

Policy Gradient

Spring, 2021

Outline

- 1 Introduction
- 2 Policy Gradient
- 3 Monte-Carlo Policy Gradient

Table of Contents

1 Introduction

2 Policy Gradient

3 Monte-Carlo Policy Gradient

Policy-Based Reinforcement Learning

- In the last lecture we approximated the value or action-value function using parameters θ ,

$$V_\theta(s) \approx V^\pi(s)$$

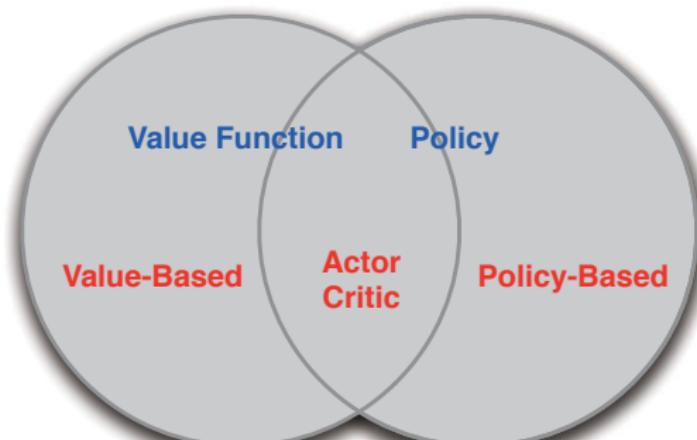
$$Q_\theta(s, a) \approx Q^\pi(s, a)$$

- A policy was generated directly from the value function
 - e.g. using ϵ -greedy
- In this lecture we will directly parametrize the **policy**

$$\pi_\theta(s, a) = \mathbb{P}[a|s, \theta]$$

- We will focus again on **model-free** reinforcement learning

Value-Based and Policy-Based RL

- Value Based
 - Learnt Value Function
 - Implicit policy
(e.g. ϵ -greedy)
 - Policy Based
 - No Value Function
 - Learnt Policy
 - Actor-Critic
 - Learnt Value Function
 - Learnt Policy
- 
- A Venn diagram consisting of two overlapping circles. The left circle is labeled "Value Function" and the right circle is labeled "Policy". The intersection of the two circles is labeled "Actor Critic". The regions outside the intersection but inside each circle are labeled "Value-Based" and "Policy-Based" respectively.

Advantages of Policy-Based RL

Advantages:

- Better convergence properties
- Effective in high-dimensional or continuous action spaces
- Can learn stochastic policies

Disadvantages:

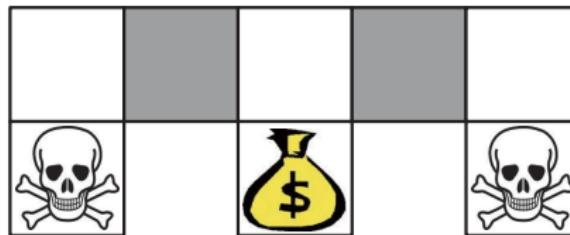
- Typically converge to a local rather than global optimum
- Evaluating a policy is typically inefficient and high variance

Example: Rock-Paper-Scissors



- Two-player game of rock-paper-scissors
 - Scissors beats paper
 - Rock beats scissors
 - Paper beats rock
- Consider policies for *iterated* rock-paper-scissors
 - A deterministic policy is easily exploited
 - A uniform random policy is optimal
(i.e. Nash equilibrium)

Example: Aliased Gridworld (1)



- The agent cannot differentiate the grey states
- Consider features of the following form (for all N, E, S, W)

$$\phi(s, a) = \mathbf{1}(\text{wall to N}, a = \text{move E})$$

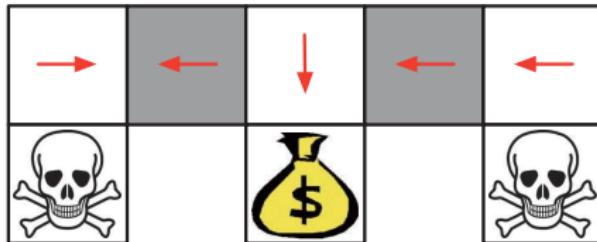
- Compare value-based RL, using an approximate value function

$$Q_\theta(s, a) = f(\phi(s, a), \theta)$$

- To policy-based RL, using a parametrized policy

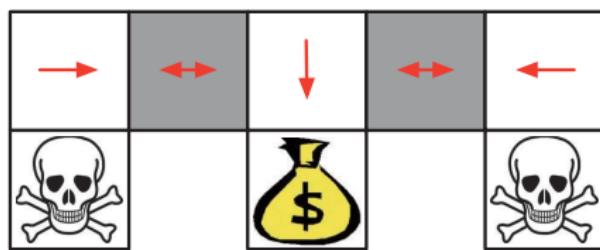
$$\pi_\theta(s, a) = g(\phi(s, a), \theta)$$

Example: Aliased Gridworld (2)



- Under aliasing, an optimal **deterministic** policy will either
 - move W in both grey states (shown by red arrows)
 - move E in both grey states
- Either way, it can get stuck and **never** reach the money
- Value-based RL learns a near-deterministic policy
 - e.g. greedy or ϵ -greedy
- So it will traverse the corridor for a long time

Example: Aliased Gridworld (3)



- An optimal stochastic policy will randomly move E or W in grey states

$$\pi_{\theta}(\text{wall to N and S, move to E}) = 0.5$$

$$\pi_{\theta}(\text{wall to N and S, move to W}) = 0.5$$

- It will reach the goal state in a few steps with high probability
- Policy-based RL can learn the optimal stochastic policy

Table of Contents

1 Introduction

2 Policy Gradient

3 Monte-Carlo Policy Gradient

Policy Gradient

Basic Components

	Actor	Env	Reward Function
Video Game			Get 20 scores when killing a monster
Go			The rule of GO

Policy Gradient

Basic Components

You cannot control				
	Actor	Env	Reward Function	
Video Game				Get 20 scores when killing a monster
Go			The rule of GO	

Policy Gradient

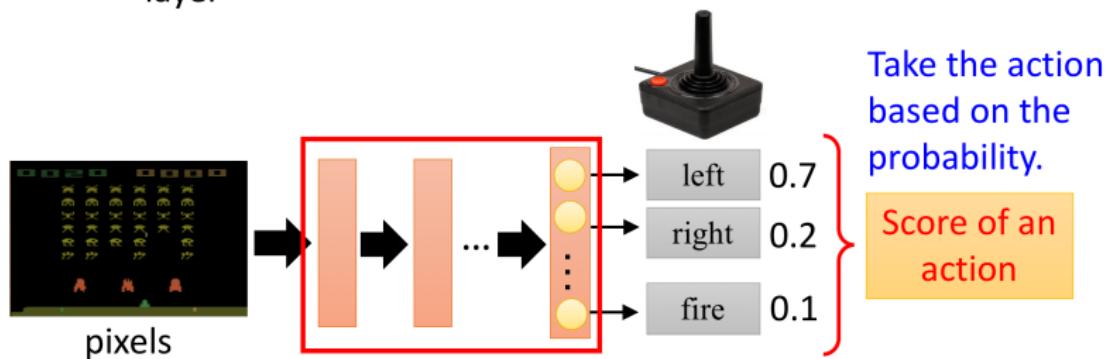
Policy of Actor

- Policy π is a network with parameter θ
 - Input: the observation of machine represented as a vector or a matrix
 - Output: each action corresponds to a neuron in output layer

