An Introduction to Stochastic Control and Reinforcement Learning

Subhrakanti Dey and Simone Garatti

Signals and Systems, Dept of Electrical Engineering, Uppsala University
Politechnico Di Milano, Milan

SIDRA Summer School, Bertinoro, Italy

July 2025

Policy Gradient and Actor-Critic Methods and applications

- **Acknowledgements**: Some slides, images and content taken from
  1. Reinforcement Learning: An Introduction. Richard S. Sutton and Andrew G. Barto, second edition, 2018.
  2. UCL Course, Reinforcement Learning, videos and slides. David Silver, 2015.
  3. Lecture slides, Department of Informatics, University of Pisa
- Lecture contents: Policy Gradient based reinforcement learning, Policy Gradient theorem, computation of policy gradients, Different variations of policy gradient algorithms, application to LQG control

Policy based RL

From value to policy

- In value based PI, we learn action-values and then select actions: first values, then the policy.
- The policy would not even exist without the action-value estimates.
- Why not learn directly the policy?

Policy approximation

Approximate the policy by $\boldsymbol{\theta} \in \mathbb{R}^{d'}$:

$$\pi_{\boldsymbol{\theta}}(a|s) = \pi(a|s, \boldsymbol{\theta}) = \Pr(A_t = a | S_t = s, \boldsymbol{\theta}_t = \boldsymbol{\theta})$$

Parametrization

Discrete and not too large action space

Common choice: numerical preferences $h(s, a, \boldsymbol{\theta})$ for state-action.

- highest preferences = highest probabilities, with **soft-max distribution**:

$$\pi_{\boldsymbol{\theta}}(a|s) = \frac{e^{h(s,a,\boldsymbol{\theta})}}{\sum_{b \in \mathcal{A}} e^{h(s,b,\boldsymbol{\theta})}}.$$

- Preferences $h(s, a, \boldsymbol{\theta})$ can be computed by linear approximation, with features, or by a neural network with weights $\boldsymbol{\theta}$. The latter is the approach used in AlphaGo family.
- Soft-max choice is flexible: if the optimal policy is deterministic, optimal actions will be driven infinitely higher than all suboptimal actions.
- Soft-max choice is flexible: can approximate a stochastic policy - not possible with action-value methods - why?

Value-based and policy-based RL

Value-based

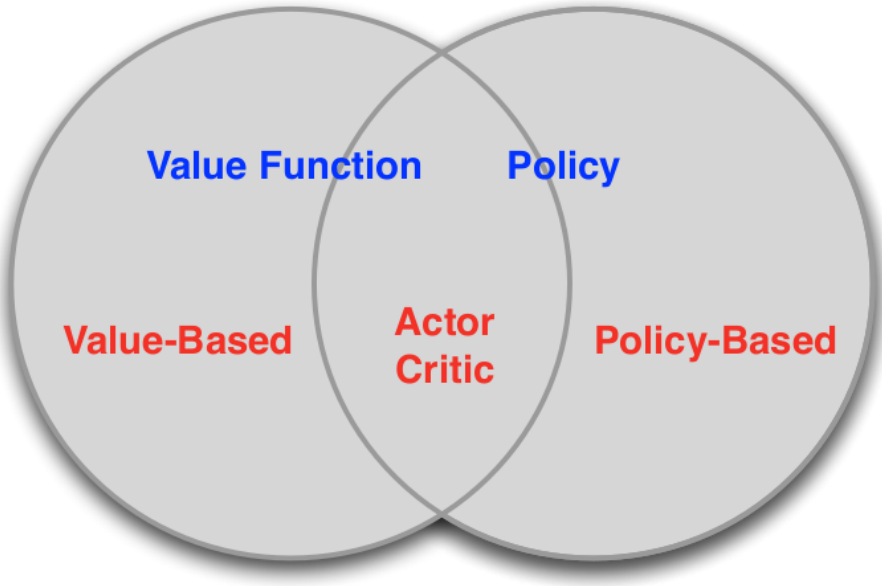
Learnt value function, implicitly defined policy.

Policy-based

No value function, learnt policy.

Actor-critic

Learnt value function, learnt policy. Value function used to "criticize" (improve) the policy.



Why learning a policy

Advantages

- Effective in continuous action spaces.
- Can learn both stochastic and deterministic policies.
- Policy can be a simpler function to approximate.
- Choice of policy parameterization: prior knowledge about the desired form of the policy. Often the most important reason.
- Smooth change of action probabilities (ϵ-greedy policy may change dramatically, why?). Thus, better convergence.

