Part I: PG methods as black box optimization
Part II: Variance reduction and baselines

Part III: Theoretical Foundations

Sham Kakade

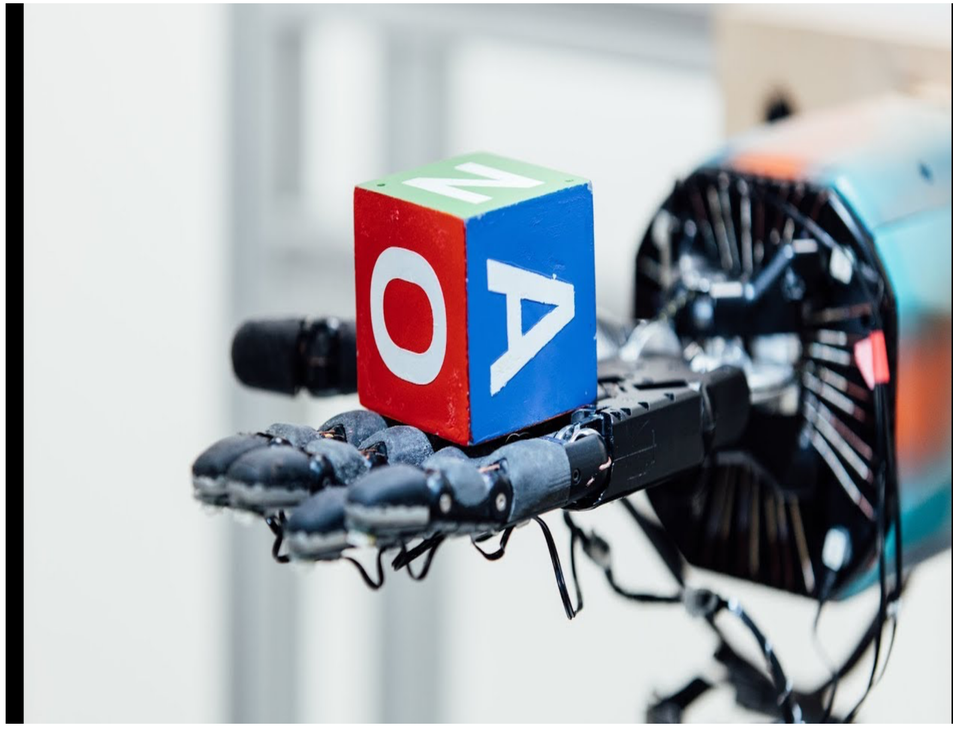
University of Washington & Microsoft Research

(with Nicolas Le Roux and Martha White)

Policy Optimization in RL

[AlphaZero, Silver et.al, 17]

[OpenAI Five, 18]



[OpenAI,19]

What should we be asking?

The “Tabular” Dynamic Programming approach

- With known model, policy iteration

The “Tabular” Dynamic Programming approach

- With known model, policy iteration
  1. Compute the state-action value every s, a pair:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \,|\, s_0 = s, a_0 = a\right]$$

The “Tabular” Dynamic Programming approach

- With known model, policy iteration
  1. Compute the state-action value every s, a pair:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \,|\, s_0 = s, a_0 = a\right]$$

  2. Update the policy π to be greedy:

$$\pi(s) \leftarrow \text{argmax}_a \; Q^{\pi}(s,a)$$

     GOTO 1

The “Tabular” Dynamic Programming approach

- With known model, policy iteration
  1. Compute the state-action value every s, a pair:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \,|\, s_0 = s, a_0 = a\right]$$

  2. Update the policy π to be greedy:

$$\pi(s) \leftarrow \text{argmax}_a \; Q^{\pi}(s,a)$$

     GOTO 1

- Convergence:
  - Policy (& value) iteration geometrically converge to the global optima.

The “Tabular” Dynamic Programming approach

- With known model, policy iteration
  1. Compute the state-action value every s, a pair:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \,|\, s_0 = s, a_0 = a\right]$$

  2. Update the policy π to be greedy:

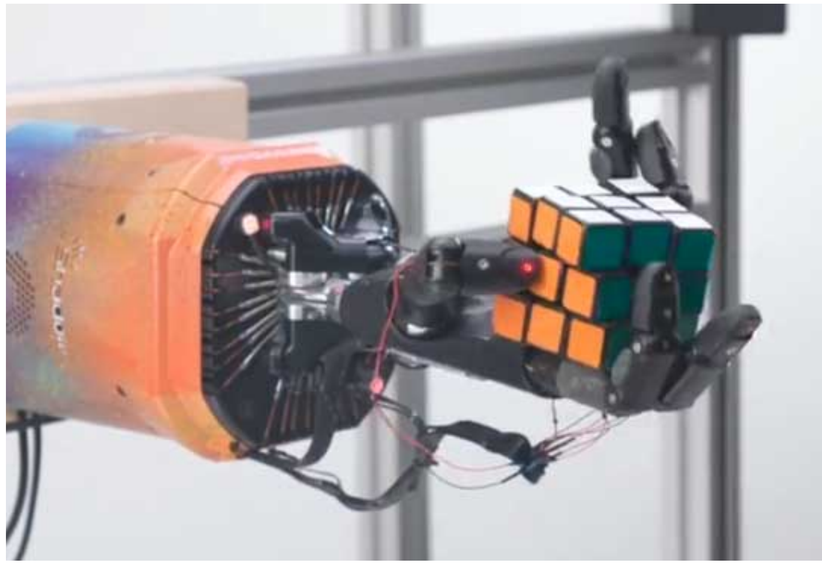
$$\pi(s) \leftarrow \text{argmax}_a \; Q^{\pi}(s,a)$$

     GOTO 1

- Convergence:
  - Policy (& value) iteration geometrically converge to the global optima.
- Generalization:
  - There are worst case approximation guarantees.

In practice, policy gradient methods rule…

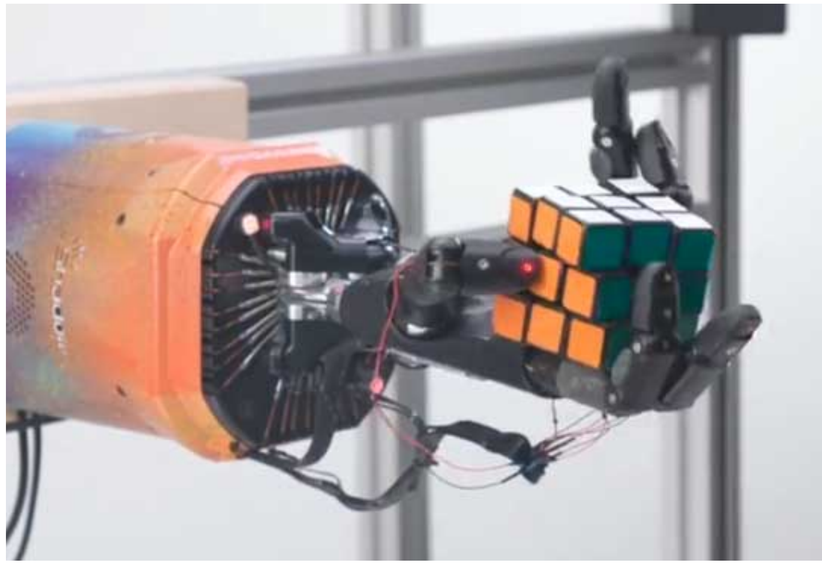
Why do we like them?



In practice, policy gradient methods rule…

Why do we like them?

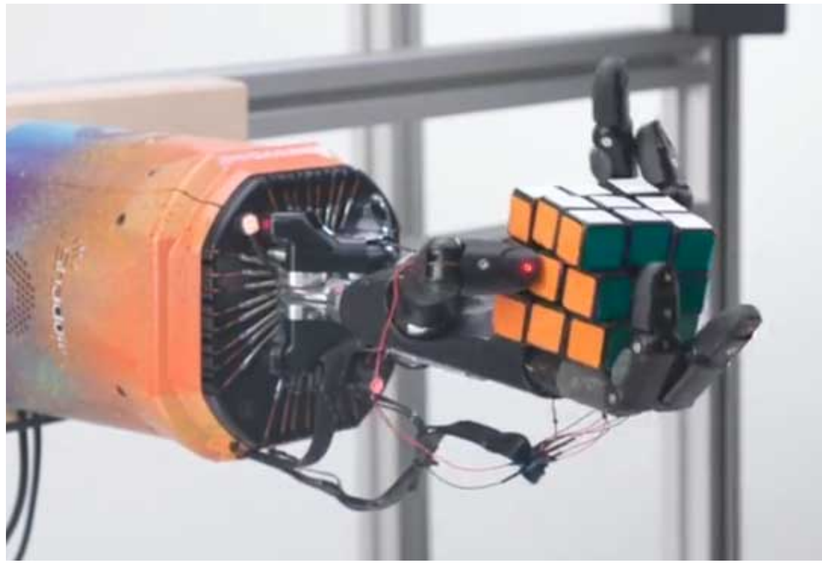
- They easily deal with large state/action spaces (through the neural net parameterization)



In practice, policy gradient methods rule…

Why do we like them?

