Greedification Operators for Policy Optimization: Investigating Forward and Reverse KL Divergences

by

Alan Chan

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Statistical Machine Learning

Department of Computing Science

University of Alberta

Abstract

Policy gradient methods typically estimate both explicit policy and value functions. The long-extant view of policy gradient methods as approximate policy iteration—alternating between policy evaluation and policy improvement by greedification—is a helpful framework to elucidate algorithmic choices. Effective policy evaluation under function approximation is being actively investigated; approximate greedification, however, has yet to be systematically explored. In this work, we highlight and investigate the difference between the forward and reverse KL divergences when used for policy improvement. We show that the reverse KL has stronger theoretical guarantees for policy improvement, but that the forward KL can also induce improvement under additional assumptions. Finally, on both small-scale and large-scale experiments, we empirically analyze the behaviour and practical performance of these variants. We observe few consistent differences between the reverse and forward KLs on discrete-action spaces, but relatively more substantial stability and convergence differences emerge on continuous-action spaces.

Preface

This thesis was compiled and extended from a submission to NeurIPS 2020, in collaboration with Sungsu Lim and Martha White. Sungsu contributed description of soft value functions, analysis of the heatmaps, the heatmaps themselves, and the continuous-action control experiments. Martha contributed parts of the introduction, the discussion on the weighting of states, some general suggestions throughout, and a modification to one or the proofs. Hugo Luis Silva also helped in checking the proofs. All other contributions are my own.

To all who have come before
And to all who will come after.

How should we like it were stars to burn
With a passion for us we could not return?
If equal affection cannot be,
Let the more loving one be me.

– "The More Loving One", W.H. Auden

Why?

– Michael Peters

Acknowledgements

From the web called "I", we often cannot trace the origins of all the strands that comprise our complexity. Many who have forged our identities are forgotten, their imprints the only evidence that we did not face all of our demons alone.

Here are lines that I can trace. In the 6th grade, Dave Fairfield taught me to pursue my curiosity, regardless of the disinterest of anyone around me. Throughout high school, Michael Peters taught me to ask why, even when doing so was embarrassing or, at first glance, nonsensical. For 7 years, competitive debate, although at times frustrating, taught me confidence and the negotiability of our perceptions of truth. At the University of Alberta, the Landing and the Sexual Assault Centre enhanced my sensitivities to both injustice and empathy.

In the past year, I have been borne on a whirlwind through research in reinforcement learning. I stand endlessly grateful for the welcoming reception I have received from this community, despite my relative newcomer status. Of this community, those who most stand out are Martha White, Daniel Graves, Rupam Mahmood, Kris de Asis, Taher Jafferjee, Nguyen Minh Nhat, Sungsu Lim, Vincent Liu, Muhammad Zaheer, and Wesley Chung.

Finally, the most radiant lines are those who still inhabit my slice of eternity, with whom I have had the fortune of stability in the ceaseless agitations of time: in no particular order, Tony Basu, Jerry Chen, Kevin Wong, Tara Jacklin, YiJi Zhao, Kevin Wang, Iara Santelices, Davis Lazowski, XingJian Li, Iryna Stryapunina, Melanie Liu, Shay Lewis, Frank Huang, Kayvon Miller, Srosh Hassan, Chris Chan, Thomas Tetz, Mario Mach, Cecilia Wu, Gloria Ng, and all those other, time-worn threads that I still hold in my memory.

Contents

1 Introduction **1**

2 Background **7**
2.1 Reinforcement Learning 7
2.2 KL Divergences 12
2.3 Approximate Greedification 14
2.3.1 Defining a Target Policy 14
2.3.2 Approximate Greedification with the KLs 17
2.3.3 The Weighting over States 19

3 Theoretical Results **21**
3.1 Reverse KL 21
3.2 Forward KL 25
3.3 Summary 34

4 Microworld Experiments **36**
4.1 Common Implementation Details 37
4.2 Continuous Action Results in the Bimodal Bandit 38
4.2.1 Loss Surface 38
4.2.2 Behaviour 40
4.3 Switch-Stay 42
4.3.1 Behaviour 43
4.4 Discrete-Action Results 46
4.5 The Impact of Stochasticity 50
4.6 Summary 52

5 Benchmark Results **55**
5.1 Implementation Details 55
5.1.1 Hyperparameters 55
5.1.2 Updating 56
5.1.3 Architecture 57
5.2 Continuous-Action Results 59
5.3 Discrete-Action Results 60
5.4 Summary 64

6 Conclusion **66**
6.1 Summary 66
6.2 Limitations 67
6.3 Future Work 69

References **71**

List of Figures

3.1 The minimal C in Proposition 3 as a function of τ, plotted against the KL divergence between $\mathcal{B}Q^{\pi_{\text{old}}}$ and π_{old} at s_1 of the MDP in Proposition 2. 33

4.1 Switch-Stay. 37
4.2 Continuous-action Bimodal Bandit. 37
4.3 KL loss over mean and standard deviation across temperature. Note that the actual action taken applies tanh to the samples of the resulting distribution (i.e., the optimal mean is at $\tanh^{-1}(0.5) \approx 0.55$). FKL loss has been upper-bounded for better visualization of minima. Arrows indicate the global minimum. 39
4.4 Each plot tracks the mean over 1000 gradient steps of each of 1000 iterates. Each iterate is represented as a translucent, coloured dot with alpha value 0.01. Temperature is varied on the x-major-axis and initial standard deviation σ_0 is varied on the y-major axis. Colour-coded by σ_0. Learning rate is 0.01. . 41
4.5 Each subplot plots the final value functions on the continuous version of switch-stay after 500 gradient steps with $\gamma = 0.9$ for 1000 iterates. The top-right corner of the polytope in each subplot is the optimal value function and the bottom-left corner is the pessimal value function. Each iterate is represented by a translucent dot with alpha value 0.01. Using RMSprop. Temperature is varied on the x-major-axis. 44
4.6 See Figure 4.5. 45
4.7 Each subplot plots the final mean (x-axis) and standard deviation (y-axis) on the continuous version of switch-stay after 500 gradient steps with $\gamma = 0.9$ for 1000 iterates. Each iterate is represented by a translucent dot with alpha value 0.1. Using RMSprop with learning rate 0.01. Temperature is varied on the x-major-axis. Every action ≤ 0 is treated as "stay" and every action > 0 is treated as "switch". The blue dotted line in subplots corresponds to the line $\mu = 0$. 46
4.8 See Figure 4.7. 47

4.9 Heatmap for the KLs on the discrete bandit. In a given subplot, the x-axis is the logit for the optimal arm and the y-axis is the logit for the suboptimal arm. 47
4.10 Each subplot tracks the learned probability of each arm for 1000 iterates over 1000 gradient steps. The learning rate is set to be 0.005. 48
4.11 Final value functions on discrete version of switch-stay after 500 gradient steps, $\gamma = 0.9$. Using RMSprop for learning rates $\in \{0.005, 0.01\}$. 49
4.12 Continuous bandit with 10 sample points, learning rate = 0.005, with RMSprop. 51
4.13 Continuous bandit with 500 sample points, learning rate = 0.005, with RMSprop. 52
4.14 Switch-stay with 10 sample points, learning rate = 0.01, with RMSprop. 53
4.15 Switch-stay with 500 sample points, learning rate = 0.01, with RMSprop. 53

5.1 Continuous-action environments. Shown are the best hyperparameters for each algorithm, which are selected by largest area under the last half of the learning curve. The colours go from hot (red, temperature = 1) to cool (yellow, temperature = 0). 60
5.2 OpenAI Gym discrete-action environments. Plot settings are identical to Figure 5.1. 61
5.3 MinAtar discrete-action environments. Plot settings are identical to those in Figure 5.1. 62
5.4 MinAtar discrete-action environments continued. Plot settings are identical to those in Figure 5.1. 63

Chapter 1

Introduction

A central issue in artificial intelligence (AI) research is the development of agents that can learn or improve themselves based upon interaction with the world. Just as humans can learn without explicit supervision, we desire artificial agents that can discover knowledge about the world through independent exploration and inquiry. After all, if we want AI systems to help us solve problems that we might not be able to solve, how much help would those agents be if they needed explicit human supervision all the time?

Much recent attention has focused on the design of agents that can learn without explicit supervision, perhaps best typified by the field of *reinforcement learning* (RL). In reinforcement learning, researchers try to develop agents that can learn to improve their own performance over time, according to a quantity called a *reward function* that is (usually) specified.The reward function encapsulates a desired *goal* of the designer: for example, a reward function may return 1 if an agent reaches the end of an obstacle course, and return 0 otherwise. Maximizing the rewards received is taken to be a proxy for achieving the goal. Such a view is not uncontroversial, but is useful in permitting the application of mathematical tools to the RL problem.

A RL agent observes the world, taking actions that subsequently change the world and (hopefully) bring about the desired goal by maximizing reward. A compact way of summarizing what actions an RL agent might take is known as a *policy*, which provides possible actions for each possible state of the world. A policy may itself be represented by a collection of *parameters* (i.e., some

numbers, like the parameters of a neural network), which collectively determine what actions an agent may take in each state. The aim of RL is therefore to develop agents that improve their own policies so as to achieve goals.

In service of this aim, one area of inquiry in reinforcement learning is *policy optimization*, where analogies are drawn between policy improvement and the mathematical field of optimization. In optimization, one is typically interested in maximizing (or minimizing) a certain quantity called an *objective function*. Much work in optimization can trace its roots to the 20th century,[1] where the demands of war and command economies necessitated the development of mathematical techniques that told you how to get the best "bang for the buck", so to speak. The fit between RL and optimization seems natural given the proxy definition of goals as rewards.

More technically, policy optimization involves explicit, parameterized policies that are modified to maximize an objective function, commonly taken to be a sum of rewards. A policy that maximizes the objective function is known as an *optimal policy*. The most popular methods are policy gradient (PG) methods, which iteratively update the policy parameters using the gradient of either the discounted return or the average reward, given by the policy gradient theorem (Sutton et al., 2000). Although not a recent invention, with early work introducing actor-critic (Konda and Tsitsiklis, 2000; Sutton, 1984), PG methods have recently seen a surge of renewed interest given their ease of application in high-dimensional, continuous action spaces when combined with neural networks (Mnih et al., 2016; Schulman et al., 2016; Wang et al., 2017). Recent developments include learning deterministic polices (Lillicrap et al., 2016; Silver et al., 2014), trust-regions (Schulman et al., 2015; Schulman et al., 2017b), continuous-action extensions of Q-learning (Haarnoja et al., 2017; Lim et al., 2018; Ryu et al., 2020), probabilistic approaches (Abdolmaleki et al., 2018; Fellows et al., 2019), and entropy regularization (Haarnoja et al., 2017; Haarnoja et al., 2018; Levine, 2018; Rawlik et al., 2013; Ziebart, 2010; Ziebart et al., 2008).

[1] Even earlier origins exist as well, as typified in the works of Fermat, Lagrange, Newton, Cauchy, and Gauss.

Theoretical and empirical work into PG is growing. Theoretically, CPI (Kakade and Langford, 2002) is an early example of theoretical insights into obtaining guaranteed policy improvement with PG . More recently, Agarwal et al. (2019a) derive finite-sample and approximation bounds for a variety of PG methods; Mei et al. (2020) show that, with a softmax policy parameterization, entropy-regularized PG converges faster than unregularized PG; Liu et al. (2019), Neu et al. (2017), and Shani et al. (2019) reformulate TRPO as mirror descent and prove convergence; Ahmed et al. (2019) show that entropy regularization may lead to smoother optimization landscapes; and Bhandari and Russo (2019)[2] show global optimality of local minima under certain restrictions on the MDP.

Empirically, recent investigations have both unveiled some of the shortcomings and increased our understanding of current PG methods. Many implementations of PG in practice use an incorrect, simplified gradient (Imani et al., 2018; Nota and Thomas, 2020; Thomas, 2014). Somewhat unfortunately, the apparent superiority of some more modern PG methods seem to be due to code-level optimizations rather than algorithmic advances (Engstrom et al., 2019; Ilyas et al., 2020). In the context of entropy-regularized PG, (Ahmed et al., 2019) suggest that entropy-regularized PG methods have smoother optimization landscapes than non-regularized methods.

One way to provide clarity in the analysis of PG methods is to revisit the insight that many PG methods can be seen as approximate policy iteration (API). To find an optimal policy, API methods (Bertsekas, 2011; Scherrer, 2014) interleave approximate policy evaluation—understanding how a policy is currently performing with a value function—and (approximate) policy improvement–making the current policy better based on policy-evaluation information. The policy improvement step is sometimes called the *greedification* step, which refers to the fact that in exact PI, the subsequent policy is set to be the greedy action at each state (i.e., the action that maximizes the current

[2]We note that the results in Bhandari and Russo (2019) bear a striking resemblance to earlier results in Scherrer and Geist (2014). In particular, both works rely on convex policy classes and a closure of the policy class under greedification.

action-value function).

The connection between PG and *Approximate* PI arises because the efficient implementation of PG methods often requires the estimation of a value function. One could estimate the PG purely through Monte Carlo samples, just as REINFORCE (Williams, 1992) does, but it is less wasteful to use data to inform future estimates of the gradient. In particular, with environment data, one is able to estimate the action-value function with temporal-difference methods (Sutton and Barto, 2018). Numerous papers have linked PG methods to policy iteration (Bhandari and Russo, 2019; Kakade and Langford, 2002; Perkins and Pendrith, 2002; Perkins and Precup, 2003; Scherrer and Geist, 2014; Sutton et al., 2000; Wagner, 2011; Wagner, 2013), including recent work connecting maximum-entropy PG and value-based methods (Nachum et al., 2017b; Nachum et al., 2019; O'Donoghue et al., 2017; Schulman et al., 2017a).

Viewing one gradient step (PG) as a policy greedification step (API) suggests that one way to understand PG methods is to understand their greedification steps; in particular, after greedifying, what is the quality of the resulting policy? For tabular policies, policy greedification is straightforward: at each state, we set the policy to place unit mass on the greedy action (or mass spread arbitrarily around the greedy actions), with zero mass on non-greedy actions. If a new policy is greedy with respect to the action-value function of an old policy, the classical policy improvement theorem (Sutton and Barto, 2018) guarantees that the new policy is at least as good as the old policy. For parameterized policies (e.g., neural-network policies), however, exact greedification in ecah state is rarely possible as not all policies will be representable by a given function approximator class.

To define an approximate greedification scheme, one can minimize the KL divergence of the current policy to a Boltzmann distribution over the action values (Wagner, 2011). The use of a Boltzmann distribution is common in pseudo-likelihood methods (Kober and Peters, 2009; Levine, 2018; Neumann et al., 2011), ensuring that one has a target distribution based on the action-values. The KL divergence is a convenient choice because stochastically estimating the objective only requires the ability to sample from the distributions and evaluate

the values of the distributions at single points. It is unclear, however, whether to use the reverse or the forward KL divergence. That is, should the policy π be the first argument of the KL divergence, or should it be the Boltzmann distribution over the action values? For example, Neumann et al. (2011) argues in favour of the reverse KL divergence as such a resulting policy would be cost-averse, while Norouzi et al. (2016) uses the forward KL divergence to induce a policy that is more exploratory (i.e., has a more diverse state visitation distribution).

The policy update for many PG methods can be seen as optimizing a reverse KL, though some work has employed the forward KL (Agarwal et al., 2019b; Nachum et al., 2017a; Norouzi et al., 2016; Vieillard et al., 2020), including implicitly some of the work in classification for RL (Farahmand et al., 2015; Lagoudakis and Parr, 2003; Lazaric et al., 2010). Despite the fact that both have been used, there is no comprehensive investigation into the differences between these two choices for approximate greedification; the typical default is the reverse KL. The reverse KL without entropy regularization corresponds to a standard actor-critic update and is easy to compute. More recently, it was shown that the reverse KL guarantees policy improvement when the KL can be minimized separately for each state (Haarnoja et al., 2018, p. 4). At the same time, reverse KL objectives have known problems, primarily that they are non-convex, even in an ideal case with a linear Boltzmann policy. For contextual bandits, Chen et al. (2019) showed improved performance when using a surrogate, forward KL objective for the smoothed risk. Some works also use the forward KL ostensibly to prevent mode collapse, given that the forward KL is mode-covering (Agarwal et al., 2019b; Mei et al., 2019). The forward KL divergence is also used in supervised learning, in the form of the cross-entropy loss.

No work thus far has explored the difference between the KLs in the framework of what policy improvement guarantees can be provided, and whether the conditions of any such policy improvement results hold in experimental settings.

The goal of this thesis is to investigate how one should perform

the greedification step for parameterized policies. In particular, for a given action-value estimate, we investigate the difference between using a forward or reverse KL divergence, primarily in the context of entropy regularisation. We ask, given that we optimize a policy to reduce either the forward or the reverse KL divergence to a Boltzmann distribution over the action values, what is the quality of the resulting policy?

We provide some clarity on this question with the following contributions.

1. We highlight four choices for greedication: forward or reverse KL to a Boltzmann distribution on the action-values, with or without entropy regularization.

2. We show that the policy improvement result for the reverse KL extends to certain function-approximation settings.

3. We construct a counterexample where optimizing for the forward KL can fail to induce policy improvement.

4. Nevertheless, we show that under some additional conditions on the temperature, KL reduction, and entropy, optimizing for the forward KL can induce policy improvement.

5. On small-scale experiments, we find that the reverse KL can converge faster, but sometimes to worse solutions, than the forward KL, particularly under continuous actions. However, depending on the degree of entropy regularization, the forward KL can provide worse solutions than the reverse KL.

6. On large-scale benchmarks, we found no consistent superiority of either KL divergence over the other, but we note intriguing trends influenced by the forward KL, entropy regularization, and the function approximation architecture.

Chapter 2

Background

In this section, we introduce the necessary background on reinforcement learning, KL divergences, and approximate greedification with KL divergences. Readers already familiar with the RL problem setting may skip to Section 2.3.

2.1 Reinforcement Learning

In designing a general artificial intelligence, a priority is the ability of any such agent to act to achieve goals in the world. Upon observing some state of the world, we would like the agent to be able to determine the best course of action according to its own knowledge; after all, we would like general artificial intelligences to help us solve our own intractable problems.

One way of formalizing these desiderata is through the reinforcement learning (RL) framework. RL is a collection of both problems and solution methods for addressing the challenge of general artificial intelligence. In RL, a "state of the world" is simply called a state, a "course of action" is called a policy, and "goal" is a numerical quantity called the return. Although the formalization of goal as return–the reward hypothesis (Sutton and Barto, 2018)–might be questionable, and further research into alternatives is desirable, such a formulation will enable us to bring centuries of mathematics to bear upon RL.

We formalize RL as a Markov Decision Process (MDP), characterized by a tuple $(\mathcal{S}, \mathcal{A}, \gamma, r, p)$. $\mathcal{S}$ is the state space; $\mathcal{A}$ is the action space; $\gamma \in [0, 1)$ is the discount factor; $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function; and for every $(s, a) \in \mathcal{S} \times \mathcal{A}$, $p(\cdot \mid s, a)$, called the transition kernel, gives the conditional

transition probabilities over $\mathcal{S}$. We also additionally specify a distribution ρ_0 over $\mathcal{S}$ of *starting states*, from which an agent begins interaction with the environment. Where necessary for our proofs, we will state additional assumptions on $\mathcal{S}$ and $\mathcal{A}$.

A *policy* is a mapping $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$, where $\Delta_{\mathcal{A}}$ is the space of probability distributions over $\mathcal{A}$. At the beginning of interaction, a state s_0 is drawn from ρ_0. At this point, and for each subsequent, discrete time step t, draws an action from its policy: $a_t \sim \pi(\cdot \mid s_t)$. The agent sends the action a_t to the environment, from which it receives the reward signal $r(s_t, a_t)$[1] and observes the next state s_{t+1}.

We will assume that our MDPs are episodic; that is, for every policy π, the Markov chain induced by π, p will almost surely reach an absorbing state s_T in finite time. Once s_T is reached, one considers the episode to have "ended", and restarts the agent by drawing another s_0 from ρ_0. In other words, our RL agents only have a limited amount of time to interact with the environment, after which they are replaced at a starting state.

Informally, the goal of an RL agent is to maximize the *expected return*, usually defined as the expectation of a discounted sum of rewards [2]. To formalize this goal, given a policy π, one may define the *value function* V^π associated to that policy.

$$V^\pi(s) := \mathbb{E}_{\pi,p}\left[\sum_{k=0}^{\infty} \gamma^k r(S_k, A_k) \mid S_0 = s\right].$$

$V^\pi(s)$ tells us, starting from state s and following policy π, what is the average return the agent receives? The expectation above is over the trajectory $(s_0, a_0, s_1, a_1, \cdots)$ induced by π and the transition kernel p. As the MDP is usually clear from the context, we will henceforth suppress p in our notation. We may also define the *action-value function*, which also conditions on a

[1] It is also possible to condition on the future state s_{t+1}, but we do not do so to minimize notational clutter.

[2] In this work, we neglect the average reward formulation of RL. Further information on this alternative may be found in (Puterman, 2014).

selected action a.

$$Q^\pi(s,a) := \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k r(S_k,A_k) \mid S_0 = s, A_0 = a\right]$$
$$= r(s,a) + \gamma\mathbb{E}_{s'\sim p(\cdot|s,a)}[V^\pi(s')] \tag{2.1}$$

For $\gamma \in [0,1)$, it is a consequence of the Banach Fixed-Point theorem that there is precisely one value function V^*, called the *optimal value-function*, such that $V^*(s) \geq V^\pi(s)$ for all π and s. Given V^*, we can define Q^* by Equation (2.1). The policy induced by selecting $a = \arg\max_b Q^*(s,b)$ at every state s is known as the *optimal (deterministic) policy* π^*. There can be more than one optimal policy, and an optimal policy may be stochastic. An optimal policy in general depends on γ,[3] but we suppress dependence on γ for simplicity. It is a standard result[4] that $V^{\pi^*} = V^*$, although this equality does not uniquely define π^* in general. We can thus formalize the RL goal as finding any π^* that induces the optimal value function V^*.

For small, finite $\mathcal{S}$ and $\mathcal{A}$, finding V^* and π^* is tractable through value iteration (VI) or policy iteration (PI) (Sutton and Barto, 2018). In policy iteration for example, one begins with an initial policy π_0 and initial value function estimate v_0. The following procedure is then iterated until the maintained estimates converge, or some other error criterion is attained.

$$\pi_{t+1} := \mathcal{G}(Q_{t+1}) \qquad\qquad Q_{t+1} := Q^{\pi_t}$$

$\mathcal{G}(Q_{t+1})$ is the policy obtained by setting π to be the Dirac delta distribution on $\arg\max_b Q_{t+1}(s,b)$ for every s. We refer to the act of applying $\mathcal{G}$ as *greedifying* a policy. We refer to the right-hand operation of PI as *policy evaluation*, which classically can be performed with dynamic programming. PI is thus the interleaving of policy greedification and policy evaluation.

