Safe Policy Improvement with an Estimated Baseline Policy

Simão, Thiago D.; Laroche, Romain; Tachet des Combes, Rémi

Publication date
2020

Document Version
Final published version

Published in
Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems

Citation (APA)
Simão, T. D., Laroche, R., & Tachet des Combes, R. (2020). Safe Policy Improvement with an Estimated Baseline Policy. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems* (pp. 1269–1277). (AAMAS '20).. http://10.5555/3398761.3398908

Safe Policy Improvement with an Estimated Baseline Policy

Thiago D. Simão*
Delft University of Technology
The Netherlands
t.diassimao@tudelft.nl

Romain Laroche
Microsoft Research Montréal
Canada
romain.laroche@microsoft.com

Rémi Tachet des Combes
Microsoft Research Montréal
Canada
remi.tachet@microsoft.com

ABSTRACT

Previous work has shown the unreliability of existing algorithms in the batch Reinforcement Learning setting, and proposed the theoretically-grounded Safe Policy Improvement with Baseline Bootstrapping (SPIBB) fix: reproduce the baseline policy in the uncertain state-action pairs, in order to control the variance on the trained policy performance. However, in many real-world applications such as dialogue systems, pharmaceutical tests or crop management, data is collected under human supervision and the baseline remains unknown. In this paper, we apply SPIBB algorithms with a baseline estimate built from the data. We formally show safe policy improvement guarantees over the true baseline even without direct access to it. Our empirical experiments on finite and continuous states tasks support the theoretical findings. It shows little loss of performance in comparison with SPIBB when the baseline policy is given, and more importantly, drastically and significantly outperforms competing algorithms both in safe policy improvement, and in average performance.

ACM Reference Format:
Thiago D. Simão, Romain Laroche, and Rémi Tachet des Combes. 2020. Safe Policy Improvement with an Estimated Baseline Policy. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020*, IFAAMAS, 9 pages.

1 INTRODUCTION

Reinforcement Learning (RL) is a framework for sequential decision-making optimization. Most RL research focuses on the online setting, where the system directly interacts with the environment and learns from it [16, 31]. While this setting might be the most efficient in simulation and in uni-device system control such as drones or complex industrial flow optimization, most real-world tasks (RWTs) involve a distributed architecture. We may cite a few: distributed devices (Internet of Things), mobile/computer applications (games, dialogue systems), or distributed lab experiments (pharmaceutical tests, crop management). These RWTs entail a high parallellization of the trajectory collection and strict communication constraints both in bandwidth and in privacy [4]. Rather than spending a small amount of computational resource after each sample/trajectory collection, it is therefore more practical to collect a dataset using a behavioral (or baseline) policy, and then train a new policy from it. This setting is called *batch RL* [11].

Classically, batch RL algorithms apply dynamic programming on the samples in the dataset [3, 10]. Laroche et al. [13] showed that in finite-state Markov Decision Processes (MDPs), these algorithms all converge to the same policy: the one that is optimal in the MDP with the maximum likelihood given the batch of data. Petrik et al. [21] show that this policy is approximately optimal to the order of the inverse square root of the minimal state-action pairs count in the dataset. Unfortunately, Laroche et al. [13] show that even on very small tasks this minimal amount is almost always zero, and that, as a consequence, it gravely impairs the reliability of the approach: dynamic programming on the batch happens to return policies that perform terribly in the real environment. If a bad policy were to be run in distributed architectures such as the aforementioned ones, the consequences would be disastrous as it would jeopardize a high number of systems, or even lives.

Several attempts have been made to design reliable batch RL algorithms, starting with robust MDPs [6, 18], which consist of considering the set of plausible MDPs given the dataset, and then find the policy for which the minimal performance over the robust MDPs set is maximal. The algorithm however tends to converge to policies that are unnecessarily conservative.

Xu and Mannor [32] considered robust regret over the optimal policy: the algorithm searches for the policy that minimizes the maximal gap with respect to the optimal performance in every MDP in the robust MDPs. However, they proved that evaluating the robust optimal regret for a fixed policy is already NP-complete with respect to the state and action sets' size and the uncertainty constraints in the robust MDPs set.

Later, Petrik et al. [21] considered the regret with respect to the behavioural policy performance over the robust MDPs set. The behavioural policy is called *baseline* in this context. Similarly, they proved that simply evaluating the robust baseline regret is already NP-complete. Concurrently, they also proposed, without theoretical grounding, the Reward-adjusted MDP algorithm (RaMDP), where the immediate reward for each transition in the batch is penalized by the inverse square root of the number of samples in the dataset that have the same state and action than the considered transition.

Recently, Laroche et al. [13] proposed Safe Policy Improvement with Baseline Bootstrapping (SPIBB), the first tractable algorithm with approximate policy improvement guarantees. Its principle consists in guaranteeing safe policy improvement by constraining the trained policy as follows: it has to reproduce the baseline policy in the uncertain state-action pairs. Nadjahi et al. [17] further improved SPIBB's empirical performance by adopting soft constraints instead. Related to this track of research, Simão and Spaan [26, 27] also developed SPIBB algorithms specifically for factored MDPs. Note that this thread of research is very distinct from online safe policy iteration, such as [7, 20, 22–24], because the online setting allows them to perform very conservative updates.

*Work done while interning at Microsoft Research Montréal.

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand.

Concurrently to robust approaches described above, another tractable and theoretically-grounded family of frequentist algorithms appeared under the name of High Confidence Policy Improvement [14, 19, 28, HCPI], relying on importance sampling estimates of the trained policy performance. The algorithm by Mandel et al. [14], based on concentration inequalities, tends to be conservative and requires hyper parameters optimization. The algorithms by Thomas et al. [29] rely on the assumption that the importance sampling estimate is normally distributed which is false when the number of trajectories is small. The algorithm by Paduraru [19] is based on bias corrected and accelerated bootstrap and tends to be too optimistic. In contrast with the robust approaches, from robust MDPs to Soft-SPIBB, HCPI may be readily applied to infinite MDPs with guarantees. However, it is well known that the importance sampling estimates have high variance, exponential with the horizon of the MDP. The SPIBB algorithm has a linear horizon dependency, given a fixed known maximal value and the common horizon/discount factor equivalence: $H = \frac{1}{1-\gamma}$ [8]. Soft-SPIBB suffers a cubic upper bound but the empirical results rather indicate a linear dependency.

Nadjahi et al. [17] perform a benchmark on randomly generated finite MDPs, baselines, and datasets. They report that the SPIBB and Soft-SPIBB algorithms are significantly the most reliable, and tie with RaMDP as the highest average performing algorithms. Additionally, they perform a benchmark on a continuous state space task, where the SPIBB and Soft-SPIBB algorithms significantly outperform RaMDP and Double-DQN [30] both in reliability and average performance. Soft-SPIBB particularly shines in the continuous state experiments.

