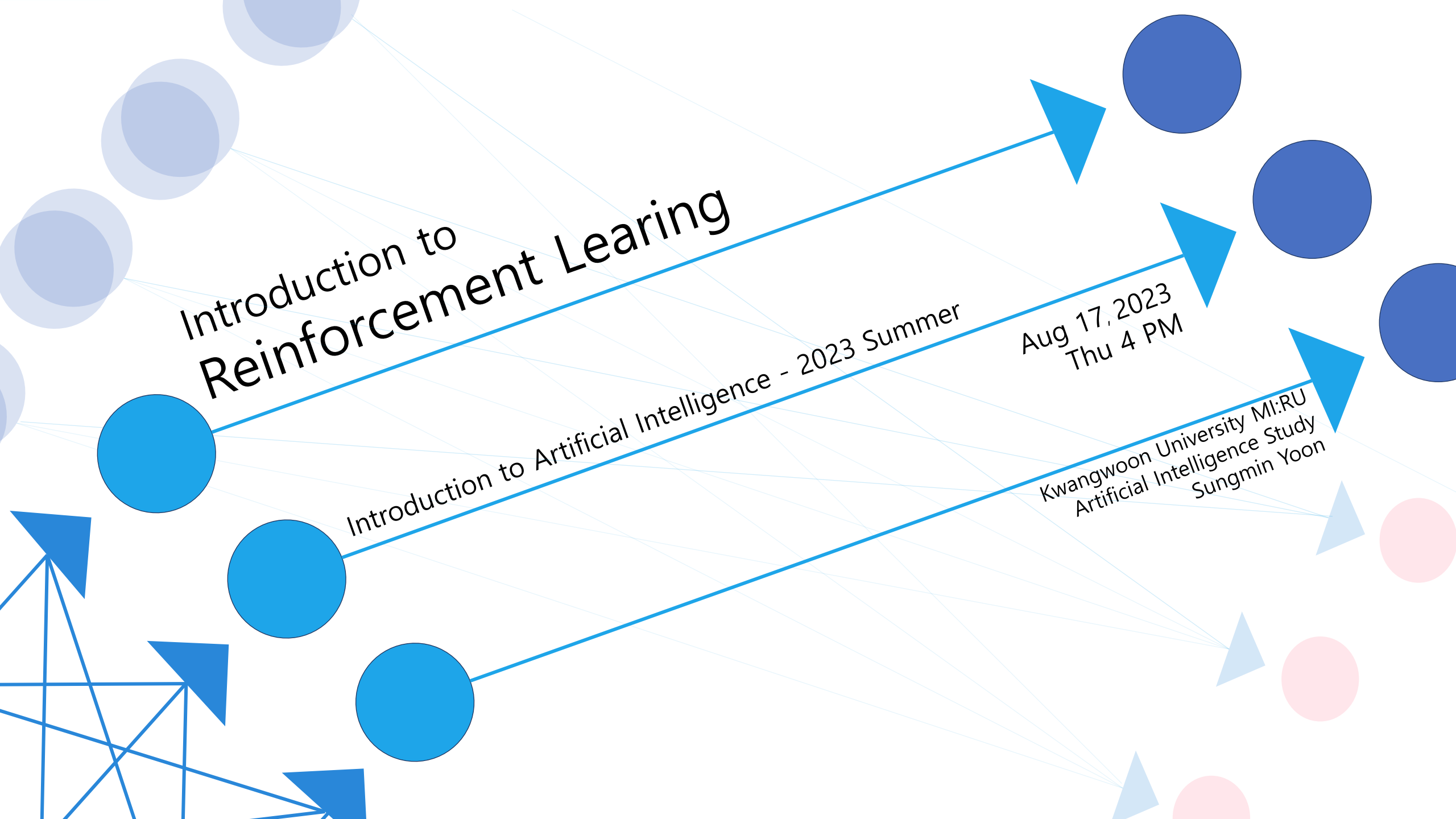

Introduction to
Reinforcement Learing
Introduction to Artificial Intelligence - 2023 Summer
Aug 17, 2023
Thu 4 PM
Kwangwoon University MI:RU
Artificial Intelligence Study
Sungmin Yoon

Intro

In this course, you will learn

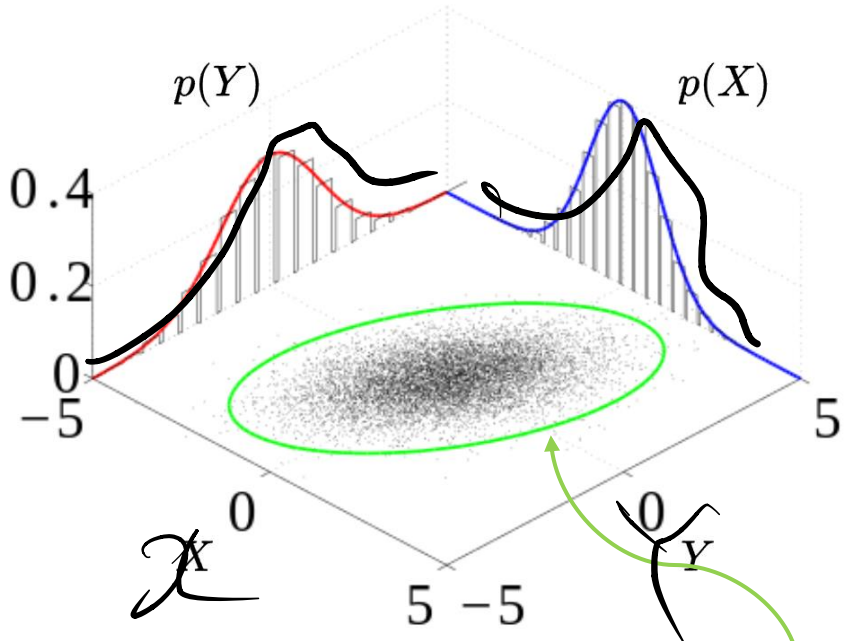
Part 1 – Quick Review of Probability Theorem
- Joint Probability
  - Joint probability distribution & Marginalization
- Expected Value as Integral
- Chain Rule

Part 2 –Reinforcement Learning
- Basic concepts & Terms
- Markov Property
- Bellman Equation

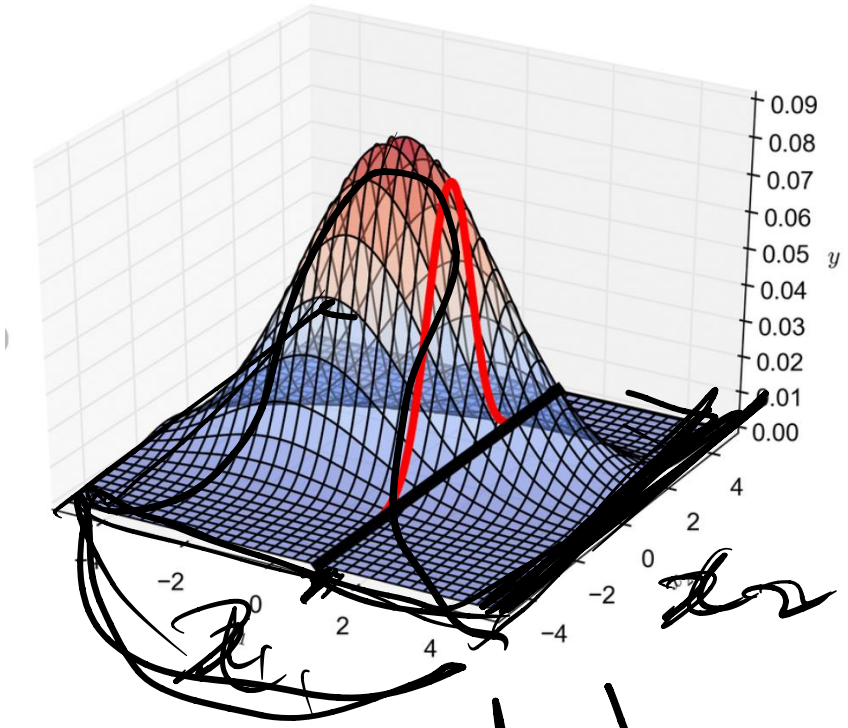
Part 3 – Policy based RL
- Calculation of Policy
  - Probability of Trajectory
  - Log-derivative trick
  - Expected gradient-log probability Lemma
- Advance of algorithms
  - (REINFORCE, A2C, A3C, PPO)

