Understanding the Impact of Entropy on Policy Optimization

Zafarali Ahmed [1 2] **Nicolas Le Roux** [1 3] **Mohammad Norouzi** [3] **Dale Schuurmans** [3 4]

Abstract

Entropy regularization is commonly used to improve policy optimization in reinforcement learning. It is believed to help with *exploration* by encouraging the selection of more stochastic policies. In this work, we analyze this claim using new visualizations of the optimization landscape based on randomly perturbing the loss function. We first show that even with access to the exact gradient, policy optimization is difficult due to the geometry of the objective function. We then qualitatively show that in some environments, a policy with higher entropy can make the optimization landscape smoother, thereby connecting local optima and enabling the use of larger learning rates. This paper presents new tools for understanding the optimization landscape, shows that policy entropy serves as a regularizer, and highlights the challenge of designing general-purpose policy optimization algorithms.

1. Introduction

Policy optimization is a family of reinforcement learning (RL) algorithms aiming to directly optimize the parameters of a policy by maximizing discounted cumulative rewards. This often involves a difficult non-concave maximization problem, even when using a simple policy with a linear state-action mapping.

Contemporary policy optimization algorithms build upon the REINFORCE algorithm (Williams, 1992). These algorithms involve estimating a noisy gradient of the optimization objective using Monte-Carlo sampling to enable stochastic gradient ascent. This estimate can suffer from high variance and several solutions have been proposed to address what is often seen as a major issue (Konda & Tsitsiklis, 2000; Greensmith et al., 2004; Schulman et al., 2015b; Tucker et al., 2018).

However, in this work we show that noisy estimates of the gradient are not necessarily the *main* issue: The optimization problem is difficult because of the geometry of the landscape. Given that "high variance" is often the reason given for the poor performance of policy optimization, it raises an important question: *How do we study the effects of different policy learning techniques on the underlying optimization problem?*

An answer to this question would guide future research directions and drive the design of new policy optimization techniques. Our work makes progress toward this goal by taking a look at one such technique: *entropy regularization*.

In RL, exploration is critical to finding good policies during optimization: If the optimization procedure does not sample a large number of diverse state-action pairs, it may converge to a poor policy. To prevent policies from becoming deterministic too quickly, researchers use entropy regularization (Williams & Peng, 1991; Mnih et al., 2016). Its success has sometimes been attributed to the fact that it "encourages exploration" (Mnih et al., 2016; Schulman et al., 2017a;b). Contrary to Q-learning (Watkins & Dayan, 1992) or Deterministic Policy Gradient (Silver et al., 2014) where the exploration is handled separately from the policy itself, direct policy optimization relies on the stochasticity of the policy being optimized for the exploration. However, policy optimization is a pure maximization problem and any change in the policy is reflected in the objective. Hence, any strategy, such as entropy regularization, can only affect learning in one of two ways: either it reduces the noise in the gradient estimates or it changes the optimization landscape.

In this work we investigate some of these questions by controlling the entropy of policies and observing its effect on the geometry of the optimization landscape. This work contains the following contributions:

- We show experimentally that the difficulty of policy optimization is strongly linked to the geometry of the objective function.
- We propose a novel visualization of the objective function that captures local information about gradient and curvature.

[1]Mila, McGill University, Montréal, Canada [2]Work done while at Google Research [3]Google Research [4]University of Alberta. Correspondence to: Zafarali Ahmed <zafarali.ahmed@mail.mcgill.ca>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

- We show experimentally that policies with higher entropy induce a smoother objective that connects solutions and enable the use of larger learning rates.

2. Approach

We take here the view that improvements due to entropy regularization might be attributed to having a better objective landscape. In Section 2.1 we introduce tools to investigate landscapes in the context of general optimization problems. In Section 2.1.2 we propose a new visualization technique to understand high dimensional optimization landscapes. We will then explain the RL policy optimization problem and entropy regularization in Section 2.2.

2.1. Understanding the Landscape of Objective Functions

We explain our experimental techniques by considering the general optimization problem and motivating the relevance of studying objective landscapes. We are interested in finding parameters $\theta \in \mathbb{R}^n$ that maximize an objective function, $\mathcal{O} : \mathbb{R}^n \to \mathbb{R}$, denoted $\theta^* = \arg\max_\theta \ \mathcal{O}(\theta)$. The optimization algorithm takes the form of gradient ascent: $\theta_{i+1} = \theta_i + \eta_i \nabla_\theta \mathcal{O}$, where η_i is the learning rate, $\nabla_\theta \mathcal{O}$ is the gradient of $\mathcal{O}$ and i is the iteration number.

Why should we study objective landscapes? The "difficulty" of this optimization problem is given by the properties of $\mathcal{O}$. For example, $\mathcal{O}$ might have kinks and valleys making it difficult to find good solutions from different initial parameters (Li et al., 2018b). Similarly, if $\mathcal{O}$ contains very flat regions, optimizers like gradient ascent can take a very long time to escape them (Dauphin et al., 2014). Alternatively, if the curvature of $\mathcal{O}$ changes rapidly with every θ_i, then it will be difficult to choose a stepsize.

In the subsequent subsections, we describe two effective techniques for visualization of the optimization landscapes.

