Approximately Optimal Approximate Reinforcement Learning

Sham Kakade and John Langford

The Goal: Reinforcement Learning on LARGE state spaces

What algorithms work?

Current Methods

1. Approximate value functions

2. Policy Gradient

Questions

1. Will the policy always improve?

2. Will the program halt?

3. Will the final policy be near-optimal?

Approximate Value function

1. Approximate state-action values.

2. Derive policy from approximate values.

$\max_s |V(s) - \widehat{V}(s)| \leq \epsilon \quad \Rightarrow$ each iteration doesn't *lose* much

Policy Iteration

Let γ = discount factor.

Let $\epsilon = \max_s |V(s) - \widehat{V}(s)|$ = approximation error

Theorem: The average reward of the next policy is no more than $O(\frac{\epsilon\gamma}{1-\gamma})$ worse

Note: some guarantees on (near) optimality exist.

Policy Gradient

$\eta(\pi) =$ average reward

Evaluate $\frac{\partial \eta(\pi)}{\partial w}$ and optimize

Fundamentally: intertwine explore and exploit

Directed exploration required

Policy Gradient Theorems

Theorem: The policy always improves with infinitesimally small steps.

Theorem: The gradient *magnitude* can be estimated quickly.

Theorem: The gradient *direction* can not always be estimated quickly, even far from the optima.

Next

1. RL terminology

2. Conservative Policy Iteration

3. Theorems

Terminology

1. $\mu(s)$ = start state distribution

2. $R(s,a)$ = reward recieved for action a in state s

3. $V_\pi(s) = (1-\gamma)E\sum_{t=0}^{\infty}\gamma^t R(s_t,a_t)$ = expected reward from state s

4. $Q_\pi(s,a) = (1-\gamma)R(s,a) + \gamma E_{s',a\sim P(s';s,a)\pi(a|s)}V_\pi(s')$ = expected reward from state s and action a

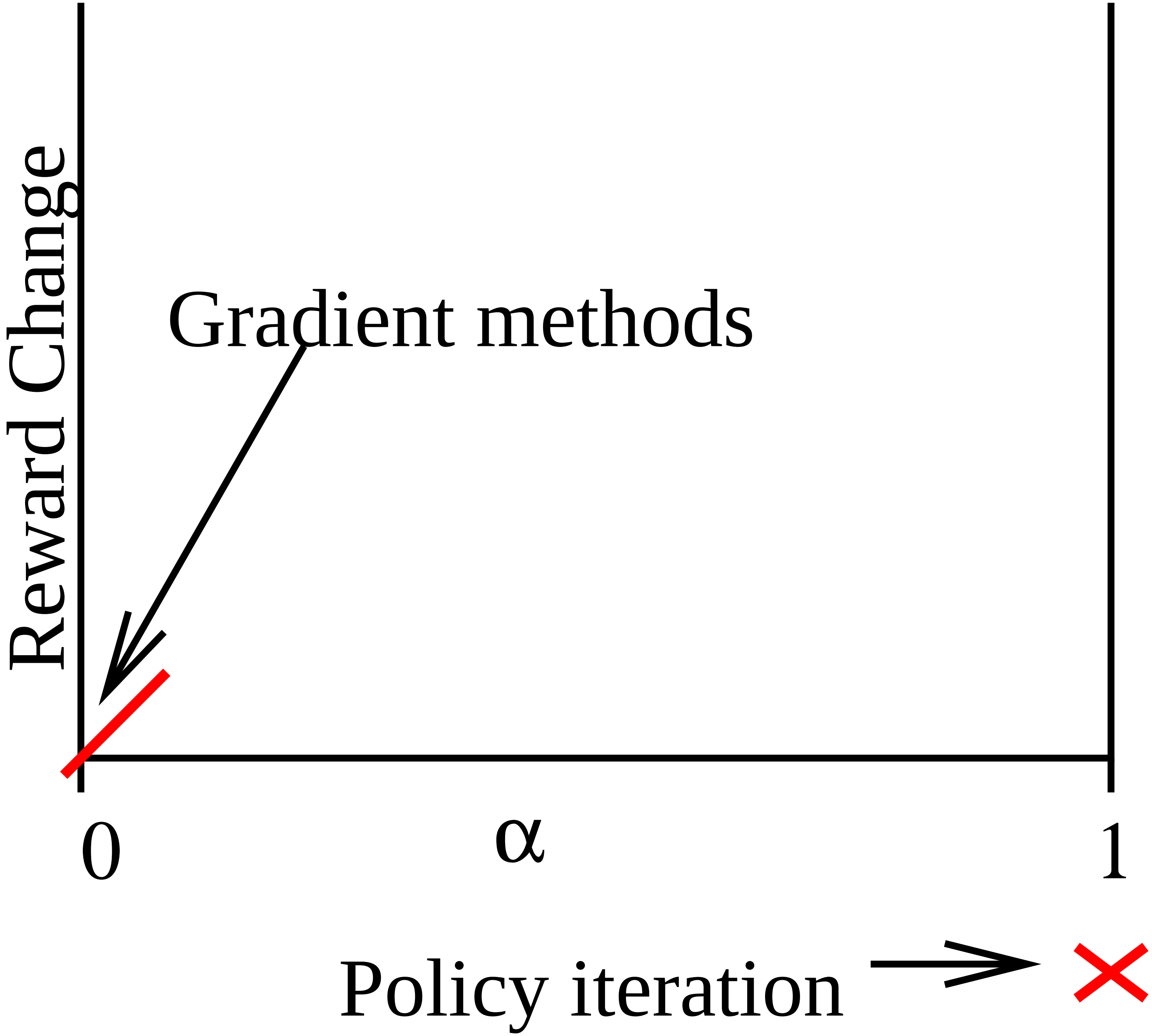
5. $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$ = advantage of taking one action over another

