535514: Reinforcement Learning
Lecture 12 — Value Function Approximation A2C Algorithm, and Optimality of PG

Ping-Chun Hsieh

April 1, 2024

Recall: Value Function Approximation (VFA)

- To scale up the model-free methods, function approximation is commonly used to learn value functions
- Idea: Approximate a value function by a parametric function

$$s \longrightarrow \boxed{\mathbf{w}} \longrightarrow \hat{V}(s; \mathbf{w}) \approx V^{\pi}(s)$$

$$s, a \longrightarrow \boxed{\mathbf{w}} \longrightarrow \hat{Q}(s, a; \mathbf{w}) \approx Q^{\pi}(s, a)$$

$$s \longrightarrow \boxed{\mathbf{w}} \longrightarrow \begin{array}{c} \hat{Q}(s, a_1; \mathbf{w}) \\ \vdots \\ \hat{Q}(s, a_{|\mathscr{A}|}; \mathbf{w}) \end{array}$$

($\mathbf{w}$ updated via MC or TD)

- Motivation: Generalize from <u>seen states</u> to <u>unseen states</u>

How to Quantify the Accuracy of VFA?

- For each state s, the squared error between $V^{\pi}(s)$ and $\hat{V}(s;\mathbf{w})$:

$$\left(V^{\pi}(s) - \hat{V}(s;\mathbf{w})\right)^2$$

- To jointly consider all states, we use mean squared error (MSE):

$$F(\mathbf{w}) := \sum_{s \in \mathcal{S}} \rho(s)\left(V^{\pi}(s) - \hat{V}(s;\mathbf{w})\right)^2$$

  - $\rho(s)$ are the weights for mixing the MSE of the states
  - If $\rho(s)$ is a probability distribution, the objective function becomes:

$$F(\mathbf{w}) := \mathbb{E}_{s \sim \rho(s)}\left[\left(V^{\pi}(s) - \hat{V}(s;\mathbf{w})\right)^2\right]$$

Some Natural Choices of $\rho(s)$

- For continuing environments:

1. choose $\rho(s) \equiv d_\mu^\pi(s) := \mathbb{E}_{s_0 \sim \mu}\left[(1-\gamma)\sum_{t=0}^{\infty} \gamma^t P(s_t = s \mid s_0, \pi)\right]$

2. choose $\rho(s) \equiv$ undiscounted stationary state distribution

- For episodic environments:

1. choose $\rho(s) \equiv d_\mu^\pi(s) := \dfrac{\mathbb{E}_{s_0 \sim \mu}\left[\sum_{t=0}^{T} \gamma^t P(s_t = s \mid s_0, \pi)\right]}{\text{normalization constant}}$

- Remark: $\rho(s)$ usually corresponds to the sampling strategy (will be discussed momentarily)

Value Function Approximation via GD

- Goal: find $\mathbf{w}$ that minimizes MSE between $V^{\pi}(s)$ and $\hat{V}(s;\mathbf{w})$

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \underbrace{\mathbb{E}_{s\sim\rho(s)}\left[\left(V^{\pi}(s) - \hat{V}(s;\mathbf{w})\right)^2\right]}_{=:F(\mathbf{w})}$$

- Suppose: We are given an oracle for querying $V^{\pi}(s)$
- Iterative GD update:

$$\begin{aligned}\mathbf{w}_{k+1} &= \mathbf{w}_k - \alpha_k \nabla_{\mathbf{w}} F(\mathbf{w}) \\ &= \mathbf{w}_k - \alpha_k \mathbb{E}_{s\sim\rho(s)}\left[\left(V^{\pi}(s) - \hat{V}(s;\mathbf{w})\right) \nabla_{\mathbf{w}} \hat{V}(s;\mathbf{w})\right]\end{aligned}$$

- GD finds a local minimum under proper step sizes α_k (Why?)

Value Function Approximation via SGD

- Iterative GD update:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \mathbb{E}_{s\sim\rho(s)}\Big[\big(V^\pi(s) - \hat{V}(s;\mathbf{w}_k)\big)\,\nabla_{\mathbf{w}}\hat{V}(s;\mathbf{w}_k)\Big]$$

- Iterative SGD update by the sampled state $S \sim \rho$:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k\Big[\big(V^\pi(S) - \hat{V}(S;\mathbf{w}_k)\big)\,\nabla_{\mathbf{w}}\hat{V}(S;\mathbf{w}_k)\Big]$$

- The SGD update converges to a <u>local minimum</u> under stochastic approximation conditions of α_k (even for NN)

In practice, we don't have an oracle for $V^{\pi}(s)$.

What shall we do?

Monte-Carlo Value Function Approximation

- Recall: Iterative SGD update by the sampled state $S \sim \rho$:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \left[\left(V^\pi(S) - \hat{V}(S; \mathbf{w}_k) \right) \nabla_{\mathbf{w}} \hat{V}(S; \mathbf{w}_k) \right]$$

- Idea: Use MC, i.e. sample return G_t to estimate $V^\pi(S)$

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \left[\left(G_t - \hat{V}(S; \mathbf{w}_k) \right) \nabla_{\mathbf{w}} \hat{V}(S; \mathbf{w}_k) \right]$$

G_t is an unbiased estimate of $V^\pi(S)$

- Equivalent to *supervised learning* with (noisy) training data as

$$(s_0, G_0), (s_1, G_1), \cdots, (s_t, G_t) \cdots$$

Monte-Carlo Value Function Approximation (Cont.)

- First-Visit MC Value Function Approximation:

Step 1: Initialize $\mathbf{w} = 0$ and $k = 1$

Step 2: Sample $\tau_k = (s_0, a_0, r_1, \cdots, s_{L_k-1}, a_{L_k-1}, r_{L_k}) \sim P_\mu^{\pi_\theta}$

For each step of the current episode $t = 0, 1, \cdots L_k - 1$

If First visit to state s in episode k

$$G_t(s) = \sum_{i=t+1}^{L_k} r_i$$

$$\Delta \mathbf{w}_k \leftarrow \Delta \mathbf{w}_k + \alpha_k \Big[\Big(G_t(s) - \hat{V}(S; \mathbf{w}_k) \Big) \nabla_{\mathbf{w}} \hat{V}(S; \mathbf{w}_k) \Big]$$

$$\mathbf{w}_{k+1} \leftarrow \mathbf{w}_k + \Delta \mathbf{w}_k, \;\; k \leftarrow k + 1$$

- Convergence to a local minimum can be achieved with proper step sizes α_k, for general function approximators (Why?)

Temporal-Difference Value Function Approximation

- Recall: Iterative SGD update by the sampled state $S \sim \rho$:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \Big[\big(V^{\pi}(S) - \hat{V}(S; \mathbf{w}_k) \big) \nabla_{\mathbf{w}} \hat{V}(S; \mathbf{w}_k) \Big]$$

- Idea: Use bootstrapping (e.g.TD(0)) to estimate $V^{\pi}(S)$

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \Big[\big(r + \gamma \hat{V}(S'; \mathbf{w}_k) - \hat{V}(S; \mathbf{w}_k) \big) \nabla_{\mathbf{w}} \hat{V}(S; \mathbf{w}_k) \Big]$$

$r + \gamma \hat{V}(S'; \mathbf{w}_k)$ is a <u>biased</u> estimate of $V^{\pi}(S)$

- Equivalent to *supervised learning* with (noisy) training data as

$$(s_0, r_1 + \gamma \hat{V}(s_1; \mathbf{w})), \cdots, (s_t, r_{t+1} + \gamma \hat{V}(s_{t+1}; \mathbf{w})) \cdots$$

- This can be easily extended to the more general TD(λ)

TD Value Function Approximation (Cont.)