Unfortunately, with a large or infinite number of states or actions, exact VI and PI become intractable. Exact VI in particular suffers from the following problems: **(1)** applying $\mathcal{G}$ across all states is not feasible if the number of states

[3]See, however, Blackwell optimal policies (Mahadevan, 1996).

[4]For example, one can see this result by examining the Bellman optimality equation for V^*.

is large or infinite; **(2)** even applying $\mathcal{G}$ at a single state may be intractable if $|\mathcal{A}|$ is large or infinite, as an inner maximization problem would have to be solved; **(3)** exact policy evaluation at all state-action pairs (s, a) may not be possible.

We thus turn to approximation, and must address two issues: **(A)** how to represent a policy over many states and **(B)** how to relax the goal of finding π^*.

To address **(A)**, we introduce a set of parameters $\theta \in \mathbb{R}^k$, for some $k \in \mathbb{N}$, to represent our policy. Instead of specifying a mapping for every state $s \in \mathcal{S}$, θ induces a function $\pi_\theta : \mathcal{S} \to \Delta_{\mathcal{A}}$. For example, π_θ could be a neural network that takes the state as input and returns a probability distribution over the actions. Note, however, that such a mapping π_θ is usually not surjective onto $\Delta_{\mathcal{A}}$; in particular, there might not be a θ^* such that $\pi_{\theta^*} = \pi^*$.

Introducing parameters to represent our policy allows us to address **(B)**. We introduce a common objective in policy optimization: the value function V^{π_θ} averaged over a distribution ρ_0 over starting states.

$$\eta(\pi_\theta) := \int_{\mathcal{S}} \rho_0(s) \int_{\mathcal{A}} \pi_\theta(a|s) Q^{\pi_\theta}(s, a)\, da\, ds.$$

Our goal may be stated now as finding the θ that maximize $\eta(\pi_\theta)$. If π_θ is differentiable with respect to θ, we may attempt to apply gradient-based optimization to this problem.

The policy gradient theorem gives us the gradient of $\eta(\pi_\theta)$ (Sutton et al., 2000),

$$\nabla_\theta \eta(\pi_\theta) = \int_{\mathcal{S}} d^{\pi_\theta}(s) \int_{\mathcal{A}} Q^{\pi_\theta}(s, a) \nabla_\theta \pi_\theta(a \mid s)\, da\, ds, \tag{2.2}$$

where

$$d^{\pi_\theta}(s) := \sum_{t=0}^{\infty} \gamma^t \Pr(s_t = s \mid s_0 \sim \rho_0, \pi) \tag{2.3}$$

is the *unnormalized discounted state visitation distribution*. Intuitively, d^{π_θ} provides the probability that π visits a given state s at any time t, discounted by γ^t.

To perform this optimization, a number of options are available. In REINFORCE (Williams, 1992), a sampled return from (s, a) is used as an unbiased estimate of $Q^{\pi_\theta}(s, a)$. While unbiased, REINFORCE is somewhat wasteful of data; it seems possible to use any collected trajectory data to improve an existing estimate of $Q^{\pi_\theta}(s, a)$, rather than start afresh with every trajectory. REINFORCE can also suffer from high variance given its complete reliance on Monte Carlo estimates of the action-value function.

To address these shortcomings, we can try to estimate Q^{π_θ}. Commonly, a biased[5], but lower-variance, choice is to use a learned estimate Q of Q^{π_θ}, obtained through policy evaluation algorithms like SARSA (Sutton and Barto, 2018). In these Actor-Critic algorithms, the actor—the policy—updates with a (biased) estimate of the above gradient, given by this Q—the critic.

In practice, if one insists on performing an update at every timestep t and does not multiply the update by γ^t, one in effect ignores $d^{\pi_\theta}(s)$ and instantiates a biased gradient update (Thomas, 2014). Despite this concern, one tends to exclude the γ^t in practice because of concerns about sample-efficiency; the longer the episode, the smaller the update at time t will be. There has yet to be a systematic empirical study of the impacts of excluding γ^t.

Using the gradient in Equation (2.2) to update the policy while learning an action-value estimate can be interpreted as Approximate Policy Iteration (API). API methods alternate between **(1)** approximate policy evaluation to obtain a new Q and **(2)** approximate greedification to get a policy π that is more greedy with respect to Q. As we show in the next section, the gradient in Equation (2.2) can be recast as the gradient of a KL divergence to a policy peaked at maximal actions under Q; reducing this KL updates the policy to increase its own probabilities of these maximal actions, and so become more greedy with respect to Q. Under this view, we obtain a clear separation between estimating Q and greedifying π. We can be agnostic to the strategy for updating Q—we

[5]We note here that relatively little attention has been paid to the assumption of *compatible features* in Sutton et al. (2000). Given this assumption, which amounts to requiring the the action-value function estimate is linear in the normalized features of the policy, replacing Q^{π_θ} with Q in the policy gradient yields no bias. Further investigation into the importance of this choice is needed.

can even use soft action values (Ziebart, 2010) or Q-learning (Watkins and Dayan, 1992)—and focus on answering: for a given Q, how can we perform an approximate greedification step and which approaches are most effective?

2.2 KL Divergences

Before we continue our discussion about greedification, it will be necessary to introduce some concepts from statistics. Later, it will be useful for us to measure the distance between π and some target policy π_{better}, which is presumably "better" than π. If we can close the distance between π and π_{better}, then we will hopefully end up with a better policy. However, we must first make sense of how to define "distance" between probability distributions.

One option is to appeal to information theory. Let X be an event with probability P. We can derive[6] a notion of the *surprise* of X; how surprised were we that X occurred? The surprise of X is given by

$$-\log P.$$

What if we were interested in how surprised we were on average, across all events? Asked another way, how much information do we get on average? Fixing some probability distribution p of a random variable X, the surprise of the event $X = x$ is

$$-\log \Pr(X = x) = -\log p(x).$$

This definition makes sense: if the event is impossible, I should be infinitely surprised; if the event was certain, I should not be surprised at all; if I observe two independent events, my joint surprise should just be the sum of the individual surprises. The negative logarithm satisfies all of these properties.

The average surprise, or *information content*, is just the surprise averaged across p.

$$\mathbb{E}_p[-\log p].$$

[6] `https://en.wikipedia.org/wiki/Information_content`

The information content is also known as the entropy, and intuitively captures the spread of the probability mass of the policy amongst the actions.

Now, given distributions p, q, we might wonder how much information might we gain from observing an event we think is drawn from q, but really is from p. This quantity is known as the *cross-entropy* between p and q.

$$\mathbb{E}_p[-\log q].$$

Suppose as well that we want to compare the difference in information between (1) knowing that the true distribution is p and (2) believing that the true distribution is q. We can form the following quantity, which will turn out to be the KL divergence.

$$\mathbb{E}_p[-\log q] - \mathbb{E}_p[-\log p].$$

We can now define a pseudo-distance[7] based on these informational considerations. Given two probability distributions p, q on $\mathcal{A}$, the KL divergence between p and q is defined as

$$\mathrm{KL}(p \parallel q) := \int_{\mathcal{A}} p(a) \log \frac{p(a)}{q(a)}\, da,$$

where p is assumed to be absolutely continuous (Billingsley, 2008) with respect to q, to ensure that the KL divergence exists. The KL divergence is zero iff $p = q$ almost everywhere, and is always non-negative.

A post-hoc reason why the KL divergence is nice to consider is that sampling the KL divergence, instead of performing the integral, requires just the ability to sample from p and to calculate p and q. This feature is in contrast to the Wasserstein metric[8] for example, which generally requires solving an infimum.

The KL divergence is not symmetric; for example, $\mathrm{KL}(p \parallel q)$ may be defined while $\mathrm{KL}(q \parallel p)$ may not even exist if q is not absolutely continuous with respect to p. This asymmetry leads to the two possible choices for measuring differences

[7]Note that the KL divergence is not a metric in the mathematical sense, as it does not obey symmetry or the triangle inequality.

[8]Some interesting work has explored the benefits of the Wasserstein metric. (Arjovsky et al., 2017) show that the Wasserstein metric induces a weaker notion of convergence than the KL divergence, allowing for better stability and convergence during GAN training.

between distributions: the reverse KL and the forward KL. Assume that p is a true distribution that we would like to match with our learned distribution q_θ, where q_θ is smooth with respect to $\theta \in \mathbb{R}^k$. The *forward* KL divergence is $\mathrm{KL}(p \parallel q_\theta)$ and the *reverse* KL divergence is $\mathrm{KL}(q_\theta \parallel p)$.

2.3 Approximate Greedification

2.3.1 Defining a Target Policy

We return to the subject of greedification. At the beginning of Section 2.2, we discussed the use of probability distances to improve a policy π. If we had access to some better policy π_{better}, we could "close the distance", using something like the KL divergence, to improve π. But what π_{better} should we use? A natural choice is the action-value function $Q(s, a)$ itself; indeed, one can view the greedification step of exact policy iteration as such a gap-closing. However, $Q(s, \cdot)$ is generally not a distribution over a since it may be negative! One solution to this problem is to apply a transformation to Q whose range is the non-negative real numbers.

Let $\tau > 0$ and let Q be an action-value function estimate. We define the transformed action-value $\mathcal{B}Q_\tau$ by

$$\mathcal{B}Q_\tau(s, a) := \frac{\exp(Q(s, a)\tau^{-1})}{\int_{\mathcal{A}} \exp(Q(s, b)\tau^{-1})\, db}. \tag{2.4}$$

Firstly, note that the definition in Equation (2.4) does not depend upon a particular policy. In other words, we can input any function of the form $f(s, a)$.

The τ in Equation (2.4) corresponds to the division by τ in the argument of the exponential. If Q is a soft action-value, to be defined subsequently, τ also refers to the temperature of the soft action-value. $\mathcal{B}Q_\tau$ provides the optimal soft greedification with respect to Q. To understand why, we turn to soft value functions (Ziebart, 2010). First, we define the *entropy* of a distribution, which captures how "spread out" the distribution is. The higher the entropy, the less the probability mass of $\pi(\cdot \mid s)$ is concentrated in any particular area.

$$\mathcal{H}(\pi(\cdot \mid s)) := -\int_{\mathcal{A}} \pi(a \mid s) \log \pi(a \mid s)\, da.$$

Now, we define the soft value functions; they are essentially just regular value functions where an entropy term is added to the reward.

$$V_\tau^\pi(s) := \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \left[r(S_k, A_k) + \tau \mathcal{H}(\pi(\cdot|S_k)) \right] \mid S_0 = s \right]$$

We can also define the soft action-value function.

$$Q_\tau^\pi(s,a) := r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[V_\tau^\pi(s')]$$

We can also write the state-value function in terms of the action-value function.

$$V_\tau^\pi(s) = \mathbb{E}_\pi[Q_\tau^\pi(s,a) - \tau \log \pi(a \mid s)].$$

Intuitively, the soft value functions penalize determinism in our policies. Using soft value functions instead of (regular) value functions changes the RL problem slightly, as we are no longer interested in just the return, but rather the return plus the determinism penalty.

One might also wonder at why the entropy bonus for the state-action pair (s,a) is not added to the reward $r(s,a)$. Intuitively, since the agent has already taken action a, it is meaningless to incentivize any randomness at s. One other consistency reason is that the definition of the soft action-value function is exactly the same as the definition of the non-soft action value function, except with $V^\pi(s')$ replaced with $V_\tau^\pi(s')$.

If we set $\pi'(\cdot|s) = \mathcal{B}Q_\tau(s,\cdot)$ for all $s \in \mathcal{S}$, then $Q_\tau^{\pi'}(s,a) \geq Q_\tau^\pi(s,a)$ for all (s,a) (Haarnoja et al., 2017, Theorem 4). As τ approaches zero, $Q_\tau^\pi(s,a)$ approaches $Q^\pi(s,a)$, which is a motivation using $\mathcal{B}Q_\tau$ as a target policy for greedification.

Another way to understand this definition, and especially why we divide $Q(s,a)$ by τ in the exponential of $\mathcal{B}Q_\tau$, is through the theory of entropy-regularized MDPs (Geist et al., 2019). One desire for a target policy might be that it is *greedy* in some sense with respect to the action values. In the case of RL with no entropy regularisation, the greedy policy is conventionally the policy that returns the maximum action at each state. The reason is, if Q^π is an action value corresponding to π, and if π' is the greedy policy with

respect to $Q^{\pi'}$, then π' is a superior policy to π according to the classical policy improvement result (Sutton and Barto, 2018).

When we introduce entropy regularisation, a different sense of *greedy* is needed, as we are interested not just in the reward r of the original MDP, but also the entropy of the policy. To understand how to do so, it is helpful to discuss another formulation of the greedy policy. Assume for the moment that the state and action spaces are finite. One definition of the Bellman operator is the following.

$$(\mathcal{T}^\pi v)[s] := \mathbb{E}_\pi[q(s,a)], \tag{2.5}$$

where $q(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[v(s')]$. In other words, we define the Bellman operator state-wise for a state s, and take the expectation with respect to the action a over the policy π. We use lower-case letters here to connote the fact that v and q may not correspond to *any* value functions. The greedy policy π_{greedy} in the non-entropy-regularized RL setting can therefore be written as

$$\pi_{greedy} := \arg\max_\pi \mathcal{T}^\pi v,$$

where the arg max is taken for each state separately. To define a sense of *greedy* for entropy-regularized RL, we can define entropy-regularized Bellman operators and perform a similar arg max.

First, for a policy π and temperature τ, we can define $\mathcal{T}^\pi_{\mathcal{H}}$, the entropy-regularized Bellman operator for π.

$$(\mathcal{T}^\pi_{\mathcal{H}} v)[s] = \mathbb{E}_\pi[q(s,a)] + \tau \mathcal{H}(\pi(\cdot \mid s))$$

where $\mathcal{H}$ is the entropy function. At a given state, the greedy policy is given by

$$\pi_{greedy}(\cdot \mid s) := \arg\max_{\pi_s \in \Delta_{\mathcal{A}}} \sum_a q(s,a)\pi_s(a) + \tau \mathcal{H}(\pi_s),$$

where π_s is a probability distribution over $\mathcal{A}$, and $\pi_s(a)$ refers to the a-th element of π_s. Since $\mathcal{H}$ is concave, $-\mathcal{H}$ is convex, so the greedy policy turns out to be the maximizing argument in the definition of the convex conjugate of $-\tau\mathcal{H}$ evaluated at $q(s,\cdot)$! Let's explicitly solve for this maximizing argument.

Since $\sum_a q(s,a)\pi_s(a) + \tau\mathcal{H}(\pi_s)$ is concave with respect to π_s, it suffices to find a stationary point, subject to the condition that $\sum_b \pi_s(b) = 1$. Setting λ as the Lagrange multiplier,

$$\begin{aligned}
&\frac{\partial}{\partial \pi_s(b)}\left(\sum_a q(s,a)\pi_s(a) + \tau\mathcal{H}(\pi_s) - \lambda\sum_a \pi_s(a)\right) = q(s,b) - \tau\log\pi_s(b) - \tau + \lambda \\
&\quad\implies \pi_s(b) = \exp\big(q(s,b)\tau^{-1} - 1 - \lambda\tau^{-1}\big) \\
&\quad\implies \pi_s(b) \propto \exp\big(q(s,b)\tau^{-1}\big) \\
&\quad\implies \pi_s(b) = \frac{\exp(q(s,b)\tau^{-1})}{\sum_a \exp(q(s,a)\tau^{-1})}.
\end{aligned}$$

One interpretation of this result is that as the temperature decreases, the soft action-value becomes closer to the unregularized action-value. In this setting, we do want to act more greedily, and care relatively less about maximizing entropy. As the temperature decreases, the Boltzmann distribution becomes more sharply peaked at the maximum of the logits.

2.3.2 Approximate Greedification with the KLs

Returning to the question of objectives, the idea is to set $q_\theta = \pi_\theta(\cdot|s)$, $p = \mathcal{B}Q_\tau(s,\cdot)$, and use a KL divergence to bring π_θ closer to $\mathcal{B}Q_\tau$. One might wonder why we cannot set $\pi(\cdot|s) = \mathcal{B}Q_\tau(s,\cdot)$ in practice and be done with it. Indeed, for discrete action spaces, we can draw actions from $\mathcal{B}Q_\tau(s,\cdot)$ easily at each time step. However, for continuous actions, even calculating $\mathcal{B}Q_\tau(s,\cdot)$ requires approximating a (usually) intractable integral. Furthermore, even in the discrete-action regime, using $\mathcal{B}Q_\tau$ might not be desirable as Q is usually just an action-value *estimate*.

Define the **Reverse KL** (RKL) for greedification at a given state s and action-value Q (Q may be soft or not):

$$\mathrm{RKL}(\theta; s, Q) := \mathrm{KL}\left(\pi_\theta(\cdot \mid s) \parallel \mathcal{B}Q_\tau(s,\cdot)\right)$$

Notice that τ plays the role of an entropy regularization parameter: a larger τ results in more entropy regularization on $\pi_\theta(\cdot \mid s)$. We can take a limiting case with no entropy regularization to get the **Hard Reverse KL**.

$$\text{Hard RKL}(\theta; s, Q) := \lim_{\tau\to 0} \tau\mathrm{RKL}(\theta; s, Q) = -\int_{\mathcal{A}} \pi_\theta(a \mid s)Q(s,a)\,da \tag{2.6}$$

If we view the action-value Q as fixed, the gradient of Equation (2.6) is exactly the inner term of the policy gradient in Equation (2.2).[9] This means that the typical policy gradient update in actor-critic can be thought of as a greedification step with a hard reverse KL.

Similarly, we can define the **Forward KL** (FKL) for greedification

$$\mathrm{FKL}(\theta; s, Q) := \mathrm{KL}\left(\mathcal{B}Q_\tau(s, \cdot) \parallel \pi_\theta(\cdot \mid s)\right)$$

Finally, we can again consider a limiting case, where the temperature parameters goes to zero, to get a **Hard Forward KL** objective.

$$\begin{aligned}
\text{Hard FKL}(\theta; s, Q) &:= \lim_{\tau \to 0} \mathrm{FKL}(\theta; s, Q) \\
&= -\lim_{\tau \to 0} \int_{\mathcal{A}} \frac{\exp(Q(s,a)\tau^{-1})}{\int_{\mathcal{A}} \exp(Q(s,b)\tau^{-1})\, db} \log \pi_\theta(a \mid s)\, da \\
&= -\int_{\mathcal{A}} \lim_{\tau \to 0} \frac{\exp(Q(s,a)\tau^{-1})}{\int_{\mathcal{A}} \exp(Q(s,b)\tau^{-1})\, db} \log \pi_\theta(a \mid s)\, da \\
&= -\int_{\mathcal{A}} 1_{a = \arg\max_b Q(s,b)} \log \pi_\theta(a \mid s)\, da \\
&= -\log \pi_\theta(\arg\max_a Q(s,a) \mid s)
\end{aligned}$$

This expression looks quite similar to the cross-entropy loss in supervised classification, if one views the maximum action of $Q(s, \cdot)$ as the correct class of state s. The FKL has been used for a CPI algorithm (Vieillard et al., 2020), but we are unaware of any literature that analyzes the Hard FKL.

Although switching π_θ and $\mathcal{B}Q_\tau$ might seem like a small change, there are several consequences. The forward KL is popularly known to be *mean-seeking*: to minimize the forward KL, π_θ will likely place mass on the a with the largest probability mass according to $\mathcal{B}Q_\tau$. Furthermore, the forward KL can be more difficult to optimize because it requires access to $\mathcal{B}Q_\tau$ to sample the gradient. But, favourably, if π_θ is parameterized with a Boltzmann distribution over θ, then the forward KL is convex with respect to θ. The reverse KL, on the other hand, is characterized as *mode-seeking*: if $\mathcal{B}Q_\tau(s, a)$ is small for a given a, then $\pi_\theta(a \mid s)$ is also forced to be small. The RKL can also be easier to optimize as

[9] We are unaware of a previous statement of this result in the literature, but some references to a connection between value-based methods with entropy regularisation and policy gradient can be found in (Nachum et al., 2017b).

access to p is not required to sample the gradient. Less favourably, however, it is generally not convex with respect to θ, even if π_θ is parameterized with a Boltzmann distribution.

2.3.3 The Weighting over States

The above greedification objectives, and corresponding gradients, are defined per state. To specify the full greedification objective across states, we need a weighting $d : \mathcal{S} \to [0, \infty)$ on the relative importance of each state; under function approximation, the agent requires this distribution to trade-off accuracy of greedification across states. The full objective for the RKL is $\int_{\mathcal{S}} d(s)\text{RKL}(\theta; s, Q)$; the other objectives are specified similarly.

This role of the weighting might seem quite different from the typical role in the policy gradient, but there are some clear connections. When averaging the gradient of the Hard RKL with weighting d, we have

$$-\int_{\mathcal{S}} d(s) \int_{\mathcal{A}} Q(s,a) \nabla \pi_\theta(a \mid s)\, da\, ds.$$

For this to correspond to the true policy gradient when $\tau = 0$ and $Q = Q^{\pi_\theta}$, the weighting should be $d = d^{\pi_\theta}$; otherwise, this quantity may not correspond to the gradient of any function (Nota and Thomas, 2020). The weighting d^{π_θ} indicates that the weighting for greedification should be higher in states closer to the start state. This choice is sensible for allocating function approximation resources; it seems key to get action selection as accurate as possible in early states given the downstream effects.

Nevertheless, other choices are possible. An open, worthwhile question is to better understand which weightings can help avoid poor stationary points and improve convergence rates. Many algorithms in practice use something closer to a uniform weighting on observed data. It is as yet not well understood what weighting is ideal, nor the implications from deviating from the policy gradient weighting. There are, however, some insights from both CPI and from the literature on policy gradients. It is clear that there are some instances where using $d \neq d^\pi$ results in convergence to a poor stationary point: $\int_{\mathcal{S}} d(s) \int_{\mathcal{A}} Q^{\pi_\theta}(s,a) \nabla \pi_\theta(a \mid s)\, da\, ds = 0$ for a certain d that produces a highly

suboptimal π_θ, whereas weighting by $d = d^{\pi_\theta}$ (or with an emphatic weighting) does not (Imani et al., 2018). This counterexample assumes exact $Q = Q^{\pi_\theta}$, but still has implications for API, if an exact policy evaluation step is used. It seems likely that a similar counterexample could be found for nearly accurate Q. On the other hand, the work on CPI indicates that the weighting with d^π can require a large number of samples to get accurate gradient estimates, and moving to a more uniform weighting over states is significantly better (Kakade and Langford, 2002).