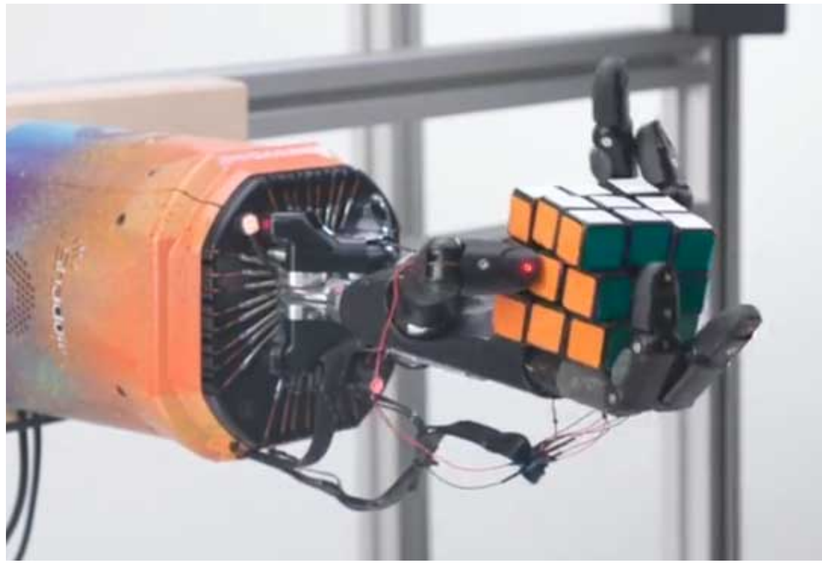
- They easily deal with large state/action spaces (through the neural net parameterization)
- We can estimate the gradient using only simulation



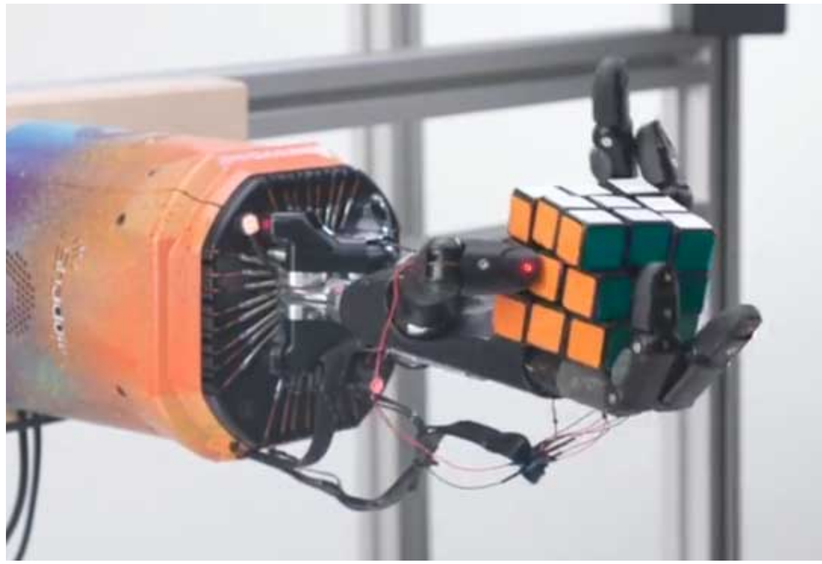
In practice, policy gradient methods rule…

Why do we like them?

- They easily deal with large state/action spaces (through the neural net parameterization)
- We can estimate the gradient using only simulation
- They directly optimize the cost function of interest!



In practice, policy gradient methods rule…



Why do we like them?

- They easily deal with large state/action spaces (through the neural net parameterization)
- We can estimate the gradient using only simulation
- They directly optimize the cost function of interest!

Questions:

- Convergence: Do policy gradient methods globally converge?
- Generalization: How do they deal with infinite states/actions?

Part III: Theoretical Foundations

Global convergence and generalization guarantees of (nonconvex) policy gradient methods.

Part III: Theoretical Foundations

Global convergence and generalization guarantees of (nonconvex) policy gradient methods.

- Part – A: small state spaces + exact gradients
  - curvature + non-convexity

Part III: Theoretical Foundations

Global convergence and generalization guarantees of (nonconvex) policy gradient methods.

- Part – A: small state spaces + exact gradients
  - curvature + non-convexity

- Part – B: large state spaces
  - generalization and distribution shift

Part III: Theoretical Foundations

Global convergence and generalization guarantees of (nonconvex) policy gradient methods.

- Part – A: small state spaces + exact gradients
  - curvature + non-convexity

- Part – B: large state spaces
  - generalization and distribution shift

Supplementary material:

- AJKS Book: Reinforcement Learning: Theory and Algorithms, Section 3
- Following along with the colab notebooks!

Part A: Small State Spaces

(and the softmax policy class)

Policy Optimization over the ”softmax” policy class
(let’s start simple!)

- Simplest way to parameterize the simplex, without constraints.

Policy Optimization over the "softmax" policy class
(let's start simple!)

- Simplest way to parameterize the simplex, without constraints.

- $\pi_\theta(a \mid s)$ is the probability of action a given state s

$$\pi_\theta(a \mid s) = \frac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})}$$

Policy Optimization over the "softmax" policy class
(let's start simple!)

- Simplest way to parameterize the simplex, without constraints.

- $\pi_\theta(a \mid s)$ is the probability of action a given state s

$$\pi_\theta(a \mid s) = \frac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})}$$

- Complete class: contains every stationary policy

Policy Optimization over the "softmax" policy class
(let's start simple!)

- Simplest way to parameterize the simplex, without constraints.

- $\pi_\theta(a \mid s)$ is the probability of action a given state s

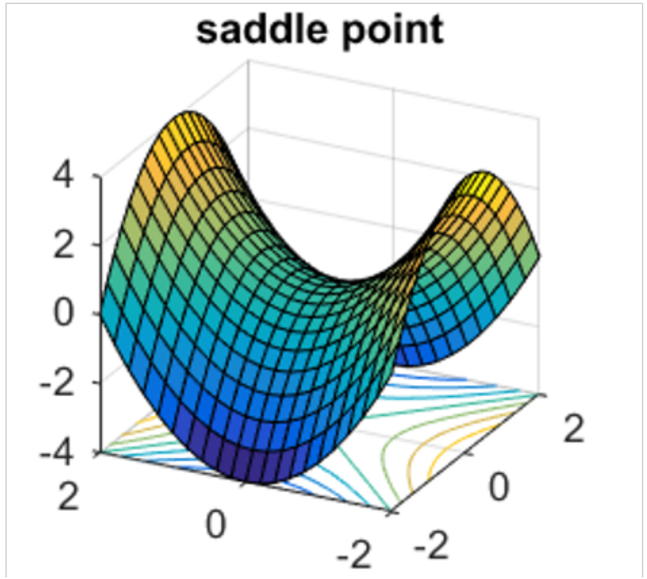
$$\pi_\theta(a \mid s) = \frac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})}$$

- Complete class: contains every stationary policy

The policy optimization problem $\max_\theta V^{\pi_\theta}(s_0)$ is non-convex.

Do we have global convergence?

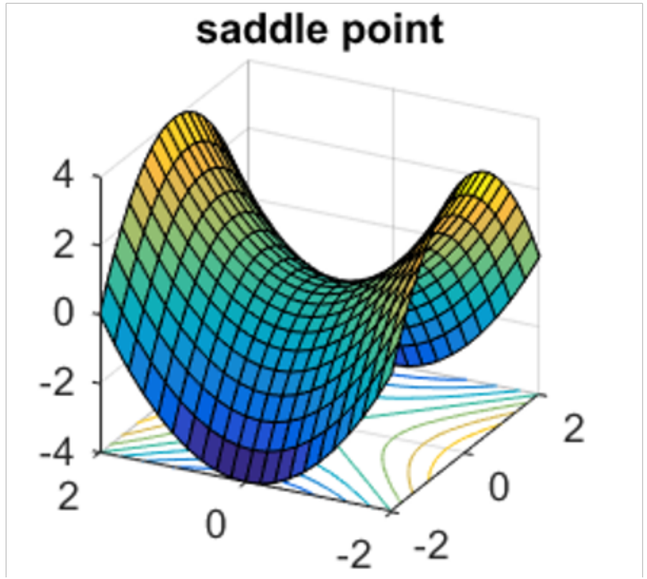
The Optimization Landscape



Supervised Learning:

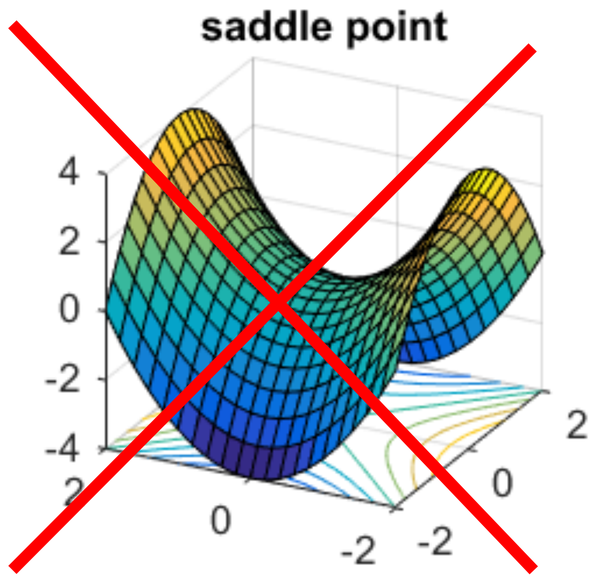
- Gradient descent tends to ‘just work’ in practice (not sensitive to initialization)
- Saddle points not a problem…

The Optimization Landscape



Supervised Learning:

- Gradient descent tends to ‘just work’ in practice (not sensitive to initialization)
- Saddle points not a problem…



Reinforcement Learning:

- In many real RL problems, we have “very” flat regions.
- Gradients can be exponentially small in the “horizon” due to lack of exploration.

