Sample Complexity of Estimating the Policy Gradient for Nearly Deterministic Dynamical Systems

Osbert Bastani
University of Pennsylvania, USA

Abstract

Reinforcement learning is a promising approach to learning robotics controllers. It has recently been shown that algorithms based on finite-difference estimates of the policy gradient are competitive with algorithms based on the policy gradient theorem. We propose a theoretical framework for understanding this phenomenon. Our key insight is that many dynamical systems (especially those of interest in robotics control tasks) are *nearly deterministic*—i.e., they can be modeled as a deterministic system with a small stochastic perturbation. We show that for such systems, finite-difference estimates of the policy gradient can have substantially lower variance than estimates based on the policy gradient theorem. Finally, we empirically evaluate our insights in an experiment on the inverted pendulum.

1 Introduction

The policy gradient is the workhorse of modern reinforcement learning. In particular, most state-of-the-art reinforcement learning algorithms aim to learn a control policy π_θ by estimating the policy gradient—i.e., the gradient $\nabla_\theta J(\theta)$ of the expected cumulative reward $J(\theta)$ with respect to the parameters θ of the control policy—in one of two ways: (i) numerically, e.g., using a finite-difference approximation (Kober et al., 2013; Mania et al., 2018), or (ii) by using the policy gradient theorem (Sutton et al., 2000) to construct estimates (Silver et al., 2014; Schulman et al., 2015a,b, 2017). However, there has been little work on theoretically understanding the tradeoffs between these two approaches, and our work aims to help fill this gap.

We are interested in applications to robotics control, which typically have continuous state and action spaces (Collins et al., 2005; Abbeel et al., 2007; Levine et al., 2016). For example, reinforcement learning can be used to learn controllers when the dynamics are unknown (Abbeel et al., 2007; Ross and Bagnell, 2012; Akametalu et al., 2014; Berkenkamp et al., 2017; Johannink et al., 2018). Understanding sample complexity is especially important in this application, since the goal is for robots to be able to learn based on real world experience, which can be very costly to obtain. Furthermore, having a theoretical understanding of sample complexity is important for developing safe reinforcement learning algorithms (Akametalu et al., 2014; Berkenkamp et al., 2017; Dean et al., 2018b).

We argue that *near determinism* is an important characteristic of dynamical systems relevant to robotics. More precisely, we study settings where the noise in the dynamics is “small” (i.e., sub-Gaussian with small constant). This setting captures robotics tasks such as grasping (Andrychowicz et al., 2018), quadcopters (Akametalu et al., 2014), walking (Collins et al., 2005), and driving (Montemerlo et al., 2008), where the dynamics are primarily deterministic but include small perturbations such as wind, friction, or slippage. We discuss this claim in detail below.

Main results. In the context of near determinism, we analyze the sample complexity of various algorithms for estimating the policy gradient $\nabla_\theta J(\theta)$. We study three algorithms: (i) an algorithm based on finite-differences, (ii) an algorithm based on the policy gradient theorem, and (iii) a model-based algorithm (i.e., it knows the system dynamics) that uses backpropagation to estimate the policy gradient. The model-based algorithm represents the best convergence rate we can hope to achieve using only random samples of the noise. We give details on these algorithms in Section 3.

Our key parameter of interest is the sub-Gaussian pa-

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108. Copyright 2020 by the author(s).

rameter σ_ζ of the system noise ζ, which is small for nearly deterministic systems. Here, we also consider dependences on the estimation error ϵ and the dimension d_Θ of the parameter space; we state theorems giving dependences on all parameters in Section 4. We prove the following bounds on the sample complexity n (i.e., the number of samples needed to get at most ϵ error with probability at least $1-\delta$):

- For the model-based estimate, $n = \tilde{\Theta}(\sigma_\zeta^2/\epsilon^2)$.
- For the finite-differences estimate, $n = \tilde{\Theta}(\sigma_\zeta^2 d_\Theta/\epsilon^4)$.
- For the estimate based on the policy gradient theorem, $n = \tilde{O}(1/\epsilon^2)$ and $n = \tilde{\Omega}(1/\epsilon)$.

Our key finding is that while both the model-based and finite-difference estimates become small as σ_ζ becomes small, the estimate based on the policy gradient theorem does not. Thus, for nearly deterministic dynamical systems, finite-difference algorithms perform significantly better. However, this improvement comes at a price—n depends on d_Θ, and furthermore quadratically more samples are needed to get to the same estimation error.

Finally, we focus on how many samples are needed to estimate the policy gradient on a single step. This understanding is already useful for applications such as safe reinforcement learning. Nevertheless, we discuss how our results connect to the problem of optimizing $J(\theta)$ in Section 4.

Motivation for near determinism. A common approach in robotics is to model the robot dynamics as deterministic (Levinson et al., 2011; Kuindersma et al., 2016). To account for stochasticity, either a stabilizing controller such as a PID controller is used (Levinson et al., 2011), or the robot's trajectory is replanned at every step (Kwon et al., 1983; Kuindersma et al., 2016). An alternative approach is to assume that the dynamics are deterministic plus a bounded perturbation at each step, and then use robust control (Akametalu et al., 2014). Both approaches implicitly assume that the deterministic portion of the dynamics are a good approximation of the full dynamics. In general, most systems that have been successfully studied in reinforcement learning are nearly deterministic, including Atari games (Mnih et al., 2015), MuJoCo benchmarks (Todorov et al., 2012; Levine and Koltun, 2013), and simulated grasping tasks (Andrychowicz et al., 2018).

More importantly, we believe that it will be challenging to increase the sample efficiency of reinforcement learning in systems where the noise is high. Indeed, our analysis shows that noise can be greatly amplified by the dynamics, so if the noise is large, we believe there is very little hope for sample-efficient reinforcement learning. In these settings, we may need to rely on techniques such as transfer learning (Taylor and Stone, 2009), meta-learning (Finn et al., 2017), or learning to plan (Tamar et al., 2016) to achieve low sample complexity.

