2\right] \le \frac{3}{2}\cdot \mathbb{E}_\mu\left[(g(s,a)-Q(s,a))^2\right] + \frac{3V_{\max}^2 \log\left(\frac{2m}{\delta}\right)}{n}.$$

Combining these statements completes the proof. □

B.4. Proof of Theorem 5

We bound

$$\begin{aligned}
J(\pi) - \mathbb{E}_{d_0}\left[\widehat{Q}(s,\pi)\right] &= \mathbb{E}_{d_0,\pi}\left[Q^\pi(s,a) - \widehat{Q}(s,a)\right] \\
&= \frac{1}{1-\gamma}\,\mathbb{E}_{d^\pi}\left[Q^\pi(s,a) - \gamma Q^\pi(s',\pi) - \widehat{Q}(s,a) - \gamma\widehat{Q}(s',\pi)\right] \\
&= \frac{1}{1-\gamma}\,\mathbb{E}_{d^\pi}\left[Q^\pi(s,a) - [\mathcal{T}^\pi Q^\pi](s,a) - \widehat{Q}(s,a) + \left[\mathcal{T}^\pi \widehat{Q}\right](s,a)\right] \\
&= \frac{1}{1-\gamma}\,\mathbb{E}_{d^\pi}\left[\left[\mathcal{T}^\pi \widehat{Q}\right](s,a) - \widehat{Q}(s,a)\right] \\
&\le \frac{\mathcal{C}_\infty^\pi}{1-\gamma}\cdot \mathbb{E}_\mu\left[\left|\left[\mathcal{T}^\pi \widehat{Q}\right](s,a) - \widehat{Q}(s,a)\right|\right] \\
&\le \max_{g\in\mathcal{G}_{\widehat{Q}}} \mathbb{E}_\mu\left[\operatorname{sgn}\left(\widehat{Q}(s,a) - g(s,a)\right)\left(\widehat{Q}(s,a) - r - \gamma\widehat{Q}(s',\pi)\right)\right]
\end{aligned}$$

By assumption, $\max_{q\in\mathcal{Q}}\|q\|_\infty \le V_{\max}$, and similarly $\max_{g\in\mathcal{G}_Q}\|g\|_\infty \le V_{\max}$ for all $Q\in\mathcal{Q}$. Then for any $Q\in\mathcal{Q}$ and $g\in\mathcal{G}_Q$ and $(s,a)\in\mathcal{S}\times\mathcal{A}$ and $r\in[0,R_{\max}]$,

$$\operatorname{sgn}(Q(s,a)-g(s,a))(Q(s,a)-r-\gamma Q(s',\pi)) \in [-V_{\max}, V_{\max}],$$

and, using Hoeffding's inequality, we have for all $Q\in\mathcal{Q}$ and $g\in\mathcal{G}_Q$ that, with probability at least $1-\delta$,

$$\begin{aligned}
&\left|\mathbb{E}_\mu\left[\operatorname{sgn}\left(\widehat{Q}(s,a) - g(s,a)\right)\left(\widehat{Q}(s,a) - r - \gamma\widehat{Q}(s',\pi)\right)\right] - \mathbb{E}_\mathcal{D}\left[\operatorname{sgn}\left(\widehat{Q}(s,a) - g(s,a)\right)\left(\widehat{Q}(s,a) - r - \gamma\widehat{Q}(s',\pi)\right)\right]\right| \\
&\qquad \le 2V_{\max}\sqrt{\frac{\log\left(\frac{2m}{\delta}\right)}{n}} := \varepsilon_{\mathsf{obj}}.
\end{aligned}$$