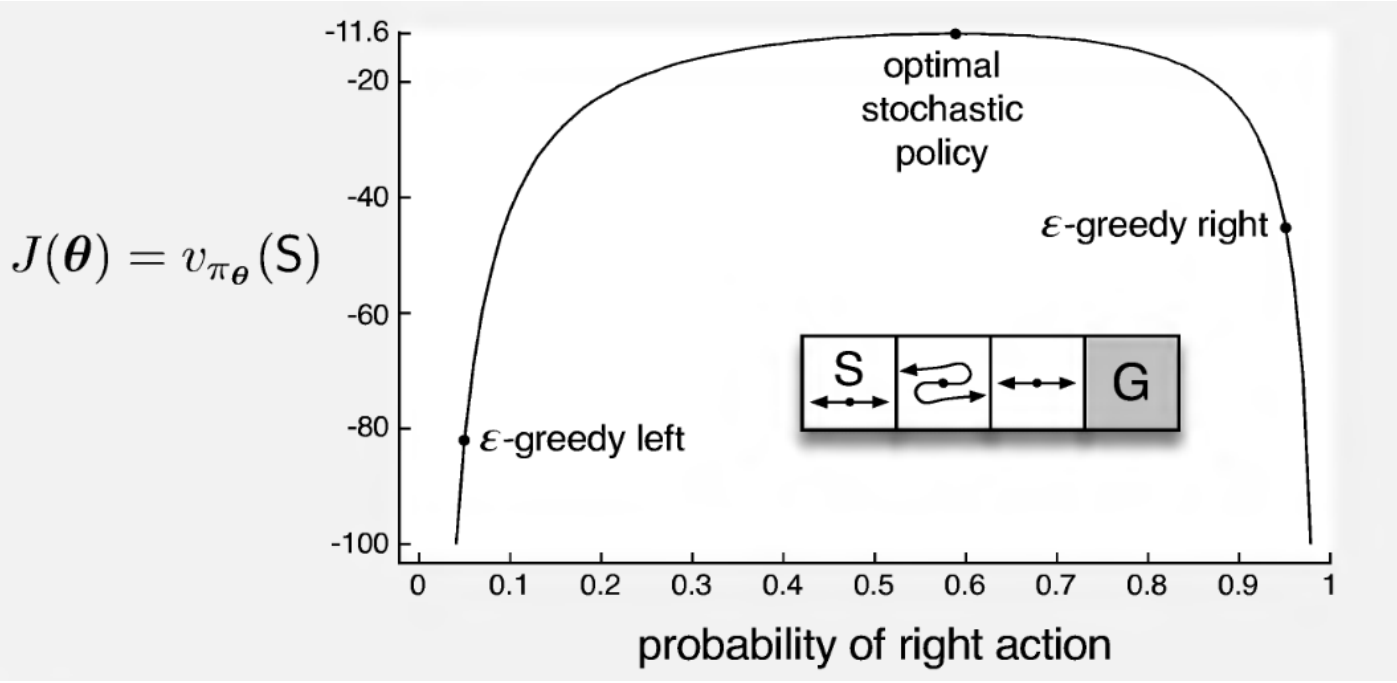
Disadvantages

- Typically converge to a local rather than global optimum.
- Evaluating a policy is typically **sample inefficient** and high variance.

Example: a game where the optimal policy is stochastic

Rock, Paper, Scissors (Lizard, Spock). A deterministic policy is easily exploited, the optimal policy is uniform random.

Example



Short gridworld with switched actions

- (s, a) are approximated by features $x(s, \text{right}) = (1, 0)$ and $x(s, \text{left}) = (0, 1)$ for all s.
- ϵ-greedy action-value methods must choose between two policies: right or left, with probability $1 - \epsilon/2$.
- With $\epsilon = 0.1$, π_{left} achieves less than -82 in S, and π_{right} less than -44. The optimal policy achieves -11.6.

How to measure the quality of a policy?

Question

Propose a way to say whether a policy $\pi_{\boldsymbol{\theta}}$ is better or worst than another $\pi_{\boldsymbol{\theta}'}$.

Answer

A policy $\pi_{\boldsymbol{\theta}}$ is better than $\pi_{\boldsymbol{\theta}'}$ if . . . [YOUR TURN]

How to measure the quality of a policy?

Question

Propose a way to say whether a policy $\pi_{\boldsymbol{\theta}}$ is better or worst than another $\pi_{\boldsymbol{\theta}'}$.

Answer

A policy $\pi_{\boldsymbol{\theta}}$ is better than $\pi_{\boldsymbol{\theta}'}$ if . . . [YOUR TURN]

Hint: what was the answer to the same question in value-based methods?

