A Theoretical Explanation of Deep RL Performance in Stochastic Environments

Cassidy Laidlaw **Banghua Zhu** **Stuart Russell** **Anca Dragan**
University of California, Berkeley
{cassidy_laidlaw,banghua,russell,anca}@cs.berkeley.edu

Abstract

Reinforcement learning (RL) theory has largely focused on proving minimax sample complexity bounds. These require *strategic* exploration algorithms that use relatively limited function classes for representing the policy or value function. Our goal is to explain why deep RL algorithms often perform well in practice, despite using *random* exploration and much more expressive function classes like neural networks. Our work arrives at an explanation by showing that many stochastic MDPs can be solved by performing only a few steps of value iteration on the random policy's Q function and then acting greedily. When this is true, we find that it is possible to separate the *exploration* and *learning* components of RL, making it much easier to analyze. We introduce a new RL algorithm, SQIRL, that iteratively learns a near-optimal policy by exploring randomly to collect rollouts and then performing a limited number of steps of fitted-Q iteration over those rollouts. We find that any regression algorithm that satisfies basic in-distribution generalization properties can be used in SQIRL to efficiently solve common MDPs. This can explain why deep RL works with complex function approximators like neural networks, since it is empirically established that neural networks generalize well in-distribution. Furthermore, SQIRL explains why random exploration works well in practice, since we show many environments can be solved by effectively estimating the random policy's Q-function and then applying zero or a few steps of value iteration. We leverage SQIRL to derive instance-dependent sample complexity bounds for RL that are exponential only in an "effective horizon" of lookahead—which is typically much smaller than the full horizon—and on the complexity of the class used for function approximation. Empirically, we also find that SQIRL performance strongly correlates with PPO and DQN performance in a variety of stochastic environments, supporting that our theoretical analysis is predictive of practical performance.

1 Introduction

The theory of reinforcement learning (RL) does not quite predict the practical successes (and failures) of deep RL. Specifically, there are two gaps between theory and practice. First, RL theoreticians focus on strategic exploration, while most deep RL algorithms explore randomly. Explaining why random exploration works in practice is difficult because theorists can show that randomly exploring algorithms' worst-case sample complexity is exponential in the horizon. Thus, most recent progress in the theory of RL has focused on strategic exploration algorithms, which use upper confidence bound (UCB) bonuses to effectively explore the state space of an environment. Second, RL theory struggles to explain why deep RL can learn efficiently while using complex function approximators like deep neural networks. This is because UCB-type algorithms only work in highly-structured environments where they can use simpler function classes to represent value functions and policies.

Our goal is to bridge these two gaps: to explain why random exploration works despite being

Mathematics of Modern Machine Learning Workshop at NeurIPS 2023.

exponentially bad in the worst-case, and to understand why deep RL succeeds despite using deep neural networks for function approximation. Some recent progress has been made on the former problem by Laidlaw et al. [1], who analyze when random exploration will succeed in *deterministic* environments. They demonstrate a surprising finding: in many deterministic environments, it is optimal to act greedily according to the Q-function of the policy that takes actions uniformly at random. This inspires their definition of a property of deterministic environments called the "effective horizon," which is roughly the number of lookahead steps a Monte Carlo planning algorithm needs to solve the environment when relying on random rollouts to evaluate leaf nodes. They then show that a randomly exploring RL algorithm called Greedy Over Random Policy (GORP) has sample complexity exponential only in the effective horizon rather than the full horizon. They show the effective horizon also predicts empirical deep RL performance.

In this work, we take inspiration from the effective horizon to analyze RL in stochastic environments with function approximation. We introduce a new RL algorithm, SQIRL (shallow Q-iteration via reinforcement learning), that generalizes GORP to stochastic environments. SQIRL iteratively learns a policy by alternating between collecting data through purely random exploration and then training function approximators on the collected data. The advantage of this algorithm is that it only relies on access to a *regression oracle* that can generalize in-distribution from i.i.d. samples, which we know works even with neural networks. Thus, unlike strategic exploration algorithms which work for only limited function classes, SQIRL helps explain why RL can work with expressive function classes. Furthermore, the SQIRL leverages the effective horizon property, helping to explain why RL works in practice using random exploration.

Theoretically, we prove instance-dependent sample complexity bounds for SQIRL that depend on a stochastic version of the effective horizon as well as properties of the regression oracle used. We show that a wide variety of function approximators can be used within SQIRL. Furthermore, to strengthen our claim that SQIRL can explain why deep RL succeeds while using random exploration and neural networks, we compare its performance to the deep RL algorithms PPO [2] and DQN [3] in over 150 stochastic environments. We find that in environments where both PPO and DQN converge to an optimal policy, SQIRL does as well 78% of the time; when both PPO and DQN fail, SQIRL never succeeds. These empirical results and theoretical contributions show that the effective horizon and the SQIRL algorithm can help explain when and why deep RL works even in stochastic environments.

2 Setup and Related Work

We consider the setting of an episodic Markov decision process (MDP) with finite horizon T. The MDP comprises a horizon $T \in \mathbb{N}$, states $s \in \mathcal{S}$, actions $a \in \mathcal{A}$, initial state distribution $p_1(s_1)$, transitions $p_t(s_{t+1} \mid s_t, a_t)$, and reward $R_t(s_t, a_t)$ for $t \in [T]$, where $[n]$ denotes the set $\{1, \ldots, n\}$. Let $J(\pi) = E_{s_1 \sim p(s_1)}[V_1^\pi(s_1)]$ denote the expected return of a policy π. The objective of an RL algorithm is to find an ϵ-optimal policy, i.e., one such that $J(\pi) \geq J^* - \epsilon$ where $J^* = \max_{\pi^*} J(\pi^*)$. We denote the sample complexity $N_{\epsilon,\delta}$ of an RL algorithm as the minimum number of timesteps of interaction with the environment needed to find an ϵ-optimal policy with probability at least $1 - \delta$. See Appendix A for a full description of our setting and notation.

Related work Most prior work in RL theory has focused finding strategic exploration-based RL algorithms which have minimax regret or sample complexity bounds [4, 5, 6, 7, 8, 9, 10, 11]. However, since the worst-case bounds for random exploration are exponential in the horizon [12, 6], minimax analysis cannot explain why random exploration works well in practice. Furthermore, while strategic exploration has been extended to broader and broader classes of function approximators [13, 14, 15, 16], even the broadest of these cannot use complex function approximators like neural networks. A much smaller set of work has analyzed random exploration [17, 18] and more general function approximators [19] in RL. However, Laidlaw et al. [1] show that the sample complexity bounds in these papers fail to explain empirical RL performance even in deterministic environments.

3 The Stochastic Effective Horizon and SQIRL

We now present our main theoretical findings extending the effective horizon property and GORP algorithm to stochastic environments. The effective horizon was motivated in Laidlaw et al. [1] by a surprising property that the authors show holds in many deterministic MDPs: acting greedily with respect to the Q-function of the random policy, i.e. ${\pi^{\text{rand}}}_t(a \mid s) = 1/A \quad \forall s, a, t$, gives an optimal policy. They call a generalization of this property *k-QVI-solvability*. To define k-QVI-solvability, we introduce some notation. One step of Q-value iteration transforms a Q-function Q to $Q' = \text{QVI}(Q)$, where

$$Q'_t(s_t, a_t) = R_t(s_t, a_t) + E_{s_{t+1}}\left[\max_{a \in \mathcal{A}} Q_{t+1}\left(s_{t+1}, a\right)\right].$$

We also denote by $\Pi(Q)$ the set of policies which act greedily with respect to the Q-function Q; that is,

$$\Pi\left(Q\right) = \left\{\pi \;\middle|\; \forall s, t \quad \pi_t(s) \in \arg\max_{a \in \mathcal{A}} Q_t(s, a)\right\}.$$

Furthermore, we define a sequence of Q-functions $Q^1, \ldots, Q^T$ by letting $Q^1 = Q^{\pi^{\text{rand}}}$ be the Q-function of the random policy and $Q^{i+1} = \text{QVI}(Q^i)$.

Definition 3.1 (k-QVI-solvable)**.** *We say an MDP is k-*QVI-solvable *for some $k \in [T]$ if every policy in $\Pi(Q^k)$ is optimal.*

If acting greedily on the random policy's Q-values is optimal, then an MDP is 1-QVI-solvable; k-QVI-solvability extends this to cases where a few steps of value iteration must be applied before acting greedily. We construct stochastic sticky-action versions of the 155 deterministic MDPs in the BRIDGE dataset and show that most of them are k-QVI-solvable for small values of k; see Appendix F.1 for details. This suggests that it may be possible to extend the effective horizon to stochastic environments. However, Laidlaw et al. [1] show that k-QVI-solvability alone is not enough to guarantee that random exploration can lead to efficient RL. They define the effective horizon by combining k with a measure of how precisely Q^k needs to be estimated to act optimally.

Definition 3.2 (k-gap)**.** *If an MDP is k-QVI-solvable, we define its k-gap as*

$$\Delta_k = \inf_{(t,s) \in [T] \times \mathcal{S}} \left(\max_{a^* \in \mathcal{A}} Q_t^k(s, a^*) - \max_{a \notin \arg\max_a Q_t^k(s,a)} Q_t^k(s, a)\right).$$

Intuitively, the smaller the k-gap, the more precisely an algorithm must estimate Q^k in order to act optimally in an MDP which is k-QVI-solvable. We can now define the *stochastic* effective horizon, which we show is closely related to the effective horizon in deterministic environments:

Definition 3.3 (Stochastic effective horizon)**.** *Given $k \in [T]$, define $\bar{H}_k = k + \log_A(1/\Delta_k^2)$ if an MDP is k-QVI-solvable and $\bar{H}_k = \infty$ otherwise. The* stochastic effective horizon *is $\bar{H} = \min_k \bar{H}_k$.*

Lemma 3.4. *The deterministic effective horizon H is bounded for any $k \in [T]$ as*

$$H \leq \min_k \left[\bar{H}_k + \log_A O\left(\log\left(TA^k\right)\right)\right].$$

Furthermore, if an MDP is k-QVI-solvable, then with probability at least $1 - \delta$, GORP will return an optimal policy with sample complexity at most $O(kT^2 A^{\bar{H}_k} \log\left(TA/\delta\right))$.

We defer all proofs to Appendix D. Lemma 3.4 shows that our definition of the stochastic effective horizon is closely related to the deterministic effective horizon definition.

SQIRL To show that the stochastic effective horizon can provide insight into when and why deep RL succeeds, we introduce the shallow Q-iteration via reinforcement learning (SQIRL) algorithm, show in in Algorithm 1. SQIRL iteratively builds a policy timestep-by-timestep. At the ith iteration, it collects m episodes of data by following previously learned policies for timesteps $t < i$ and then acting randomly after. It then uses *regression* to estimate the random policy's Q-function, followed by $k - 1$ steps of *fitted Q-iteration* (FQI) [20]. We show in Appendix B that SQIRL is a generalization of the GORP algorithm from Laidlaw et al. [1] to stochastic environments. Algorithm 1 depends on a *regression oracle* REGRESS that can effectively perform regression of the random policy's Q-function and FQI. We give examples of regression oracles in Appendix C.

Sample complexity bounds Our main theoretical result shows that we can bound the sample complexity of SQIRL based on the stochastic effective horizon and properties of the regression oracle:

Theorem 3.5 (Informal)**.** *Suppose* REGRESS *can estimate Q-functions via regression and fitted Q-iteration from m samples, giving Q-functions which have population error which is $O\big(D(\delta)\frac{\log m}{m}\big)$ with probability at least $1 - \delta$. Then if the MDP is k-QVI-solvable for some $k \in [T]$, there is a univeral constant C such that SQIRL (Algorithm 1) will return an ϵ-optimal policy with probability at least $1 - \delta$ if $m \geq C \frac{kTA^k D(\delta/kt)}{\Delta_k^2 \epsilon} \log \frac{kTAD(\delta/kt)}{\Delta_k \epsilon}$. Thus, the sample complexity of SQIRL is at most*

$$N_{\epsilon,\delta}^{\textit{SQIRL}} \leq \widetilde{O}\left(\tfrac{1}{\epsilon} kT^3 A^{\bar{H}_k} D(\delta/kt) \log D(\delta/kt)\right). \tag{1}$$

Algorithm 1 The shallow Q-iteration via reinforcement learning (SQIRL) algorithm.

procedure SQIRL(k, m, REGRESS)
  for $i = 1, \dots, T$ **do**
    Collect m episodes by following π_t for $t < i$ and π^{rand} thereafter to obtain $\{(s_t^j, a_t^j, y_t^j)\}_{j=1}^m$.
    $\hat{Q}_{i+k-1}^1 \leftarrow \text{REGRESS}(\{(s_{i+k-1}^j, a_{i+k-1}^j, \sum_{t=i+k-1}^T R_t(s_t^j, a_t^j))\}_{j=1}^m)$.
    for $t = i + k - 2, \dots, i$ **do**
      $\hat{Q}_t^{i+k-t} \leftarrow \text{REGRESS}(\{(s_t^j, a_t^j, R_t(s_t^j, a_t^j) + \max_{a \in \mathcal{A}} \hat{Q}_{t+1}^{i+k-t-1}(s_{t+1}^j, a))\}_{j=1}^m)$.
    end for
    Define π_i by $\pi_i(s) \leftarrow \arg\max_a \hat{Q}_i^k(s, a)$.
  end for
return $\pi_1, \dots, \pi_T$.
end procedure

Algorithm	Envs. solved
PPO	102
DQN	77
SQIRL	68

Table 1: The number of sticky-action BRIDGE environments (out of 155) solved by three RL algorithms. Our SQIRL algorithm solves about 2/3 of the environments that PPO does and almost as many as DQN.