Probability Theorem

Joint Probability



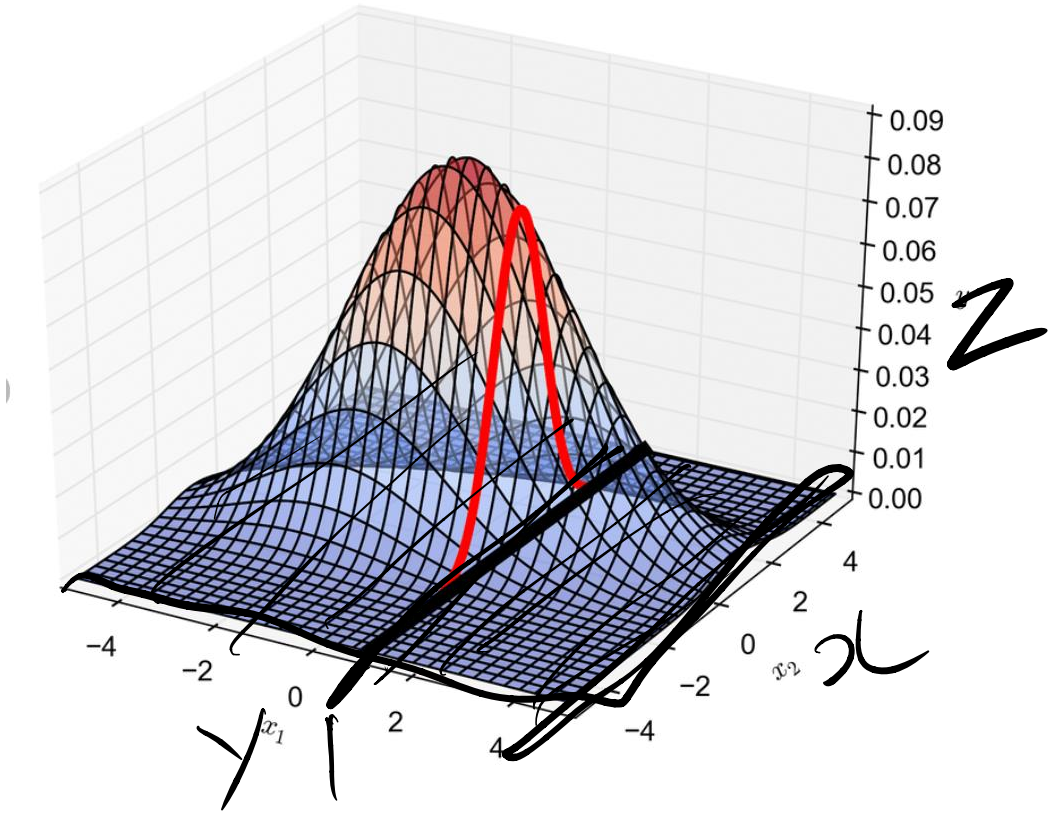
$$\int_x \int_y f_{X,Y}(x,y)\, dy\, dx = 1$$



$$\int_{x_1}\int_{x_2} p(x_1,x_2)\, dx_2\, dx_1$$

Probability Theorem

Marginalization

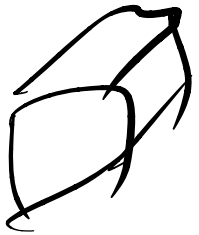


$$P(X) = \int_y P(X, Y = y) \mathrm{d}y$$

Probability Theorem

Expected Value as Integral

$$\int_x p(x)\,dx = 1$$



$$X = [1, 2, 3, 4, 5, 6]$$

$$p(x) = \frac{1}{6} \text{ for each } X$$

$$\int_x x\,p(x) = E[X]$$

$$\mathrm{E}[X] = \int_{-\infty}^{\infty} x f(x)\,dx.$$

$$E[X] = \int x\,p(x)\,dx$$

$$\int p(x) = 3.5 \quad X$$

$$\frac{1}{6}(1+2+3+4+5+6)$$

Probability Theorem

Chain Rule

Chain rule — Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space. Let $A_1, \ldots, A_n \in \mathcal{A}$. Then

$$\mathbb{P}(A_1 \cap A_2 \cap \ldots \cap A_n) = \mathbb{P}(A_1)\mathbb{P}(A_2 \mid A_1)\mathbb{P}(A_3 \mid A_1 \cap A_2) \cdot \ldots \cdot \mathbb{P}(A_n \mid A_1 \cap \cdots \cap A_{n-1})$$

$$= \mathbb{P}(A_1) \prod_{j=2}^{n} \mathbb{P}(A_j \mid A_1 \cap \cdots \cap A_{j-1}).$$

Proof

The formula follows immediately by recursion

$$\begin{aligned} (1) \quad & \mathbb{P}(A_1)\mathbb{P}(A_2 \mid A_1) && = && \mathbb{P}(A_1 \cap A_2) \\ (2) \quad & \mathbb{P}(A_1)\mathbb{P}(A_2 \mid A_1)\mathbb{P}(A_3 \mid A_1 \cap A_2) && = && \mathbb{P}(A_1 \cap A_2)\mathbb{P}(A_3 \mid A_1 \cap A_2) \\ & && = && \mathbb{P}(A_1 \cap A_2 \cap A_3), \end{aligned}$$

where we used the definition of the conditional probability in the first step.

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

$$P(A \cap B) = P(B)P(A|B)$$

$$P(A|B,C) = \frac{P(A,B,C)}{P(B,C)}$$

$$P(A,B|C) = P(A|B,C)P(B,C)$$

$$P(A|B,C)P(B|C)P(C)$$

Probability of trajectory

$$P(S_0, a_0, \ldots, a_T) = P(S_0, a_0, S_1 \cdots a_T)$$

$$P(S_0)P(a_0, S_1 \cdots a_T \mid S_0)$$

$$P(a_0|S_0)P(S_1 \cdots a_T \mid S_0, a_0)$$

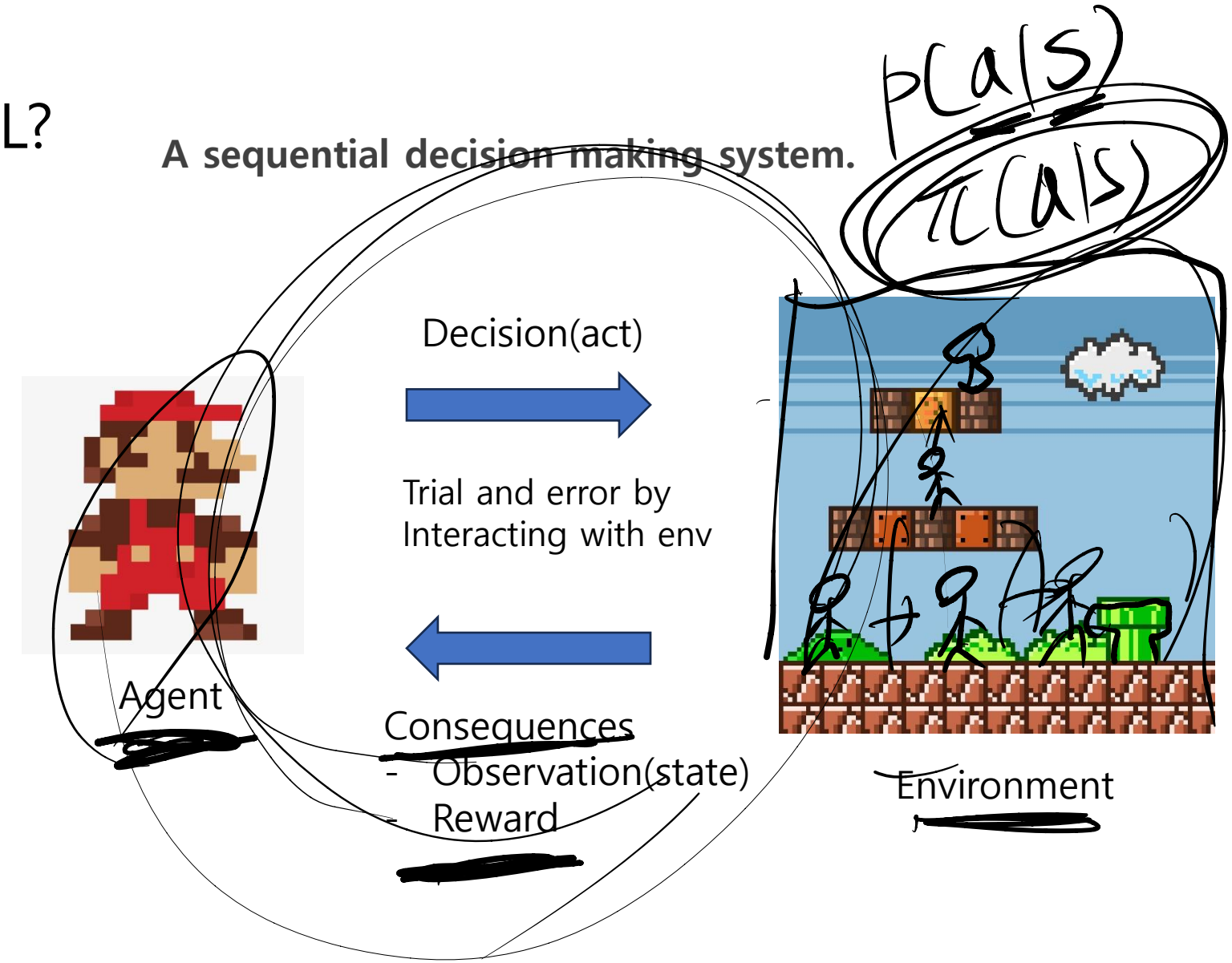
$$P(S_1|a_0, S_0)P(a_1 \cdots a_T \mid S_1, a_0, S_0)$$

$$P(a_1|S_1, a_0, S_0)P(S_2 \cdots a_T \mid a_1, S_1, a_0, S_0)$$

Key concept

What is RL?

A sequential decision making system.