The starting point for function approximation methods was choosing a loss function $f : \mathbb{R}^d \to \mathbb{R}$ to optimize. We chose the Mean Squared Value Error:

$$f(w) = \overline{\text{VE}(w)} = \sum_{s \in \mathcal{S}} \mu_\pi(s)[v_\pi(s) - \hat{v}(s, w)]^2$$

How to measure the quality of a policy?

A natural objective function: the value of starting states

Maximize the start value: $J(\boldsymbol{\theta}) := v_{\pi_{\boldsymbol{\theta}}}(s_0)$. Optimization problem. Several methods avalaible. We use gradient ascent.

Recall the value function definition

Denoting by τ trajectories in the MDP, the value function of a policy π is:

$$\begin{aligned} v_\pi(s_0) &= \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s_0] \\ &= \sum_\tau \mathbb{P}_{\pi,p}(\tau = s_0, a_0, r_1, s_1, \dots)\text{return}(\tau) \end{aligned}$$

Question

Given a trajectory τ starting from s_0:

$$\tau = s_0, a_0, r_1, s_1, a_1, r_2, \dots$$

what is its probability $\mathbb{P}_{\pi,p}(\tau)$?

Why is computing $\nabla_\theta(J)$ a problem?

Value function definition

Denoting by τ trajectories in the MDP, the value function of a policy π is:

$$\begin{aligned} v_\pi(s_0) &= \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s_0] \\ &= \sum_\tau \mathbb{P}_{\pi,p}(\tau = s_0, a_0, r_1, s_1, \dots)\text{return}(\tau) \end{aligned}$$

What happens when π is π_θ?

When $\pi = \pi_\theta$, the probability of τ depends on θ:

$$\mathbb{P}_{\pi_\theta,p}(\tau) = \pi_\theta(a_0|s_0)p(s_1, r_1|s_0, a_0)\pi_\theta(a_1|s_1)p(s_2, r_2|s_1, a_1) \cdot \dots$$

Why is computing $\nabla_\theta(J)$ a problem?

A distributional point of view

The value function of a policy π can be rewritten as:

$$\begin{aligned} v_\pi(s_0) &= \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s_0] \\ &= \sum_\tau \mathbb{P}_{\pi,p}(\tau = s_0, a_0, r_1, s_1, \dots)\text{return}(\tau) \\ &= \sum_{s \in \mathcal{S}} \mu_{\pi,s_0}(s) \sum_{a \in \mathcal{A}} \pi(a|s) r(s,a) \\ &= \sum_{s \in \mathcal{S}, a \in \mathcal{A}} \mu_{\pi,s_0}(s,a) r(s,a) \end{aligned}$$

Why is computing $\nabla_\theta(J)$ a problem?

Problem

Since the probability of trajectories τ depends on the **unknown** distribution model p, we cannot compute **exactly** the gradient ∇J.

Remark

We don't need an exact solution! As long as we can **estimate** the gradient, we can find a θ **close** to the maximum, and thus an agent that will do **approximately** well.

Question

Propose a strategy to estimate ∇J. Hint: remember that

$$\nabla J(\boldsymbol{\theta}) = \nabla v_{\pi_{\boldsymbol{\theta}}}(s_0) = \nabla \mathbb{E}[\dots]$$

Why is computing $\nabla_\theta(J)$ a problem?

Problem

Since the probability of trajectories τ depends on the **unknown** distribution model p, we cannot compute **exactly** the gradient ∇J.

Remark

We don't need an exact solution! As long as we can **estimate** the gradient, we can find a θ **close** to the maximum, and thus an agent that will do **approximately** well.

Question

Propose a strategy to estimate ∇J. Hint: remember that

$$\nabla J(\boldsymbol{\theta}) = \nabla v_{\pi_{\boldsymbol{\theta}}}(s_0) = \nabla \mathbb{E}[\dots]$$

Strategy

Write the gradient ∇J as an expected value $\mathbb{E}[\dots]$ of something that can be estimated.

Analytical computation

Example

Consider a simple **one-step** MDP: every episode starts from a state s_0 and terminates after one time-step, with reward $q_{\pi_\theta}(s_0, a)$.

Exercise

Compute the gradient $\nabla J(\theta)$ of the start value function $J(\theta) = v_{\pi_\theta}(s_0)$ for one-step MDP.

Analytical computation

Exercise

Consider a simple **one-step** MDP: every episode starts from a state s_0 and terminates after one time-step, with reward $q_{\pi_\theta}(s_0, a)$. Compute the gradient $\nabla J(\theta)$.