2.1.1. Linear Interpolations

One approach to visualize an objective function is to interpolate θ in the 1D subspace between two points θ_0 and θ_1 (Chapelle & Wu, 2010; Goodfellow et al., 2015) by evaluating the objective at $\mathcal{O}((1-\alpha)\theta_0 + \alpha\theta_1)$ for $0 \leq \alpha \leq 1$. Such visualizations can tell us about the existence of valleys or monotonically increasing paths of improvement between the parameters. Typically θ_0 and θ_1 are initial parameters or solutions obtained through the optimization.

Though this technique provides interesting visualizations, conclusions are limited to the 1D slice: Draxler et al. (2018) show that even though the local optima are isolated in the 1D slice, these local optima can be connected by a manifold of equal value. Hence, we must be careful to conclude general properties about the landscape using this visualization. In the next section, we describe a new visualization technique that, together with linear interpolations, can serve as a powerful tool for landscape analysis.

2.1.2. Objective Function Geometry using Random Perturbations

To overcome some of the limitations described in Section 2.1.1, we develop a new method to locally characterize the properties of $\mathcal{O}$. In particular, we use this technique to (1) classify points in the parameter space as local optimum, saddle point, or flat regions; and (2) measure curvature of the objective during optimization.

To understand the local geometry of $\mathcal{O}$ around a point θ_0 we sample directions d uniformly at random on the unit ball. We then probe how $\mathcal{O}$ is changing along the sampled direction by evaluating at a pair of new points: $\theta_d^+ = \theta_0 + \alpha d$ and $\theta_d^- = \theta_0 - \alpha d$ for some value α. After collecting multiple such samples and calculating the change for each pair with respect to the initial point, $\Delta_d^{\mathcal{O}^+} = \mathcal{O}(\theta_d^+) - \mathcal{O}(\theta_0)$ and $\Delta_d^{\mathcal{O}^-} = \mathcal{O}(\theta_d^-) - \mathcal{O}(\theta_0)$, we can then classify a point θ_0 according to:

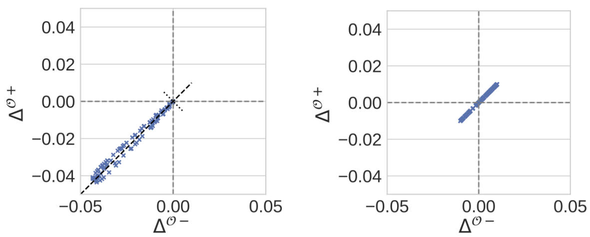
1. If $\Delta_d^{\mathcal{O}+} < 0$ and $\Delta_d^{\mathcal{O}-} < 0$ for all d, θ_0 is a local maximum.
2. If $\Delta_d^{\mathcal{O}+} > 0$ and $\Delta_d^{\mathcal{O}-} > 0$ for all d, θ_0 is a local minimum.
3. If $\Delta_d^{\mathcal{O}+} \approx -\Delta_d^{\mathcal{O}-}$, θ_0 is in an almost linear region.
4. If $\Delta_d^{\mathcal{O}+} \approx \Delta_d^{\mathcal{O}-} \approx 0$, θ_0 is in an almost flat region.

In practice, since we only sample a finite number of directions, we can only reason about the probability of being in such a state. Further, we can also observe a combination of these pairs, for instance in the case of saddle points[1].

As an example, consider $\mathcal{O}(\theta) = -(1-\theta_0\theta_1)^2$ that has a saddle point and a manifold of local optima (Goodfellow et al., 2015). The proposed technique can distinguish between the local optimum at $\theta = (-0.5, -2)$ and saddle point at $\theta = (0, 0)$ (Figure 1). Other examples on simple quadratics are shown in Figure S1 and S2.

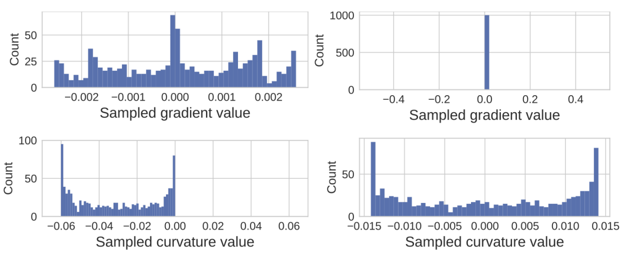
Our method captures a lot of information about the local geometry. To summarize it, we can go one step further and disentangle information about gradient and curvature. If we assume $\mathcal{O}$ is locally quadratic, i.e., $\mathcal{O}(\theta) \approx a^T\theta + \frac{1}{2}\theta^T H \theta$, where a is the linear component and H is a symmetric

[1] This list is not exhaustive and one can imagine detecting many more scenarios



(a) Apparent local optimum (b) Saddle point

Figure 1. **Demonstration of random perturbation technique.** (a) If all perturbations are strictly negative, it implies that the point is likely a local optima. Given that some perturbations evaluate to 0, suggests some flat directions alluding to the connected manifold (b) If both perturbations are positive or both are negative it implies that the point is a saddle. See Section 2.1 for detailed explanation.