Despite these appealing results, there is a caveat: the SPIBB and Soft-SPIBB algorithms requires the baseline policy as input. However, the behavior policy is not always available. Consider for instance application involving human interactions, such as dialogue systems [25] and the medical sector. In these situations it is common to have access to the observations and actions that were taken in a trajectory but not the policy that was followed. To overcome this issue, *we investigate the use of SPIBB and Soft-SPIBB algorithms in the setting where the baseline policy is unknown.*

Our aim is to answer a very natural question arising from the existing SPIBB analysis, whether access to the baseline is required or not. Therefore, our contributions are threefold:

(1) We formally prove safety bounds for SPIBB and Soft-SPIBB algorithms with estimated baseline policies in finite MDPs (Section 3).
(2) We consolidate the theoretical results with empirical results in finite randomly generated MDPs, unknown baselines, and datasets (Section 4.1, https://github.com/RomainLaroche/SPIBB).
(3) We apply the method on a continuous state task by investigating two types of behavioural cloning, and show that it outperforms competing algorithms by a large margin, in particular on small datasets (Section 4.2, https://github.com/rems75/SPIBB-DQN).

In summary, our results bring the SPIBB framework a step closer to many RWTs where the behavior policy is unknown.

2 BACKGROUND

This section reviews the previous technical results relevant for this work.

2.1 Preliminaries

A Markov Decision Process (MDP) is the standard formalism to model sequential decision making problems in stochastic environments. An MDP M is defined as $M = \langle \mathcal{X}, \mathcal{A}, P, R, \gamma \rangle$, where $\mathcal{X}$ is the state space, $\mathcal{A}$ is the set of actions the agent can execute, $P : \mathcal{X} \times \mathcal{A} \to \Delta_{\mathcal{X}}$ is the stochastic transition function, $R : \mathcal{X} \times \mathcal{A} \to [-R_{\max}, R_{\max}]$ is a stochastic immediate reward function, γ is the discount factor. Without loss of generality, we assume that the initial state is deterministically x_i.

A policy $\pi : \mathcal{X} \to \Delta_{\mathcal{A}}$ represents how the agent interacts with the environment. The value of a policy π starting from a state $x \in \mathcal{X}$ is given by the expected sum of discounted future rewards:

$$V_M^{\pi}(x) = \mathbb{E}_{\pi, M, x_0 = x}\left[\sum_{t \geq 0} \gamma^t R(x_t, a_t)\right]. \tag{1}$$

Therefore, the performance of a policy, denoted $\rho(\pi, M)$, is the value in the initial state x_i. The goal of a reinforcement learning agent is to find a policy $\pi : \mathcal{X} \to \Delta_{\mathcal{A}}$ that maximizes its expected sum of discounted rewards, however the agent does not have access to the dynamics of the true environment $M^* = \langle \mathcal{X}, \mathcal{A}, P^*, R^*, \gamma \rangle$.

In the batch RL setting, the algorithm receives as an input the dataset of previous transitions collected by executing a baseline policy π_b: $\mathcal{D} = \langle x_k, a_k, r_k, x'_k, t_k \rangle_{k \in [\![1, |\mathcal{D}|]\!]}$, where the starting state of the transition is $x_k = x_i$ if $t_k = 0$ and $x_k = x'_{k-1}$ otherwise, $a_k \sim \pi_b(\cdot|x_k)$ is the performed action, $r_k \sim R(x_k, a_k)$ is the immediate reward, $x'_k \sim P(\cdot|x_k, a_k)$ is the reached state, and the trajectory-wise timestep is $t_k = 0$ if the previous transition was final and $t_k = t_{k-1} + 1$ otherwise.

We build from a dataset $\mathcal{D}$ the Maximum Likelihood Estimate (MLE) MDP $\widehat{M} = \langle \mathcal{X}, \mathcal{A}, \widehat{P}, \widehat{R}, \gamma \rangle$, as follows:

$$\widehat{P}(x'|x, a) = \frac{N_{\mathcal{D}}(x, a, x')}{N_{\mathcal{D}}(x, a)},$$
$$\widehat{R}(x, a) = \frac{\sum_{\langle x_j = x, a_j = a, r_j, x'_j \rangle \in \mathcal{D}} r_j}{N_{\mathcal{D}}(x, a)},$$

where $N_{\mathcal{D}}(x, a)$ and $N_{\mathcal{D}}(x, a, x')$ are the state-action pair counts and next-state counts in the dataset $\mathcal{D}$. We also consider the robust MDPs set Ξ, *i.e.* the set of plausible MDPs such that the true environment MDP M^* belongs to it with high probability $1 - \delta$:

$$\Xi = \{M = \langle \mathcal{X}, \mathcal{A}, R, P, \gamma \rangle \text{ s.t. } \forall x, a, \left.\begin{array}{l} ||P(\cdot|x, a) - \widehat{P}(\cdot|x, a)||_1 \leq e_\delta(x, a), \\ |R(x, a) - \widehat{R}(x, a)| \leq e_\delta(x, a) R_{max} \end{array}\right\}, \tag{2}$$

where $e_\delta(x, a)$ is a model error function on the estimates of $\widehat{M}$ for a state-action pair (x, a), which is classically upper bounded with concentration inequalities.

In the next section, we discuss an objective for these algorithms that aims to guarantee a safe policy improvement for the new policy.

2.2 Approximate Safe Policy Improvement

Laroche et al. [13] investigate the setting where the agent receives as input the dataset $\mathcal{D}$ and must compute a new policy π that approximately improves with high probability the baseline. Formally, the safety criterion can be defined as:

$$\mathbb{P}\left(\rho(\pi, M^*) \geq \rho(\pi_b, M^*) - \zeta\right) \geq 1 - \delta, \tag{3}$$

where ζ is a hyper-parameter indicating the improvement approximation and $1-\delta$ is the high confidence hyper-parameter. Petrik et al. [21] demonstrate that the optimization of this objective is NP-hard. To make the problem tractable, Laroche et al. [13] end up considering an approximate solution by maximizing the policy in the MLE-MDP while constraining the policy to be approximately improving in the robust MDPs set Ξ. More formally, they seek:

$$\operatorname*{argmax}_{\pi} \rho(\pi, \widehat{M}), \text{ s.t. } \forall M \in \Xi, \rho(\pi, M) \geq \rho(\pi_b, M) - \zeta.$$

Given a hyper-parameter $N_\wedge$, their algorithm Π_b-SPIBB constrains the policy search to the set Π_b of policies that reproduce the baseline probabilities in the state-action pairs that are present less than $N_\wedge$ times in the dataset $\mathcal{D}$:

$$\Pi_b = \{\pi \,|\, \pi(a|x) = \pi_b(a|x) \text{ if } N_{\mathcal{D}}(x,a) < N_\wedge\}. \tag{4}$$

We now recall the safe policy improvement guaranteed by the algorithm Π_b-SPIBB:

Theorem 2.1 (Safe policy improvement with baseline bootstrapping). *Let π_b^* be the optimal policy constrained to Π_b in the MLE-MDP. Then, π_b^* is a ζ-approximate safe policy improvement over the baseline π_b with high probability $1-\delta$, where:*

$$\zeta = \frac{4V_{max}}{1-\gamma}\sqrt{\frac{2}{N_\wedge}\log\frac{2|\mathcal{X}||\mathcal{A}|2^{|\mathcal{X}|}}{\delta}} - \rho(\pi_b^*, \widehat{M}) + \rho(\pi_b, \widehat{M}).$$

Our work also considers the algorithm Soft-SPIBB [17], that constrains the policy search such that the cumulative state-local error never exceeds ϵ, with ϵ a fixed hyper-parameter. More formally, the policy constraint is expressed as follows:

$$\Pi_\sim = \left\{\pi \,\middle|\, \forall x, \sum_{a\in\mathcal{A}} e_\delta(x,a)\left|\pi(a|x) - \pi_b(a|x)\right| \leq \epsilon\right\}. \tag{5}$$

Under some assumptions, Nadjahi et al. [17] demonstrate a looser safe policy improvement bound. Nevertheless, the policy search is less constrained and their empirical evaluation reveals that Soft-SPIBB safely finds better policies than SPIBB.