Related work. The theoretical work in reinforcement learning algorithms has primarily focused on Q-learning (Watkins and Dayan, 1992; Kearns and Singh, 2002; Kakade et al., 2003; Jin et al., 2018), especially for Markov decision processes (MDPs) with finite state and action spaces. There has been some work on understanding the sample complexity of reinforcement learning with function approximation—e.g., for fitted value iteration (Munos and Szepesvári, 2008), for fitted policy iteration (Antos et al., 2008; Lazaric et al., 2012; Farahmand et al., 2015, 2016), fitted Q-iteration (Tosatto et al., 2017), and the TD(0) algorithm (Dalal et al., 2018). For robotics tasks, where state and action spaces are typically continuous, the most successful approaches are predominantly based on policy gradient estimation (Collins et al., 2005; Kober et al., 2013), for which there has been relatively little work. In this direction, (Kakade et al., 2003) has analyzed the sample complexity of algorithms based on the policy gradient theorem, but they do not study the dependence of the sample complexity on the magnitude of the system noise. Furthermore, their work assumes finite state and action spaces and bounded rewards, and they do not consider finite-difference algorithms.

There has been work characterizing a key design choice of finite-difference algorithms—i.e., the distribution of perturbations used to numerically estimate the policy gradient (Roberts and Tedrake, 2009). They measure the performance of different choices using the signal-to-noise ratio. In contrast, our goal is to understand the sample complexity of different algorithms for nearly deterministic systems.

There has recently been work on understanding the sample complexity of learning controllers; however, they focus on linear dynamical systems, and on different algorithms—e.g., temporal difference learning (Tu and Recht, 2018b) or model-based algorithms (Dean et al., 2018a; Tu and Recht, 2018a). There has also been work in this setting studying the possibility of reducing variance by controlling the noise in the dynamics (Malik et al., 2019); in the setting we study, we cannot control the noise.

There has been recent work comparing approaches based on exploration in the action space (based on the policy gradient theorem) to exploration in the state space (based on finite difference methods) (Vemula et al., 2019). Our focus on nearly deterministic systems enables us to obtain qualitatively different insights compared to theirs. In particular, they find that approaches based on finite differences perform better for problems with a long time horizon. However, we analyze a more realistic model, and find that this insight no longer holds. Instead, approaches based on finite differences outperform approaches based on the policy gradient theorem for nearly deterministic systems.

Our analysis differs in three key ways. First, they assume an upper bound $J(\theta) \leq J_{\max}$, which is a very strong assumption. Second, their analysis does not model stochastic dynamics. Instead, they assume that $J(\theta)$ is deterministic, but they can only obtain observations $J(\theta)+\zeta$, where ζ is i.i.d. noise. In contrast, our analysis considers both stochastic dynamics, as well as how noise is propagated through the dynamics. This distinction substantially complicates our analysis, but is necessary for us to understand the implications of near determinism (since we need to understand how the dynamics can amplify noise). Finally, unlike their work, we provide lower bounds for our main results.

Connection to optimizing $J(\theta)$. Estimating the policy gradient can be used in conjunction with stochastic gradient descent to optimize $J(\theta)$. There is a large body of work on understanding the convergence rate of stochastic gradient descent (Robbins and Monro, 1985; Spall et al., 1992; Bottou and Bousquet, 2008; Moulines and Bach, 2011), of which policy gradient algorithms are a special case. Indeed, (Vemula et al., 2019) uses these techniques to bound the complexity of optimizing $J(\theta)$.

There are several reasons why we focus on understanding the sample complexity of a single gradient step rather than the sample complexity of optimization. First, they rely on the strong assumption that $J(\theta)$ is bounded—i.e., $J(\theta) \leq J_{\max}$ for some $J_{\max} \in \mathbb{R}_+$. Second, it would be much more difficult to derive lower bounds on optimization—existing lower bounds are for the setting where the objective f coming from a very general function family, and these bounds may not apply when f is restricted to be the objective of a reinforcement learning problem. In contrast, for sample complexity, we derive matching (or almost matching) upper and lower bounds. Third, the sample complexity of estimating $\nabla_\theta J(\theta)$ is of intrinsic interest—for example, it is an important prerequisite for safe reinforcement learning algorithms (Akametalu et al., 2014; Berkenkamp et al., 2017; Dean et al., 2018b). Finally, focusing on sample complexity simplifies our key insight. In particular, consider the the completely deterministic setting—optimizing a deterministic function using gradient descent may still take many steps, but "estimating" the gradient only requires a single sample.

Additionally, we note that sample complexity is directly related to the complexity of optimizing $J(\theta)$. In particular, the bounds in Vemula et al. (2019) all depend directly on the variance σ^2 of the observations $J(\theta)+\zeta$. Our proof bounds the sample complexity of estimating $\nabla J(\theta)$ by bounding the sub-Gaussian parameter of $J(\theta)$, which is an upper bound on the variance of $J(\theta)$. Thus, smaller sample complexity translates to smaller complexity of optimizing $J(\theta)$.

Finally, our focus on estimating the gradient does not address the problem of exploration. In terms of optimization, gradient estimates can be used in conjunction with gradient descent to efficiently find local minima, whereas exploration is needed to find global minima. Understanding the sample complexity of exploration is an important but orthogonal problem that we leave to future work.

2 Preliminaries

We consider a dynamical system with states $S \subseteq \mathbb{R}^{d_S}$, actions $A \subseteq \mathbb{R}^{d_A}$, and transitions

$$s_{t+1} = f(s_t, a_t) + \zeta_t \quad \text{where} \quad \zeta_t \sim p(\zeta),$$

where $f : S \times A \to S$ is deterministic and $\zeta \in \mathbb{R}^{d_S}$ is a random perturbation. We consider deterministic control policies $\pi_\theta : S \to A$ with parameters $\theta \in \Theta \subseteq \mathbb{R}^{d_\Theta}$. Except in the case of the model-based policy gradient algorithm, we assume that both f and p are unknown. We separate f from p since we are interested in settings where ζ is small. Also, we that assume ζ_t is independent of s_t and a_t. This assumption enables us to substantially simplify the model-based policy gradient (since we avoid taking a derivatives of p), and it also simplifies our analyses of other algorithms.