Then using this concentration in the last line of the previous block,

$$\begin{aligned}
J(\pi) - \mathbb{E}_{d_0}\left[\widehat{Q}(s,\pi)\right] &\le \max_{g\in\mathcal{G}_{\widehat{Q}}} \mathbb{E}_\mathcal{D}\left[\operatorname{sgn}\left(\widehat{Q}(s,a) - g(s,a)\right)\left(\widehat{Q}(s,a) - r - \gamma\widehat{Q}(s',\pi)\right)\right] + \varepsilon_{\mathsf{obj}} \\
&\le \max_{g\in\mathcal{G}_{Q^\pi}} \mathbb{E}_\mathcal{D}[\operatorname{sgn}(Q^\pi(s,a) - g(s,a))(Q^\pi(s,a) - r - \gamma Q^\pi(s',\pi))] + \varepsilon_{\mathsf{obj}} \\
&\le \max_{g\in\mathcal{G}_{Q^\pi}} \mathbb{E}_\mu[\operatorname{sgn}(Q^\pi(s,a) - g(s,a))(Q^\pi(s,a) - r - \gamma Q^\pi(s',\pi))] + 2\cdot\varepsilon_{\mathsf{obj}} \\
&= \max_{g\in\mathcal{G}_{Q^\pi}} \mathbb{E}_\mu[\operatorname{sgn}(Q^\pi(s,a) - g(s,a))(Q^\pi(s,a) - [\mathcal{T}^\pi Q^\pi](s,a))] + 2\cdot\varepsilon_{\mathsf{obj}} \\
&= \max_{g\in\mathcal{G}_{Q^\pi}} \mathbb{E}_\mu[\operatorname{sgn}(Q^\pi(s,a) - g(s,a))(Q^\pi(s,a) - Q^\pi(s,a))] + 2\cdot\varepsilon_{\mathsf{obj}} \\
&= 2\cdot\varepsilon_{\mathsf{obj}},
\end{aligned}$$

where in the first and third inequalities we apply the above concentration inequality, and in the second inequality we use the fact that $\widehat{Q}$ is the minimizer of the empirical objective, i.e.,

$$\widehat{Q} = \arg\min_{Q\in\mathcal{Q}} \max_{g\in\mathcal{G}_Q} \mathbb{E}_\mathcal{D}[\operatorname{sgn}(Q(s,a)-g(s,a))(Q(s,a)-r-Q(s',\pi))].$$

Combining the above inequalities, we obtain the theorem statement,

$$J(\pi) - \mathbb{E}_{d_0}\left[\widehat{Q}(s,\pi)\right] \le 2\cdot\mathcal{C}_\infty^\pi\cdot\varepsilon_{\mathsf{obj}}.$$

C. Experiment Details

C.1. Environment Setup: Noise and State Resetting

State Resetting. Monte-Carlo rollouts for Q-value estimation rely on the ability to (re)set the simulator to a particular state from the offline dataset. To the best of our knowledge, Mujoco environment does not natively support state resetting, and assigning values to the observation vector does not really change the underlying state. However, state resetting can still be implemented by manually assigning the values of the position vector `qpos` and the velocity vector `qvel`.

Noise. As mentioned in Section 6, we add noise to Hopper to create more challenging stochastic environments and create model selection tasks where candidate simulators have different levels of stochasticity. Here we provide the details about how we inject randomness into the deterministic dynamics of Hopper. Mujoco engine realizes one-step transition by leveraging `mjData.{ctrl, qfrc_applied, xfrc_applied}` objects [Dee], where `mjData.ctrl` corresponds to the action taken by our agent, and `mjData.{qfrc_applied, xfrc_applied}` are the user-defined perturbations in the joint space and Cartesian coordinates, respectively. To inject randomness into the transition at a noise level of σ, we first sample an isotropic Gaussian noise with variance σ^2 as the stochastic force in `mjData.xfrc[:3]` upon each transition, which jointly determines the next state with the input action `mjData.ctrl`, leaving the joint data `mjData.qfrc` intact.

C.2. Experiment Settings

MF/MB.G/N. The settings of different experiments are summarized in Table 1. We first run DDPG in the environment of $g = -30, \sigma = 32$, and obtain 15 deterministic policies $\{\pi_{0:14}\}$ from the checkpoints. The first 10 are used as target policies in MF.G/N experiments, and MB.G/N use fewer due to the high computational cost. For the main results (Section 6.1), the choice of $M^\star$ is usually the two ends plus the middle point of the grid ($\mathcal{M}_\mathbf{g}$ or $\mathcal{M}_\mathbf{n}$). The corresponding behavior policy is an epsilon-greedy version of one of the target policies, denoted as π_i^ϵ, which takes the deterministic action of $\pi_i(s)$ with probability 0.7, and add a unit-variance Gaussian noise to $\pi_i(s)$ with the remaining 0.3 probability.