Chapter 3

Theoretical Results

In this section, we consider the policy improvement guarantees, or lack thereof, for the reverse and forward KL. To the best of our knowledge, there is as yet only one with guarantees: the reverse KL when assuming exact minimization in each state (Haarnoja et al., 2018, Lemma 2). We first provide an extension of this result to rely only upon reverse KL minimization *on average* across states. Next, we provide a counterexample where optimizing the forward KL *does not* induce policy improvement. Finally, we discuss further assumptions that can be made to ensure that forward KL does induce policy improvement.

3.1 Reverse KL

Throughout, we assume that the class of policies Π consists of policies whose entropies are finite; this assumption is not restrictive for finite action-spaces as entropy is always finite in that setting. The assumption of finite entropies is necessary just to ensure that the soft value functions are well-defined.

First, we note a strengthening of the original result for policy improvement under the reverse KL (Haarnoja et al., 2018). By examining the proof of their Lemma 2, their new policy π_{new} does not have to minimize the reverse KL; rather, it suffices that π_{new} is smaller in reverse KL than π_{old} at every state s.

Lemma 1 (Policy Improvement under RKL Reduction, Restatement of Lemma 2 (Haarnoja et al., 2018))**.** *For $\pi_{\text{old}}, \pi_{\text{new}} \in \Pi$, if for all s*

$$\mathrm{KL}(\pi_{\text{new}}(\cdot \mid s) \parallel \mathcal{B}Q_\tau^{\pi_{\text{old}}}(s, \cdot)) \leq \mathrm{KL}(\pi_{\text{old}}(\cdot \mid s) \parallel \mathcal{B}Q_\tau^{\pi_{\text{old}}}(s, \cdot))$$

then $Q_\tau^{\pi_{\text{new}}}(s,a) \geq Q_\tau^{\pi_{\text{old}}}(s,a)$ *for all* (s,a) *and* $\tau > 0$.

Proof. Same proof as in Haarnoja et al. (2018). □

It turns out that a version of this result is true under expectation. First, however, it will be useful to note a "soft" counterpart to the classical performance difference lemma (Kakade and Langford, 2002). To state the performance difference lemma, we need to define some concepts.

Definition 1 (Performance Criterion)**.** *For a start state distribution* ρ_0*, the performance criterion is defined to be*

$$\eta(\pi) := \mathbb{E}_{\rho_0}[V^\pi(s)].$$

Definition 2 (State Visitation Distribution)**.** *For a policy* π *and starting state distribution* ρ*, the (future) state visitation distribution is defined as follows.*

$$d^\pi(s) := (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \Pr(S_t = s \mid s_0 \sim \rho_0, \pi),$$

where $\Pr(s_t = s \mid s_0 \sim \rho_0, \pi)$ *is the probability of the* t*-th state with respect to the start state distribution and the trajectory induced by* π*.*

Often in the literature, Definition 2 is defined without the additional $(1-\gamma)$ term in front. This version is known as the *unnormalized* visitation distribution.

Definition 3 (Advantage)**.** *For any policy* π*, the advantage is*

$$A^\pi(s,a) := Q^\pi(s,a) - V^\pi(s).$$

The advantage asks, what is the average benefit if I take action a in state s, as opposed to drawing an action from π?

The classical performance difference lemma is the following.

Lemma 2 (Performance Difference Lemma (Kakade and Langford, 2002))**.** *For any policies* $\pi_{\text{old}}, \pi_{\text{new}}$*, we obtain the following relationship when using non-soft values,*

$$\eta(\pi_{\text{new}}) = \eta(\pi_{\text{old}}) + \frac{1}{1-\gamma}\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[A^{\pi_{\text{old}}}(s,a)],$$

where the notation $\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}$ *indicates the distributions for the states and actions, respectively*

$$\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[A^{\pi_{\text{old}}}(s,a)] := \int_{\mathcal{S}} d^{\pi_{\text{new}}}(s) \int_{\mathcal{A}} \pi_{\text{new}}(a|s) A^{\pi_{\text{old}}}(s,a)\ da\ ds.$$

Let's define the soft performance criterion.

Definition 4 (Soft Performance Criterion)**.** *The soft performance criterion is defined by the following, for a given temperature* $\tau > 0$*.*

$$\eta_\tau(\pi) := \mathbb{E}_{\rho_0}[V_\tau^\pi(s)].$$

It will also be helpful to have a soft version of the advantage. An intuition for the advantage in the non-soft setting is that it should be zero when averaged over π. To enforce this requirement in the soft setting, we require a small modification.

Definition 5 (Soft Advantage)**.** *For a policy* π *and temperature* $\tau > 0$*, the soft advantage is*

$$A_\tau^\pi(s,a) := Q_\tau^\pi(s,a) - \tau \log \pi(a \mid s) - V_\tau^\pi(s).$$

If $\tau = 0$, we recover the usual definition of the advantage function. It is also true that $\mathbb{E}_\pi[A_\tau^\pi(s,a)] = 0$, by definition of the soft value functions. We are ready for a soft performance difference lemma.

Lemma 3 (Soft Performance Difference)**.** *For any policies* $\pi_{\text{old}}, \pi_{\text{new}}$*, the following is true for any* $\tau \geq 0$*.*

$$\eta_\tau(\pi_{\text{new}}) - \eta_\tau(\pi_{\text{old}}) = \\ \frac{1}{1-\gamma}\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[A_\tau^{\pi_{\text{old}}}(s,a)] + \frac{\tau}{1-\gamma}\mathbb{E}_{d^{\pi_{\text{new}}}}[\text{KL}(\pi_{\text{new}}(\cdot \mid s) \parallel \pi_{\text{old}}(\cdot \mid s))].$$

Proof. The calculation is straightforward. When we write $\mathbb{E}$ without any subscripts, we mean the expectation over the trajectory distribution induced

by ρ_0 and π_{new}.

$$\frac{1}{1-\gamma}\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[A_\tau^{\pi_{\text{old}}}(s,a)]] = \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t A^{\pi_{\text{old}}}(s_t,a_t)\right]$$

$\triangleright$ by definition of the visitation distribution

$$= \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t(Q_\tau^{\pi_{\text{old}}}(s_t,a_t) - \tau\log\pi_{\text{old}}(\cdot \mid s_t) - V^{\pi_{\text{old}}}(s_t))\right]$$

$\triangleright$ by definition of the soft advantage

$$= \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t(r(s_t,a_t) + \gamma V_\tau^{\pi_{\text{old}}}(s_{t+1}) - \tau\log\pi_{\text{old}}(\cdot \mid s_t) - V_\tau^{\pi_{\text{old}}}(s_t))\right]$$

$\triangleright$ from expanding $Q_\tau^{\pi_{\text{old}}}$ and pulling the expectation outside

$$= \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t(r(s_t,a_t) - \tau\log\pi_{\text{old}}(\cdot \mid s_t)) - V_\tau^{\pi_{\text{old}}}(s_0)\right]$$

$\triangleright$ from the telescoping series $\gamma V_\tau^{\pi_{\text{old}}}(s_{t+1}) - V_\tau^{\pi_{\text{old}}}(s_t)$

$$= -\eta_\tau(\pi_{\text{old}}) + \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t(r(s_t,a_t) - \tau\log\pi_{\text{old}}(\cdot \mid s_t))\right]$$

$\triangleright$ by definition of $\eta_\tau(\pi_{\text{old}})$

$$= -\eta_\tau(\pi_{\text{old}}) + \eta_\tau(\pi_{\text{new}}) + \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t\tau(\log\pi_{\text{new}}(\cdot \mid s_t) - \log\pi_{\text{old}}(\cdot \mid s_t))\right]$$

$\triangleright$ adding and subtracting $\tau\log\pi_{\text{new}}(\cdot \mid s_t)$

$$= -\eta_\tau(\pi_{\text{old}}) + \eta_\tau(\pi_{\text{new}}) + \frac{\tau}{1-\gamma}\mathbb{E}_{d^{\pi_{\text{new}}}}[\mathrm{KL}(\pi_{\text{new}}(\cdot \mid s) \parallel \pi_{\text{old}}(\cdot \mid s))]$$

□

The presence of the extra KL term in Lemma 3 is intriguing. Since the KL divergence is non-negative, one may be able improve upon $\eta_\tau(\pi_{\text{old}})$ even with a negative advantage $A_\tau^\pi(s,a)$; the degree of this "bonus" is modulated by τ. If we set $\tau = 0$, we recover the classical performance difference lemma.

Proposition 1 (Policy Improvement under Average RKL Reduction)**.** *For* $\pi_{\text{old}}, \pi_{\text{new}} \in \Pi$*, if*

$$\mathbb{E}_{d^{\pi_{\text{new}}}}[\mathrm{KL}(\pi_{\text{old}}(\cdot \mid s) \parallel \mathcal{B}Q_\tau^{\pi_{\text{old}}}(s,\cdot))] \geq \mathbb{E}_{d^{\pi_{\text{new}}}}[\mathrm{KL}(\pi_{\text{new}}(\cdot \mid s) \parallel \mathcal{B}Q_\tau^{\pi_{\text{old}}}(s,\cdot))],$$

then $\eta_\tau(\pi_{\text{new}}) \geq \eta_\tau(\pi_{\text{old}})$.

Proof. First, we calculate the KL inequality.

$$\mathbb{E}_{d^{\pi_{\text{new}}}}[\text{KL}(\pi_{\text{old}}(\cdot \mid s) \parallel \mathcal{B}Q_\tau^{\pi_{\text{old}}}(s,\cdot))] \geq \mathbb{E}_{d^{\pi_{\text{new}}}}[\text{KL}(\pi_{\text{new}}(\cdot \mid s) \parallel \mathcal{B}Q_\tau^{\pi_{\text{old}}}(s,\cdot))]$$

$$\implies \mathbb{E}_{d^{\pi_{\text{new}}}}[\mathbb{E}_{\pi_{\text{old}}}[\tau \log \pi_{\text{old}}(\cdot \mid s) - Q^{\pi_{\text{old}}}(s,\cdot)]]$$

$$\geq \mathbb{E}_{d^{\pi_{\text{new}}}}[\mathbb{E}_{\pi_{\text{new}}}[\tau \log \pi_{\text{new}}(\cdot \mid s) - Q^{\pi_{\text{old}}}(s,\cdot)]]$$

$\triangleright$ expanding the RKL definition and multiplying by τ

$$\implies -\mathbb{E}_{d^{\pi_{\text{new}}}}[V^{\pi_{\text{old}}}(s)] \geq \mathbb{E}_{d^{\pi_{\text{new}}}}[\mathbb{E}_{\pi_{\text{new}}}[\tau \log \pi_{\text{new}}(\cdot \mid s) - Q^{\pi_{\text{old}}}(s,\cdot)]$$

$\triangleright$ because $V_\tau^{\pi_{\text{old}}}(s) = \mathbb{E}_{\pi_{\text{old}}}[Q_\tau^{\pi_{\text{old}}}(s,a) - \tau \log \pi_{\text{old}}(a \mid s)]$

$$\implies \mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[Q_\tau^{\pi_{\text{old}}}(s,a) - V_\tau^{\pi_{\text{old}}}(s)] \geq \tau \mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[\log \pi_{\text{new}}(\cdot \mid s)]$$

$\triangleright$ rearranging

$$\implies \mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[A_\tau^{\pi_{\text{old}}}(s,a)] \geq \tau \mathbb{E}_{d^{\pi_{\text{new}}}}[\text{KL}(\pi_{\text{new}}(\cdot \mid s) \parallel \pi_{\text{old}}(\cdot \mid s))]$$

$\triangleright$ subtracting $\tau \mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[\log \pi_{\text{old}}(\cdot \mid s)]$ from both sides

Since the KL divergence is non-negative, the inequality we just derived implies that the RHS of Lemma 3 is also non-negative. Hence, $\eta_\tau(\pi_{\text{new}}) \geq \eta_\tau(\pi_{\text{old}})$. $\square$

Note that we cannot set $\tau = 0$ in the above proof as we divide by τ in $\mathcal{B}Q$.

We remark that the guarantee for Proposition 1 is in terms of η_τ, whereas the result in Lemma 1 is in terms of Q_τ^π. The reason for this distinction is that in assuming an average KL reduction in Proposition 1, there does not seem to be a way to extract the action values on a per-state basis. Hence, the guarantee must of necessity be an average guarantee across states, rather than a guarantee for the value function at each state.

3.2 Forward KL

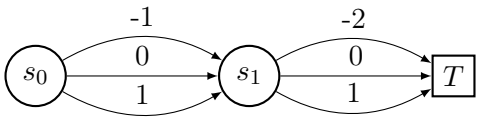
Unfortunately, the same policy improvement properties do not hold unmodified for the forward KL.

Proposition 2 (Counterexample for Policy Improvement with FKL)**.** *There exists an MDP, a state s', an initial policy π_{old}, policy π_{new}, and temperature $\tau > 0$ such that for any $\gamma \in (0, 1]$*

$$\forall s \in S,\ \text{KL}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s,\cdot) \parallel \pi_{\text{old}}(\cdot \mid s) \geq \text{KL}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s,\cdot) \parallel \pi_{\text{new}}(\cdot \mid s))$$

but $\forall a \in \mathcal{A}$, $Q_\tau^{\pi_{\text{new}}}(s', a) < Q_\tau^{\pi_{\text{old}}}(s', a)$.

Proof. Let us first construct the MDP. The MDP is episodic with two states and three actions in each state. s_0 is the start state, transitioning to state s_1 under any action, which then transitions to the terminal state under any action. The numbers near the arrows represent the reward upon taking the action corresponding to an arrow, from a state. From top to bottom, we have actions a_0, a_1, a_2.



Let our initial policy π_{old} be the equiprobable policy for all states and actions. That is, $\forall\,(s, a)$, $\pi_{\text{old}}(a \mid s) = \frac{1}{3}$. Consider π_{new} to be the following.

$$\begin{aligned}
\pi_{\text{new}}(\cdot \mid s_0) &= \pi_{\text{old}}(\cdot \mid s_0) \\
\pi_{\text{new}}(a_0 \mid s_1) &= 3/8 \\
\pi_{\text{new}}(a_1 \mid s_1) &= 2/8 \\
\pi_{\text{new}}(a_2 \mid s_1) &= 3/8
\end{aligned}$$

Our strategy is the following: show that (1) π_{new} reduces the forward KL as it "commits" more at s_1, but that (2) π_{old} has higher entropy and thus a higher soft action-value at s_0. A straightforward calculation provides the soft value functions of π_{old} and π_{new} at s_1.

$$\begin{aligned}
V_\tau^{\pi_{\text{old}}}(s_1) &= -1/3 + \tau \ln 3 \\
V_\tau^{\pi_{\text{new}}}(s_1) &= -2 \cdot 3/8 + 3/8 + \tau \mathcal{H}(\pi_{\text{new}}(\cdot \mid s_1)) = -3/8 + \tau \mathcal{H}(\pi_{\text{new}}(\cdot \mid s_1))
\end{aligned}$$

Recall that the soft action-value can be written as

$$Q_\tau^\pi(s, a) := r(s, a) + \gamma \mathbb{E}_{s' \sim p(\cdot \mid s, a)}[V^\pi(s)].$$

Because this MDP is deterministic, we can rewrite the expectation above as

the evaluation $V^{\pi}(s')$. The soft action-value of π_{old} is

$$\begin{aligned}
Q_\tau^{\pi_{\text{old}}}(s_0, a_0) &= -1 + \gamma\, V_\tau^{\pi_{\text{old}}}(s_1) = \gamma\,\tau \ln 3 - \frac{3+\gamma}{3} & Q_\tau^{\pi_{\text{old}}}(s_1, a_0) &= -2 \\
Q_\tau^{\pi_{\text{old}}}(s_0, a_1) &= 0 + \gamma\, V_\tau^{\pi_{\text{old}}}(s_1) = \gamma\,\tau \ln 3 - \frac{\gamma}{3} & Q_\tau^{\pi_{\text{old}}}(s_1, a_1) &= 0 \\
Q_\tau^{\pi_{\text{old}}}(s_0, a_2) &= 1 + \gamma\, V_\tau^{\pi_{\text{old}}}(s_1) = \frac{3-\gamma}{3} + \gamma\,\tau\, \ln 3 & Q_\tau^{\pi_{\text{old}}}(s_1, a_2) &= 1
\end{aligned}$$

The exponentiated action-value $\mathcal{B}Q_\tau$ is as follows.

$$\mathcal{B}Q_\tau(s,a) := \frac{\exp(Q(s,a)\tau^{-1})}{\sum_b \exp(Q(s,b)\tau^{-1})}$$

The partition function at s_1 is given by the following.

$$Z(s_1) := \sum_a \exp\big(Q_\tau^{\pi_{\text{old}}}(s_1,a)\tau^{-1}\big) = 1 + \exp\big(\tau^{-1}\big) + \exp\big(-2\tau^{-1}\big).$$

Calculating the exponentiated soft action-values at s_1,

$$\begin{aligned}
\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_1, a_0) &= \frac{\exp(-2\tau^{-1})}{1 + \exp(\tau^{-1}) + \exp(-2\tau^{-1})}, \\
\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_1, a_1) &= \frac{1}{1 + \exp(\tau^{-1}) + \exp(-2\tau^{-1})}, \\
\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_1, a_2) &= \frac{\exp(\tau^{-1})}{1 + \exp(\tau^{-1}) + \exp(-2\tau^{-1})}.
\end{aligned}$$

The new soft action-value at s_0 is given by

$$\begin{aligned}
Q_\tau^{\pi_{\text{new}}}(s_0, a_0) &= -1 + \gamma\, V_\tau^{\pi_{\text{new}}}(s_1) \\
&= -1 - \frac{3\gamma}{8} + \gamma\,\tau\,\mathcal{H}(\pi_{\text{new}}(\cdot \mid s_1)) \\
&= \frac{-8 - 3\gamma}{8} + \gamma\,\tau\,\mathcal{H}(\pi_{\text{new}}(\cdot \mid s_1)), \\
Q_\tau^{\pi_{\text{new}}}(s_0, a_1) &= -\frac{3\,\gamma}{8} + \gamma\,\tau\,\mathcal{H}(\pi_{\text{new}}(\cdot \mid s_1)), \\
Q_\tau^{\pi_{\text{new}}}(s_0, a_2) &= \frac{8 - 3\gamma}{8} + \gamma\,\tau\,\mathcal{H}(\pi_{\text{new}}(\cdot \mid s_1))
\end{aligned}$$

Let's now compare $Q_\tau^{\pi_{\text{new}}}$ and $Q_\tau^{\pi_{\text{old}}}$. Note that entropy is maximized by a uniform distribution, so $\tau\,\mathcal{H}(\pi_{\text{new}}(\cdot \mid s_1)) < \tau \ln 3$, regardless of the value of $\tau > 0$ and $\gamma > 0$. Let's put the reward terms under a common denominator.

$$\begin{aligned}
Q_\tau^{\pi_{\text{old}}}(s_0, a_0) - Q_\tau^{\pi_{\text{new}}}(s_0, a_0) &> -\frac{3+\gamma}{3} + \frac{8+3\gamma}{8} &&= \frac{\gamma}{24}, \\
Q_\tau^{\pi_{\text{old}}}(s_0, a_1) - Q_\tau^{\pi_{\text{new}}}(s_0, a_1) &> -\frac{\gamma}{3} + \frac{3\,\gamma}{8} &&= \frac{\gamma}{24}, \\
Q_\tau^{\pi_{\text{old}}}(s_0, a_2) - Q_\tau^{\pi_{\text{new}}}(s_0, a_2) &> \frac{3-\gamma}{3} - \frac{8-3\gamma}{8} &&= \frac{\gamma}{24}.
\end{aligned}$$

Thus, for any $\gamma \in (0, 1]$, $Q_\tau^{\pi_{\text{new}}}(s_0, a) < Q_\tau^{\pi_{\text{old}}}(s_0, a)$ for all actions a.

Let's now compare the forward KL divergences. Because the policies are the same at s_0, we have

$$\mathrm{KL}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_0, \cdot) \parallel \pi_{\text{new}}(\cdot \mid s_0)) = \mathrm{KL}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_0, \cdot) \parallel \pi_{\text{old}}(\cdot \mid s_0)).$$

Calculating the rest of the forward KL divergences,

$$\begin{aligned}
&\mathrm{KL}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_1, \cdot) \parallel \pi_{\text{old}}(\cdot \mid s_1)) = \\
&\quad - \mathcal{H}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_1, \cdot)) - \log(1/3)\frac{\exp(-2\tau^{-1}) + 1 + \exp(\tau^{-1})}{1 + \exp(\tau^{-1}) + \exp(-2\tau^{-1})} \\
&\mathrm{KL}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_1, \cdot) \parallel \pi_{\text{new}}(\cdot \mid s_1)) = \\
&\quad - \mathcal{H}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_1, \cdot)) - \frac{\exp(-2\tau^{-1})\log(3/8) + \log(2/8) + \exp(\tau^{-1})\log(3/8)}{1 + \exp(\tau^{-1}) + \exp(-2\tau^{-1})}
\end{aligned}$$

In order to have $\mathrm{KL}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_1, \cdot) \parallel \pi_{\text{old}}(\cdot \mid s_1)) > \mathrm{KL}(\mathcal{B}Q_\tau^{\pi_{\text{old}}}(s_1, \cdot) \parallel \pi_{\text{new}}(\cdot \mid s_1))$, it is sufficient and necessary to have

$$\begin{aligned}
&- \log(1/3)\frac{\exp(-2\tau^{-1}) + 1 + \exp(\tau^{-1})}{1 + \exp(\tau^{-1}) + \exp(-2\tau^{-1})} \\
&\quad > -\frac{\exp(-2\tau^{-1})\log(3/8) + \log(2/8) + \exp(\tau^{-1})\log(3/8)}{1 + \exp(\tau^{-1}) + \exp(-2\tau^{-1})}.
\end{aligned}$$

Or, after simplifying,

$$\begin{aligned}
&- \log(1/3)(\exp\big(-2\tau^{-1}\big) + 1 + \exp\big(\tau^{-1}\big)) \\
&\quad > -(\exp\big(-2\tau^{-1}\big)\log(3/8) + \log(2/8) + \exp\big(\tau^{-1}\big)\log(3/8)) \\
&\iff \exp\big(-2\tau^{-1}\big)(\log(3/8) - \log(1/3)) + \log(2/8) - \log(1/3) \\
&\qquad + \exp\big(\tau^{-1}\big)(\log(3/8) - \log(1/3)) > 0 \\
&\iff \exp\big(-2\tau^{-1}\big)\log(9/8) + \log(3/4) + \exp\big(\tau^{-1}\big)\log(9/8) > 0 \\
&\iff \log(9/8) + \log(3/4)\exp\big(2\tau^{-1}\big) + \exp\big(3\tau^{-1}\big)\log(9/8) > 0.
\end{aligned}$$

Setting $y = \exp(\tau^{-1})$, we are interested in the cubic inequality

$$\log(9/8) + \log(3/4)y^2 + y^3\log(9/8) > 0.$$

Noting that the leading term of the cubic is positive, there must be some y_0 such that for $y > y_0$, the inequality above holds. This y_0 must be the largest root

of the cubic expression. A root-solver gives $y_0 > 2.25$, so the above inequality holds for $\exp(\tau^{-1}) = y > 2.25$, or in other words for $1.23 \approx \frac{1}{\log 2.25} > \tau$.