- TD(0) Value Function Approximation:

Step 1: Initialize $\mathbf{w} = 0$ and $k = 1$

Step 2: Sample $\tau_k = (s_0, a_0, r_1, \cdots, s_{L_k-1}, a_{L_k-1}, r_{L_k}) \sim P_\mu^{\pi_\theta}$

For each step of the current episode $t = 0,1,\cdots L_k - 1$

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha \left[\left(r_{t+1} + \gamma \hat{V}(s_{t+1}; \mathbf{w}) - \hat{V}(s_t; \mathbf{w}) \right) \nabla_{\mathbf{w}} \hat{V}(s_t; \mathbf{w}) \right]$$

Put Everything Together:

PG + TD + VFA = Advantage Actor-Critic (A2C)

Advantage Actor-Critic (A2C) via TD

- An example of actor-critic algorithm:
  - Critic: estimate V^{π_θ} by TD(0) bootstrapping
  - Actor: updates policy parameters θ by policy gradient
- Advantage Actor-Critic (A2C):

Step 1: Initialize θ_0 and step size α

Step 2: Sample a trajectory $\tau = (s_0, a_0, r_1, \cdots) \sim P_\mu^{\pi_\theta}$

For each step of the current trajectory $t = 0,1,2,\cdots$

$$\Delta\theta_k \leftarrow \Delta\theta_k + \alpha\gamma^t \left(r_t + \gamma\hat{V}(s_{t+1}) - \hat{V}(s_t)\right) \nabla_\theta \log \pi_\theta(a_t \,|\, s_t)$$

Update value function $\hat{V}(s_t)$ by TD(0)

$$\theta_{k+1} \leftarrow \theta_k + \Delta\theta_k$$

A2C With Value Function Approximation

- $\hat{V}_w(s)$ is learned by value function approximation (e.g. by using a neural network or linear combinations of features)

- Advantage Actor-Critic (A2C) with Value Function Approximation:

Step 1: Initialize θ_0, w_0 and step sizes α_θ, α_w

Step 2: Sample a trajectory $\tau = (s_0, a_0, r_1, \cdots) \sim P_\mu^{\pi_\theta}$

For each step of the current trajectory $t = 0,1,2,\cdots$

$$\Delta\theta_k \leftarrow \Delta\theta_k + \alpha_\theta \gamma^t \left(r_t + \gamma \hat{V}_{w_k}(s_{t+1}) - \hat{V}_{w_k}(s_t)\right) \nabla_\theta \log \pi_\theta(a_t | s_t)$$

$$\Delta w_k \leftarrow \Delta w_k + \alpha_w \left(r_t + \gamma \hat{V}_{w_k}(s_{t+1}) - \hat{V}_{w_k}(s_t)\right) \nabla_w \hat{V}_w(s_t)|_{w=w_k}$$

$$\theta_{k+1} \leftarrow \theta_k + \Delta\theta_k, w_{k+1} \leftarrow w_k + \Delta w_k$$

Mnih et al., “Asynchronous Methods for Deep Reinforcement Learning”, ICML 2016

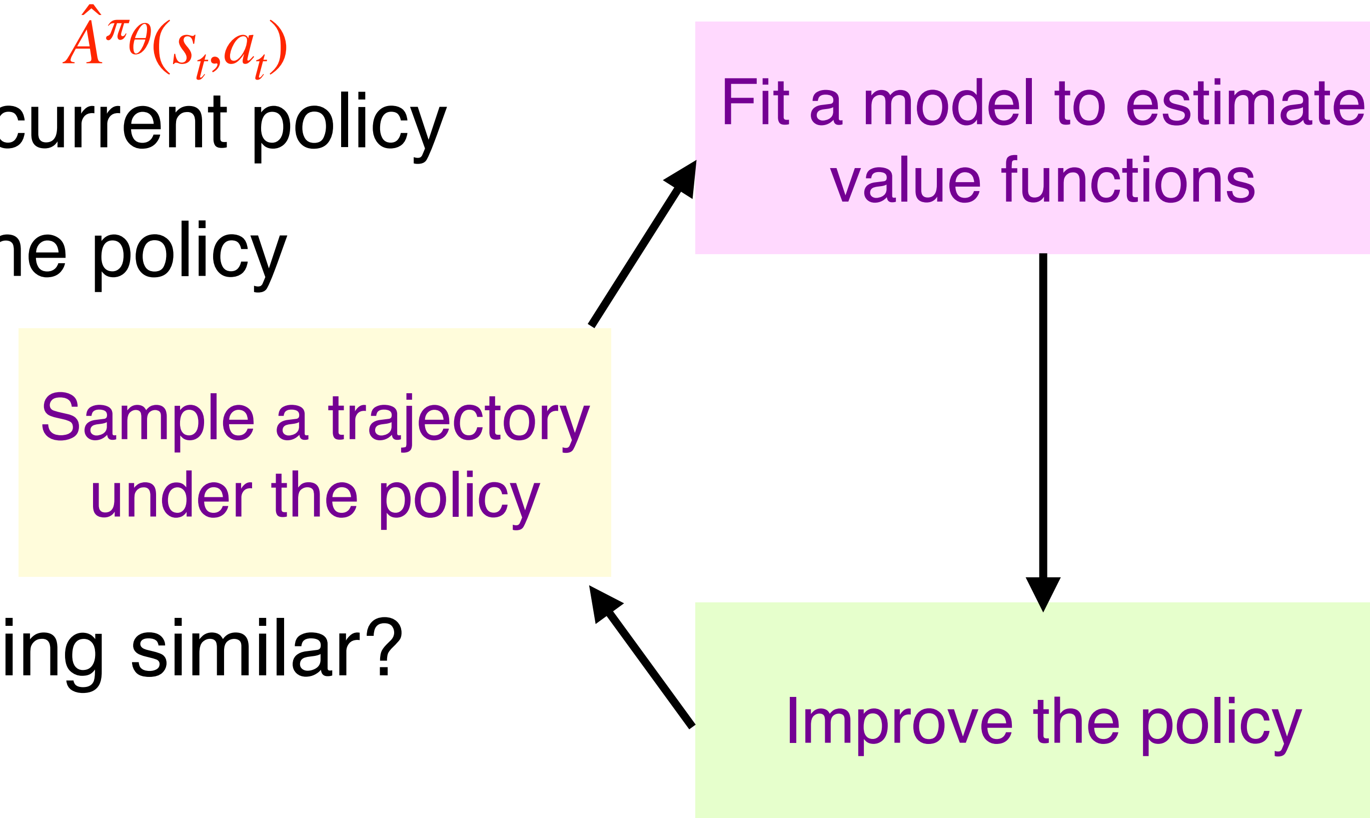
Another Interpretation of A2C

- In A2C, the policy parameters θ are updated as:

$$\Delta\theta_k \leftarrow \Delta\theta_k + \alpha_\theta \gamma^t \underbrace{\left(\overbrace{r_t + \gamma \hat{V}_{w_k}(s_{t+1})}^{\hat{Q}^{\pi_\theta}(s_t,a_t)} - \hat{V}_{w_k}(s_t)\right)}_{\hat{A}^{\pi_\theta}(s_t,a_t)} \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

Step 1: Estimate $A^{\pi_\theta}(s_t, a_t)$ for the current policy

Step 2: Use $\hat{A}^{\pi_\theta}(s_t, a_t)$ to improve the policy



- Question: Have we seen anything similar?