MF/MB.Off.G/N. In the above setup, the behavior and the target policies all stem from the same DDPG training procedure. While these policies still have significant differences (see Figure 1L), the distribution shift is relatively mild. For the data coverage experiments (Section 6.3), we prepare a different set of behavior policies that intentionally offer poor coverage: these policies, denoted as π_i^{poor}, are obtained by running DDPG with a different neural architecture (than the one used for generating $\pi_{0:14}$) in a different environment of $g = -60, \sigma = 100$. We also provide the parallel of our main experiments in Figures 2 and 3 under these behavior policies with poor coverage in Appendix E.1.

MF.T.G. This experiment is for data coverage (Section 6.3), where $\mathcal{D}$ is a mixture of two datasets, one sampled from π_7 (which is the sole target policy being considered) and one from π_i^{poor} that has poor coverage. They are mixed together under different ratios as explained in Section 6.3.

	Gravity g	**Noise Level σ**	**Groundtruth Model $M^\star$ and Behavior Policy π_b**	**Target Policies Π**
MF.G	LIN$(-51, -9, 15)$	100	$\{(M_i, \pi_i^\epsilon), i \in \{0, 7, 14\}\}$	$\{\pi_{0:9}\}$
MF.N	-30	LIN$(10, 100, 15)$	$\{(M_i, \pi_i^\epsilon), i \in \{0, 7, 14\}\}$	$\{\pi_{0:9}\}$
MB.G	LIN$(-36, -24, 5)$	100	$\{(M_i, \pi_i^\epsilon), i \in \{0, 2, 4\}\}$	$\{\pi_{0:5}\}$
MB.N	-30	LIN$(10, 100, 5)$	$\{(M_i, \pi_i^\epsilon), i \in \{0, 2, 4\}\}$	$\{\pi_{0:5}\}$
MF.OFF.G	LIN$(-51, -9, 15)$	100	$\{(M_i, \pi_i^{\text{poor}}), i \in \{0, 7, 14\}\}$	$\{\pi_{0:9}\}$
MF.OFF.N	-30	LIN$(10, 100, 15)$	$\{(M_i, \pi_i^{\text{poor}}), i \in \{0, 7, 14\}\}$	$\{\pi_{0:9}\}$
MF.T.G	LIN$(-51, -9, 15)$	100	$\{(M_i, \pi_8 \,\&\, \pi_i^{\text{poor}}), i \in \{0, 7, 14\}\}$	$\{\pi_8\}$

Table 1. Details of experiment settings. LIN(a, b, n) (per numpy convention) refers to the arithmetic sequence with n elements, starting from a and ending in b (e.g. LIN$(0, 1, 6) = \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$).

D. Additional Experiment Results

E. Subgrid Studies and Coverage Experiments

Subgrid Studies. Figures 6 and 5 show more complete results for investigating the sensitivity to misspecification and gaps in Section 6.2 across 4 settings (good/poor coverage and gravity/noise grid).

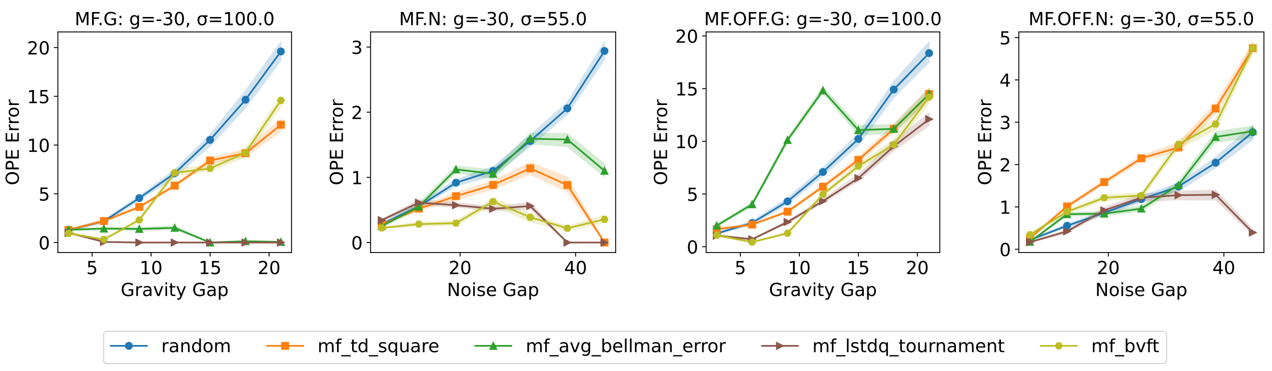

Figure 5. Subgrid studies for gaps. Plot **MF.N** is identical to Figure 4L.