Reinforcement Learning

Intuitive understanding

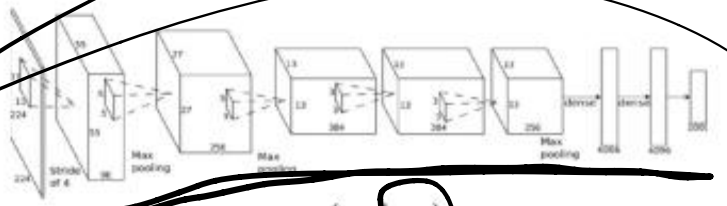
Emergence of Locomotion Behaviours in Rich Environments

Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, S. M. Ali Eslami, Martin Riedmiller, David Silver
DeepMind

https://arxiv.org/abs/1707.02286
https://www.youtube.com/watch?v=hx_bgoTF7bs

- Example of misspecified rewards
  - CoastRunner
    - Reward : pass the checkpoint, get a power up item
  - Escape Room
    - Reward : -1 per second
      - Trained policy : suicide at start
  - Half cheetah (hard to escape local optima)
    - Reward : speed
      - Trained policy : Backflip and crawl

Terminology & notation



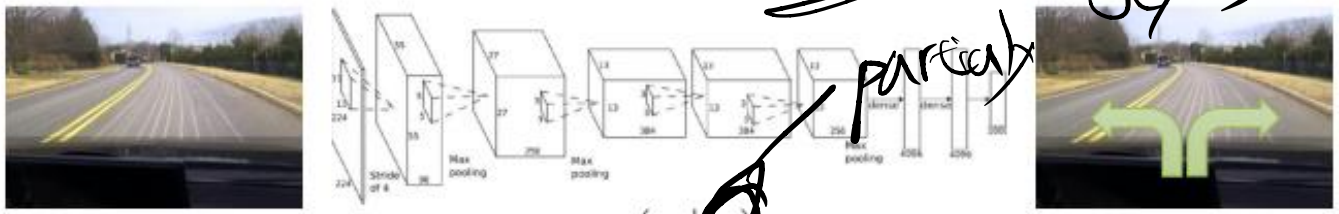
$\mathbf{o}_t$ $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ $\mathbf{a}_t$

$\mathbf{s}_t$ – state
$\mathbf{o}_t$ – observation
$\mathbf{a}_t$ – action

$\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ – policy
$\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ – policy (fully observed)

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reward functions



$\mathbf{o}_t$ $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ $\mathbf{a}_t$

which action is better or worse?

$\mathbf{s}$, $\mathbf{a}$, $r(\mathbf{s},\mathbf{a})$, and $p(\mathbf{s}'|\mathbf{s},\mathbf{a})$ define a Markov decision process

$r(\mathbf{s},\mathbf{a})$: reward function

tells us which states and actions are better

high reward

low reward

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Definitions

Markov chain

$\mathcal{M} = \{\mathcal{S}, \mathcal{T}\}$

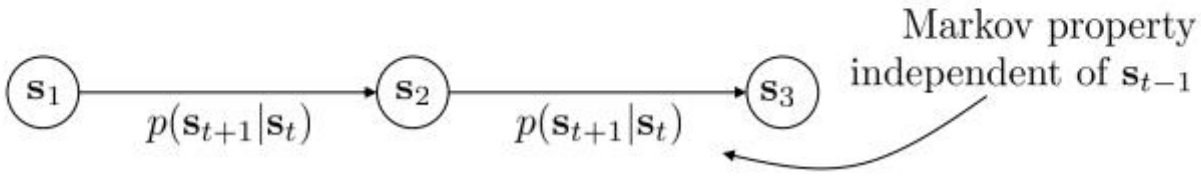
$\mathcal{S}$ – state space — states $s \in \mathcal{S}$ (discrete or continuous)

$\mathcal{T}$ – transition operator — $p(s_{t+1}|s_t)$

why "operator"? — let $\mu_{t,i} = p(s_t = i)$ — $\vec{\mu}_t$ is a vector of probabilities

let $\mathcal{T}_{i,j} = p(s_{t+1} = i | s_t = j)$ — then $\vec{\mu}_{t+1} = \mathcal{T}\vec{\mu}_t$

Andrey Markov



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reinforcement Learning

Definitions

Markov decision process $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, r\}$

$\mathcal{S}$ – state space — states $s \in \mathcal{S}$ (discrete or continuous)

$\mathcal{A}$ – action space — actions $a \in \mathcal{A}$ (discrete or continuous)

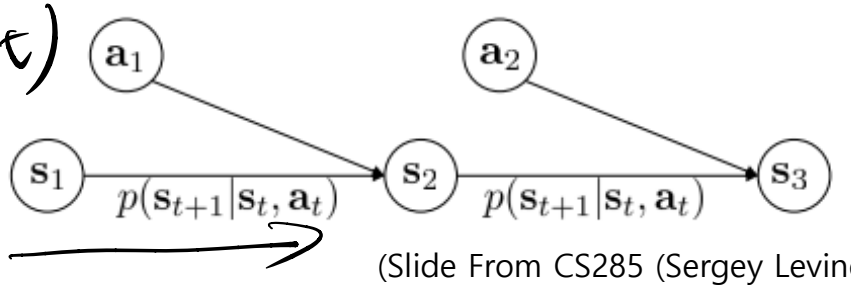
$\mathcal{T}$ – transition operator (now a tensor!)

let $\mu_{t,j} = p(s_t = j)$

let $\xi_{t,k} = p(a_t = k)$

let $\mathcal{T}_{i,j,k} = p(s_{t+1} = i | s_t = j, a_t = k)$

$$\mu_{t+1,i} = \sum_{j,k} \mathcal{T}_{i,j,k} \mu_{t,j} \xi_{t,k}$$



Richard Bellman

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Definitions

Markov decision process $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, r\}$

$\mathcal{S}$ – state space — states $s \in \mathcal{S}$ (discrete or continuous)

$\mathcal{A}$ – action space — actions $a \in \mathcal{A}$ (discrete or continuous)

$\mathcal{T}$ – transition operator (now a tensor!)

r – reward function — $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$

$r(s_t, a_t)$ – reward

Richard Bellman

Bellman Eqn.

$V(s_t)$ $Q(s_t, a_t)$

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Definitions

partially observed Markov decision process $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{E}, r\}$

$\mathcal{S}$ – state space — states $s \in \mathcal{S}$ (discrete or continuous)

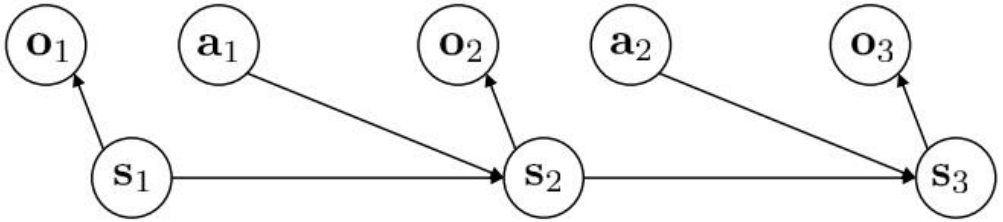
$\mathcal{A}$ – action space — actions $a \in \mathcal{A}$ (discrete or continuous)

$\mathcal{O}$ – observation space — observations $o \in \mathcal{O}$ (discrete or continuous)

$\mathcal{T}$ – transition operator (like before)

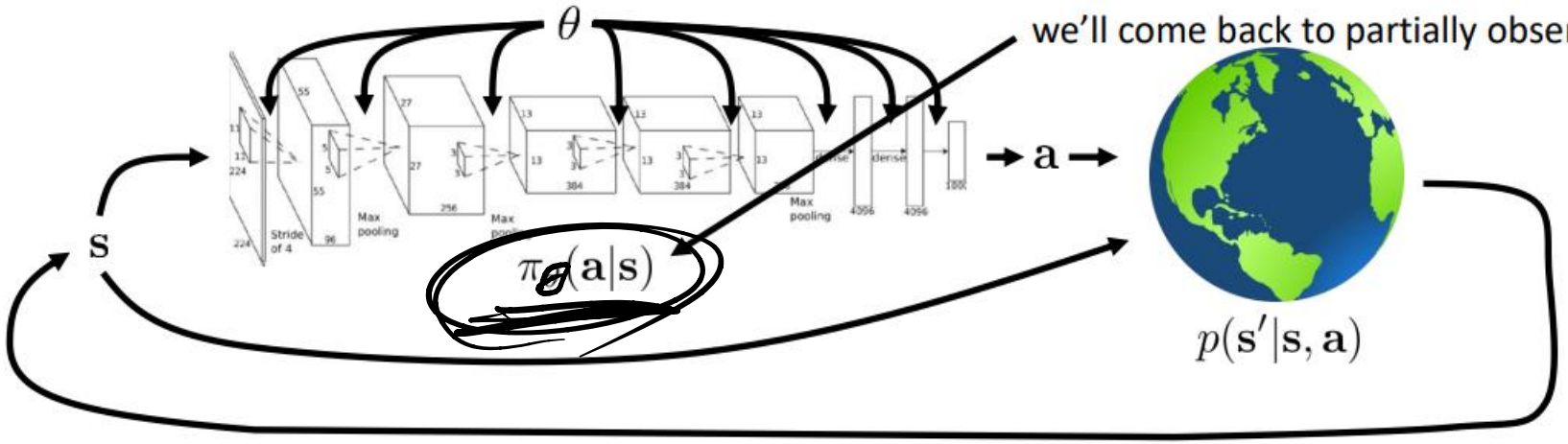
$\mathcal{E}$ – emission probability $p(o_t|s_t)$

r – reward function — $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

The goal of reinforcement learning



$$p_\theta(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T) = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