(a) Apparent local optimum (b) Saddle point

Figure 2. **Demonstration of projecting random perturbations.** (a) At the apparent local optimum, there is a tiny gradient and all curvatures are zero or negative. (b) At the saddle point, there is no gradient and curvatures are both positive and negative.

matrix (i.e., Hessian), then:

$$\Delta_d^{\mathcal{O}+} - \Delta_d^{\mathcal{O}-} = 2\alpha \nabla \mathcal{O}(\theta_0)^T d \,, \tag{1}$$
$$\Delta_d^{\mathcal{O}+} + \Delta_d^{\mathcal{O}-} = \alpha^2 d^T H d \,. \tag{2}$$

The derivation is done in Appendix S.1.2. Therefore, projections of our scatter plots capture information about the components of the gradient and Hessian in the random direction d. By repeatedly sampling many directions we eventually recover how the gradient and curvature vary in many directions around θ_0. We can use a histogram to describe the density of these curvatures (Figure 2). In particular, the maximum and minimum curvature values obtained from this technique are close to the maximum and minimum eigenvalues of H. This curvature spectrum is related to eigenvalue spectra which have been used before to analyze neural networks (Le Cun et al., 1991; Dauphin et al., 2014).

While this work analyzes models with hundreds of parameters, a technique based on random perturbations is likely to miss some direction in higher dimensions. This is particularly problematic if these dimensions are the ones an optimizer would follow. We consider this limitation in the context of stochastic gradient methods, where $\nabla_\theta \mathcal{O}$ is a noisy estimate of the true gradient. At points where the objective does not change much, the true gradient is small compared to the noise introduced by the stochasticity. Therefore, the direction followed by the stochastic gradient method is dominated by noise: to escape quickly, there must be many directions to follow that increase the objective.

We give an example of the behaviour of our technique in various toy settings in Section S.1 and the methods are summarized in Figure S4.

Equipped with these tools, we now describe specific details about the RL optimization problem in the next section.

2.2. The Policy Optimization Problem

In policy optimization, we aim to learn parameters, θ, of a policy, $\pi_\theta(a|s)$, such that when acting in an environment the sampled actions, a, maximize the discounted cumulative rewards, i.e., $\mathcal{O}_{ER}(\theta) = \mathbb{E}_{\pi_\theta}[\sum_{t=1}^{\infty} \gamma^t r_t]$, where γ is a discount factor and r_t is the reward at time t. The gradient is given by the policy gradient theorem (Sutton et al., 2000) as: $\nabla_\theta \mathcal{O}_{ER}(\theta) = \int_s d^{\pi_\theta}(s) \int_a \nabla_\theta \pi_\theta(a|s) Q^{\pi_\theta}(s,a) \mathrm{d}a \mathrm{d}s$ where d^π is the stationary distribution of states and $Q^\pi(a_t, s_t)$ is the expected discounted sum of rewards starting state s, taking action a and then sampling actions according to the policy, $a \sim \pi(\cdot|s)$.

One approach to prevent premature convergence to a deterministic policy is to use entropy regularization (Schulman et al., 2017a). This is often done by augmenting the rewards with an entropy term, $\mathbb{H}(\pi(\cdot|s_t)) = \mathbb{E}_{a\sim\pi(\cdot|s_t)}[-\log \pi(a|s_t)]$, weighted by τ, i.e. $r_t^\tau = r_t + \tau \mathbb{H}(\pi(\cdot|s_t))$, and results in a slightly different gradient:

$$\nabla_\theta \mathcal{O}_{ENT}(\theta) = \int_s d^{\pi_\theta}(s) \int_a \pi(a|s) \Big[Q^{\tau,\pi_\theta}(s,a) \nabla_\theta \log \pi(a|s) + \tau \nabla_\theta \mathbb{H}(\pi(\cdot|s)) \Big] \mathrm{d}a\mathrm{d}s \tag{3}$$

where $Q^{\tau,\pi_\theta}(s,a)$ is the expected discounted sum of entropy-augmented rewards. Q^{τ,π_θ} can be calculated exactly if the dynamics of the environment are known (Sutton et al. (2000), Appendix S.2) or estimated by executing π_θ in the environment (Williams (1992), Appendix S.2.1). It is noteworthy that both $\mathcal{O}_{ER}$ and $\nabla \mathcal{O}_{ENT}$ depend on π_θ: Therefore, any change in the policy will change the experience distribution, d^{π_θ} and be reflected in both the objective and gradient.

3. Results

Now that we have the tools to investigate objective landscapes from Section 2.1, we return to questions related to entropy and policy optimization. Firstly, in Section 3.1, we use environments with no gradient estimation error to

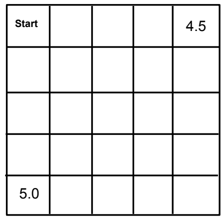
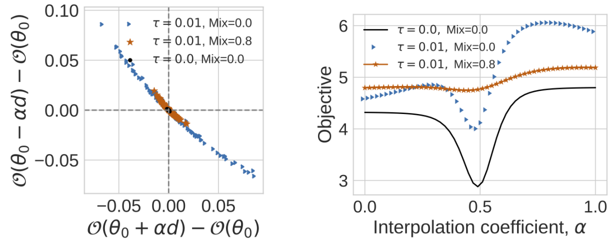


Figure 3. Gridworld used in the experiments in Section 3.1. There are two locally optimal policies: always going right and always going bottom.



(a) (b)

Figure 4. **Objective function geometry around solutions in the Gridworld.** (a) Scatter plot for the change in objective for different random directions. Without entropy, no sampled directions show improvement in the objective (black circles). However, with either entropy or a more stochastic policy (blue triangle and orange star) many directions give positive improvement. (b) Linear interpolation between two solution policies for sub-optimal and optimal rewards are separated by a valley of poor solutions in the true objective (black). Using a stochastic policy and entropy regularized objective connects these local optima (orange stars) in this 1D slice. See Section 3.1 for a detailed explanation.

provide evidence for policy optimization being difficult due to the geometry of the objective function. Our main contribution is in Section 3.2, where we observe the smoothing effect of stochastic policies on the optimization landscape in high dimensional environments. Put together, these results should highlight the difficulty and environment-dependency of designing optimization techniques that are orthogonal to variance reduction of the gradient estimate.

