Policy Gradient Methods for Reinforcement Learning with Function Approximation

NeurIPS 2000

Sutton McAllester Singh Mansour

Presenter: Silviu Pitis

Date: January 21, 2020

Talk Outline

- Problem statement, background & motivation
- Topics:
  - Statement of policy gradient theorem
  - Derivation of policy gradient theorem
  - Action-independent baselines
  - Compatible value function approximation
  - Convergence of policy iteration with compatible fn approx

Problem statement

We want to learn a parameterized behavioral policy:

$$\pi_\theta : S \to A$$

that optimizes the long-run sum of (discounted) rewards:

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[R(\tau)] = \mathbb{E}_{\tau \sim \pi_\theta} \left\{ \sum_{t=0}^{\infty} \gamma^t r_t \;\middle|\; s_0 \right\}$$

This is exactly the reinforcement learning problem!

note: the paper also considers the average reward formulation (same results apply)

Traditional approach: Greedy value-based methods

Traditional approaches (e.g., DP, Q-learning) learn a **value function**:

$$Q_\theta(s,a) = \mathbb{E}_{\tau\sim\pi_\theta}\left\{\sum_{t=0}^{\infty}\gamma^t r_t \;\middle|\; s_0 = s, a_0 = a\right\}$$

$$J(\theta) = \mathbb{E}_{a_0\sim\pi_\theta(s_0)}\, Q_\theta(s_0,a_0)$$

They then induce a policy using a greedy argmax:

$$\pi_\theta(s) = \mathrm{argmax}_a\, Q_\theta(s,a)$$

Two problems with greedy, value-based methods

1) They can diverge when using function approximation, as small changes in the value function can cause large changes in the policy

In fully observed, tabular case, guaranteed to have an optimal deterministic policy.

2) Traditionally focused on deterministic actions, but optimal policy may be stochastic when using function approximation (or when environment is partially observed)

Proposed approach: Policy gradient methods

- Instead of acting greedily, policy gradient approaches parameterize the policy directly, and optimize it via gradient descent on the cost function:

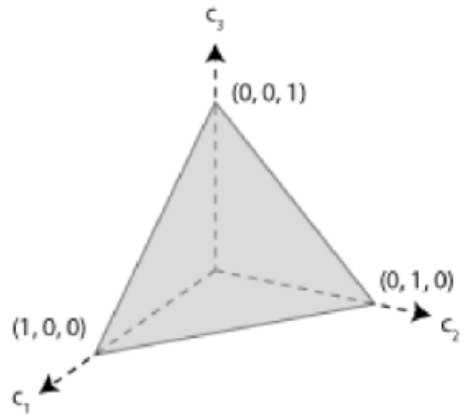
$$\pi_\theta^{(t+1)} = \pi_\theta^{(t)} - \alpha \nabla_\theta J(\theta)$$

- **NB1:** cost must be differentiable with respect to theta! Non-degenerate, stochastic policies ensure this.
- **NB2:** Gradient descent converges to a local optimum of the cost function → so do policy gradient methods, but only if they are unbiased!

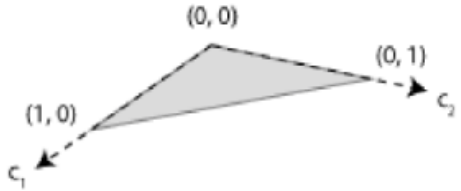
Stochastic Policy Value Function Visualization

We use stochastic policies, because the expectation of an arbitrary cost function is differentiable with respect to them.