Algorithms		Sample complexity comparison	
		Correl.	Median ratio
SQIRL	PPO	0.79	1.00
SQIRL	DQN	0.63	0.55
PPO	DQN	0.57	0.55

Table 2: A comparison of the empirical sample complexities of SQIRL, PPO, and DQN in the sticky-action BRIDGE environments. SQIRL's sample complexity has higher Spearman correlation with PPO and DQN than they do with each other. Furthermore, SQIRL tends to have similar sample complexity to PPO and better sample complexity than DQN.

See Appendix B for a formal version of Theorem 3.5 and further analysis. To understand the bound on the sample complexity of SQIRL given in (2), first compare it to GORP's sample complexity in Lemma 3.4. Like GORP, SQIRL has sample complexity exponential in only the effective horizon. As Appendix C shows, in many cases $D(\delta) \asymp d + \log(1/\delta)$, where d is the pseudo-dimension of the hypothesis class used by the regression oracle. Then, the sample complexity of SQIRL is $\widetilde{O}(kT^3 A^{\bar{H}_k} d/\epsilon)$—ignoring log factors, just a Td/ϵ factor more than the sample complexity of GORP. The additional factor of d is necessary because SQIRL must learn a Q-function that generalizes over many states, while GORP can estimate the Q-values at a single state in deterministic environments. See Table 3 for a comparison of these sample complexity bounds to others in the literature.

4 Experiments

While our theoretical results strongly suggest that SQIRL and the stochastic effective horizon can explain deep RL performance, we also want to validate these insights empirically. We compare the sample complexity of PPO [2], DQN [3], and SQIRL in 155 stochastic sticky-action versions of the BRIDGE environments from Laidlaw et al. [1] (see Appendix E for details). The results of our experiments are shown in Tables 1 and 2 and Figure 3. As shown in Table 1, SQIRL solves about two-thirds as many environments as PPO and nearly as many as DQN. This shows that SQIRL is not simply a useful algorithm in theory—it can solve a wide variety of stochastic environments in practice. It also suggests that the assumptions we introduce in Section 3 actually hold for RL in realistic environments with neural network function approximation.

Furthermore, as shown in Table 1 and Figure 3, SQIRL's sample complexity correlates better with that of PPO and DQN than they correlate with each other (as measured by Spearman correlation). We also report the median ratio of the sample complexities of each pair of algorithms to see if they agree in absolute scale. We find that SQIRL tends to have similar sample complexity to PPO and better sample complexity than DQN. The fact that there is a close match between the performance of SQIRL and deep RL algorithms—when deep RL has low sample complexity, so does SQIRL, and vice versa—suggests that our theoretical explanation for why SQIRL succeeds is also a good explanation for why deep RL succeeds.

Acknowledgments and Disclosure of Funding

We would like to thank Sam Toyer for feedback on drafts as well as Jiantao Jiao for helpful discussions.

This work was supported by a grant from Open Philanthropy to the Center for Human-Compatible Artificial Intelligence at UC Berkeley. Cassidy Laidlaw is supported by an Open Philanthropy AI Fellowship.

References

[1] Cassidy Laidlaw, Stuart Russell, and Anca Dragan. Bridging RL Theory and Practice with the Effective Horizon. In *Advances in Neural Information Processing Systems*, 2023. URL `http://arxiv.org/abs/2304.09853`. arXiv:2304.09853 [cs, stat].

[2] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv:1707.06347 [cs]*, August 2017. URL `http://arxiv.org/abs/1707.06347`. arXiv: 1707.06347.

[3] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 1476-4687. doi: 10.1038/nature14236. URL `https://www.nature.com/articles/nature14236`. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 7540 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Computer science Subject_term_id: computer-science.

[4] Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E. Schapire. Contextual Decision Processes with low Bellman rank are PAC-Learnable. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1704–1713. PMLR, July 2017. URL `https://proceedings.mlr.press/v70/jiang17c.html`. ISSN: 2640-3498.

[5] Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax Regret Bounds for Reinforcement Learning. *arXiv:1703.05449 [cs, stat]*, July 2017. URL `http://arxiv.org/abs/1703.05449`. arXiv: 1703.05449.

[6] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I. Jordan. Is Q-learning Provably Efficient? *arXiv:1807.03765 [cs, math, stat]*, July 2018. URL `http://arxiv.org/abs/1807.03765`. arXiv: 1807.03765.

[7] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I. Jordan. Provably Efficient Reinforcement Learning with Linear Function Approximation. *arXiv:1907.05388 [cs, math, stat]*, August 2019. URL `http://arxiv.org/abs/1907.05388`. arXiv: 1907.05388.

[8] 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 *Proceedings of the Thirty-Second Conference on Learning Theory*, pages 2898–2933. PMLR, June 2019. URL `https://proceedings.mlr.press/v99/sun19a.html`. ISSN: 2640-3498.

[9] Zhuoran Yang, Chi Jin, Zhaoran Wang, Mengdi Wang, and Michael I. Jordan. On Function Approximation in Reinforcement Learning: Optimism in the Face of Large State Spaces, December 2020. URL `http://arxiv.org/abs/2011.04622`. arXiv:2011.04622 [cs, math, stat].

[10] Kefan Dong, Jian Peng, Yining Wang, and Yuan Zhou. Root-n-Regret for Learning in Markov Decision Processes with Function Approximation and Low Bellman Rank. In *Proceedings of Thirty Third Conference on Learning Theory*, pages 1554–1557. PMLR, July 2020. URL `https://proceedings.mlr.press/v125/dong20a.html`. ISSN: 2640-3498.

[11] Omar Darwiche Domingues, Pierre Ménard, Emilie Kaufmann, and Michal Valko. Episodic Reinforcement Learning in Finite MDPs: Minimax Lower Bounds Revisited. In *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, pages 578–598. PMLR, March 2021. URL https://proceedings.mlr.press/v132/domingues21a.html. ISSN: 2640-3498.

[12] Sven Koenig and Reid G. Simmons. Complexity analysis of real-time reinforcement learning. In *Proceedings of the eleventh national conference on Artificial intelligence*, AAAI'93, pages 99–105, Washington, D.C., July 1993. AAAI Press. ISBN 978-0-262-51071-4.

[13] Chi Jin, Qinghua Liu, and Sobhan Miryoosefi. Bellman Eluder Dimension: New Rich Classes of RL Problems, and Sample-Efficient Algorithms. In *Advances in Neural Information Processing Systems*, volume 34, pages 13406–13418. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/6f5e4e86a87220e5d361ad82f1ebc335-Abstract.html.

[14] Simon Du, Sham Kakade, Jason Lee, Shachar Lovett, Gaurav Mahajan, Wen Sun, and Ruosong Wang. Bilinear Classes: A Structural Framework for Provable Generalization in RL. In *Proceedings of the 38th International Conference on Machine Learning*, pages 2826–2836. PMLR, July 2021. URL https://proceedings.mlr.press/v139/du21a.html. ISSN: 2640-3498.

[15] Dylan J. Foster, Sham M. Kakade, Jian Qian, and Alexander Rakhlin. The Statistical Complexity of Interactive Decision Making, December 2021. URL http://arxiv.org/abs/2112.13487. arXiv:2112.13487 [cs, math, stat].

[16] Zixiang Chen, Chris Junchi Li, Huizhuo Yuan, Quanquan Gu, and Michael Jordan. A General Framework for Sample-Efficient Function Approximation in Reinforcement Learning. September 2022. URL https://openreview.net/forum?id=dqITIpZ5Z4b.

[17] Yao Liu and Emma Brunskill. When Simple Exploration is Sample Efficient: Identifying Sufficient Conditions for Random Exploration to Yield PAC RL Algorithms, April 2019. URL http://arxiv.org/abs/1805.09045. arXiv:1805.09045 [cs, stat].

[18] Chris Dann, Yishay Mansour, Mehryar Mohri, Ayush Sekhari, and Karthik Sridharan. Guarantees for Epsilon-Greedy Reinforcement Learning with Function Approximation. In *Proceedings of the 39th International Conference on Machine Learning*, pages 4666–4689. PMLR, June 2022. URL https://proceedings.mlr.press/v162/dann22a.html. ISSN: 2640-3498.

[19] Dhruv Malik, Aldo Pacchiano, Vishwak Srinivasan, and Yuanzhi Li. Sample Efficient Reinforcement Learning In Continuous State Spaces: A Perspective Beyond Linearity. In *Proceedings of the 38th International Conference on Machine Learning*, pages 7412–7422. PMLR, July 2021. URL https://proceedings.mlr.press/v139/malik21c.html. ISSN: 2640-3498.

[20] András Antos, Csaba Szepesvári, and Rémi Munos. Fitted Q-iteration in continuous action-space MDPs. In *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. URL https://papers.nips.cc/paper_files/paper/2007/hash/da0d1111d2dc5d489242e60ebcbaf988-Abstract.html.

[21] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-Based Batch Mode Reinforcement Learning. *Journal of Machine Learning Research*, 6(18):503–556, 2005. ISSN 1533-7928. URL http://jmlr.org/papers/v6/ernst05a.html.

[22] Peter L. Bartlett, Nick Harvey, Chris Liaw, and Abbas Mehrabian. Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks, October 2017. URL http://arxiv.org/abs/1703.02930. arXiv:1703.02930 [cs].

[23] Aad W. van der Vaart and Jon A. Wellner. Uniform Entropy Numbers. In Aad W. van der Vaart and Jon A. Wellner, editors, *Weak Convergence and Empirical Processes: With Applications to Statistics*, Springer Series in Statistics, pages 134–153. Springer, New York, NY, 1996. ISBN 978-1-4757-2545-2. doi: 10.1007/978-1-4757-2545-2_18. URL https://doi.org/10.1007/978-1-4757-2545-2_18.

[24] Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge, 2018. ISBN 978-1-108-41519-4. doi: 10.1017/9781108231596. URL https://www.cambridge.org/core/books/highdimensional-probability/797C466DA29743D2C8213493BD2D2102.

[25] Vladimir Koltchinskii. Local Rademacher Complexities and Oracle Inequalities in Risk Minimization. *The Annals of Statistics*, 34(6):2593–2656, 2006. ISSN 0090-5364. URL https://www.jstor.org/stable/25463523. Publisher: Institute of Mathematical Statistics.

[26] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *The Journal of Machine Learning Research*, 22(1):12348–12355, 2021. URL https://dl.acm.org/doi/abs/10.5555/3546258.3546526. Publisher: JMLRORG.

[27] Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018. URL https://www.jair.org/index.php/jair/article/view/11182.

Appendix

A Setting

Here, we expand on Section 2 to provide a full description of the MDP setting we consider.

As described in Section 2, we consider episodic MDPs that consist of a horizon $T \in \mathbb{N}$, states $s \in \mathcal{S}$, actions $a \in \mathcal{A}$, initial state distribution $p_1(s_1)$, transitions $p_t(s_{t+1} \mid s_t, a_t)$, and reward $R_t(s_t, a_t)$ for $t \in [T]$. We assume that $A = |\mathcal{A}| \geq 2$ is finite. While we do not explicitly consider discounted MDPs, our analysis is easily extendable to incorporate a discount rate.

An RL agent interacts with the MDP for a number of episodes, starting from a state $s_1 \sim p(s_1)$. At each step $t \in [T]$ of an episode, the agent observes the state s_t, picks an action a_t, receives reward $R(s_t, a_t)$, and transitions to the next state $s_{t+1} \sim p(s_{t+1} \mid s_t, a_t)$. A policy π is a set of functions $\pi_1, \ldots, \pi_t : \mathcal{S} \to \Delta(\mathcal{A})$, which defines for each state and timestep a distribution $\pi_t(a \mid s)$ over actions. If a policy is deterministic at some state, then with slight abuse of notation we denote $a = \pi_t(s)$ to be the action taken by π_t in state s. We assume that the total reward $\sum_{t=1}^{t} R_t(s_t, a_t)$ is bounded almost surely in $[0, 1]$; any bounded reward function can be normalized to satisfy this assumption.