  Policy iteration!

Step 1: Policy evaluation for the current policy (i.e. find $Q^\pi(s, a)$)

Step 2: Policy improvement based on Bellman optimality equations

(Slide Credit: Sergey Levine)

A2C ≈ Policy Iteration, But With Slight Difference

Policy iteration:

Step 1: Policy evaluation for the current policy (i.e. find $Q^{\pi}(s, a)$)

Step 2: Policy improvement based on Bellman optimality equations

A2C:

Step 1: Estimate $A^{\pi_\theta}(s_t, a_t)$ for the current policy

Step 2: Use $\hat{A}^{\pi_\theta}(s_t, a_t)$ to improve the policy

- Question: Is policy guaranteed to be improved in Step 2 of A2C?

During Lec 6-Lec 12, we have learned many things:

1. PG, Stochastic PG, and REINFORCE
2. Model-Free Prediction
3. Value Function Approximation

Several unanswered questions:

1. Does PG ensure convergence to an optimal policy?

(Our focus today)

2. Do TD(0)/TD(λ) ensure convergence to the true value function?

Next Topic: PG Converges to Optimality

Agarwal, Kakade, Lee, and Mahajan, “On the Theory of Policy Gradient Methods: Optimality, Approximation, and Distributional Shift”, COLT 2020

Mei et al., “On the Global Convergence Rates of Softmax Policy Gradient Methods,” ICML 2020

Breakthrough: Optimality of PG

The “REINFORCE” paper

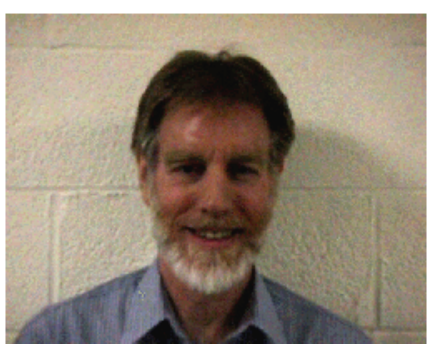
Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning

Ronald J. Williams
College of Computer Science
Northeastern University
Boston, MA 02115

Appears in *Machine Learning, 8*, pp. 229-256, 1992.

Abstract

This article presents a general class of associative reinforcement learning algorithms for connectionist networks containing stochastic units. These algorithms, called REINFORCE algorithms, are shown to make weight adjustments in a direction that lies along the gradient of expected reinforcement in both immediate-reinforcement tasks and certain limited forms of delayed-reinforcement tasks, and they do this without explicitly computing gradient estimates or even storing information from which such estimates could be computed. Specific examples of such algorithms are presented, some of which bear a close relationship to certain existing algorithms while others are novel but potentially interesting in their own right. Also given are results that show how such algorithms can be naturally integrated with backpropagation. We close with a brief discussion of a number of additional issues surrounding the use of such algorithms, including what is known about their limiting behaviors as well as further considerations that might be used to help develop similar but potentially more powerful reinforcement learning algorithms.



Ronald Williams

arXiv 2019 (COLT 2020)

On the Theory of Policy Gradient Methods: Optimality, Approximation, and Distribution Shift

Alekh Agarwal* Sham M. Kakade† Jason D. Lee‡ Gaurav Mahajan§

Abstract

Policy gradient methods are among the most effective methods in challenging reinforcement learning problems with large state and/or action spaces. However, little is known about even their most basic theoretical convergence properties, including: if and how fast they converge to a globally optimal solution or how they cope with approximation error due to using a restricted class of parametric policies. This work provides provable characterizations of the computational, approximation, and sample size properties of policy gradient methods in the context of discounted Markov Decision Processes (MDPs). We focus on both: “tabular” policy parameterizations, where the optimal policy is contained in the class and where we show global convergence to the optimal policy; and parametric policy classes (considering both log-linear and neural policy classes), which may not contain the optimal policy and where we provide agnostic learning results. One central contribution of this work is in providing approximation guarantees that are average case — which avoid explicit worst-case dependencies on the size of state space — by making a formal connection to supervised learning under *distribution shift*. This characterization shows an important interplay between estimation error, approximation error, and exploration (as characterized through a precisely defined condition number).

arXiv 2019 (OR Journal 2023)

Global Optimality Guarantees For Policy Gradient Methods

Jalaj Bhandari and Daniel Russo

Columbia University

Abstract

Policy gradients methods apply to complex, poorly understood, control problems by performing stochastic gradient descent over a parameterized class of polices. Unfortunately, even for simple control problems solvable by standard dynamic programming techniques, policy gradient algorithms face non-convex optimization problems and are widely understood to converge only to a stationary point. This work identifies structural properties – shared by several classic control problems – that ensure the policy gradient objective function has no suboptimal stationary points despite being non-convex. When these conditions are strengthened, this objective satisfies a Polyak-lojasiewicz (gradient dominance) condition that yields convergence rates. We also provide bounds on the optimality gap of any stationary point when some of these conditions are relaxed.

Keywords: Reinforcement learning, policy gradient methods, policy iteration, dynamic programming, gradient dominance.

On the Global Convergence Rates of Softmax Policy Gradient Methods

Jincheng Mei♣♠* Chenjun Xiao♣ Csaba Szepesvári♡♣ Dale Schuurmans♠♣

♣University of Alberta ♡DeepMind ♠Google Research, Brain Team

Abstract

We make three contributions toward better understanding policy gradient methods in the tabular setting. First, we show that with the true gradient, policy gradient with a softmax parametrization converges at a $O(1/t)$ rate, with constants depending on the problem and initialization. This result significantly expands the recent asymptotic convergence results. The analysis relies on two findings: that the softmax policy gradient satisfies a Łojasiewicz inequality, and the minimum probability of an optimal action during optimization can be bounded in terms of its initial value. Second, we analyze entropy regularized policy gradient and show that it enjoys a significantly faster linear convergence rate $O(e^{-c \cdot t})$ toward softmax optimal policy ($c > 0$). This result resolves an open question in the recent literature. Finally, combining the above two results and additional new $\Omega(1/t)$ lower bound results, we explain how entropy regularization improves policy optimization, even with the true gradient, from the perspective of convergence rate. The separation of rates is further explained using the notion of non-uniform Łojasiewicz degree. These results provide a theoretical understanding of the impact of entropy and corroborate existing empirical studies.

methods (Sutton et al., 2000). As an approach to RL, the appeal of policy gradient methods is that they are conceptually straightforward and under some regularity conditions they guarantee monotonic improvement of the value. A secondary appeal is that policy gradient methods were shown to achieve effective empirical performance (e.g., Schulman et al., 2015; 2017).