Both algorithms presented in this section assume the behavior policy π_b is known and can be used during the computation of a new policy. In the next section, we get to the main contribution of this paper, where we investigate how these algorithms can be applied when π_b is not given.

3 BASELINE ESTIMATES

In this section, we consider that the true baseline is unknown and implement a baseline estimate in order for the SPIBB and Soft-SPIBB algorithms to still be applicable. Before we start our analysis, we present an auxiliary lemma.

Let $d_M^\pi(x,a)$ be the discounted sum of visits of state-action pair $(x,a) \in \mathcal{X}\times\mathcal{A}$ while following policy π in MDP M and $d_{\mathcal{D}}$ is the state-action discounted distribution in dataset $\mathcal{D}$.

Lemma 3.1. *Considering that the trajectories in $\mathcal{D}$ are i.i.d. sampled, the L_1 deviation of the empirical discounted sum of visits of state-action pairs is bounded. We have the following concentration bound:*

$$\mathbb{P}\left(\left\|d_{M^*}^{\pi_b} - d_{\mathcal{D}}\right\|_1 (1-\gamma) \geq \varepsilon\right) \leq \left(2^{|\mathcal{X}||\mathcal{A}|} - 2\right)\exp\left(-\frac{N\varepsilon^2}{2}\right), \tag{6}$$

where N is the number of trajectories in $\mathcal{D}$.

Proof. Let $\mathcal{T} = (\mathcal{X}\times\mathcal{A})^{\mathbb{N}}$ denote the set of trajectories and $T = (T_1, \ldots, T_N)$ be a set of N $\mathcal{T}$-valued random variables. For a given $E \subset \mathcal{X}\times\mathcal{A}$, we define the function f_E on $\mathcal{T}$ as:

$$f_E(T) = f_E(T_1, \ldots, T_N) \coloneqq (1-\gamma)\sum_{i=1}^{N}\sum_{t\geq 0}\gamma^t \mathbb{1}(T_i^t \in E),$$

where T_i^t is the state-action pair on trajectory i at time t. In particular, we have that

$$f_E(\mathcal{D}) = N(1-\gamma)d_{\mathcal{D}}(E) \text{ and} \tag{7}$$

$$\mathbb{E}[f_E(T)] = N(1-\gamma)d_{M^*}^{\pi_b}(E), \tag{8}$$

where $d_{\mathcal{D}}(E)$ and $d_{M^*}^{\pi_b}(E)$ denote the mass of set E under $d_{\mathcal{D}}$ and $d_{M^*}^{\pi_b}$ respectively.

For two sets T and T' differing only on one trajectory, say the k-th, we have:

$$|f_E(T) - f_E(T')| = |(1-\gamma)\sum_{t\geq 0}\gamma^t\left(\mathbb{1}(T_k^t \in E) - \mathbb{1}(T'^t_k \in E)\right)| \leq 1.$$

This allows us to apply the independent bounded difference inequality by McDiarmid [15, Theorem 3.1], which gives us:

$$\mathbb{P}\left(f_E(T) - \mathbb{E}[f_E(T)] \geq \bar{\varepsilon}\right) \leq \exp\left(-2\frac{\bar{\varepsilon}^2}{N}\right). \tag{9}$$

We know that

$$\left\|d_{M^*}^{\pi_b} - d_{\mathcal{D}}\right\|_1 (1-\gamma) = \max_{E\subset\mathcal{X}\times\mathcal{A}} 2(1-\gamma)(d_{\mathcal{D}}(E) - d_{M^*}^{\pi_b}(E)).$$

This guarantees from a coarse union bound and equations 7, 8 and 9 that:

$$\begin{aligned}
&\mathbb{P}\left(\left\|d_{M^*}^{\pi_b} - d_{\mathcal{D}}\right\|_1 (1-\gamma) \geq \varepsilon\right)\\
&\leq \sum_{E\subset\mathcal{X}\times\mathcal{A}} \mathbb{P}\left((1-\gamma)(d_{\mathcal{D}}(E) - d_{M^*}^{\pi_b}(E)) \geq \frac{\varepsilon}{2}\right)\\
&= \sum_{E\subset\mathcal{X}\times\mathcal{A}} \mathbb{P}\left((1-\gamma)\left(\frac{f_E(\mathcal{D})}{N(1-\gamma)} - \frac{\mathbb{E}[f_E(\mathcal{D})]}{N(1-\gamma)}\right) \geq \frac{\varepsilon}{2}\right)\\
&\leq \sum_{E\subset\mathcal{X}\times\mathcal{A}} \exp\left(-2\frac{(\frac{N\varepsilon}{2})^2}{N}\right)\\
&\leq \left(2^{|\mathcal{X}||\mathcal{A}|} - 2\right)\exp\left(-\frac{N\varepsilon^2}{2}\right),
\end{aligned}$$

where in the sum over subsets, we ignored the empty and full sets for which the probability is trivially 0. □

3.1 Algorithm and analysis

We construct the Maximum Likelihood Estimate of the baseline $\widehat{\pi}_b$ (MLE baseline) as follows:

$$\widehat{\pi}_b(a|x) = \begin{cases} \frac{N_{\mathcal{D}}(x,a)}{N_{\mathcal{D}}(x)} & \text{if } N_{\mathcal{D}}(x) > 0, \\ \frac{1}{|\mathcal{A}|} & \text{otherwise,} \end{cases} \tag{10}$$

where $N_{\mathcal{D}}(x)$ is the number of transitions starting from state x in dataset $\mathcal{D}$. Using this MLE policy, we may prove approximate safe policy improvement:

Theorem 3.2 (Safe policy improvement with a baseline estimate). *Given an algorithm α relying on the baseline π_b to train a ζ-approximate safe policy improvement π_b^* over π_b with high probability $1-\delta$. Then, α with an MLE baseline $\widehat{\pi}_b$ allows to train a $\widehat{\zeta}$-approximate safe policy improvement $\widehat{\pi}_b^*$ over π_b with high probability $1-\widehat{\delta}$:*

$$\widehat{\delta} = \delta + 2\delta', \tag{11}$$

$$\widehat{\zeta} = \zeta + \frac{2R_{\max}}{1-\gamma}\sqrt{\frac{3|\mathcal{X}||\mathcal{A}| + 4\log\frac{1}{\delta'}}{2N}}, \tag{12}$$

where N is the number of trajectories in the dataset $\mathcal{D}$ and $1-\delta'$ controls the uncertainty stemming from the baseline estimation.