3.1. Entropy Helps Even with the Exact Gradient

The high variance in gradient estimates due to using samples from a stochastic policy in a stochastic environment is often the reason given for poor optimization. To emphasize that policy optimization is difficult even if we solved the high variance issue, we conduct experiments in a setting where the optimization procedure has access to the exact gradient. We then link the poor optimization performance to visualizations of the objective function. Finally, we show how having an entropy augmented reward and, in general, a more stochastic policy changes this objective resulting in overall improvement in the solutions found.

3.1.1. Experimental Setup: Environments with No Variance in the Gradient Estimate

To investigate if mitigating the high variance problem is the key to better optimization, we set our experiment in an environment where the gradient can be calculated exactly. In particular, we replace the integrals with summations and use environment dynamics to calculate Equation 3 resulting in no sampling error. We chose a 5×5 Gridworld with one suboptimal and one optimal reward at the corners (Figure 3). Our agent starts in the top left corner and has four actions parameterized by a categorical distribution $\pi(a|s_t) \propto \exp(\theta^T s_t)$ and states are given by their one-hot representation. As such there are two locally optimal policies: go down, π_{opt} and go right, π_{sub}. We refer to the case where the entropy weight $\tau = 0$ as the *true objective*.

3.1.2. Are Poor Gradient Estimates the Main Issue with Policy Optimization?

After running exact gradient ascent in the Gridworld starting from different random initializations of θ_0, we find that about 25% of these initializations led to a sub-optimal final policy: there is some inherent difficulty in the geometry of the optimization landscape independent of sampling noise. To get a better understanding of this landscape, we analyze two solutions that parameterize policies that are nearly deterministic for their respective rewards θ_{sub} and θ_{opt}. The objective function around θ_{sub} has a negligible gradient and small strictly negative curvature values indicating that the solution is in a very flat region (Figure 4a, black circles). On a more global scale, θ_{sub} and θ_{opt} are located in flat regions separated by a sharp valley of poor solutions (Figure 4b, black circles).

These results suggest that at least some of the difficulty in policy optimization comes from the flatness and valleys in the objective function independent of poor gradient estimates. In the next sections, we investigate effect of entropy regularization on the objective function.

3.1.3. Why Does Using Entropy Regularization Find Better Solutions?

Our problem setting is such that an RL practitioner would intuitively think of using entropy regularization to encourage the policy to "keep exploring" even after finding R_{sub}. Indeed, including entropy ($\tau > 0$) and decaying it during optimization, reduces the proportion of sub-optimal solutions found by the optimization procedure to 0 (Figure S5)[2]. We explore reasons for the improved performance in this section.

We see in Figure 4a (orange stars) that augmenting the

[2]We classify a policy as *optimal* if it achieves a return greater than that of the deterministic policy reaching R_{sub}.

objective with entropy results in many directions of positive improvement at θ_{sub}: it is no longer a flat solution.

We also show that interpolating a stochastic[3] version of the policy can connect the two local optima in this slice: A smooth, and in some cases monotonically increasing, path now appears in this augmented objective to a region with high value in the true objective (Figure 4b orange stars, S6). This means that if we knew a good direction a priori, a line search would find a better solution.

This section provided a different and more accurate interpretation for entropy regularization by connecting it to changes in objective function geometry. Given that entropy regularization encourages our policies to be more stochastic, we now ask *What is it about stochastic policies that helps learning?* In the next section, we explore two possible reasons for this in a more realistic high dimensional continuous control problem.

3.2. More Stochastic Policies Induce Smoother Objectives in Some Environments

In Section 3.1.3 we saw that entropy and, more generally, stochasticity can induce a "nicer" objective to optimize. Our second experimental setup allows us to answer questions about the optimization implications of stochastic policies. We show qualitatively that high entropy policies can speed up learning and improve the final solutions found. We empirically investigate some reasons for these improvements related to smoothing. We also show that these effects are environment-specific highlighting the challenges of policy optimization.

3.2.1. Experimental Setup: Environments that Allow Explicit Study of Policy Entropy

Continuous control tasks from the MuJoCo simulator (Todorov et al., 2012; Brockman et al., 2016) facilitate studying the impact of entropy because we can parameterize policies using Gaussian distributions. In particular, since entropy regularization increases the entropy of a policy, we can study the impact of entropy on optimization by controlling the stochasticity of the Gaussian policy. Specifically, the entropy of a Gaussian distribution depends only on σ, and thus we control σ explicitly to study varying levels of entropy. We use a large batch size to control for the variance reduction effects of a larger σ (Zhao et al., 2011).

To keep the analysis as simple as possible, we parameterize the mean by $\theta^T s_t$ which is known to result in good performance (Rajeswaran et al., 2017). Since we do not have access to transition and reward dynamics, we cannot calculate the policy gradient exactly and use the REINFORCE gradient estimator (Williams (1992), Appendix S.2.1). To study performance during learning we consider the *deterministic* performance[4] by evaluating the learned policy with $\sigma = 0$.