The Optimization Landscape

Lemma: [Higher order vanishing gradients]

Suppose there are $S \leq 1/(1-\gamma)$ states in the MDP. With random initialization, all k-th higher-order gradients, for $k < S/\log(S)$, the spectral norm of the gradients are bounded by $2^{-S/2}$.

The Optimization Landscape

Thrun ‘92

Lemma: [Higher order vanishing gradients]

Suppose there are $S \leq 1/(1-\gamma)$ states in the MDP. With random initialization, all k-th higher-order gradients, for $k < S/\log(S)$, the spectral norm of the gradients are bounded by $2^{-S/2}$.

How might we address this?

Colab Notebook III-a: PG on $V^{\theta}(\mu)$

MDP:

- $S = 20, \;\; 1/(1-\gamma) = 20$
- $V^{\star}(s_0) \approx 7$

Colab Notebook III-a: PG on $V^\theta(\mu)$

MDP:

- $S = 20, \quad 1/(1-\gamma) = 20$
- $V^\star(s_0) \approx 7$

One idea for a fix:
use a surrogate objective:

$V^\theta(\mu) = E_{s\sim\mu}[V^\theta(s)]$

$\theta \leftarrow \theta + \eta \nabla_\theta V^\theta(\mu)$

Colab Notebook III-a: PG on $V^\theta(\mu)$

MDP:

- $S = 20, \quad 1/(1-\gamma) = 20$
- $V^\star(s_0) \approx 7$

One idea for a fix:
use a surrogate objective:

$$V^\theta(\mu) = E_{s\sim\mu}[V^\theta(s)]$$

$$\theta \leftarrow \theta + \eta \nabla_\theta V^\theta(\mu)$$

Let's try it out!

Colab Notebook III-a: PG on $V^\theta(\mu)$

MDP:

- $S = 20, \quad 1/(1-\gamma) = 20$
- $V^\star(s_0) \approx 7$

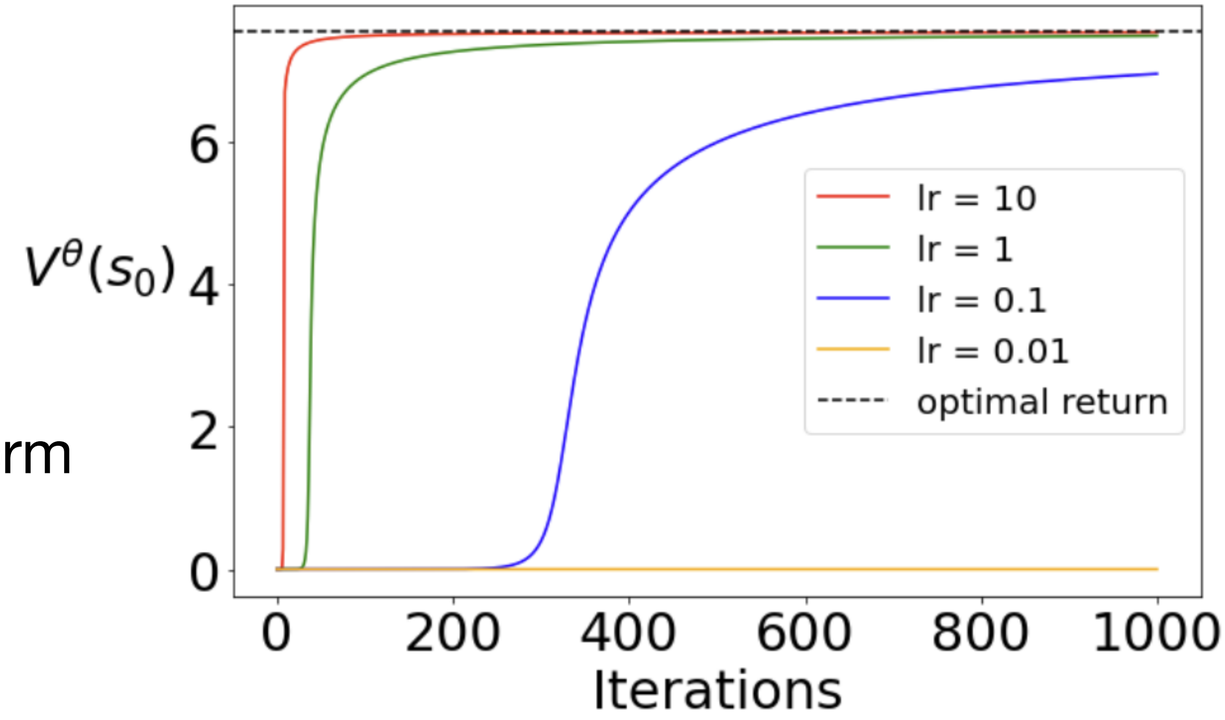
One idea for a fix:
use a surrogate objective:

$$V^\theta(\mu) = E_{s\sim\mu}[V^\theta(s)]$$

$$\theta \leftarrow \theta + \eta \nabla_\theta V^\theta(\mu)$$

Let's try it out!

- with exact gradients & μ as the uniform measure, we globally converge
  (curves show different learning rates)



Colab Notebook III-a: PG on $V^\theta(\mu)$

MDP:

- $S = 20, \quad 1/(1-\gamma) = 20$
- $V^\star(s_0) \approx 7$

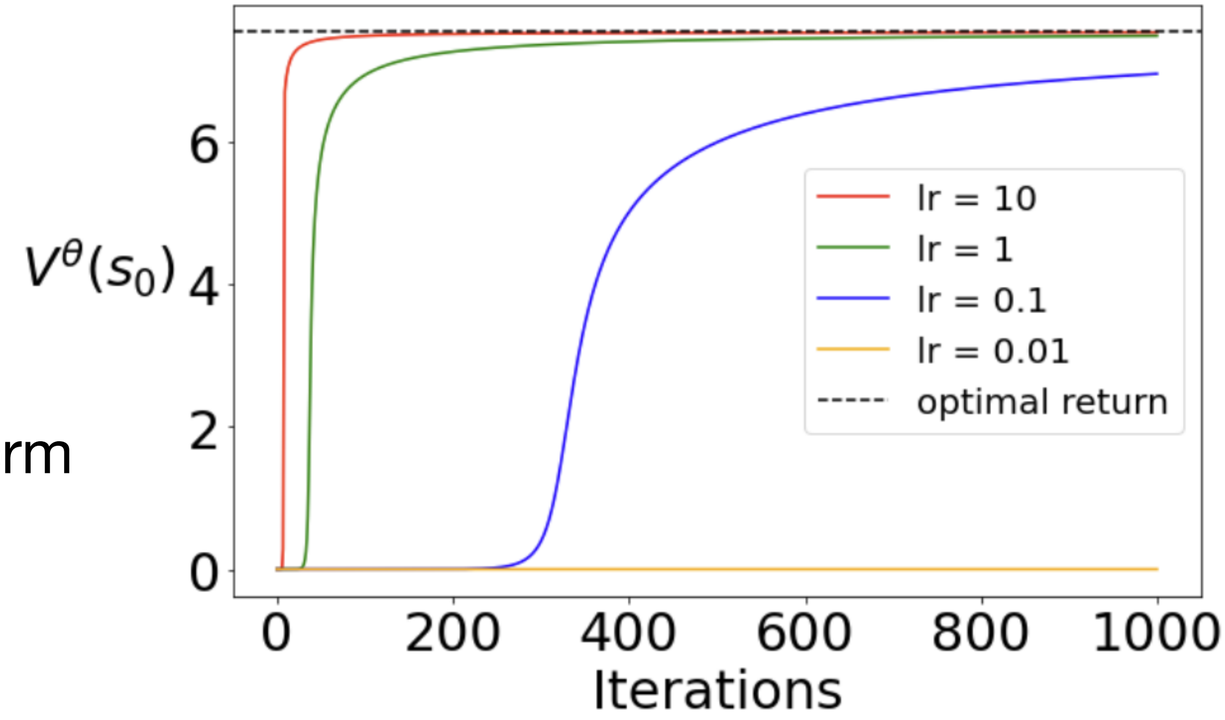
One idea for a fix:
use a surrogate objective:

$$V^\theta(\mu) = E_{s\sim\mu}[V^\theta(s)]$$

$$\theta \leftarrow \theta + \eta \nabla_\theta V^\theta(\mu)$$

Let's try it out!

- with exact gradients & μ as the uniform measure, we globally converge
  (curves show different learning rates)
- Does this idea always work?