Using a policy to select actions in an MDP induces a distribution over states and actions with $a_t \sim \pi_t(\cdot \mid s_t)$. We use P_π and E_π to refer to the probability measure and expectation with respect to this distribution for a particular policy π. We denote a policy's Q-function $Q_t^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and value function $V_t^\pi : \mathcal{S} \to \mathbb{R}$ for each $t \in [T]$, defined as:

$$Q_t^\pi(s, a) = E_\pi\left[\sum_{t'=t}^{T} R_{t'}(s_{t'}, a_{t'}) \mid s_t = s, a_t = a\right] \qquad V_t^\pi(s) = E_\pi\left[\sum_{t'=t}^{T} R_{t'}(s_{t'}, a_{t'}) \mid s_t = s\right]$$

Let $J(\pi) = E_{s_1 \sim p(s_1)}[V_1^\pi(s_1)]$ denote the expected return of a policy π. The objective of an RL algorithm is to find an ϵ-optimal policy, i.e., one such that $J(\pi) \geq J^* - \epsilon$ where $J^* = \max_{\pi^*} J(\pi^*)$. Suppose that after interacting with the environment for n timesteps (i.e., counting one episode as T timesteps), an RL algorithm returns a policy π^n. We define the (ϵ, δ) sample complexity $N_{\epsilon,\delta}$ of an RL algorithm as the minimum number of timesteps needed to return an ϵ-optimal policy with probability at least $1 - \delta$, where the randomness is over the environment and the RL algorithm:

$$N_{\epsilon,\delta} = \min\left\{n \in \mathbb{N} \mid \mathbb{P}\left(J(\pi^n) \geq J^* - \epsilon\right) \geq 1 - \delta\right\}.$$

B Full analysis of SQIRL

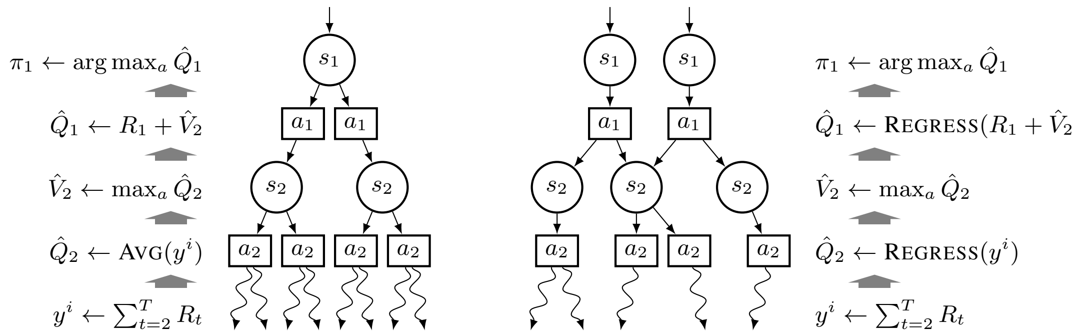
In this appendix, we thoroughly compare GORP and SQIRL, showing how SQIRL can be considered an extension of GORP to stochastic environments (Figure 1). Then, we present our full analysis of SQIRL's sample complexity.

Algorithm 2 The greedy over random policy (GORP) algorithm [1].

procedure GORP(k, m)
 for $i = 1, \ldots, T$ **do**
  for $a_{i:i+k-1} \in \mathcal{A}^k$ **do**
   sample m episodes following $\pi_1, \ldots, \pi_{i-1}$, then actions $a_{i:i+k-1}$, and finally π^{rand}.
   $\hat{Q}_i(s_i, a_{i:i+k-1}) \leftarrow \frac{1}{m}\sum_{j=1}^{m}\sum_{t=i}^{T}\gamma^{t-i}R(s_t^j, a_t^j)$.
  end for
  $\pi_i(s_i) \leftarrow \arg\max_{a_i \in \mathcal{A}} \max_{a_{i+1:i+k-1} \in \mathcal{A}^{k-1}} \hat{Q}_i(s_i, a_i, a_{i+1:i+k-1})$.
 end for
 return π
end procedure

Recall the two theory-practice divides we aim to bridge: first, understanding why random exploration works in practice despite being exponentially inefficient in theory; and second, explaining why using deep neural networks for function approximation is feasible in practice despite having little theoretical justification. SQIRL is designed to address both of these. It generalizes the GORP algorithm to stochastic environments, giving sample complexity exponential only in the stochastic effective horizon $\bar{H}$ rather than the full horizon T. It also allows the use of a wide variety of function approximators that only need to satisfy relatively mild conditions; these are satisfied by neural networks and many other function classes.

GORP The GORP algorithm (Algorithm 2 and Figure 1a) is difficult to generalize to the stochastic



(a) The GORP algorithm.

(b) The SQIRL algorithm.

Figure 1: We introduce the shallow Q-iteration via reinforcement learning (SQIRL) algorithm, which uses random exploration and function approximation to efficiently solve environments with a low stochastic effective horizon. SQIRL is a generalization of the GORP algorithm [1] to stochastic environments. In the figure, both algorithms are shown solving the first timestep of a 2-QVI-solvable MDP. The GORP algorithm (left) uses random rollouts to estimate the random policy's Q-values at the leaf nodes of a "search tree" and then backs up these values to the root node. It is challenging to generalize this algorithm to stochastic environments because both the initial state and transition dynamics are random. This makes it impossible to perform the steps of GORP where it averages over random rollouts and backs up values along deterministic transitions. SQIRL replaces these steps with *regression* of the random policy's Q-values at leaf nodes and *fitted Q-iteration* (FQI) for backing up values, allowing it to efficiently learn in stochastic environments.

case because many of its components are specific to deterministic environments. In particlular, GORP learns a sequence of actions that solve a deterministic MDP by simulating a Monte Carlo planning algorithm. At each iteration, it collects m episodes for each k-long action sequence by playing the previous learned actions, the k-long action sequence, and then sampling from the π^{rand}. Then, it picks the action sequence with the highest mean return and adds it to the sequence of learned actions.

At first, it seems very difficult to translate GORP to the stochastic setting. It learns an open-loop sequence of actions, while stochastic environments can only be solved by a closed-loop policy. It also relies on being able to repeatedly reach the same states to estimate their Q-values, which in a stochastic MDP is often impossible due to randomness in the transitions.

Regressing the random policy's Q-function To understand how we overcome these challenges, start by considering the first iteration of GORP ($i = 1$) when $k = 1$. In this case, GORP simply estimates the Q-function of the random policy ($Q^1 = Q^{\pi^{\text{rand}}}$) at the start state s_1 for each action as an empirical average over random rollouts. The difficulty in stochastic environments is that the start state s_1 is sampled from a distribution $p(s_1)$ instead of being fixed. How can we precisely estimate $Q^1(s_1, a)$ over a variety of states and actions when we may never sample the same start state twice? Our key insight is to replace an *average* over random rollouts with *regression* of the Q-function from samples of the form (s_1, a_1, y), where $y = \sum_{t=1}^{T} R_t(s_t, a_t)$. Standard regression algorithms attempt to estimate the conditional expectation $E[y \mid s_1, a_1]$. Since in this case $E[y \mid s_1, a_1] = Q_1^1(s_1, a_1)$, if our regression algorithm works well then it should output $\hat{Q}_1^1 \approx Q_1^1$.

If we can precisely regress $\hat{Q}_1^1 \approx Q_1^1$, then for most states s_1 we should have $\arg\max_a \hat{Q}_1^1(s_1, a) \subseteq \arg\max_a Q_1^1(s_1, a)$. This, combined with the MDP being 1-QVI-solvable, means that by setting $\pi_1(s_1) \in \arg\max_a \hat{Q}_1^1(s_1, a)$, i.e., by acting greedily according to $\hat{Q}_1^1$ for the first timestep, π_1 should take optimal actions most of the time. Furthermore, if we fix π_1 for the remainder of training, then this means there is a fixed distribution over s_2, meaning we can also regress $\hat{Q}_2^1 \approx Q_2^1$, and thus learn π_2; then we can repeat this process as in GORP to learn policies $\pi_3, \ldots, \pi_T$ for all timesteps.

Extending to $k - 1$ steps of Q iteration While this explains how to extend GORP to stochastic environments when $k = 1$, what about when $k > 1$? In this case, GORP follows the first action of the k-action sequence with the highest estimated return. However, in stochastic environments, it rarely makes sense to consider a fixed k-action sequence, since generally after taking one action the agent must base its next action the specific state it reached. Thus, again it is unclear how to extend this part of GORP to the stochastic case. To overcome this challenge, we combine two insights. First, we can reformulate picking the (first action of the) action sequence with the highest estimated return as a series of Bellman backups, as shown in Figure 1a.

Approximating backups with fitted Q iteration Our second insight is that we can implement these backups in stochastic environments via fitted-Q iteration [21], which estimates Q_t^j by regressing from samples of the form (s_t, a_t, y), where $y = R_t(s_t, a_t) + \max_{a_{t+1} \in \mathcal{A}} Q_{t+1}^{j-1}(s_{t+1}, a_{t+1})$. Thus, we can implement the $k-1$ backups of GORP by performing $k-1$ steps of fitted-Q iteration. This allows us to extend GORP to stochastic environments when $k > 1$. Putting together these insights gives the shallow Q-iteration via reinforcement learning (SQIRL) algorithm, which is presented in full as Algorithm 1.

Regression assumptions To implement the regression and FQI steps, SQIRL uses a *regression oracle* $\text{REGRESS}(\{(s^j, a^j, y^j)_{j=1}^m\})$ which takes as input a dataset of tuples (s^j, a^j, y^j) for $j \in [m]$ and outputs a function $\hat{Q} : \mathcal{S} \times \mathcal{A} \to [0, 1]$ that aims to predict $E[y \mid s, a]$. In order to analyze the sample complexity of SQIRL, we require the regression oracle to satisfy some basic properties, which we formalize in the following assumption.

Assumption B.1 (Regression oracle conditions)**.** *Suppose the codomain of the regression oracle* $\text{REGRESS}(\cdot)$ *is* $\mathcal{H}$. *Define* $\mathcal{V} = \{V(s) = \max_{a \in \mathcal{A}} Q(s, a) \mid Q \in \mathcal{H}\}$ *as the class of possible value functions induced by outputs of* REGRESS. *We assume there are functions* $F : (0, 1] \to (0, \infty)$ *and* $G : [1, \infty) \times (0, 1] \to (0, \infty)$ *such that the following conditions hold.*

(Regression) *Let* $Q = Q_t^1$ *for any* $t \in [T]$. *Suppose a dataset* $\{(s, a, y)\}_{j=1}^m$ *is sampled i.i.d. from a distribution* $\mathcal{D}$ *such that* $y \in [0, 1]$ *almost surely and* $E_{\mathcal{D}}[y \mid s, a] = Q(s, a)$. *Then with probability greater than* $1 - \delta$ *over the sample,*

$$E_{\mathcal{D}}\big[(\hat{Q}(s, a) - Q(s, a))^2\big] \leq O\big(\tfrac{\log m}{m} F(\delta)\big) \qquad \textit{where} \quad \hat{Q} = \text{REGRESS}\big(\{(s^j, a^j, y^j)\}_{j=1}^m\big).$$

(Fitted Q-iteration) *Let* $Q = Q_t^i$ *for any* $t \in [T-1]$ *and* $i \in [k-1]$*; define* $V(s) = \max_{a \in \mathcal{A}} Q_{t+1}^{i-1}(s, a)$. *Suppose a dataset* $\{(s, a, s')\}_{j=1}^m$ *is sampled i.i.d. from a distribution* $\mathcal{D}$ *such that* $s' \sim p_t(\cdot \mid s, a)$. *Then with probability greater than* $1 - \delta$ *over the sample, we have for all* $\hat{V} \in \mathcal{V}$ *uniformly,*

$$\begin{aligned} &E_{\mathcal{D}}\big[(\hat{Q}(s, a) - Q(s, a))^2\big]^{1/2} \leq \alpha E_{\mathcal{D}}\big[(\hat{V}(s') - V(s'))^2\big]^{1/2} + O\Big(\sqrt{\tfrac{\log m}{m} G(\alpha, \delta)}\Big) \\ &\qquad \textit{where} \quad \hat{Q} = \text{REGRESS}(\{(s^j, a^j, R_t(s^j, a^j) + \hat{V}'(s'^j))\}_{j=1}^m). \end{aligned}$$

While the conditions in Assumption B.1 may seem complex, they are relatively mild: we show in Appendix C that they are satisfied by a broad class of regression oracles. The first condition simply says that the regression oracle can take i.i.d. unbiased samples of the random policy's Q-function and accurately estimate it in-distribution. The error must decrease as $\widetilde{O}(F(\delta)/m)$ as the sample size m increases for some $F(\delta)$ which depends on the regression oracle. For instance, we will show that least-squares regression over a hypothesis class of pseudo-dimension d dimensions satisfies the first condition with $F(\delta) = \widetilde{O}(d + \log(1/\delta))$.

The second condition is a bit more unusual. It controls how error propagates from an approximate value function at timestep $t+1$ to a Q-function estimated via FQI from the value function at timestep t. In particular, the assumption requires that the root mean square (RMS) error in the Q-function be at most α times the RMS error in the value function, plus an additional term of $\widetilde{O}(\sqrt{G(\alpha, \delta)/m})$ where $G(\alpha, \delta)$ can again depend on the regression oracle used. In linear MDPs, we can show that this condition is also satisfied by linear regression with $\alpha = 1$ and $G(1, \delta) = \widetilde{O}(d + \log(1/\delta))$.

We now present a formalization of Theorem 3.5 (stated informally in Section 3):

Theorem B.2 (SQIRL sample complexity)**.** *Fix* $\alpha \geq 1$, $\delta \in (0, 1]$, *and* $\epsilon \in (0, 1]$. *Suppose* REGRESS *satisfies Assumption B.1 and let* $D = F(\frac{\delta}{kT}) + G(\alpha, \frac{\delta}{kT})$. *Then if the MDP is k-QVI-solvable for some* $k \in [T]$, *there is a univeral constant* C *such that SQIRL (Algorithm 1) will return an* ϵ*-optimal policy with probability at least* $1 - \delta$ *if* $m \geq C \frac{kT\alpha^{2(k-1)} A^k D}{\Delta_k^2 \epsilon} \log \frac{kT\alpha AD}{\Delta_k \epsilon}$. *Thus, the sample complexity of SQIRL is at most*

$$N_{\epsilon,\delta}^{\textit{SQIRL}} \leq \widetilde{O}\left(kT^3 \alpha^{2(k-1)} A^{\bar{H}_k} D \log(\alpha D)/\epsilon\right). \tag{2}$$

To understand the bound on the sample complexity of SQIRL given in (2), first compare it to GORP's

Setting	Sample complexity bounds	
	Strategic exploration	SQIRL
Tabular MDP	$\widetilde{O}(T^5 SA/\epsilon^2)$	$\widetilde{O}(kT^3 SA^{\bar{H}_k+1}/\epsilon)$
Linear MDP	$\widetilde{O}(T^4 d^2/\epsilon^2)$	$\widetilde{O}(kT^3 dA^{\bar{H}_k}/\epsilon)$
Q-functions with finite pseudo-dimension	—	$\widetilde{O}(k5^k T^3 dA^{\bar{H}_k}/\epsilon)$

Table 3: A comparison of our bounds for the sample complexity of SQIRL with bounds from the literature on strategic exploration [5, 11, 13]. SQIRL can solve stochastic MDPs with a sample complexity that is exponential only in the effective horizon $\bar{H}_k$. Also, since SQIRL only requires a regression oracle that can estimate Q-functions, it can be used with a wide variety of function classes, including any with finite pseudo-dimension. In contrast, it is difficult to extend the bounds for strategic exploration to more general function classes.

sample complexity in Lemma 3.4. Like GORP, SQIRL has sample complexity exponential in only the effective horizon. As we will see, in many cases we can set $\alpha = 1$ and $D \asymp d + \log(kT/\delta)$, where d is the pseudo-dimension of the hypothesis class used by the regression oracle. Then, the sample complexity of SQIRL is $\widetilde{O}(kT^3 A^{\bar{H}_k} d/\epsilon)$—ignoring log factors, just a Td/ϵ factor more than the sample complexity of GORP. The additional factor of d is necessary because SQIRL must learn a Q-function that generalizes over many states, while GORP can estimate the Q-values at a single state in deterministic environments. The $1/\epsilon$ dependence on the desired suboptimality is standard for stochastic environments; for instance, see the strategic exploration bounds in Table 3.