Solution

The gradient $\nabla J(\theta)$ of the start value function $J(\theta) = v_{\pi_\theta}(s_0)$ is given by:

$$\begin{aligned}
\nabla J(\theta) &= \nabla \sum_a \pi_\theta(a|s_0) q_{\pi_\theta}(s_0, a) \\
&= \sum_a (\nabla \pi_\theta(a|s_0) q_{\pi_\theta}(s_0, a) + \pi_\theta(a|s_0) \nabla q_{\pi_\theta}(s_0, a)) \\
&= \sum_a q_{\pi_\theta}(s_0, a) \nabla \pi_\theta(a|s_0).
\end{aligned}$$

Analytical computation

Policy gradient theorem

The gradient $\nabla J(\boldsymbol{\theta})$ of the start value function $J(\boldsymbol{\theta}) = v_{\pi_{\boldsymbol{\theta}}}(s_0)$ is given by:

$$\nabla J(\boldsymbol{\theta}) = K \sum_s \mu(s) \sum_a q_{\pi_{\boldsymbol{\theta}}}(s, a) \nabla \pi_{\boldsymbol{\theta}}(a|s)$$

where the constant K is the average length of episodes, and μ is the on-policy distribution under $\pi_{\boldsymbol{\theta}}$.

Proof

Page 325 of Sutton-Barto. Look at this link for details on the last equality.

Use PG Theorem to estimate the gradient

Overall strategy

Write the gradient as an expected value, so that we can sample it.

Question

Can you write $\nabla J(\boldsymbol{\theta})$ as an expected value? Something like this:

$$\begin{aligned}\nabla J(\boldsymbol{\theta}) &\propto \sum_s \mu(s) \sum_a q_{\pi_{\boldsymbol{\theta}}}(s,a) \nabla \pi_{\boldsymbol{\theta}}(a|s) \\ &= \mathbb{E}_?[?]\end{aligned}$$

Hint

The on-policy distribution μ is a probability distribution.

Use PG Theorem to estimate the gradient

Overall strategy

Write the gradient as an expected value, so that we can sample it.

Answer

Can you write $\nabla J(\boldsymbol{\theta})$ as an expected value? Yes!

$$\begin{aligned}\nabla J(\boldsymbol{\theta}) &\propto \sum_s \mu(s) \sum_a q_{\pi_{\boldsymbol{\theta}}}(s,a) \nabla \pi_{\boldsymbol{\theta}}(a|s) \\ &= \mathbb{E}_{s\sim\mu}[\sum_a q_{\pi_{\boldsymbol{\theta}}}(s,a) \nabla \pi_{\boldsymbol{\theta}}(a|s)]\end{aligned}$$

Can you estimate the above expectation?

The on-policy distribution μ measures how often a state s occurs under the target policy π: if π is followed, then states will be encountered in $\mu(s)$ proportions (on average). Then...

Use PG Theorem to estimate the gradient

Overall strategy

Write the gradient as an expected value **that can be sampled by trajectories**.

Answer

By following online trajectories, states will be visited in correct proportions:

$$\begin{aligned}\nabla J(\boldsymbol{\theta}) &\propto \sum_s \mu(s) \sum_a q_{\pi_{\boldsymbol{\theta}}}(s,a) \nabla \pi_{\boldsymbol{\theta}}(a|s) \\ &= \mathbb{E}_{s\sim\mu}[\sum_a q_{\pi_{\boldsymbol{\theta}}}(s,a) \nabla \pi_{\boldsymbol{\theta}}(a|s)] \\ &= \mathbb{E}_{\tau\sim\mathbb{P}_{\pi_\theta,p}}[\sum_a q_{\pi_{\boldsymbol{\theta}}}(S_t,a) \nabla \pi_{\boldsymbol{\theta}}(a|S_t)]\end{aligned}$$

All-actions algorithm

Remark

Every instantiation of the stochastic gradient ascent using:

$$\nabla J(\boldsymbol{\theta}) \propto \sum_s \mu(s) \sum_a q_{\pi_{\boldsymbol{\theta}}}(s,a) \nabla \pi_{\boldsymbol{\theta}}(a|s)$$
$$= \mathbb{E}_\mu [\sum_a q_{\pi_{\boldsymbol{\theta}}}(s,a) \nabla \pi_{\boldsymbol{\theta}}(a|s)]$$

gives a **Policy Gradient update rule**, and a corresponding PG algorithm.

Example

The **all-actions** PG update rule is:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha \sum_a \hat{q}(S_t, a, \mathbf{w}) \nabla \pi(a|S_t, \boldsymbol{\theta}).$$

where $\hat{q}$ is any estimate of the q-value function q_π of π.