Global Convergence of PG for Softmax

$$V^{\theta}(\mu) = E_{s\sim\mu}[V^{\theta}(s)]$$

$$\theta \leftarrow \theta + \eta \nabla_{\theta} V^{\theta}(\mu)$$

Theorem [Vanilla PG for Softmax Policy class]

Suppose μ has full support over the state space. Then, for all states s,

$$V^{\theta}(s) \rightarrow V^{\star}(s)$$

Global Convergence of PG for Softmax

$$V^{\theta}(\mu) = E_{s\sim\mu}[V^{\theta}(s)]$$

$$\theta \leftarrow \theta + \eta \nabla_{\theta} V^{\theta}(\mu)$$

Theorem [Vanilla PG for Softmax Policy class]

Suppose μ has full support over the state space. Then, for all states s,

$$V^{\theta}(s) \rightarrow V^{\star}(s)$$

- Even though problem is non-convex, we have global convergence.
  - proof is detailed/asymptotic (see AJKS)

Global Convergence of PG for Softmax

$$V^{\theta}(\mu) = E_{s\sim\mu}[V^{\theta}(s)]$$

$$\theta \leftarrow \theta + \eta \nabla_{\theta} V^{\theta}(\mu)$$

Theorem [Vanilla PG for Softmax Policy class]

Suppose μ has full support over the state space. Then, for all states s,

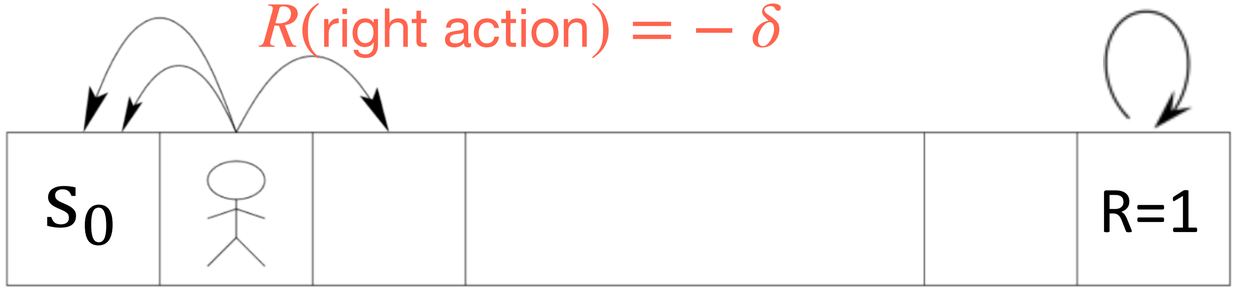
$$V^{\theta}(s) \rightarrow V^{\star}(s)$$

- Even though problem is non-convex, we have global convergence.
  - proof is detailed/asymptotic (see AJKS)
- Rate could be exponentially slow in terms of $\#\text{states}$
  - Let's take a closer look!

Colab Notebook III-b: the “anti-shaped” MDP

“Anti-shaped” modificiation:

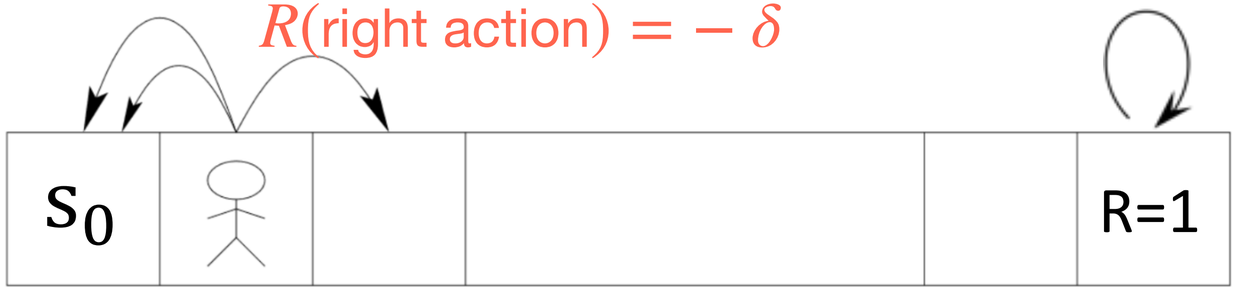
- $-\delta = 0.5$ reward for moving right
  (otherwise the same MDP)
- $V^\star(s_0) \approx 0.05$



Colab Notebook III-b: the “anti-shaped” MDP

“Anti-shaped” modificiation:

- $-\delta = 0.5$ reward for moving right
  (otherwise the same MDP)
- $V^\star(s_0) \approx 0.05$



Theory:

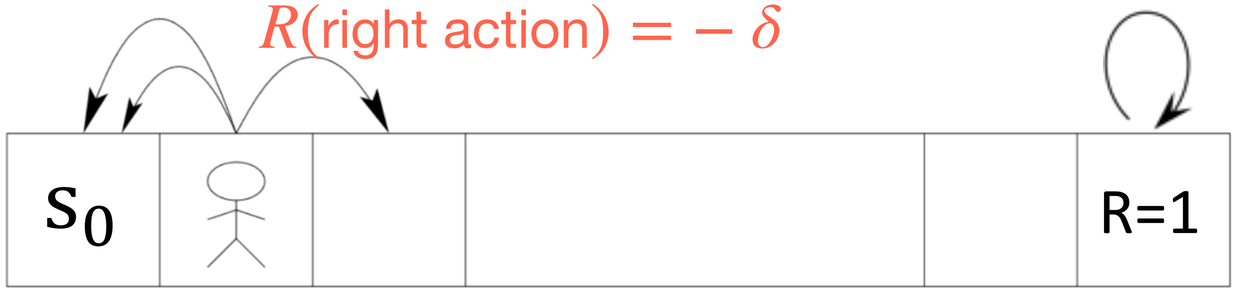
- exact gradients on $V^\theta(\mu) = E_{s\sim\mu}[V^\theta(s)]$

  & μ being uniform ⇒
  asymptotic global converge

Colab Notebook III-b: the “anti-shaped” MDP

“Anti-shaped” modificiation:

- $-\delta = 0.5$ reward for moving right
  (otherwise the same MDP)
- $V^\star(s_0) \approx 0.05$



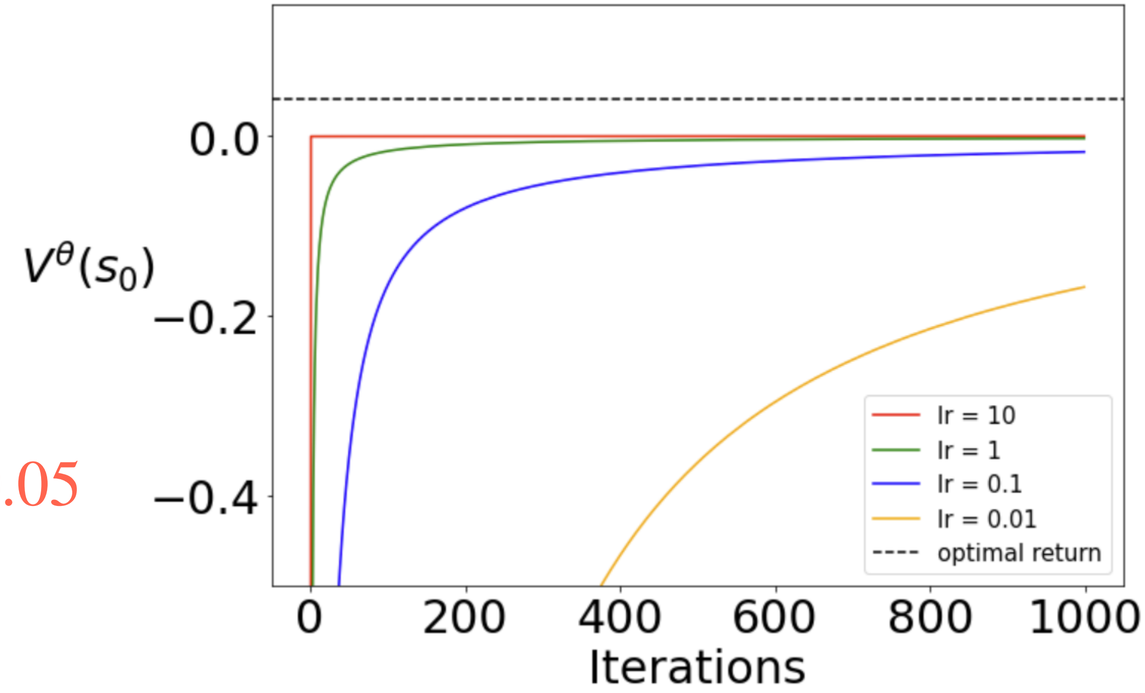
Theory:

- exact gradients on $V^\theta(\mu) = E_{s\sim\mu}[V^\theta(s)]$

  & μ being uniform ⇒
  asymptotic global converge

Practice:

- we never see convergence to opt, $V^\star(s_0) \approx 0.05$
  (stalls at $V(s_0) = 0$)
- rate could be exponentially slow



Colab Notebook III-c: Entropy Regularization

- (for softmax) we get flat gradients when π becomes deterministic
- heuristic: use entropy regularization to penalize against deterministic policies

$$L_{ent}(\theta) := V^{\theta}(\mu) + \lambda \sum_{s,a} \mu(s)\pi_{\theta}(a \mid s) \log \pi_{\theta}(a \mid s)$$

$$\theta \leftarrow \theta + \eta \nabla L_{ent}(\theta)$$

Colab Notebook III-c: Entropy Regularization

- (for softmax) we get flat gradients when π becomes deterministic
- heuristic: use entropy regularization to penalize against deterministic policies

$$L_{ent}(\theta) := V^{\theta}(\mu) + \lambda \sum_{s,a} \mu(s)\pi_{\theta}(a \mid s)\log \pi_{\theta}(a \mid s)$$

$$\theta \leftarrow \theta + \eta \nabla L_{ent}(\theta)$$

Theory:

- again, with exact gradients on $V^{\theta}(\mu) = E_{s\sim\mu}[V^{\theta}(s)]$ & μ being uniform

  $\Rightarrow$

  asymptotic global converge

  (see AJKS, Chapter 10: Further Reading)

Colab Notebook III-c: Entropy Regularization

- (for softmax) we get flat gradients when π becomes deterministic
- heuristic: use entropy regularization to penalize against deterministic policies

$$L_{ent}(\theta) := V^{\theta}(\mu) + \lambda \sum_{s,a} \mu(s)\pi_{\theta}(a \mid s)\log \pi_{\theta}(a \mid s)$$

$$\theta \leftarrow \theta + \eta \nabla L_{ent}(\theta)$$

Theory:

- again, with exact gradients on $V^{\theta}(\mu) = E_{s\sim\mu}[V^{\theta}(s)]$ & μ being uniform

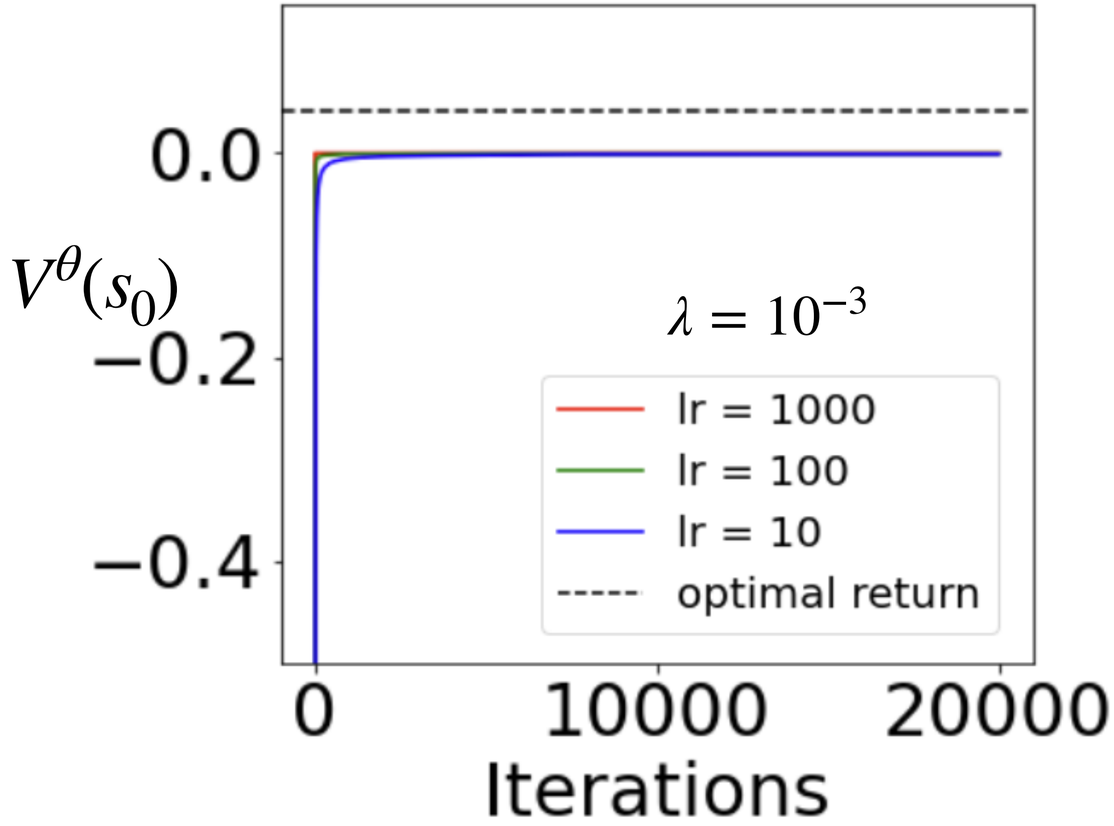
  $\Rightarrow$

  asymptotic global converge

  (see AJKS, Chapter 10: Further Reading)

Practice: (on the anti-shaped example)

- we never see convergence to opt, $V^{\star}(s_0) \approx 0.05$ (regardless of how we set λ)
- rate could be exponentially slow



Global Convergence: Softmax + Log Barrier regularization

$$L_\lambda(\theta) := V^\theta(\mu) + \frac{\lambda}{A}\sum_{s,a}\mu(s)\log\pi_\theta(a\,|\,s)$$

$$\theta \leftarrow \theta + \eta\,\nabla L_\lambda(\theta)$$

Theorem [PG: Softmax+Log Barrier]

S : #states, A : #actions, H : Horizon $= 1/(1-\gamma)$

Suppose $\mu = \text{uniform}_S$ and with appropriate settings of λ and η

After $\frac{S^4A^2H^6}{\epsilon^2}$ iterations, we have for all s,

$$V^\theta(s) \geq V^\star(s) - \epsilon$$

Global Convergence: Softmax + Log Barrier regularization

$$L_\lambda(\theta) := V^\theta(\mu) + \frac{\lambda}{A}\sum_{s,a}\mu(s)\log\pi_\theta(a\,|\,s)$$

$$\theta \leftarrow \theta + \eta\,\nabla L_\lambda(\theta)$$

Theorem [PG: Softmax+Log Barrier]

S : #states, A : #actions, H : Horizon $= 1/(1-\gamma)$

Suppose $\mu = \text{uniform}_S$ and with appropriate settings of λ and η

After $\frac{S^4A^2H^6}{\epsilon^2}$ iterations, we have for all s,

$$V^\theta(s) \geq V^\star(s) - \epsilon$$

- Even though problem is non-convex, we a have poly iteration complexity.

Global Convergence: Softmax + Log Barrier regularization

$$L_\lambda(\theta) := V^\theta(\mu) + \frac{\lambda}{A}\sum_{s,a}\mu(s)\log\pi_\theta(a\,|\,s)$$

$$\theta \leftarrow \theta + \eta\,\nabla L_\lambda(\theta)$$

Theorem [PG: Softmax+Log Barrier]

S : #states, A : #actions, H : Horizon $= 1/(1-\gamma)$

Suppose $\mu = \text{uniform}_S$ and with appropriate settings of λ and η

After $\frac{S^4A^2H^6}{\epsilon^2}$ iterations, we have for all s,

$$V^\theta(s) \geq V^\star(s) - \epsilon$$

- Even though problem is non-convex, we a have poly iteration complexity.
- Log barrier and uniform μ helps with conditioning problems.
  - succinct proof: shows that $\pi_\theta(a\,|\,s)$ doesn't become too small (see AJKS)
  - log barrier reg = KL-regularization ≠ entropy regularization

Colab Notebook III-d: Log Barrier Regularization

- Log barrier aggressively penalizes against deterministic policies
- polynomial upper bound is slow

$$L_\lambda(\theta) := V^\theta(\mu) + \frac{\lambda}{SA} \sum_{s,a} \log \pi_\theta(a \mid s)$$

$$\theta \leftarrow \theta + \eta \nabla L_\lambda(\theta)$$

Colab Notebook III-d: Log Barrier Regularization

- Log barrier aggressively penalizes against deterministic policies
- polynomial upper bound is slow

$$L_\lambda(\theta) := V^\theta(\mu) + \frac{\lambda}{SA} \sum_{s,a} \log \pi_\theta(a \mid s)$$

$$\theta \leftarrow \theta + \eta \nabla L_\lambda(\theta)$$

Theory:

- with exact gradients on $L_\lambda(\theta)$
  & μ being uniform
  $\Rightarrow$
  global converge at polynomial rate

Colab Notebook III-d: Log Barrier Regularization

- Log barrier aggressively penalizes against deterministic policies
- polynomial upper bound is slow

$$L_\lambda(\theta) := V^\theta(\mu) + \frac{\lambda}{SA} \sum_{s,a} \log \pi_\theta(a \mid s)$$