Types of regression oracles In Appendix C, we show that a broad class of regression oracles satisfy Assumption B.1. This gives sample complexity bounds shown in Table 3 for SQIRL in tabular and linear MDPs, two settings which are well studied in the strategic exploration literature. However, we find that SQIRL can also solve a much broader range of environments than strategic exploration. For instance, if the regression oracle is implemented via least-squares optimization over a hypothesis class with finite pseudo-dimension d, and that hypothesis class contains $Q^1, \ldots, Q^k$, then we obtain a $\widetilde{O}(k5^k T^3 dA^{\bar{H}_k}/\epsilon)$ bound on SQIRL's sample complexity. In contrast, it has thus far proven intractable to study strategic exploration algorithms in such general environments.

When considering our bounds on SQIRL, note that in realistic cases k is quite small. As shown in Figure 2, many environments can be approximately solved with $k = 1$. We also run all experiments in Section 4 with $k \leq 3$. Thus, although SQIRL's sample complexity is exponential in k, in practice this is fine. Overall, our analysis of the SQIRL algorithm shows theoretically why RL can succeed in complex environments while using random exploration and function approximation. We now turn to validating our theoretical insights empirically.

C Least-squares regression oracles

In this appendix, we prove that many least-squares regression oracles satisfy Assumption B.1 and thus can be used in SQIRL. These regression oracles minimize the empirical least-squares loss on the training data over some hypothesis class $\mathcal{H}$: $\textsc{Regress}(\{(s^j, a^j, y^j)\}_{j=1}^m) = \arg\min_{Q \in \mathcal{H}} \frac{1}{m} \sum_{j=1}^m (Q(s^j, a^j) - y^j)^2$. Proving that Assumption B.1 is satisfied for such an oracle depends on some basic properties of $\mathcal{H}$. First, we require that $\mathcal{H}$ is of bounded complexity, since otherwise it is impossible to learn a Q-function that generalizes well. We formalize this by requiring a simple bound on the covering number of $\mathcal{H}$:

Definition C.1. *Suppose $\mathcal{H}$ is a hypothesis class of functions $Q : \mathcal{S} \times \mathcal{A} \to [0, 1]$. We say $\mathcal{H}$ is a* VC-type *hypothesis class if for any probability measure P over $\mathcal{S} \times \mathcal{A}$, the $L_2(P)$ covering number of $\mathcal{H}$ is bounded as $N(\mathcal{H}, L_2(P); \epsilon) \leq \left(\frac{B}{\epsilon}\right)^d$, where $\|Q - Q'\|_{L_2(P)}^2 = E_P[(Q(s, a) - Q'(s, a))^2]$.*

Many hypothesis classes are VC-type. For instance, if $\mathcal{H}$ has finite pseudo-dimension d, then it is VC-type with $d = d$ and $B = O(1)$. If $\mathcal{H}$ is parameterized by θ in a bounded subset of $\mathbb{R}^d$ and Q^θ is Lipschitz in its parameters, then $\mathcal{H}$ is also VC-type with $d = d$ and $B = O(\log(\rho L))$, where $\|\theta\|_2 \leq \rho$ and L is the Lipschitz constant. See Appendix C.1 for more information.

Besides bounding the complexity of $\mathcal{H}$, we also need it to be rich enough to fit the Q-functions in the

MDP. We formalize this in the following two conditions.

Definition C.2. *We say $\mathcal{H}$ is* k-realizable *if for all $i \in [k]$ and $t \in [T]$, $Q_t^i \in \mathcal{H}$.*

Definition C.3. *We say $\mathcal{H}$ is* closed under QVI *if for any $t \in \{2, \ldots, T\}$, $\hat{Q}_t \in \mathcal{H}$ implies that QVI$(\hat{Q}_t) \in \mathcal{H}$.*

Assuming that $\mathcal{H}$ is k-realizable is very mild: we would expect that function approximation-based RL would not work at all if the function approximators cannot fit Q-functions in the MDP. The second assumption, that $\mathcal{H}$ is closed under QVI, is more restrictive. However, it turns out this is not necessary for proving that Assumption B.1 is satisfied; if $\mathcal{H}$ is not closed under QVI, then it just results in slightly worse sample complexity bounds.

Theorem C.4. *Suppose $\mathcal{H}$ is k-realizable and of VC-type for constants B and d. Then least squares regression over $\mathcal{H}$ satisfies Assumption B.1 with*

$$F(\delta) = O\left(d \log(Bd) + \log(1/\delta)\right)$$
$$G(\alpha, \delta) = O\left(\left(d \log(ABd/(\alpha - 2)) + \log(1/\delta)\right)/(\alpha - 2)^4\right).$$

Furthermore, if $\mathcal{H}$ is also closed under QVI, then we can remove all $(\alpha - 2)$ factors in G.

Theorem C.4 allows us to immediately bound the sample complexity bounds of SQIRL in a number of settings. For instance, consider a linear MDP with state-action features $\phi(s, a) \in \mathbb{R}^d$. We can let $\mathcal{H} = \{\hat{Q}(s, a) = w^\top \phi(s, a) \mid w^\top \phi(s, a) \in [0, 1] \quad \forall (s, a) \in \mathcal{S} \times \mathcal{A}\}$. This hypothesis class is realizable for any k, closed under QVI, and of VC-type, meaning SQIRL's sample complexity is at most $\widetilde{O}(kT^3 dA^{\bar{H}_k}/\epsilon)$. Since tabular MDPs are a special case of linear MDPs with $d = SA$, this gives bounds for the tabular case as well. Table 3 shows a comparison between these bounds and previously known bounds for *strategic* exploration.

However, our analysis can also handle much more general cases than any strategic exploration bounds in the literature. For instance, suppose $\mathcal{H}$ consists of neural networks with n parameters and ℓ layers, and say that $\mathcal{H}$ is k-realizable, but not necessarily closed under QVI. Then $\mathcal{H}$ has pseudo-dimension of $d = O(n\ell \log(n))$ [22] and we can bound the sample complexity of SQIRL by $\widetilde{O}(k 5^k T^3 n \ell A^{\bar{H}_k}/\epsilon)$, where we use $\alpha = \sqrt{5}$.

C.1 VC-type hypothesis classes

We now describe two cases when hypothesis classes are of VC-type; thus, by Theorem C.4 these hypothesis classes satisfy Assumption B.1 and can be used as part of SQIRL.

Example C.5. *We say $\mathcal{H}$ has pseudo-dimension d if the collection of all subgraphs of the functions in $\mathcal{H}$ forms a class of sets with VC dimension d. Then by Theorem 2.6.7 of van der Vaart and Wellner [23], $\mathcal{H}$ is a VC-type class with*

$$N(\mathcal{H}, L_2(P); \epsilon) \leq Cd(16e)^d \left(\frac{1}{\epsilon}\right)^{2(d-1)} = O\left(\left(\frac{4e}{\epsilon}\right)^{2d}\right).$$

Example C.6. *Suppose $\mathcal{H}$ is parameterized by $\theta \in \Theta \subseteq \mathbb{R}^d$ with $\|\theta\|_2 \leq B \; \forall \theta$, and $Q_\theta(s, a)$ is Lipschitz in θ, i.e.,*

$$|Q_\theta(s, a) - Q_{\theta'}(s, a)| \leq L\|\theta - \theta'\|_2 \quad \forall \theta, \theta' \in \Theta.$$

By Corollary 4.2.13 of Vershynin [24], the ϵ-covering number of $\{\theta \in \mathbb{R}^d \mid \|\theta\|_2 \leq B\}$ is bounded as $(1 + 2B/\epsilon)^d$. Therefore, the ϵ-packing number of $\mathcal{H}$ is bounded as $(1 + 4B/\epsilon)^d$ (Lemma 4.2.8 of Vershynin [24]); this in turn implies that the ϵ packing number of Θ is bounded identically, since any ϵ-packing of Θ is also an ϵ-packing of $\mathcal{H}$, which means that the ϵ-covering number of Θ is also bounded as $(1 + 4B/\epsilon)^d$. If we take $\mathcal{N}_{\epsilon/L}$ to be an ϵ/L-covering of Θ, then for any Q_θ, there must be some $\theta' \in \mathcal{N}_{\epsilon/L}$ such that $\|\theta - \theta'\|_2 \leq \epsilon/L$, which implies for any probability measure P that

$$\begin{aligned}
\|Q_\theta(s, a) - Q_{\theta'}(s, a)\|_{L_2(P)} &= E_P\left[\left(Q_\theta(s, a) - Q_{\theta'}(s, a)\right)^2\right]^{1/2} \\
&\leq E_P\left[L^2 \|\theta - \theta'\|_2^2\right]^{1/2} \\
&= L\|\theta - \theta'\|_2 \leq \epsilon.
\end{aligned}$$

Thus $\{Q_\theta \mid \theta \in \mathcal{N}_{\epsilon/L}\}$ is an ϵ-covering of $\mathcal{H}$, which implies that the $L_2(P)$ covering number of $\mathcal{H}$ is bounded as

$$N(\mathcal{H}, L_2(P); \epsilon) \leq N(\Theta, L_2(P); \epsilon/L) \leq (1 + 4BL/\epsilon)^d = O\left(\left(\frac{4BL}{\epsilon}\right)^d\right).$$

D Proofs

D.1 Proof of Lemma 3.4

Lemma 3.4. *The deterministic effective horizon H is bounded for any $k \in [T]$ as*

$$H \leq \min_k \left[\bar{H}_k + \log_A O\left(\log\left(TA^k\right)\right)\right].$$

Furthermore, if an MDP is k-QVI-solvable, then with probability at least $1-\delta$, GORP will return an optimal policy with sample complexity at most $O(kT^2 A^{\bar{H}_k} \log(TA/\delta))$.

Proof. The bound on H follows immediately from Theorem 5.4 of Laidlaw et al. [1] by noticing that in our setting, the Q and value functions are always upper-bounded by 1. The bound the sample complexity of GORP then follows from Lemma 5.3 of Laidlaw et al. [1]. ■

D.2 Proof of Theorem B.2

To prove our bounds on the sample complexity of SQIRL, we first introduce a series of auxiliary lemmas.

Lemma D.1. *Suppose that an MDP is k-QVI solvable and we iteratively find deterministic policies $\pi_1, \ldots, \pi_T$ such that for each t, $P_\pi(\pi_t(s_t) \notin \arg\max_a Q_t^k(s_t, a)) \leq \epsilon/T$, where states s_t are sampled by following policies $\pi_1, \ldots, \pi_{t-1}$ for timesteps 1 to $t-1$. Then π is ϵ-optimal in the overall MDP, i.e.*

$$J(\pi) \geq \max_{\pi^*} J(\pi^*) - \epsilon.$$

Proof. Let $\mathcal{E}$ denote the event that there is some $t \in [T]$ when $a_t \notin \arg\max_a Q_t^k(s_t, a)$. By a union bound, we have $P_\pi(\mathcal{E}) \leq \epsilon$. Now, let π^* be a policy in $\Pi(Q^k)$ that agrees with π at all states and timesteps where $\pi_t(s) \in \arg\max_a Q_t^k(s, a)$. We can write $\tilde{\mathcal{E}}$ as the event that $\exists t \in [T]$, $\pi(s_t) \neq \pi^*(s_t)$, which is equivalent to $\mathcal{E}$ under the distribution induced by π. We can now decompose $J(\pi)$ as

$$\begin{aligned}
J(\pi) &= E_\pi\left[\sum_{t=1}^T R_t(s_t, a_t)\right] \\
&\geq E_\pi\left[\sum_{t=1}^T R_t(s_t, a_t) \mid \neg\mathcal{E}\right] P_\pi(\neg\mathcal{E}) \\
&= E_{\pi^*}\left[\sum_{t=1}^T R_t(s_t, a_t) \mid \neg\tilde{\mathcal{E}}\right] P_\pi(\neg\mathcal{E}) \\
&= E_{\pi^*}\left[\sum_{t=1}^T R_t(s_t, a_t) \mid \neg\tilde{\mathcal{E}}\right] P_\pi(\neg\mathcal{E}) + E_{\pi^*}\left[\sum_{t=1}^T R_t(s_t, a_t) \mid \tilde{\mathcal{E}}\right] (P_\pi(\mathcal{E}) - P_\pi(\mathcal{E})) \\
&= E_{\pi^*}\left[\sum_{t=1}^T R_t(s_t, a_t)\right] - E_{\pi^*}\left[\sum_{t=1}^T R_t(s_t, a_t) \mid \tilde{\mathcal{E}}\right] P_\pi(\mathcal{E}) \\
&\geq J(\pi^*) - \epsilon.
\end{aligned}$$