Policy Gradient

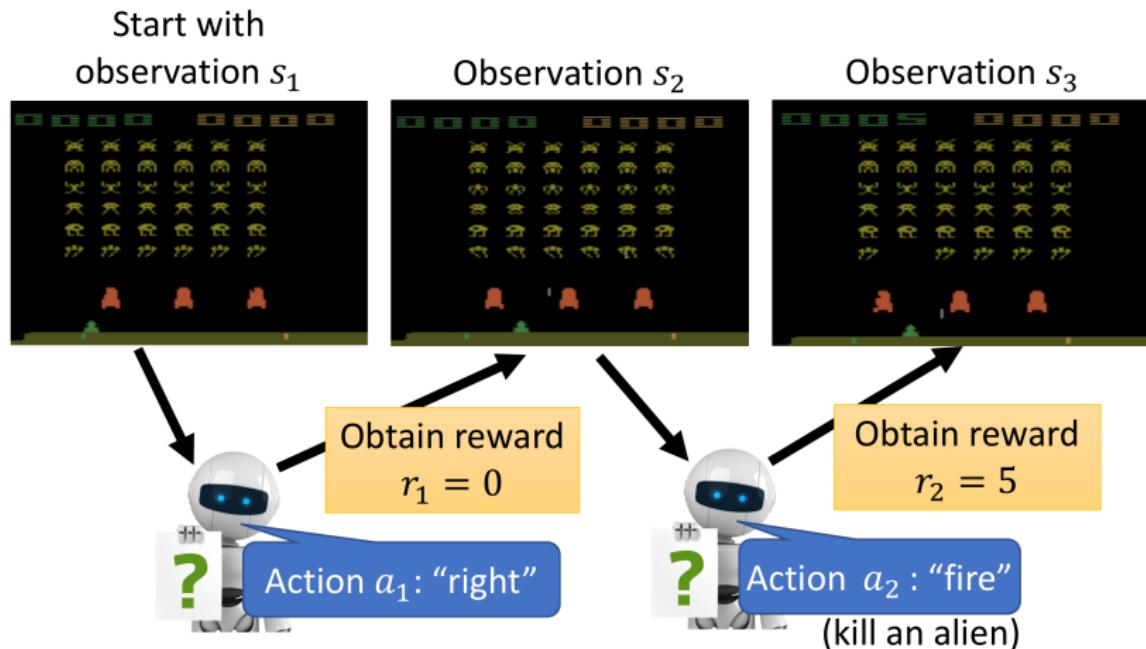
Policy of Actor

- Policy π is a network with parameter θ
 - Input: the observation of machine represented as a vector or a matrix
 - Output: each action corresponds to a neuron in output layer



Policy Gradient

Example: Playing Video Game



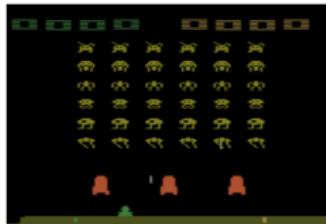
Policy Gradient

Example: Playing Video Game

Start with
observation s_1



Observation s_2



Observation s_3



After many turns



Obtain reward r_T

Action a_T

This is an episode.

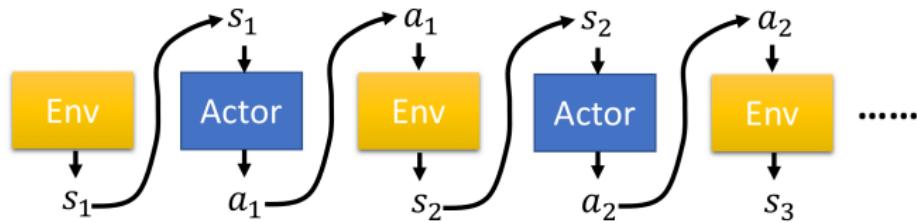
Total reward:

$$R = \sum_{t=1}^T r_t$$

We want the total
reward be maximized.

Policy Gradient

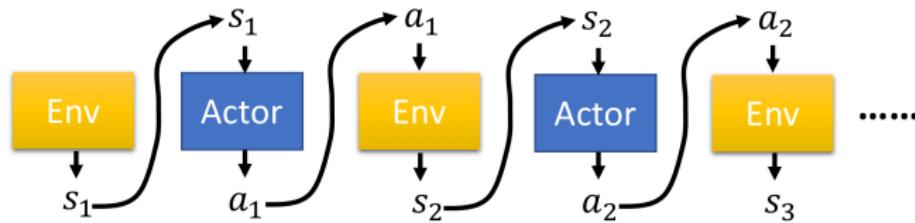
Actor, Environment, Reward



Trajectory $\tau = \{s_1, a_1, s_2, a_2, \dots, s_T, a_T\}$

Policy Gradient

Actor, Environment, Reward



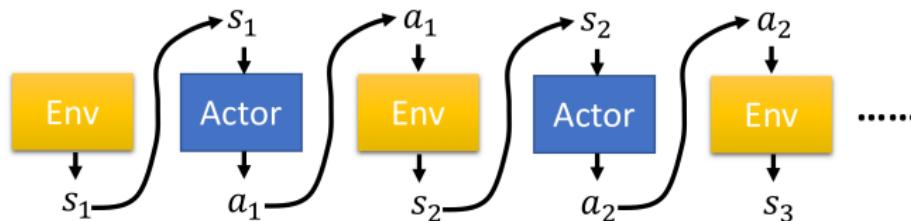
Trajectory $\tau = \{s_1, a_1, s_2, a_2, \dots, s_T, a_T\}$

$$p_{\theta}(\tau)$$

$$= p(s_1)p_{\theta}(a_1|s_1)p(s_2|s_1, a_1)p_{\theta}(a_2|s_2)p(s_3|s_2, a_2)\dots$$