We are interested in controlling the system over a finite horizon $T \in \mathbb{G}$—given a reward function $R : S \times A \to \mathbb{R}$, the goal is to find the policy π_θ that maximizes the expected cumulative reward

$$J(\theta) = \mathbb{E}_{p_\theta(\alpha)}\left[\sum_{t=0}^{T-1} R(s_t, a_t)\right],$$

where $p_\theta(\alpha)$ is the distribution over rollouts $\alpha = ((s_0, a_0), ..., (s_{T-1}, a_{T-1}))$ when using π_θ, and where we assume the initial state $s_0 \in S$ is deterministic and known. Note that α is determined by θ and $\vec{\zeta} = (\zeta_0, ..., \zeta_{T-1})$, so an expectation over $p_\theta(\alpha)$ is equivalent to one over $p(\vec{\zeta})$. We are interested in estimating the policy gradient

$$D(\theta) = \nabla_\theta J(\theta)$$

so we can perform gradient ascent on θ. As usual, let

$$Q_\theta^{(t)}(s,a) = \mathbb{E}_{p(\zeta)}\left[R(s,a) + V_\theta^{(t+1)}(f(s,a)+\zeta)\right]$$
$$V_\theta^{(t)}(s) = Q_\theta^{(t)}(s,\pi_\theta(s)),$$

for $t \in \{0,1,...,T-1\}$, where $V_\theta^{(T)}(s) = 0$, denote the Q function and value function, respectively (Sutton and Barto, 2018). In particular, $J(\theta) = V_\theta^{(0)}(s_0)$.

Remark 2.1. Our results straightforwardly extend to dynamical systems with time varying dynamics and rewards. Also, we can relax our assumption that the initial state s_0 is deterministic—i.e., to handle an initial state distribution p_0, we can modify the dynamics on the first step to be $s_1 = s_0 + \zeta_0$, where $s_0 = 0$ and $\zeta_0 \sim p_0$. Furthermore, our results can be extended to the case where the noise ζ appears nonlinearly in the transitions, as long as it can be reparameterized (Kingma and Welling, 2014)—i.e., the transitions can be written in the form $s' = f(s,\zeta,a)$, where $\zeta \sim p(\zeta)$ i.i.d. for some $p(\zeta)$. We require that f is Lipschitz in ζ. Most kinds of noise considered in practice can be expressed in this form, though it may not satisfy the Lipschitz condition. Finally, our results can be extended to handle Martingale difference noise sequences by using the Azuma-Hoeffding inequality in place of the Hoeffding inequality.

3 Policy Gradient Algorithms

We now describe the policy gradient estimation algorithms that we consider.

Model-based policy gradient. When f is known, we can estimate the policy gradient as

$$\nabla_\theta J(\theta) = \mathbb{E}_{p(\vec{\zeta})}[\nabla_\theta \hat{J}(\theta;\vec{\zeta})]$$
$$\hat{J}(\theta;\vec{\zeta}) = \sum_{t=0}^{T-1} R(s_t,a_t).$$

since a rollout α is determined by $\vec{\zeta}$. In particular, we have estimator $\nabla_\theta J(\theta) \approx \hat{D}_{\text{MB}}(\theta)$, where

$$\hat{D}_{\text{MB}}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \hat{J}(\theta;\vec{\zeta}^{(i)})$$

where $\vec{\zeta}^{(i)} \sim p(\vec{\zeta})$ i.i.d. for $i \in [n]$.

Policy gradient theorem. The policy gradient theorem is formulated for stochastic policies—i.e., $\pi_\theta(a \mid s)$ is the probability of taking action a in state s. We assume a distribution $p_\xi(\xi)$ of action perturbations that does not depend on θ—i.e., $a_t = \pi_\theta(s_t) + \xi_t$, where $\xi_t \sim p_\xi(\xi)$. Then, we have $\tilde{\pi}_\theta(a \mid s) = p_\xi(a - \pi_\theta(s))$. The following are the modified Q and value functions:

$$\tilde{Q}_\theta^{(t)}(s,a) = R(s,a) + \mathbb{E}_{p(\zeta)}\left[\tilde{V}_\theta^{(t+1)}(f(s,a)+\zeta)\right]$$
$$\tilde{V}_\theta^{(t)}(s) = \mathbb{E}_{\tilde{\pi}_\theta(a|s)}\left[\tilde{Q}_\theta^{(t)}(s,a)\right],$$

where $\tilde{V}_\theta^{(T)}(s) = 0$ as before. Then, the following is the policy gradient theorem (Sutton et al., 2000):

Theorem 3.1. *Letting $\tilde{p}_\theta(\alpha)$ be the distribution over rollouts when using $\tilde{\pi}_\theta$, we have*

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tilde{p}_\theta(\alpha)}\left[\sum_{t=0}^{T-1} \tilde{Q}_\theta^{(t)}(s_t,a_t)\nabla_\theta \log \tilde{\pi}_\theta(a_t \mid s_t)\right].$$

The key challenge to using Theorem 3.1 to estimate $\nabla_\theta J(\theta)$ is to estimate $\tilde{Q}^{(t)}(s_t,a_t)$. The simplest approach is to estimate it using a single rollout (Williams, 1992):

$$\tilde{Q}_\theta^{(t)}(s,a) = \mathbb{E}_{\tilde{p}_\theta(\alpha)}\left[\hat{Q}_\theta^{(t)}(\alpha)\right]$$
$$\hat{Q}_\theta^{(t)}(\alpha) = \sum_{i=t}^{T-1} R(s_i,a_i).$$

A common technique to reduce variance is to normalize $\tilde{Q}_\theta^{(t)}(s,a)$ by subtracting the value function (Schulman et al., 2015b). In particular, the advantage function $\tilde{A}_\theta^{(t)}(s,a) = \tilde{Q}_\theta^{(t)}(s,a) - \tilde{V}_\theta^{(t)}(s)$ measures the advantage of using action a instead of using $\tilde{\pi}_\theta$ in state s at time t. Then, we have

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tilde{p}_\theta(\alpha)}\left[\sum_{t=0}^{T-1} \tilde{A}_\theta^{(t)}(\alpha)\nabla_\theta \log \tilde{\pi}_\theta(a_t \mid s_t)\right].$$

Unlike $\tilde{Q}_\theta^{(t)}$, we cannot estimate $\tilde{A}_\theta^{(t)}$ using a single rollout. One approach is to estimate $f_\phi^{(t)}(s) \approx \tilde{V}_\theta^{(t)}(s)$, and then estimate $\tilde{A}_\theta^{(t)}$ using $f_\phi^{(t)}$. We assume that our estimate of $\tilde{V}_\theta^{(t)}$ is exact—in particular, we consider the following estimator $\nabla_\theta J(\theta) \approx \hat{D}_{\text{PG}}(\theta)$:

$$\hat{D}_{\text{PG}}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\sum_{t=0}^{T-1} \hat{A}_\theta^{(t)}(\alpha^{(i)})\nabla_\theta \log \tilde{\pi}_\theta(a_t^{(i)} \mid s_t^{(i)})$$
$$\hat{A}_\theta^{(t)}(\alpha) = \hat{Q}_\theta^{(t)}(\alpha) - \tilde{V}_\theta^{(t)}(s_t),$$

where $\alpha^{(i)} \sim p_\theta(\alpha)$ i.i.d. for each $i \in [n]$.