■

Lemma D.2. *Let $\mathcal{D}$ be a distribution over states and actions such that $P_\mathcal{D}(a \mid s) = 1/A$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}$. Then for any Q and $\hat{Q} : \mathcal{S} \times \mathcal{A} \to [0, 1]$, defining $V(s) = \max_{a \in \mathcal{A}} Q(s, a)$ and $\hat{V}$ analogously, we have*

$$E_\mathcal{D}\left[\left(\hat{V}(s) - V(s)\right)^2\right] \leq A E_\mathcal{D}\left[\left(\hat{Q}(s, a) - Q(s, a)\right)^2\right].$$

Proof. We have

$$\begin{aligned}
E_{\mathcal{D}}\left[\left(\hat{V}(s)-V(s)\right)^2\right] &= E_{\mathcal{D}}\left[\left(\max_{a\in\mathcal{A}}\hat{Q}(s,a)-\max_{a\in\mathcal{A}}Q(s,a)\right)^2\right] \\
&\leq E_{\mathcal{D}}\left[\max_{a\in\mathcal{A}}\left(\hat{Q}(s,a)-Q(s,a)\right)^2\right] \\
&\leq E_{\mathcal{D}}\left[\sum_{a\in\mathcal{A}}\left(\hat{Q}(s,a)-Q(s,a)\right)^2\right] \\
&= AE_{\mathcal{D}}\left[\frac{1}{A}\sum_{a\in\mathcal{A}}\left(\hat{Q}(s,a)-Q(s,a)\right)^2\right] \\
&= AE_{\mathcal{D}}\left[\left(\hat{Q}(s,a)-Q(s,a)\right)^2\right],
\end{aligned}$$

where the final equality follows from the fact that $P_{\mathcal{D}}(a \mid s) = 1/A$. ■

Lemma D.3. *Suppose the MDP is k-QVI solvable and let π_i be the policy constructed by stochastic GORP at timestep i. Then with probability at least $1-\delta/T$,*

$$P_\pi\left(\pi_i(s)\notin\arg\max_a Q_i^k(s,a)\right) \leq O\left(\frac{\alpha^{2k-2}A^k\left(F(\frac{\delta}{kT})+G(\alpha,\frac{\delta}{kT})\right)\log m}{m\Delta_k^2}\right).$$

Proof. Let all expectations and probabilities E and P be with respect to the distribution of states and actions induced by following $\pi_1,\ldots,\pi_{i-1}$ for $t<i$ and π^{rand} thereafter. To simplify notation, we write for any $Q_t, Q_t' : \mathcal{S}\times\mathcal{A}\to[0,1]$ or $V_t, V_t' : \mathcal{S}\to[0,1]$,

$$\begin{aligned}
\|Q_t - Q_t'\|_2^2 &= E\left[(Q_t(s_t,a_t)-Q_t'(s_t,a_t))^2\right] \\
\|V_t - V_t'\|_2^2 &= E\left[(V_t(s_t)-V_t'(s_t))^2\right].
\end{aligned}$$

Let $\hat{V}_t^{i+k-t}(s) = \max_{a\in\mathcal{A}}\hat{Q}_t^{i+k-t}(s,a)$ and $V_t^{i+k-t}(s) = \max_{a\in\mathcal{A}}Q_t^{i+k-t}(s,a)$. Consider the following three facts:

1. By Assumption B.1 part 1, with probability at least $1-\delta/(kT)$,

$$\left\|\hat{Q}_{i+k-1}^1 - Q_{i+k-1}^1\right\|_2^2 \leq C_1\frac{F(\frac{\delta}{kT})\log m}{m}.$$

2. By Lemma D.2, for all $t\in\{2,\ldots,k\}$,

$$\left\|\hat{V}_t^{i+k-t}-V_t^{i+k-t}\right\|_2^2 \leq A\left\|\hat{Q}_t^{i+k-t}-Q_t^{i+k-t}\right\|_2^2.$$

3. By Assumption B.1 part 2, for any $t\in\{1,\ldots,k-1\}$, with probability at least $1-\delta/(kT)$

$$\left\|\hat{Q}_t^{i+k-t}-Q_t^{i+k-t}\right\|_2 \leq \alpha\left\|\hat{V}_{t+1}^{i+k-t-1}-V_{t+1}^{i+k-t-1}\right\|_2 + \sqrt{C_2\frac{G(\alpha,\frac{\delta}{kT})\log m}{m}}.$$

   Note that it is key that this bound is uniform over all $\hat{V}_{t+1}^{i+k-t-1}\in\mathcal{V}$, since $\hat{V}_{t+1}^{i+k-t-1}$ is estimated based on the same data used to regress $\hat{Q}_t^{i+k-t}$.

Via a union bound all of the above facts hold with probability at least $1-\delta/T$. We will combine them to recursively show for $t\in\{1,\ldots,k\}$,

$$\left\|\hat{Q}_t^{i+k-t}-Q_t^{i+k-t}\right\|_2 \leq \left(4(\alpha\sqrt{A})^{k-t}-3\right)\sqrt{\frac{D\log m}{m}} \qquad \text{where} \quad D = C_1F(\tfrac{\delta}{kT})+C_2G(\alpha,\tfrac{\delta}{kT}). \tag{3}$$

The base case $t = k$ is true by fact 1. Now let $t < k$ and assume the above holds for $t + 1$. By facts 2 and 3,

$$\begin{aligned}
\left\|\hat{Q}_t^{i+k-t} - Q_t^{i+k-t}\right\|_2 &\leq \alpha \left\|\hat{V}_{t+1}^{i+k-t-1} - V_{t+1}^{i+k-t-1}\right\|_2 + \sqrt{C_2 \frac{G(\alpha, \frac{\delta}{kT}) \log m}{m}} \\
&\leq \alpha\sqrt{A} \left\|\hat{Q}_{t+1}^{i+k-t-1} - Q_{t+1}^{i+k-t-1}\right\|_2 + \sqrt{C_2 \frac{G(\alpha, \frac{\delta}{kT}) \log m}{m}} \\
&\leq \alpha\sqrt{A} \left(4(\alpha\sqrt{A})^{k-t-1} - 3\right) \sqrt{\frac{D \log m}{m}} + \sqrt{\frac{D \log m}{m}} \\
&= \left(4(\alpha\sqrt{A})^{k-t} - 3\alpha\sqrt{A} + 1\right) \sqrt{\frac{D \log m}{m}} \\
&\leq \left(4(\alpha\sqrt{A})^{k-t} - 3\right) \sqrt{\frac{D \log m}{m}},
\end{aligned}$$

where the last inequality follows from $A \geq 2$ and $\alpha \geq 1$. Thus, by setting $t = i$ in (3), we see that with probability at least $1 - \delta/T$,

$$\left\|\hat{Q}_i^k - Q_i^k\right\|_2^2 \leq O\left(\frac{\alpha^{2k-2} A^{k-1} \left(F(\frac{\delta}{kT}) + G(\alpha, \frac{\delta}{kT})\right) \log m}{m}\right) \tag{4}$$

$$\begin{aligned}
&P_\pi(\pi_i(s_i) \notin \arg\max_a Q_i^k(s_i, a) \\
&\leq P_\pi\left(\arg\max_a \hat{Q}_i^k(s_i, a) \not\subseteq \arg\max_a Q_i^k(s_i, a)\right) \\
&\overset{\text{(i)}}{\leq} P_\pi\left(\exists a \in \mathcal{A} \quad \text{s.t.} \quad \left|\hat{Q}_i^k(s_i, a) - Q_i^k(s_i, a)\right| \geq \Delta_k/2\right) \\
&\leq \sum_{a \in \mathcal{A}} P_\pi\left(\left|\hat{Q}_i^k(s_i, a) - Q_i^k(s_i, a)\right| \geq \Delta_k/2\right) \\
&= A\left(\frac{1}{A} \sum_{a \in \mathcal{A}} P_\pi\left(\left|\hat{Q}_i^k(s_i, a) - Q_i^k(s_i, a)\right| \geq \Delta_k/2\right)\right) \\
&= AP_\pi\left(\left|\hat{Q}_i^k(s_i, a_i) - Q_i^k(s_i, a_i)\right| \geq \Delta_k/2\right) \\
&\overset{\text{(ii)}}{\leq} \frac{A}{(\Delta_k/2)^2} E_\pi\left[\left(\hat{Q}_i^k(s_i, a_i) - Q_i^k(s_i, a_i)\right)^2\right] \\
&= \frac{A}{(\Delta_k/2)^2} \left\|\hat{Q}_i^k - Q_i^k\right\|_2^2 \\
&= O\left(\frac{\alpha^{2k-2} A^k \left(F(\frac{\delta}{kT}) + G(\alpha, \frac{\delta}{kT})\right) \log m}{m\Delta_k^2}\right).
\end{aligned}$$

Here, (i) follows from Definition 3.2 of the k-gap and (ii) follows from Markov's inequality. ■

Theorem B.2 (SQIRL sample complexity)**.** *Fix $\alpha \geq 1$, $\delta \in (0, 1]$, and $\epsilon \in (0, 1]$. Suppose* Regress *satisfies Assumption B.1 and let $D = F(\frac{\delta}{kT}) + G(\alpha, \frac{\delta}{kT})$. Then if the MDP is k-QVI-solvable for some $k \in [T]$, there is a univeral constant C such that SQIRL (Algorithm 1) will return an ϵ-optimal policy with probability at least $1 - \delta$ if $m \geq C \frac{kT\alpha^{2(k-1)} A^k D}{\Delta_k^2 \epsilon} \log \frac{kT\alpha AD}{\Delta_k \epsilon}$. Thus, the sample complexity of SQIRL is at most*

$$N_{\epsilon,\delta}^{\textit{SQIRL}} \leq \widetilde{O}\left(kT^3\alpha^{2(k-1)} A^{\bar{H}_k} D \log(\alpha D)/\epsilon\right). \tag{2}$$

Proof. Given the lower bound on m, we can bound

$$\frac{\log m}{m} = O\left(\frac{\Delta_k^2 \epsilon}{T\alpha^{2k-2} A^k D}\right).$$

Combining this with Lemma D.3, we see that with probability at least $1-\delta$, for all $i \in [T]$

$$P_\pi\left(\pi_i(s_i) \notin \arg\max_a Q_i^k(s_i, a)\right) \leq \epsilon/T.$$

Thus, by Lemma D.1, π is ϵ-optimal in the overall MDP $\mathcal{M}$. ■

D.3 Proof of Theorem C.4

Theorem C.4. *Suppose $\mathcal{H}$ is k-realizable and of VC-type for constants B and d. Then least squares regression over $\mathcal{H}$ satisfies Assumption B.1 with*

$$F(\delta) = O\left(d\log(Bd) + \log(1/\delta)\right)$$
$$G(\alpha, \delta) = O\left(\left(d\log(ABd/(\alpha-2)) + \log(1/\delta)\right)/(\alpha-2)^4\right).$$

Furthermore, if $\mathcal{H}$ is also closed under QVI, then we can remove all $(\alpha - 2)$ factors in G.

Proof. Throughout the proof, we will use the notation that $\|Q - Q'\|_2^2 = E_\mathcal{D}[(Q(s,a) - Q'(s,a))^2]$ and $\|V - V'\|_2^2 = E_\mathcal{D}[(V(s') - V'(s'))^2]$.