Policy Gradient

Actor, Environment, Reward



Trajectory $\tau = \{s_1, a_1, s_2, a_2, \dots, s_T, a_T\}$

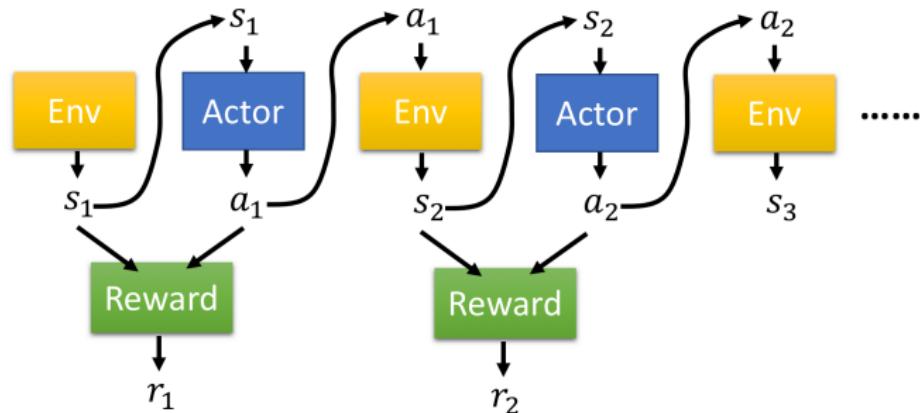
$$p_\theta(\tau)$$

$$= p(s_1)p_{\theta}(a_1|s_1)p(s_2|s_1, a_1)p_{\theta}(a_2|s_2)p(s_3|s_2, a_2) \dots$$