Consider update rules

$$\pi_{\text{new}} \leftarrow \alpha\pi' + (1-\alpha)\pi$$

$\alpha \to 0$ = policy gradient

$\alpha = 1$ = policy iteration


Reward Change
Gradient methods
0
α
1
Policy iteration

The idea

1. Use uniform state distribution $\mu \Rightarrow$ avoid exploration problems

2. Use finite $\alpha > 0$.

3. l_1 metric instead of l_∞ metric

Definitions

$$d_{\pi,\mu}(s) \equiv \Pr_{\pi,\mu,\text{world}}(s) \;=\; \text{natural metric on states}$$

$$\mathcal{A}_{\pi,\mu}(\pi') \equiv E_{s \sim d_{\pi,\mu}(s)} E_{a \sim \pi'(a;s)} A(s,a) = \;\text{policy advantage}$$

As $\alpha \to 0$,

$$\eta_\mu(\pi_{\text{new}}) - \eta_\mu(\pi) \propto \mathcal{A}_{\pi,\mu}(\pi')$$

Ingredient: Supervised Learning

Input space: s, a

Output space: $Q(s, a)$

Input distribution: $d_{\pi,\mu}(s), \pi(a|s)$

Output distribution: $\widehat{Q}(s, a)$ s.t. $E\widehat{Q}(s, a) = Q(s, a)$

Use greedy evaluation for new policy, π'.

Conservative Policy Iteration

Let $G_\mu(\pi)$ = regression algorithm finding π' with large policy advantage.

1. $\pi' = G_\mu(\pi)$

2. if $\mathcal{A}_{\pi,\mu}(\pi') > \epsilon$

   (a) then $\pi \leftarrow \alpha\pi' + (1-\alpha)\pi$, goto 1

   (b) else output π

Analysis

Theorem: (progress) The policy always improves:

$$\eta_\mu(\pi_{\text{new}}) - \eta_\mu(\pi) \geq \frac{\alpha}{1-\gamma}\left(\mathcal{A}_{\pi,\mu}(\pi') - \frac{2\alpha\gamma\epsilon}{1-\gamma(1-\alpha)}\right)$$

Theorem: (termination) CPI halts in $O\left(R^2/\epsilon^2\right)$ iterations.

Proof: Take derivative w.r.t. α, find extrema,

$\Rightarrow \eta_\mu(\pi_{\text{new}}) - \eta_\mu(\pi) \geq \frac{\mathcal{A}_{\pi,\mu}(\pi')^2}{8R}$

Theorem: (optimality) The output policy satisfies:

$$\eta_D(\pi^*) - \eta_D(\pi) \leq \frac{\epsilon}{1-\gamma} \max_s \frac{d_{\pi^*,D}(s)}{d_{\pi,\mu}(s)}$$

$$\leq \frac{\epsilon}{(1-\gamma)^2} \max_s \frac{d_{\pi^*,D}(s)}{\mu(s)}$$

Note: back to l_∞ norm.

Prior information $\rightarrow$ tune $\mu(s)$

Needed

1. (progress) Supervised learning (regression) algorithm
2. (progress & termination) Evaluation of Policy advantage
3. (optimality) μ restart distribution

Conclusion

RL = supervised learning + restart distribution

Work in progress

1. Algorithms for directed exploration

   (a) finite, MDP: solved (E^3 + Sven/Thrun/Simmons)

   (b) Large/infinite MDP: unsolved

2. Approximate model building

3. Experimental validation