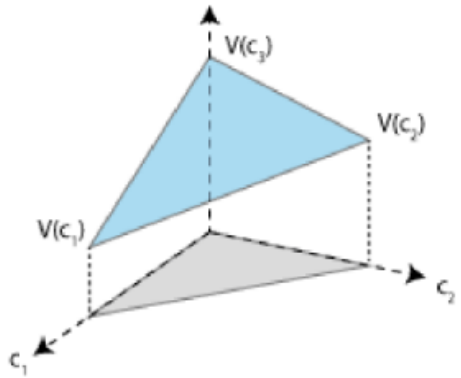
Discrete Case: Visually



Stochastic action space in 3-d



2-d parameterization of stochastic action space



Value as a function of the 2-d parameterization

Source: Me (2018)

Stochastic Policy Gradient Descent Visualization

$$\theta_{k+1} := \theta_k + \eta \nabla_\theta J(\theta_k).$$

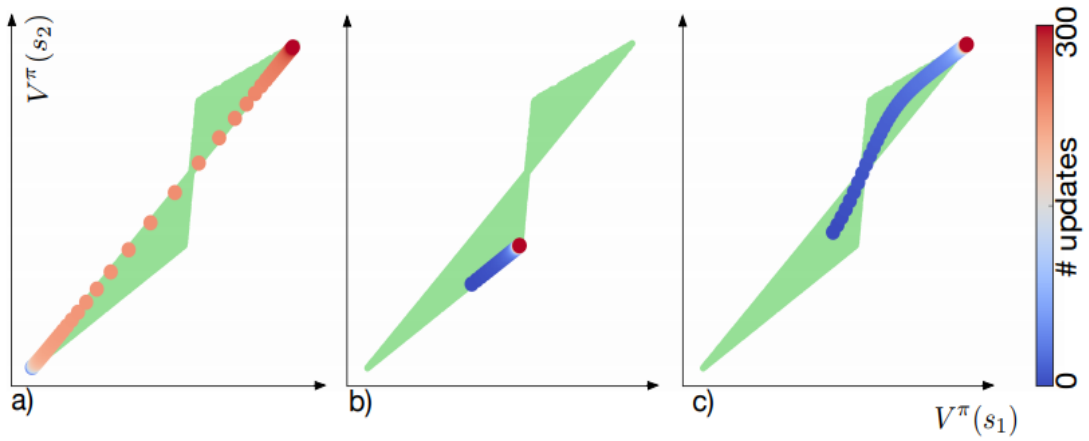


Figure 9. Value functions generated by policy gradient.

Figure 9 shows that the convergence rate of policy gradient strongly depends on the initial condition. In particular, Figure 9a),b) show accumulation points along the update path (not shown here, the method does eventually converge to V^*). This behaviour is sensible given the dependence of $\nabla_\theta J(\theta)$ on $\pi(\cdot \mid s)$, with gradients vanishing at the boundary of the polytope.

Source: Dadashi et. al. (ICLR 2019)

Unbiasedness is critical

- Gradient descent converges → so do **unbiased** policy gradient methods!

- Recall the definition of the bias of an estimator:
  - An estimator $\hat{X}$ of X has bias: $\mathbb{E}\left[\hat{X} - X\right]$
  - It is unbiased if its bias equals 0.

- This is important to keep in mind, as not all policy gradient algorithms are unbiased, so may not converge to a local optimum of the cost function.

Recap

- Traditional value-based methods may diverge when using function approximation → directly optimize the policy using gradient descent

Let's now look at the paper's 3 contributions:

1) Policy gradient theorem --- statement & derivation

2) Baselines & compatible value function approximation

3) Convergence of Policy Iteration with compatible function approx

Policy gradient theorem (2 forms)

Recall the objective:

Sutton 2000

NB: This is the true future value of the policy, not an approximation!

$$\nabla_\theta J(\theta) = \int_s \mathrm{d}s\, d^\pi(s) \int_a \mathrm{d}a\, \nabla_\theta \pi_\theta(a|s)\, Q^{\pi_\theta}(s,a)$$

Modern form

$$\nabla_\theta J(\theta) = E_{(s,a)\sim\pi_\theta} [\nabla \log \pi_\theta(a|s)\ Q^{\pi_\theta}(s,a)]$$

The two forms are equivalent

$$\nabla_\theta J(\theta) = \int_s \mathrm{d}s \; d^\pi(s) \int_a \mathrm{d}a \; \nabla_\theta \pi_\theta(a|s) \; Q^{\pi_\theta}(s,a)$$