$$= p(s_1) \prod_{t=1}^T p_\theta(a_t|s_t)p(s_{t+1}|s_t, a_t)$$

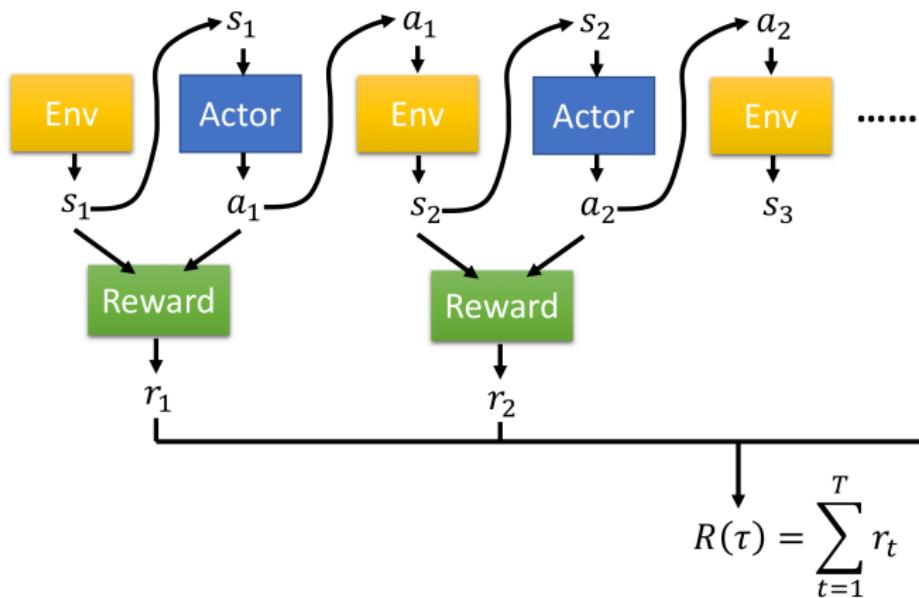
Policy Gradient

Actor, Environment, Reward



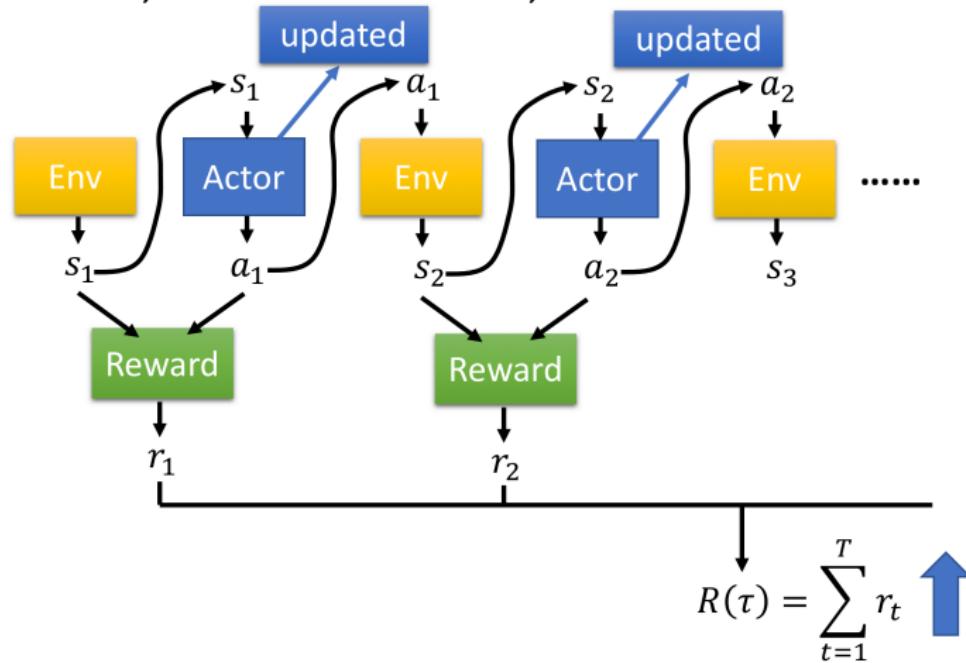
Policy Gradient

Actor, Environment, Reward



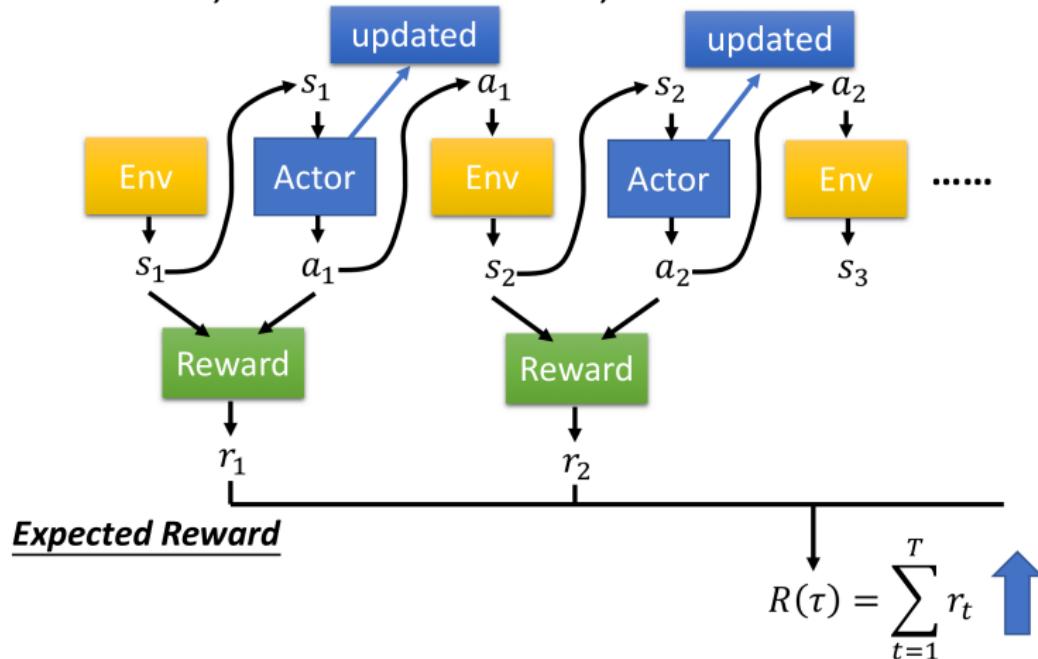
Policy Gradient

Actor, Environment, Reward



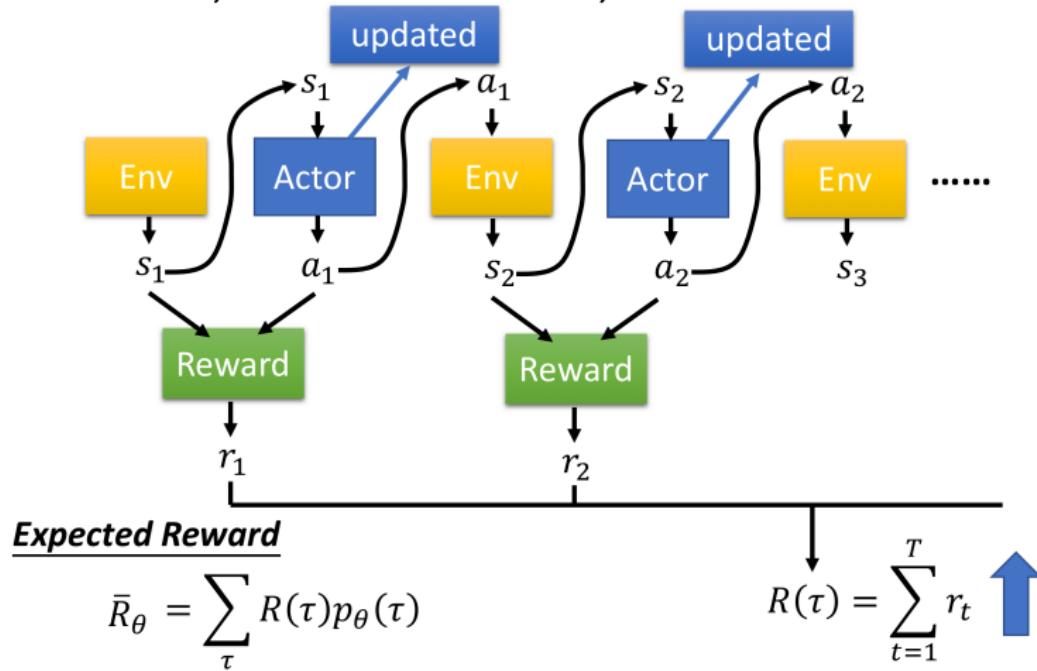
Policy Gradient

Actor, Environment, Reward



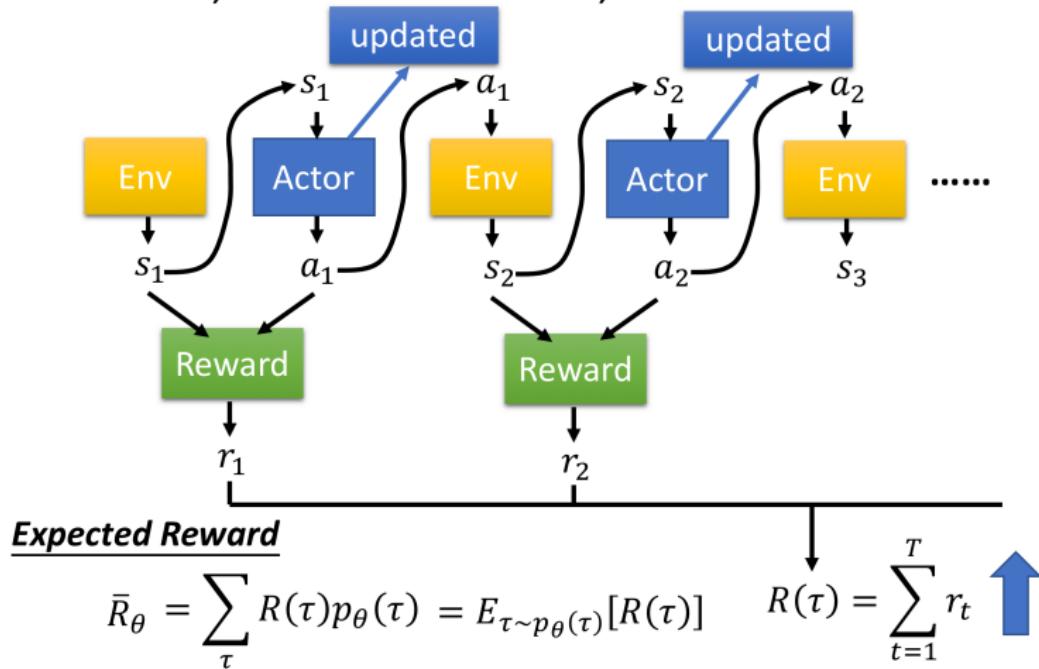
Policy Gradient

Actor, Environment, Reward



Policy Gradient

Actor, Environment, Reward



Policy Gradient

Policy Gradient $\bar{R}_\theta = \sum_\tau R(\tau)p_\theta(\tau)$ $\nabla \bar{R}_\theta = ?$

Policy Gradient

Policy Gradient $\bar{R}_\theta = \sum_\tau R(\tau)p_\theta(\tau)$ $\nabla \bar{R}_\theta = ?$

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau) \nabla p_\theta(\tau)$$

$R(\tau)$ do not have to be differentiable

Policy Gradient

Policy Gradient $\bar{R}_\theta = \sum_\tau R(\tau)p_\theta(\tau)$ $\nabla \bar{R}_\theta = ?$

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau) \nabla p_\theta(\tau)$$

$R(\tau)$ do not have to be differentiable
It can even be a black box.

Policy Gradient

Policy Gradient $\bar{R}_\theta = \sum_\tau R(\tau)p_\theta(\tau)$ $\nabla \bar{R}_\theta = ?$

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau) \nabla p_\theta(\tau) = \sum_\tau R(\tau) p_\theta(\tau) \frac{\nabla p_\theta(\tau)}{p_\theta(\tau)}$$

$R(\tau)$ do not have to be differentiable
It can even be a black box.