Remark 3.2. A common approach is to use an estimate $\hat{Q}_\theta^{(t)}(s,a)$ of the Q function in place of $\hat{Q}_\theta^{(t)}(\alpha)$. This approach reduces variance, but may introduce bias. For instance, for dynamical systems with continuous actions, the deterministic policy gradient (DPG) algorithm uses this approach Silver et al. (2014). We consider the algorithm described above for two reasons. First, our focus is on estimating the policy gradient, rather than understanding the sample complexity of Q-learning, which is required to analyze DPG. Second, it is hard to prove bounds for DPG since it relies on the *derivative* of the Q function, which cannot be bounded without additional assumptions. For example, suppose we train a random forest $\hat{Q}_\theta^{(t)}(s,a)$. Even if this model achieves achieves good accuracy, its gradient would be zero nearly everywhere since this model is piecewise constant; thus, it would not be useful in the context of the DPG algorithm.

Finite-difference policy gradient. We can use finite-differences to estimate $\nabla_\theta J(\theta)$.

Theorem 3.3. *For any $f : \mathcal{X} \to \mathbb{R}$ (where $\mathcal{X} \subseteq \mathbb{R}^d$) where ∇f is $L_{\nabla f}$-Lipschitz continuous,* [1]

$$\nabla_x f(x) = \sum_{k=1}^{d} \frac{f(x+\lambda\nu^{(k)}) - f(x-\lambda\nu^{(k)})}{2\lambda} \cdot \nu^{(k)} + \Delta$$

where $\nu^{(k)} = \delta_k$ (where δ_k is the Kronecker delta), and $\Delta \in \mathbb{R}$ satisfies $\|\Delta\| \le L_{\nabla f} d\lambda$.

We give a proof in Appendix E. Then, the finite difference approximation of the policy gradient is

$$\nabla_\theta J(\theta) \approx \sum_{k=1}^{d_\Theta} \frac{J(\theta+\lambda\nu^{(k)}) - J(\theta-\lambda\nu^{(k)})}{2\lambda} \cdot \nu^{(k)}.$$

We can estimate $J(\theta)$ using samples $\vec{\zeta} \sim p(\vec{\zeta})$, which yields the estimator $\nabla_\theta J(\theta) \approx \hat{D}_{\text{FD}}(\theta)$, where

$$\hat{D}_{\text{FD}}(\theta) = \sum_{k=1}^{d_\Theta} \Bigg[\frac{\frac{1}{n}\sum_{i=1}^n \hat{J}(\theta+\lambda\nu^{(k)}; \vec{\zeta}^{(k,i)})}{2\lambda} - \frac{\frac{1}{n}\sum_{j=1}^n \hat{J}(\theta-\lambda\nu^{(k)}; \vec{\eta}^{(k,j)})}{2\lambda} \Bigg] \cdot \nu^{(k)}$$

where $\vec{\zeta}^{(k,i)}, \vec{\eta}^{(k,j)} \sim p(\vec{\zeta})$ i.i.d. for $k \in [m]$ and $i,j \in [n]$. Note that we use separate samples $\zeta^{(k,i)}$ and $\eta^{(k,j)}$ to estimate $J(\theta+\lambda\nu^{(k)})$ and $J(\theta-\lambda\nu^{(k)})$, respectively. If we are using a simulator, then we can reduce variance by using the same samples to estimate both terms.

Remark 3.4. Typically, rather than choose a fixed set of basis vectors $\nu^{(1)}, ..., \nu^{(k)}$, finite-difference algorithms choose random vectors from a spherically symmetric distribution—e.g., $\nu \sim \mathcal{N}(0, \sigma^2 I_{d_\theta})$ (Spall et al., 1992; Mania et al., 2018). Our choice of a fixed basis simplifies our analysis.

[1] We assume the L_2 norm throughout.

4 Main Results

Sample complexity. Recall that the policy gradient $\nabla_\theta J(\theta)$ must be estimated from sampled rollouts $\zeta \sim p_\theta(\zeta)$. Our goal is to understand the tradeoffs in sample complexity of estimating $\nabla_\theta J(\theta)$ between various different reinforcement learning algorithms.

Definition 4.1. Let X be a random vector, and let $\hat{\mu}_X^{(n)} = n^{-1}\sum_{i=1}^n x^{(i)}$, where $x^{(1)}, ..., x^{(n)} \sim p_X(x)$ i.i.d. The *sample complexity* of $n_X(\epsilon, \delta)$ of X is the smallest $n \in \mathbb{N}$ such that

$$\Pr_{x^{(1)},...,x^{(n)} \sim p_X(x)} \left[\|\hat{\mu}_X^{(n)}\| \ge \epsilon \right] \le \delta.$$

We are interested in the sample complexity $n_{\hat{D}}$ of $\hat{D}(\zeta) - \nabla_\theta J(\theta)$, where $\hat{D}(\zeta)$ is an estimate of $\nabla_\theta J(\theta)$ using a single rollout $\zeta \sim p_\theta(\zeta)$.

Assumptions. We let $f_\theta(s) = f(s, \pi_\theta(s))$ and $R_\theta(s) = R(s, \pi_\theta(s))$. Similarly, for a stochastic policy $\pi_\theta(s) + \xi$ (where $\xi \sim p(\xi)$), we let $\tilde{f}_\theta(s,\xi) = f(s, \pi_\theta(s) + \xi)$ and $\tilde{R}_\theta(s) = \mathbb{E}_{p(\xi)}[R(s, \pi_\theta(s) + \xi)]$. Next, to ensure convergence, we make regularity assumptions about the dynamics and our control policy; see Appendix F & G for definitions.