Despite the prevalence and importance of policy optimization in RL, the theoretical understanding of policy gradient method has, until recently, been severely limited. A key barrier to understanding is the inherent non-convexity of the value landscape with respect to standard policy parametrizations. As a result, little has been known about the global convergence behavior of policy gradient method. Recently, important new progress in understanding the convergence behavior of policy gradient has been achieved. As in this paper we will restrict ourselves to the tabular setting, we analyze the part of the literature that also deals with this setting. While the tabular setting is clearly limiting, this is the setting where so far the cleanest results have been achieved and understanding this setting is a necessary first step towards the bigger problem of understanding RL algorithms. Returning to the discussion of recent work, Bhandari & Russo (2019) showed that, without parametrization, projected gradient ascent on the simplex does not suffer from spurious local optima. In concurrent work, Agarwal et al. (2019) showed that *(i)* without parametrization, projected gradient ascent

Journal of Machine Learning Research 23 (2022) 1-36 Submitted 1/22; Published 8/22

On the Convergence Rates of Policy Gradient Methods

Lin Xiao LINX@FB.COM
Meta AI Research
Seattle, WA 98109, USA

Editor: Alekh Agarwal

Abstract

We consider infinite-horizon discounted Markov decision problems with finite state and action spaces and study the convergence rates of the projected policy gradient method and a general class of policy mirror descent methods, all with direct parametrization in the policy space. First, we develop a theory of weak gradient-mapping dominance and use it to prove sharp sublinear convergence rate of the projected policy gradient method. Then we show that with geometrically increasing step sizes, a general class of policy mirror descent methods, including the natural policy gradient method and a projected Q-descent method, all enjoy a linear rate of convergence without relying on entropy or other strongly convex regularization. Finally, we also analyze the convergence rate of an inexact policy mirror descent method and estimate its sample complexity under a simple generative model.

Keywords: discounted Markov decision problem, policy gradient, gradient domination, policy mirror descent, sample complexity.

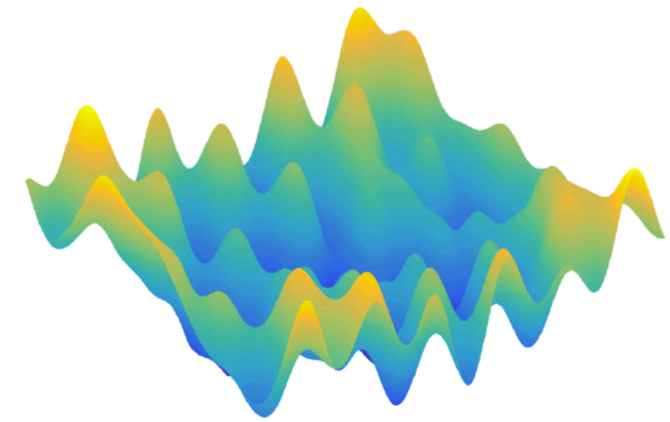
JMLR 2022

ICML 2020

2019

1992

Gradient Descent for General Non-Convex Problems



- Challenges of Non-Convex Problems
  - Bumps and local minima everywhere
  - No efficient first-order methods for convergence to optimum in general
  - The best hope is on convergence to stationary points (i.e. $\nabla f(x) = 0$)
- Convergence of Gradient Descent for Non-Convex & Smooth Functions

(i) $\|\nabla f(x_k)\|_2 \to 0, \quad k \to \infty$ (ii) $\min_{0 \le k \le T} \|\nabla f(x_k)\|_2 = \sqrt{\frac{2L(f(x_0) - f(x^*))}{T}}$

- RL is a Non-Convex Problem in General
  - **As a result, it was believed that PG can only converge to a stationary point (an open problem for almost 30 years)**

Recall: The (Exact) PG Algorithm

- The "Exact PG" algorithm (= *policy gradient + gradient ascent*)

Step 1: Initialize policy parameters θ_0, step size η, and $k = 0$

Step 2: Evaluate π_{θ_k} and get $Q^{\pi_{\theta_k}}(s, a)$, $V^{\pi_{\theta_k}}(s)$, $A^{\pi_{\theta_k}}(s, a)$

Step 3: Update the policy parameters by

$$
\begin{aligned}
\theta_{k+1} &= \theta_k + \eta \cdot \nabla_\theta V^{\pi_\theta}(\mu) \Big|_{\theta=\theta_k} \\
&= \theta_k + \eta \cdot \frac{1}{1-\gamma} \mathbb{E}_{s \sim d_\mu^{\pi_\theta}} \mathbb{E}_{a \sim \pi_\theta(\cdot|s)} \Big[Q^{\pi_\theta}(s, a) \nabla_\theta \log \pi_\theta(a \,|\, s) \Big] \Big|_{\theta=\theta_k} \\
&= \theta_k + \eta \cdot \frac{1}{1-\gamma} \mathbb{E}_{s \sim d_\mu^{\pi_\theta}} \mathbb{E}_{a \sim \pi_\theta(\cdot|s)} \Big[A^{\pi_\theta}(s, a) \nabla_\theta \log \pi_\theta(a \,|\, s) \Big] \Big|_{\theta=\theta_k}
\end{aligned}
$$

(Repeat Steps 2 & 3 until termination)

Nice Properties of Exact PG

(1) Exact PG achieves strict improvement

(2) Exact PG converges to optimality at a rate $O(1/t)$

PG Under Softmax Policy Parametrization

- (P4): $$\nabla_\theta V^{\pi_\theta}(\mu) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\mu^{\pi_\theta}}\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[A^{\pi_\theta}(s,a)\,\nabla_\theta \log \pi_\theta(a\,|\,s)\right]$$

Softmax policies: $\pi_\theta(a|s) = \dfrac{\exp(\theta_{s,a})}{\sum_{a'\in\mathcal{A}}\exp(\theta_{s,a'})}$ $(\theta \in \mathbb{R}^{S\times A})$

Fact: (P4) under tabular softmax policies can be written as

$$\frac{\partial V^{\pi_\theta}(\mu)}{\partial \theta_{s,a}} = \frac{1}{1-\gamma} d_\mu^{\pi_\theta}(s)\pi_\theta(a\,|\,s)A^{\pi_\theta}(s,a)$$

- Question: Can you see any interesting implication?

Performance Difference Lemma

Lemma [Kakade and Langford, ICML 2002]

For any two policies π_{old}, π_{new} and any state distribution μ,

$$V^{\pi_{new}}(\mu) - V^{\pi_{old}}(\mu) = \frac{1}{1-\gamma}\mathbb{E}_{s' \sim d_\mu^{\pi_{new}}}\mathbb{E}_{a' \sim \pi_{new}(\cdot|s')}\left[A^{\pi_{old}}(s', a')\right]$$

Question: What if μ is deterministic?

Question: How to ensure the new policy is better than the old one?

(1) PG Achieves Monotonic Policy Improvement

- **Recall (Partial ordering of policies)**:

$$\pi \geq \pi' \quad \text{if} \quad V^{\pi}(s) \geq V^{\pi'}(s), \forall s$$

- **Theorem (Monotonic Policy Improvement)**:

Under vanilla PG with step size $\eta \leq (1-\gamma)^2/5$, for all k and for all states s and actions a, we have

$$V_{k+1}(s) \geq V_k(s); \quad Q_{k+1}(s,a) \geq Q_k(s,a);$$

Hence, $\pi^{\theta_{k+1}} \geq \pi^{\theta_k}$, for all k

This is a direct result of Performance Difference Lemma

(2) Exact PG Converges to Optimality at a Rate $O(1/t)$

- **Theorem (Convergence Rate of PG)**: [Mei et al., 2020]

  Assume $\mu(s) > 0$ for all s. Under $\eta = \frac{(1-\gamma)^3}{8}$, Exact PG achieves the following

$$V^*(\rho) - V^{\pi_{\theta_t}}(\rho) \leq \left(\frac{16 \cdot S}{\inf_{t \geq 1} \pi_t(a^* \mid s)^2 \cdot (1-\gamma)^6} \cdot \left\| \frac{d_\mu^{\pi^*}}{\mu} \right\|_\infty^2 \cdot \left\| \frac{1}{\mu} \right\|_\infty \right) \cdot \frac{1}{t}$$

- Question: How is this possible? Shouldn't PG get stuck easily?