Proof. We are ultimately interested in the performance improvement of $\widehat{\pi}_b^*$ with respect to the true baseline π_b in the true environment M^*. To do so, we decompose the difference into two parts:

$$\begin{aligned} \rho(\widehat{\pi}_b^*, M^*) - \rho(\pi_b, M^*) = &\underbrace{\rho(\widehat{\pi}_b^*, M^*) - \rho(\widehat{\pi}_b, M^*)}_{\alpha\text{-SPI guarantee}} \\ &+ \underbrace{\rho(\widehat{\pi}_b, M^*) - \rho(\pi_b, M^*)}_{\text{baseline estimate approximation}}. \end{aligned} \tag{13}$$

Regarding the first term, note that, while $\widehat{\pi}_b$ is not the true baseline, it is the MLE baseline, meaning in particular that it was more likely to generate the dataset $\mathcal{D}$ than the true one. Hence, we may consider it as a potential behavioural policy, and apply the safe policy improvement guarantee provided by algorithm α to bound the difference.

Regarding the second term, we need to use the distributional formulation of the performance of any policy π:

$$\rho(\pi, M) = \sum_{x\in\mathcal{X}}\sum_{a\in\mathcal{A}} d_M^{\pi}(x,a)\mathbb{E}[R(x,a)]. \tag{14}$$

Then, we may rewrite the second term in Equation 13 and upper bound it using Hölder's inequality as follows:

$$\begin{aligned} &\sum_{x\in\mathcal{X}}\sum_{a\in\mathcal{A}}\left(d_{M^*}^{\widehat{\pi}_b}(x,a) - d_{M^*}^{\pi_b}(x,a)\right)\mathbb{E}[R^*(x,a)] \\ &\qquad \leq \left\|d_{M^*}^{\widehat{\pi}_b} - d_{M^*}^{\pi_b}\right\|_1 R_{\max}. \end{aligned} \tag{15}$$

Next, we decompose the state-action discounted visits divergence as follows:

$$\left\|d_{M^*}^{\widehat{\pi}_b} - d_{M^*}^{\pi_b}\right\|_1 \leq \underbrace{\left\|d_{M^*}^{\pi_b} - d_{\mathcal{D}}\right\|_1}_{\text{Lemma 3.1}} + \underbrace{\left\|d_{M^*}^{\widehat{\pi}_b} - d_{\mathcal{D}}\right\|_1}_{\text{positive correlation}}. \tag{16}$$

For the first term, we can use the concentration inequality from Lemma 3.1[1]. With a little calculus and by setting the right value to ε, we obtain with high probability $1-\delta'$:

$$\left\|d_{M^*}^{\pi_b} - d_{\mathcal{D}}\right\|_1 \leq \frac{1}{1-\gamma}\sqrt{\frac{3|\mathcal{X}||\mathcal{A}| + 4\log\frac{1}{\delta'}}{2N}}.$$

Regarding the second term of Equation 16, we may observe that there is a correlation between $\widehat{\pi}_b$ and $d_{\mathcal{D}}$ through $\mathcal{D}$, but it is a positive correlation, meaning that the divergence between the distributions is smaller than the one with an independently drawn dataset of the same size. As a consequence, we are also able to upper bound it by assuming independence, and using the same development as for the first term. This finally gives us from Equation 16 and with high probability $1-2\delta'$:

$$\left\|d_{M^*}^{\widehat{\pi}_b} - d_{M^*}^{\pi_b}\right\|_1 \leq \frac{2}{1-\gamma}\sqrt{\frac{3|\mathcal{X}||\mathcal{A}| + 4\log\frac{1}{\delta'}}{2N}}, \tag{17}$$

which allows us to conclude the proof using union bounds. □

3.2 Theorem 3.2 discussion

SPIBB and Soft-SPIBB safe policy improvement guarantees exhibit a trade-off (controlled with their respective hyper-parameters $\frac{1}{\sqrt{N_\wedge}}$ and ϵ) between upper bounding the true policy improvement error (first term in Theorem 2.1) and allowing maximal policy improvement in the MLE MDP (next terms). When the hyper-parameters are set to 0, the true policy improvement error is null, because, trivially, no policy improvement is allowed: the algorithm is forced to reproduce the baseline. When the hyper-parameters grow, larger improvements are permitted, but the error upper bound term also grows. When the hyper-parameters tend to $+\infty$, the algorithms are not constrained anymore and find the optimal policy in the MLE MDP. In that case, the error is no longer upper bounded, resulting in poor safety performance.

When using the MLE baseline instead of the true baseline, Theorem 3.2 introduces another error upper bound term accounting for the accurateness of the baseline estimate that cannot be reduced by hyper-parameter settings. That fact is entirely expected, as otherwise we could consider an empty dataset, pretend it was generated with an optimal policy and expect a safe policy improvement over it. Another interesting point is that the bound depends on the number of trajectories, not the number of state-action visits, nor the total number of samples. Indeed, even with a huge number of samples, if there were collected only from a few trajectories, the variance may still be high, since future states visited on the trajectory depend on the previous transitions.

Regarding the MDP parameters dependency, the upper bound grows as the square root of the state set size, as for standard SPIBB, but also grows as the square root of the action set size contrarily to SPIBB that has a logarithmic dependency, which may cause issues in some RL problems. The direct horizon dependency is the same (linear). But one could argue that it is actually lower. The maximal value V_{max} in the SPIBB bounds can reach $\frac{R_{max}}{1-\gamma}$, making the dependency in H quadratic, while the N in our denominator

[1] We need to rescale with $(1-\gamma)$ the state-action discounted visits to make it sum to 1 since the original bound applies to probability distributions.

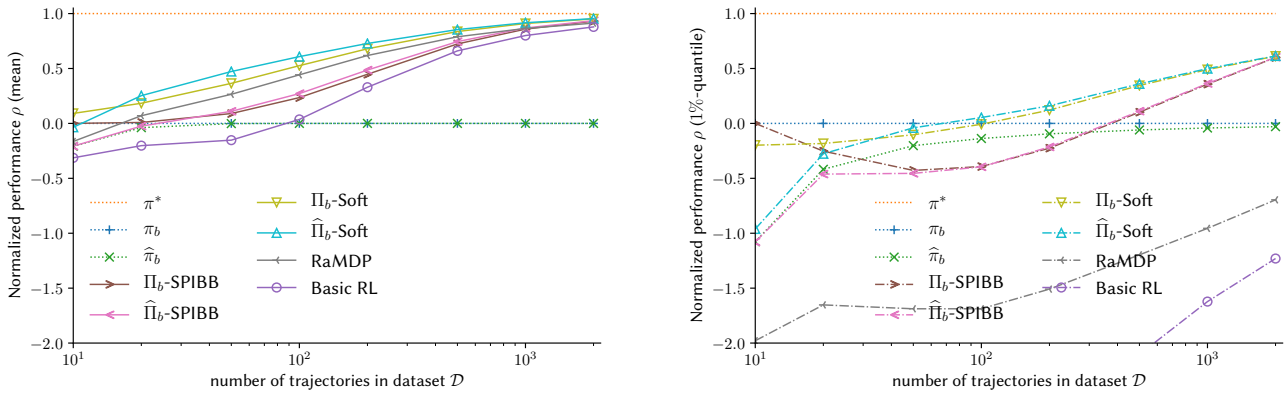

Figure 1: Finite MDPs with $\eta = 0.9$, $N_\wedge = 7$ and $\epsilon = 0.5$. On the left, the mean curves, on the right, the 1%-quantile curves.

may be regarded as a hidden horizon (since $N \approx \frac{|\mathcal{D}|}{H}$), making the total dependency $\approx H^{3/2}$. In both cases, those are better than the Soft-SPIBB cubic dependency.