Policy Gradient

Policy Gradient $\bar{R}_\theta = \sum_\tau R(\tau)p_\theta(\tau)$ $\nabla \bar{R}_\theta = ?$

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau) \nabla p_\theta(\tau) = \sum_\tau R(\tau) p_\theta(\tau) \frac{\nabla p_\theta(\tau)}{p_\theta(\tau)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \sum_\tau R(\tau) p_\theta(\tau) \nabla \log p_\theta(\tau)$$

$$\begin{aligned} \nabla f(x) &= \\ f(x) \nabla \log f(x) & \end{aligned}$$

Policy Gradient

Policy Gradient $\bar{R}_\theta = \sum_\tau R(\tau)p_\theta(\tau)$ $\nabla \bar{R}_\theta = ?$

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau) \nabla p_\theta(\tau) = \sum_\tau R(\tau) p_\theta(\tau) \frac{\nabla p_\theta(\tau)}{p_\theta(\tau)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \boxed{\sum_\tau} R(\tau) p_\theta(\tau) \boxed{\nabla \log p_\theta(\tau)}$$

$$= E_{\tau \sim p_\theta(\tau)} [R(\tau) \nabla \log p_\theta(\tau)]$$

$$\nabla f(x) = f(x) \nabla \log f(x)$$

Policy Gradient

Policy Gradient $\bar{R}_\theta = \sum_\tau R(\tau)p_\theta(\tau)$ $\nabla \bar{R}_\theta = ?$

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau) \nabla p_\theta(\tau) = \sum_\tau R(\tau) p_\theta(\tau) \frac{\nabla p_\theta(\tau)}{p_\theta(\tau)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \boxed{\sum_\tau} R(\tau) p_\theta(\tau) \boxed{\nabla \log p_\theta(\tau)}$$

$$\nabla f(x) = f(x) \nabla \log f(x)$$

$$= E_{\tau \sim p_\theta(\tau)} [R(\tau) \nabla \log p_\theta(\tau)] \approx \frac{1}{N} \sum_{n=1}^N R(\tau^n) \nabla \log p_\theta(\tau^n)$$

Policy Gradient

Policy Gradient $\bar{R}_\theta = \sum_\tau R(\tau)p_\theta(\tau)$ $\nabla \bar{R}_\theta = ?$

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau) \nabla p_\theta(\tau) = \sum_\tau R(\tau) p_\theta(\tau) \frac{\nabla p_\theta(\tau)}{p_\theta(\tau)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \boxed{\sum_\tau} R(\tau) p_\theta(\tau) \boxed{\nabla \log p_\theta(\tau)}$$

$$\nabla f(x) = f(x) \nabla \log f(x)$$

$$= E_{\tau \sim p_\theta(\tau)} [R(\tau) \nabla \log p_\theta(\tau)] \approx \frac{1}{N} \sum_{n=1}^N R(\tau^n) \nabla \log p_\theta(\tau^n)$$

$$= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$



Policy Gradient

Policy Gradient

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\begin{aligned}\nabla \bar{R}_\theta = \\ \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)\end{aligned}$$