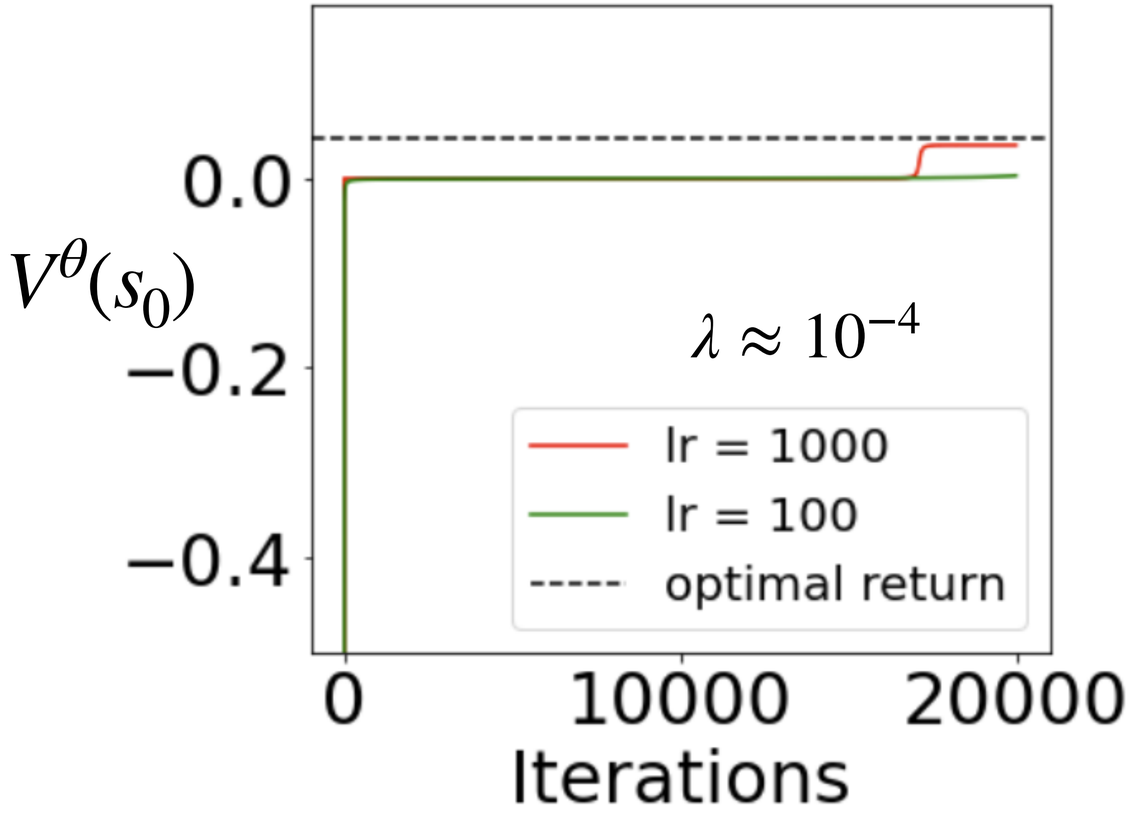
$$\theta \leftarrow \theta + \eta \nabla L_\lambda(\theta)$$

Theory:

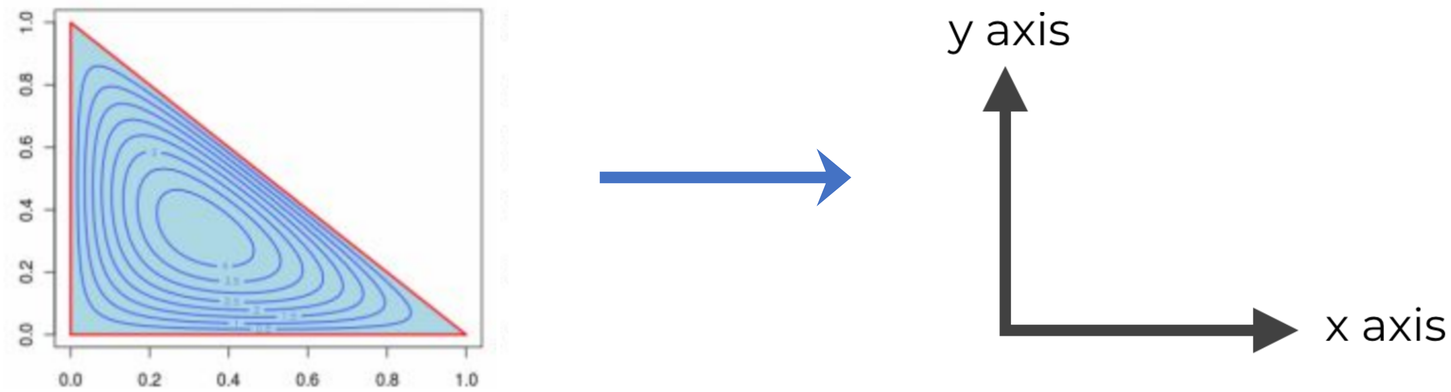
- with exact gradients on $L_\lambda(\theta)$
  & μ being uniform
  $\Rightarrow$
  global converge at polynomial rate

Practice: (on the anti-shaped example)

- for small λ & appropriate learn. rate, we convergence near to opt
- rate is polynomial, yet slow

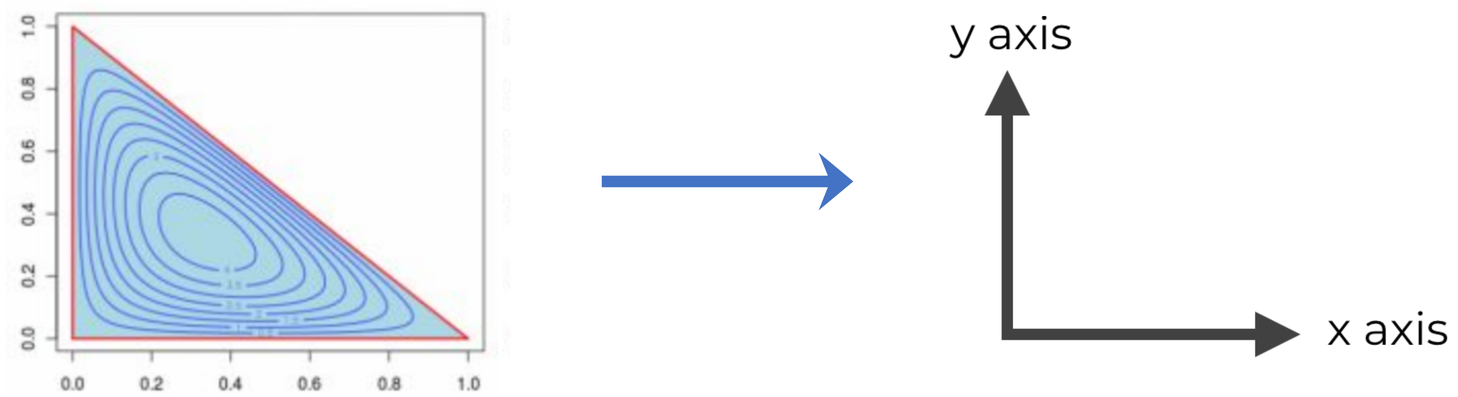


Preconditioning: The Natural Policy Gradient (NPG)



- Practice: most methods are gradient based, usually variants of:
  NPG [K. ‘01]; TRPO [Schulman ‘15]; PPO [Schulman ‘17]

Preconditioning: The Natural Policy Gradient (NPG)



- Practice: most methods are gradient based, usually variants of:
  NPG [K. ‘01]; TRPO [Schulman ‘15]; PPO [Schulman ‘17]
- NPG warps the distance metric to stretch the corners out (using the Fisher information metric) to move ‘more’ near the boundaries. The update is:

$$F(\theta) = E_{s,a\sim\pi_\theta}\left[\nabla \log \pi_\theta(a \mid s)\, \nabla \log \pi_\theta(a \mid s)^\top\right]$$

$$\theta \leftarrow \theta + \eta F(\theta)^{-1} \nabla V^\theta(s_0)$$

NPG and “soft” policy iteration

- The softmax policy class: $\pi_\theta(a \mid s) \propto \exp(\theta_{s,a})$

NPG and “soft” policy iteration

- The softmax policy class: $\pi_\theta(a \mid s) \propto \exp(\theta_{s,a})$
- At iteration t, the NPG update rule:

$$\theta \leftarrow \theta + \eta F(\theta)^{-1} \nabla V^\theta(s_0)$$

is equivalent to a “soft” policy iteration update rule:

$$\pi(a \mid s) \leftarrow \pi(a \mid s) \frac{\exp(\eta Q^\pi(s,a))}{Z}$$

NPG and “soft” policy iteration

- The softmax policy class: $\pi_\theta(a \mid s) \propto \exp(\theta_{s,a})$
- At iteration t, the NPG update rule:

$$\theta \leftarrow \theta + \eta F(\theta)^{-1} \nabla V^\theta(s_0)$$

is equivalent to a “soft” policy iteration update rule:

$$\pi(a \mid s) \leftarrow \pi(a \mid s) \frac{\exp(\eta Q^\pi(s,a))}{Z}$$

What happens for this non-convex update rule?

Global Convergence of NPG

Theorem [NPG]

Set $\eta = (1-\gamma)^2 \log A$.

For the softmax policy class, we have after T iterations and for all states s_0

$$V^{(T)}(s_0) \geq V^{\star}(s_0) - \frac{2}{(1-\gamma)^2 T}$$

Global Convergence of NPG

Theorem [NPG]

Set $\eta = (1-\gamma)^2 \log A$.

For the softmax policy class, we have after T iterations and for all states s_0

$$V^{(T)}(s_0) \geq V^{\star}(s_0) - \frac{2}{(1-\gamma)^2 T}$$

- Dimension free iteration complexity: (No dependence on S, A, μ)

Global Convergence of NPG

Theorem [NPG]

Set $\eta = (1-\gamma)^2 \log A$.

For the softmax policy class, we have after T iterations and for all states s_0

$$V^{(T)}(s_0) \geq V^{\star}(s_0) - \frac{2}{(1-\gamma)^2 T}$$

- Dimension free iteration complexity: (No dependence on S, A, μ)
- Also a “fast rate”.

Global Convergence of NPG

Theorem [NPG]

Set $\eta = (1-\gamma)^2 \log A$.