(Sutton 2000)

$$= \int_s \mathrm{d}s \; d^\pi(s) \int_a \mathrm{d}a \; \pi_\theta(a|s) \frac{\nabla_\theta \pi_\theta(a|s)}{\pi_\theta(a|s)} \; Q^{\pi_\theta}(s,a)$$

$$= E_{(s,a)\sim\pi_\theta} [\nabla \log \pi_\theta(a|s) \; Q^{\pi_\theta}(s,a)]$$

(Modern form)

Trajectory Derivation: REINFORCE Estimator

$$V_\theta = \mathbb{E}_{\tau|\theta}[R(\tau)] = \int \mathrm{d}\tau \; \pi_\theta(\tau) R(\tau)$$

$$\nabla V_\theta = \nabla \int \mathrm{d}\tau \; \pi_\theta(\tau) R(\tau)$$

$$= \int \mathrm{d}\tau \; \nabla \pi_\theta(\tau) R(\tau)$$

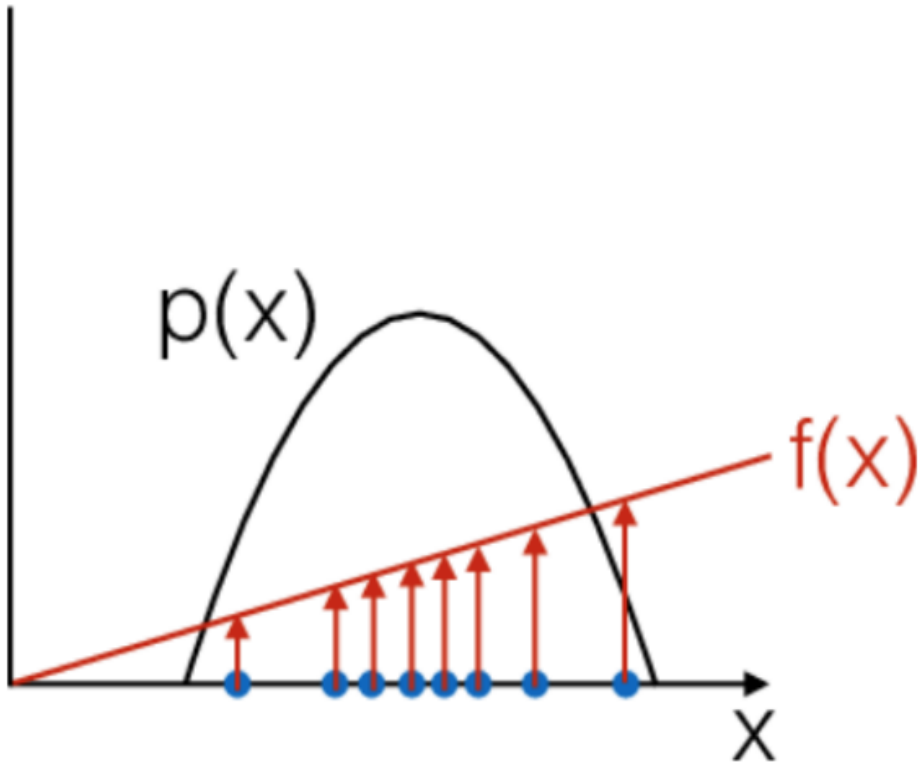
$$= \int \mathrm{d}\tau \; \pi_\theta(\tau) \nabla \log \pi_\theta(\tau) R(\tau)$$

"Score function gradient estimator" also known as "REINFORCE gradient estimator" --- very generic, and very useful!

NB: R(tau) is arbitrary (i.e., can be non-differentiable!)