Policy Gradient

Policy Gradient

Given policy π_θ

$$\tau^1: (s_1^1, a_1^1) \quad R(\tau^1)$$

$$(s_2^1, a_2^1) \quad R(\tau^1)$$

⋮

⋮

$$\tau^2: (s_1^2, a_1^2) \quad R(\tau^2)$$

$$(s_2^2, a_2^2) \quad R(\tau^2)$$

⋮

⋮

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta =$$

$$\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$

Policy Gradient

Policy Gradient

Given policy π_θ

$$\tau^1: (s_1^1, a_1^1) \quad R(\tau^1)$$

$$(s_2^1, a_2^1) \quad R(\tau^1)$$

⋮

⋮

$$\tau^2: (s_1^2, a_1^2) \quad R(\tau^2)$$

$$(s_2^2, a_2^2) \quad R(\tau^2)$$

⋮

⋮

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

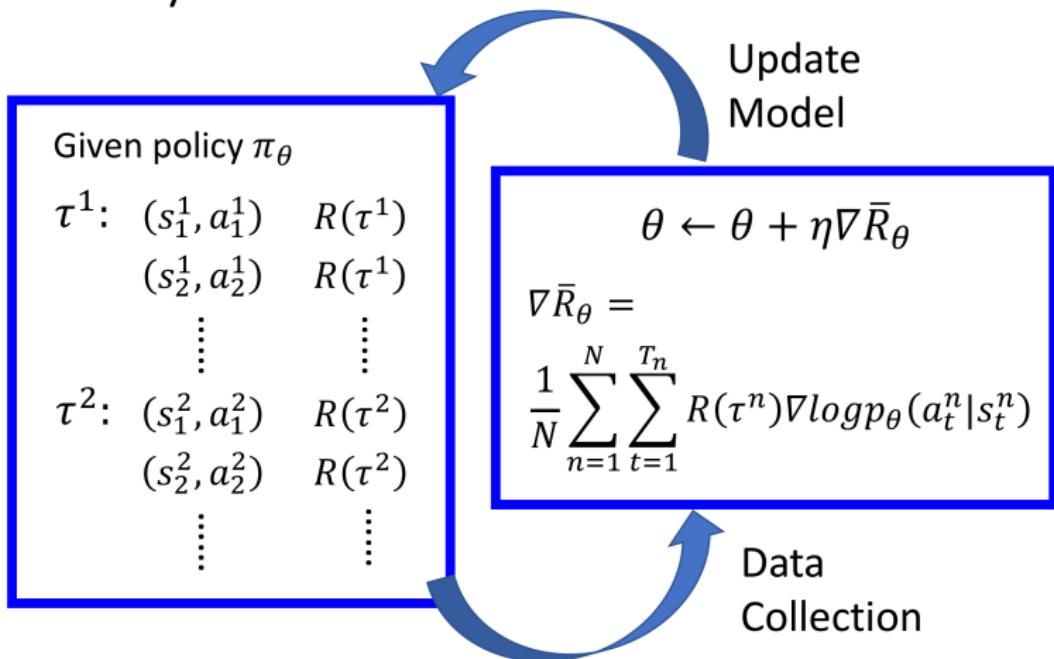
$$\nabla \bar{R}_\theta =$$

$$\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$

Data
Collection

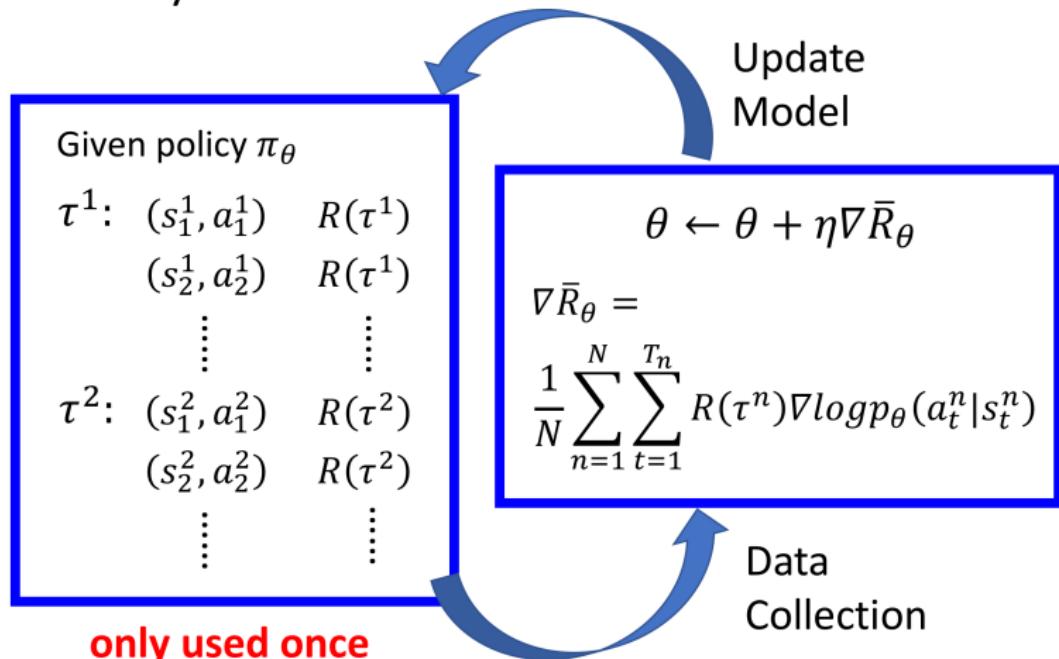
Policy Gradient

Policy Gradient



Policy Gradient

Policy Gradient



Monte-Carlo Policy Gradient (REINFORCE)

- Update parameters by stochastic gradient ascent
- Using policy gradient theorem
- Using return v_t as an unbiased sample of $Q^{\pi_\theta}(s_t, a_t)$

$$\Delta\theta_t = \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$$

function REINFORCE

 Initialise θ arbitrarily

for each episode $\{s_1, a_1, r_2, \dots, s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ **do**

for $t = 1$ to $T - 1$ **do**

$\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$

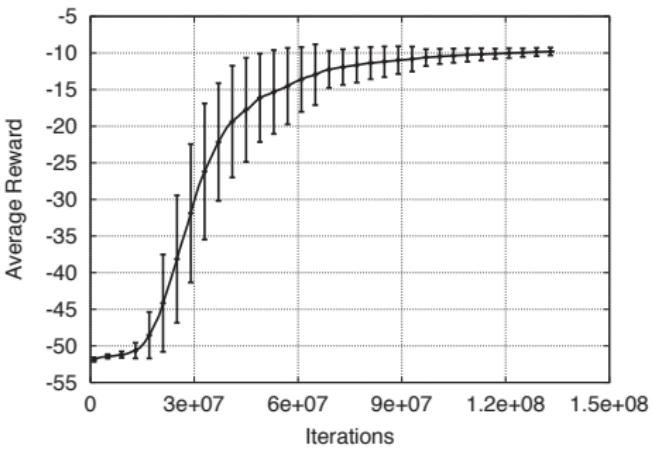
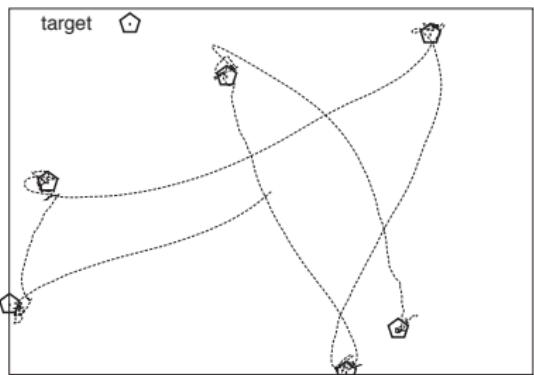
end for

end for

return θ

end function

Puck World Example



- Continuous actions exert small force on puck
- Puck is rewarded for getting close to target
- Target location is reset every 30 seconds
- Policy is training using variant of Monte-Carlo policy gradient

Policy Gradient

Tip 1: Add a Baseline

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$

Policy Gradient

Tip 1: Add a Baseline

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

It is possible that $R(\tau^n)$ is always positive.

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$

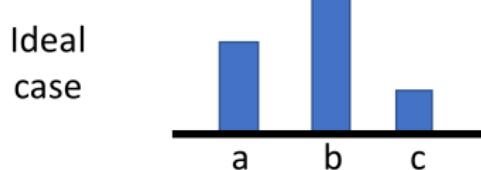
Policy Gradient

Tip 1: Add a Baseline

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

It is possible that $R(\tau^n)$ is always positive.

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$



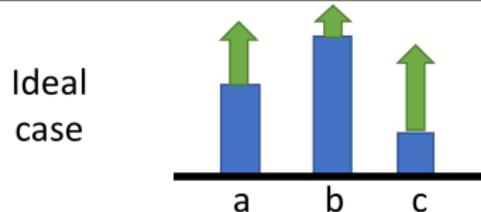
Policy Gradient

Tip 1: Add a Baseline

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

It is possible that $R(\tau^n)$ is always positive.

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$



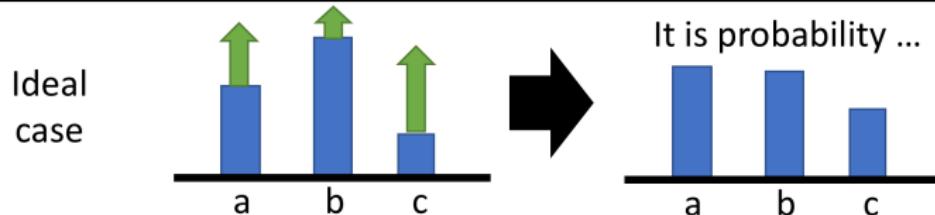
Policy Gradient

Tip 1: Add a Baseline

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

It is possible that $R(\tau^n)$ is always positive.

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$



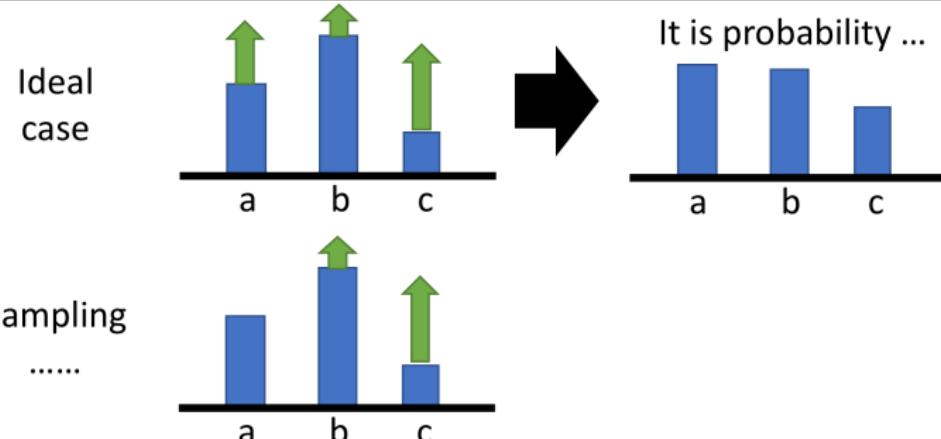
Policy Gradient

Tip 1: Add a Baseline

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

It is possible that $R(\tau^n)$ is always positive.