For the softmax policy class, we have after T iterations and for all states s_0

$$V^{(T)}(s_0) \geq V^{\star}(s_0) - \frac{2}{(1-\gamma)^2 T}$$

- Dimension free iteration complexity: (No dependence on S, A, μ)
- Also a “fast rate”.
- Even though problem is non-convex, a mirror descent analysis applies. Analysis idea from [Even-Dar, K., Mansour 2009]

Colab Notebook III-e: NPG

- The NPG update rule: $\pi(a \mid s) \leftarrow \pi(a \mid s) \frac{\exp(\eta Q^{\pi}(s, a))}{Z}$

Colab Notebook III-e: NPG

- The NPG update rule: $\pi(a \mid s) \leftarrow \pi(a \mid s) \frac{\exp(\eta Q^{\pi}(s,a))}{Z}$

Theory:

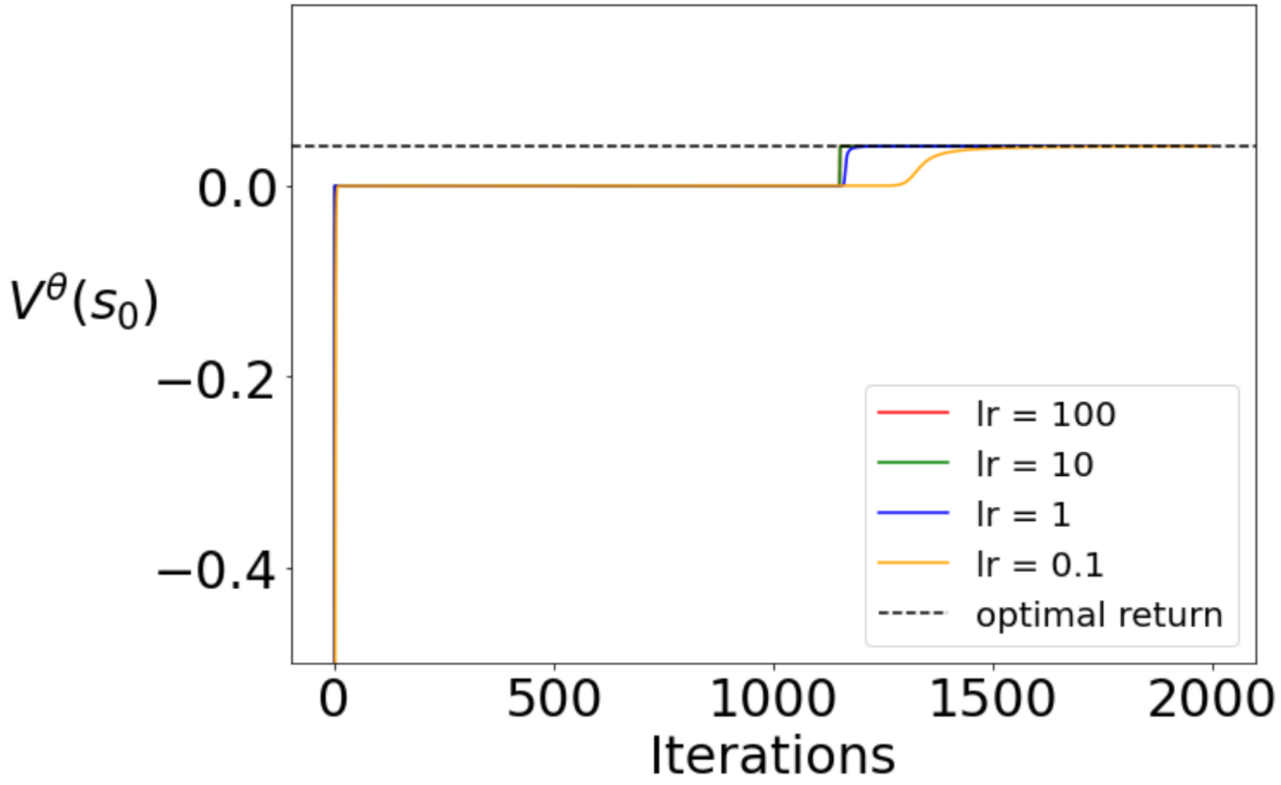
- global converge at a dimension free rate

Colab Notebook III-e: NPG

- The NPG update rule: $\pi(a \mid s) \leftarrow \pi(a \mid s) \frac{\exp(\eta Q^{\pi}(s,a))}{Z}$

Theory:
- global converge at a dimension free rate

Practice: (on the anti-shaped example)
- convergence to opt, $V^{\star}(s_0) \approx 0.05$ is relatively fast



Part-B: Large State Spaces

What about generalization and sampling?

Policy Classes

$\pi_\theta(a \mid s)$ is the probability of action a given s, parameterized by

$$\pi_\theta(a \mid s) \propto \exp(f_\theta(s, a))$$

Policy Classes

$\pi_\theta(a \mid s)$ is the probability of action a given s, parameterized by

$$\pi_\theta(a \mid s) \propto \exp(f_\theta(s, a))$$

- Softmax policy class: $f_\theta(s, a) = \theta_{s,a}$

Policy Classes

$\pi_\theta(a \mid s)$ is the probability of action a given s, parameterized by

$$\pi_\theta(a \mid s) \propto \exp(f_\theta(s, a))$$

- Softmax policy class: $f_\theta(s, a) = \theta_{s,a}$
- Log-linear policy class: $f_\theta(s, a) = \overrightarrow{\theta} \cdot \overrightarrow{\phi}(s, a)$ where $\overrightarrow{\phi}(s, a) \in R^d$

Policy Classes

$\pi_\theta(a \mid s)$ is the probability of action a given s, parameterized by

$$\pi_\theta(a \mid s) \propto \exp(f_\theta(s, a))$$

- Softmax policy class: $f_\theta(s, a) = \theta_{s,a}$
- Log-linear policy class: $f_\theta(s, a) = \vec{\theta} \cdot \vec{\phi}(s, a)$ where $\vec{\phi}(s, a) \in R^d$
- Neural policy class: $f_\theta(s, a)$ is a neural network

NPG & Log Linear Policy Classes

- Feature vector $\phi(s,a) \in \mathbb{R}^d$, $\pi_\theta(a \mid s) \propto \exp(\theta \cdot \phi_{s,a})$

NPG & Log Linear Policy Classes

- Feature vector $\phi(s,a) \in \mathbb{R}^d$, $\pi_\theta(a \mid s) \propto \exp(\theta \cdot \phi_{s,a})$
- At iteration t, the NPG update rule:

$$\theta \leftarrow \theta + \eta F(\theta)^{-1} \nabla V^\theta(\mu)$$

is equivalent to the "soft"+approximate policy iteration update:

NPG & Log Linear Policy Classes

- Feature vector $\phi(s,a) \in \mathbb{R}^d$, $\pi_\theta(a \mid s) \propto \exp(\theta \cdot \phi_{s,a})$
- At iteration t, the NPG update rule:

$$\theta \leftarrow \theta + \eta F(\theta)^{-1} \nabla V^\theta(\mu)$$

is equivalent to the "soft"+approximate policy iteration update:

1. approximate the Q^θ with the the features:

$$w_\star \in \text{argmin}_w E_{s,a \sim d(\cdot|\pi,\mu)} \left[\left(Q^\theta(s,a) - w \cdot \phi_{s,a} \right)^2 \right].$$

where $d(\cdot \mid \pi, \mu)$ is "on-policy" distribution starting from $s_0, a_0 \sim \mu$

NPG & Log Linear Policy Classes

- Feature vector $\phi(s,a) \in \mathbb{R}^d$, $\pi_\theta(a \mid s) \propto \exp(\theta \cdot \phi_{s,a})$
- At iteration t, the NPG update rule:

$$\theta \leftarrow \theta + \eta F(\theta)^{-1} \nabla V^\theta(\mu)$$

is equivalent to the "soft"+approximate policy iteration update:

1. approximate the Q^θ with the the features:

$$w_\star \in \text{argmin}_w E_{s,a \sim d(\cdot|\pi,\mu)} \left[\left(Q^\theta(s,a) - w \cdot \phi_{s,a} \right)^2 \right].$$

where $d(\cdot \mid \pi, \mu)$ is "on-policy" distribution starting from $s_0, a_0 \sim \mu$

2. policy update

$$\pi(a \mid s) \leftarrow \frac{\pi(a \mid s) \exp(\eta w_\star \cdot \phi_{s,a})}{Z_s}$$

(Z_s is the normalizing constant)

Realizable case: NPG + log linear policy classes

Realizable case: NPG + log linear policy classes

- Realizability: Suppose that $Q^{\theta}(s, a)$ is a linear function in $\phi(s, a)$

Realizable case: NPG + log linear policy classes

- Realizability: Suppose that $Q^{\theta}(s,a)$ is a linear function in $\phi(s,a)$
- Supervised learning error: our estimate $\widehat{w}^{t}$ has bounded regression error (say due to sampling)

$$E_{s,a\sim d(\cdot|\pi^t,\mu)}\left[\left(Q^{\theta}(s,a) - \widehat{w}^{t}\cdot\phi_{s,a}\right)^2\right] \leq \epsilon_{\text{stat}}$$