REINFORCE algorithm

Replace $\sum_a$ as an expected value using random actions A_t:

$$\begin{aligned}
\nabla J(\boldsymbol{\theta}) &= \mathbb{E}_{\tau\sim\mathbb{P}_{\pi_\theta},p}[\sum_a q_{\pi_\theta}(S_t,a)\nabla\pi_{\boldsymbol{\theta}}(a|S_t)] \\
&= \mathbb{E}_{\tau\sim\mathbb{P}_{\pi_\theta},p}\left[\sum_a \pi_{\boldsymbol{\theta}}(a|S_t)q_{\pi_\theta}(S_t,a)\frac{\nabla\pi_{\boldsymbol{\theta}}(a|S_t)}{\pi_{\boldsymbol{\theta}}(a|S_t)}\right] \\
&= \mathbb{E}_{\tau\sim\mathbb{P}_{\pi_\theta},p}\left[q_{\pi_\theta}(S_t,A_t)\frac{\nabla\pi_{\boldsymbol{\theta}}(A_t|S_t)}{\pi_{\boldsymbol{\theta}}(A_t|S_t)}\right] \\
&= \mathbb{E}_{\tau\sim\mathbb{P}_{\pi_\theta},p}\left[G_t\frac{\nabla\pi_{\boldsymbol{\theta}}(A_t|S_t)}{\pi_{\boldsymbol{\theta}}(A_t|S_t)}\right].
\end{aligned}$$

Definition

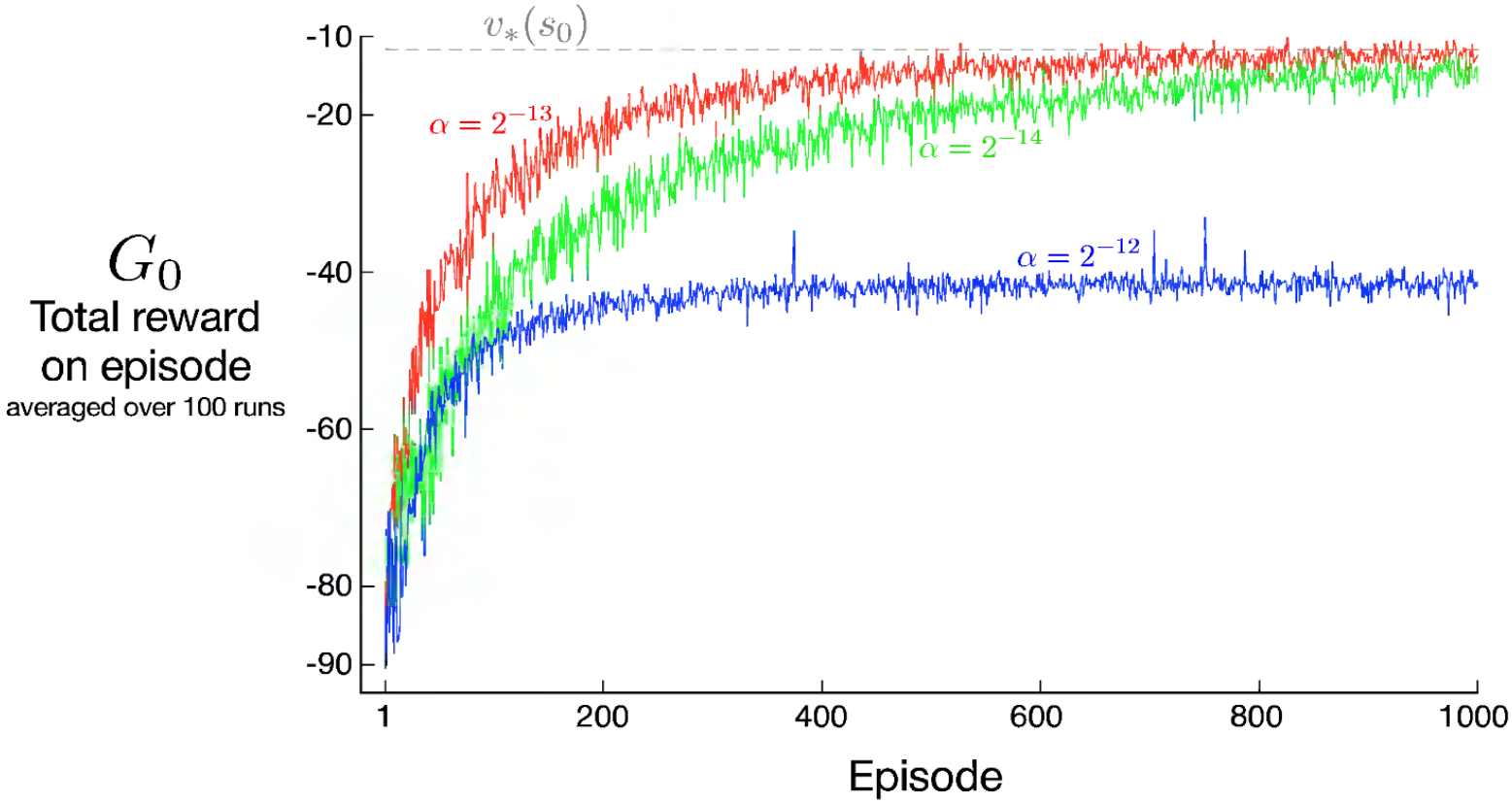
The REINFORCE update rule is:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha G_t\frac{\nabla\pi_{\boldsymbol{\theta}_t}(A_t|S_t)}{\pi_{\boldsymbol{\theta}_t}(A_t|S_t)}.$$