A Fundamental Idea: *Polyak-Łojasiewicz (PL) Condition* in Non-Convex Optimization

Question: When can GD succeed under non-convex objective functions?

Boris Polyak

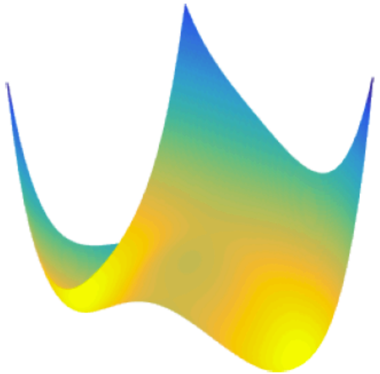
Polyak-Łojasiewicz Condition

Gradient norm Sub-optimality gap

$$\|\nabla f(x)\|^2 \geq c \cdot (f(x) - f(x^*)) \quad \text{for some } c > 0$$

Interpretation: GD will not stop until it reaches an optimal point

Surprise: **PG actually satisfies a PL condition!**



Polyak-Łojasiewicz (PL) Condition in RL

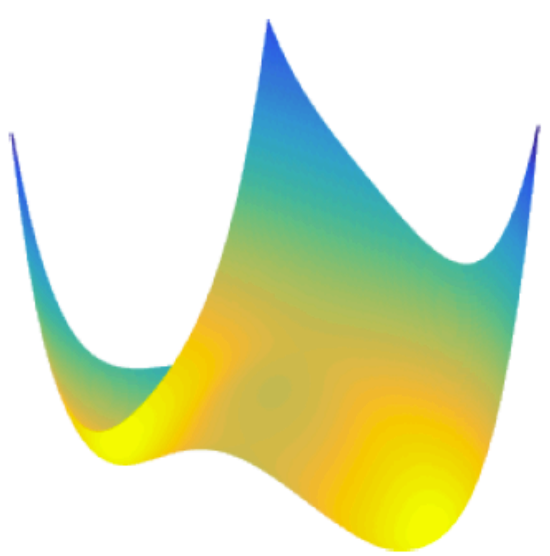
PL Condition for Policy Gradient [Mei et al., ICML 2020]

Gradient norm   Sub-optimality gap

$$\left\| \frac{\partial V^{\pi_\theta}(\mu)}{\partial \theta} \right\|_2 \geq \frac{\min_s \pi_\theta(a^*(s)|s)}{\sqrt{S} \cdot \left\| d_\rho^{\pi^*} / d_\mu^{\pi_\theta} \right\|_\infty} \cdot \left[V^*(\rho) - V^{\pi_\theta}(\rho) \right] .$$

Nuance: PL depends on $\min_{s \in S} \pi_\theta(a^*(s)|s)$, which can be small

Gradient could be extremely small if the current policy π_θ is far from an optimal one



Example: 1-State MDP and PL Condition

- Use 1-state, 2-action MDP to see why PL condition is possible in RL

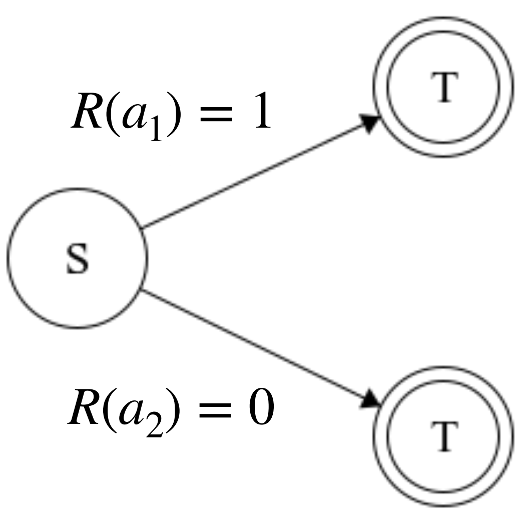
Policy parameters: $\theta \equiv [\theta_{a_1}, \theta_{a_2}]$

Reward function: $R \equiv [R(a_1), R(a_2)]$

Case 2 Case 1

$$\left\|\frac{\partial V^{\pi_\theta}(\mu)}{\partial \theta}\right\|_2 \geq \frac{\min_s \pi_\theta(a^*(s)|s)}{\sqrt{S} \cdot \left\|d_\rho^{\pi^*}/d_\mu^{\pi_\theta}\right\|_\infty} \cdot [V^*(\rho) - V^{\pi_\theta}(\rho)]$$

$$V^{\pi_\theta}(s) = \pi_\theta(a_1) \cdot R(a_1) + \pi_\theta(a_2) \cdot R(a_2)$$

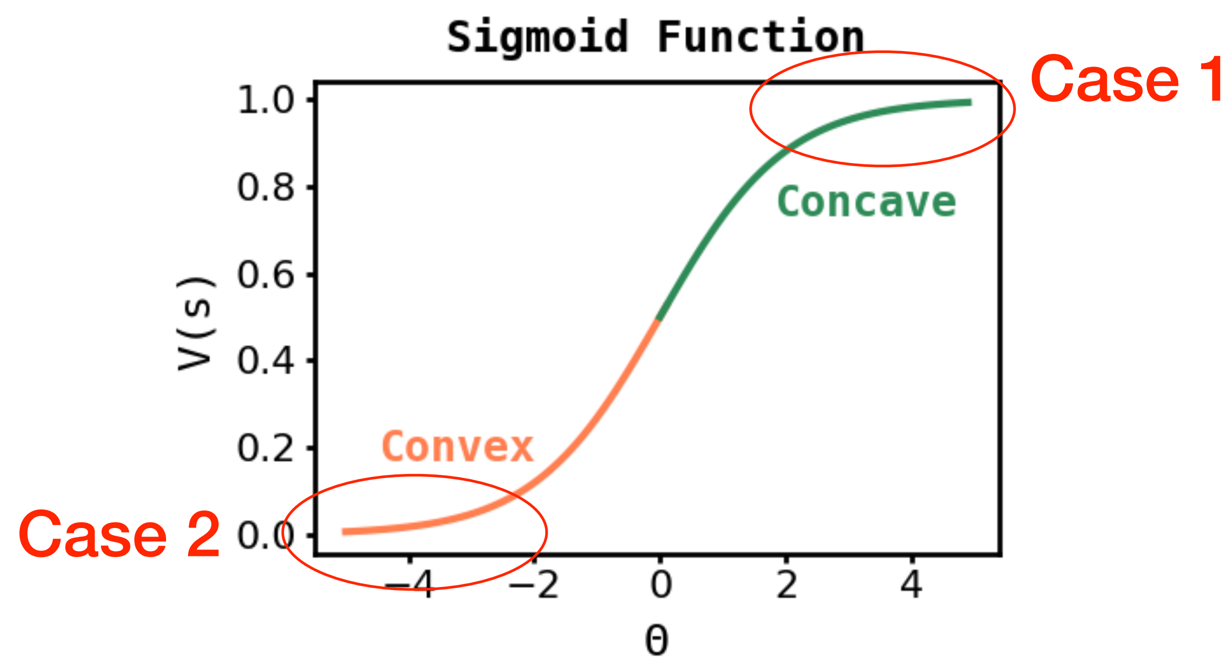


$$= \frac{\exp(\theta_{a_1})}{\exp(\theta_{a_1}) + \exp(\theta_{a_2})}$$

$$= \frac{1}{1 + \exp(\theta_{a_2} - \theta_{a_1})}$$

$$= \frac{1}{1 + \exp(-\Theta)}$$

(where $\Theta := \theta_{a_1} - \theta_{a_2}$)