□

Proposition 2 shows that there is an MDP and two policies in which one policy π_{new} can have a lower FKL, but fail to be a better policy for small enough τ. It might seem like Proposition 2 could contradict the RKL improvement result in Lemma 1, but Proposition 2 is a statement about a particular policy, and not about any policy that reduces the RKL.

From examining the proof of Proposition 2, it might also be surprising that the result holds for sufficiently small τ. That is, for $\tau < \tau_0$, for some τ_0, there will always be a policy that reduces the FKL, but that has strictly worse performance than the previous policy. Such a result might seem unintuitive since one would expect that the more one reduces the FKL, the better the policy should be, given that the FKL is minimized by setting the subsequent policy to be exactly the target distribution. Nevertheless, such reasoning says nothing about the path of policies to the superior, target distribution policy.

A natural question is if this counterexample is pathological. Since reducing the FKL to 0 corresponds to reducing the RKL to 0 (i.e., setting $\pi_{\text{new}} = \mathcal{B}Q_\tau^{\pi_{\text{old}}}$), it seems plausible that if the FKL is reduced enough, the new policy π_{new} will be better. One may also wonder if by selecting a temperature judiciously enough, optimizing for the forward KL might induce policy improvement. With some qualifications, the answer is in the positive.

Proposition 3 (Policy Improvement for FKL with Sufficient Reduction)**.** *Assume a discrete action space with $|\mathcal{A}| < \infty$, with a policy space Π that consists of policies where $\pi(a \mid s) > 0$ for all a. Let C, $\pi_{\text{old}}, \pi_{\text{new}} \in \Pi$ be such that for a state s,*

$$\text{KL}(\mathcal{B}Q^{\pi_{\text{old}}}(s, \cdot) \parallel \pi_{\text{new}}(\cdot \mid s)) + C \leq \text{KL}(\mathcal{B}Q^{\pi_{\text{old}}}(s, \cdot) \parallel \pi_{\text{old}}(\cdot \mid s)), \tag{3.1}$$

where C additionally satisfies

$$C \geq \frac{1}{2}\sum_a \left(1 - \frac{1}{\pi_{\text{old}}(a \mid s)}\right)^2 \left(1 + \frac{Q^{\pi_{\text{old}}}(s,a)}{\tau} + \frac{\exp(\tau^{-1}Q^{\pi_{\text{old}}}(s,a))Q^{\pi_{\text{old}}}(s,a)^2}{2\tau^2}\right)$$
$$+ \frac{1}{2}\sum_a \exp\big(\tau^{-1}Q^{\pi_{\text{old}}}(s,a)\big)\tau^{-2}Q^{\pi_{\text{old}}}(s,a)^2(1 - \pi_{\text{old}}(a \mid s))$$

with $\tau > 0$. Then,

$$\sum_a Q^{\pi_{\text{old}}}(s,a)\pi_{\text{old}}(a \mid s) \leq \sum_a Q^{\pi_{\text{old}}}(s,a)\pi_{\text{new}}(a \mid s).$$

Proof. For the steps below, we will need to consider shifted, positive values $Q^{\pi_{\text{old}}}(s,a) + b$ where $b := \max(0, -\min_{a\in\mathcal{A}} Q^{\pi_{\text{old}}}(s,a)) + 0.01$. This shift does not affect Equation (3.1) because the Boltzmann function is invariant to shifts in the logits. It suffices to focus on a single state s. We will suppress dependence on this state for notational simplicity. Let $\pi(a) := \pi_{\text{old}}(a|s)$ and $Q(a) := Q^{\pi_{\text{old}}}(s,a) + b > 0$. After cancelling the common entropy term and multiplying both sides by the partition function, Equation (3.1) yields

$$\sum_a \exp\big(\tau^{-1}Q(a)\big)\log\frac{1}{\pi_{\text{new}}(a)} + C \leq \sum_a \exp\big(\tau^{-1}Q(a)\big)\log\frac{1}{\pi_{\text{old}}(a)}. \tag{3.2}$$

For $x \geq 0$, we have $1+x \leq \exp(x)$ and for $x \in (0,1)$ we have $\log\frac{1}{x} = -\log(x) \geq 1 - x$. Since we assumed $Q(a) > 0$ and π_{old}, π_{new} are not deterministic, these inequalities may be applied. Applying them to Equation (3.2) gives

$$\sum_a (1 + \tau^{-1}Q(a))(1 - \pi_{\text{new}}(a)) + C \leq \sum_a \exp\big(\tau^{-1}Q(a)\big)\log\frac{1}{\pi_{\text{old}}(a)}. \tag{3.3}$$

To get any further, we proceed by Taylor expansion. We will expand $\exp(x)$ around $x = 0$ and $\log(x)$ around $x = 1$. Now, using the Lagrange form of the

remainder of Taylor expansions yields

$$\begin{aligned}
\exp\big(\tau^{-1}Q(a)\big) &= 1 + \tau^{-1}Q(a) + \frac{1}{2}\exp(\xi_a)\tau^{-2}Q(a)^2, \\
\implies \exp\big(\tau^{-1}Q(a)\big) &\leq 1 + \tau^{-1}Q(a) + \frac{1}{2}\exp\big(\tau^{-1}Q(a)\big)\tau^{-2}Q(a)^2, \\
\log(\pi_{\text{old}}(a)) &= \pi_{\text{old}}(a) - 1 - \frac{1}{2\kappa_a^2}(\pi_{\text{old}}(a) - 1)^2, \\
\implies \log\frac{1}{\pi_{\text{old}}(a)} &= 1 - \pi_{\text{old}}(a) + \frac{1}{2\kappa_a^2}(\pi_{\text{old}}(a) - 1)^2 \\
\implies \log\frac{1}{\pi_{\text{old}}(a)} &\leq 1 - \pi_{\text{old}}(a) + \frac{1}{2\pi_{\text{old}}(a)^2}(\pi_{\text{old}}(a) - 1)^2 \\
\implies \log\frac{1}{\pi_{\text{old}}(a)} &\leq 1 - \pi_{\text{old}}(a) + \frac{1}{2}\left(1 - \frac{1}{\pi_{\text{old}}(a)}\right)^2
\end{aligned}$$

for some $\xi_a \in (0, \tau^{-1}Q(a))$ and $\kappa_a \in (\pi_{\text{old}}(a), 1)$. Using the inequalities on the RHS of Equation (3.3) yields the following.

$$\begin{aligned}
&\sum_a \exp\big(\tau^{-1}Q(a)\big)\log\frac{1}{\pi_{\text{old}}(a)} \\
&\qquad \leq \sum_a (1 + \tau^{-1}Q(a))(1 - \pi_{\text{old}}(a)) + \mathbf{X},
\end{aligned}$$

where $\mathbf{X}$ represents terms we have neglected for the moment. From re-examining Equation (3.3), noting that $\sum_a(1 - \pi_{\text{old}}(a)) = \sum_a(1 - \pi_{\text{new}}(a))$, and canceling out $\sum_a \tau^{-1}Q(a)$ on both sides, we have

$$\begin{aligned}
-\sum_a \tau^{-1}Q(a)\pi_{\text{new}}(a) + C &\leq -\sum_a \tau^{-1}Q(a)\pi_{\text{old}}(a) + \mathbf{X} \\
C - \mathbf{X} &\leq \tau^{-1}\sum_a Q(\pi_{\text{new}}(a) - \pi_{\text{old}}(a))
\end{aligned}$$

Notice that

$$\begin{aligned}
\sum_a Q(a)\pi_{\text{new}}(a) &= \sum_a (Q^{\pi_{\text{old}}}(s,a) + b)\pi_{\text{new}}(a) \\
&= \sum_a Q^{\pi_{\text{old}}}(s,a)\pi_{\text{new}}(a) + b\sum_a \pi_{\text{new}}(a) \\
&= \sum_a Q^{\pi_{\text{old}}}(s,a)\pi_{\text{new}}(a) + b \\
\implies \sum_a Q(a)(\pi_{\text{new}}(a) - \pi_{\text{old}}(a)) &= \sum_a Q^{\pi_{\text{old}}}(s,a)(\pi_{\text{new}}(a) - \pi_{\text{old}}(a))
\end{aligned}$$

Therefore, to show that $\sum_a Q^{\pi_{\text{old}}}(s,a)\pi_{\text{new}}(a) \geq \sum_a Q^{\pi_{\text{old}}}(s,a)\pi_{\text{old}}(a)$, it suffices that $C - \mathbf{X} \geq 0$, or equivalently that $C \geq \mathbf{X}$.

$$\mathbf{X} := \frac{1}{2}\sum_a \left(1 - \frac{1}{\pi_{\text{old}}(a)}\right)^2 (1 + \tau^{-1}Q(a) + \frac{1}{2}\exp\big(\tau^{-1}Q(a)\big)\tau^{-2}Q(a)^2) + \frac{1}{2}\sum_a \exp\big(\tau^{-1}Q(a)\big)\tau^{-2}Q(a)^2(1 - \pi_{\text{old}}(a))$$

Noting that every term is non-negative by assumption, the conclusion follows. □

The conclusions of Proposition 3 hold if C is large enough; as τ decreases, C must be larger, which explain the failure of policy improvement in Proposition 2. On the other hand, if τ is large, then the requirement for C becomes less onerous, although the resulting conclusion might be quite weak; soft action-values for large τ become dominated by the entropy term, weighing much less the contribution of reward from the environment.

How large can the minimal C required get? Using the MDP of Proposition 2, in Figure 3.1 we plot the minimal C as a function of temperature. The minimal C is in fact larger than $\text{KL}(\mathcal{B}Q^{\pi_{\text{old}}}(s_1,\cdot) \parallel \pi_{\text{old}}(\cdot \mid s_1))$! This suggests that the minimal C is in fact an overestimate, which seems due to the looseness of the bounds, $\log x \leq x - 1$ in particular, we used in deriving it.

If the conclusions of Figure 3.1 do hold across all states, and with an additional assumption, we can guarantee policy improvement.

Corollary 1. *If*

$$\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{old}}}[Q_\tau^{\pi_{\text{old}}}(s,a)] \leq \mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[Q_\tau^{\pi_{\text{old}}}(s,a)],$$

and if

$$\tau\mathbb{E}_{d^{\pi_{\text{new}}}}[\mathcal{H}(\pi_{\text{new}}(\cdot \mid s))] \geq \tau\mathbb{E}_{d^{\pi_{\text{new}}}}[\mathcal{H}(\pi_{\text{old}}(\cdot \mid s))],$$

then $\eta_\tau(\pi_{\text{new}}) \geq \eta_\tau(\pi_{\text{old}})$. *This conclusion also holds for* $\tau = 0$.

Proof. If $\tau = 0$, then $\sum_a \pi_{\text{old}}(a \mid s)Q^{\pi_{\text{old}}}(s,a) = V^{\pi_{\text{old}}}(s)$. If the above holds for all states s, then $V^{\pi_{\text{old}}}(s) \leq \sum_a Q^{\pi_{\text{old}}}(s,a)\pi_{\text{new}}(a \mid s)$ for all states and so

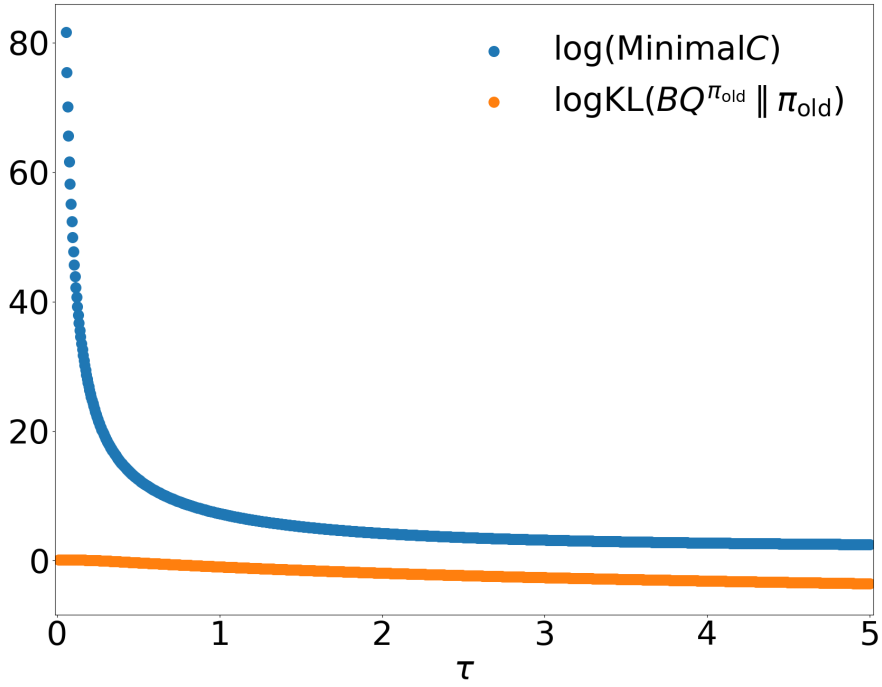


Figure 3.1: The minimal C in Proposition 3 as a function of τ, plotted against the KL divergence between $\mathcal{B}Q^{\pi_{\text{old}}}$ and π_{old} at s_1 of the MDP in Proposition 2.

$V^{\pi_{\text{new}}} \geq V^{\pi_{\text{old}}}$ by the classical policy improvement theorem (Sutton and Barto, 2018). It follows that $\eta(\pi_{\text{new}}) \geq \eta(\pi_{\text{old}})$.

If $\tau > 0$, we may use Lemma 3, the soft performance difference lemma. Note that since we are working with soft value functions, we have the following relation between the soft state-value and the soft action-value.

$$\sum_a Q_\tau^{\pi_{\text{old}}}(s,a)\pi_{\text{old}}(a \mid s) = V_\tau^{\pi_{\text{old}}}(s) - \tau\mathcal{H}(\pi_{\text{old}}(\cdot \mid s)).$$

Using the first assumption of the Corollary,

$$\begin{aligned}\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{old}}}[V_\tau^{\pi_{\text{old}}}(s) + \tau \log \pi_{\text{old}}(\cdot \mid s)] &= \mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{old}}}[Q_\tau^{\pi_{\text{old}}}(s,a)] \\ &\leq \mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[Q_\tau^{\pi_{\text{old}}}(s,a)].\end{aligned}$$

Rearranging and taking expectations, we have

$$\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[Q_\tau^{\pi_{\text{old}}}(s,a) - V_\tau^{\pi_{\text{old}}}(s)] \geq \tau \mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{old}}}[\log \pi_{\text{old}}(\cdot \mid s)],$$

$$\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[A_\tau^{\pi_{\text{old}}}(s,a)] \geq \tau \mathbb{E}_{d^{\pi_{\text{new}}}}[\mathbb{E}_{\pi_{\text{old}}}[\log \pi_{\text{old}}(\cdot \mid s)] - \mathbb{E}_{\pi_{\text{new}}}[\log \pi_{\text{old}}(\cdot \mid s)]].$$

▷ adding $-\tau \mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[\log \pi_{\text{old}}(\cdot \mid s)]$ to both sides

From Gibbs's inequality, the entropy of any distribution is strictly less than the cross-entropy of that distribution with any other distribution. In other words,

$$\mathcal{H}(\pi_{\text{new}}(\cdot \mid s)) \leq \mathcal{H}(\pi_{\text{new}}(\cdot \mid s), \pi_{\text{old}}(\cdot \mid s))$$

$$-\mathbb{E}_{\pi_{\text{new}}}[\log \pi_{\text{new}}(\cdot \mid s)] \leq -\mathbb{E}_{\pi_{\text{new}}}[\log \pi_{\text{old}}(\cdot \mid s))]$$

Applying this inequality,

$$\begin{aligned}
\mathbb{E}_{d^{\pi_{\text{new}}},\pi_{\text{new}}}[A_\tau^{\pi_{\text{old}}}(s,a)] &\geq \tau \mathbb{E}_{d^{\pi_{\text{new}}}}[\mathbb{E}_{\pi_{\text{old}}}[\log \pi_{\text{old}}(\cdot \mid s)] - \mathbb{E}_{\pi_{\text{new}}}[\log \pi_{\text{new}}(\cdot \mid s)]] \\
&= \tau \mathbb{E}_{d^{\pi_{\text{new}}}}[\mathcal{H}(\pi_{\text{new}}(\cdot \mid s)) - \mathcal{H}(\pi_{\text{old}}(\cdot \mid s))] \\
&\geq 0 \\
&\quad \triangleright \text{ by the entropy assumption of the Corollary}
\end{aligned}$$

□

The entropy assumption in Corollary 1 is a bit strange; for π_{new} to be better, it must "commit" less to actions, the opposite of what we would expect a policy to do when learning optimal actions.

3.3 Summary

Here are some takeaways of the above discussion.

1. The RKL has a stronger policy improvement result than the FKL as it requires only that the RKL of π_{new} be no greater than the RKL of π_{old}.

2. The FKL can fail to induce policy improvement, but this failure is linked to a temperature that is too low and/or an insufficient reduction in the FKL. For a large enough reduction and large enough temperature, coupled with an additional entropy assumption, policy improvement follows.

There are some limitations of this theory. We had to assume a finite action space for Proposition 3. The reason here was to ensure that we could sensibly manipulate the remainder terms in the Taylor expansions. A general treatment of action spaces would at least require some work into the measurability and integrability of such terms as a function of the action. Moreover, the theory developed in this chapter is under ideal settings, assuming in particular access to the true value functions. For Proposition 3 and corollary 1, the looseness of the bounds we employed resulted in extremely strong sufficient conditions for FKL policy improvement. As we will see in our experiments, the FKL is often able to induce policy improvement in practice, suggesting a gap between the theory developed and the practical performance.

Chapter 4

Microworld Experiments

The goal in this section is to understand differences between FKL and RKL in terms of (1) the loss surface and (2) the behaviour of iterates optimized under the losses. By behaviour, we mean whether the iterates reach multiple local optima, how stable iterates under that loss are, and how often iterates reach the global optimum (or optima). Given the fine-grained nature of our questions, we focus upon small-scale environments, which we call *microworlds*. Doing so allows us to avoid any possible confounding factors associated with larger, more complicated environments, and furthermore allows us more fully to separate any issues to do with stochasticity.

We begin with continuous actions, and note results for discrete actions in Section 4.4. We use two types of low-dimensional microworlds to allow us to visualize and thoroughly investigate behavior.

Our first microworld is a **Bimodal Bandit** in Figure 4.2. For continuous actions, we designed a continuous bandit with action space $[-1, 1]$ and reward function $Q(a) := \exp\big(-\frac{1}{2}(\frac{2a+1}{0.2})^2\big) + \frac{3}{2}\exp\big(-\frac{1}{2}(\frac{2a-1}{0.2})^2\big)$. The two unequal modes at -0.5 and 0.5 enable us to test the mean-seeking and mode-seeking behavior as well as simulate a realistic scenario where the agent's policy parameterization (here, unimodal) cannot represent the true distribution (bimodal). For discrete actions, we designed a discrete bandit with rewards $(1, 1.5)$.

Our second microworld is the **Switch-Stay** domain in Figure 4.1. From s_0, action 0 (stay) gives a reward of 1 and transitions to state 0. From s_1, action 0 gives a reward of 2 and transitions to s_1. From s_0, action 1 (switch) gives a

reward of -1 and transitions to s_1, while action 1 from s_1 gives a reward of 0 and transitions to s_0.

To adapt this environment to the continuous action setting, we treat actions > 0 as switch and actions ≤ 0 as stay. We set $\gamma = 0.9$ to ensure that the optimal action from s_0 is to switch, which ensures the existence of a short-term/long-term trade-off inherent to realistic RL environments.

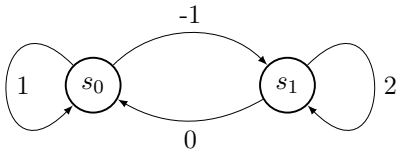


Figure 4.1: Switch-Stay.

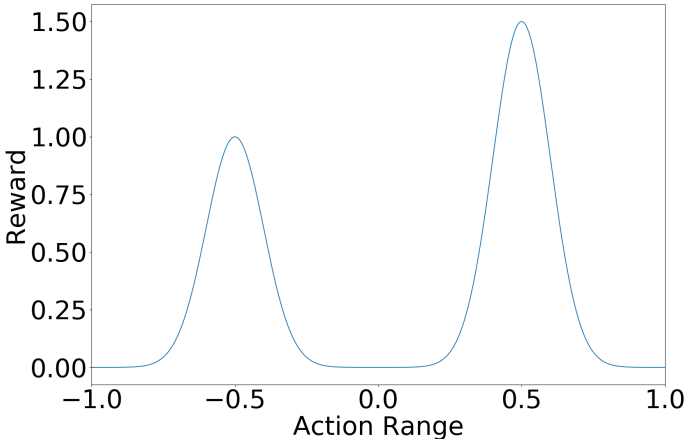


Figure 4.2: Continuous-action Bimodal Bandit.

4.1 Common Implementation Details

All policies are tabular in the state. To calculate the FKL and RKL for the continuous action setting, we use the Clenshaw-Curtis (Clenshaw and Curtis, 1960) numerical integration scheme with 1024 points from the package quadpy (Schlömer, n.d.), excluding the first and the last points at -1 and 1 because of numerical stability. Numerical integration is used to minimize any confounding

influence of stochasticity on the behaviour, but we also note additional results for Monte Carlo integration in Section 4.5.

To calculate the Hard FKL, we use the true maximum action as determined by the environment. In the discrete action setting, we may calculate the non-hard FKL losses in closed form by summing across all actions. For Switch-Stay, we calculate and optimize the mean KL across the two states.