To study the landscape, we use the techniques described in Section 2.1 to analyze θ under different values of σ. Specifically, we obtain multiple values of θ by optimizing different values of σ. To understand how objective landscapes change, we re-evaluate interpolated slices and random perturbations under a different value of σ. We consider $\sigma = 0$ to be the policy we are interested in and refer to the objective calculated as the *true objective*.

3.2.2. What is the Effect of Entropy on Learning Dynamics?

We first show that optimizing a more stochastic policy can result in faster learning in more complicated environments and better final policies in some.

In Hopper and Walker high entropy policies ($\sigma > 0.1$) quickly achieve higher rewards than low entropy policies ($\sigma = 0.1$) (Figure 5ab). In HalfCheetah, even though high entropy policies learn quicker (Figure 5c), the differences are less apparent and are more strongly influenced by the initialization seed (Figure S9). In both Hopper and Walker2d, the mean reward of final policies found by optimizing high entropy policies is 2 to 8 times larger than a policy with $\sigma = 0.1$ whereas, in HalfCheetah, all policies converge to a similar final reward commonly observed in the literature (Schulman et al., 2017b).

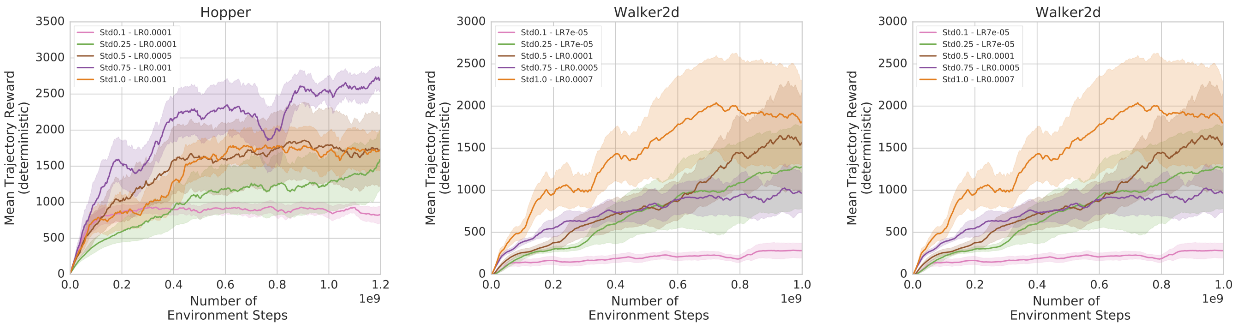
Though statistical power to make fine-scale conclusions is limited, the qualitative trend holds: More stochastic policies perform better in terms of speed of learning and, in some environments, final policy learned. In the next two sections we investigate some reasons for these performance improvements as well as the discrepancy with HalfCheetah.

3.2.3. Why do high entropy policies learn quickly?

We first focus on the speed of learning: A hint for the answer comes from our hyperparameter search over constant learning rates. In Hopper and Walker, the best learning rate increases consistently with entropy: The learning rate for $\sigma = 1$ is 10 times larger than for $\sigma = 0.1$. At every time step, the optimal learning rate is tied to the local curva-

[3] This is done by setting the policy being evaluated to $(1 - \text{Mix})\pi_{(1-\alpha)\theta_0+\alpha\theta_1}(a|s) + \frac{\text{Mix}}{|A|}$ so that every action has a minimum probability of $\frac{\text{Mix}}{|A|}$.

[4] In this setting, the online performance does not matter, we only rely on the stochasticity for "exploration". We note that the deterministic performance will be better than the online stochastic performance of the policy.



(a) (b) (c)

Figure 5. **Learning curves for policies with different entropy in continuous control tasks** In all environments using a high entropy policy results in faster learning. These effects are less apparent in HalfCheetah. In Hopper and Walker high entropy policies also find better final solutions. Learning rates are shown in the legends and entropy is controlled using the standard deviation. Solid curve represents the average of 5 random seeds. Shaded region represents half a standard deviation for readability. Individual learning curves are shown in Figure S7 for Hopper, Figure S8 for Walker and Figure S9 for HalfCheetah. See Section 3.2.3 and 3.2.4 for discussion.

ture of the loss. If the curvature changes, the best constant learning rate is the smallest one which works across all curvatures. The change in optimal learning rate cannot be solely explained by the implicit rescaling of the parameters induced by the change in σ since the loss also increases faster with higher entropy, which would not happen with a mere scalar reparametrization. Thus, that difference in optimal learning rate suggests that adding entropy instead damps the variations of curvature along the trajectory, facilitating optimization with a constant learning rate η.

To investigate, we calculate the curvature of the objective during the first few thousand iterations of the optimization procedure. In particular, we record the curvature in a direction of improvement[5]. As expected, curvature values fluctuate with a large amplitude for low values of σ (Figure 6a, S10, S11). In this setting, selecting a large and constant η might be more difficult compared to an objective induced by a policy with a larger σ. In contrast, the magnitude of fluctuations are only marginally affected by increasing σ in HalfCheetah (Figure 6b and S12) which might explain why using a more stochastic policy in this environment does not facilitate the use of larger learning rates.

In this section, we showed that fluctuations in the curvature of objectives decrease for more stochastic policies in some environments. The implications for these are two-fold: (1) It provides evidence for why high entropy policies facilitate the use of a larger learning rate; and (2) The impact of entropy can be highly environment specific. In the next section, we shift our focus to investigate the reasons for improved quality of *final* policies found when optimizing high entropy policies.