First, we will prove the regression part of Assumption B.1. To do so, we use results on least-squares regression from Koltchinskii [25]. Note that our definition of VC-type classes coincides with condition (2.1) in Koltchinskii [25]. By combining Example 3 from Section 2.5 and Theorem 13 of Koltchinskii [25], we have that for any $\bar{Q}(s,a)$ with $\mathbb{E}[y \mid s, a] = \bar{Q}(s,a)$, and for any $\lambda \in (0, 1]$,

$$\mathbb{P}\left(\left\|\hat{Q} - \bar{Q}\right\|_2^2 \leq (1+\lambda)\inf_{\tilde{Q}\in\mathcal{H}}\left\|\tilde{Q} - \bar{Q}\right\|_2^2 + O\left(\frac{d}{m\lambda^2}\log\left(\frac{Bdm}{\lambda}\right) + \frac{u+1}{\lambda m}\right)\right) \leq \log\left(\frac{em}{u}\right)e^{-u}.$$
$$\text{where} \quad \hat{Q} = \textsc{Regress}(\{(s^j, a^j, y^j)\}_{j=1}^m). \tag{5}$$

If we set

$$u = \log\left(\frac{e + \log m}{\delta}\right) \geq 1,$$

then the right-hand side of (5) is bounded as

$$\log\left(\frac{em}{u}\right)e^{-u} \leq \log(em)e^{-u} = \log(em)\frac{\delta}{e + \log m} < \delta.$$

Thus, plugging this value of u into (5), we have that with probability at least $1-\delta$,

$$\left\|\hat{Q} - \bar{Q}\right\|_2^2 \leq (1+\lambda)\inf_{\tilde{Q}\in\mathcal{H}}\left\|\tilde{Q} - \bar{Q}\right\|_2^2 + O\left(\frac{d\log\left(\frac{Bdm}{\lambda}\right) + \log\frac{m}{\delta}}{m\lambda^2}\right). \tag{6}$$

For the regression condition of Assumption B.1, we have $\bar{Q} = Q_t^k \in \mathcal{H}$. Thus, $\inf_{\tilde{Q}\in\mathcal{H}}\|\tilde{Q} - \bar{Q}\|_2^2 = 0$, and we can set $\lambda = 1$ in (6) to obtain

$$\left\|\hat{Q} - Q_t^k\right\|_2^2 \leq O\left(\frac{d\log(Bdm) + \log\frac{m}{\delta}}{m}\right),$$

leading to the desired bound of

$$F(\delta) = O\left(d\log(Bd) + \log\frac{1}{\delta}\right).$$

To the fitted Q-iteration condition of Assumption B.1, we begin by defining a norm ρ on $\mathcal{V} \times \mathcal{H}$ by

$$\rho\Big((V, Q), (V', Q')\Big) = \max\{\|V - V'\|_2, \|Q - Q'\|_2\}.$$

Note that since we showed in Lemma D.2 that

$$E_\mathcal{D}\left[(V(s') - V'(s'))^2\right] \leq A E_{\mathcal{D}, a'\sim\text{Unif}(\mathcal{A})}\left[(Q(s', a') - Q'(s', a'))^2\right]$$
$$\text{where} \quad V(s') = \max_{a'\in\mathcal{A}} Q(s', a'), V'(s') = \max_{a'\in\mathcal{A}} Q'(s', a'),$$

this implies that any ϵ-cover of $\mathcal{H}$ is also an $\epsilon\sqrt{A}$-cover of $\mathcal{V}$ with respect to $L_2(P)$ for any distribution

P over s'. Thus, by the definition of VC-type classes, we have

$$N(\mathcal{V}, L_2(\mathcal{D}); \epsilon) \le \left(\frac{B\sqrt{A}}{\epsilon}\right)^d$$
$$N(\mathcal{H}, L_2(\mathcal{D}); \epsilon) \le \left(\frac{B}{\epsilon}\right)^d$$
$$N(\mathcal{V} \times \mathcal{H}, \rho; \epsilon) \le \left(\frac{B\sqrt{A}}{\epsilon}\right)^d \left(\frac{B}{\epsilon}\right)^d \le \left(\frac{B\sqrt{A}}{\epsilon}\right)^{2d}.$$

Now define $\mathcal{W} \subseteq \mathcal{V} \times \mathcal{H}$ as

$$\mathcal{W} = \left\{ (\hat{V}, \hat{Q}) \in \mathcal{V} \times \mathcal{H} \;\middle|\; \hat{Q} = \textsc{Regress}\left(\left\{ (s^j, a^j, R_t(s^j, a^j) + \hat{V}(s'^j)) \right\}_{j=1}^m \right) \right\}.$$

By properties of packing and covering numbers, since any ϵ-packing of $\mathcal{W}$ is also an ϵ-packing of $\mathcal{V} \times \mathcal{H}$, we have

$$N(\mathcal{W}, \rho; \epsilon) \le M(\mathcal{W}, \rho; \epsilon/2) \le M(\mathcal{V} \times \mathcal{H}, \rho; \epsilon/2) \le N(\mathcal{V} \times \mathcal{H}, \rho; \epsilon/2) \le \left(\frac{2B\sqrt{A}}{\epsilon}\right)^{2d}.$$

Thus, let $\mathcal{N}_{1/\sqrt{m}}$ be a $1/\sqrt{m}$-covering of $\mathcal{W}$ with size at most $(2B\sqrt{Am})^{2d}$.

Fix any $(\hat{V}, \hat{Q}) \in \mathcal{N}_{1/\sqrt{m}}$ and define $\bar{Q}(s,a) = E[R_t(s,a) + \hat{V}(s') \mid s, a]$. Then by an identical argument to (6), with probability at least $1-\delta$, for any $\lambda \in (0,1]$,

$$\left\|\hat{Q} - \bar{Q}\right\|_2^2 \le (1+\lambda) \inf_{\tilde{Q} \in \mathcal{H}} \left\|\tilde{Q} - \bar{Q}\right\|_2^2 + O\left(\frac{d \log\left(\frac{Bdm}{\lambda}\right) + \log \frac{m}{\delta}}{m\lambda^2} \right).$$

We can extend this to a bound on all $(\hat{V}, \hat{Q}) \in \mathcal{N}_{1/\sqrt{m}}$ by dividing δ by $|\mathcal{N}_{1/\sqrt{m}}|$ and applying a union bound. Thus, with probability at least $1-\delta$, for all $(\hat{V}, \hat{Q}) \in \mathcal{N}_{1/\sqrt{m}}$ and any $\lambda \in (0,1]$,

$$\begin{aligned}
\left\|\hat{Q} - \bar{Q}\right\|_2^2 &\le (1+\lambda) \inf_{\tilde{Q} \in \mathcal{H}} \left\|\tilde{Q} - \bar{Q}\right\|_2^2 + O\left(\frac{d \log\left(\frac{Bdm}{\lambda}\right) + d\log(BAm) + \log \frac{m}{\delta}}{m\lambda^2} \right) \\
&= (1+\lambda) \inf_{\tilde{Q} \in \mathcal{H}} \left\|\tilde{Q} - \bar{Q}\right\|_2^2 + O\left(\frac{d \log\left(\frac{BAdm}{\lambda}\right) + \log \frac{m}{\delta}}{m\lambda^2} \right).
\end{aligned}$$

Finally, we extend this to a bound over all $(\hat{V}, \hat{Q}) \in \mathcal{W}$. For any $(\hat{V}, \hat{Q}) \in \mathcal{W}$, there must be some $(\hat{V}', \hat{Q}') \in \mathcal{N}_{1/\sqrt{m}}$ such that $\rho((\hat{V}, \hat{Q}), (\hat{V}', \hat{Q}')) \le 1/\sqrt{m}$. Let $\bar{Q}'(s,a) = E[R_t(s,a) + \hat{V}'(s') \mid s, a]$. Then

$$\begin{aligned}
\|\bar{Q} - \bar{Q}'\|_2^2 &= E_{\mathcal{D}}\left[\left(\bar{Q}(s,a) - \bar{Q}'(s,a)\right)^2 \right] \\
&= E_{\mathcal{D}}\left[\left(E_{\mathcal{D}}\left[\hat{V}(s') - \hat{V}'(s') \right] \right)^2 \right] \\
&\le E_{\mathcal{D}}\left[\left(\hat{V}(s') - \hat{V}'(s') \right)^2 \right] \le \frac{1}{m},
\end{aligned}$$

where the second-to-last inequality follows from Jensen's inequality. Thus, by the triangle inequality,

$$\begin{aligned}
\left\|\hat{Q} - \bar{Q}\right\|_2 &\le \left\|\hat{Q} - \hat{Q}'\right\|_2 + \left\|\hat{Q}' - \bar{Q}'\right\|_2 + \left\|\bar{Q}' - \bar{Q}\right\|_2 \\
&\le \left\|\hat{Q}' - \bar{Q}'\right\|_2 + \frac{2}{\sqrt{m}} \\
&\le \sqrt{1+\lambda} \inf_{\tilde{Q}' \in \mathcal{H}} \left\|\tilde{Q}' - \bar{Q}'\right\|_2 + O\left(\sqrt{ \frac{d \log\left(\frac{BAdm}{\lambda}\right) + \log \frac{m}{\delta}}{m\lambda^2} } \right).
\end{aligned}$$

for all $(\hat{V}, \hat{Q}) \in \mathcal{W}$ and any $\lambda \in (0,1]$ with probability at least $1-\delta$.

We now consider the two possible conditions in the theorem. If $\mathcal{H}$ is both k-realizable and closed under QVI, then this implies $\bar{Q}' \in \mathcal{H}$ for all $(\hat{V}', \hat{Q}') \in \mathcal{N}_{1/\sqrt{m}}$, meaning $\inf_{\tilde{Q}' \in \mathcal{H}} \|\tilde{Q}' - \bar{Q}'\|_2 = 0$. Thus, we can set $\lambda = 1$ in the above bound to obtain

$$\begin{aligned}\left\|\hat{Q} - Q\right\|_2 &\leq \|\bar{Q} - Q\|_2 + \left\|\hat{Q} - \bar{Q}\right\|_2 \\ &\leq \left\|\hat{V} - V\right\|_2 + O\left(\sqrt{\frac{d \log\left(\frac{BAdm}{\lambda}\right) + \log \frac{m}{\delta}}{m\lambda^2}}\right),\end{aligned}$$

showing that the FQI condition of Assumption B.1 holds with

$$G(\alpha, \delta) = O\Big(d \log(BAd) + \log(1/\delta)\Big).$$

Otherwise, if $\mathcal{H}$ is only k-realizable, then this implies $Q \in \mathcal{H}$. Thus,

$$\inf_{\tilde{Q}' \in \mathcal{H}} \left\|\tilde{Q}' - \bar{Q}'\right\|_2 \leq \|Q - \bar{Q}'\|_2 \leq \left\|V - \hat{V}\right\|_2 + \frac{1}{\sqrt{m}}.$$

This implies that

$$\begin{aligned}\left\|\hat{Q} - Q\right\|_2 &\leq \|\bar{Q} - Q\|_2 + \left\|\hat{Q} - \bar{Q}\right\|_2 \\ &\leq \left\|\hat{V} - V\right\|_2 + \sqrt{1+\lambda}\left\|V - \hat{V}\right\|_2 + \frac{1}{\sqrt{m}} + O\left(\sqrt{\frac{d \log\left(\frac{BAdm}{\lambda}\right) + \log \frac{m}{\delta}}{m\lambda^2}}\right) \\ &\leq (2 + \sqrt{\lambda})\left\|\hat{V} - V\right\|_2 + O\left(\sqrt{\frac{d \log\left(\frac{BAdm}{\lambda}\right) + \log \frac{m}{\delta}}{m\lambda^2}}\right).\end{aligned}$$

Setting $\alpha = 2 + \sqrt{\lambda}$ shows that the FQI condition of Assumption B.1 holds with

$$G(\alpha, \delta) = O\left(\frac{d \log\left(\frac{BAd}{\alpha - 2}\right) + \log \frac{1}{\delta}}{(\alpha - 2)^4}\right).$$

■

E Experiment details

In this appendix, we describe details of the experiments from Section 4. We use the implementations of PPO and DQN from Stable-Baselines3 [26], and in general use their hyperparameters which have been optimized for Atari games. For network archictures, we use convolutional neural nets similar to those used by Mnih et al. [3]. We use a discount rate of $\gamma = 1$ for the Atari and Procgen environments in BRIDGE but $\gamma = 0.99$ for the MiniGrid environments, as otherwise we found that RL completely failed.

In practice, we slightly modify Algorithm 1 in a few ways for use with deep neural networks. Following standard practice in deep RL, we use a single neural network to regress the Q-function across all timesteps, rather than using a separate Q-network for each timestep. However, we still "freeze" the greedy policy at each iteration (line 8 in Algorithm 1) by storing a copy of the networks' weights from iteration i and using it for acting on timestep $t = i$ in future iterations. Second, we stabilize training by using a replay buffer to store the data collected from the environment and then sampling minibatches from it for training the Q-network. Note that neither of these changes the core algorithm: our implementation is still entirely based around iteratively estimating Q^k by using regression and fitted-Q iteration.

In each environment, we run PPO, DQN, SQIRL, and GORP for 5 million timesteps. We use the Stable-Baselines3 implementations of PPO and DQN [26]. During training, we evaluate the latest policy every 10,000 training timesteps for 100 episodes. We also calculate the exact optimal return using the tabular representations of the environments from the BRIDGE dataset; we modify the tabular representations to add sticky actions and then run value iteration. If the mean evaluation return of the algorithm reaches the optimal return, we consider the algorithm to have solved the environment. We say the empirical sample complexity of the algorithm in the environment is the number of timesteps needed to reach that optimal return.

Since SQIRL takes parameters k and m, we need to tune these parameters for each environment. For each $k \in \{1, 2, 3\}$, we perform a binary search over values of m to find the smallest value for which SQIRL solves the environment. We also slightly tune the hyperparameters of PPO and DQN; see Appendices E and F for all experiment details and results. We do not claim that SQIRL is as practical as PPO or DQN, since it requires much more hyperparameter tuning; instead, we mainly see SQIRL as a tool for understanding deep RL.

PPO We use the following hyperparameters for PPO:

Hyperparameter	Value
Training timesteps	5,000,000
Rollout length	$\{128, 1280\}$
SGD minibatch size	256
SGD epochs per iteration	4
Optimizer	Adam
Learning rate	2.5×10^{-4}
GAE coefficient (λ)	0.95
Entropy coefficient	0.01
Clipping parameter	0.1
Value function coefficient	0.5

Table 4: Hyperparameters we use for PPO.

For each environment, we try rollout lengths of 128 and 1280 as we found this was the most important parameter to tune.

DQN We use the following hyperparameters for DQN:

Hyperparameter	Value
Training timesteps	5,000,000
Timesteps before learning starts	0
Replay buffer size	100,000
Target network update frequency	8,000
Final ϵ	0.01
SGD minibatch size	32
Env. steps per gradient step	4
Optimizer	Adam
Learning rate	10^{-4}

Table 5: Hyperparameters we use for DQN.

We try decaying the ϵ value for ϵ-greedy over the course of either 500 thousand or 5 million timesteps, as we found this was the most sensitive hyperparameter to tune for DQN.