REINFORCE pseudocode

Input: A differentiable policy parametrization $\pi_{\boldsymbol{\theta}}(a|s)$.
Parameter: Step size $\alpha > 0$**.**
Initialize: Initialize $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ **arbitrarily.**
do
  Generate episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$ following $\pi_{\boldsymbol{\theta}}(\cdot|\cdot)$
  for $t = 0, 1, \ldots, T-1$ **do**
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \gamma^t G \nabla \ln \pi_{\boldsymbol{\theta}}(A_t|S_t)$
  end
while True

REINFORCE example



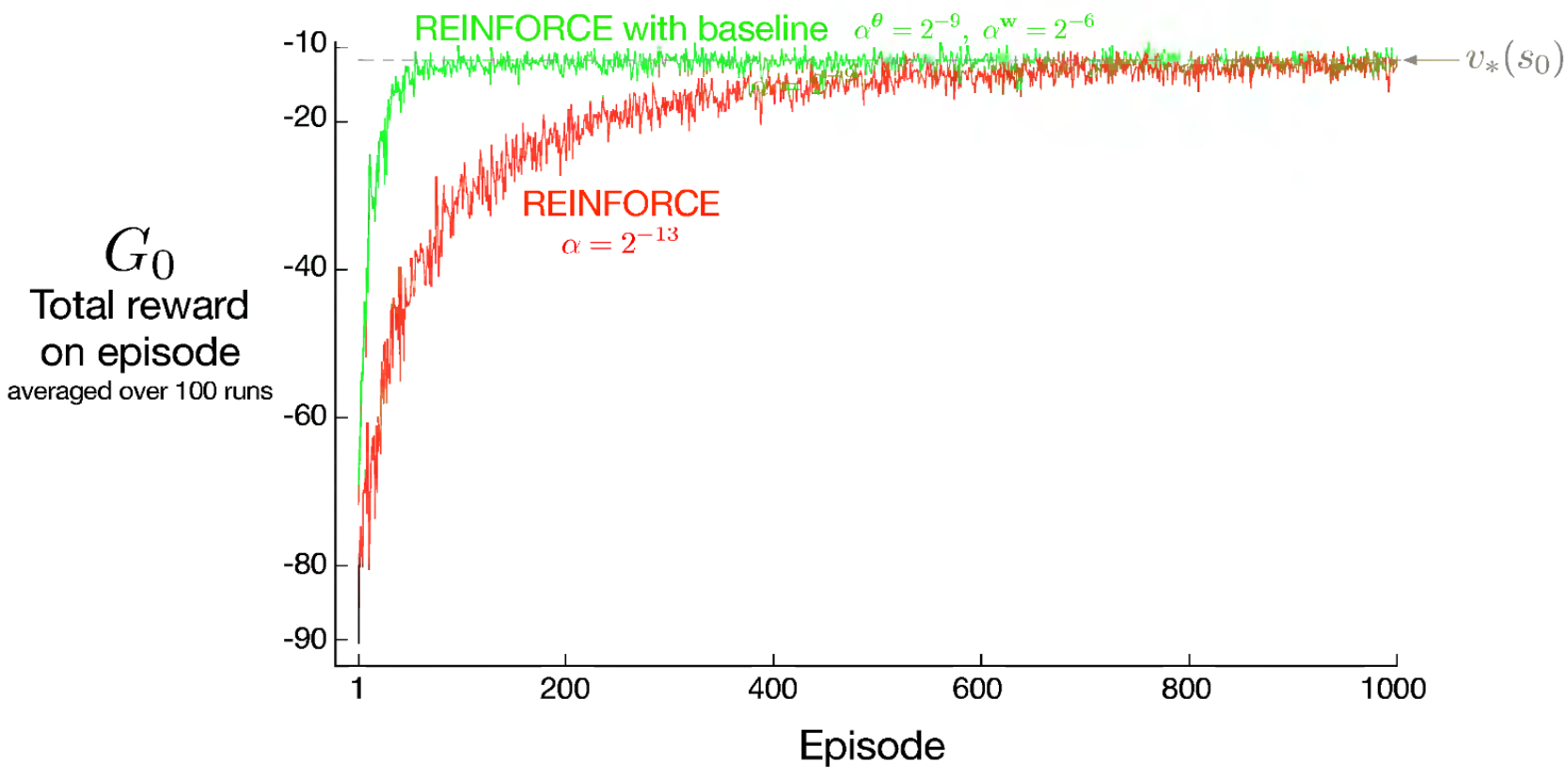
- Stochastic gradient method: good convergence properties.
- Expected update over episode: in the same direction as the performance gradient. Thus, convergence to a local optimum.
- Monte Carlo method: high variance and thus slow learning.
- To limit the variance, we can use a **baseline**.