3.2.4. Can High Entropy Policies Reduce the Number of Local Optima in the Objective?

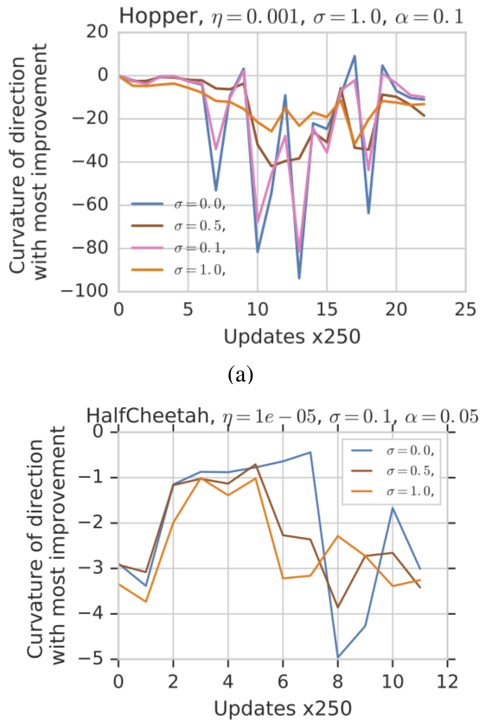
In this section, we improve our understanding of which values of θ are reachable at the end of optimization. We are trying to understand *Why do high entropy policies learn better final solutions?* Specifically, we attempt to classify the local geometry of parameters and investigate the effect of making the policy more stochastic. We will then argue that high entropy policies induce a more connected landscape.

Final solutions in Hopper for $\sigma = 0.1$ have roughly 3 times more directions with negative curvature than $\sigma = 1.0$. This suggests that final solutions found when optimizing a high entropy policy lie in regions that are flatter and some directions *might* lead to improvement. To understand if a more stochastic policy can facilitate an improvement from a poor solution, we visualize the local objective for increasing values of σ. Figure 7 shows this analysis for one such solution (Figure S7d) where the objective oscillates: The stochastic gradient direction is dominated by noise. For deterministic policies 84% of directions have a detectable[6] negative curvature (Figure 7a) with the rest having near-zero curvature: The solution is likely near a local optimum. When the policy is made more stochastic, the number of directions with negative curvature reduces dramatically suggesting that the solution might be in a linear region. However, just because there are fewer directions with negative curvature, it does not imply that any of them reach good final policies.

To verify that there exists *at least* one path of improvement to a good solution, we linearly interpolate between this so-

[5]We selected the direction of improvement closest to the 90th percentile which would be robust to outliers.

[6]Taking into account noise in the sampling process.



(a)

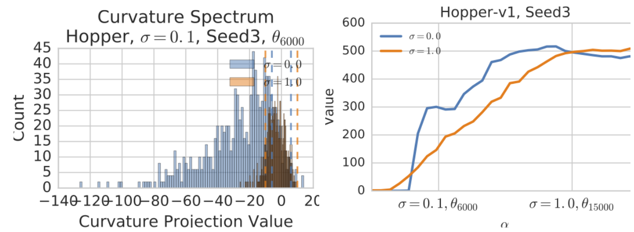
(b)

Figure 6. **Curvature during training** (a) The curvature for the direction of most improvement in Hopper fluctuates rapidly for optimization objectives with low entropy ($\sigma \in \{0.0, 0.1\}$) compared to those of high entropy ($\sigma \in \{0.5, 1.0\}$). (b) Entropy does not seem to have any effect on HalfCheetah.

lution and parameters for a good final policy obtained by optimizing $\sigma = 1.0$ starting from the same random initialization (Figure 7b). Surprisingly, even though this is just one very specific direction, there exists a monotonically increasing path to a better solution in the high entropy objective: If the optimizer knew the direction in advance, a simple line search would have improved upon a bad solution when using a high entropy policy. This finding extends to other pairs of parameters (Figure S14 for Hopper and Figure S15 and S16 for Walker2d) but not all (Figure S17) indicating that *some* slices of the objective function may become easier to optimize and find better solutions.

Our observations do not extend to HalfCheetah, where we were unable to find such pairs (Figure S17c) and specifically, objectives around final solutions did not change much for different values of σ (Figure S18). These observations suggest that the objective landscape in HalfCheetah is not significantly affected by changing the policy entropy and explains the marginal influence of entropy in finding better solutions in this environment.

As seen before the impact of entropy on the objective function seems to be environment specific. However, in environ-



(a)

(b)

Figure 7. **Analyzing solutions in objectives given by different amounts of entropy** (a) For $\sigma = 0.0$, 85% of curvature values are negative. When σ is increased to 1, nearly all curvature values are within sampling noise (indicated by dashed horizontal lines). (b) A linear interpolation shows that a monotonically increasing path to a better solution exists from the poor parameter vector. See Figures S14 and S15 for a different seed and in Walker respectively. See Figure S17 for negative examples.

ments where the objective functions are affected by having a more stochastic policy, we have evidence that they can reduce at least a few local optima by connecting different regions of parameter space.