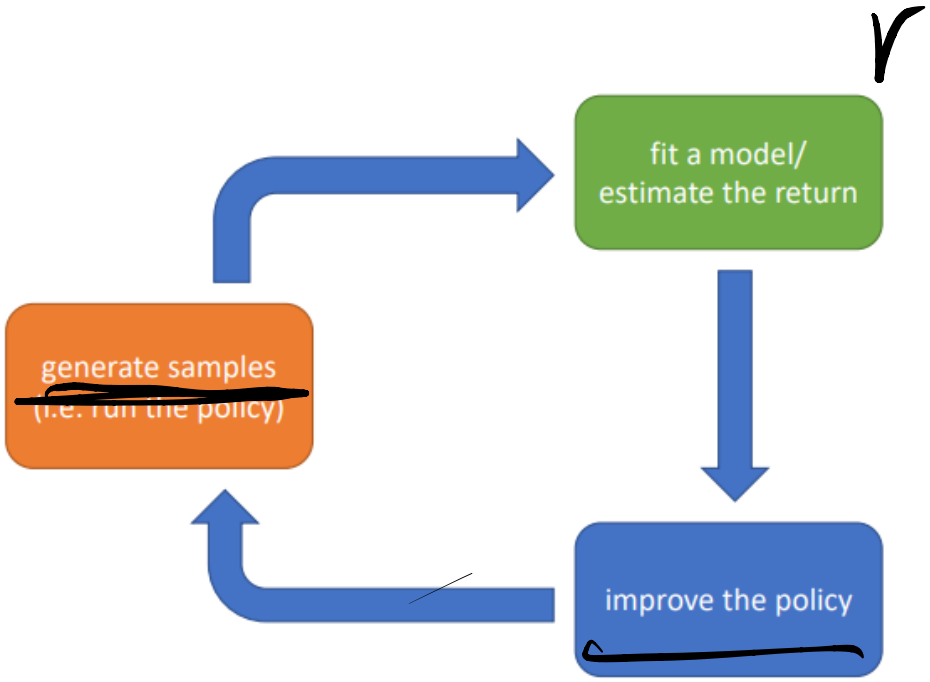
$p_\theta(\tau)$

$$\theta^\star = \arg\max_\theta E_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right]$$

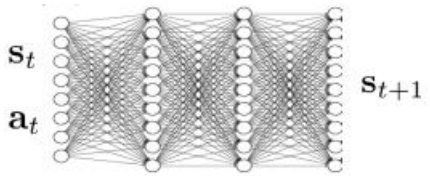
(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reinforcement Learning

The anatomy of a reinforcement learning algorithm



learn f_ϕ such that $\mathbf{s}_{t+1} \approx f_\phi(\mathbf{s}_t, \mathbf{a}_t)$



backprop through f_ϕ and r to train $\pi_\theta(\mathbf{s}_t) = \mathbf{a}_t$

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Definition: Q-function

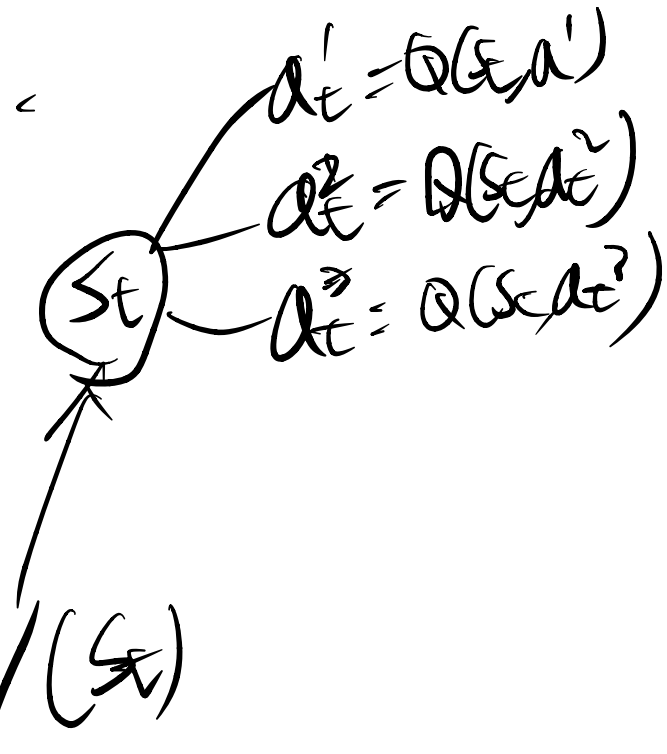
$Q^\pi(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{T} E_{\pi_\theta}[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t, \mathbf{a}_t]$: total reward from taking $\mathbf{a}_t$ in $\mathbf{s}_t$

Definition: value function

$V^\pi(\mathbf{s}_t) = \sum_{t'=t}^{T} E_{\pi_\theta}[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t]$: total reward from $\mathbf{s}_t$

$V^\pi(\mathbf{s}_t) = E_{\mathbf{a}_t \sim \pi(\mathbf{a}_t|\mathbf{s}_t)}[Q^\pi(\mathbf{s}_t, \mathbf{a}_t)]$

$E_{\mathbf{s}_1 \sim p(\mathbf{s}_1)}[V^\pi(\mathbf{s}_1)]$ is the RL objective!



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reinforcement Learning

Value Iteration : State-Value function V를 반복적으로 계산 ⇒ 정책 도출

Bellman-Equation을 활용해 V 업데이트 (DP)

Policy Iteration : 정책평가와 개선의 반복

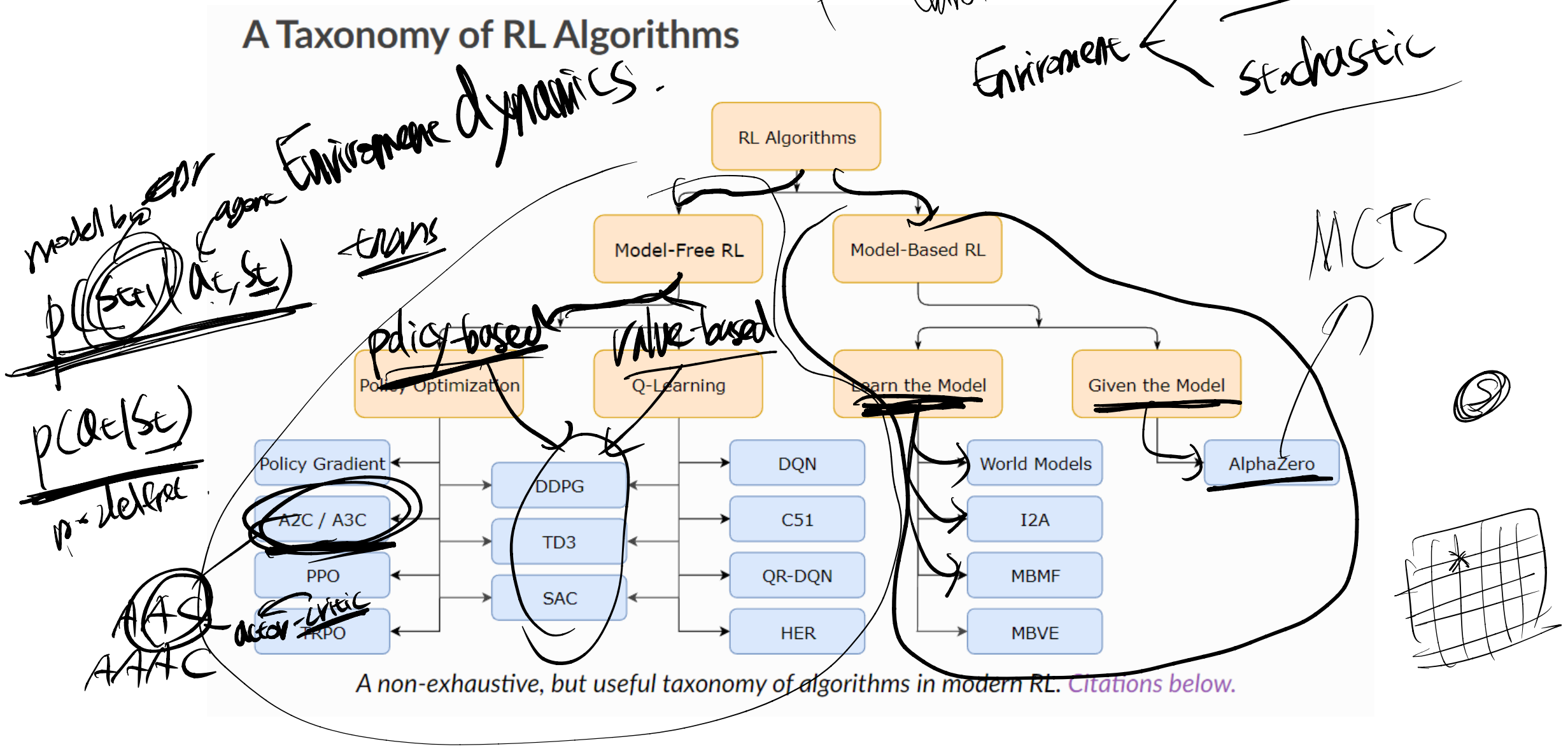
정책평가 : 현재 정책에 대한 Value 계산

정책 개선 : 계산한 V를 바탕으로 정책 update.