SQIRL We use the following hyperparameters for SQIRL:

Hyperparameter	Value
Training timesteps	5,000,000
Replay buffer size	1,000,000
k	{1, 2, 3}
SGD minibatch size	128
SGD epochs per iteration	10
Optimizer	Adam
Learning rate	10^{-4}

Table 6: Hyperparameters we use for SQIRL.

As we describe in the main text, we run SQIRL with $k \in \{1, 2, 3\}$ and tune m via binary search.

F Full results

In this appendix, we present our full experimental results.

F.1 k-QVI-solvability in stochastic environments

To see if stochastic environments are commonly k-QVI-solvable for small values of k, we constructed sticky-action versions of the 155 deterministic MDPs in the BRIDGE dataset [1]. Sticky actions are a common and effective method for turning deterministic MDPs into stochastic ones [27] by introducing a 25% chance at each timestep of repeating the action from the previous timestep, regardless of the new action taken. We analyzed the minimum values of k for which these MDPs are approximately k-QVI-solvable, i.e., where one can achieve at least 95% of the optimal return (measured from the minimum return) by acting greedily with respect to Q^k. The results are shown in Figure 2. Many environments are approximately k-QVI-solvable for very low values of k; more than half are approximately 1-QVI-solvable. Furthermore, these are the environments where deep RL algorithms like PPO are most likely to find an optimal policy, suggesting that k-QVI-solvability is key to deep RL's success in stochastic environments.

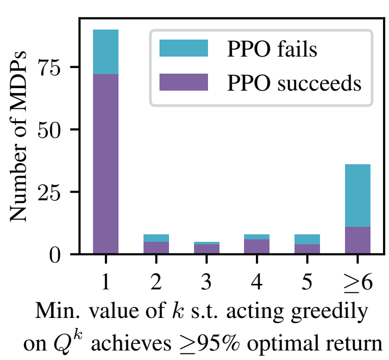


Figure 2: Among sticky-action versions of the MDPs in the BRIDGE dataset, more than half can be approximately solved by acting greedily with respect to the random policy's Q-function ($k = 1$); many more can be by applying just a few steps of Q-value iteration before acting greedily ($2 \leq k \leq 5$). When k is low, we observe that deep RL algorithms like PPO are much more likely to solve the environment.

F.2 Additional plots

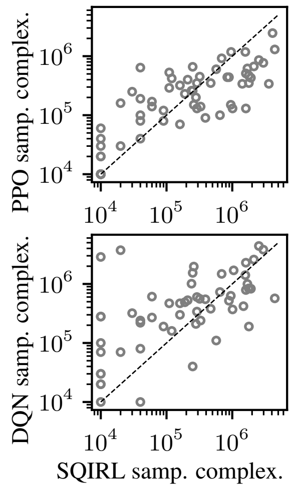


Figure 3: The empirical sample complexity of SQIRL correlates closely with that of PPO and DQN, suggesting that our theoretical analysis of SQIRL is a powerful tool for understanding when and why deep RL works in stochastic environments.

F.3 Table of empirical sample complexities

This table lists the empirical sample complexities of PPO, DQN, and SQIRL. For comparison, we also report the sample complexity of GORP [1] in each environment. If the algorithm did not solve the environment in 5 million timesteps, we write $> 5 \times 10^6$.

	PPO	DQN	SQIRL	GORP
ALIEN$_{10}$	3.34×10^6	4.82×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
AMIDAR$_{20}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
ASSAULT$_{10}$	1.00×10^4	2.80×10^5	1.00×10^4	2.93×10^4
ASTERIX$_{10}$	2.00×10^5	2.10×10^5	2.80×10^5	$> 5 \times 10^6$
ASTEROIDS$_{10}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
ATLANTIS$_{10}$	6.00×10^4	1.00×10^5	1.00×10^4	$> 5 \times 10^6$
ATLANTIS$_{20}$	3.40×10^5	$> 5 \times 10^6$	1.42×10^6	$> 5 \times 10^6$
ATLANTIS$_{30}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
ATLANTIS$_{40}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
ATLANTIS$_{50}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
ATLANTIS$_{70}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
BANKHEIST$_{10}$	8.60×10^5	4.40×10^6	2.54×10^6	$> 5 \times 10^6$
BATTLEZONE$_{10}$	1.30×10^5	2.29×10^6	1.61×10^6	$> 5 \times 10^6$
BEAMRIDER$_{20}$	$> 5 \times 10^6$	4.43×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
BOWLING$_{30}$	6.30×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
BREAKOUT$_{10}$	8.00×10^4	8.00×10^4	4.00×10^4	3.16×10^4
BREAKOUT$_{20}$	2.60×10^5	1.01×10^6	2.40×10^5	$> 5 \times 10^6$
BREAKOUT$_{30}$	2.46×10^6	$> 5 \times 10^6$	4.10×10^6	$> 5 \times 10^6$
BREAKOUT$_{40}$	1.67×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
BREAKOUT$_{50}$	1.22×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
BREAKOUT$_{70}$	2.52×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
BREAKOUT$_{100}$	2.77×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
BREAKOUT$_{200}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
CENTIPEDE$_{10}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
CHOPPERCOMMAND$_{10}$	$> 5 \times 10^6$	3.36×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
CRAZYCLIMBER$_{20}$	4.00×10^4	1.00×10^4	4.00×10^4	$> 5 \times 10^6$

CrazyClimber$_{30}$	4.20×10^5	1.60×10^5	1.20×10^5	$> 5 \times 10^6$
DemonAttack$_{10}$	2.21×10^6	5.50×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$
Enduro$_{10}$	4.50×10^5	5.30×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$
FishingDerby$_{10}$	2.10×10^5	2.05×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
Freeway$_{10}$	1.00×10^4	2.00×10^4	1.00×10^4	8.10×10^3
Freeway$_{20}$	1.00×10^4	1.00×10^4	1.00×10^4	2.30×10^6
Freeway$_{30}$	2.30×10^5	4.90×10^5	1.90×10^5	$> 5 \times 10^6$
Freeway$_{40}$	2.80×10^5	5.10×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$
Freeway$_{50}$	4.70×10^5	8.40×10^5	1.78×10^6	$> 5 \times 10^6$
Freeway$_{70}$	2.01×10^6	3.63×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
Freeway$_{100}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Freeway$_{200}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Frostbite$_{10}$	9.00×10^4	5.50×10^5	3.90×10^5	$> 5 \times 10^6$
Gopher$_{30}$	1.40×10^5	6.10×10^5	6.00×10^4	$> 5 \times 10^6$
Gopher$_{40}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Hero$_{10}$	3.00×10^4	7.00×10^4	2.00×10^4	$> 5 \times 10^6$
IceHockey$_{10}$	6.00×10^4	1.65×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
Kangaroo$_{20}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Kangaroo$_{30}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
MontezumaRevenge$_{15}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
MsPacman$_{20}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
NameThisGame$_{20}$	1.50×10^5	6.30×10^5	9.50×10^5	$> 5 \times 10^6$
Phoenix$_{10}$	8.20×10^5	4.85×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
Pong$_{20}$	6.00×10^5	1.10×10^5	5.70×10^5	$> 5 \times 10^6$
Pong$_{30}$	4.70×10^5	3.50×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$
Pong$_{40}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Pong$_{50}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Pong$_{70}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Pong$_{100}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
PrivateEye$_{10}$	1.20×10^5	1.90×10^5	9.00×10^4	$> 5 \times 10^6$
Qbert$_{10}$	1.60×10^5	3.73×10^6	2.00×10^4	$> 5 \times 10^6$
Qbert$_{20}$	5.60×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
RoadRunner$_{10}$	1.50×10^5	1.97×10^6	2.60×10^5	$> 5 \times 10^6$
Seaquest$_{10}$	2.00×10^4	2.80×10^5	1.00×10^4	$> 5 \times 10^6$
Skiing$_{10}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
SpaceInvaders$_{10}$	6.40×10^5	2.40×10^5	4.00×10^4	$> 5 \times 10^6$
Tennis$_{10}$	$> 5 \times 10^6$	1.29×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
TimePilot$_{10}$	1.00×10^5	7.30×10^5	6.50×10^5	$> 5 \times 10^6$
Tutankham$_{10}$	7.70×10^5	3.81×10^6	2.97×10^6	$> 5 \times 10^6$
VideoPinball$_{10}$	$> 5 \times 10^6$	1.70×10^6	1.05×10^6	$> 5 \times 10^6$
WizardOfWor$_{20}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
BigFish$^{E0}_{10}$	$> 5 \times 10^6$	4.60×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$
BigFish$^{E1}_{10}$	3.20×10^5	3.00×10^5	1.60×10^5	$> 5 \times 10^6$
BigFish$^{E2}_{10}$	1.30×10^6	5.70×10^5	4.43×10^6	$> 5 \times 10^6$
BigFish$^{H0}_{10}$	$> 5 \times 10^6$	4.20×10^5	1.48×10^6	$> 5 \times 10^6$
Chaser$^{E0}_{20}$	1.90×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Chaser$^{E1}_{20}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Chaser$^{E2}_{20}$	1.70×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Chaser$^{H0}_{20}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Climber$^{E0}_{10}$	7.00×10^4	4.70×10^5	1.60×10^5	$> 5 \times 10^6$
Climber$^{E1}_{10}$	8.00×10^4	$> 5 \times 10^6$	9.00×10^4	$> 5 \times 10^6$
Climber$^{E2}_{10}$	4.40×10^5	$> 5 \times 10^6$	8.40×10^5	4.28×10^7
Climber$^{H0}_{10}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
CoinRun$^{E0}_{10}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
CoinRun$^{E1}_{10}$	1.18×10^6	$> 5 \times 10^6$	9.60×10^5	$> 5 \times 10^6$

COINRUN$_{10}^{E2}$	9.90×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
COINRUN$_{10}^{H0}$	1.00×10^4	2.00×10^4	1.00×10^4	$> 5 \times 10^6$
DODGEBALL$_{10}^{E0}$	3.50×10^5	1.90×10^5	1.79×10^6	$> 5 \times 10^6$
DODGEBALL$_{10}^{E1}$	6.20×10^5	1.00×10^6	1.70×10^6	$> 5 \times 10^6$
DODGEBALL$_{10}^{E2}$	2.60×10^5	1.54×10^6	2.50×10^5	$> 5 \times 10^6$
DODGEBALL$_{10}^{H0}$	6.70×10^5	2.59×10^6	2.13×10^6	$> 5 \times 10^6$
FRUITBOT$_{40}^{E0}$	$> 5 \times 10^6$	4.80×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$
FRUITBOT$_{40}^{E1}$	1.52×10^6	3.77×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
FRUITBOT$_{40}^{E2}$	6.60×10^5	4.00×10^4	2.50×10^5	$> 5 \times 10^6$
FRUITBOT$_{40}^{H0}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
HEIST$_{10}^{E1}$	5.70×10^5	1.65×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
JUMPER$_{10}^{H0}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
JUMPER$_{20}^{E0}$	1.00×10^4	$> 5 \times 10^6$	1.00×10^4	382
JUMPER$_{20}^{E1}$	1.00×10^4	$> 5 \times 10^6$	1.00×10^4	$> 5 \times 10^6$
JUMPER$_{20}^{E2}$	1.90×10^5	$> 5 \times 10^6$	4.00×10^4	$> 5 \times 10^6$
JUMPER$_{20}^{EX}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
LEAPER$_{20}^{E1}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
LEAPER$_{20}^{E2}$	2.20×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
LEAPER$_{20}^{H0}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
LEAPER$_{20}^{EX}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
MAZE$_{30}^{E0}$	1.00×10^4	$> 5 \times 10^6$	1.00×10^4	1.78×10^3
MAZE$_{30}^{E1}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
MAZE$_{30}^{E2}$	2.00×10^4	$> 5 \times 10^6$	1.00×10^4	2.35×10^4
MAZE$_{30}^{H0}$	2.83×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
MAZE$_{100}^{EX}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
MINER$_{10}^{E0}$	2.90×10^5	$> 5 \times 10^6$	1.10×10^5	$> 5 \times 10^6$
MINER$_{10}^{E1}$	1.30×10^5	3.40×10^5	2.90×10^5	$> 5 \times 10^6$
MINER$_{10}^{E2}$	4.00×10^4	2.87×10^6	1.00×10^4	$> 5 \times 10^6$
MINER$_{10}^{H0}$	4.00×10^5	5.30×10^5	2.10×10^5	$> 5 \times 10^6$
NINJA$_{10}^{E0}$	5.50×10^5	3.24×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
NINJA$_{10}^{E1}$	2.09×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
NINJA$_{10}^{E2}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
NINJA$_{10}^{H0}$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
PLUNDER$_{10}^{E0}$	3.00×10^4	3.00×10^4	1.00×10^4	$> 5 \times 10^6$
PLUNDER$_{10}^{E1}$	1.00×10^4	3.00×10^4	1.00×10^4	$> 5 \times 10^6$
PLUNDER$_{10}^{E2}$	1.30×10^5	3.70×10^5	1.01×10^6	$> 5 \times 10^6$
PLUNDER$_{10}^{H0}$	1.00×10^4	7.00×10^4	1.00×10^4	$> 5 \times 10^6$
STARPILOT$_{10}^{E0}$	2.43×10^6	6.70×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$
STARPILOT$_{10}^{E1}$	4.30×10^5	8.30×10^5	1.92×10^6	$> 5 \times 10^6$
STARPILOT$_{10}^{E2}$	8.30×10^5	1.66×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
STARPILOT$_{10}^{H0}$	$> 5 \times 10^6$	2.70×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$
EMPTY-5X5	1.40×10^5	2.40×10^5	3.30×10^5	$> 5 \times 10^6$
EMPTY-6X6	1.70×10^5	2.70×10^5	6.00×10^4	$> 5 \times 10^6$
EMPTY-8X8	2.50×10^5	3.20×10^5	3.00×10^4	$> 5 \times 10^6$
EMPTY-16X16	8.90×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
DOORKEY-5X5	4.40×10^5	5.20×10^5	8.90×10^5	$> 5 \times 10^6$
DOORKEY-6X6	1.60×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
DOORKEY-8X8	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
DOORKEY-16X16	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
MULTIROOM-N2-S4	3.50×10^5	6.00×10^5	2.90×10^5	$> 5 \times 10^6$
MULTIROOM-N4-S5	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
MULTIROOM-N6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
KEYCORRIDORS3R1	1.17×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$