4. Related Work

There have been many visualization techniques for objective functions proposed in the last few years (Goodfellow et al., 2015; Li et al., 2018b; Draxler et al., 2018). In contrast to our work, many of these project the high dimensional objective into one or two useful dimensions. Our technique is closely related to those of Li et al. (2018b), who used random directions for 2D interpolations, and of Li et al. (2018a); Fort & Scherlis (2018) who studied optimization in random hyperplanes and hyperspheres. Our work differs in that we interpolate in many more directions to summarize how the objective function changes locally. Keskar et al. (2017) used a summary metric from random perturbations to measure sharpness of an objective similar to our measure of curvature in Figure 6.

Understanding the impact of entropy on the policy optimization problem was first studied by Williams & Peng (1991). A different kind of entropy regularization has been explored in the context of deep learning: Chaudhari et al. (2017) show that such a penalty induces objectives with higher β-smoothness and complement our smoothness results. Recent work by Neu et al. (2017) has shown the equivalence between the type of entropy used and a dual optimization algorithm.

The motivation of our work is closely related to Rajeswaran et al. (2017); Henderson et al. (2018); Ilyas et al. (2018) in appealing to the community to study the policy optimization problem more closely. In particular, Ilyas et al. (2018) show that in deep RL, gradient estimates can be uncorrelated with the true gradient despite having good optimization performance. This observation complements our work in saying that high variance might not be the main issue for policy optimization. The authors use 2D interpolations to show that an ascent in surrogate objectives used in PPO did not necessarily correspond to an ascent in the true objective. Our results provide a potential explanation to this phenomenon: surrogate objectives can connect regions of parameter space where the true objective might decrease.

5. Discussion and Future Directions

The difficulty of policy optimization. Our work aims to redirect some research focus from the high variance issue to the study of better optimization techniques. In particular, even if we were able to perfectly estimate the gradient, policy optimization would still be difficult due to the geometry of the objective function used in RL.

Specifically, our experiments bring to light two issues unique to policy optimization. Firstly, given that we are optimizing probability distributions, many reparameterizations can result in the same distribution. This results in objective functions that are especially susceptible to having flat regions and difficult geometries (Figure 4b, S15c). There are a few solutions to this issue: As pointed out by Kakade (2001), methods based on the natural gradient are well equipped to deal with plateaus induced by probability distributions. Alternatively, given the empirical success of natural policy gradient inspired methods like TRPO and surrogate objective methods like PPO suggests that these techniques are well motivated in RL (Schulman et al., 2015a; 2017b; Rajeswaran et al., 2017). Such improvements are orthogonal to the noisy gradient problem and suggest that making policy optimization easier is a fruitful line of research.

Secondly, the landscape we are optimizing is problem dependent and is particularly surprising in our work. Given that the mechanics of many MuJoCo tasks are very similar, our observations on Hopper and HalfCheetah are vastly different. If our analysis was restricted to just Hopper and Walker, our conclusions with respect to entropy would have been different. This presents a challenge for both studying and designing optimization techniques.

The MuJoCo environments considered here are deterministic given the random seed: An interesting and important extension would be to investigate other sources of noise and in general answering *What aspects of the environment induce difficult objectives?* Our proposed method will likely be useful in answering at least a few such questions.

Sampling strategies. Our learning curves are not surprising under the mainstream interpretation of entropy regularization: A small value of σ will not induce a policy that adequately "explores" the whole range of available states and actions. However, our results on HalfCheetah tell a different story: All values of σ converged to policy with similar final reward (Figure 5c).

Our combined results from curvature analysis and linear interpolations (Figure 6 and 7) have shown that the geometry of the objective function is linked to the entropy of the policy being optimized. Thus using a more stochastic policy, in this case, induced by entropy regularization, facilitates the use of a larger learning rate and might provide more directions of improvement.

Our results should hold for any technique that works by increasing policy entropy or collects data in a more uniform way[7]. Investigating how other *directed* or *structured* sampling schemes impact the landscape will be useful to inform the design of new techniques. We conjecture that the ultimate effect of these techniques in RL is to make the objective function smoother and thus easier to optimize.

Smoothing. Finally, our experimental results make one suggestion: Smoothing can help learning. Therefore, *How can we leverage these observations to make new algorithms?* The smoothing effect of entropy regularization, if decayed over the optimization procedure, is akin to techniques that start optimizing, easier, highly smoothed objectives and then progressively making them more complex (Chapelle & Wu, 2010; Gulcehre et al., 2017). Perhaps some work should be directed on alternate smoothing techniques: Santurkar et al. (2018) suggests that techniques like batch normalization also smooth the objective function and might be able to replicate some of the performance benefits. In the context of RL, Q-value smoothing has been explored by Nachum et al. (2018); Fujimoto et al. (2018) that resulted in performance gains for an off-policy policy optimization algorithm.

In summary, our work has provided a new tool for and highlighted the importance of studying the underlying optimization landscape in direct policy optimization. We have shown that these optimization landscapes are highly environment-dependent making it challenging to come up with general purpose optimization algorithms. We show that optimizing policies with more entropy results in a smoother objective function that can be optimized with a larger learning rate. Finally, we identify a myriad of future work that might be of interest to the community with significant impact.