REINFORCE with baseline

Definition

The REINFORCE with baseline update rule is:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha(G_t - b(s))\frac{\nabla \pi_{\boldsymbol{\theta}_t}(A_t|S_t)}{\pi_{\boldsymbol{\theta}_t}(A_t|S_t)}.$$



Actor-critic methods: reducing variance using a critic

REINFORCE with baseline is not an AC method

The value function in $G_t - \hat{v}(S_t, \mathbf{w})$ is not used for the estimate G_t that will be used as a target for the policy update. It is not a **critic** for the policy.

A true AC method

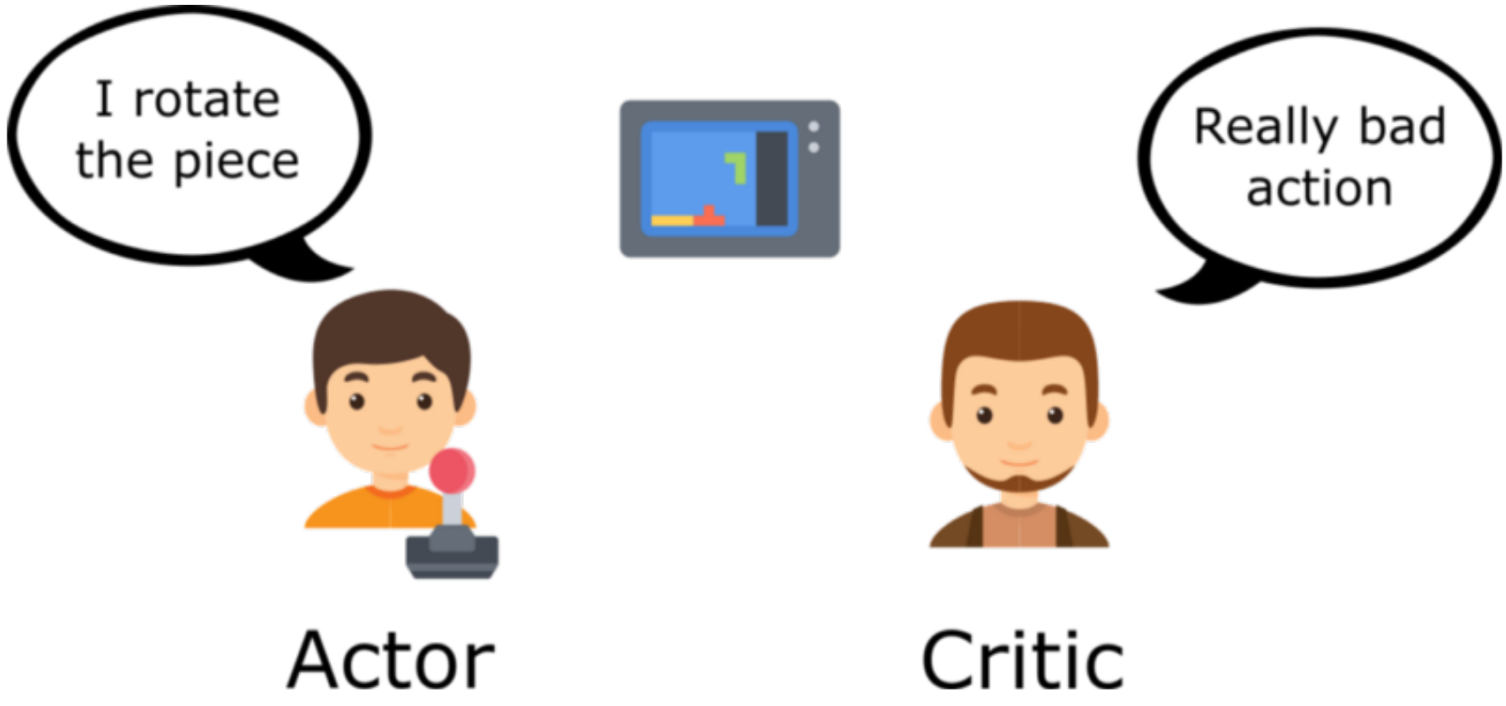
Replace the full return G_t with a one-step return:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha(G_{t:t+1} - \hat{v}(S_t, \mathbf{w}))\frac{\nabla \pi_{\boldsymbol{\theta}_t}(A_t|S_t)}{\pi_{\boldsymbol{\theta}_t}(A_t|S_t)}.$$