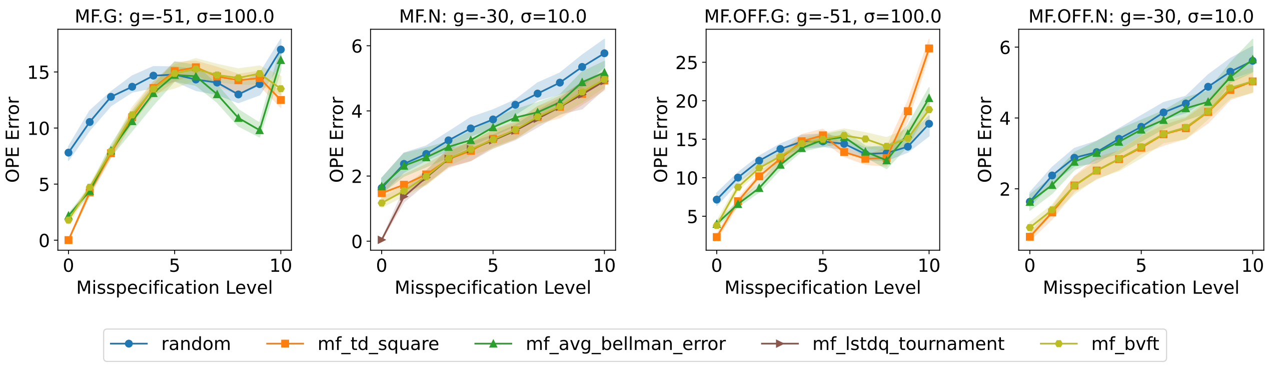

Figure 6. Subgrid studies for misspecification. Plot **MF.N** is identical to Figure 4M.

Data Coverage. Figure 7 shows more complete results for the data coverage experiment in Section 6.3, including more choices of $M^\star$.

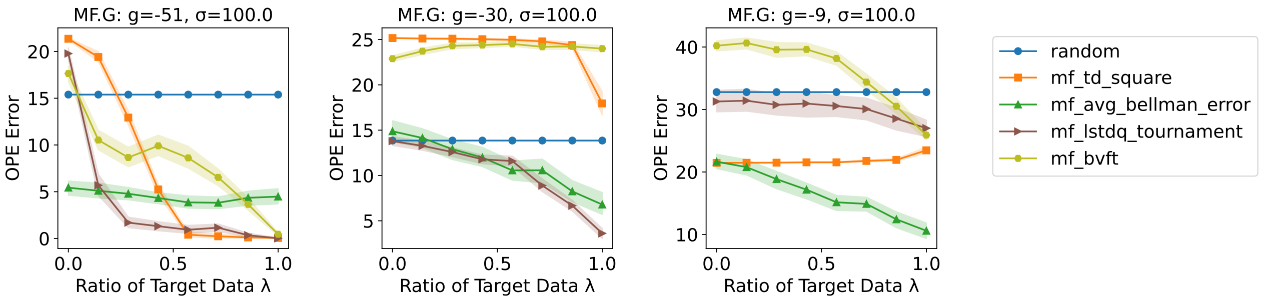

Figure 7. Data coverage results. Left figure is identical to Figure 4L.

E.1. Poor Coverage Results

We now show the counterpart of our model-free main results (Figure 2) under behavior policies that offer poor coverage. This makes the problem very challenging and no single algorithm have strong performance across the board. For example, naïve model-based demonstrate strong performance in **MF.OFF.G** (top row of Figure 8) and resilience to poor coverage, while still suffers catastrophic failures in **MF.OFF.N**. While LSTD-Tournament generally is more reliable than other methods, it also has worse-than-random performance in one of the environments in **MF.OFF.G**.