Assumption 4.2. *We assume that f, R, π_θ, f_θ, $\tilde{f}_\theta$, R_θ and $\tilde{R}_\theta$ are Lipschitz continuous and are twice continuously differentiable with Lipschitz continuous first derivative.*

Remark 4.3. This standard assumption is needed to ensure that we can estimate the gradient using finite differences. It is somewhat strong—e.g., it rules out commonly used quadratic rewards. In practice, the state space is often compact, in which case the Lipschitz continuity assumption becomes redundant. However, we cannot handle discontinuous rewards or dynamics (including piecewise constant rewards). In these cases, the policy gradient may diverge near the discontinuities; thus, the sample complexity of estimating this gradient may diverge as well. In principle, we could handle discontinuities as long as the policy visits these discontinuities with zero probability.

Finally, for any function h, we let L_h denote its Lipschitz constant and $\bar{L}_h = \max\{L_{\nabla h}, L_h, 1\}$.

Assumption 4.4. *We assume that $p(\zeta)$ is σ_ζ-subgaussian.*

This assumption is required for proving concentration—e.g., it is typically assumed in the context of safe reinforcement learning (Akametalu et al., 2014; Berkenkamp et al., 2017). In practice, perturbations due to noise are often bounded (which implies the noise is sub-Gaussian), especially for our setting of interest—e.g., forces due to wind, friction, or slippage have bounded magnituded. We are interested in settings where σ_ζ is small.

Definition 4.5. A system is *nearly deterministic* if $\sigma_\zeta \ll 1$.

In particular, we are interested in the dependence of the sample complexity on σ_ζ.

Main theorems. For the model-based policy gradient, we have:

Theorem 4.6. *For $\delta \leq 1/2$, the sample complexity of $\hat{D}_{MB}(\theta) - \nabla_\theta J(\theta)$ satisfies*

$$\sqrt{n_{MB}(\epsilon,\delta)} = \tilde{O}\left(T^8 \bar{L}_{R_\theta} \bar{L}_{f_\theta}^{5T} \sigma_\zeta d_A/\epsilon\right)$$
$$\sqrt{n_{MB}(\epsilon,\delta)} = \tilde{\Omega}\left(T \bar{L}_{f_\theta}^{T-3} \sigma_\zeta/\epsilon\right).$$

For the policy gradient based on Theorem 3.1:

Theorem 4.7. *For the choice $p_\xi(\xi) = \mathcal{N}(\xi \mid \vec{0}, \sigma_\zeta^2 I_{d_A})$, $\hat{D}_{PG}(\theta) - \nabla_\theta J(\theta)$ has sample complexity*

$$\sqrt{n_{PG}(\epsilon,\delta)} = \tilde{O}\left(T^6 (L_R + L_{\tilde{R}_\theta}) \bar{L}_f L_\pi \bar{L}_{\tilde{f}_\theta}^T d^4/\epsilon\right),$$

where $d = \max\{d_S, d_A\}$, for ϵ sufficiently small—i.e., $\epsilon = \Omega(T^6(L_R + L_{\tilde{R}_\theta})\bar{L}_f L_\pi \bar{L}_{\tilde{f}_\theta}^T d^4)$. Next,

$$\sqrt{n_{PG}(\epsilon,\delta)} \geq \sqrt{n_\xi\left(\epsilon/\bar{L}_{f_\theta}^{T-2}, \delta\right)}$$
$$\sqrt{n_{PG}(\epsilon,\delta)} = \tilde{\Omega}\left(\min\left\{\bar{L}_{f_\theta}^{T/2}/\epsilon^{1/2}, 1/\delta^{1/2}\right\}\right).$$

The first lower bound holds for any $p_\xi(\xi)$ that is everywhere differentiable on $\mathbb{R}$ and satisfies $\lim_{\xi\to\pm\infty} \xi \cdot p_\xi(\xi) = 0$, where n_ξ is the sample complexity of estimating $\mathbb{E}_{p_\xi(\xi)}[\xi \cdot \nabla_\xi \log p_\xi(\xi)]$ using samples from p_ξ. The second lower bound holds for $p_\xi(\xi) = \mathcal{N}(0, \sigma_\xi^2)$, for any $\sigma_\xi \in \mathbb{R}_+$.

We have shown two lower bounds—one for an arbitrary distribution p_ξ (in terms of a sample complexity n_ξ related to p_ξ), and one for the specific choice where p_ξ is Gaussian (as is the case in our upper bound). Also, note that our upper bound depends on choosing the action noise to have variance σ_ζ. In principle, the first lower bound holds even if p_ξ depends on the problem parameters; however, then n_ξ may depend on these parameters as well. The second lower bound is independent of the the action noise σ_ξ, so it holds even if σ_ξ depends on the problem parameters.

Remark 4.8. Note that the upper and lower bounds have a gap on the order of $\epsilon^{1/2}$. We believe that this gap is due to limitations in our analysis. In particular, our lower bounds depend on a lower bound on the tail of the χ_n^2 distribution, which has exponential tails. In contrast, our other lower bounds depend on Gaussian tails, which are doubly exponential. Intuitively, since the χ_n^2 distribution has a longer tail, it should not have lower sample complexity.

Remark 4.9. Note that the second lower bound contains a dependence on $\delta^{-1/2}$, which is unusual. However, this term only has a role if the first term in the minimum is very large. Furthermore, the first term depends as usual on $\log(1/\delta)$ (which is not shown since we omit log factors).

Remark 4.10. Actor-critic approaches reduce variance by using function approximation to obtain lower variance estimates of the advantage $\tilde{A}_\theta^{(t)}$ (Schulman et al., 2015b). However, our lower bounds hold even if the advantage is known exactly. Thus, while actor-critic approaches can reduce variance, they do not affect our main insight that these estimates remain noisy for nearly deterministic dynamical systems.