One may consider other baseline estimates than the MLE, using Bayesian priors for instance, and infer new bounds. This should work as long as the baseline estimate remains a policy that could have generated the dataset.

4 EMPIRICAL ANALYSIS

Our empirical analysis reproduces the most challenging experiments found in Laroche et al. [13] and Nadjahi et al. [17]. We split it in two parts, the first considers random MDPs with finite state spaces and the second MDPs with continuous state spaces.

4.1 Random finite MDPs

4.1.1 Setup: The objective of this experiment is to empirically analyse the consistency between the theoretical findings and the practice. The experiment is run on finite MDPs that are randomly generated, with randomly generated baseline policies from which trajectories are obtained. We recall the setting below.

The true environment is a randomly generated MDP with 50 states, 4 actions, and a transition connectivity of 4: a given state-action pair may transit to 4 different states at most. The reward function is 0 everywhere, except for transitions entering the goal state, in which case the trajectory terminates with a reward of 1. The goal state is the hardest state to reach from the initial one.

The baselines are also randomly generated with a predefined level of performance specified by a ratio η between the optimal policy π^* performance and the uniform policy $\tilde{\pi}$ performance: $\rho(\pi_b, M) = \eta\rho(\pi^*, M) + (1-\eta)\rho(\tilde{\pi}, M)$. For more details on the process, we refer the interested reader to the original papers. Two values for η were considered: the experiments with $\eta = 0.9$ are reported here. The experiments with $\eta = 0.1$ had similar results and are omitted for lack of space. We also study the influence of the dataset size $|\mathcal{D}| \in [10, 20, 50, 100, 200, 500, 1000, 2000]$.

4.1.2 Competing algorithms: Our plots display nine curves:

- π^*: the optimal policy,
- π_b: the true baseline,
- $\widehat{\pi}_b$: the MLE baseline,
- $\Pi_b/\widehat{\Pi}_b$-SPIBB: SPIBB with their respective baselines,
- $\Pi_b/\widehat{\Pi}_b$-Soft: Soft-SPIBB with their respective baselines,
- RaMDP: Reward-adjusted MDP,
- and Basic RL: dynamic programming on the MLE MDP.

All the algorithms are compared using their optimal hyper-parameter according to previous work. Our hyper-parameter search with the MLE baselines did not show significant differences and we opted to report results with the same hyper-parameter values. Soft-SPIBB algorithms are the ones coined as Approx. Soft SPIBB by Nadjahi et al. [17].

4.1.3 Performance indicators: Given the random nature of the MDP and baseline generations, we normalize the performance to allow inter-experiment comparison:

$$\rho = \frac{\rho(\pi, M^*) - \rho(\pi_b, M^*)}{\rho(\pi^*, M^*) - \rho(\pi_b, M^*)}. \tag{18}$$

Thus, the optimal policy always has a normalized performance of 1, and the true baseline a normalized performance of 0. A positive normalized performance means a policy improvement, and a negative normalized performance an infringement of the policy improvement objective. Figures either report the average normalized performance of the algorithms or its 1%-quantile[2]. Each setting is processed on 250k seeds, to ensure that every performance gap visible to the naked eye is significant.

4.1.4 Empirical results: Figure 1 shows the results with $\eta = 0.9$, *i.e.* the hard setting where the behavior baseline is almost optimal, and therefore difficult to improve.

Performance of the MLE baseline. First, we notice that the mean performance of the MLE baseline $\widehat{\pi}_b$ is slightly lower than the true baseline policy π_b for small datasets. As $|\mathcal{D}|$ increases, the performance of $\widehat{\pi}_b$ quickly increases to reach the same level. The 1%-quantile is significantly lower when the number of trajectories is reduced.

[2]Note the difference with previously reported results in SPIBB papers, which focused on the conditional value at risk indicator.

Soft-SPIBB with true and estimated baselines. Comparing the results of Π_b-Soft and $\widehat{\Pi}_b$-Soft curves, it is surprising that the policy computed using an estimated policy as a baseline yields better results than the one computed with the true policy. Notice that the estimated baseline $\widehat{\pi}_b$ has a higher variance than the true baseline π_b. If we consider the impact of this variance in a given state, it means that sometimes the best (resp. worst) action will be taken more often (resp. less). When it is the case, the trained policy will be better than what could have been done with the true baseline. Sometimes, the opposite will happen, but in this case, the algorithm will try to avoid reaching this state and choose an alternative path. This means that in expectation, this does not average out and the variance in the baseline estimation might be beneficial.

SPIBB with true and estimated baselines. Analysing the performance of the $\widehat{\Pi}_b$-SPIBB algorithm, we notice that it also slightly improves over Π_b-SPIBB on the mean normalized performance. As far as safety is concerned, we see that the 1%-quantile of policies computed with $\widehat{\Pi}_b$-SPIBB falls close to the 1%-quantile of the estimated baseline $\widehat{\pi}_b$ for small datasets and close to the 1%-quantile of the policies Π_b-SPIBB for datasets with around 100 trajectories. It is expected as $\widehat{\Pi}_b$-SPIBB tends to reproduce the baseline for very small datasets, and improves over it for larger ones. That statement is also true of $\widehat{\Pi}_b$-Soft.

RaMDP and Basic RL. Finally, it is interesting to observe that although RaMDP and Basic RL can compute policies with rather high mean performance, these algorithms often return policies performing much worse than the MLE policy $\widehat{\pi}_b$ (as seen in their 1%-quantile).

4.2 Continuous MDPs

4.2.1 Helicopter domain: For MDPs with continuous state space, we focus on the helicopter environment [13, Figure 2]. In this stochastic domain, the state is defined by the position and velocity of the helicopter. The agent has a discrete set of 9 actions to control the thrust applied in each dimension. The helicopter begins in a random position of the bottom-left corner with a random initial velocity. The episode ends if the helicopter's speed exceeds some threshold, giving a reward of -1, or if it leaves the valid region, in which case the agent gets a reward between -1 and 10 depending on how close it is to the top-right corner. Using a fixed behavior policy π_b we generate $1,000$ datasets for each algorithm. We report results for two dataset sizes: $3,000$ and $10,000$ transitions.

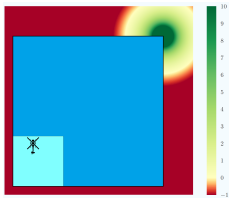

Figure 2: Helicopter.

4.2.2 Behavioural cloning: In infinite MDPs, there is no MLE baseline definition. We have to lean on behavioural cloning techniques. We compare here two straightforward ones in addition to the true behavior policy π_b: a baseline estimate $\widehat{\pi}_c$ based on the same pseudo-counts used by the algorithms, and a neural-based baseline estimate $\widehat{\pi}_n$ that uses a standard probabilistic classifier.