Gradient approaches 0 if

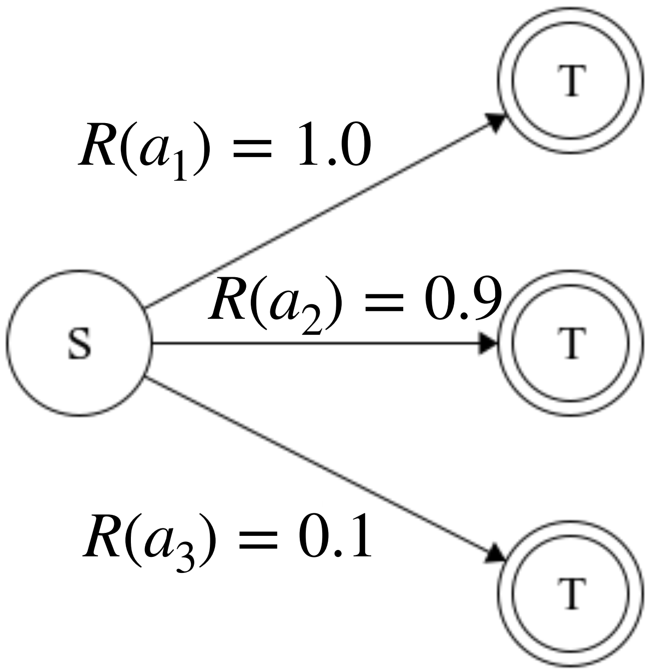
(Case 1) $\Theta \rightarrow +\infty$

(Case 2) $\Theta \rightarrow -\infty$

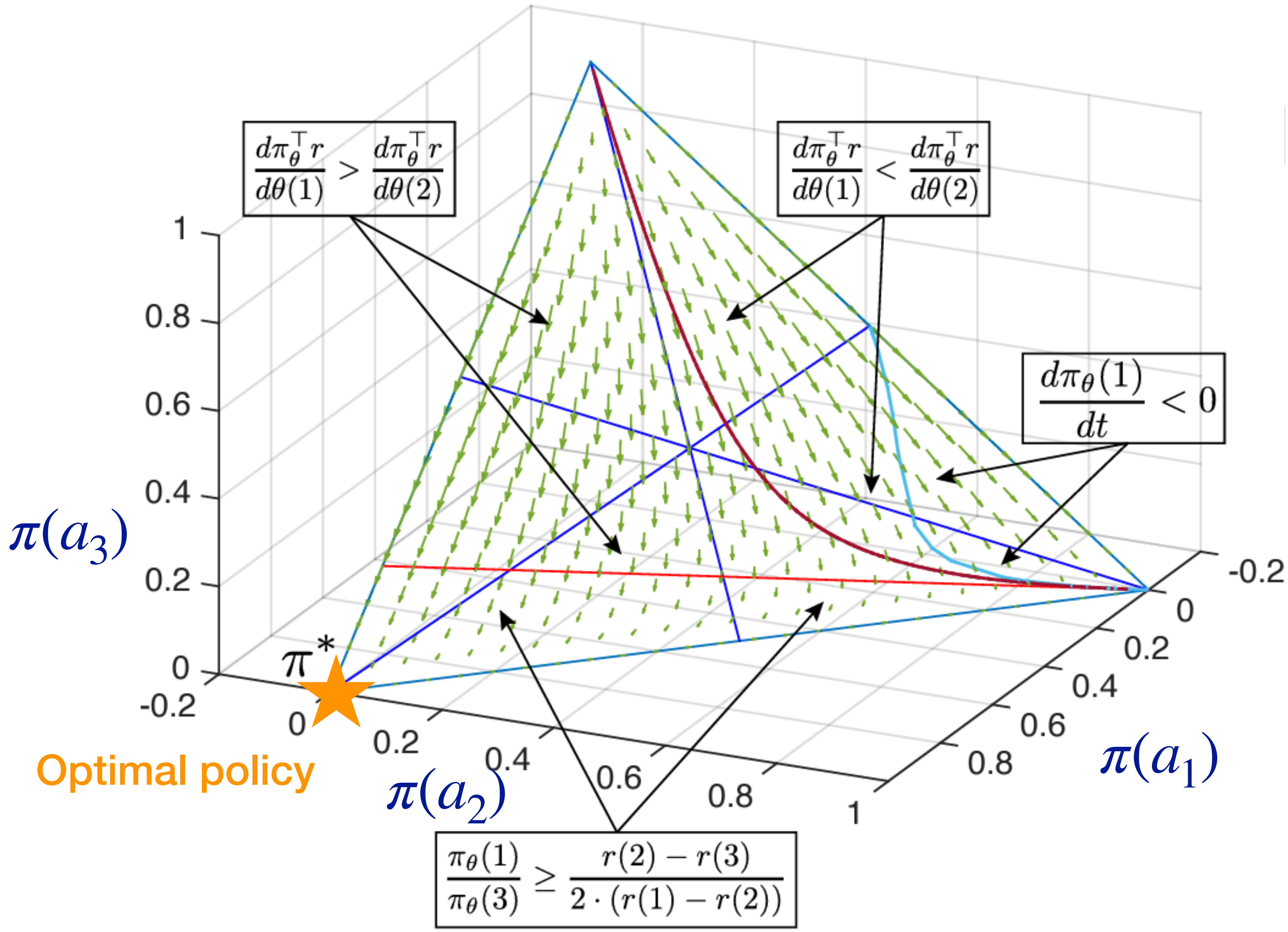
Yen-Ju Chen, Nai-Chieh Huang, and Ping-Chun Hsieh, "Accelerated Policy Gradient: On the Nesterov Momentum for Reinforcement Learning "(available at https://arxiv.org/abs/2310.11897)