⇒ Grid World example

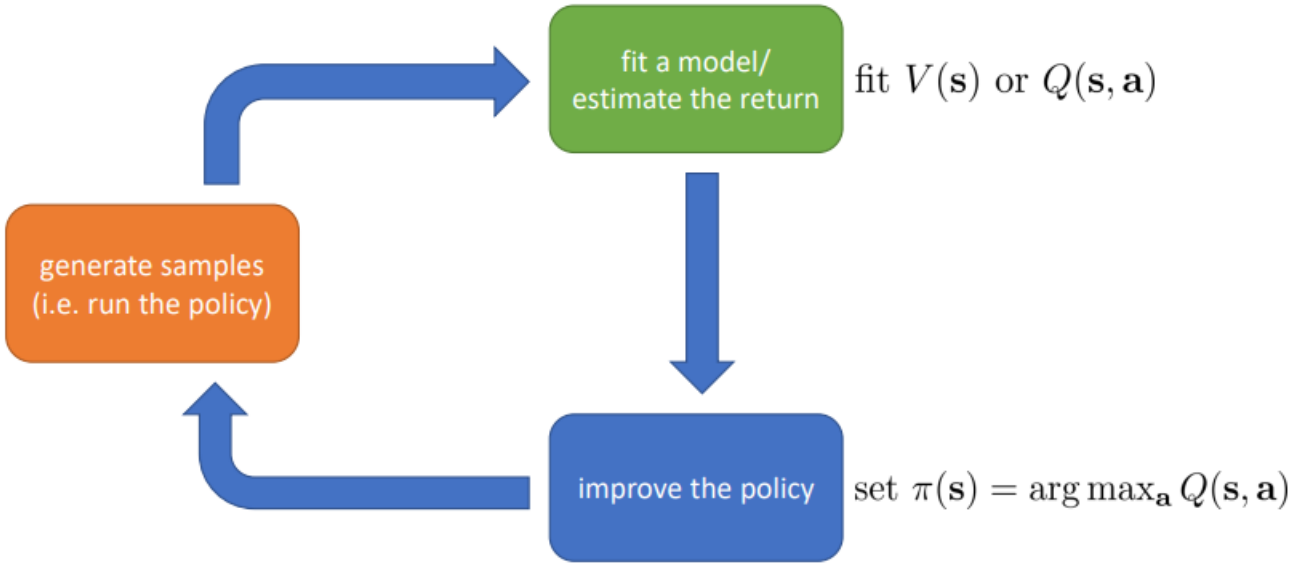
(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reinforcement Learning



Reinforcement Learning

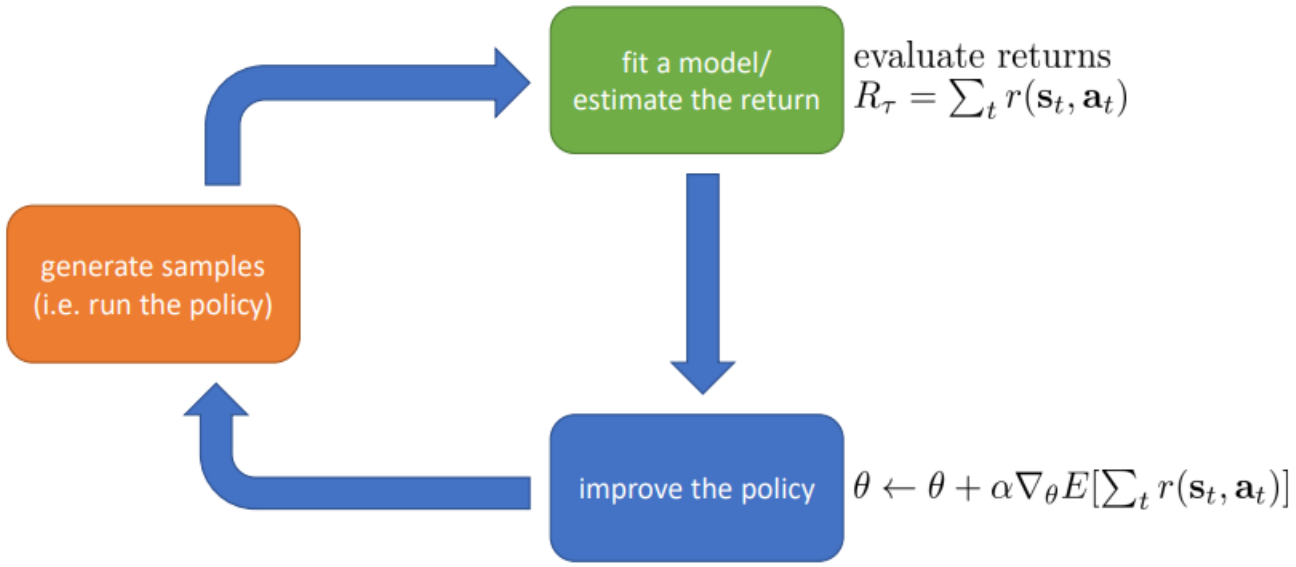
Value function based algorithms



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reinforcement Learning

Direct policy gradients



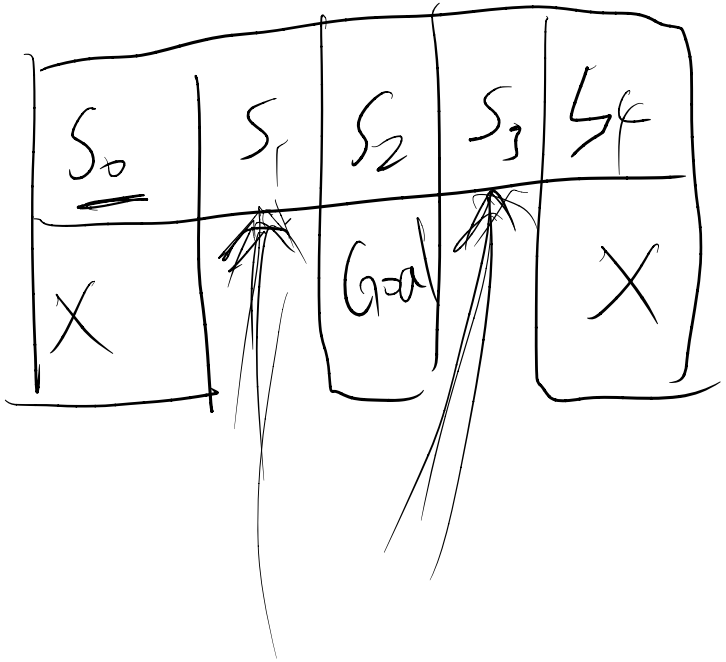
(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reinforcement Learning

Value based VS Policy based

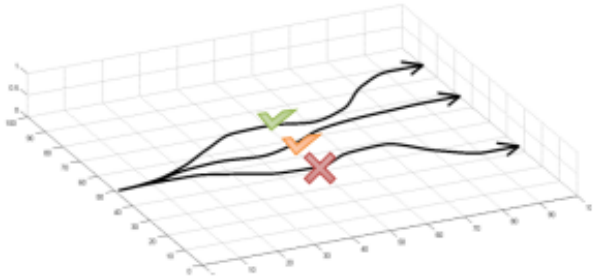
Continuous action

Stochastic policy
(especially in partially observable)



Evaluating the objective

$$\theta^\star = \arg\max_\theta \underbrace{E_{\tau\sim p_\theta(\tau)}\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right]}_{J(\theta)}$$



$$J(\theta) = E_{\tau\sim p_\theta(\tau)}\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right] \approx \frac{1}{N}\sum_i \sum_t r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$$

sum over samples from π_θ

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Direct policy differentiation

$$\theta^\star = \arg\max_\theta \underbrace{E_{\tau\sim p_\theta(\tau)}\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right]}_{J(\theta)}$$

a convenient identity

$$p_\theta(\tau)\nabla_\theta \log p_\theta(\tau) = p_\theta(\tau)\frac{\nabla_\theta p_\theta(\tau)}{p_\theta(\tau)} = \nabla_\theta p_\theta(\tau)$$

$$J(\theta) = E_{\tau\sim p_\theta(\tau)}[r(\tau)] = \int p_\theta(\tau) r(\tau) d\tau$$

$$\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t)$$

$$(\log f(x))' = \frac{f'(x)}{f(x)}$$

$$\nabla_\theta J(\theta) = \int \nabla_\theta p_\theta(\tau) r(\tau) d\tau = \int p_\theta(\tau)\nabla_\theta \log p_\theta(\tau) r(\tau) d\tau = E_{\tau\sim p_\theta(\tau)}[\nabla_\theta \log p_\theta(\tau) r(\tau)]$$

$$\nabla p_\theta(\tau) = \frac{\nabla p_\theta(\tau)}{p_\theta(\tau)} p_\theta(\tau) = p_\theta(\tau)\log p_\theta(\tau)$$

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Direct policy differentiation

$$\theta^\star = \arg\max_\theta J(\theta)$$

$$J(\theta) = E_{\tau\sim p_\theta(\tau)}[r(\tau)]$$

$$\nabla_\theta J(\theta) = E_{\tau\sim p_\theta(\tau)}[\nabla_\theta \log p_\theta(\tau) r(\tau)]$$

chain rule.