Realizable case: NPG + log linear policy classes

- Realizability: Suppose that $Q^\theta(s,a)$ is a linear function in $\phi(s,a)$
- Supervised learning error: our estimate $\widehat{w}^t$ has bounded regression error (say due to sampling)

$$E_{s,a\sim d(\cdot|\pi^t,\mu)}\left[\left(Q^\theta(s,a) - \widehat{w}^t \cdot \phi_{s,a}\right)^2\right] \leq \epsilon_{\text{stat}}$$

- Conditioning (i.e. "feature coverage"): $\|\phi_{s,a}\| \leq 1$ and define

$$\kappa = 1/\sigma_{\min}\left(E_{s,a\sim\mu}\left[\phi_{s,a}\phi_{s,a}^\top\right]\right)$$

Realizable case: NPG + log linear policy classes

- Realizability: Suppose that $Q^\theta(s,a)$ is a linear function in $\phi(s,a)$
- Supervised learning error: our estimate $\widehat{w}^t$ has bounded regression error (say due to sampling)

$$E_{s,a\sim d(\cdot|\pi^t,\mu)}\left[\left(Q^\theta(s,a) - \widehat{w}^t \cdot \phi_{s,a}\right)^2\right] \leq \epsilon_{\text{stat}}$$

- Conditioning (i.e. "feature coverage"): $\|\phi_{s,a}\| \leq 1$ and define

$$\kappa = 1/\sigma_{\min}\left(E_{s,a\sim\mu}\left[\phi_{s,a}\phi_{s,a}^\top\right]\right)$$

Theorem [NPG]

A : #actions, H : Horizon $= 1/(1-\gamma)$, Norm bound: $\|\widehat{w}^t\| \leq W$

After T iterations, the NPG algorithm returns a π s.t.

$$V^{(T)}(s_0) \geq V^\star(s_0) - HW\sqrt{\frac{2\log A}{T}} - \sqrt{4AH^3\kappa\ \epsilon_{\text{stat}}}$$

NPG+Log Linear Case

(just notation for sample based approach)

NPG+Log Linear Case
(just notation for sample based approach)

- For a state-action distribution υ, define:
  $L(w;\theta,\upsilon) := E_{s,a\sim\upsilon}\left[(Q^{\pi_\theta}(s,a) - w\cdot\phi_{s,a})^2\right]$.

NPG+Log Linear Case
(just notation for sample based approach)

- For a state-action distribution υ, define:
  $L(w;\theta,\upsilon) := E_{s,a\sim\upsilon}\left[(Q^{\pi_\theta}(s,a) - w\cdot\phi_{s,a})^2\right]$.
- The NPG update is equivalent to:

NPG+Log Linear Case
(just notation for sample based approach)

- For a state-action distribution υ, define:
  $L(w;\theta,\upsilon) := E_{s,a\sim\upsilon}\left[(Q^{\pi_\theta}(s,a) - w\cdot\phi_{s,a})^2\right]$.
- The NPG update is equivalent to:
  1. approximate the Q^θ with the the features:
     $\widehat{w}^{(t)} \approx \text{argmin}_w L(w;\theta^{(t)},d^{(t)})$.
     where $d^{(t)} = d(\cdot\,|\,\pi^{(t)},\mu)$ is the "on-policy" distribution, starting at $s_0, a_0 \sim \mu$

NPG+Log Linear Case
(just notation for sample based approach)

- For a state-action distribution υ, define:
  $L(w;\theta,\upsilon) := E_{s,a\sim\upsilon}\left[(Q^{\pi_\theta}(s,a) - w\cdot\phi_{s,a})^2\right]$.
- The NPG update is equivalent to:
  1. approximate the Q^θ with the the features:
     $\widehat{w}^{(t)} \approx \text{argmin}_w L(w;\theta^{(t)},d^{(t)})$.
     where $d^{(t)} = d(\cdot\,|\,\pi^{(t)},\mu)$ is the "on-policy" distribution, starting at $s_0, a_0 \sim \mu$
  2. policy update: $\pi(a\,|\,s)^{(t+1)} = \pi^{(t)}(a\,|\,s)\exp(\eta w^{(t)}\cdot\phi_{s,a})/Z_s$

NPG: Conv. Rate with Approx+Est. Errors

NPG: Conv. Rate with Approx+Est. Errors

- Supervised learning error: Suppose the excess risk and approx error are bounded as:

$$L(w^{(t)}; \theta^{(t)}, d^{(t)}) - L(w_\star^{(t)}; \theta^{(t)}, d^{(t)}) \leq \epsilon_{\text{stat}},$$

$$L(w_\star^{(t)}; \theta^{(t)}, d^{(t)}) \leq \epsilon_{\text{approx}},$$

NPG: Conv. Rate with Approx+Est. Errors

- Supervised learning error: Suppose the excess risk and approx error are bounded as:

  $L(w^{(t)}; \theta^{(t)}, d^{(t)}) - L(w_\star^{(t)}; \theta^{(t)}, d^{(t)}) \leq \epsilon_{\text{stat}}$,

  $L(w_\star^{(t)}; \theta^{(t)}, d^{(t)}) \leq \epsilon_{\text{approx}}$,

- Conditioning (i.e. “feature coverage”): $\|\phi_{s,a}\| \leq 1$ and define

$$\kappa = 1/\sigma_{\min}\left(E_{s,a\sim\mu}\left[\phi_{s,a}\phi_{s,a}^\top\right]\right)$$

NPG: Conv. Rate with Approx+Est. Errors

- Supervised learning error: Suppose the excess risk and approx error are bounded as:

  $L(w^{(t)}; \theta^{(t)}, d^{(t)}) - L(w_\star^{(t)}; \theta^{(t)}, d^{(t)}) \le \epsilon_{\text{stat}}$,

  $L(w_\star^{(t)}; \theta^{(t)}, d^{(t)}) \le \epsilon_{\text{approx}}$,

- Conditioning (i.e. "feature coverage"): $\|\phi_{s,a}\| \le 1$ and define

$$\kappa = 1/\sigma_{\min}\left(E_{s,a\sim\mu}\left[\phi_{s,a}\phi_{s,a}^\top\right]\right)$$

Theorem [NPG]

A : #actions, H : Horizon $= 1/(1-\gamma)$

After T iterations, the NPG algorithm returns a π s.t.

$$V^{(T)}(s_0) \ge V^\star(s_0) - HW\sqrt{\frac{2\log A}{T}} + \sqrt{4AH^3\left(\kappa\cdot\epsilon_{\text{stat}} - \left\|\frac{d^\star}{\mu}\right\|_\infty \cdot \epsilon_{\text{approx}}\right)}$$

where $\left\|\frac{a}{b}\right\|_\infty = \max_i \left|\frac{a_i}{b_i}\right|$.

NPG: Conv. Rate with Approx+Est. Errors

- Supervised learning error: Suppose the excess risk and approx error are bounded as:

$$L(w^{(t)}; \theta^{(t)}, d^{(t)}) - L(w^{(t)}_\star; \theta^{(t)}, d^{(t)}) \le \epsilon_{\text{stat}},$$

$$L(w^{(t)}_\star; \theta^{(t)}, d^{(t)}) \le \epsilon_{\text{approx}},$$

- Conditioning (i.e. "feature coverage"): $\|\phi_{s,a}\|$

$$\kappa = 1/\sigma_{\min}\left(E_s\right.$$

- This is a distribution shift factor, due to misalignment between $d^{(t)}$ and $d^\star$.
- This guarantee is stronger than approximate dynamic programming.
- See AJKS!

Theorem [NPG]

A : #actions, H : Horizon $= 1/(1-\gamma)$

After T iterations, the NPG algorithm returns a π s.t.

$$V^{(T)}(s_0) \ge V^\star(s_0) - HW\sqrt{\frac{2\log A}{T}} + \sqrt{4AH^3\left(\kappa \cdot \epsilon_{\text{stat}} - \left\|\frac{d^\star}{\mu}\right\|_\infty \cdot \epsilon_{\text{approx}}\right)}$$

where $\left\|\frac{a}{b}\right\|_\infty = \max_i \left|\frac{a_i}{b_i}\right|$.

Thank you!

Thank you!

- Theory and practice of PG methods:
  - Part I: PG methods as black box optimization
  - Part II: Variance reduction and baselines
  - Part III: Theoretical foundations: global convergence and generalization

Thank you!

- Theory and practice of PG methods:
  - Part I: PG methods as black box optimization
  - Part II: Variance reduction and baselines
  - Part III: Theoretical foundations: global convergence and generalization
- Supplementary material:
  - AJKS Book: Reinforcement Learning: Theory and Algorithms, Section 3
  - See colab notebooks!

Thank you!

- Theory and practice of PG methods:
  - Part I: PG methods as black box optimization
  - Part II: Variance reduction and baselines
  - Part III: Theoretical foundations: global convergence and generalization
- Supplementary material:
  - AJKS Book: Reinforcement Learning: Theory and Algorithms, Section 3
  - See colab notebooks!
- RL is a vibrant and increasingly important area. Please participate!

Thank you!

- Theory and practice of PG methods:
  - Part I: PG methods as black box optimization
  - Part II: Variance reduction and baselines
  - Part III: Theoretical foundations: global convergence and generalization
- Supplementary material:
  - AJKS Book: Reinforcement Learning: Theory and Algorithms, Section 3
  - See colab notebooks!
- RL is a vibrant and increasingly important area. Please participate!

Thanks for making the notebooks!


Alan Chan


Shivam Garg


Dhawal Gupta


Abhishek Naik