Example: 1-State MDP and Gradient Flow

- 1-state, 3-action MDPs



Gradient Flow

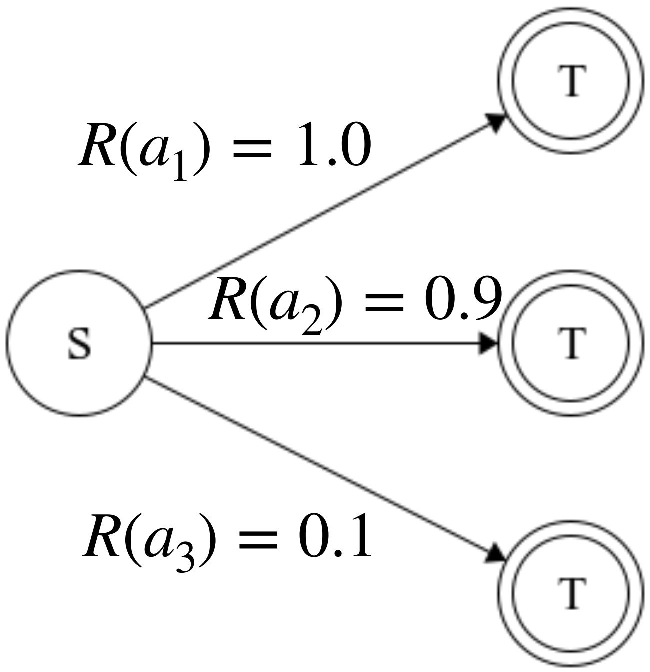


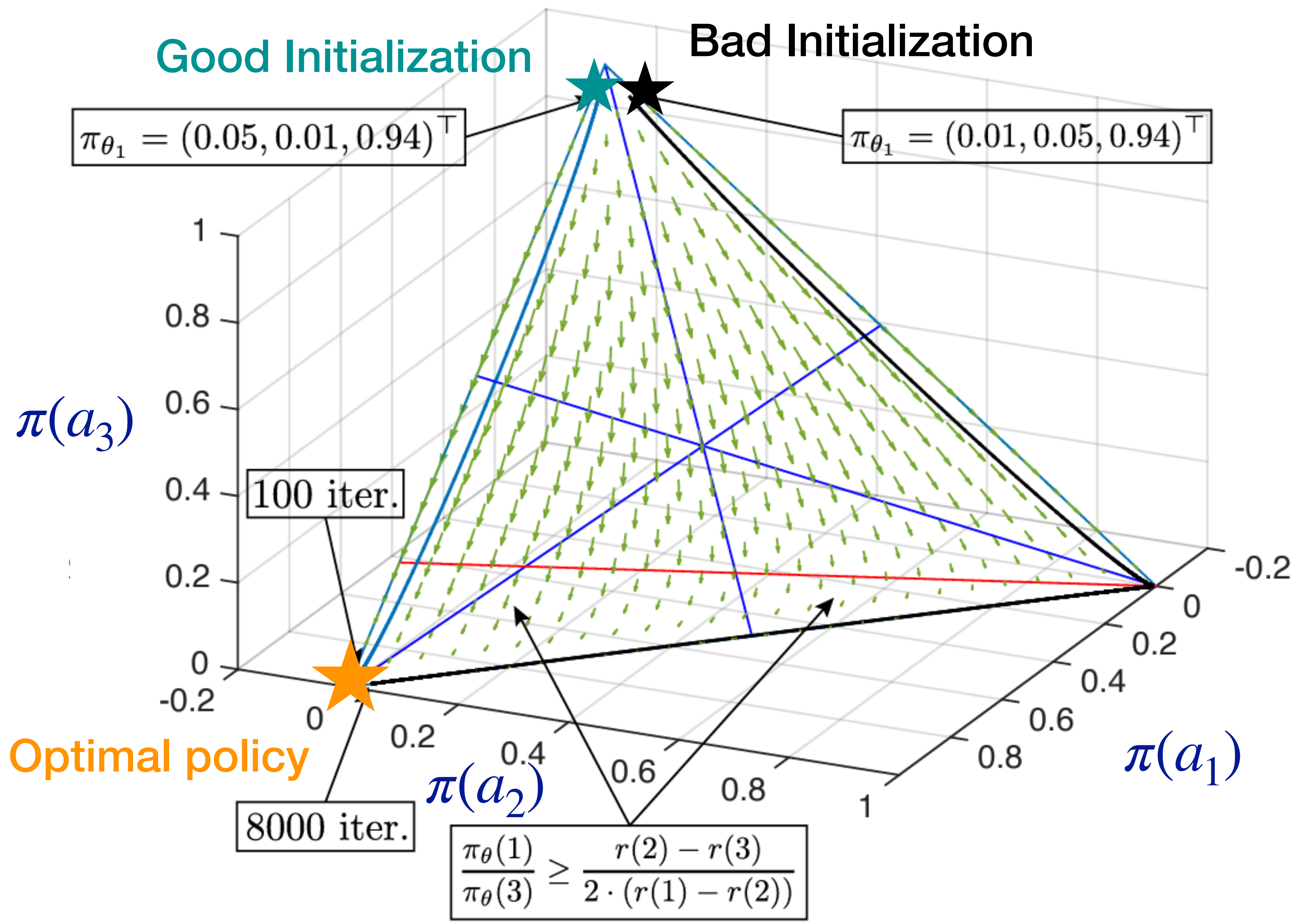
Mei et al., On the Global Convergence Rates of Softmax Policy Gradient Methods, ICML 2020

Example: 1-State MDP and Gradient Flow

- 1-state, 3-action MDPs

Good vs Bad Initializations

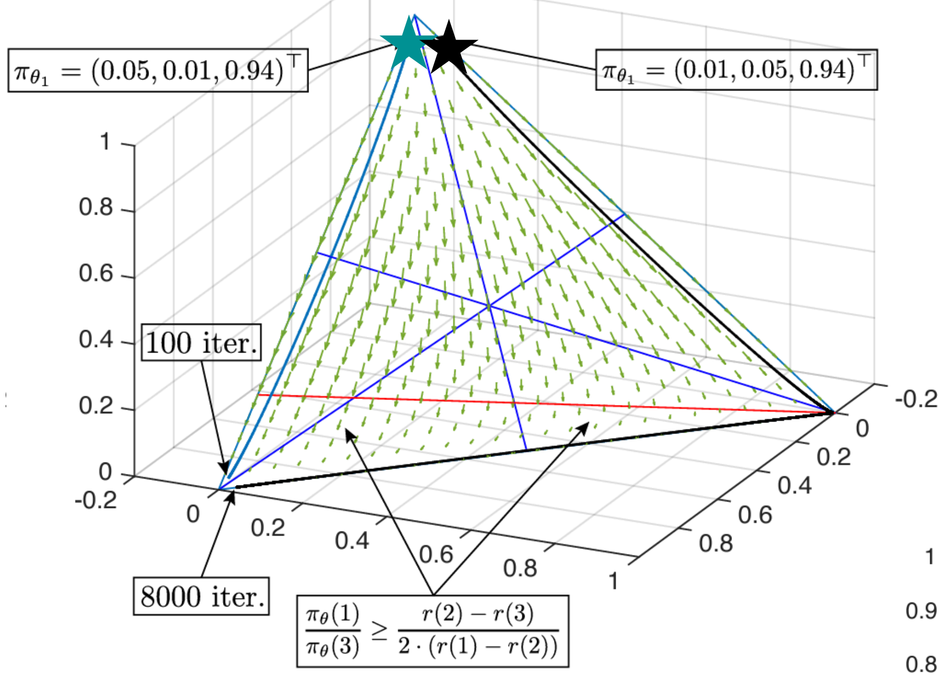




Mei et al., On the Global Convergence Rates of Softmax Policy Gradient Methods, ICML 2020