For the finite-difference policy gradient:

Theorem 4.11. *The sample complexity of $\hat{D}_{FD}(\theta) - \nabla_\theta J(\theta)$ satisfies*

$$\sqrt{n_{FD}(\epsilon,\delta)} = \tilde{O}\left(T^9 \bar{L}_{R_\theta}^2 \bar{L}_{f_\theta}^{5T} \sigma_\zeta d_A^2 \sqrt{d_\Theta}/\epsilon^2\right)$$
$$\sqrt{n_{FD}(\epsilon,\delta)} = \tilde{\Omega}\left(T \bar{L}_{f_\theta}^{3(T-3)} \sigma_\zeta d_\Theta/\epsilon^2\right).$$

The first bound (i.e., the upper bound) holds for a choice $\lambda = O(\epsilon/T^5 \bar{L}_{R_\theta} \bar{L}_{f_\theta}^{4T} d_A)$. The second bound (i.e., the lower bound) holds for any $\lambda \in \mathbb{R}_+$, $\epsilon \leq 1$, and $\delta \leq 1/2$,

Note that our upper bound is for the choice $\lambda = O(\epsilon)$, but our lower bound holds for arbitrary λ.

Remark 4.12. In an abuse of notation, in Theorem 4.11, we have ignored the fact that n_{FD} must always be at least $2d_\Theta$; in particular, it does not go to zero as σ_ζ goes to zero. This discrepancy in Theorem 4.11 arises because there is an implicit assumption we use when inverting Hoeffding's inequality that $n \geq 1$—more precisely, Hoeffding's inequality gives a bound of the form

$$\Pr[\hat{\mu}_X^{(n)} \geq \epsilon] \leq \delta,$$

where $\hat{\mu}_X^{(n)}$ is an estimate of $\mu_X = \mathbb{E}[X]$ using n samples, and $\delta \geq e^{-n\epsilon^2/(2\sigma^2)}$. Solving for n yields $n \geq 2\sigma^2 \log(1/\delta)/\epsilon^2$. However, if $\sigma = 0$, then δ is not well defined, so it does not mean we can get an estimate of μ_X using $n = 0$ samples; instead, we need to take $n = 1$. In our proof of Theorem 4.11, we apply Hoeffding's inequality $2d_\Theta$ times (since we estimate the gradient of each component separately), so we need $n \geq 2d_\Theta$.

Proof strategy. We give a high-level overview of our proof strategy, focusing on Theorem 4.6. Our proof proceeds in two steps. First, we prove an upper bound

$$|\hat{D}_{\mathrm{MB}}(\theta) - \nabla_\theta J(\theta)| \le AE + B, \qquad (1)$$

where $E = T^{-1}\sum_{t=0}^{T-1}\|\zeta_t\|$ and $A, B \in \mathbb{R}_+$ do not depend on $\vec{\zeta}$. This step uses induction based on the recursive structure of V_θ. Second, we prove Lemma G.7; we state a simplified version:

Lemma 4.13. *Let X be a σ_X-sub-Gaussian random vector over $\mathbb{R}^d$, and let Y be a random vector over $\mathbb{R}^{d'}$ satisfying $\|Y\| \le A\|X\|_1 + B$, where $A, B \in \mathbb{R}_+$. Then Y is σ_Y-sub-Gaussian, where $\sigma_Y = \tilde{O}(A\sigma_X d + B)$.*

Combined with (1), we conclude that $\hat{D}_{\mathrm{MB}}(\theta) - \nabla_\theta J(\theta)$ is sub-Gaussian, from which we can use Hoeffding's inequality (see Lemma G.3) to complete the proof. For the lower bound, we construct a system where $J(\theta)$ is Gaussian. The proof of Theorem 4.7 follows similarly, except we need to use analogous results for sub-exponential random variables. In particular, we prove Lemma H.7, an analog of Lemma G.7. The proof of Theorem 4.11 also follows similarly, but we need to account for the bias in the finite-difference estimate of $\nabla_\theta J(\theta)$ from Theorem 3.3.

5 Discussion

Dependence on σ_ζ. Both n_{MB} and n_{FD} scale linearly in σ_ζ. Thus, the corresponding algorithms perform very well when σ_ζ is small. In contrast, n_{PG} does not become small when σ_ζ becomes small. Intuitively, if p_ξ is wide, then the action noise adds uncertainty to $\hat{D}_{\mathrm{PG}}(\theta)$. On the other hand, if p_ξ is narrow, then $\nabla_\theta \log \tilde{\pi}_\theta(a \mid s) = \nabla_\theta \log p_\xi(a - \pi_\theta(s))$ becomes large—in particular, p_ξ must change rapidly for some values of ξ, and must have large gradient at such values of ξ.

A key point is that in the first lower bound for n_{PG} (i.e., for arbitrary p_ξ), even though we do not know its explicit dependence on ϵ, δ, T, and $\bar{L}_{f_\theta}$, we know that it is completely independent of σ_ζ. Thus, regardless of how p_ξ is chosen (e.g., even if it chosen based on the problem parameters), the sample complexity does not become small as σ_ζ becomes small.

Full determinism ($\sigma_\zeta = 0$). When $\sigma_\zeta = 0$, we have $n_{\mathrm{MB}} = 1$ (i.e., we only need a single sample to estimate $\nabla_\theta J(\theta)$) and $n_{\mathrm{FD}} = 2d_\Theta$ (i.e., we need two samples to estimate the derivative of each parameter, taking λ small enough to get ϵ error). For the case of n_{PG}, our lower bound in Theorem 4.7 still holds—the dynamical system we use to obtain the lower bound has no noise in the dynamics. In particular, a large number of samples are still needed to obtain good estimates (i.e., possibly exponential in T).

Dependence on ϵ. Both n_{MB} and n_{PG} depend quadratically on ϵ (ignoring the gap between the upper and lower bounds for n_{PG}). In contrast, n_{FD} depends quartically on ϵ. This gap arises because according to Theorem 3.3, the finite-differences error of $\hat{D}_{\mathrm{FD}}(\theta)$ (assuming there is no noise) depends linearly on λ. Thus, we must choose $\lambda = O(\epsilon)$ to obtain error at most ϵ. If the dynamical system and control policy are both linear, then this error goes away, so the dependence on ϵ becomes quadratic.

Dependence on d_Θ. Only n_{FD} depends on d_Θ—whereas the other two algorithms make use of the fact that we can compute $\nabla_\theta \pi_\theta$, the finite-difference approximation ignores this ability.