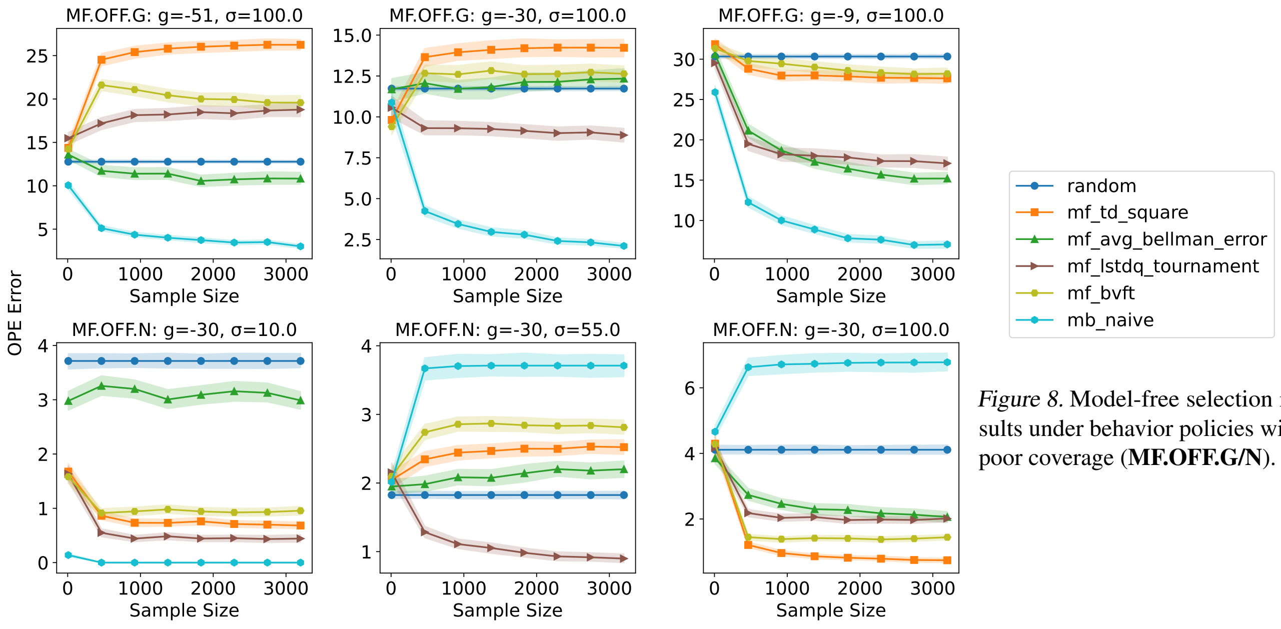

Figure 8. Model-free selection results under behavior policies with poor coverage (**MF.OFF.G/N**).

E.2. LSTDQ Family

As mentioned at the end of Section 3, our LSTD-Tournament can have several variants depending on how we design and transform the linear features. Here we compare 3 of them in Figure 9. The LSTD-Tournament method in all other figures corresponds to the "normalized_diff" version.

- **Vanilla:** $\phi_{i,j} = [Q_i, Q_j]$.
- **Normalized:** $\phi_{i,j} = [Q_i/c_i, Q_j/c_j]$, where $c_i = \sqrt{\mathbb{V}_{(s,a)\sim\mu}[Q_i(s,a)]}$ normalizes the discriminators to unit variance on the data distribution. In practice these variance parameters are estimated from data.
- **Normalized_diff:** $\phi_{i,j} = [Q_i/c_i, (Q_j - Q_i)/c_{j,i}]$, where c_i and $c_{j,i}$ performs normalization in the same way as above.