Intuition of Score function gradient estimator

$$\hat{g}_i = f(x_i)\nabla_\theta \log p(x_i|\theta)$$



Source: Emma Brunskill

Trajectory Derivation Continued

$$\nabla V_\theta = \mathbb{E}_\theta \left[\nabla \log \pi_\theta(\tau) R(\tau) \right]$$

$$= \mathbb{E}_\theta \left[\nabla \log \pi_\theta(\tau) \left(\sum_{t=0}^{T-1} r_t \right) \right]$$

$$= \mathbb{E}_\theta \left[\nabla \log \left(p(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t) P(s_{t+1}|s_t, a_t) \right) \left(\sum_{t=0}^{T-1} r_t \right) \right]$$

$$= \mathbb{E}_\theta \left[\left(\sum_{t=0}^{T-1} \nabla \log \pi_\theta(a_t|s_t) \right) \left(\sum_{t=0}^{T-1} r_t \right) \right]$$

Almost in modern form!
Just one more step...

Trajectory Derivation, Final Step

$$\nabla V_\theta = \mathbb{E}_\theta \left[\sum_{t=0}^{T-1} \nabla \log \pi_\theta(a_t|s_t) \sum_{i=0}^{T-1} r_i \right]$$

$$= \mathbb{E}_\theta \left[\sum_{t=0}^{T-1} \nabla \log \pi_\theta(a_t|s_t) \sum_{i=t}^{T-1} r_i \right]$$

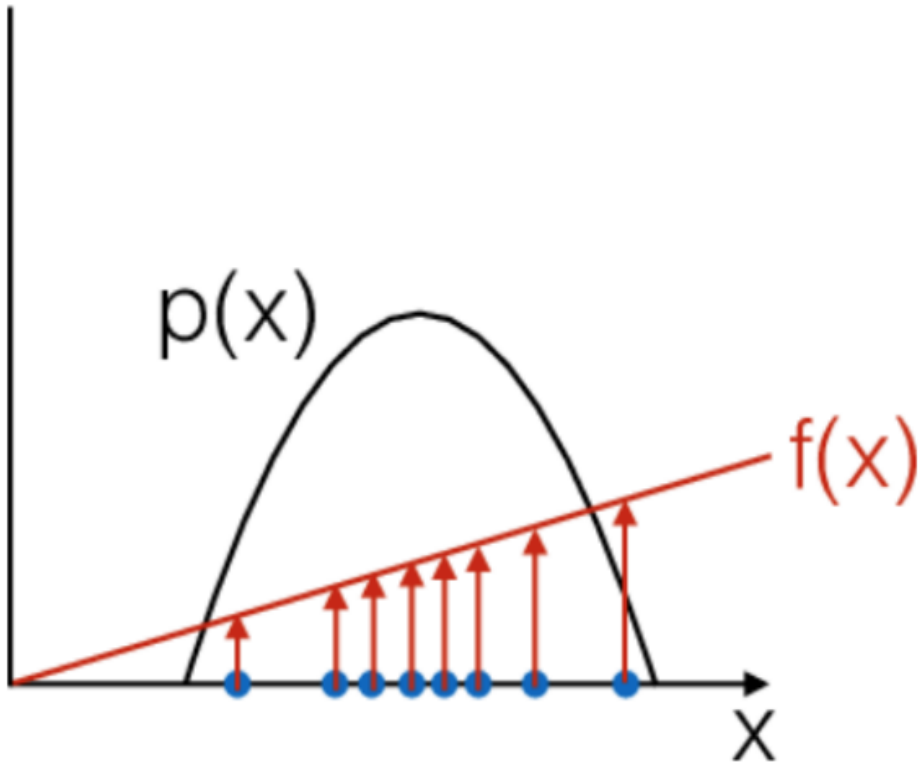
Since earlier rewards do not depend on later actions.

$$= \mathbb{E}_\theta \left[\sum_{t=0}^{T-1} \nabla \log \pi_\theta(a_t|s_t) Q^{\pi_\theta}(s_t, a_t) \right]$$

And this now (proportional to) modern form!