$$p_\theta(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T) = p(\mathbf{s}_1)\prod_{t=1}^{T}\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

$p_\theta(\tau)$

log of both sides

$$\log p_\theta(\tau) = \log p(\mathbf{s}_1) + \sum_{t=1}^{T}\log\pi_\theta(\mathbf{a}_t|\mathbf{s}_t) + \log p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

항 날리기 → 미분

$$\nabla_\theta\left[\log p(\mathbf{s}_1) + \sum_{t=1}^{T}\log\pi_\theta(\mathbf{a}_t|\mathbf{s}_t) + \log p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)\right]$$

$$\nabla_\theta J(\theta) = E_{\tau\sim p_\theta(\tau)}\left[\left(\sum_{t=1}^{T}\nabla_\theta\log\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)\right)\left(\sum_{t=1}^{T}r(\mathbf{s}_t, \mathbf{a}_t)\right)\right]$$

Don't let the past distract you

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

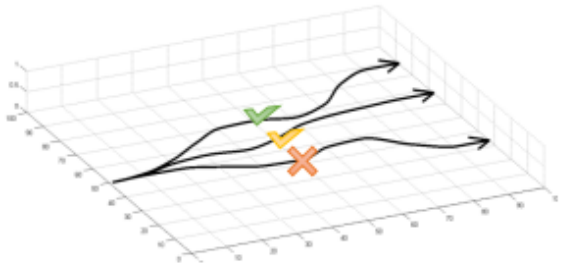
Evaluating the policy gradient

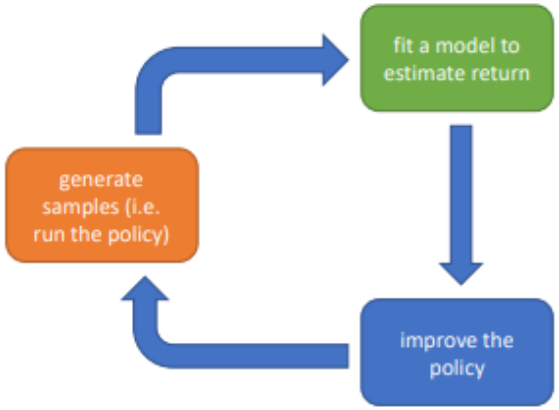
recall: $J(\theta) = E_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right] \approx \frac{1}{N} \sum_i \sum_t r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$

$$\nabla_\theta J(\theta) = E_{\tau \sim p_\theta(\tau)} \left[\left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_t | \mathbf{s}_t) \right) \left(\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \right) \right]$$

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t} | \mathbf{s}_{i,t}) \right) \left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) \right)$$

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$





REINFORCE algorithm:

1. sample $\{\tau^i\}$ from $\pi_\theta(\mathbf{a}_t | \mathbf{s}_t)$ (run the policy)
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i | \mathbf{s}_t^i) \right) \left(\sum_t r(\mathbf{s}_t^i, \mathbf{a}_t^i) \right)$
3. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

Gaussian distribution

Example: Gaussian policies

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) \right) \left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) \right)$$

example: $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t) = \mathcal{N}(f_{\text{neural network}}(\mathbf{s}_t); \Sigma)$

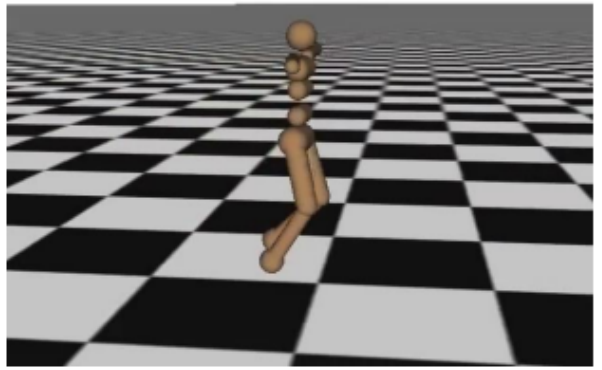
$$\log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) = -\frac{1}{2}\|f(\mathbf{s}_t) - \mathbf{a}_t\|_\Sigma^2 + \text{const}$$

$$\nabla_\theta \log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) = -\frac{1}{2}\Sigma^{-1}(f(\mathbf{s}_t) - \mathbf{a}_t)\frac{df}{d\theta}$$

REINFORCE algorithm:

1. sample $\{\tau^i\}$ from $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ (run it on the robot)
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i)\right) \left(\sum_t r(\mathbf{s}_t^i, \mathbf{a}_t^i)\right)$
3. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

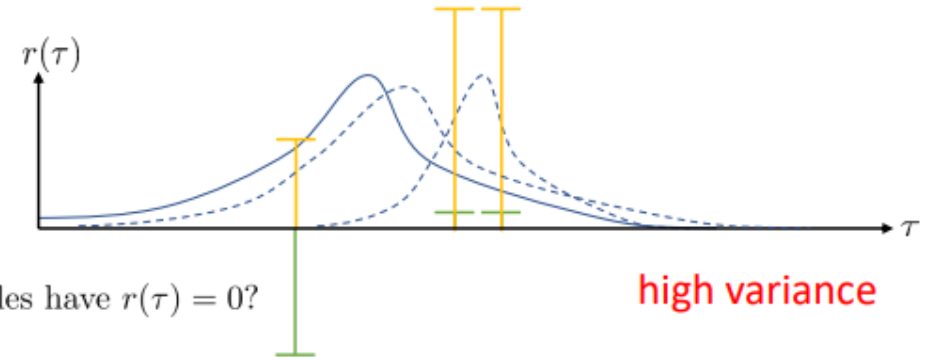
Iteration 2000



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

What is wrong with the policy gradient?

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \log \pi_\theta(\tau) r(\tau)$$



even worse: what if the two "good" samples have $r(\tau) = 0$?

high variance

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reducing variance

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) \right) \left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) \right)$$

Causality: policy at time t' cannot affect reward at time t when $t < t'$

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) \left(\sum_{t'=t}^{T} r(\mathbf{s}_{i,t'}, \mathbf{a}_{i,t'}) \right)$$

"reward to go"

$\hat{Q}_{i,t}$

return 값

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Baselines

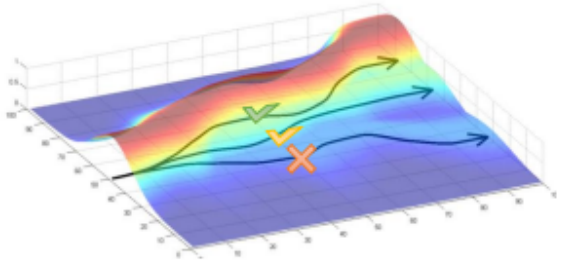
a convenient identity

$$p_\theta(\tau)\nabla_\theta \log p_\theta(\tau) = \nabla_\theta p_\theta(\tau)$$

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \log p_\theta(\tau)[r(\tau) - b]$$

$$b = \frac{1}{N}\sum_{i=1}^{N} r(\tau)$$

but... are we *allowed* to do that??



$$E[\nabla_\theta \log p_\theta(\tau) b] = \int p_\theta(\tau)\nabla_\theta \log p_\theta(\tau) b \, d\tau = \int \nabla_\theta p_\theta(\tau) b \, d\tau = b\nabla_\theta \int p_\theta(\tau) d\tau = b\nabla_\theta 1 = 0$$

subtracting a baseline is *unbiased* in expectation!

average reward is *not* the best baseline, but it's pretty good!

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Analyzing variance

can we write down the variance?

$$\mathrm{Var}[x] = E[x^2] - E[x]^2$$

$$\nabla_\theta J(\theta) = E_{\tau\sim p_\theta(\tau)}[\nabla_\theta \log p_\theta(\tau)(r(\tau) - b)]$$

$$\mathrm{Var} = E_{\tau\sim p_\theta(\tau)}[(\nabla_\theta \log p_\theta(\tau)(r(\tau) - b))^2] - \underbrace{E_{\tau\sim p_\theta(\tau)}[\nabla_\theta \log p_\theta(\tau)(r(\tau) - b)]^2}_{\text{this bit is just } E_{\tau\sim p_\theta(\tau)}[\nabla_\theta \log p_\theta(\tau) r(\tau)]}$$

(baselines are unbiased in expectation)

$$\frac{d\mathrm{Var}}{db} = \frac{d}{db} E[g(\tau)^2(r(\tau) - b)^2] = \frac{d}{db}\left(E[g(\tau)^2 r(\tau)^2] - 2E[g(\tau)^2 r(\tau) b] + b^2 E[g(\tau)^2]\right)$$

$$= -2E[g(\tau)^2 r(\tau)] + 2bE[g(\tau)^2] = 0$$

$$b = \frac{E[g(\tau)^2 r(\tau)]}{E[g(\tau)^2]}$$

This is just expected reward, but weighted by gradient magnitudes!

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Policy gradient is on-policy

$$\theta^\star = \arg\max_\theta J(\theta)$$

$$J(\theta) = E_{\tau\sim p_\theta(\tau)}[r(\tau)]$$

$$\nabla_\theta J(\theta) = E_{\tau\sim p_\theta(\tau)}[\nabla_\theta \log p_\theta(\tau) r(\tau)]$$

this is trouble...