Dependence on T. All of the sample complexities depend exponentially on T. As we show in our lower bounds, this dependence is unavoidable—it arises from the fact that the dynamics cause the state (and therefore the rewards) to grow exponentially large in T. A common assumption made in prior work is that the rewards are bounded uniformly by $R_{\max} \in \mathbb{R}_+$ (Kearns and Singh, 2002; Kakade et al., 2003). Intuitively, our results indicate that without stronger assumptions, $R_{\max}$ may be exponentially large. In practice, rewards for continuous control tasks are often quadratic, and can indeed be exponentially in magnitude.

An important aspect is that estimation is substantially easier when the current policy is good. In our bounds, the base of the exponential dependence is always $\bar{L}_{f_\theta}$. If the initial policy π_θ provides relatively stable control, then we may expect that $L_{f_\theta} \le 1$—i.e., the states remain bounded in magnitude. Then, we have $\bar{L}_{f_\theta} = 1$, so our bounds no longer depend exponentially on T. This insight suggests the importance of good initialization for fast estimation.

Indeed, policy gradient estimators can have high variance in practice. As an example, consider the cart-pole problem with continuous action space, with random initial state and where the reward function is the negative distance to origin. We empirically estimated that the MSE of the model-based policy gradient estimator using $n = 1$ on a randomly initialized policy for this benchmark is 3.5×10^7. This error is substantially reduced when the policy is stable—for a trained cart-pole policy, we estimate that the MSE of the model-based policy gradient estimator is just 5.2×10^{-2}.

6 Experiments

We empirically evaluated the effect of σ_ζ on the performance of the different algorithms.

Dynamical system. We use the inverted pendulum (Tedrake, 2018) (specifically, using the dynamics from OpenAI Gym (Brockman et al., 2016)), which

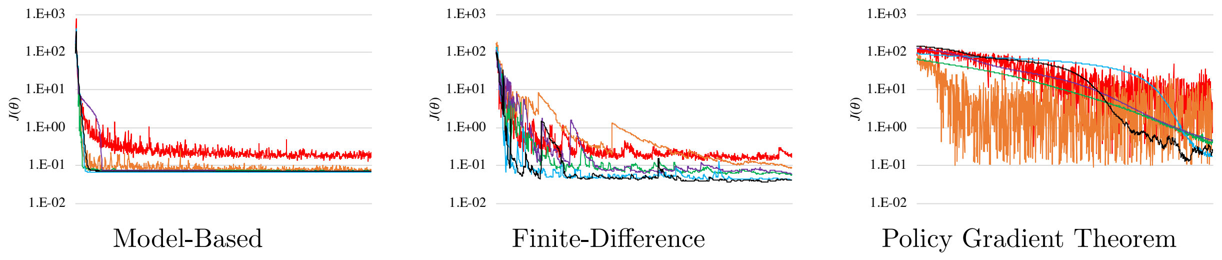

Figure 1: The cumulative expected reward $J(\theta)$ as a function of the number of gradient steps $i \in \{1, 2, ..., 1000\}$. The black, purple, blue, green, orange, and red curves correspond to $\sigma_\zeta = \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$, respectively. The x-axis is the number of gradient steps taken (or equivalently, the number of rollouts), and the y-axis is $-J(\theta)$.

has state space $S = \mathbb{R}^2$ (i.e., angle ϑ and angular velocity ω) and actions $A = \mathbb{R}$ (i.e., applied torque). Letting f be the (deterministic) pendulum dynamics, we consider the system $s_{t+1} = f(s_t, a_t) + \zeta_t$, where $\zeta_t \sim \mathcal{N}(0, \sigma_\zeta^2)$ i.i.d. We use the rewards

$$R((\vartheta, \omega), a) = -(w_\vartheta \cdot (\vartheta - \vartheta_0)^2 + w_\omega \cdot \omega^2 + w_a \cdot a^2),$$

where ϑ_0 is the angle corresponding to the upright position, and $w_\vartheta = 1$, $w_\omega = 10^{-1}$, and $w_a = 10^{-2}$. Our goal is to control the system over a horizon of $T = 50$ steps, from a fixed start state $s_0 = (\vartheta_0', 0)$, where $\vartheta_0' = 0.05$. For the control policy, we used a neural network π_θ with a single hidden layer with 100 neurons, ReLU activations, and linear outputs. As usual, we randomly initialize the weights; to reduce variance, we initialized the policy to have a reasonably high reward by running our model-based algorithm until $J(\theta) \geq -100$.

Algorithms. We use stochastic gradient descent in conjunction with each of the three estimation algorithms. On each gradient step, we use a single sample to estimate the gradient, and we take 1000 gradient steps. We modify the finite-difference algorithm to use a single random sample $\nu \sim \text{Uniform}(S^{d_\Theta - 1})$ (i.e., the uniform distribution on the unit sphere in $\mathbb{R}^{d_\Theta}$), rather than summing over the d_Θ basis vectors $\nu^{(k)}$. This choice may improve the dependence of the sample complexity on d_Θ; however, it should not affect dependence on σ_ζ, which is our parameter of interest.

For the algorithm based on the policy gradient theorem, we use action noise $\xi \sim \mathcal{N}(0, \sigma_\xi I_{d_A})$. For each choice of σ_ζ, we used cross-validation to identify the optimal hyperparameters: the learning rate υ (for all algorithms), the parameter λ (for the finite-differences algorithm), and the action noise σ_ξ (for the algorithm based on the policy gradient theorem).

Results. Average the results of each algorithm over 20 runs; the algorithms have very high variance, so we discard runs that do not converge. In Figure 1, we show the learning curves for $\sigma_\zeta \in \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ (i.e., $J(\theta)$ as a function of the number of gradient steps). The darker colors correspond to smaller noise. We show enlarged versions of these plots in Appendix I.

Note that unlike the other two algorithms, the finite-difference algorithm actually uses 2000 sampled rollouts (since it uses two per gradient step). However, this detail does not affect our insights regarding the relative convergence rate of different algorithms for different σ_ζ.