Variance Reduction

$$\hat{g}_i = f(x_i)\nabla_\theta \log p(x_i|\theta)$$



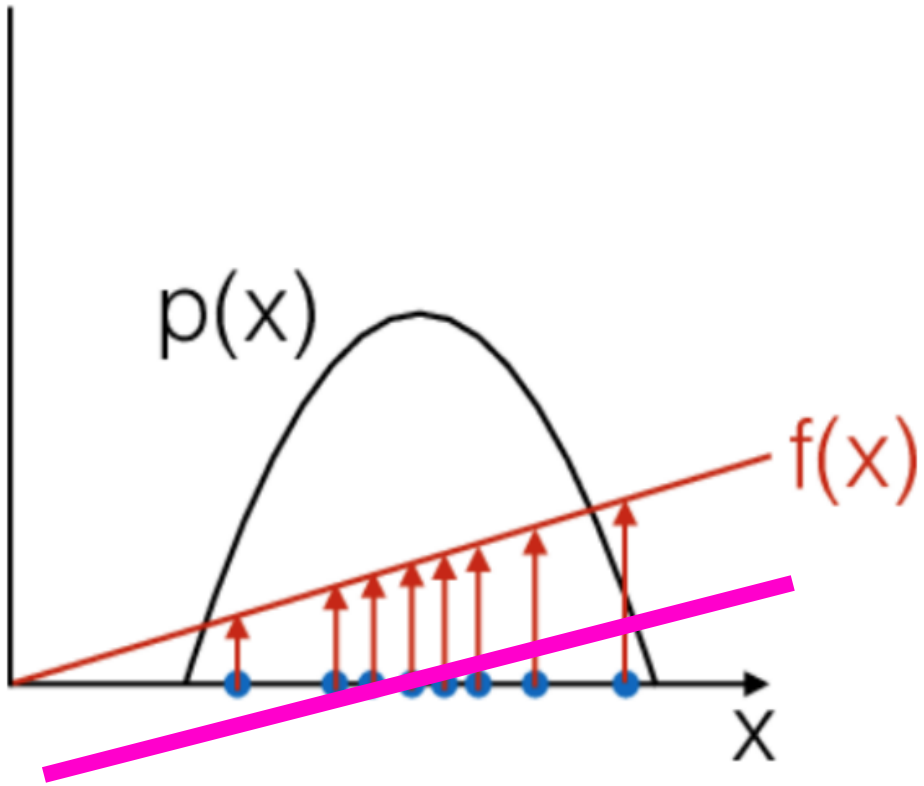
If f(x) is positive everywhere, we are always positively reinforcing the same policy!

If we could somehow provide negative reinforcement for bad actions, we can reduce variance...

Source: Emma Brunskill

Variance Reduction

$$\hat{g}_i = f(x_i)\nabla_\theta \log p(x_i|\theta)$$



If f(x) is positive everywhere, we are always positively reinforcing the same policy!

If we could somehow provide negative reinforcement for bad actions, we can reduce variance...

Source: Emma Brunskill

Last step: Subtracting an Action-independent Baseline I

- Note that, in general

$$\begin{aligned}\mathbb{E}\left[b\nabla_\theta \log \pi(A_t|S_t)\right] &= \mathbb{E}\left[\sum_a \pi(a|S_t) b\nabla_\theta \log \pi(a|S_t)\right] \\ &= \mathbb{E}\left[b\nabla_\theta \sum_a \pi(a|S_t)\right] \\ &= \mathbb{E}\left[b\nabla_\theta 1\right] \\ &= 0\end{aligned}$$

- This holds only if b does not depend on the action (though it can depend on the state)
- Implies we can subtract a baseline to reduce variance

Source: Hado Van Hasselt

Last step: Subtracting an Action-independent Baseline II

- A good baseline is $v_\pi(S_t)$

$$\nabla_\theta J_\theta(\pi) = \mathbb{E}\left[\sum_{t=0} \nabla_\theta \log \pi(A_t|S_t)(q_\pi(S_t, A_t) - v_\pi(S_t))\right]$$

Source: Hado Van Hasselt

Compatible Value Function Approximation

- Policy gradient theorem uses an unbiased estimator of the future rewards, $Q^{\pi_\theta}(s, a)$
- What if we use a value function $Q^w(s, a)$ to approximate $Q^{\pi_\theta}(s, a)$? Does our convergence guarantee disappear?
- In general, yes.
- But not if we use a *compatible function approximator* --- Sutton et al. Provides a sufficient (but strong) condition for a function approximator to be compatible (i.e., provide an unbiased policy gradient estimate).