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$



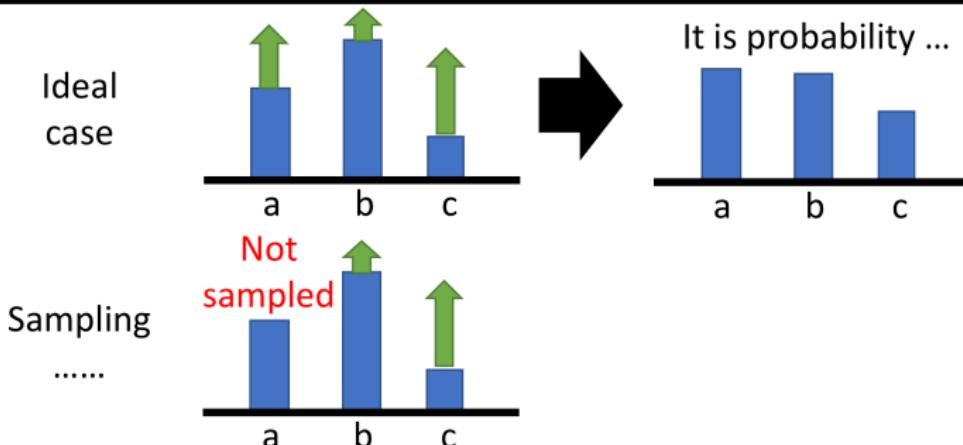
Policy Gradient

Tip 1: Add a Baseline

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

It is possible that $R(\tau^n)$ is always positive.

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$



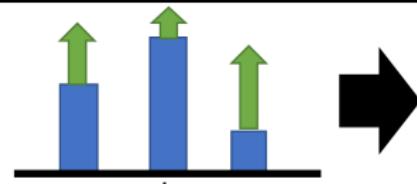
Policy Gradient

Tip 1: Add a Baseline

$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$ It is possible that $R(\tau^n)$ is always positive.

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$

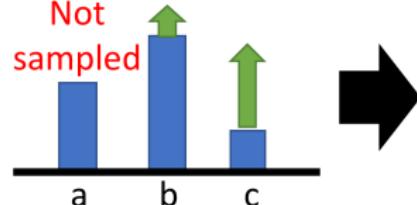
Ideal
case



It is probability ...

Sampling

• • • •



a b c

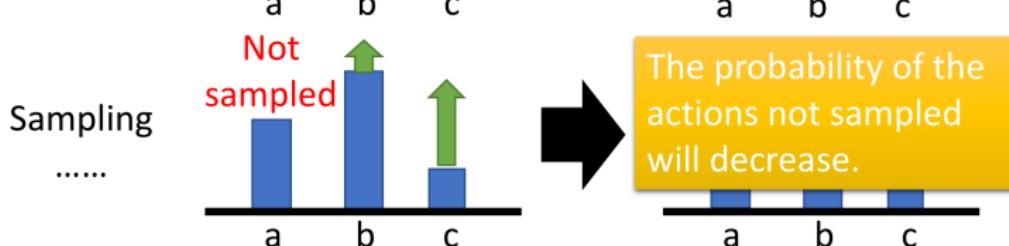
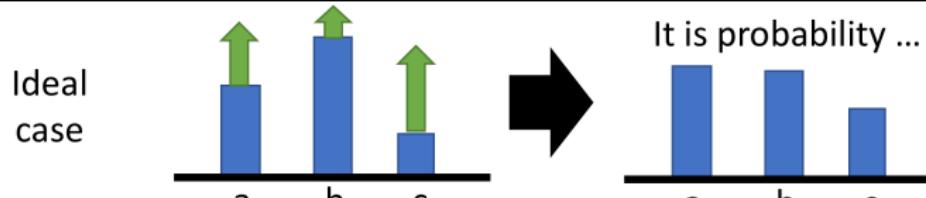
Policy Gradient

Tip 1: Add a Baseline

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta$$

It is possible that $R(\tau^n)$ is always positive.

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n)$$



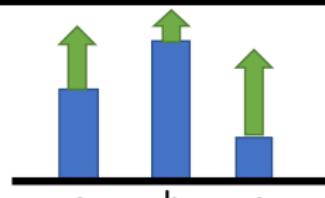
Policy Gradient

Tip 1: Add a Baseline

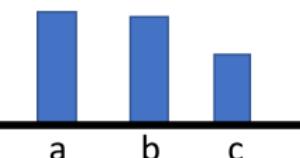
$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta \quad \boxed{\text{It is possible that } R(\tau^n) \text{ is always positive.}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (R(\tau^n) - b) \nabla \log p_\theta(a_t^n | s_t^n) \quad b \approx E[R(\tau)]$$

Ideal case

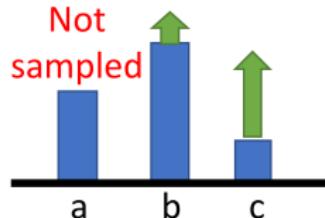


It is probability ...

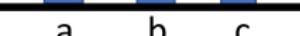


Sampling

.....



The probability of the actions not sampled will decrease.



Policy Gradient

Tip 2: Assign Suitable Credit

$$\begin{array}{cccccc}
 \times 3 & \times 3 & \times 3 & \times -7 & \times -7 & \times -7 \\
 (s_a, a_1) & (s_b, a_2) & (s_c, a_3) & (s_a, a_2) & (s_b, a_2) & (s_c, a_3) \\
 +5 & +0 & -2 & -5 & +0 & -2 \\
 R = +3 & & & R = -7 & &
 \end{array}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (R(\tau^n) - b) \nabla \log p_\theta(a_t^n | s_t^n)$$

Policy Gradient

Tip 2: Assign Suitable Credit

$\times 3$	$\times -2$	$\times -2$	$\times -7$	$\times -2$	$\times -2$
(s_a, a_1)	(s_b, a_2)	(s_c, a_3)	(s_a, a_2)	(s_b, a_2)	(s_c, a_3)
+5	+0	-2	-5	+0	-2
$R = +3$			$R = -7$		

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (R(\tau^n) - b) \nabla \log p_\theta(a_t^n | s_t^n)$$

Policy Gradient

Tip 2: Assign Suitable Credit

$\times 3$	$\times -2$	$\times -2$	$\times -7$	$\times -2$	$\times -2$
(s_a, a_1)	(s_b, a_2)	(s_c, a_3)	(s_a, a_2)	(s_b, a_2)	(s_c, a_3)
+5	+0	-2	-5	+0	-2