Generalization

One-step return can be replaced with n-step or λ return. It works.

Actor-Critic



Summary of policy gradient algorithms

General form of policy gradient

We have several different estimators, sharing a common path:

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim \mathbb{P}_{\pi_\theta, p}} \left[\square \frac{\nabla \pi_{\boldsymbol{\theta}}(A_t|S_t)}{\pi_{\boldsymbol{\theta}}(A_t|S_t)} \right]$$

where $\square$ can be:

- REINFORCE: $\square = G_t$. Unbiased $\Rightarrow$ not actor-critic.
- q-value actor-critic: $\square = q(S_t, A_t, \mathbf{w})$.
- Advantage actor-critic (A2C): $\square = A(S_t, A_t, \mathbf{w})$.
- TD(0) actor-critic: $\square = R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}) - \hat{v}(S_t, \mathbf{w})$.
- TD(n) actor-critic: $\square = G_{t:t+n} - \hat{v}(S_t, \mathbf{w})$.
- TD(λ) actor-critic: $\square = G_t^\lambda - \hat{v}(S_t, \mathbf{w})$.

Policy Gradient based model-free Linear Quadratic Regulator (LQR)

- Infinite horizon LQR problem

$$\textbf{minimize}\, E\left[\sum_{t=0}^{\infty}(x_t^T Q x_t + u_t^T R u_t)\right]$$
$$\textbf{such that}\, x_{t+1} = Ax_t + Bu_t,\ x_0 \sim \mathcal{D}, Q, R > 0$$

- For a known model, it is well known that the optimal control solution is

$$u_t = -K^* x_t, K^* = -(B^T P B + R)^{-1} B^T P A$$
$$\textit{where}\, P = A^T P A + Q - A^T P B (B^T P B + R)^{-1} B^T P A$$

- What about the model-free case (when we do not know A, B ?)
- Many recent studies: we will focus on Fazel *et al*, Global Convergence of Policy Gradient Methods for the LQR, ICML 2018

Policy Gradient based model-free Linear Quadratic Regulator (LQR)

- For a generic control law $u_t = -Kx_t$, define

$$C(K) = E_{x_0 \sim \mathcal{D}} \left[\sum_{t=0}^{\infty} (x_t^T Q x_t + u_t^T R u_t) \right]$$

- Standard Policy Gradient

$$K_{i+1} = K_i - \eta \nabla C(K), \ \nabla C(K) = 2(R + B^T P_K B)K - B^T P_k A)\Sigma_K,$$

$$P_k = Q + K^T R K + (A - BK)^T P_k (A - BK), \ \Sigma_K = E_{x_0 \sim \mathcal{D}} \sum_{t=0}^{\infty} x_t x_t^T$$

- Interestingly, $C(K)$ turns out to be non-convex in K, and yet a globally optimal solution exists and can be found.

Policy Gradient based model-free Linear Quadratic Regulator (LQR)

- A better alternative: Natural Policy Gradient (NPG)
- Assuming a parametrized policy $\pi_\theta(u_t|x_t)$,

$$\theta_{i+1} = \theta_i - \eta G_{\theta_i}^{-1} \nabla C(\theta_i),\ G_\theta = E\left[\sum_{t=0}^{\infty} \nabla \pi_\theta(u_t|x_t) \nabla \pi_\theta(u_t|x_t)^T\right]$$

- For the LQR problem choose $\pi(u_t|x_t) = \mathcal{N}(-Kx, \sigma^2 I)$ (why noisy policy?)
- NPG Update:

$$K_{i+1} = K_i - \eta \nabla C(K_i) {\Sigma_{K_i}}^{-1}$$

- For model free case, one needs to estimate the Fisher information matrix G_K and the gradient of $C(K)$
- For the model based scenario, one can show that the NPG enjoys linear convergence (exponentially decaying error norm $||K_i - K^*||$) for a well-chosen stepsize η
- For the model-free case with estimated gradients from samples, one can establish similar behaviour with high probability under suitably large sample size