- Neural networks change only a little bit with each gradient step
- On-policy learning can be extremely inefficient!

can't just skip this!

REINFORCE algorithm:

1. sample $\{\tau^i\}$ from $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ (run it on the robot)
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i)\right)\left(\sum_t r(\mathbf{s}_t^i, \mathbf{a}_t^i)\right)$
3. $\theta \leftarrow \theta + \alpha\nabla_\theta J(\theta)$

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Recap: policy gradients

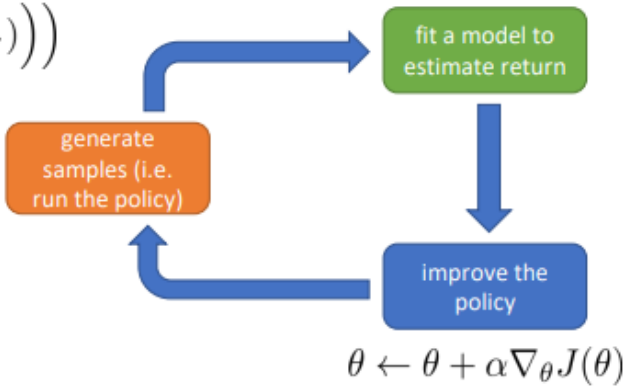
REINFORCE algorithm:

1. sample $\{\tau^i\}$ from $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ (run the policy)
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i) \left(\sum_{t'=t}^{T} r(\mathbf{s}_{t'}^i, \mathbf{a}_{t'}^i)\right)\right)$
3. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) \hat{Q}_{i,t}^{\pi}$$

"reward to go"

$$\hat{Q}^\pi(\mathbf{x}_t, \mathbf{u}_t) = \sum_{t'=t}^{T} r(\mathbf{x}_{t'}, \mathbf{u}_{t'})$$



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

What about the baseline?

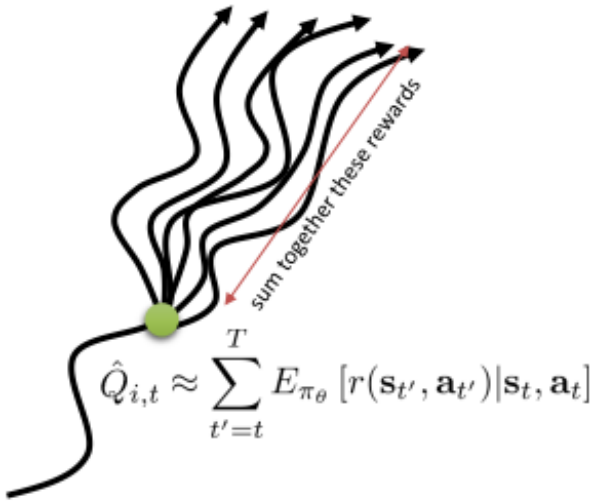
$Q(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{T} E_{\pi_\theta}\left[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t, \mathbf{a}_t\right]$: true *expected* reward-to-go

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t})\left(Q(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) - V(\mathbf{s}_{i,t})\right)$$

$$b_t = \frac{1}{N}\sum_i Q(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$$

average what?

$$V(\mathbf{s}_t) = E_{\mathbf{a}_t \sim \pi_\theta(\mathbf{a}_t|\mathbf{s}_t)}[Q(\mathbf{s}_t, \mathbf{a}_t)]$$



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

State & state-action value functions

$Q^\pi(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{T} E_{\pi_\theta}\left[r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) | \mathbf{s}_t, \mathbf{a}_t\right]$: total reward from taking $\mathbf{a}_t$ in $\mathbf{s}_t$

$V^\pi(\mathbf{s}_t) = E_{\mathbf{a}_t \sim \pi_\theta(\mathbf{a}_t|\mathbf{s}_t)}[Q^\pi(\mathbf{s}_t, \mathbf{a}_t)]$: total reward from $\mathbf{s}_t$

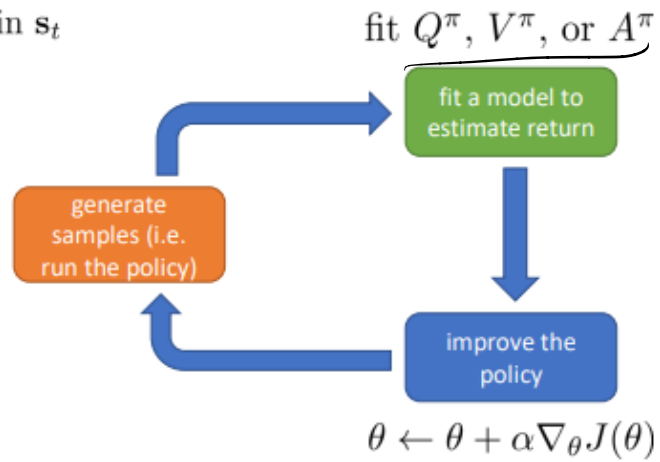
$A^\pi(\mathbf{s}_t, \mathbf{a}_t) = Q^\pi(\mathbf{s}_t, \mathbf{a}_t) - V^\pi(\mathbf{s}_t)$: how much better $\mathbf{a}_t$ is

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) A^\pi(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$$

the better this estimate, the lower the variance

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) \left(\sum_{t'=1}^{T} r(\mathbf{s}_{i,t'}, \mathbf{a}_{i,t'}) - b \right)$$

unbiased, but high variance single-sample estimate



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Value function fitting

$$Q^\pi(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{T} E_{\pi_\theta}\left[r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) | \mathbf{s}_t, \mathbf{a}_t\right]$$

$$V^\pi(\mathbf{s}_t) = E_{\mathbf{a}_t \sim \pi_\theta(\mathbf{a}_t|\mathbf{s}_t)}[Q^\pi(\mathbf{s}_t, \mathbf{a}_t)]$$

$$A^\pi(\mathbf{s}_t, \mathbf{a}_t) = Q^\pi(\mathbf{s}_t, \mathbf{a}_t) - V^\pi(\mathbf{s}_t)$$

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) A^\pi(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$$

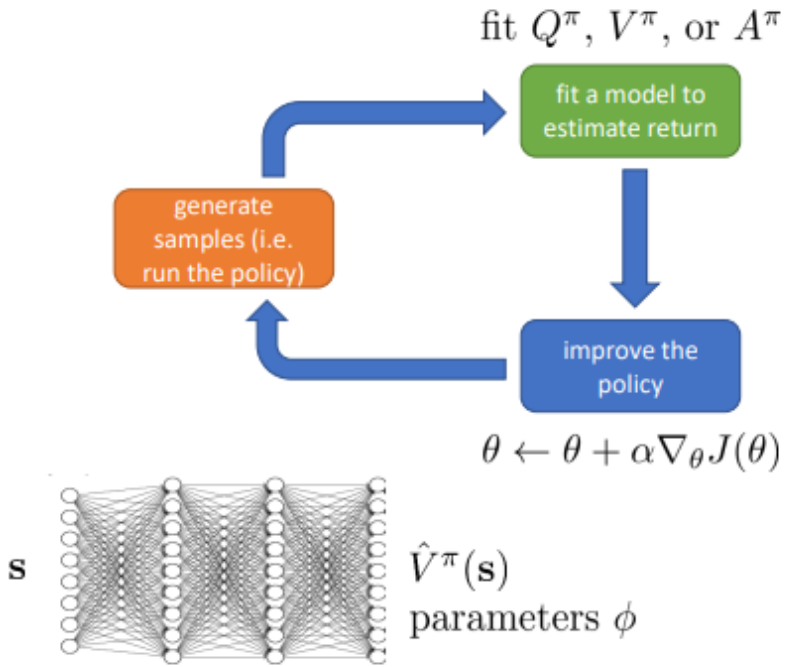
fit *what* to *what*?

Q^π, V^π, A^π?

$$Q^\pi(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + \sum_{t'=t+1}^{T} E_{\pi_\theta}\left[r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) | \mathbf{s}_t, \mathbf{a}_t\right]$$

$$A^\pi(\mathbf{s}_t, \mathbf{a}_t) \approx r(\mathbf{s}_t, \mathbf{a}_t) + V^\pi(\mathbf{s}_{t+1}) - V^\pi(\mathbf{s}_t)$$

let's just fit $V^\pi(\mathbf{s})$!



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Policy evaluation

$$V^{\pi}(\mathbf{s}_t) = \sum_{t'=t}^{T} E_{\pi_\theta}\left[r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) | \mathbf{s}_t\right]$$

$$J(\theta) = E_{\mathbf{s}_1 \sim p(\mathbf{s}_1)}[V^{\pi}(\mathbf{s}_1)]$$

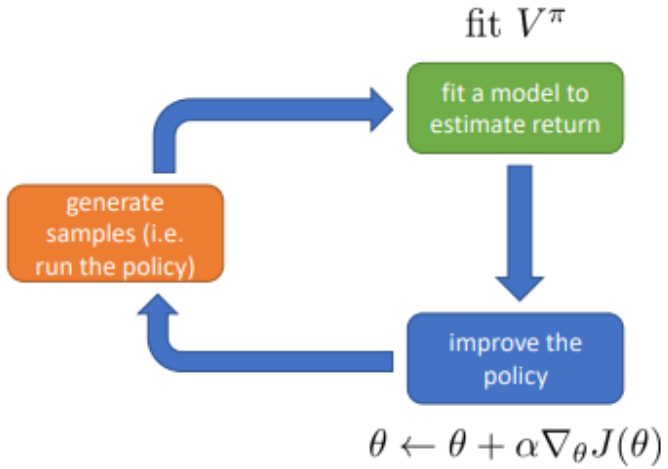
how can we perform policy evaluation?