We use the true action-values when calculating the KL losses; in the bimodal bandit, the action-value is given by the reward function, while in Switch-Stay it is calculated (i.e., not learned). For policy parameterizations, in continuous action settings we use a Gaussian policy with mean and variance learned as $(\hat{\mu}, \log(1 + \exp(\hat{\sigma}))$ and in discrete action settings we use a softmax policy. The action sampled from the learned Gaussian is passed through tanh to ensure that the action is in the feasible range $[-1, 1]$ and to avoid the bias induced in the policy gradient when action ranges are not enforced (Chou et al., 2017).

Finally, we use the RMSprop optimizer (Tieleman and Hinton, 2012). Overall trends for Adam (Kingma and Ba, 2015) were similar to those for RMSprop, while results for SGD resulted in slower learning for both FKL and RKL and a wider range of limit points, most likely due to oscillation from the constant step-size. We focus on RMSprop here to avoid any confounding factors associated with momentum.

4.2 Continuous Action Results in the Bimodal Bandit

Firstly, we might expect the FKL to have a smoother loss surface. Given that policies often are part of an exponential class (e.g., softmax policy), having the policy π be the second argument of $\mathrm{KL}(p \parallel q)$ might be advantageous as $\mathrm{KL}(p \mid \pi)$ removes the exponential in π.

4.2.1 Loss Surface

We visualize the KL loss surfaces in Figure 4.3 with five different temperatures. The heatmaps depict the loss for each mean and standard deviation pair. The

last row depicts the target distribution over which the KL loss is optimized. The surfaces suggest the following.

1) The FKL surface has a single valley, while the RKL surface has two valleys that are separated from one another. In this sense, the FKL surface seems much smoother than the RKL surface, suggesting that iterates under the FKL will more likely reach the global optimum than iterates under the RKL, which seem likely to fall into either of the valleys.

2) The smoothness of the RKL landscape increases with temperature as the gap between the peaks becomes less steep. A higher temperature also causes the valley in the FKL map to become less sharply peaked, and for the optimal μ to move closer to 0. The optimal μ for the FKL seems to move more quickly to zero, as τ increases, than the optimal μ for the RKL, although both eventually reach 0. It is possible that the FKL may become suboptimal sooner than the RKL as τ increases.

3) It may seem strange that two valleys exist for the RKL at $\tau = 0$ given that the target distribution is unimodal. Note, however, that when $\tau = 0$, the loss function is no longer a distributional loss; that is, we are no longer minimizing any pseudo-distance between the policy and a distribution.

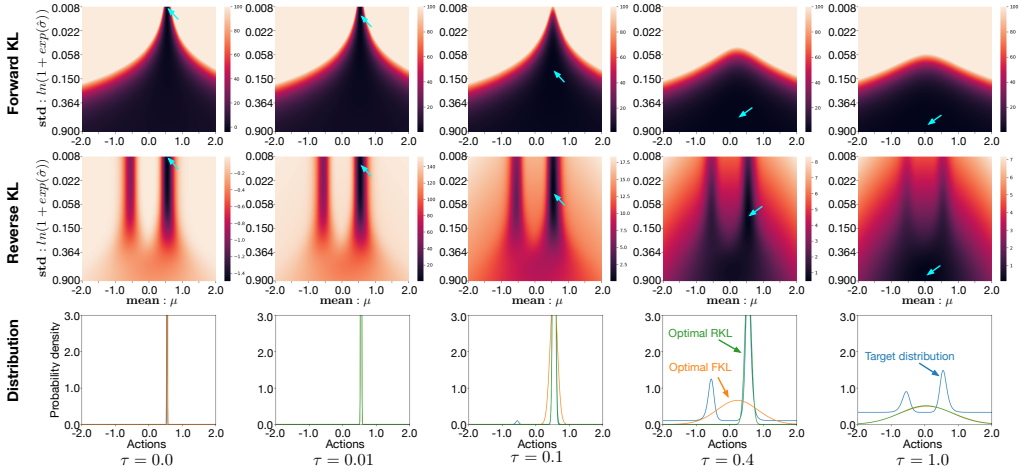


Figure 4.3: KL loss over mean and standard deviation across temperature. Note that the actual action taken applies tanh to the samples of the resulting distribution (i.e., the optimal mean is at $\tanh^{-1}(0.5) \approx 0.55$). FKL loss has been upper-bounded for better visualization of minima. Arrows indicate the global minimum.

4.2.2 Behaviour

To confirm whether our intuitions about the smooth loss surface result in iterates that reach the global optimum, we visualize 1000 random (mean, standard deviation) iterates over 1000 gradient steps to minimize either the FKL or RKL. The mean is initialized uniformly in $(-0.95, 0.95)$ and $\hat{\sigma}$ is initialized uniformly in $(\log(\exp(0.1) - 1), \log(\exp(1) - 1))$, so that the initial standard deviation σ_0 is in $(0.1, 1)$. We only show one learning rate, but results are similar for different learning rates. From looking at Figures 4.4a and 4.4b, we observe the following.

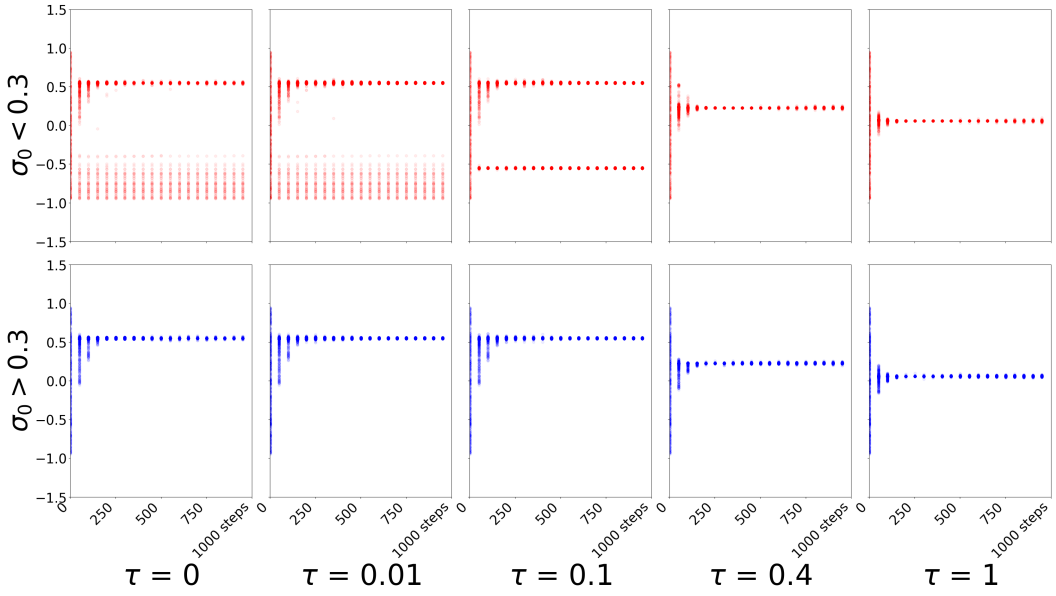
1) FKL seems to facilitate more stable convergence of the iterates to the global optimum when $\sigma_0 > 0.3$. Indeed, outside of that setting, all iterates converge to a single optimum across all temperatures.

2) Behaviour can vary across different standard deviation initializations. For $\tau < 0.4$, FKL iterates learn the optimal mode for all settings except when the $\sigma_0 < 0.3$. The achieved limit points of RKL iterates can differ depending on σ_0, such as with $\tau = 0.4$.

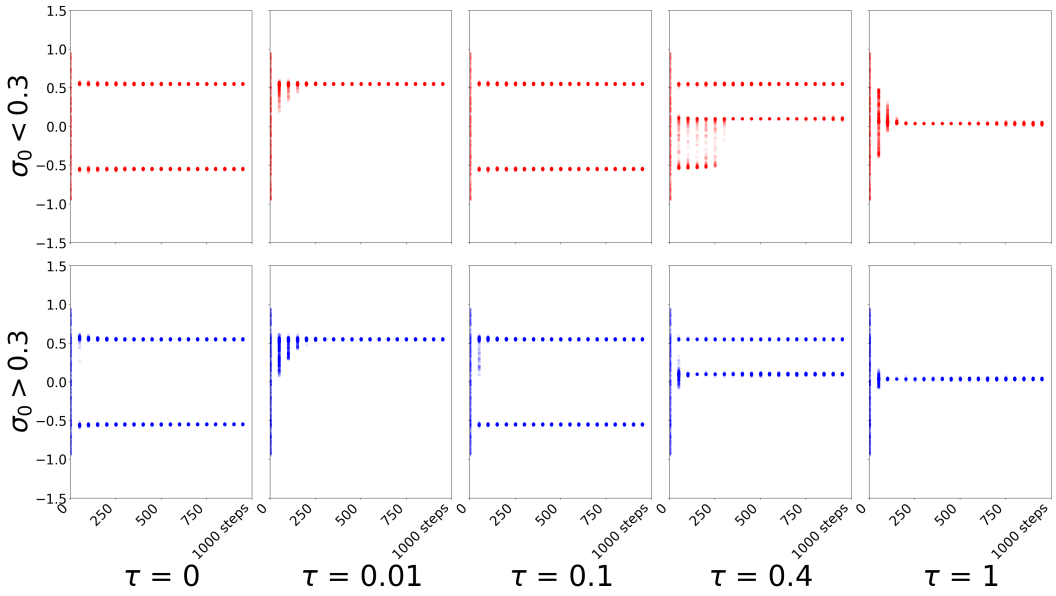
3) RKL iterates converge to different local optima. The only case in which RKL iterates only converged to the optimal mode was for $\tau = 0.01$.

4) At $\tau = 0.4$, FKL iterates converged to a point closer to zero than the RKL points, some of which remained at 0.5. This observation is consistent with our visualization of the loss surface, where the global optimum of the FKL loss moved to 0 more quickly than did the global optimum of the RKL loss.

These results are generally consistent with the FKL and RKL heatmaps, suggesting that for large enough σ_0, the FKL has a much smoother optimization landscape that directs iterates to the global minimum, but the global minimum might be suboptimal with respect to the original target distribution. One discrepancy with the visualization of the loss surfaces, however, is that both RKL and FKL exhibited limit points that do not appear to correspond to any optima on the heatmaps. In particular, FKL with $\sigma_0 < 0.3$ and $\tau \leq 0.1$ have iterates that stay in a neighbourhood away from 0, while the RKL at $\tau = 0.4$ and $\sigma_0 < 0.3$ has iterates that converge to 0. The same phenomenon was



(a) Forward KL.



(b) Reverse KL.

Figure 4.4: Each plot tracks the mean over 1000 gradient steps of each of 1000 iterates. Each iterate is represented as a translucent, coloured dot with alpha value 0.01. Temperature is varied on the x-major-axis and initial standard deviation σ_0 is varied on the y-major axis. Colour-coded by σ_0. Learning rate is 0.01.

also present with both smaller and larger learning rates, and with RMSprop and SGD. While our initial surprise at the RKL limit point may be due to a limitation of our heatmaps–indeed, 0 might be a local minimum that is hard

to distinguish amidst the dark area of Figure 4.3–the FKL limit point is more puzzling. The area around the line $\sigma = 0.3$ in Figure 4.3 for $\tau = 0.01$ seems not to contain a local optimum.

An earlier version of this experiment used integration points within the range $[-0.98, 0.98]$, rather than using all points but {-1, 1}. In this earlier setting, RKL iterates often diverged for $\sigma_0 > 0.3$ while FKL iterates maintained the same behaviour, suggesting that FKL is more robust to this type of bias than RKL. We comment more on the effect of numerical integration error in Section 4.5.

4.3 Switch-Stay

Given our previous results, we might expect the FKL to perform better. If FKL iterates can reach the global optimum more easily, surely those policies would be better than policies updated under the RKL? One important factor, which we explore further below, is that the global minimum of the FKL objective may not correspond well with the optimal solution of the original, unregularized objective.

We investigate the behavior of iterates when there is more than one state: the Switch-Stay environment. As a simple instantiation of the full RL problem, we are interested in understanding any possible differences between FKL and RKL in the presence of short-term/long-term trade-offs. In particular, on Switch-Stay, from state 0 one should incur a short-term penalty by switichng to state 1 to maximize return, given that $\gamma = 0.9$.

Another reason why we selected the Switch-Stay environment is for ease of visualization. Since the MDP has only two states, one can plot any value function as a point on a 2-dimensional plane. In particular, one can view the entire space of value functions, shown recently to be a polytope in the discrete-action setting (Dadashi et al., 2019). Plotting the value function corresponding to a policy as it changes over time allows us to view the progress of an algorithm and avoids the stochasticity inherent in performing rollouts and plotting learning curves.

To plot the boundary of the value function polytope in Switch-Stay, we plot the value functions corresponding to semi-deterministic policies (i.e., policies that are deterministic in at least one state) (Dadashi et al., 2019). Given a policy π, the value function V^π is calculated exactly as $(I-\gamma P_\pi)^{-1} r_\pi$, where P_π and r_π are respectively the transition matrix and the reward function induced by π.

In the continuous version of switch-stay, we treat any action ≤ 0 as stay, and any action > 0 as switch. To calculate the value function corresponding to a continuous policy π, we convert π to an equivalent discrete policy π_{discrete} of the underlying discrete MDP. The conversion requires the calculation of the probability that π outputs an action ≤ 0 in each state, which we do with numerical integration of the policy PDF. We then calculate the value function of π as $(I - \gamma P_{\pi_{\text{discrete}}})^{-1} r_{\pi_{\text{discrete}}}$.

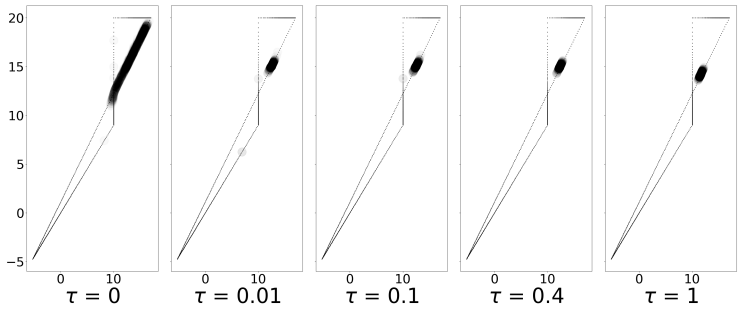
For the hard FKL, we require access to the greedy action of the action-value function. In the continuous-action setting, this greedy action is usually infeasible to obtain. For the purposes of this experiment, if the greedy action is stay, we represent it in $[-1, 1]$ by drawing a uniform random number from $[-1, 0]$. If the greedy action is switch, we represent it as a uniform random number in $[0, 1]$. This design choice is meant to simulate noisy access to the greedy action in practice.

As the policy is tabular in the states, we simply use the mean KL across states as the loss.

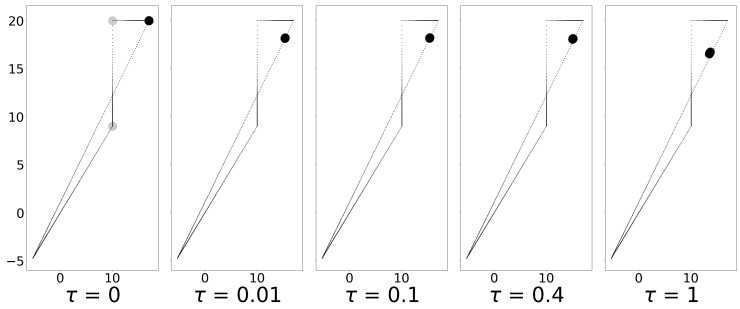
4.3.1 Behaviour

For all of these experiments, we initialized means in the range $(-0.95, 0.95)$. All experiments are run for 500 gradient steps and each experiment has 1000 iterates. We plot the value function of the final policy for each iterate and experiment in Figures 4.5 and 4.6, by visualizing the value function polytope (Dadashi et al., 2019).

1) FKL with $\tau = 0$ converged noticeably slower than the other temperatures, which seems to be an artifact of our encoding of continuous actions to the underlying discrete dynamics of switch-stay, and the fact that we used random



(a) Forward KL, learning rate = 0.005.



(b) Reverse KL, learning rate = 0.005.

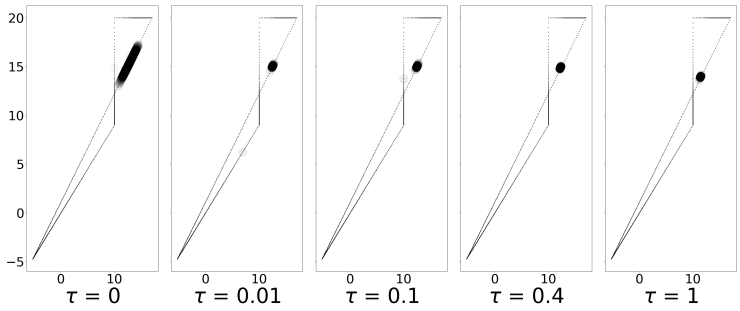
Figure 4.5: Each subplot plots the final value functions on the continuous version of switch-stay after 500 gradient steps with $\gamma = 0.9$ for 1000 iterates. The top-right corner of the polytope in each subplot is the optimal value function and the bottom-left corner is the pessimal value function. Each iterate is represented by a translucent dot with alpha value 0.01. Using RMSprop. Temperature is varied on the x-major-axis.

tie-breaking when computing the arg max for hard FKL.

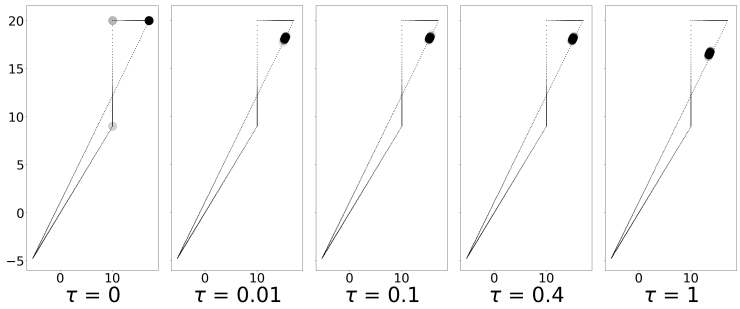
2) RKL iterates converge slightly faster than FKL iterates across all temperature settings. RKL iterates with $\tau = 0$ sometimes converged to non-optimal value functions on the corners.

3) The limiting value functions of the FKL iterates seem more suboptimal than the limiting value functions of the RKL iterates; the latter are closer to the optimal value function of the original MDP. This result is consistent with our observations in the continuous bandit; although the FKL optimum may be more easily reached, that optimal point may be suboptimal with respect to an "original" objective (in this case, the optimal value function of the unregularized MDP).

To further our understanding of **3)**, we plot the final means and standard deviations of the learned policies for the learning rate of 0.01. In Figures 4.7



(a) Forward KL, learning rate = 0.01.



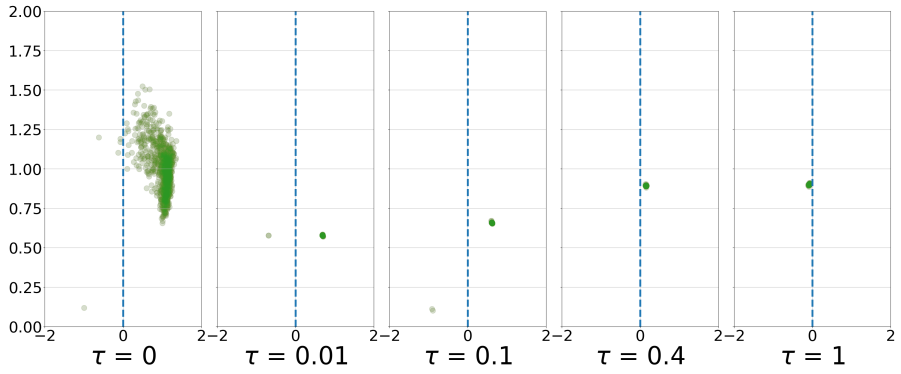
(b) Reverse KL, learning rate = 0.01.

Figure 4.6: See Figure 4.5.

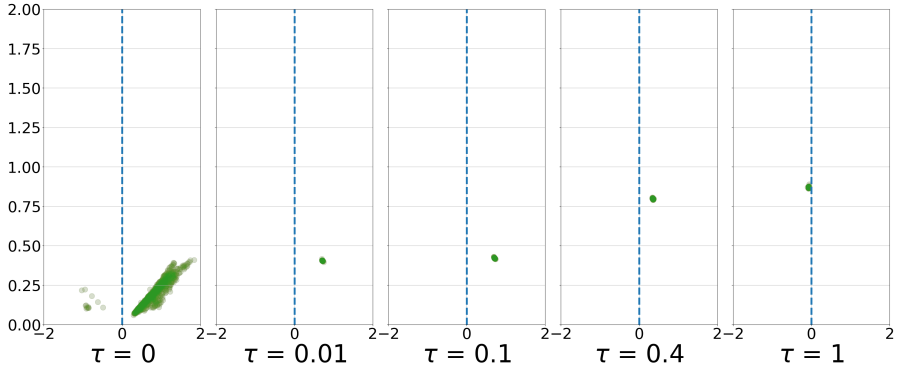
and 4.8, we see that the final FKL iterates have higher standard deviation, meaning that the final policies are further from the optimal deterministic policy of the unregularized MDP. Put informally, the FKL tends to "commit" less than the RKL.

In Figures 4.7 and 4.8 as well, we note that the spread of the final iterates is much larger for the leftmost column, representing $\tau = 0$, than for any of the other columns. This spread is consistent with the spread observed in the leftmost columns of Figures 4.5 and 4.6. For the RKL, this spread seems to be a representation of the local minima. For the FKL, this result is likely an artifact of our encoding of the maximum action as a uniform random point in either [0, 1] or [-1, 0], depending on if the maximum action is respectively stay or switch.

We also note here that when fewer integration points were used in an earlier version of this experiment, RKL exhibited substantial instability and a large number of iterates across temperatures and σ_0 values converged to suboptimal deterministic policies. This phenomenon might be related to the



(a) Forward KL on state 0.