Acknowledgements

The authors would like to thank Robert Dadashi and Saurabh Kumar for their consistent and useful discussions through-

[7]For example, the analysis in Section 3.2 encompasses both the "naive" and "proper" entropy bonuses from Schulman et al. (2017a).

out this project; Riashat Islam, Pierre Thodoroff, Nishanth Anand, Yoshua Bengio, Sarath Chandar and the anonymous reviewers for providing detailed feedback on a draft of this manuscript; Prakash Panangaden and Clara Lacroce for a discussion on interpretations of the proposed visualization technique; Pierre-Luc Bacon for teaching the first author an alternate way to prove the policy gradient theorem; and Valentin Thomas, Pierre-Antoine Manzagol, Subhodeep Moitra, Utku Evci, Marc G. Bellemare, Fabian Pedregosa, Pablo Samuel Castro, Kelvin Xu and the entire Google Brain Montreal team for thought-provoking feedback during meetings.

References

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Chapelle, O. and Wu, M. Gradient descent optimization of smoothed information retrieval metrics. *Information retrieval*, 13(3):216–235, 2010.

Chaudhari, P., Choromanska, A., Soatto, S., LeCun, Y., Baldassi, C., Borgs, C., Chayes, J., Sagun, L., and Zecchina, R. Entropy-sgd: Biasing gradient descent into wide valleys. *International Conference on Learning Representations*, 2017.

Chou, P.-W., Maturana, D., and Scherer, S. Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution. In *International Conference on Machine Learning*, pp. 834–843, 2017.

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems*, pp. 2933–2941, 2014.

Draxler, F., Veschgini, K., Salmhofer, M., and Hamprecht, F. A. Essentially no barriers in neural network energy landscape. *International Conference on Machine Learning*, 2018.

Fort, S. and Scherlis, A. The goldilocks zone: Towards better understanding of neural network loss landscapes. *Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)*, 2018.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. *International Conference on Machine Learning*, 2018.

Fujita, Y. and Maeda, S.-i. Clipped action policy gradient. *International Conference on Machine Learning*, 2018.

Goodfellow, I. J., Vinyals, O., and Saxe, A. M. Qualitatively characterizing neural network optimization problems. *International Conference on Learning Representations*, 2015.

Greensmith, E., Bartlett, P. L., and Baxter, J. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov): 1471–1530, 2004.

Gulcehre, C., Moczulski, M., Visin, F., and Bengio, Y. Mollifying networks. *International Conference on Learning Representations 2017*, 2017.

Henderson, P., Romoff, J., and Pineau, J. Where did my optimum go?: An empirical analysis of gradient descent optimization in policy gradient methods. *European Workshop on Reinforcement Learning*, 2018.

Ilyas, A., Engstrom, L., Santurkar, S., Tsipras, D., Janoos, F., Rudolph, L., and Madry, A. Are deep policy gradient algorithms truly policy gradient algorithms? *arXiv preprint arXiv:1811.02553*, 2018.

Kakade, S. A natural policy gradient. In *Neural Information Processing Systems*, volume 14, pp. 1531–1538, 2001.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. *International Conference on Learning Representations 2017*, 2017.

Khetarpal, K., Ahmed, Z., Cianflone, A., Islam, R., and Pineau, J. Re-evaluate: Reproducibility in evaluating reinforcement learning algorithms. *2nd Reproducibility in Machine Learning Workshop at ICML 2018*, 2018.

Konda, V. R. and Tsitsiklis, J. N. Actor-critic algorithms. In *Advances in neural information processing systems*, pp. 1008–1014, 2000.

Le Cun, Y., Kanter, I., and Solla, S. A. Eigenvalues of covariance matrices: Application to neural-network learning. *Physical Review Letters*, 66(18):2396, 1991.

Li, C., Farkhoor, H., Liu, R., and Yosinski, J. Measuring the intrinsic dimension of objective landscapes. *International Conference on Learning Representations (ICLR)*, 2018a.

Li, H., Xu, Z., Taylor, G., and Goldstein, T. Visualizing the loss landscape of neural nets. *International Conference on Learning Representations, Workshop Track*, 2018b.

Mandt, S., Hoffman, M. D., and Blei, D. M. Stochastic gradient descent as approximate bayesian inference. *The Journal of Machine Learning Research*, 18(1):4873–4907, 2017.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Nachum, O., Norouzi, M., Tucker, G., and Schuurmans, D. Smoothed action value functions for learning gaussian policies. *International Conference on Machine Learning*, 2018.

Neu, G., Jonsson, A., and Gómez, V. A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*, 2017.

Rajeswaran, A., Lowrey, K., Todorov, E. V., and Kakade, S. M. Towards generalization and simplicity in continuous control. In *Advances in Neural Information Processing Systems*, pp. 6550–6561, 2017.

Santurkar, S., Tsipras, D., Ilyas, A., and Madry, A. How does batch normalization help optimization?(no, it is not about internal covariate shift). *arXiv preprint arXiv:1805.11604*, 2018.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015a.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

Schulman, J., Chen, X., and Abbeel, P. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*, 2017a.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017b.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, 2014.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems (IROS)*, pp. 5026–5033. IEEE, 2012.

Tucker, G., Bhupatiraju, S., Gu, S., Turner, R. E., Ghahramani, Z., and Levine, S. The mirage of action-dependent baselines in reinforcement learning. *International Conference on Learning Representations (Workshop Track)*, 2018.

Watkins, C. J. and Dayan, P. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Williams, R. J. and Peng, J. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Zhao, T., Hachiya, H., Niu, G., and Sugiyama, M. Analysis and improvement of policy gradient estimation. In *Advances in Neural Information Processing Systems*, pp. 262–270, 2011.