Monte Carlo policy evaluation (this is what policy gradient does)

$$V^{\pi}(\mathbf{s}_t) \approx \sum_{t'=t}^{T} r(\mathbf{s}_{t'}, \mathbf{a}_{t'})$$

$$V^{\pi}(\mathbf{s}_t) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t'=t}^{T} r(\mathbf{s}_{t'}, \mathbf{a}_{t'})$$

(requires us to reset the simulator)

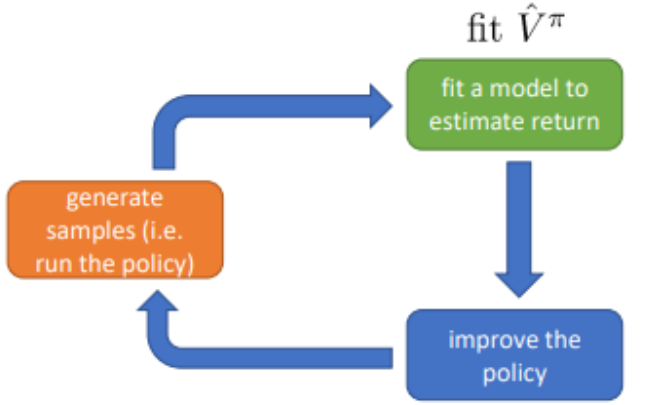


(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

An actor-critic algorithm

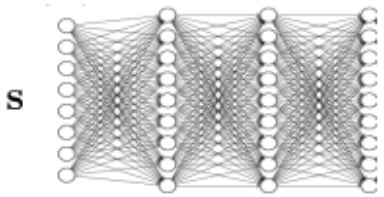
batch actor-critic algorithm:

1. sample $\{\mathbf{s}_i, \mathbf{a}_i\}$ from $\pi_\theta(\mathbf{a}|\mathbf{s})$ (run it on the robot)
2. fit $\hat{V}^\pi_\phi(\mathbf{s})$ to sampled reward sums
3. evaluate $\hat{A}^\pi(\mathbf{s}_i, \mathbf{a}_i) = r(\mathbf{s}_i, \mathbf{a}_i) + \hat{V}^\pi_\phi(\mathbf{s}'_i) - \hat{V}^\pi_\phi(\mathbf{s}_i)$
4. $\nabla_\theta J(\theta) \approx \sum_i \nabla_\theta \log \pi_\theta(\mathbf{a}_i|\mathbf{s}_i)\hat{A}^\pi(\mathbf{s}_i, \mathbf{a}_i)$
5. $\theta \leftarrow \theta + \alpha\nabla_\theta J(\theta)$



$$y_{i,t} \approx \sum_{t'=t}^{T} r(\mathbf{s}_{i,t'}, \mathbf{a}_{i,t'})$$

$$\mathcal{L}(\phi) = \frac{1}{2}\sum_i \left\| \hat{V}^\pi_\phi(\mathbf{s}_i) - y_i \right\|^2$$



$\hat{V}^\pi(\mathbf{s})$
parameters ϕ

$$V^\pi(\mathbf{s}_t) = \sum_{t'=t}^{T} E_{\pi_\theta}\left[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t\right]$$

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Aside: discount factors

$$y_{i,t} \approx r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) + \hat{V}^{\pi}_{\phi}(\mathbf{s}_{i,t+1})$$

$$\mathcal{L}(\phi) = \frac{1}{2} \sum_{i} \left\| \hat{V}^{\pi}_{\phi}(\mathbf{s}_i) - y_i \right\|^2$$

what if T (episode length) is ∞?

$\hat{V}^{\pi}_{\phi}$ can get infinitely large in many cases

episodic tasks

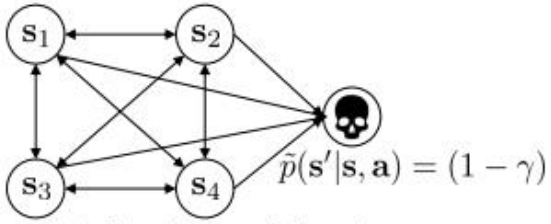
continuous/cyclical tasks

simple trick: better to get rewards sooner than later

γ changes the MDP:

$$y_{i,t} \approx r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) + \gamma \hat{V}^{\pi}_{\phi}(\mathbf{s}_{i,t+1})$$

discount factor $\gamma \in [0, 1]$ (0.99 works well)



$$\tilde{p}(\mathbf{s}'|\mathbf{s}, \mathbf{a}) = (1 - \gamma)$$

$$\tilde{p}(\mathbf{s}'|\mathbf{s}, \mathbf{a}) = \gamma p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$$

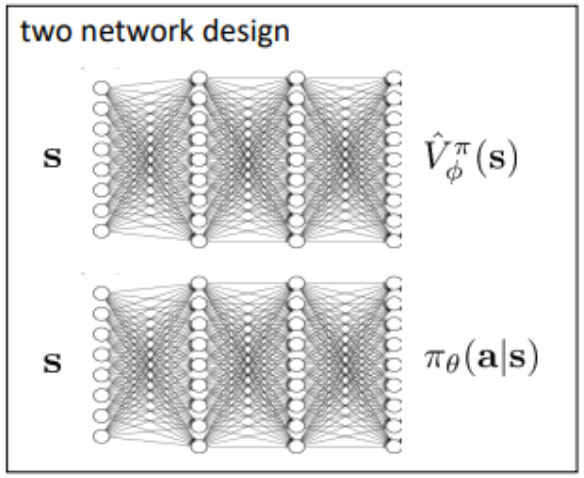
(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Architecture design

online actor-critic algorithm:

1. take action $\mathbf{a} \sim \pi_\theta(\mathbf{a}|\mathbf{s})$, get $(\mathbf{s}, \mathbf{a}, \mathbf{s}', r)$
2. update $\hat{V}^\pi_\phi$ using target $r + \gamma \hat{V}^\pi_\phi(\mathbf{s}')$
3. evaluate $\hat{A}^\pi(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \gamma \hat{V}^\pi_\phi(\mathbf{s}') - \hat{V}^\pi_\phi(\mathbf{s})$
4. $\nabla_\theta J(\theta) \approx \nabla_\theta \log \pi_\theta(\mathbf{a}|\mathbf{s}) \hat{A}^\pi(\mathbf{s}, \mathbf{a})$
5. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

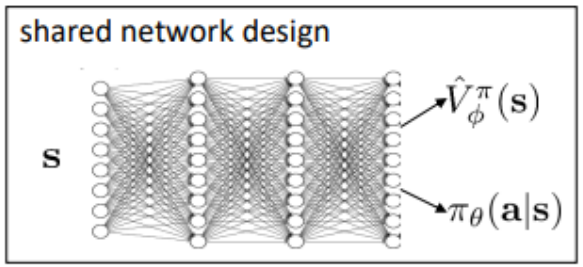
two network design



+ simple & stable

- no shared features between actor & critic

shared network design



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

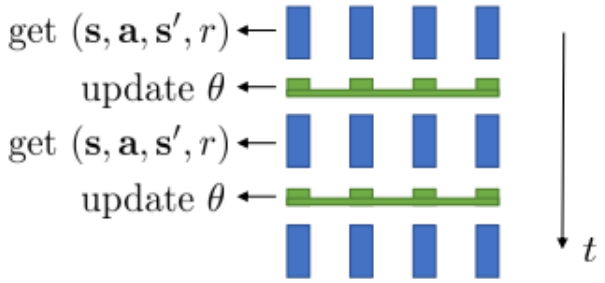
Online actor-critic in practice

online actor-critic algorithm:

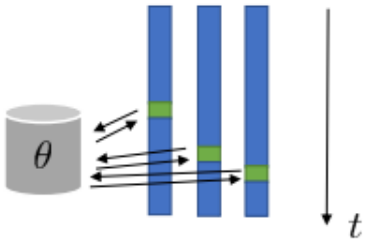
1. take action $\mathbf{a} \sim \pi_\theta(\mathbf{a}|\mathbf{s})$, get $(\mathbf{s}, \mathbf{a}, \mathbf{s}', r)$
2. update $\hat{V}^\pi_\phi$ using target $r + \gamma \hat{V}^\pi_\phi(\mathbf{s}')$
3. evaluate $\hat{A}^\pi(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \gamma \hat{V}^\pi_\phi(\mathbf{s}') - \hat{V}^\pi_\phi(\mathbf{s})$
4. $\nabla_\theta J(\theta) \approx \nabla_\theta \log \pi_\theta(\mathbf{a}|\mathbf{s}) \hat{A}^\pi(\mathbf{s}, \mathbf{a})$
5. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

works best with a batch (e.g., parallel workers)

synchronized parallel actor-critic



asynchronous parallel actor-critic



(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reinforcement Learning

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reinforcement Learning

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)

Reinforcement Learning

(Slide From CS285 (Sergey Levine) Lecture : https://rail.eecs.berkeley.edu/deeprlcourse-fa21/)