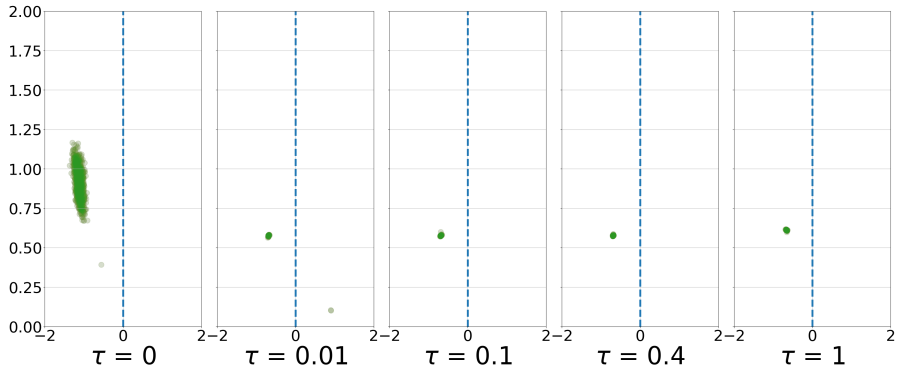
(b) Reverse KL on state 0.

Figure 4.7: Each subplot plots the final mean (x-axis) and standard deviation (y-axis) on the continuous version of switch-stay after 500 gradient steps with $\gamma = 0.9$ for 1000 iterates. Each iterate is represented by a translucent dot with alpha value 0.1. Using RMSprop with learning rate 0.01. Temperature is varied on the x-major-axis. Every action ≤ 0 is treated as "stay" and every action > 0 is treated as "switch". The blue dotted line in subplots corresponds to the line $\mu = 0$.

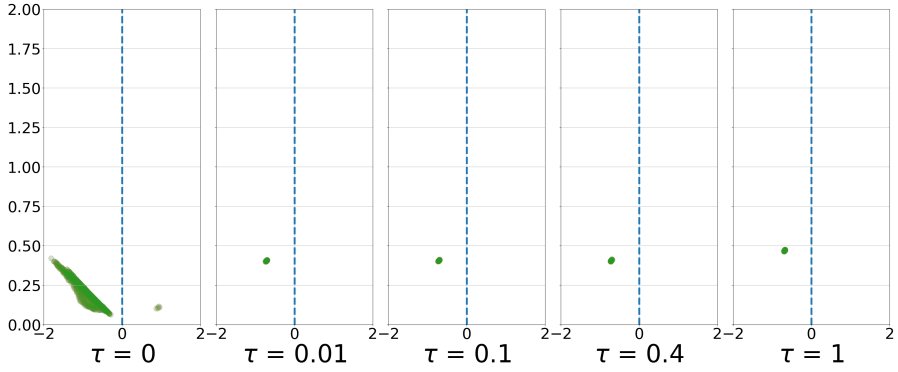
effect of stochasticity when approximating the KL loss, which is discussed in Section 4.5.

4.4 Discrete-Action Results

Overall, there is markedly less distinction between RKL and FKL in the discrete action setting with a softmax policy parameterization. In the heatmap in Figure 4.9, the loss surfaces for both KLs are very similar for a given temperature, in contrast to the heatmap for the continuous bandit. As the temperature increases, the black region (region with low loss) moves closer



(a) Forward KL on state 1.



(b) Reverse KL on state 1.

Figure 4.8: See Figure 4.7.

to the middle of the plot, which is consistent with the fact that the target distribution becomes closer to uniform as the temperature increases.

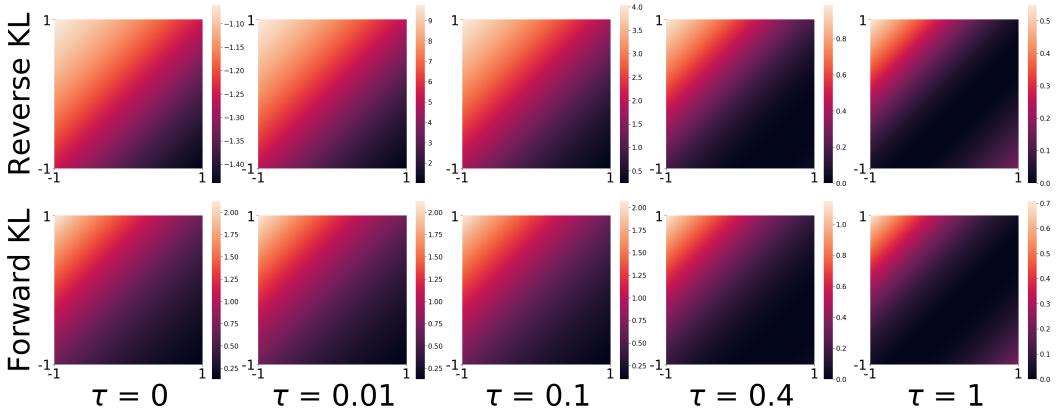
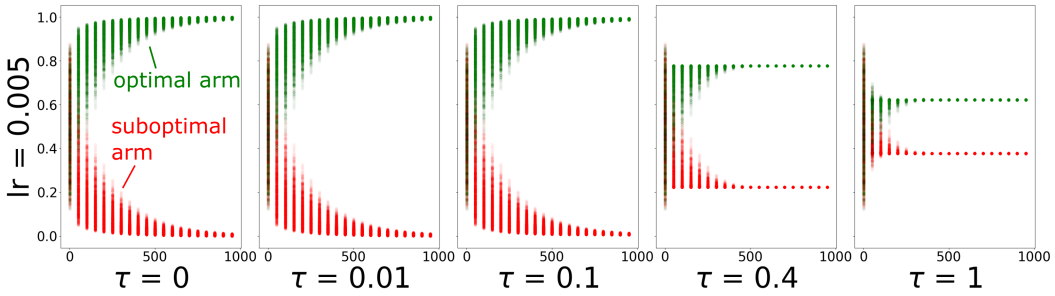


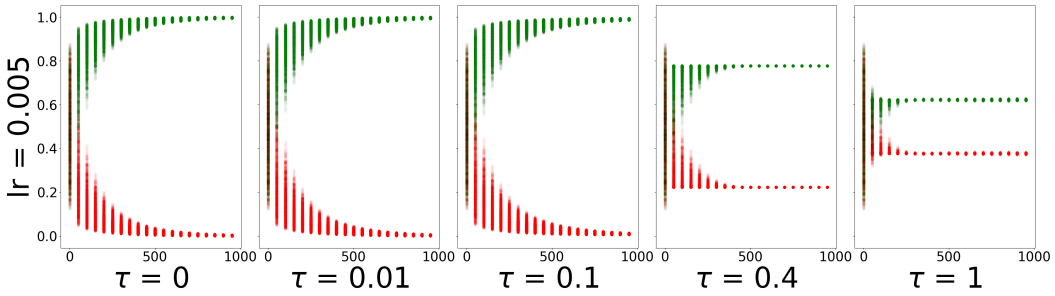
Figure 4.9: Heatmap for the KLs on the discrete bandit. In a given subplot, the x-axis is the logit for the optimal arm and the y-axis is the logit for the suboptimal arm.

Next, we track the progress of 1000 iterates over 1000 gradient steps. For each iterate, we initialize each logit–one for each arm–uniformly in $(-1, 1)$. The behaviour policy is thus the result of the softmax function applied to the logit vector.

When looking at 1000 iterates over 1000 gradient steps, both RKL and FKL iterates learn the optimal arms in Figures 4.10a and 4.10b. There is little difference in the behaviour of iterates under either KL; there seems just to be one global optimum to which iterates under either KL converge reliably.



(a) Forward KL.



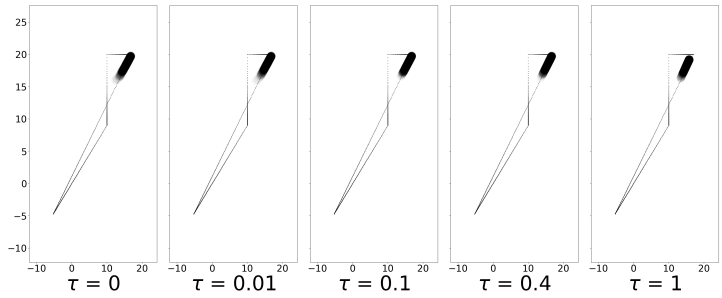
(b) Reverse KL.

Figure 4.10: Each subplot tracks the learned probability of each arm for 1000 iterates over 1000 gradient steps. The learning rate is set to be 0.005.

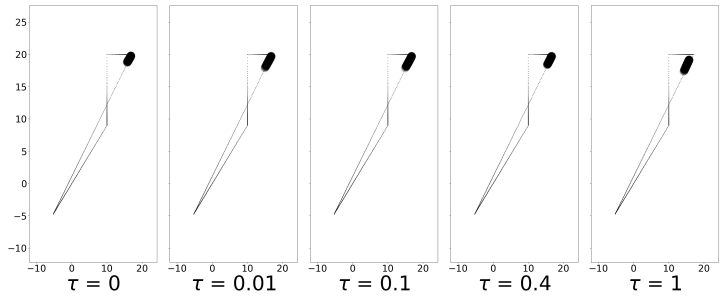
Results for discrete Switch-Stay are in Figure 4.11. Both FKL and RKL iterates move in the direction of the optimal value function, but RKL iterates seem to converge faster. While a difference in convergence speed did also exist on the continuous version of Switch-Stay, the distinction is less notable here.

For higher temperatures, the limit point of the iterates is slightly further away from the optimal value function of the original MDP than for lower temperatures. This result is to be expected given that in general, the optimal entropy-regularized policy is different from the optimal non-regularized policy

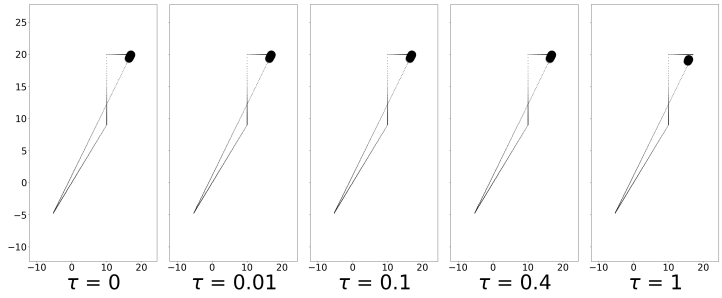
(Geist et al., 2019).



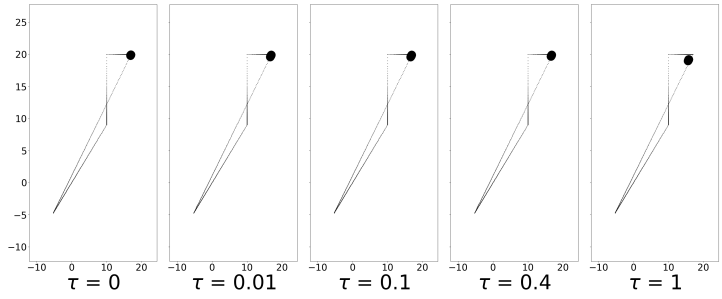
(a) Forward KL, learning rate = 0.005.



(b) Reverse KL, learning rate = 0.005.



(c) Forward KL, learning rate = 0.01.



(d) Reverse KL, learning rate = 0.01.

Figure 4.11: Final value functions on discrete version of switch-stay after 500 gradient steps, $\gamma = 0.9$. Using RMSprop for learning rates $\in \{0.005, 0.01\}$.

4.5 The Impact of Stochasticity

Although with discrete actions it is practical to sum across all actions when calculating the KL losses, difficulty emerges with high-dimensional continuous action spaces. Quadrature methods scale exponentially with the dimension of the action-space, leaving methods like Clenshaw-Curtis impractical; Monte-Carlo integration seems the only feasible answer in this setting.

We repeated the continuous-action microworld experiments to understand any differences induced by using Monte-Carlo integration (i.e., sampling actions from the current policy) instead of Clenshaw-Curtis quadrature to calculate the losses.

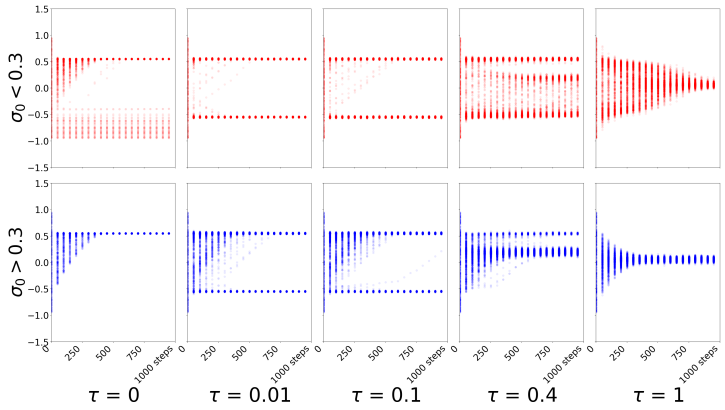
Since Hard FKL only depends upon the maximum action, we do not modify the algorithm in this regime. Hard and soft RKL are modified to use sampled actions from the current policy π. To derive a sampling-based version of soft FKL, we use weighted importance sampling to approximate the integral $-\int_{\mathcal{A}} \mathcal{B}Q(s,a) \log \pi(a \mid s)\, da$, noting that the omitted term $\mathcal{H}(\mathcal{B}Q(\cdot \mid s))$ does not depend on π. In particular, for samples $\{a_i\}_{i=1}^n \sim \pi(\cdot \mid s)$ we estimate

$$-\int_{\mathcal{A}} \mathcal{B}Q(s,a) \log \pi(a \mid s)\, da \approx \sum_{i=1}^{n} \rho_i \log \pi(a_i \mid s),$$
$$\rho_i := \frac{\tilde{\rho}_i}{\sum_{j=1}^{n} \tilde{\rho}_j}$$
$$\tilde{\rho}_i := \frac{\exp(Q(s,a_i)\tau^{-1})}{\pi(a_i \mid s)}.$$

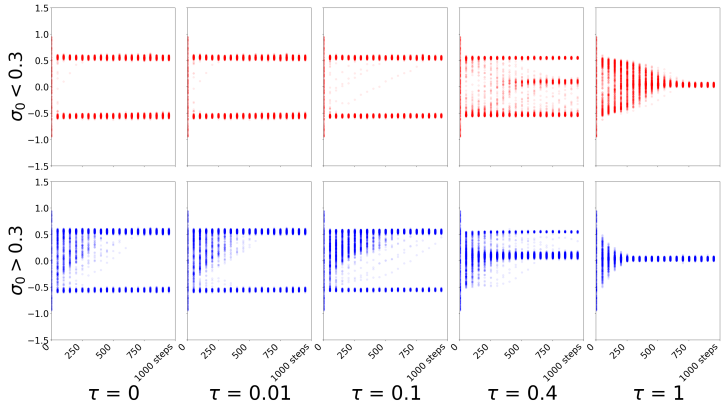
Note that we do not have to estimate $\mathcal{H}(\mathcal{B}Q(s,\cdot))$, the other term in the FKL, as it does not depend on π. Weighted importance sampling lets us avoid calculating the partition function in $\mathcal{B}Q$, and may be a fruitful avenue to explore in making FKL practical for high-dimensional action spaces.

While overall behaviour for the continuous bandit is consistent with the results using Clenshaw-Curtis quadrature, the iterates seem to cluster more slowly around their limit points, or not at all.

On Switch-Stay, more notable differences emerged. RKL iterates converged to minima to which they did not converge in the Clenshaw-Curtis regime, even for larger numbers of sample points. In Figure 4.15b, there is an interesting



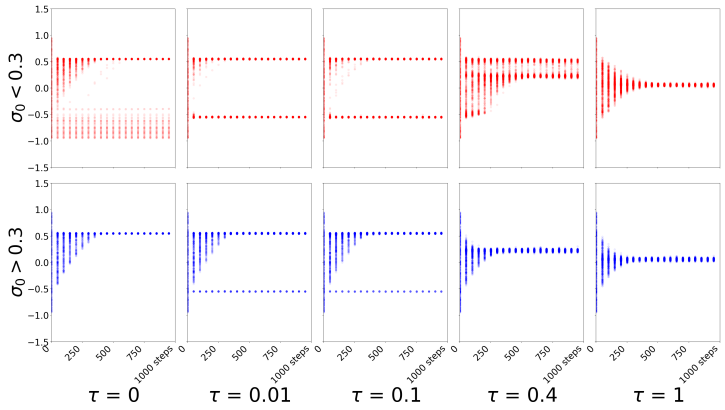
(a) Forward KL.



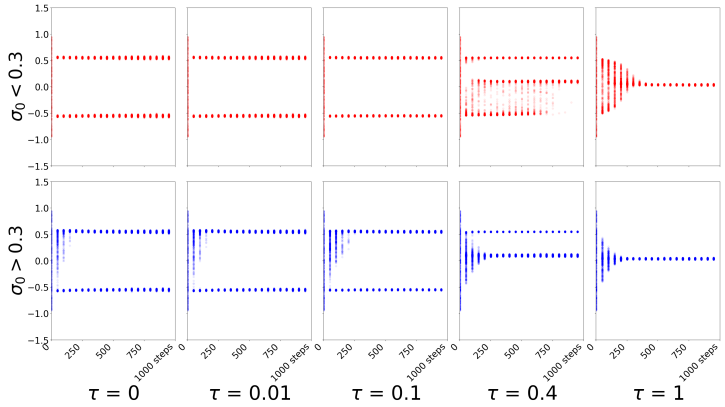
(b) Reverse KL.

Figure 4.12: Continuous bandit with 10 sample points, learning rate = 0.005, with RMSprop.

trend across temperatures. Temperatures below 0.4 induced many suboptima far from the optimal value function, while temperatures 0.4 and 1 seemed better at clustering RKL iterates near the optimal value function. On the other hand, FKL seemed relatively insensitive both to the temperature and the number of sample points. This relative insensitivity could be due to having a smoother loss landscape to begin with, which tends to direct iterates to a single global optimum. Nevertheless, that global optimum was often quite suboptimal with respect to the unregularized MDP, especially since many RKL iterates were much closer to the optimal policy of the unregularized MDP.



(a) Forward KL.



(b) Reverse KL.

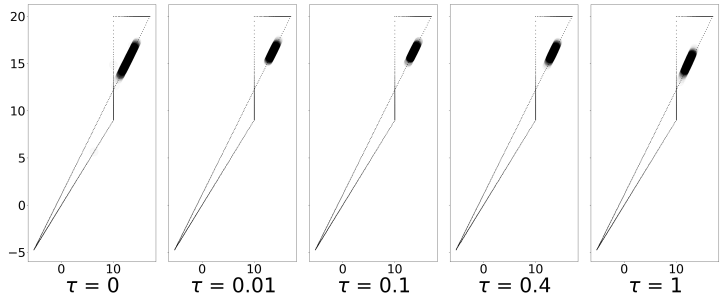
Figure 4.13: Continuous bandit with 500 sample points, learning rate = 0.005, with RMSprop.

4.6 Summary

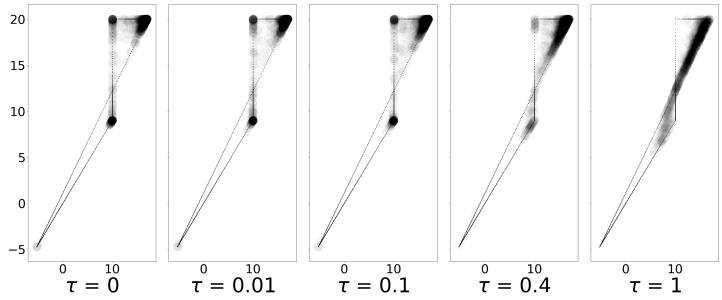
Given the amount of data in this chapter, it is helpful to summarize our discussion thus far.

1. The relative difference between the KLs in continuous-action settings is larger than that in discrete-action settings.

2. On our bimodal bandit, the FKL was better at directing iterates to the global optimum, excepting some iterates that stayed near the suboptimal mode when the initial standard deviation was set sufficiently low. However, as the temperature increased, the global optimum of the FKL seemed to approach a suboptimal point with respect to the bandit reward function faster than the global optimum of the RKL did.

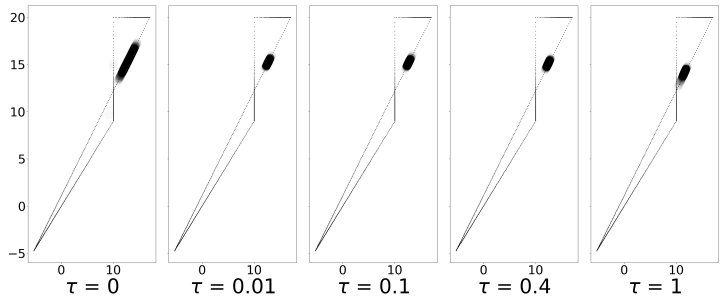


(a) Forward KL.

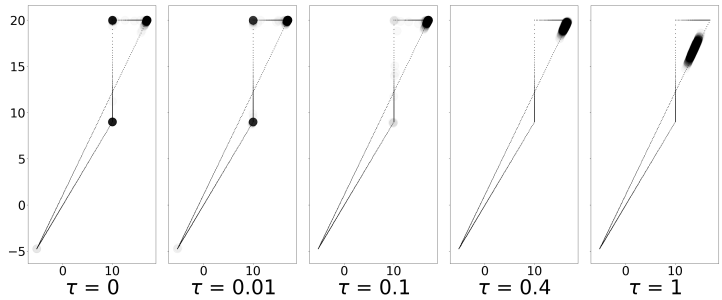


(b) Reverse KL.

Figure 4.14: Switch-stay with 10 sample points, learning rate = 0.01, with RMSprop.



(a) Forward KL.



(b) Reverse KL.

Figure 4.15: Switch-stay with 500 sample points, learning rate = 0.01, with RMSprop.

3. On the continuous version of Switch-Stay, the FKL iterates converged more slowly than the RKL iterates and to a policy that was less optimal, which seemed primarily due to a larger final standard deviation for the FKL iterates. The increased suboptimality of the FKL solution is consistent with our bandit results.

4. In the continuous-action setting, stochastic sampling of the loss function impacted RKL more than FKL. RKL tended to exhibit more spurious minima in this setting, even when increasing the number of sample points in the calculation of the loss.

5. In the discrete-action setting, little significant difference emerged between FKL and RKL. Both converged to nearly identical optima, although the convergence of iterates under FKL seemed slightly slower.

We suspect that the differences here are heavily due to the policy parameterization. One reason for the difference in the number of minima seems to be that a Gaussian policy is unimodal, whereas a softmax policy can represent multimodal distributions. As well, given that the behaviour in the bandit setting depended on the initial standard deviation, we hypothesize that the added complexity of learning the standard deviation contributes to the differences we observed between continuous and discrete actions.

We also did not learn action-values in our experiments. Especially given the additional differences between the KLs that we observed with stochasticity, using learned action-values might drastically change our results. In this setting, the accuracy of the action-value estimate would depend on the actions taken by the agent, which in turn are affected by the process of policy optimization. For instance, if an agent tends to commit to an action at a given state and not try other actions, the resulting action-value estimate could be poor. In the literature in general, more investigation into the interplay of learning value functions and policy optimization is necessary.