The *count-based policy* follows a principle similar to the MLE policy. It uses a pseudo-count for state-action pairs $\tilde{N}(x,a)$ defined according to the sum of the euclidean distance $\|x-x'\|_2$ from the state x and all states of transitions in the dataset where the action a was executed [13, Section 3.4]:

$$\tilde{N}_{\mathcal{D}}(x,a) = \sum_{\langle x_j, a_j=a, r_j, x'_j\rangle \in \mathcal{D}} \max\left\{0, 1 - \frac{\|x-x_j\|_2}{d_0}\right\}, \qquad (19)$$

where d_0 is a hyper-parameter to impose a minimum similarity before increasing the counter of a certain state. We also compute the state pseudo-count using this principle: $\tilde{N}_{\mathcal{D}}(x) = \sum_{a\in\mathcal{A}} \tilde{N}_{\mathcal{D}}(x,a)$. This way, we can define the count-based baseline estimate replacing the count in Equation 10 by its pseudo-count counterpart:

$$\widehat{\pi}_c(a|x) = \begin{cases} \frac{\tilde{N}_{\mathcal{D}}(x,a)}{\tilde{N}_{\mathcal{D}}(x)} & \text{if } \tilde{N}_{\mathcal{D}}(x) > 0, \\ \frac{1}{|\mathcal{A}|} & \text{otherwise.} \end{cases} \qquad (20)$$

The *neural-based policy* $\widehat{\pi}_n(a|x)$ is estimated using a supervised learning approach. We train a probabilistic classifier using a neural network to minimize the negative log-likelihood with respect to the actions in the dataset.

We use the same architecture as the one used to train the Double-DQN models, which is shared among all the algorithms in the helicopter domain experiments: a fully connected neural network with 3 hidden layers of 32, 128 and 28 neurons respectively, and 9 outputs corresponding to the 9 actions.

To avoid overfitting, we split the dataset in two parts: 80% for training and 20% for validation. During training, we evaluate the classifier on the validation dataset at the end of every epoch and keep the network with the smallest validation loss.

4.2.3 Competing algorithms:

- π_b: the true baseline,
- $\widehat{\pi}_c$: the pseudo-count-based estimate of the baseline,
- $\widehat{\pi}_n$: the neural-based estimate of the baseline,
- $\Pi_b/\widehat{\Pi}_c/\widehat{\Pi}_n$-SPIBB: SPIBB with their respective baselines,
- $\Pi_b/\widehat{\Pi}_c/\widehat{\Pi}_n$-Soft: Soft-SPIBB with their respective baselines,
- RaMDP: Double-DQN with Reward-adjusted MDP,
- and Double-DQN: basic deep RL algorithm.

4.2.4 Hyper-parameters. Building on the results presented by Nadjahi et al. [17], we set the hyper-parameters for the experiments with $|\mathcal{D}| = 10,000$ ($|\mathcal{D}| = 3,000$) as follows: Π_b-SPIBB with $N_\wedge = 3$ ($N_\wedge = 1$), Π_b-Soft with $\epsilon = 0.6$ ($\epsilon = 0.8$), and RaMPD with $\kappa = 1$ ($\kappa = 1.75$). For the algorithms using an estimated baseline we run a parameter search considering $N_\wedge \in [2,3,4,5]$ ($N_\wedge \in [0.5,1,2,3]$) for SPIBB and $\epsilon \in [0.4, 0.6, 0.8, 1]$ ($\epsilon \in [0.6, 0.8, 1, 1.2, 1.5, 1.8, 2]$) for Soft-SPIBB and set the parameters for the main experiments as follows: $\widehat{\Pi}_n$-SPIBB and $\widehat{\Pi}_c$-SPIBB with $N_\wedge = 3.0$ ($N_\wedge = 1.0$), and $\widehat{\Pi}_n$-Soft and $\widehat{\Pi}_c$-Soft with $\epsilon = 0.6$ ($\epsilon = 0.8$).

4.2.5 Performance indicators: The plots represent for each algorithm a modified box-plot where the caps show the 10%-quantile and 90%-quantile, the upper and lower limits of the box are the 25% and 75% quantiles and the middle line in black shows the median. We also show the average of each algorithm (dashed lines in green) and finally add a swarm-plot to enhance the distribution visualization. The table provides additional details, including the

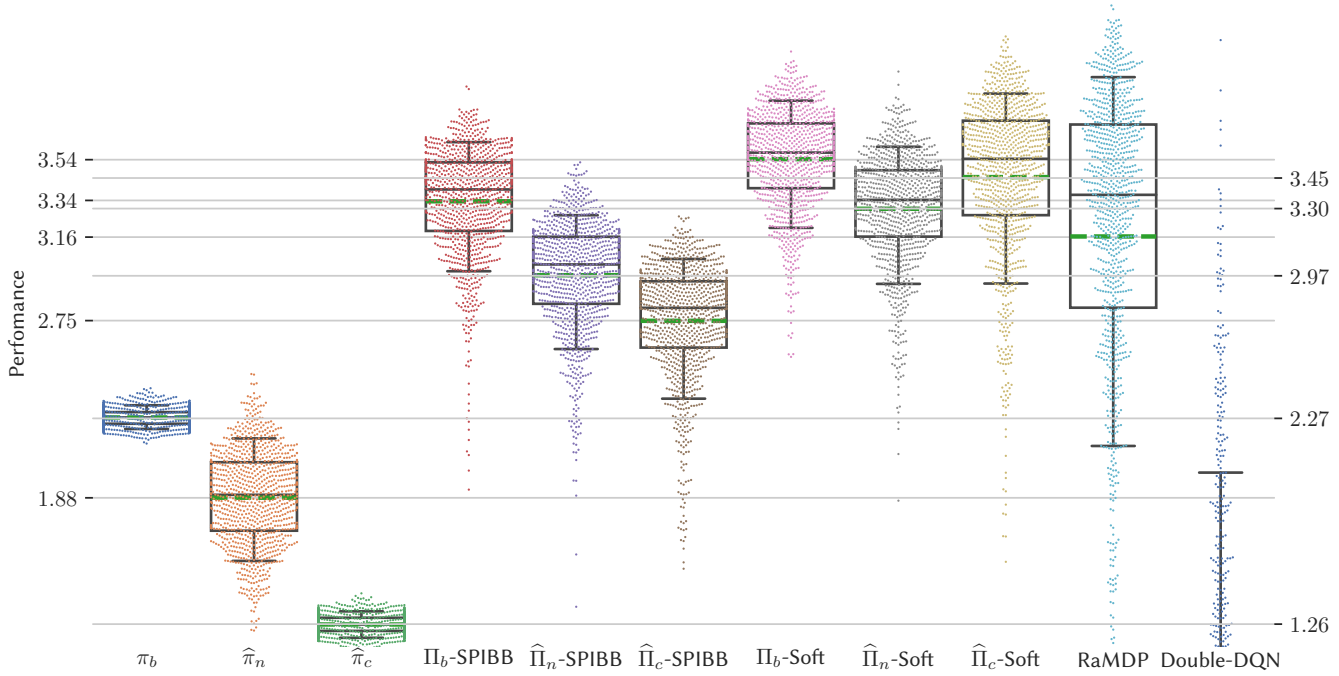

Figure 3: $|\mathcal{D}| = 10,000$. The green dashed line shows the average and the caps show the 10% and 90% percentile. Each dot on the swarm plots displays the evaluation of a seed.