Compatible Function Approximation

- If the following two conditions are satisfied:

  1. Value function approximator is compatible to the policy

$$\nabla_w Q_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)$$

  2 Value function parameters w minimize the mean-squared error

$$\varepsilon = \mathbb{E}_{\pi_\theta}\left[(Q^{\pi_\theta}(s,a) - Q_w(s,a))^2\right]$$

- Then the policy gradient is exact,

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ Q_w(s,a)\right]$$

- Remember:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ Q^{\pi_\theta}(s,a)\right]$$

Source: Russ Salakhutdinov

Proof

- If w is chosen to minimize mean-squared error, gradient of ε w.r.t. w must be zero,

$$\nabla_w \varepsilon = 0$$
$$\mathbb{E}_{\pi_\theta}\left[(Q^\theta(s,a) - Q_w(s,a))\nabla_w Q_w(s,a)\right] = 0$$
$$\mathbb{E}_{\pi_\theta}\left[(Q^\theta(s,a) - Q_w(s,a))\nabla_\theta \log \pi_\theta(s,a)\right] = 0$$
$$\mathbb{E}_{\pi_\theta}\left[Q^\theta(s,a)\nabla_\theta \log \pi_\theta(s,a)\right] = \mathbb{E}_{\pi_\theta}\left[Q_w(s,a)\nabla_\theta \log \pi_\theta(s,a)\right]$$

- So $Q_w(s, a)$ can be substituted directly into the policy gradient,

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)\right]$$

Source: Russ Salakhutdinov

Recap: Compatible Value Function Approx.

- If we approximate the true future reward $Q^{\pi_\theta}(s, a)$ with an approximator $Q^w(s, a)$ such that $\nabla_w Q^w(s, a) = \nabla_\theta \log \pi_\theta(a|s)$ the policy gradient estimator remains unbiased → gradient descent converges to a local optimum.
- Sutton uses this this to prove the convergence of policy iteration when using a compatible value function approximator.

Critique I: Bias & Variance Tradeoffs

- Monte Carlo returns provide high variance estimates, so we typically want to use a **critic** $Q^w(s, a)$ to estimate future returns.
- But unless the critic is compatible, it will introduce bias.
- "Tsitsiklis (personal communication) points out that [the critic] being linear in $\nabla_\theta \log \pi_\theta(a|s)$ may be the only way to satisfy the [compatible value function approximation] condition."
- Empirically speaking, we use non-compatible (biased) critics because they perform better.

Critique II: Policy Gradients are On Policy

$$\nabla_\theta J(\theta) = E_{(s,a)\sim\pi_\theta}\left[\nabla \log \pi_\theta(s,a)\ Q^{\pi_\theta}(s,a)\right]$$

- The policy gradient theorem is, by definition, **on policy**.
- **Recall**: *on-policy* methods learn from dat that they themselves generate; *off-policy* methods (e.g., Q-learning) can learn from data produced by other (possibly unknown) policies.
- To use off-policy data with policy gradients, we need to use importance sampling, which results in high variance.
- Limits the ability to use data from previous iterates.

Recap

- Traditional value-based methods may diverge when using function approximation → directly optimize the policy using gradient descent
- We do this with the **policy gradient theorem:**

$$\nabla_\theta J(\theta) = E_{(s,a)\sim\pi_\theta}\left[\nabla \log \pi_\theta(a|s)\ Q^{\pi_\theta}(s,a)\right]$$

- Some key takeaways:
  - REINFORCE log-gradient trick is very useful (know it!)
  - We can reduce the variance by using a baseline
  - There is thing called compatible approximation, but to my knowledge its not so practical
  - IMO, the main limitation of policy gradient methods is their on-policyness (but see DPG!)