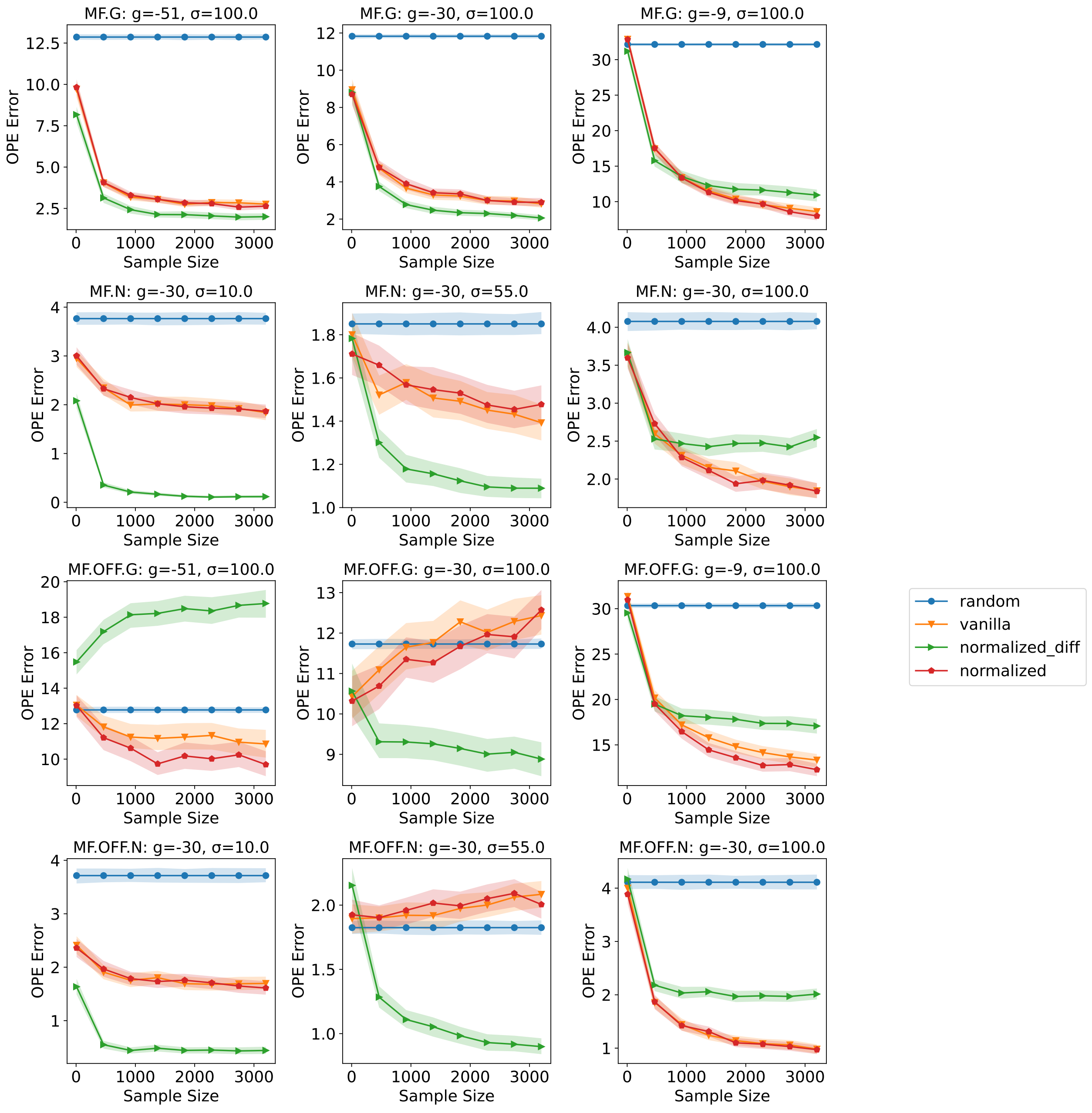

Figure 9. Comparison of variants of LSTD-Tournament.

E.3. $O(1)$ Rollouts

In our experiment design, we use a fairly significant number of rollouts $l = 128$ to ensure relatively accurate estimation of the Q-values. However, for the average Bellman error and the LSTD-Tournament algorithms, they enjoy convergence even when l is a constant. For example, consider the average Bellman error:

$$\mathbb{E}_{\mathcal{D}}[Q_i(s,a) - r - \gamma Q_i(s', \pi)],$$

which is an estimation of $\mathbb{E}_{\mu}[Q_i(s,a) - r - \gamma Q_i(s', \pi)]$. Thanks to its linearity in Q_i, replacing Q_i with its few-rollout (or even single-rollout) Monte-Carlo estimates will leave the unbiasedness of the estimator intact, and Hoeffding's inequality implies convergence as the sample size $n = |\mathcal{D}|$ increases, even when l stays as a constant, which is an advantage compared to other methods. That said, in practice, having a relatively large l can still be useful as it reduces the variance of each individual random variable that we average across $\mathcal{D}$, and the effect can be significant when n is relatively small.