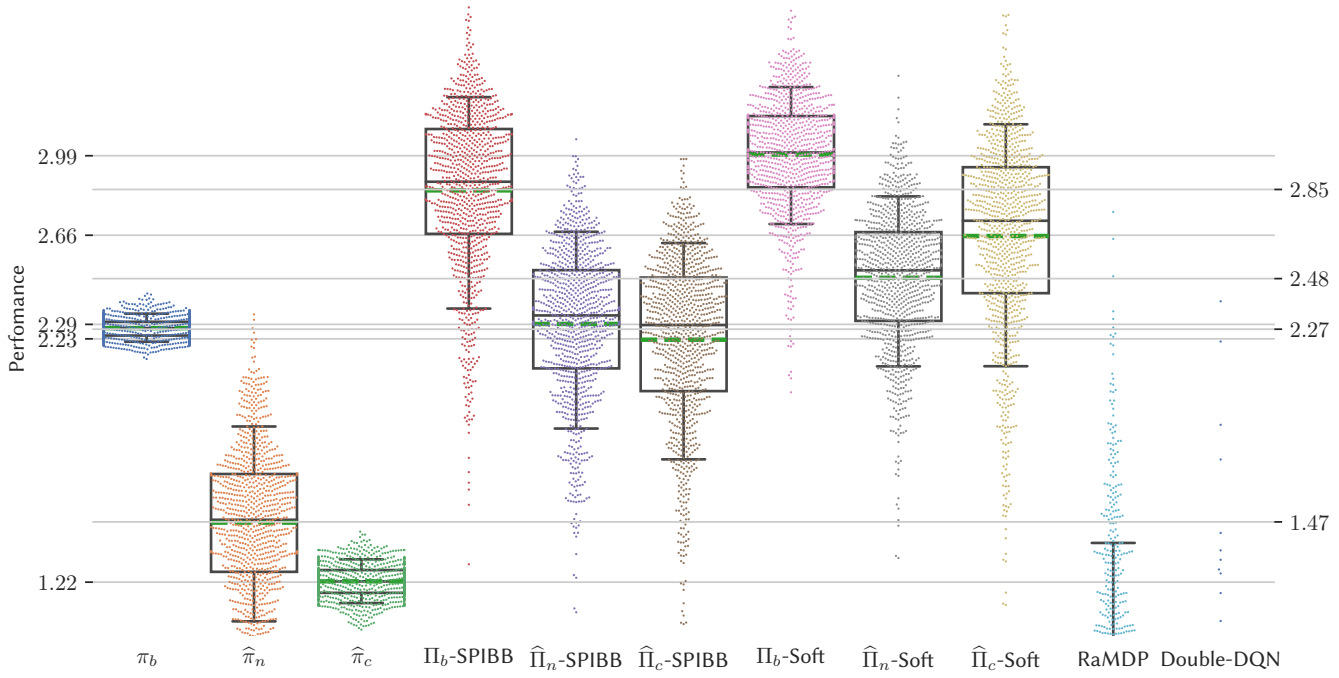

Figure 4: $|\mathcal{D}| = 3,000$. The green dashed line shows the average and the caps show the 10% and 90% percentile. Each dot on the swarm plots displays the evaluation of a seed.

		$\lvert\mathcal{D}\rvert = 3,000$				$\lvert\mathcal{D}\rvert = 10,000$			
Baseline	Algorithm	$\mathbb{P}(\rho(\pi) > \rho(\pi_b))$	avg perf	10%-qtl	1%-qtl	$\mathbb{P}(\rho(\pi) > \rho(\pi_b))$	avg perf	10%-qtl	1%-qtl
π_b		0.499	2.27	2.22	2.18	0.499	2.27	2.22	2.18
$\widehat{\pi}_n$	baseline	0.002	1.47	1.06	0.75	0.032	1.88	1.57	1.34
$\widehat{\pi}_c$		0.000	1.22	1.13	1.05	0.000	1.26	1.19	1.14
π_b		0.928	2.85	2.36	1.90	0.992	3.34	2.99	2.39
$\widehat{\pi}_n$	SPIBB	0.582	2.29	1.86	1.43	0.973	2.97	2.61	2.15
$\widehat{\pi}_c$		0.514	2.23	1.73	1.21	0.930	2.75	2.37	1.75
π_b		0.990	2.99	2.71	2.31	1.000	3.54	3.21	2.82
$\widehat{\pi}_n$	Soft-SPIBB	0.760	2.48	**2.12**	**1.71**	**0.996**	3.30	**2.93**	**2.47**
$\widehat{\pi}_c$		**0.785**	**2.66**	**2.11**	1.51	0.980	**3.45**	**2.93**	2.09
N/A	RaMDP	0.006	0.37	-0.75	-0.99	0.876	3.16	2.13	0.23
N/A	Double-DQN	0.001	-0.77	-1.00	-1.00	0.076	0.25	-0.97	-1.00

Table 1: Numerical results for the two size of datasets. The key performance indicators are respectively the percentage of policy improvement over the true baseline, the average performance of the trained policies, the 10%-quantile, and the 1%-quantile. For each column, we bold the best performing algorithm that is not using the true baseline π_b.

percentage of policies that showed a performance above the average performance of the true baseline policy.

4.2.6 Results: The results are reported numerically in Table 1 and graphically on Figure 3 for $|\mathcal{D}| = 10,000$ and Figure 4 for $|\mathcal{D}| = 3,000$.

Empiric baseline polices. On Figure 3, we observe that the baseline policies $\widehat{\pi}_c$ and $\widehat{\pi}_n$ have a performance poorer than the true behavior policy π_b. On the one hand, the neural-based baseline estimate $\widehat{\pi}_n$ can get values close to the performance of the true behavior policy, however, it has a high variance and even the 90%-quantile is below the mean of the true policy. On the other hand, the count-based policy $\widehat{\pi}_c$ has a low variance, but it has a much lower mean performance. In general, we observe a larger performance loss than in finite MDPs between the true baseline and the estimated baseline.

SPIBB. With SPIBB, the neural-based baseline estimate leads to better results for all indicators. The loss in average performance makes it worse than RaMDP in the $|\mathcal{D}| = 10,000$ datasets, but it is more reliable and yields more consistently to policy improvements. On the $|\mathcal{D}| = 3,000$ datasets, it demonstrates a higher robustness with respect to the small datasets, still compared to RaMDP.

Soft-SPIBB. The Soft-SPIBB results with baseline estimates are impressive. The loss of performance with respect to Soft-SPIBB with the true baseline is minor. We highlight that, although the policy based on pseudo-counts has a lower performance than the true one (1 point difference), it still achieves a strong performance when used with Soft-SPIBB (less than 0.1 point difference). This indicates that the proposed method is robust with respect to the performance of the estimated policy. It seems that Soft-SPIBB changes are much more forgiving the baseline approximations.

Small dataset. The experiment with a small dataset $|\mathcal{D}| = 3,000$ (Figure 4) aims to evaluate the robustness of these algorithms. We observe that the estimated policies have a performance even lower than in the experiment with $|\mathcal{D}| = 10,000$. While RaMDP's performance indicators dramatically plummet, even largely lower than the behavioural cloning policies, the algorithm SPIBB using the estimated policies usually returns policies with a performance similar to the true baseline π_b. Most exciting, the algorithm Soft-SPIBB manages to improve upon π_b with all the baselines policies, obtaining a mean performance above the average performance of π_b, and a 10%-quantile slightly lower than that of the true baseline when using the estimated policies.