$R = +3$ $R = -7$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (\cancel{R(\cdot)} - b) \nabla \log p_\theta(a_t^n | s_t^n)$$


$$\sum_{t'=t}^{T_n} r_{t'}^n$$

Policy Gradient

Tip 2: Assign Suitable Credit

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (\cancel{R(\tau^n)} - b) \nabla \log p_\theta(a_t^n | s_t^n)$$


$$\sum_{t'=t}^{T_n} r_{t'}^n$$

Add discount factor

Policy Gradient

Tip 2: Assign Suitable Credit

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (R(\epsilon^n) - b) \nabla \log p_\theta(a_t^n | s_t^n)$$
$$\sum_{t'=t}^{T_n} r_{t'}^n \rightarrow \sum_{t'=t}^{T_n} \gamma^{t'-t} r_{t'}^n$$

Add discount factor $\gamma < 1$

Policy Gradient

Tip 2: Assign Suitable Credit

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (\cancel{R(\epsilon^n)} - b) \nabla \log p_\theta(a_t^n | s_t^n)$$

Can be state-dependent

$$\sum_{t'=t}^{T_n} r_{t'}^n \xrightarrow{\text{Add discount factor}} \sum_{t'=t}^{T_n} \gamma^{t'-t} r_{t'}^n$$

$\gamma < 1$

Policy Gradient

Tip 2: Assign Suitable Credit

Advantage Function $A^\theta(s_t, a_t)$

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (R(\tau^n) - b) \nabla \log p_\theta(a_t^n | s_t^n)$$

Can be state-dependent

Add discount factor $\gamma < 1$

$$\sum_{t'=t}^{T_n} r_{t'}^n \rightarrow \sum_{t'=t}^{T_n} \gamma^{t'-t} r_{t'}^n$$

Policy Gradient

Tip 2: Assign Suitable Credit

Advantage Function

$$A^\theta(s_t, a_t)$$

How good it is if we take a_t other than other actions at s_t .
Estimated by “critic” (later)

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (R(\epsilon^n) - b) \nabla \log p_\theta(a_t^n | s_t^n)$$

Can be state-dependent

$\sum_{t'=t}^{T_n} r_{t'}^n \rightarrow \sum_{t'=t}^{T_n} \gamma^{t'-t} r_{t'}^n$

Add discount factor $\gamma < 1$

Table of Contents

- 1 Introduction
- 2 Policy Gradient
- 3 Monte-Carlo Policy Gradient

Policy Objective Functions

- Goal: given policy $\pi_\theta(s, a)$ with parameters θ , find best θ
- But how do we measure the quality of a policy π_θ ?
- In episodic environment we can use the **start value**

$$J_1(\theta) = V^{\pi_\theta}(s_1) = \mathbb{E}[v_1]$$

- In continuing environment we can use the **average value**

$$J_{avV}(\theta) = \sum_s d^{\pi_\theta}(s) V^{\pi_\theta}(s)$$

- Or the **average reward per time-step**

$$J_{avR}(\theta) = \sum_s d^{\pi_\theta}(s) \sum_a \pi_\theta(s, a) \mathcal{R}_s^a$$

where $d^{\pi_\theta}(s)$ is **stationary distribution** of
Markov chain for π_θ

Policy Optimization

- Policy based reinforcement learning is an **optimization** problem
- Find θ that maximizes $J(\theta)$
- Some approaches do not use gradient
 - Hill climbing
 - Simplex / amoeba / Nelder Mead
 - Genetic algorithms
- Greater efficiency often possible using gradient
 - Gradient descent
 - Conjugate gradient
 - Quasi-newton
- We focus on gradient descent, many extensions possible
- And on methods that exploit sequential structure

Score Function

- We now compute the policy gradient *analytically*
- Assume policy π_θ is differentiable whenever it is non-zero
- and we know the gradient $\nabla_\theta \pi_\theta(s, a)$
- Likelihood ratios exploit the following identity

$$\begin{aligned}\nabla_\theta \pi_\theta(s, a) &= \pi_\theta(s, a) \frac{\nabla_\theta \pi_\theta(s, a)}{\pi_\theta(s, a)} \\ &= \pi_\theta(s, a) \nabla_\theta \log \pi_\theta(s, a)\end{aligned}$$

- The score function is $\nabla_\theta \log \pi_\theta(s, a)$

Softmax Policy

- We will use a softmax policy as a running example
- Weight actions using linear combination of features $\phi(s, a)^T \theta$
- Probability of action is proportional to exponentiated weight

$$\pi_\theta(s, a) \propto e^{\phi(s, a)^T \theta}$$

- The score function is

$$\nabla_\theta \log \pi_\theta(s, a) = \phi(s, a) - \mathbb{E}_{\pi_\theta}[\phi(s, \cdot)]$$

Gaussian Policy

- In continuous action spaces, a Gaussian policy is natural
- Mean is a linear combination of state features $\mu(s) = \phi(s)^T \theta$
- Variance may be fixed σ^2 , or can also be parametrized
- Policy is Gaussian, $a \sim \mathcal{N}(\mu(s), \sigma^2)$
- The score function is

$$\nabla_{\theta} \log \pi_{\theta}(s, a) = \frac{(a - \mu(s))\phi(s)}{\sigma^2}$$

One-Step MDPs

- Consider a simple class of **one-step** MDPs
 - Starting in state $s \sim d(s)$
 - Terminating after one time-step with reward $r = \mathcal{R}_{s,a}$
- Use likelihood ratios to compute the policy gradient

$$\begin{aligned} J(\theta) &= \mathbb{E}_{\pi_\theta}[r] \\ &= \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(s, a) \mathcal{R}_{s,a} \end{aligned}$$

$$\begin{aligned} \nabla_\theta J(\theta) &= \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(s, a) \nabla_\theta \log \pi_\theta(s, a) \mathcal{R}_{s,a} \\ &= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s, a) r] \end{aligned}$$

Policy Gradient Theorem

- The policy gradient theorem generalizes the likelihood ratio approach to multi-step MDPs
- Replaces instantaneous reward r with long-term value $Q^\pi(s, a)$
- Policy gradient theorem applies to start state objective, average reward and average value objective

Theorem

For any differentiable policy $\pi_\theta(s, a)$, for any of the policy objective functions $J = J_1, J_{avR}$, or $\frac{1}{1-\gamma}J_{avV}$, the policy gradient is

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

Monte-Carlo Policy Gradient (REINFORCE)

- Update parameters by stochastic gradient ascent
- Using policy gradient theorem
- Using return v_t as an unbiased sample of $Q^{\pi_\theta}(s_t, a_t)$

$$\Delta\theta_t = \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$$

function REINFORCE

 Initialise θ arbitrarily

for each episode $\{s_1, a_1, r_2, \dots, s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ **do**

for $t = 1$ to $T - 1$ **do**

$\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$

end for

end for

return θ

end function