Our key finding is that the learning curves for the model-based and finite-differences are ordered based on the choice of σ_ζ—i.e., the curves tend to converge more quickly for smaller choices of σ_ζ. This effect is most apparent in the curves for the finite-differences algorithms, where curves for smaller σ_ζ (black and blue) converge much faster than those for larger σ_ζ (red and orange). In contrast, the learning curves for the policy gradient based algorithm do not have strong dependence on σ_ζ. For example, the fastest curve to converge (at least initially) for the policy gradient based algorithm is for our second-largest choice $\sigma_\zeta = 10^{-2}$ (orange), whereas the slowest to converge is for $\sigma_\zeta = 10^{-4}$ (blue). These results mirror our theoretical insights.

Finally, as expected, the model-based algorithm converges most quickly, followed by the finite-differences and policy gradient theorem based algorithms.

7 Conclusion

We have analyzed the sample complexity of algorithms for estimating the policy gradient for nearly deterministic dynamical systems. Future work includes leveraging these results in safe reinforcement learning algorithms, and understanding the sample complexity of optimizing $J(\theta)$.

Acknowledgements

This work was supported by NSF CCF-1910769.

References

Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Y Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Advances in neural information processing systems*, pages 1–8, 2007.

Anayo K Akametalu, Shahab Kaynama, Jaime F Fisac, Melanie Nicole Zeilinger, Jeremy H Gillula, and Claire J Tomlin. Reachability-based safe learning with gaussian processes. In *CDC*, pages 1424–1431. Citeseer, 2014.

Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *arXiv preprint arXiv:1808.00177*, 2018.

András Antos, Csaba Szepesvári, and Rémi Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1): 89–129, 2008.

Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *Advances in Neural Information Processing Systems*, pages 908–918, 2017.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Seok-Ho Chang, Pamela C Cosman, and Laurence B Milstein. Chernoff-type bounds for the gaussian error function. *IEEE Transactions on Communications*, 59(11):2939–2944, 2011.

Steve Collins, Andy Ruina, Russ Tedrake, and Martijn Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307(5712):1082–1085, 2005.

Gal Dalal, Balázs Szörényi, Gugan Thoppe, and Shie Mannor. Finite sample analyses for td (0) with function approximation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. Regret bounds for robust adaptive control of the linear quadratic regulator. In *NIPS*, 2018a.

Sarah Dean, Stephen Tu, Nikolai Matni, and Benjamin Recht. Safely learning to control the constrained linear quadratic regulator. *arXiv preprint arXiv:1809.10121*, 2018b.

Amir-massoud Farahmand, Doina Precup, André M.S. Barreto, and Mohammad Ghavamzadeh. Classification-based approximate policy iteration. *IEEE Transactions on Automatic Control*, 2015.

Amir-massoud Farahmand, Mohammad Ghavamzadeh, Csaba Szepesvári, and Shie Mannor. Regularized policy iteration with nonparametric function spaces. *JMLR*, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, pages 4868–4878, 2018.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. *arXiv preprint arXiv:1812.03201*, 2018.

Sham Machandranath Kakade et al. *On the sample complexity of reinforcement learning*. PhD thesis, University of London London, England, 2003.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots*, 40(3):429–455, 2016.

W Hi Kwon, AM Bruckstein, and T Kailath. Stabilizing state-feedback design via the moving horizon method. *International Journal of Control*, 37(3): 631–643, 1983.

Tor Lattimore and Csaba Szepesvári. Bandit algorithms. *preprint*, 2018.

Alessandro Lazaric, Mohammad Ghavamzadeh, and Rémi Munos. Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, 13(Oct):3041–3074, 2012.

Sergey Levine and Vladlen Koltun. Guided policy search. In *International Conference on Machine Learning*, pages 1–9, 2013.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 163–168. IEEE, 2011.

Dhruv Malik, Ashwin Pananjady, Kush Bhatia, Koulik Khamaru, Peter L Bartlett, and Martin J Wainwright. Derivative-free methods for policy optimization: Guarantees for linear quadratic systems. In *AISTATS*, 2019.

Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search provides a competitive approach to reinforcement learning. In *NIPS*, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518 (7540):529, 2015.

Michael Montemerlo, Jan Becker, Suhrid Bhat, Hendrik Dahlkamp, Dmitri Dolgov, Scott Ettinger, Dirk Haehnel, Tim Hilden, Gabe Hoffmann, Burkhard Huhnke, et al. Junior: The stanford entry in the urban challenge. *Journal of field Robotics*, 25(9): 569–597, 2008.

Eric Moulines and Francis R Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems*, pages 451–459, 2011.

Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(May):815–857, 2008.

Herbert Robbins. A remark on stirling's formula. *The American mathematical monthly*, 62(1):26–29, 1955.

Herbert Robbins and Sutton Monro. A stochastic approximation method. In *Herbert Robbins Selected Papers*, pages 102–109. Springer, 1985.

John W Roberts and Russ Tedrake. Signal-to-noise ratio analysis of policy gradient algorithms. In *Advances in neural information processing systems*, pages 1361–1368, 2009.

Stephane Ross and J Andrew Bagnell. Agnostic system identification for model-based reinforcement learning. In *ICML*, 2012.

Walter Rudin et al. *Principles of mathematical analysis*, volume 3. McGraw-hill New York, 1976.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015a.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *ICLR*, 2015b.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *ICML*, 2014.

James C Spall et al. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE transactions on automatic control*, 37(3):332–341, 1992.

Karl Stromberg. *Probability for analysts*. CRC Press, 1994.

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

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

Aviv Tamar, Yi Wu, Garrett Thomas, Sergey Levine, and Pieter Abbeel. Value iteration networks. In *Advances in Neural Information Processing Systems*, pages 2154–2162, 2016.

Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685, 2009.

Russ Tedrake. *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation*. 2018. URL `http://underactuated.mit.edu/`.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE, 2012.

Samuele Tosatto, Matteo Pirotta, Carlo D'Eramo, and Marcello Restelli. Boosted fitted q-iteration. In *ICML*, 2017.

Stephen Tu and Benjamin Recht. The gap between model-based and model-free methods on the linear quadratic regulator: An asymptotic viewpoint. *arXiv preprint arXiv:1812.03565*, 2018a.

Stephen Tu and Benjamin Recht. Least-squares temporal difference learning for the linear quadratic regulator. In *ICML*, 2018b.

Anirudh Vemula, Wen Sun, and J. Andrew Bagnell. Contrasting exploration in parameter and action space: A zeroth order optimization perspective. In *AISTATS*, 2019.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Martin J. Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*. Cambridge University Press, 2019.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

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