A similar but slightly more subtle version of this property also holds for LSTD-Tournament. Take the vanilla version in Section 3 as example, we need to estimate

$$\mathbb{E}_{\mathcal{D}}[Q_j(s,a)(Q_i(s,a) - r - \gamma Q_i(s', \pi))].$$

Again, we can replace Q_j and Q_i with their Monte-Carlo estimates, as long as the Monte-Carlo trajectories for Q_i and Q_j are independent. This naturally holds in our implementation when $j \neq i$, but is violated when $j = i$ since $Q_j(s,a)$ and $Q_i(s,a)$ will share the same set of random rollouts, leading to biases. A straightforward resolution is to divide the Monte-Carlo rolllouts into two sets, and $Q_j(s,a)$ and $Q_i(s,a)$ can use different sets when $j \neq i$. We empirically test this procedure in Figure 10, where the OPE errors of average Bellman error and different variants of LSTD-Tournament are plotted against the number of rollouts l. In both the left and the middle plots, a relatively small number of rollouts suffices for good performance. However, the right plot still requires a large number of rollouts, potentially due to n not being sufficiently large.

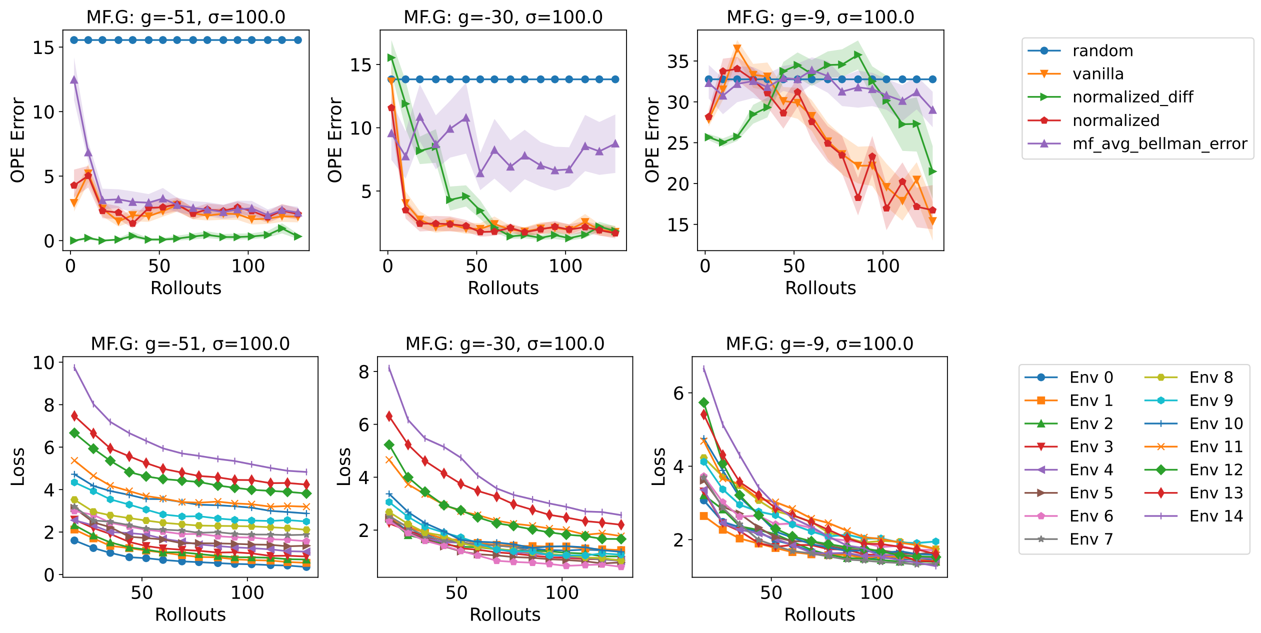

Figure 10. The effect of small rollouts in LSTD-Tournament methods. Sample size is fixed at $n = 3200$ and only l (the number of rollouts) varies. The top row shows the OPE error (i.e., final performance), whereas the bottom row shows the convergence of loss estimates for LSTD-Tournament as a function of rollouts.