KeyCorridorS3R2	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
KeyCorridorS3R3	3.38×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
KeyCorridorS4R3	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
Unlock	1.10×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
UnlockPickup	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
BlockedUnlockPickup	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
ObstructedMaze-1Dl	3.62×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
ObstructedMaze-1Dlh	4.42×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
ObstructedMaze-1Dlhb	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
FourRooms	2.22×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
LavaCrossingS9N1	5.30×10^5	4.70×10^5	1.10×10^5	$> 5 \times 10^6$
LavaCrossingS9N2	4.50×10^5	5.60×10^5	3.20×10^5	$> 5 \times 10^6$
LavaCrossingS9N3	5.10×10^5	7.90×10^5	1.61×10^6	$> 5 \times 10^6$
LavaCrossingS11N5	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
SimpleCrossingS9N1	4.60×10^5	7.50×10^5	$> 5 \times 10^6$	$> 5 \times 10^6$
SimpleCrossingS9N2	3.40×10^5	$> 5 \times 10^6$	3.60×10^6	$> 5 \times 10^6$
SimpleCrossingS9N3	3.50×10^5	3.80×10^5	4.70×10^5	$> 5 \times 10^6$
SimpleCrossingS11N5	1.78×10^6	$> 5 \times 10^6$	$> 5 \times 10^6$	$> 5 \times 10^6$
LavaGapS5	1.00×10^5	2.20×10^5	4.00×10^4	$> 5 \times 10^6$
LavaGapS6	9.20×10^5	1.48×10^6	6.90×10^5	$> 5 \times 10^6$
LavaGapS7	1.18×10^6	1.41×10^6	1.58×10^6	$> 5 \times 10^6$

F.4 Table of returns

This table lists the optimal returns in each sticky-action MDP as well as the highest returns achieved by PPO, DQN, and SQIRL.

		Returns		
MDP	Optimal policy	PPO	DQN	SQIRL
Alien$_{10}$	158.13	158.5	158.5	153.2
Amidar$_{20}$	76.49	64.76	75.29	63.36
Assault$_{10}$	105.0	105.0	105.0	105.0
Asterix$_{10}$	327.53	336.5	343.5	340.5
Asteroids$_{10}$	170.81	133.3	127.3	126.9
Atlantis$_{10}$	187.5	192.0	190.0	194.0
Atlantis$_{20}$	740.91	754.0	723.0	749.0
Atlantis$_{30}$	1,829.52	1,238.0	995.0	1,124.0
Atlantis$_{40}$	2,620.35	1,794.0	1,186.0	1,686.0
Atlantis$_{50}$	4,856.81	3,683.0	1,865.0	3,339.0
Atlantis$_{70}$	7,932.88	6,108.0	3,347.0	5,356.0
BankHeist$_{10}$	26.15	26.3	27.1	26.2
BattleZone$_{10}$	1,497.07	1,620.0	1,520.0	1,500.0
BeamRider$_{20}$	129.23	126.72	129.8	122.32
Bowling$_{30}$	8.8	8.81	5.77	7.7
Breakout$_{10}$	1.17	1.41	1.21	1.21
Breakout$_{20}$	1.93	1.95	1.99	1.98
Breakout$_{30}$	2.5	2.59	2.27	2.57
Breakout$_{40}$	2.61	2.7	2.21	2.53
Breakout$_{50}$	2.69	2.69	2.12	2.58
Breakout$_{70}$	2.9	2.95	2.28	2.59
Breakout$_{100}$	3.08	3.12	1.9	2.6
Breakout$_{200}$	3.08	2.83	0.73	2.56
Centipede$_{10}$	1,321.17	900.0	1,213.24	1,244.89
ChopperCommand$_{10}$	553.42	469.0	560.0	516.0
CrazyClimber$_{20}$	324.9	334.0	326.0	334.0
CrazyClimber$_{30}$	698.07	701.0	705.0	709.0
DemonAttack$_{10}$	37.07	37.1	38.3	34.6

Enduro_{10}	4.27	4.3	4.37	3.95
FishingDerby_{10}	7.5	7.58	7.5	5.95
Freeway_{10}	1.0	1.0	1.0	1.0
Freeway_{20}	2.0	2.0	2.0	2.0
Freeway_{30}	3.75	3.77	3.77	3.76
Freeway_{40}	4.75	4.76	4.79	4.69
Freeway_{50}	5.75	5.81	5.76	5.77
Freeway_{70}	8.49	8.49	8.5	7.69
Freeway_{100}	11.83	10.84	11.41	10.43
Freeway_{200}	23.84	21.53	21.99	20.66
Frostbite_{10}	66.72	67.9	67.5	68.0
Gopher_{30}	18.75	19.4	19.0	19.2
Gopher_{40}	112.93	83.8	72.0	73.2
Hero_{10}	74.71	75.0	75.0	75.0
IceHockey_{10}	1.0	1.0	1.0	0.86
Kangaroo_{20}	186.32	172.0	182.0	164.0
Kangaroo_{30}	444.7	200.0	370.0	211.0
$\text{MontezumaRevenge}_{15}$	22.53	0.0	0.0	0.0
MsPacman_{20}	460.6	282.9	454.0	408.1
NameThisGame_{20}	94.02	96.2	96.2	95.0
Phoenix_{10}	179.89	181.0	182.6	178.4
Pong_{20}	-1.01	-1.0	-1.0	-1.0
Pong_{30}	-1.61	-1.6	-1.53	-1.82
Pong_{40}	-1.11	-1.19	-1.26	-2.09
Pong_{50}	-1.36	-1.74	-1.53	-3.02
Pong_{70}	-1.48	-3.17	-2.45	-5.34
Pong_{100}	-1.41	-5.62	-3.53	-8.79
PrivateEye_{10}	98.44	99.0	100.0	100.0
Qbert_{10}	350.0	356.25	352.25	354.5
Qbert_{20}	579.71	587.0	555.5	541.25
RoadRunner_{10}	474.71	520.0	494.0	489.0
Seaquest_{10}	20.0	20.0	20.0	20.0
Skiing_{10}	$-8,011.57$	$-9,013.0$	$-8,425.4$	$-8,380.3$
$\text{SpaceInvaders}_{10}$	33.91	34.0	35.2	34.9
Tennis_{10}	0.8	0.0	0.8	0.0
TimePilot_{10}	131.25	136.0	136.0	143.0
Tutankham_{10}	16.51	16.89	16.93	16.54
VideoPinball_{10}	1,744.95	1,388.09	1,816.01	1,786.05
WizardOfWor_{20}	260.35	100.0	100.0	100.0
BigFish_{10}^{E0}	3.53	2.27	3.65	2.09
BigFish_{10}^{E1}	2.97	3.0	2.98	2.99
BigFish_{10}^{E2}	6.86	6.87	6.93	6.89
BigFish_{10}^{H0}	2.59	1.81	2.61	2.62
Chaser_{20}^{E0}	0.88	0.88	0.4196	0.8108
Chaser_{20}^{E1}	0.88	0.84	0.8	0.8696
Chaser_{20}^{E2}	0.88	0.8744	0.5428	0.846
Chaser_{20}^{H0}	0.88	0.84	0.838	0.84
Climber_{10}^{E0}	1.93	1.94	1.93	1.97
Climber_{10}^{E1}	1.75	1.84	1.58	1.79
Climber_{10}^{E2}	10.92	11.0	10.56	11.0
Climber_{10}^{H0}	1.22	1.0	1.0	1.0
CoinRun_{10}^{E0}	8.82	8.8	5.4	7.9
CoinRun_{10}^{E1}	8.36	8.4	8.0	8.6
CoinRun_{10}^{E2}	6.94	7.1	5.7	4.6
CoinRun_{10}^{H0}	10.0	10.0	10.0	10.0
$\text{Dodgeball}_{10}^{E0}$	7.81	8.06	8.72	7.98

Dodgeball$^{E1}_{10}$	5.3	5.3	5.38	5.3
Dodgeball$^{E2}_{10}$	5.71	5.88	5.82	6.08
Dodgeball$^{H0}_{10}$	4.13	4.28	4.18	4.2
Fruitbot$^{E0}_{40}$	1.99	1.9	2.45	1.48
Fruitbot$^{E1}_{40}$	3.6	3.65	3.7	1.0
Fruitbot$^{E2}_{40}$	0.89	0.91	0.91	0.93
Fruitbot$^{H0}_{40}$	1.85	0.0	0.08	0.09
Heist$^{E1}_{10}$	9.38	9.5	9.7	0.7
Jumper$^{H0}_{10}$	1.33	0.0	0.0	0.0
Jumper$^{E0}_{20}$	10.0	10.0	8.5	10.0
Jumper$^{E1}_{20}$	10.0	10.0	5.2	10.0
Jumper$^{E2}_{20}$	10.0	10.0	8.0	10.0
Jumper$^{EX}_{20}$	2.77	0.0	0.0	0.0
Leaper$^{E1}_{20}$	4.92	0.0	0.0	0.0
Leaper$^{E2}_{20}$	9.92	10.0	9.9	8.5
Leaper$^{H0}_{20}$	6.42	0.0	0.2	0.0
Leaper$^{EX}_{20}$	5.7	0.0	0.0	0.0
Maze$^{E0}_{30}$	10.0	10.0	0.0	10.0
Maze$^{E1}_{30}$	7.42	0.0	0.0	0.0
Maze$^{E2}_{30}$	10.0	10.0	9.9	10.0
Maze$^{H0}_{30}$	9.99	10.0	7.8	0.0
Maze$^{EX}_{100}$	9.76	0.0	0.0	0.0
Miner$^{E0}_{10}$	0.91	0.93	0.01	0.96
Miner$^{E1}_{10}$	1.63	1.72	1.73	1.66
Miner$^{E2}_{10}$	1.0	1.0	1.0	1.0
Miner$^{H0}_{10}$	2.97	2.97	2.98	2.97
Ninja$^{E0}_{10}$	6.53	6.8	7.0	5.7
Ninja$^{E1}_{10}$	6.08	6.1	5.6	0.0
Ninja$^{E2}_{10}$	2.0	0.0	0.0	0.0
Ninja$^{H0}_{10}$	2.22	0.0	0.0	0.0
Plunder$^{E0}_{10}$	1.0	1.0	1.0	1.0
Plunder$^{E1}_{10}$	1.0	1.0	1.0	1.0
Plunder$^{E2}_{10}$	0.56	0.62	0.59	0.6
Plunder$^{H0}_{10}$	1.0	1.0	1.0	1.0
Starpilot$^{E0}_{10}$	7.02	7.07	7.16	6.46
Starpilot$^{E1}_{10}$	4.33	4.37	4.36	4.39
Starpilot$^{E2}_{10}$	3.58	3.62	3.66	3.52
Starpilot$^{H0}_{10}$	3.38	3.33	3.42	2.83
Empty-5x5	1.0	1.0	1.0	1.0
Empty-6x6	1.0	1.0	1.0	1.0
Empty-8x8	1.0	1.0	1.0	1.0
Empty-16x16	1.0	1.0	0.0	0.0
DoorKey-5x5	1.0	1.0	1.0	1.0
DoorKey-6x6	1.0	1.0	0.0	0.0
DoorKey-8x8	1.0	0.0	0.0	0.0
DoorKey-16x16	1.0	0.0	0.0	0.0
MultiRoom-N2-S4	1.0	1.0	1.0	1.0
MultiRoom-N4-S5	1.0	0.0	0.0	0.0
MultiRoom-N6	1.0	0.0	0.0	0.0
KeyCorridorS3R1	1.0	1.0	0.0	0.0
KeyCorridorS3R2	1.0	0.0	0.0	0.0
KeyCorridorS3R3	1.0	1.0	0.0	0.0
KeyCorridorS4R3	1.0	0.0	0.0	0.0
Unlock	1.0	1.0	0.0	0.0

UnlockPickup	1.0	0.0	0.0	0.0
BlockedUnlockPickup	1.0	0.0	0.0	0.0
ObstructedMaze-1Dl	1.0	1.0	0.0	0.0
ObstructedMaze-1Dlh	1.0	1.0	0.0	0.0
ObstructedMaze-1Dlhb	1.0	0.0	0.0	0.0
FourRooms	1.0	1.0	0.0	0.0
LavaCrossingS9N1	1.0	1.0	1.0	1.0
LavaCrossingS9N2	0.93	1.0	1.0	1.0
LavaCrossingS9N3	0.7	1.0	1.0	1.0
LavaCrossingS11N5	0.73	0.0	0.0	0.0
SimpleCrossingS9N1	1.0	1.0	1.0	0.0
SimpleCrossingS9N2	1.0	1.0	0.0	1.0
SimpleCrossingS9N3	1.0	1.0	1.0	1.0
SimpleCrossingS11N5	1.0	1.0	0.0	0.0
LavaGapS5	1.0	1.0	1.0	1.0
LavaGapS6	1.0	1.0	1.0	1.0
LavaGapS7	1.0	1.0	1.0	1.0