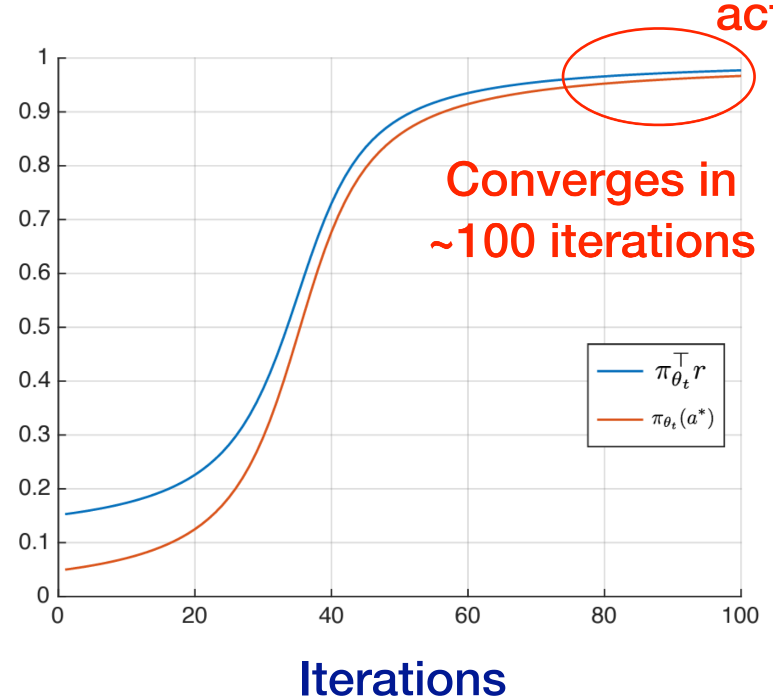
Example: Convergence vs Initializations

Good Initialization Bad Initialization

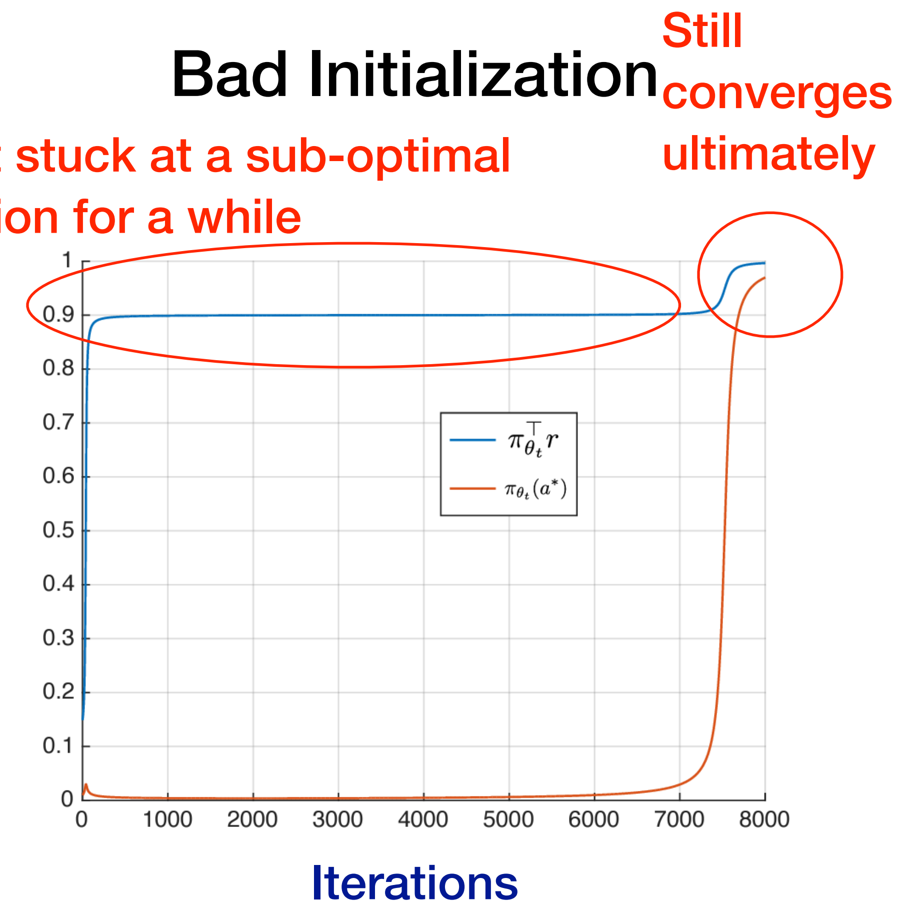


$$\left\| \frac{\partial V^{\pi_\theta}(\mu)}{\partial \theta} \right\|_2 \geq \frac{\min_s \pi_\theta(a^*(s)|s)}{\sqrt{S} \cdot \left\| d_\rho^{\pi^*} / d_\mu^{\pi_\theta} \right\|_\infty} \cdot [V^*(\rho) - V^{\pi_\theta}(\rho)] \,.$$

Good Initialization



Bad Initialization



Interpretations: PG convergence is sensitive to initialization as PL condition is "non-uniform" and depends on $\min_{s \in S} \pi_\theta(a^*(s)|s)$

Let's Show the Convergence Rate of PG!

A Taste of Theoretical Analysis

- To characterize the convergence, we have to know *how much improvement does the algorithm obtain after T steps*?

- How do we determine how far we walk in one day?

- Estimate the distance of every step!

 (Slide Credit: Yen-Ju Chen and Nai-Chieh Huang)

Let's Quantify One-Step Difference!

$$\boxed{\left\|\frac{\partial V^{\pi_\theta}(\mu)}{\partial \theta}\right\|_2 \geq \frac{\min_s \pi_\theta(a^*(s)|s)}{\sqrt{S}\cdot\left\|d_\rho^{\pi^*}/d_\mu^{\pi_\theta}\right\|_\infty}\cdot\left[V^*(\rho)-V^{\pi_\theta}(\rho)\right]}$$

One-Step Difference

$$\delta_t - \delta_{t+1} = V^{\pi_{\theta_{t+1}}}(\mu) - V^{\pi_{\theta_t}}(\mu) \geq \frac{(1-\gamma)^3}{16}\|\nabla_\theta V^{\pi_{\theta_t}}(\mu)\|_2^2$$ **(Improvement due to PG)**

(PL condition) $$\geq \frac{(1-\gamma)^3}{16S}\left\|\frac{d_\mu^{\pi^*}}{d_\mu^{\pi_{\theta_t}}}\right\|_2^{-2}\cdot\min_{s\in S}\pi_{\theta_t}(a^*(s)|s)^2\cdot\underbrace{[V^*(\mu)-V^{\pi_{\theta_t}}(\mu)]^2}_{=:\delta_t}$$

$$\geq \frac{(1-\gamma)^3}{16S}\left\|\frac{d_\mu^{\pi^*}}{d_\mu^{\pi_{\theta_t}}}\right\|_2^{-2}\cdot\inf_{s\in S, t>1}\pi_{\theta_t}(a^*(s)|s)^2\cdot\underbrace{[V^*(\mu)-V^{\pi_{\theta_t}}(\mu)]^2}_{=:\delta_t}$$

(This term is now independent of time)

Final Step: What kind of δ_t can satisfy that $\delta_t - \delta_{t+1} \geq c\cdot\delta_t$?

A Quick Summary of What We Discuss So Far

Next Question: Could we do better?

Let's Take a Second Look

- **Theorem (Convergence Rate of PG)**: [Mei et al., 2020]

  Assume $\mu(s) > 0$ for all s. Under $\eta = \frac{(1-\gamma)^3}{8}$, Exact PG achieves the following

$$V^*(\rho) - V^{\pi_{\theta_t}}(\rho) \leq \left(\frac{16 \cdot S}{\inf_{t\geq 1} \pi_t(a^* \mid s)^2 \cdot (1-\gamma)^6} \cdot \left\| \frac{d_\mu^{\pi^*}}{\mu} \right\|_\infty^2 \cdot \left\| \frac{1}{\mu} \right\|_\infty \right) \cdot \frac{1}{t}$$

Imagine that you are a scientist in DeepMind doing RL theory.

What questions would you ask?

(Q1) Is $O(\frac{1}{t})$ the best that PG can do?

(Q2) Can we shave the dependency on S (which could be large in practice)?

(Q3) Can we find another algorithm to achieve rate faster than $O(\frac{1}{t})$?