Hyper-parameter sensitivity. The hyper-parameter search gave us extra insights on the behavior of the algorithms SPIBB and Soft-SPIBB using estimated baselines. We noticed that these algorithms do not have a high sensitivity to their hyper-parameters, since the performance is stable in a wide range of values, specially the Soft-SPIBB variations. We sometimes notice a tradeoff that has to be made between variance reduction and expectation maximization.

5 CONCLUSION

This paper addresses the problem of performing safe policy improvement in batch RL without direct access to the baseline, *i.e.* the behavioural policy of the dataset. We provide the first theoretical guarantees for safe policy improvement in this setting, and show on finite and continuous MDPs that the algorithm is tractable and significantly outperforms all competing algorithms that do not have access to the baseline. We also empirically confirm the limits of the approach when the number of trajectories in the dataset is low.

Currently, the limitation of SPIBB methods is the lack of algorithms to compute the parametric uncertainty of the estimated model. [1, 2, 5] investigated some methods for optimism-based exploration, which proved to not be robust enough for pessimism based purpose, where there is a requirement for exhaustiveness. Our future work in priority addresses this issue, but also the multi-batch setting, when there are several sequential updates [12], extending the method to continuous action spaces [9], and investigating the use of SPIBB in a full online setting, as a value estimation stabilizer.

REFERENCES

[1] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. 2016. Unifying Count-based Exploration and Intrinsic Motivation. In *Proceedings of the 29th Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., Barcelona, Spain, 1471–1479.

[2] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. 2019. Exploration by Random Network Distillation. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*. OpenReview.net, New Orleans, LA, USA.

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

[4] Raphaël Féraud, Reda Alami, and Romain Laroche. 2019. Decentralized Exploration in Multi-Armed Bandits. In *Proceedings of the 36th International Conference on Machine Learning, (ICML)*. PMLR, Long Beach, California, USA, 1901–1909.

[5] Lior Fox, Leshem Choshen, and Yonatan Loewenstein. 2018. DORA The Explorer: Directed Outreaching Reinforcement Action-Selection. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*. OpenReview.net, Vancouver, BC, Canada,.

[6] Garud N Iyengar. 2005. Robust Dynamic Programming. *Mathematics of Operations Research* 30, 2 (2005), 257–280.

[7] Sham Kakade and John Langford. 2002. Approximately Optimal Approximate Reinforcement Learning. In *Proceedings of the 19th International Conference on Machine Learning, (ICML)*, Vol. 2. Morgan Kaufmann, Sydney, Australia, 267–274.

[8] Levente Kocsis and Csaba Szepesvári. 2006. Bandit Based Monte-carlo Planning. In *Proceedings of the 4th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*. Springer, Skopje, Macedonia, 282–293.

[9] Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. 2019. Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction. In *Proceedings of the 32nd Advances in Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., Vancouver, BC, Canada, 11761–11771.

[10] Michail G. Lagoudakis and Ronald Parr. 2003. Least-squares Policy Iteration. *Journal of machine learning research* 4, Dec (2003), 1107–1149.

[11] Sascha Lange, Thomas Gabel, and Martin Riedmiller. 2012. *Batch Reinforcement Learning*. Springer Berlin Heidelberg, Berlin, Heidelberg, 45–73. https://doi.org/10.1007/978-3-642-27645-3_2

[12] Romain Laroche and Rémi Tachet des Combes. 2019. Multi-batch Reinforcement Learning. In *Proceedings of the 4th Reinforcement Learning and Decision Making (RLDM)*.

[13] Romain Laroche, Paul Trichelair, and Rémi Tachet des Combes. 2019. Safe Policy Improvement with Baseline Bootstrapping. In *Proceedings of the 36th International Conference on Machine Learning, (ICML)*. PMLR, Long Beach, California, USA, 3652–3661.

[14] Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. 2014. Offline Policy Evaluation Across Representations with Applications to Educational Games. In *Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. IFAAMAS/ACM, Paris, France, 1077–1084.

[15] Colin McDiarmid. 1998. Concentration. In *Probabilistic Methods for Algorithmic Discrete Mathematics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 195–248.

[16] 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. 2015. Human-level Control Through Deep Reinforcement Learning. *Nature* 518, 7540 (2015), 529.

[17] Kimia Nadjahi, Romain Laroche, and Rémi Tachet des Combes. 2019. Safe Policy Improvement with Soft Baseline Bootstrapping. In *Proceedings of the 17th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*.

[18] Arnab Nilim and Laurent El Ghaoui. 2005. Robust Control of Markov Decision Processes with Uncertain Transition Matrices. *Operations Research* 53, 5 (2005), 780–798.

[19] Cosmin Paduraru. 2013. *Off-policy Evaluation in Markov Decision Processes*. Ph.D. Dissertation. McGill University.

[20] Matteo Papini, Matteo Pirotta, and Marcello Restelli. 2017. Adaptive Batch Size for Safe Policy Gradients. In *Proceedings of the 30th Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., Long Beach, California, USA, 3591–3600.

[21] Marek Petrik, Mohammad Ghavamzadeh, and Yinlam Chow. 2016. Safe Policy Improvement by Minimizing Robust Baseline Regret. In *Proceedings of the 29th Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., Barcelona, Spain, 2298–2306.

[22] Matteo Pirotta, Marcello Restelli, Alessio Pecorino, and Daniele Calandriello. 2013. Safe Policy Iteration. In *Proceedings of the 30th International Conference on Machine Learning, (ICML)*. JMLR.org, Atlanta, GA, USA, 307–315.

[23] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. JMLR.org, Lille, France, 1889–1897.

[24] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *arXiv:1707.06347* (2017).

[25] Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building End-to-end Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Thirtieth AAAI Conference on Artificial Intelligence*. AAAI Press, Phoenix, Arizona, USA, 3776–3784.

[26] Thiago D. Simão and Matthijs T. J. Spaan. 2019. Safe Policy Improvement with Baseline Bootstrapping in Factored Environments. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. AAAI Press, Honolulu, Hawaii, USA, 4967–4974.

[27] Thiago D. Simão and Matthijs T. J. Spaan. 2019. Structure Learning for Safe Policy Improvement. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*. ijcai.org, Macao, China, 3453–3459.

[28] Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. 2015. High Confidence Policy Improvement. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. JMLR.org, Lille, France, 2380–2388.

[29] Philip S Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. 2015. High-Confidence Off-Policy Evaluation. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. AAAI Press, Austin, Texas, USA, 3000–3006.

[30] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep Reinforcement Learning with Double Q-learning. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. AAAI Press, Phoenix, Arizona, USA, 2094–2100.

[31] Harm Van Seijen, Mehdi Fatemi, Romain Laroche, Joshua Romoff, Tavian Barnes, and Jeffrey Tsang. 2017. Hybrid Reward Architecture for Reinforcement Learning. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., Long Beach, California, USA, 5392–5402.

[32] Huan Xu and Shie Mannor. 2009. Parametric Regret in Uncertain Markov Decision Processes. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*. IEEE, Shanghai, China, 3606–3613.