Chapter 5

Benchmark Results

We compare the KL methods on benchmark continuous and discrete-action environments, using non-linear function approximation. Here, we wish to understand (1) if our observations from the microworld experiments apply to more complicated environments and (2) if any new differences as a result of function approximation or increased environment complexity.

5.1 Implementation Details

5.1.1 Hyperparameters

Hyperparameter sweeps are performed separately for each domain. We sweep over the learning rates $\{10^{-5}, 10^{-4}, 10^{-3}\}$; for Pendulum, we additionally sweep over actor learning rate 10^{-2}. In the continuous-action setting, we sweep both the actor and the critic learning rate, while in the discrete-action setting we have a shared learning rate because of a shared architecture. We sweep temperatures in $\{0.01, 0.1, 1\}$ for the soft action-value methods and set the temperature in $\mathcal{B}Q$ and the temperature in the soft action-value function to be the same value. For example, if $\tau = 0.01$, then we learn a soft action-value function with $\tau = 0.01$ and use a KL target distribution proportional to $\exp(Q(s, a)\tau^{-1})$. RMSprop (Tieleman and Hinton, 2012) is used as the optimizer to be consistent with our focus in the microworld domains.

We perform 30 runs for all hyperparameter settings and plot the mean return averaged over the past 20 episodes; shaded areas represent standard errors.

5.1.2 Updating

We perform bootstrapping as per the recommendations in Pardo et al. (2018). If a transition (s, a, r, sp) results in the true termination of the episode, the bootstrap target for value function updates is r instead of the usual $r - \tau \log \pi(a \mid s) + \gamma V(sp)$. If the transition results in a timeout, $r - \tau \log \pi(a \mid s) + \gamma V(sp)$ is used. Otherwise, $r - \tau \log \pi(a \mid s) + \gamma V(sp)$ is used as the bootstrap target.

On the continuous-action domains we use Clenshaw-Curtis (Clenshaw and Curtis, 1960) numerical integration scheme to estimate the FKL and the RKL. To generate the points for Clenshaw-Curtis, we use the Python package, quadpy.[1]. For computational efficiency, we use 64 points for 1D environments, and $l = 6$ for multi-dimensional environments using the nested univariate quadrature formula for Clenshaw-Curtis (Gerstner and Griebel, 1998). In learning value functions we learn both $Q(s, a)$ and $V(s)$, using the same target as done in SAC (Haarnoja et al., 2018). In particular, the target for $V(s)$ is $Q(s, a) - \log(\pi(a \mid s))$. On preliminary experiments, we found that the target of $Q(s, a) - \log(\pi(a \mid s))$ performed better than the target of $r(s) - \tau \log \pi(a \mid s) + \gamma V(s')$ for all methods.

On the discrete-action domains, the value function update target is $r(s) - \tau \log \pi(a \mid s) + \gamma V(s')$ instead of the update that SAC uses, $Q(s, a) - \log \pi(a \mid s)$. Preliminary experiments indicated that the former update was superior for all methods. Both Q and V are learned.

For reference, we also include the gradient updates of the various KL

[1] `https://github.com/nschloe/quadpy`

divergences.

$$\textbf{RKL} : \nabla_\theta \mathrm{KL}(\pi_\theta(\cdot \mid s) \parallel \mathcal{B}Q_\tau(s, \cdot)) \\ = -\nabla_\theta \mathcal{H}(\pi_\theta(\cdot \mid s)) - \int_{\mathcal{A}} \nabla_\theta \pi_\theta(a \mid s) \log \mathcal{B}Q_\tau(a \mid s)\, da \tag{5.1}$$

$$\textbf{FKL} : \nabla_\theta \mathrm{KL}(\mathcal{B}Q_\tau(s, \cdot) \parallel \pi_\theta(\cdot \mid s)) \\ = -\int_{\mathcal{A}} \mathcal{B}Q_\tau(a \mid s) \nabla_\theta \log \pi_\theta(a \mid s)\, da \tag{5.2}$$

$$\textbf{Hard RKL} : -\int_{\mathcal{A}} \nabla_\theta \pi_\theta(a \mid s) Q(s, a)\, da \tag{5.3}$$

$$\textbf{Hard FKL} : -\nabla_\theta \log \pi_\theta \left(\arg\max_b Q(s, b) \mid s \right). \tag{5.4}$$

Algorithm 1 Agent for Benchmark Experiments

Given: policy π_θ (parameters θ); action-value estimate Q_v (parameters v); state-value estimate V_w (parameters w); experience replay buffer; choice of KL divergence; temperature $\tau \geq 0$; learning rates for θ, v, w; optimizer (e.g., RMSprop).
for $t = 0, \ldots,$ **do**
  Draw action $a_t \sim \pi_\theta(\cdot \mid s_t)$
  Observe and store transition $(s_t, a_t, s_{t+1}, r, \text{DoneFlag})$. Replace the oldest transition if the buffer size limit is reached.
  if samples in buffer ≥ 32 **then**
    Draw random mini-batch of size 32 from buffer.
    for each transition $(s, a, r, sp, \text{DoneFlag})$ in the mini-batch **do**
      Calculate KL gradients according to Equations (5.1) to (5.4), based on choice of τ and KL divergence.
    end for
    Update w, v with DiscreteUpdateValueFunctions if action space is discrete; otherwise call ContinuousUpdateValueFunctions.
    Update θ with the learning rate and optimizer.
  end if
end for

In all our plots, we refer to the number of "frames" on the x-axis. We use "frames" synonomously with "timestep".

5.1.3 Architecture

On our continuous-action domains, all policy and value function networks are implemented as two-layer neural networks of size 128, with ReLU activations.

Algorithm 2 DiscreteUpdateValueFunctions

Given: policy π_θ (parameters θ); action-value estimate Q_v (parameters v); state-value estimate V_w (parameters w); temperature $\tau \geq 0$; optimizers (e.g., RMSprop) for v and w; batch of data $\mathcal{D}$, with each transition written in the form $(s, a, r, sp, \text{DoneFlag})$

To update the state-value function, pass the following as the gradient to the optimizer for w.

$$-\mathbb{E}_{\mathcal{D}}[(r - \tau \log \pi(a \mid s) + \gamma \cdot (1 - \text{DoneFlag}) \cdot V_w(sp) - V(s))\nabla_w V(s)]$$

To update the action-value function, pass the following as the gradient to the optimizer for v.

$$-\mathbb{E}_{\mathcal{D}}[(r + \gamma \cdot (1 - \text{DoneFlag}) \cdot V_w(sp) - Q_v(s, a))\nabla_v Q_v(s, a)]$$

Algorithm 3 ContinuousUpdateValueFunctions

Given: policy π_θ (parameters θ); action-value estimate Q_v (parameters v); state-value estimate V_w (parameters w); temperature $\tau \geq 0$; optimizers (e.g., RMSprop) for v and w; batch of data $\mathcal{D}$, with each transition written in the form $(s, a, r, sp, \text{DoneFlag})$

For each state s in the batch $\mathcal{D}$, draw a new action $\tilde{a} \sim \pi_\theta(\cdot \mid s)$.

To update the state-value function, pass the following as the gradient to the optimizer for w.

$$\mathbb{E}_{\mathcal{D}}[(V_w(s) - Q_v(s, \tilde{a}) + \tau \log \pi_\theta(\tilde{a} \mid s))\nabla V_w(s)]$$

To update the action-value function, pass the following as the gradient to the optimizer for v.

$$-\mathbb{E}_{\mathcal{D}}[(r + \gamma \cdot (1 - \text{DoneFlag}) \cdot V_w(sp) - Q_v(s, a))\nabla_v Q_v(s, a)]$$

We use experience replay of buffer size 10^6 with batch size of 32. On our discrete-action domains, we employ the following architectures.

OpenAI Gym: The architecture is a two-layer neural network, with the policy and value functions as separate heads off of the main two-layer body. ReLU activations are used between layers. We report results for three hidden layer sizes: small (32), medium (128), and large (512).

MinAtar: The architecture is a convolutional network into one fully-connected layer for each of the policy, action value function, and state value function. The convolutional layer has 16 3x3 convolutions with stride 1, the same as in Young and Tian (2019); we vary the size of the fully-connected layer in $\{32, 128\}$. ReLU activations are used between layers.

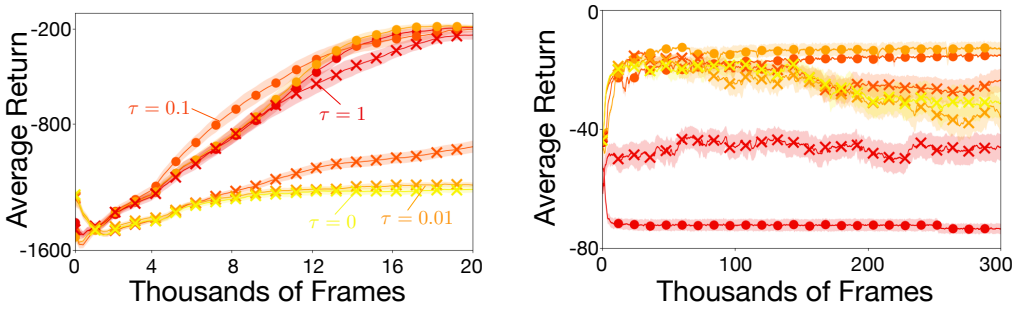
On the discrete-action domains, experience replay is used with a buffer size of 10^5 and a batch size of 32.

5.2 Continuous-Action Results

We compare agents on Pendulum (Brockman et al., 2016), Reacher, and Swimmer (Todorov et al., 2012), with results shown in Figure 5.1. We exclude Hard FKL in our comparison since it requires access to $\max_a Q(s, a)$, which is difficult to obtain with continuous actions.

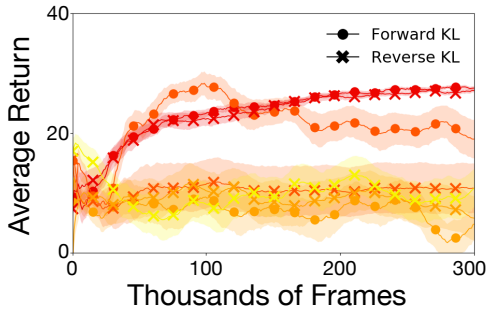
For certain temperatures, FKL either learns faster than or achieves superior final return to RKL. This difference is especially striking on Pendulum Figure 5.1a, where FKL learns vastly faster than RKL for $\tau = 0.1, 0.01$. It's interesting to note that RKL with $\tau = 1$ has a similarly-shaped learning curve to FKL with $\tau = 0.01, 0.1, 1$. In a sense, the higher temperature of RKL with $\tau = 1$ might be compensating for committal behaviour of RKL that we observed in the microworld experiments. It does seem possible to be too non-committal. On Reacher in Figure 5.1b, FKL with $\tau = 1$ completely fails to learn, while all other settings learn appreciably. Nevertheless, these observations suggest a connection between the impacts of FKL and of entropy regularization, as we discuss further below in the discrete-action setting.

It is difficult to comment on the importance of the policy parameteri-



(a) Pendulum

(b) Reacher



(c) Swimmer

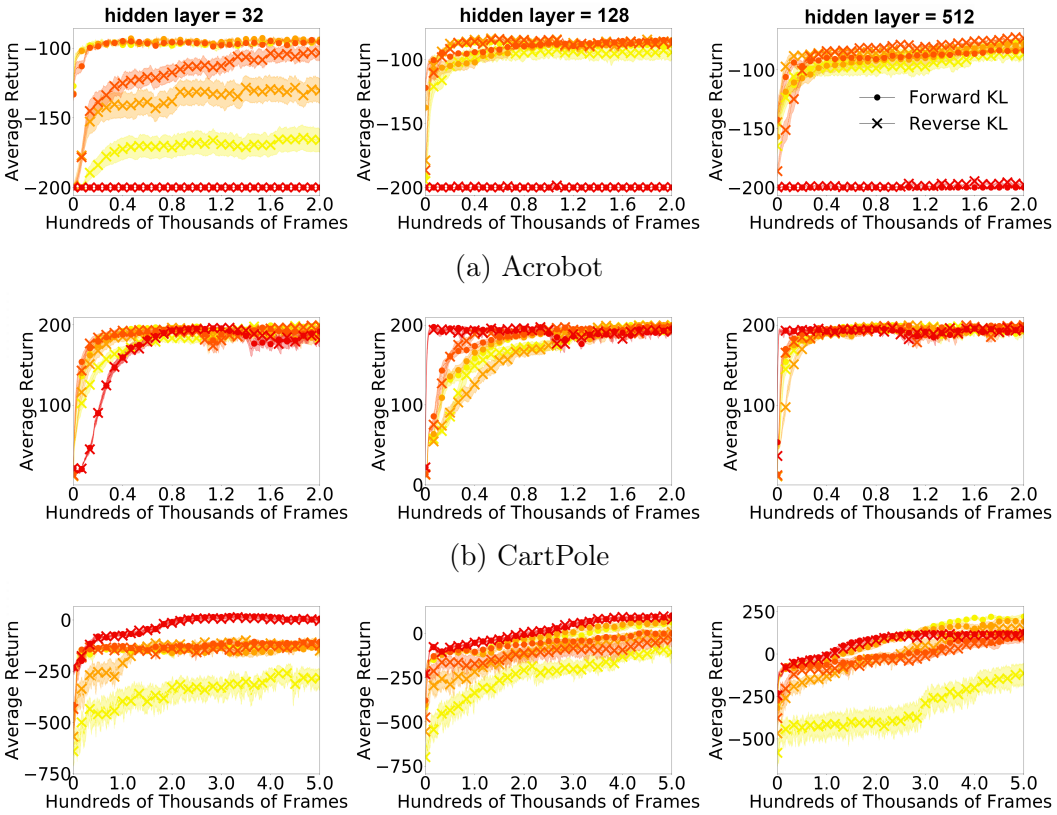
Figure 5.1: Continuous-action environments. Shown are the best hyperparameters for each algorithm, which are selected by largest area under the last half of the learning curve. The colours go from hot (red, temperature = 1) to cool (yellow, temperature = 0).

zation for these experiments relative to our microworld experiments. Any influence from the Gaussian policy parameterization is conflated with function approximation. Moreover, as we will see below, no stark pattern seems to divide continuous and discrete action settings, as one did in our microworld experiments.

5.3 Discrete-Action Results

We report results for environments from the OpenAI gym (Brockman et al., 2016) and MinAtar (Young and Tian, 2019).

The results on the OpenAI Gym environments suggest that for small hidden layer sizes, FKL may be at least as good as RKL, and sometimes better. As the hidden layer size increases, however, this dominance is negated. For a hidden



(a) Acrobot

(b) CartPole

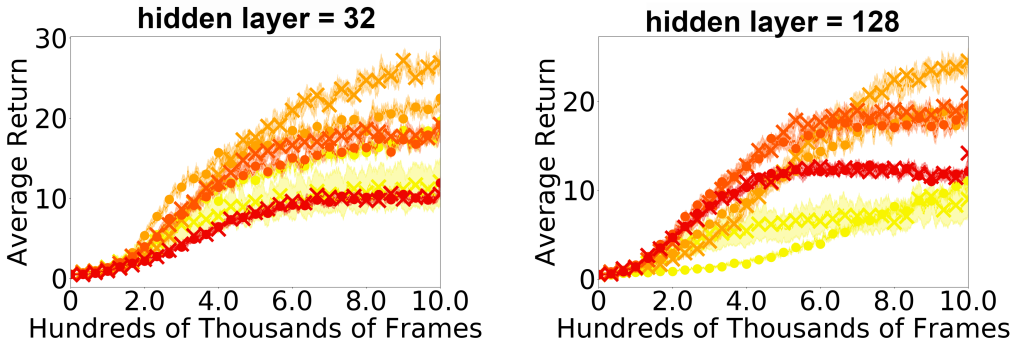
(c) Lunar Lander.

Figure 5.2: OpenAI Gym discrete-action environments. Plot settings are identical to Figure 5.1.

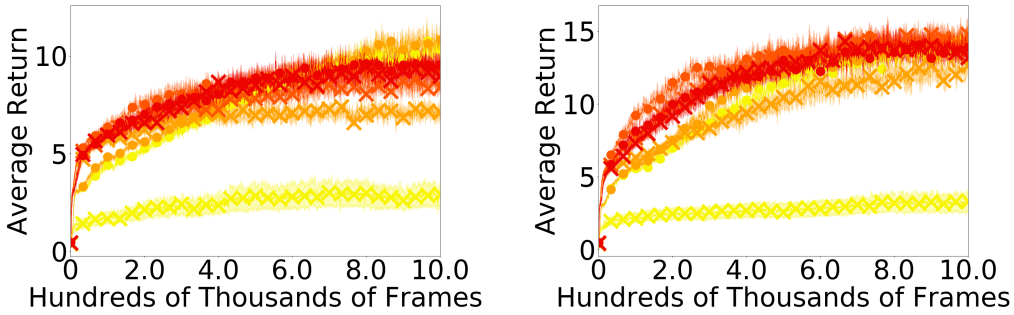
layer size of 128, RKL seems to learn a little bit faster than FKL for $\tau \neq 1$, and RKL has a slightly higher final performance for a hidden layer size of 512, although the difference does not appear to be significant. It is possible that these observations are related to how the optimization landscape of a neural network changes with larger hidden layer sizes. The influence of the choice of policy gradient objective on the optimal function approximation architecture deserves further consideration.

The superiority of FKL in Lunar Lander for $\tau = 0, 0.01$ and sometimes $\tau = 0.1$ is interesting. For these temperatures, FKL with non-zero temperature consistently matches or exceeds the corresponding learning curve for RKL. Indeed, when $\tau = 0$, a large gap separates FKL and RKL. A possible reason for this dominance is the difficulty of exploration in Lunar Lander: one must learn to manoeuvre the spacecraft, discover the correct landing area, learn how

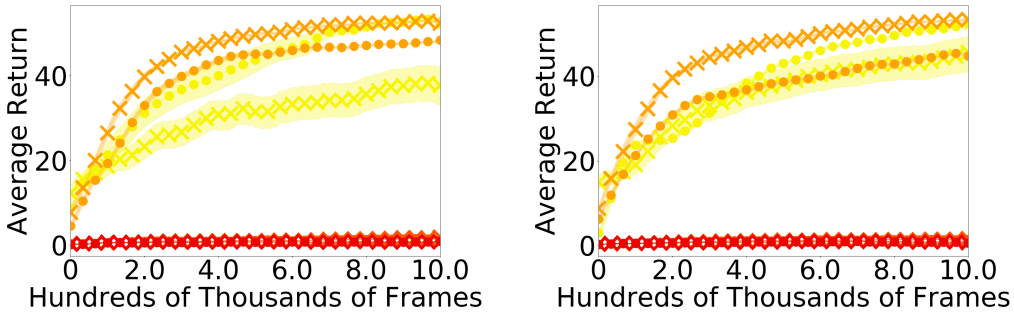
to land correctly, and learn to turn off the fuel. That FKL might "commit" less (i.e., have higher standard deviation than RKL, all other things equal) suggests that more actions are tried to facilitate discovery of good actions. We further discuss exploration below in our MinAtar experiments.



(a) Asterix



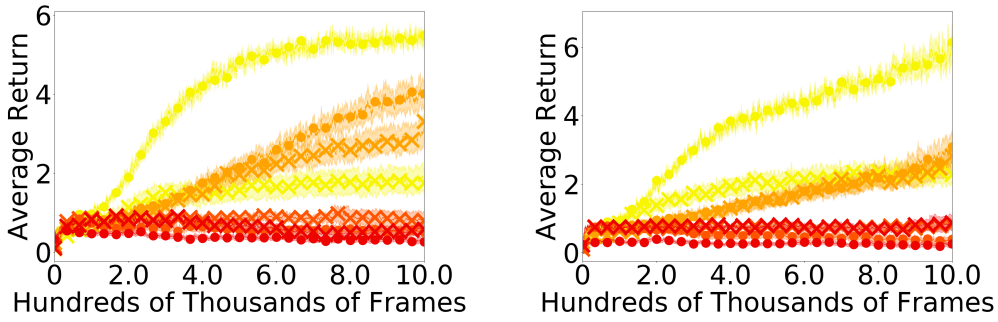
(b) Breakout



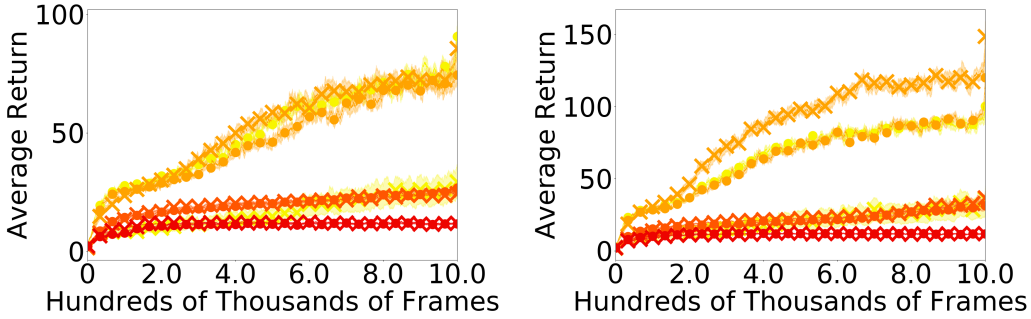
(c) Freeway

Figure 5.3: MinAtar discrete-action environments. Plot settings are identical to those in Figure 5.1.

As with the OpenAI Gym environments, there is no consistent dominance of either KL over the other in the MinAtar environments. The most striking



(a) Seaquest



(b) Space Invaders

Figure 5.4: MinAtar discrete-action environments continued. Plot settings are identical to those in Figure 5.1.

observation is the superiority of FKL with $\tau = 0$ over all other methods and across both hidden layer sizes on Seaquest in Figure 5.4a. As Seaquest is generally considered to be a hard exploration problem, we might expect FKL to perform better because of its relative slowness in moving probability mass to regions with high target density, as with RKL. If this reason were the only relevant factor, we would expect FKL to be superior across all temperatures; however, FKL performs similarly to RKL for the other temperatures, excepting slightly better final performance for $\tau = 0.1$ and a hidden layer size of 32.

Another possibility to explain the results in Seaquest is to differentiate between the supposed exploration benefits of entropy and of FKL. When we say that a method benefits exploration, we mean that the method induces a state visitation distribution whose support is larger (i.e., covers more of the state space). Accumulating more transitions from more diverse parts of the state space presumably allows for more accurate estimates of the action value function,

and hence more reliable policy improvement. Entropy-regularized RL, as it is currently formulated, only benefits exploration by proxy, through penalizing the negative entropy of the policy. In the context of reward maximization, entropy is only a means to an end; at times, the means may conflict with the end. A policy with higher entropy may have a more diverse state visitation distribution, but it may be prevented from exploiting that information to the fullest capacity because of the penalty to negative entropy. In contrast, FKL benefits exploration by causing the agent's policy to commit more slowly to actions that apparently have high value under the current value function estimate. Especially since value function estimates can be atrocious (Ilyas et al., 2020), this non-committal behaviour may help the policy avoid spurious local optima. However, FKL does not necessarily prevent a policy from committing to a particular action. Furthermore, nothing in principle prevents a policy under the hard FKL from converging to the optimal policy of the original MDP, whereas introducing entropy regularization induces a different optimal, entropy-regularized policy.

Nevertheless, the effects of entropy and of non-committal behaviour might not be independent in all scenarios. On Freeway in Figure 5.3c, RKL bests FKL for $\tau = 0.01$, but the opposite is true for $\tau = 0$. It is conceivable that a higher temperature offsets the committal behaviour of RKL enough to impact exploration positively, while when $\tau = 0$, the RKL objective induces the policy to place large mass on actions that only appear to be optimal. A particularly striking failure case of RKL with $\tau = 0$ is on Breakout in Figure 5.3b, where this setting fails to learn appreciably compared to all other KL and τ combinations. A next step to investigate the exploration effects of entropy and of FKL would be to plot the induced state distributions of FKL and RKL over time, along with the corresponding value function estimation errors.

5.4 Summary

We examined the differences between FKL and RKL in high-dimensional environments that necessitate function approximation. While any differences

depended on the environment and the temperature, there are a few key take-aways.

1. We hypothesize that using the FKL may benefit exploration (i.e., inducing a state visitation distribution with larger support).

2. Entropy-regularization and the FKL in conjunction may both benefit exploration, but may also inhibit learning.

3. The size of the hidden layer, and the function approximation architecture in general, may affect the relative superiority of FKL to RKL.

Chapter 6

Conclusion

6.1 Summary

Based on our theoretical and empirical analyses, we can summarize our findings as follows.

1. **Theoretically**, while the FKL may not be guaranteed to induce policy improvement as reliably as the RKL, policy improvement can still occur if a sufficiently high reduction in FKL occurs, with a sufficiently high temperature, and with a sufficiently high entropy in the new policy. These combined conditions seem quite strong, but weaker conditions may be able to be derived through more sophisticated techniques. We hypothesize that the superior policy improvement result of the RKL may be related to its committing to actions with high action-value that we observed in our experiments. As Neumann et al. (2011) also observes, the RKL seems less averse to costs than the FKL.

2. **On the microworld experiments**, there were more differences between FKL and RKL in the continuous-action setting than in the discrete-action setting. In the former, the FKL tended to have a smoother loss landscape that directed iterates to a global optimum, although this global optimum was sometimes less optimal, especially with higher temperature, than the optima of the RKL. In both continuous-action and discrete-action settings, iterates under the RKL tended to have limit points closer to the global optimum of the unregularized Switch-Stay problem. These results,

too, are related to the cost-averse nature of the RKL. One additional, confounding factor seems to be the policy parameterization, as a tanh-Gaussian policy cannot represent all possible probability distributions; given this limitation, the RKL and FKL enforce different trade-offs.

3. **On our benchmark experiments**, while there was no consistent dominance of either KL over the other, some interesting trends emerged. Using the FKL may benefit exploration in possibly encouraging a state visitation distribution with wider support. This impact intersects with the impact of entropy regularization; we observed that using both the FKL and a high degree of entropy-regularization could prevent learning, although this effect was heavily environment-dependent. We also observed some evidence that the FKL was superior to RKL under small hidden layer sizes, but this observation did not hold across all environments.

An important conclusion from this work is that the FKL is promising for policy greedification, even though it is rarely used. To start using it, we need simpler ways to optimize it. We used numerical integration for most of our experiments, but such a method does not scale well to high-dimensional action spaces and is susceptible to truncation error. Indeed, two applications of integration are required: (1) to calculate the partition function and (2) to calculate the loss. If $\tau = 0$, then one must instead find the maximum action of a continuous action value function, which seems equally difficult. Sampling-based approaches like weighted importance sampling, which we used in one of our microworld experiments, would be fruitful to explore further in the context of large-scale environments.

6.2 Limitations

There are some limitations of the current study that we should keep in mind, especially in informing future work.

1. Theoretically, we assumed that true action-values are available. In practice, action-value estimates tend to be quite poor, and work remains

to be done in characterizing the possibility of policy improvement with these estimates.

2. Our policy improvement result for the FKL was weaker than the corresponding RKL result and required strong assumptions. In our experiments, the fact that the FKL was sometimes superior to the RKL suggests that weaker assumptions may suffice.

3. In our microworld experiments, we also focused on the case of having access to the true action-values. Having to learn the action-values would have added an additional confounding factor to our experiments, but is essential for expanding the scope of applicability of this work.

4. On our continuous-action experiments, we did not test FKL with weighted importance sampling. Our microworld experiments suggested that FKL could perform well with this method, obviating the need for expensive quadrature procedures.

5. While our theoretical results did not assume any prior reward structure, we did not explore how different reward structures may impact the policy improvement differences between the KL divergences in our experiments.

6. We used RMSprop on our benchmark experiments. Although results for RMSprop and Adam were similar on our microworlds, the impact of momentum may be greater in more complex environments.

7. For all of our continuous-action experiments, our policy was the pushforward distribution induced by applying tanh to the output of a Gaussian distribution. We made this design choice to ensure that any output actions would remain in [-1, 1], avoiding the bias (Chou et al., 2017) in the policy gradient that results when action constraints are ignored. While we believe our choice to be reasonable, it would be interesting to understand if our results hold for the unmodified Gaussian policy, or for another pushforward distribution.

6.3 Future Work

A natural question from this study is why the differences were the largest for continuous actions in our microworld experiments. One potential reason is the policy parameterization: the Gaussian policy is likely more restrictive than the softmax as it cannot capture multimodal structure. Learning the standard deviation of a Gaussian policy may be another source of instability. In contrast, a softmax policy can represent multiple modes, and does not separate the parameterization of the measure of central tendency (e.g., mean) and the measure of variation (e.g., standard deviation). With a Gaussian policy, FKL seems to have a better optimization surface (having smooth and single optima across different temperatures) despite the multimodality of the target distribution in our continuous bandit. However, none of these observations may hold for other policy parameterizations. A promising next step is to compare FKL and RKL with different policy parameterizations for continuous actions.

Recent work into alternative policy paramaterizations has explored the Beta distribution (Chou et al., 2017), quantile regression (Richter and Wattenhofer, 2019), and normalizing flows (Ward et al., 2019). While the latter two works in particular have focused on the motivation of multimodality for domains that have multiple goals, we believe that the relevance of multimodality for optimization is as important.

We should also recall the approach of soft Q-learning (Haarnoja et al., 2017), which approximates the target policy with a number of particles, updating each particle with Stein variational gradient descent (Liu and Wang, 2016; Liu et al., 2017). In this case, it is not necessary to have explicit policies. This approach, however, does make it difficult to deploy a learned policy to other environments.

Let us not forget that there are many other possible choices for a greedification objective! Besides the KL divergences, one may consider the Wasserstein distance, Cramer distance, the JS divergence, and many more. One reason we focused on the KL in this work was its ease of optimization, compared to the Wasserstein distance for example. There may however be other cogent reasons

for selecting an objective; modeling the quantiles of the policy, for instance, suggests using the quantile loss.

The choice of target distribution in the greedification objective is another sticky issue. The Boltzmann distribution over action values is a natural choice for entropy-regularized RL, but one might not want to be tied this framework, especially given sensitivity to the temperature parameter and exploration that is undirected. Instead of maximizing the entropy of a distribution over actions, one could try to maximize the entropy of the discounted state visitation distribution (Islam et al., 2019). If the goal is exploration of the state space, perhaps one should let the agent decide how to do so, rather than imposing the proxy of high entropy in the action distribution.

Finally, let us return to a point we noted in the introduction. Most policy gradient methods today do not follow the gradient of *any* objective function (Nota and Thomas, 2020; Thomas, 2014) because a γ^t term is omitted from the update. Thomas (2014) showed that the resulting semi-gradient corresponds to one of two terms in the gradient of the average-reward objective; this "semi-gradient" neglects the effect of the policy parameters upon the state visitation distribution. Coupled with the fact that one tries to set γ as close to 1 as possible in practice while avoiding instability issues, this line of work suggests that we should try instead to optimize the average-reward objective instead of the discounted objective. In many applications of RL, one would presumably desire those agents to optimize for a long-sighted criterion, rather than one susceptible to nearsightedness as a result of a discount factor too far from 1. In fact, the original policy gradient theorem includes a policy gradient for the average reward objective (Sutton et al., 2000). In introducing natural gradient methods, Kakade (2002) focuses on the average reward setting. Although the average reward criterion is most suitable for the continuing–rather than episodic–setting, it might be possible to lift insights from the former to the latter. We hope that future research will address all of these shortfalls.

References

Abdolmaleki, Abbas et al. (2018). "Maximum a Posteriori Policy Optimisation." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=S1ANxQW0b. 2

Agarwal, Alekh et al. (2019a). "Optimality and Approximation with Policy Gradient Methods in Markov Decision Processes." In: *arXiv preprint arXiv:1908.00261*. 3

Agarwal, Rishabh et al. (June 2019b). "Learning to Generalize from Sparse and Underspecified Rewards." In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 130–140. 5

Ahmed, Zafarali et al. (June 2019). "Understanding the Impact of Entropy on Policy Optimization." In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 151–160. URL: http://proceedings.mlr.press/v97/ahmed19a.html. 3

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (2017). *Wasserstein GAN*. arXiv: 1701.07875 [stat.ML]. 13

Bertsekas, Dimitri P. (2011). "Approximate Policy Iteration: A Survey and Some New Methods." In: *Journal of Control Theory and Applications* 9.3, pp. 310–335. 3

Bhandari, Jalaj and Daniel Russo (2019). *Global Optimality Guarantees For Policy Gradient Methods*. arXiv: 1906.01786 [cs.LG]. 3, 4

Billingsley, Patrick (2008). *Probability and Measure*. John Wiley & Sons. 13

Brockman, Greg et al. (2016). "OpenAI Gym." In: *arXiv preprint arXiv:1606.01540*. 59, 60

Chen, Minmin et al. (2019). "Surrogate Objectives for Batch Policy Optimization in One-step Decision Making." In: *Advances in Neural Information Processing Systems*, pp. 8825–8835. 5

Chou, Po-Wei, Daniel Maturana, and Sebastian Scherer (2017). "Improving Stochastic Policy Gradients in Continuous Control with Deep Reinforcement Learning using the Beta Distribution." In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 834–843. 38, 68, 69

Clenshaw, Charles W and Alan R Curtis (1960). "A Method for Numerical Integration on an Automatic Computer." In: *Numerische Mathematik* 2.1, pp. 197–205. 37, 56

Dadashi, Robert et al. (June 2019). "The Value Function Polytope in Reinforcement Learning." In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 1486–1495. URL: `http://proceedings.mlr.press/v97/dadashi19a.html`. 42, 43

Engstrom, Logan et al. (2019). "Implementation Matters in Deep RL: A Case Study on PPO and TRPO." In: *International Conference on Learning Representations*. 3

Farahmand, Amir-massoud et al. (2015). "Classification-based Approximate Policy Iteration." In: *IEEE Transactions on Automatic Control* 60.11, pp. 2989–2993. 5

Fellows, Matthew et al. (2019). "Virel: A Variational Inference Framework for Reinforcement Learning." In: *Advances in Neural Information Processing Systems*, pp. 7120–7134. 2

Geist, Matthieu, Bruno Scherrer, and Olivier Pietquin (June 2019). "A Theory of Regularized Markov Decision Processes." In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 2160–2169. URL: `http://proceedings.mlr.press/v97/geist19a.html`. 15, 49

Gerstner, Thomas and Michael Griebel (1998). "Numerical Integration using Sparse Grids." In: *Numerical Algorithms* 18.3, p. 209. 56

Haarnoja, Tuomas et al. (2017). "Reinforcement Learning with Deep Energy-based Policies." In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 1352–1361. 2, 15, 69

Haarnoja, Tuomas et al. (2018). "Soft Actor-Critic: Off-policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor." In: *arXiv preprint arXiv:1801.01290*. 2, 5, 21, 22, 56

Ilyas, Andrew et al. (2020). "A Closer Look at Deep Policy Gradients." In: *International Conference on Learning Representations*. 3, 64

Imani, Ehsan, Eric Graves, and Martha White (2018). "An Off-policy Policy Gradient Theorem using Emphatic weightings." In: *Advances in Neural Information Processing Systems*, pp. 96–106. 3, 20

Islam, Riashat et al. (2019). *Entropy Regularization with Discounted Future State Distribution in Policy Gradient Methods*. arXiv: `1912.05104 [cs.LG]`. 70

Kakade, Sham M. (2002). "A Natural Policy Gradient." In: *Advances in neural information processing systems*, pp. 1531–1538. 70

Kakade, Sham and John Langford (2002). "Approximately Optimal Approximate Reinforcement Learning." In: *ICML*. Vol. 2, pp. 267–274. 3, 4, 20, 22

Kingma, Diederik P and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization." In: *International Conference on Learning Representations*. 38

Kober, Jens and Jan R Peters (2009). "Policy search for motor primitives in robotics." In: *Advances in neural information processing systems*, pp. 849–856. 4

Konda, Vijay R and John N Tsitsiklis (2000). "Actor-critic algorithms." In: *Advances in neural information processing systems*, pp. 1008–1014. 2

Lagoudakis, Michail G and Ronald Parr (2003). "Reinforcement Learning as Classification: Leveraging Modern Classifiers." In: *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 424–431. 5

Lazaric, Alessandro, Mohammad Ghavamzadeh, and Rémi Munos (2010). "Analysis of a Classification-based Policy Iteration Algorithm." In: 5

Levine, Sergey (2018). "Reinforcement learning and control as probabilistic inference: Tutorial and review." In: *arXiv preprint arXiv:1805.00909*. 2, 4

Lillicrap, Timothy P et al. (2016). "Continuous Control with Deep Reinforcement Learning." In: *International Conference on Learning Representations*. 2

Lim, Sungsu et al. (2018). "Actor-Expert: A Framework for using Q-learning in Continuous Action Spaces." In: *arXiv preprint arXiv:1810.09103*. 2

Liu, Boyi et al. (2019). "Neural Trust Region/Proximal Policy Optimization Attains Globally Optimal Policy." In: *Advances in Neural Information Processing Systems*, pp. 10564–10575. 3

Liu, Qiang and Dilin Wang (2016). *Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm*. arXiv: `1608.04471 [stat.ML]`. 69

Liu, Yang et al. (2017). *Stein Variational Policy Gradient*. arXiv: `1704.02399 [cs.LG]`. 69

Mahadevan, Sridhar (1996). "Average reward reinforcement learning: Foundations, algorithms, and empirical results." In: *Machine learning* 22.1-3, pp. 159–195. 9

Mei, Jincheng et al. (2019). "On principled entropy exploration in policy optimization." In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, pp. 3130–3136. 5

Mei, Jincheng et al. (2020). "On the Global Convergence Rates of Softmax Policy Gradient Methods." en. In: p. 10. 3

Mnih, Volodymyr et al. (2016). "Asynchronous Methods for Deep Reinforcement Learning." In: *International conference on machine learning*, pp. 1928–1937. 2

Nachum, Ofir, Mohammad Norouzi, and Dale Schuurmans (2017a). "Improving Policy Gradient by Exploring Under-appreciated Rewards." In: *International Conference on Learning Representations*. 5

Nachum, Ofir et al. (2017b). "Bridging the Gap between Value and Policy Based Reinforcement Learning." In: *Advances in Neural Information Processing Systems*, pp. 2775–2785. 4, 18

Nachum, Ofir et al. (2019). "AlgaeDICE: Policy Gradient from Arbitrary Experience." In: *arXiv preprint arXiv:1912.02074*. 4

Neu, Gergely, Anders Jonsson, and Vicenç Gómez (2017). "A Unified View of Entropy-regularized Markov Decision Processes." In: *arXiv preprint arXiv:1705.07798*. 3

Neumann, Gerhard et al. (2011). "Variational inference for policy search in changing situations." In: *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pp. 817–824. 4, 5, 66

Norouzi, Mohammad et al. (2016). "Reward Augmented Maximum Likelihood for Neural Structured Prediction." In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee et al. Curran Associates, Inc., pp. 1723–1731. URL: `http://papers.nips.cc/paper/6547-reward-augmented-maximum-likelihood-for-neural-structured-prediction.pdf`. 5

Nota, Chris and Philip S. Thomas (2020). "Is the Policy Gradient a Gradient?" In: *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020)*. 3, 19, 70

O'Donoghue, Brendan et al. (2017). "Combining Policy gradient and Q-learning." In: *International Conference on Learning Representations*. 4

Pardo, Fabio et al. (July 2018). "Time Limits in Reinforcement Learning." In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, pp. 4045–4054. URL: `http://proceedings.mlr.press/v80/pardo18a.html`. 56

Perkins, Theodore J. and Mark D. Pendrith (2002). "On the Existence of Fixed Points for Q-Learning and Sarsa in Partially Observable Domains." In: *Proceedings of the Nineteenth International Conference on Machine Learning*. ICML '02. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 490–497. ISBN: 1558608737. 4

Perkins, Theodore J and Doina Precup (2003). "A Convergent Form of Approximate Policy Iteration." In: *Advances in neural information processing systems*, pp. 1627–1634. 4

Puterman, Martin L. (2014). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons. 8

Rawlik, Konrad, Marc Toussaint, and Sethu Vijayakumar (2013). "On stochastic optimal control and reinforcement learning by approximate inference." In: *Twenty-third international joint conference on artificial intelligence*. 2

Richter, Oliver and Roger Wattenhofer (2019). *Learning Policies through Quantile Regression*. arXiv: `1906.11941 [cs.LG]`. 69

Ryu, Moonkyung et al. (2020). "CAQL: Continuous Action Q-Learning." In: *International Conference on Learning Representations*. 2

Scherrer, Bruno (2014). "Approximate Policy Iteration Schemes: A Comparison." In: *International Conference on Machine Learning*, pp. 1314–1322. 3

Scherrer, Bruno and Matthieu Geist (2014). "Local Policy Search in a Convex Space and Conservative Policy Iteration as Boosted Policy Search." In:

Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, pp. 35–50. 3, 4

Schlömer, Nico (n.d.). "quadpy: Numerical Integration (Quadrature, Cubature) in Python (2018)." In: *URL https://github. com/nschloe/quadpy.[Online* (). 37

Schulman, John, Xi Chen, and Pieter Abbeel (2017a). "Equivalence between Policy Gradients and Soft Q-learning." In: *arXiv preprint arXiv:1704.06440*. 4

Schulman, John et al. (July 2015). "Trust Region Policy Optimization." In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, pp. 1889–1897. URL: `http://proceedings.mlr.press/v37/schulman15.html`. 2

Schulman, John et al. (2016). "High-dimensional Continuous Control using Generalized Advantage Estimation." In: *International Conference on Learning Representations*. 2

Schulman, John et al. (2017b). "Proximal Policy Optimization Algorithms." In: *arXiv preprint arXiv:1707.06347*. 2

Shani, Lior, Yonathan Efroni, and Shie Mannor (2019). "Adaptive trust region policy optimization: Global convergence and faster rates for regularized mdps." In: *arXiv preprint arXiv:1909.02769*. 3

Silver, David et al. (2014). "Deterministic Policy Gradient Algorithms." In: *Proceedings of the 31 st International Conference on Machine Learning*. 2

Sutton, Richard S. (1984). "Temporal Credit Assignment in Reinforcement Learning." PhD thesis. 2

Sutton, Richard S. and Andrew G. Barto (2018). *Reinforcement Learning: An Introduction*. MIT press. 4, 7, 9, 11, 16, 33

Sutton, Richard S. et al. (2000). "Policy Gradient Methods for Reinforcement Learning with Function Approximation." In: *Advances in Neural Information Processing Systems*, pp. 1057–1063. 2, 4, 10, 11, 70

Thomas, Philip (June 2014). "Bias in Natural Actor-Critic Algorithms." In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 1. Bejing, China: PMLR, pp. 441–448. URL: `http://proceedings.mlr.press/v32/thomas14.html`. 3, 11, 70

Tieleman, Tijmen and Geoffrey Hinton (2012). "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude." In: *COURSERA: Neural networks for machine learning* 4.2, pp. 26–31. 38, 55

Todorov, Emanuel, Tom Erez, and Yuval Tassa (2012). "MuJoCo: A Physics Engine for Model-based Control." In: *IROS*. IEEE, pp. 5026–5033. 59

Vieillard, Nino, Olivier Pietquin, and Matthieu Geist (2020). "Deep Conservative Policy Iteration." In: *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. 5, 18

Wagner, Paul (2011). "A Reinterpretation of the Policy Oscillation Phenomenon in Approximate Policy Iteration." In: *Advances in Neural Information Processing Systems*, pp. 2573–2581. 4

Wagner, Paul (2013). "Optimistic Policy Iteration and Natural Actor-Critic: A Unifying View and a Non-optimality Result." In: *Advances in Neural Information Processing Systems*, pp. 1592–1600. 4

Wang, Ziyu et al. (2017). "Sample Efficient Actor-Critic with Experience Replay." In: *International Conference on Learning Representations*. 2

Ward, Patrick Nadeem, Ariella Smofsky, and Avishek Joey Bose (2019). *Improving Exploration in Soft-Actor-Critic with Normalizing Flows Policies*. arXiv: 1906.02771 [cs.LG]. 69

Watkins, Christopher JCH and Peter Dayan (1992). "Q-learning." In: *Machine learning* 8.3-4, pp. 279–292. 12

Williams, Ronald J (1992). "Simple Statistical Gradient-following Algorithms for Connectionist Reinforcement Learning." In: *Machine learning* 8.3-4, pp. 229–256. 4, 11

Young, Kenny and Tian Tian (2019). "MinAtar: An Atari-inspired Testbed for More Efficient Reinforcement Learning Experiments." In: *arXiv preprint arXiv:1903.03176*. 59, 60

Ziebart, B. (2010). "Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy." In: *PhD thesis, Carnegie Mellon University*. 2, 12, 14

Ziebart, Brian D et al. (2008). "Maximum entropy inverse reinforcement learning." In: *Aaai*. Vol. 8. Chicago, IL, USA, pp. 1433–1438. 2