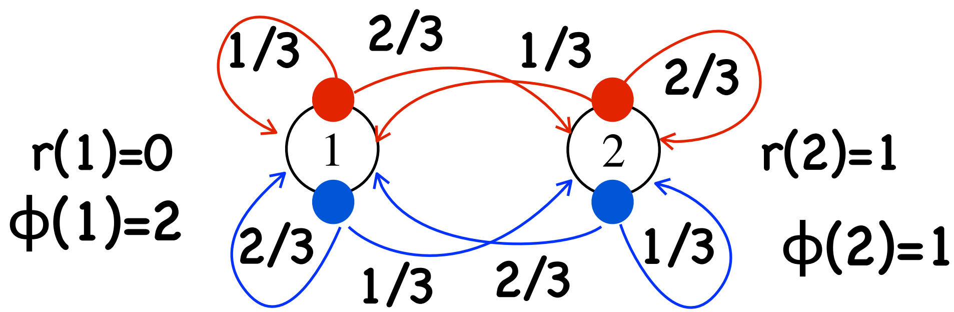
南京大学
人工智能学院
SCHOOL OF ARTIFICIAL INTELLIGENCE, NANJING UNIVERSITY

Lecture 7
Policy Gradient

Policy degradation in value function based methods

[Bartlett. An Introduction to Reinforcement Learning Theory: Value Function Methods. Advanced Lectures on Machine Learning, LNAI 2600]



optimal policy: red

$V^*(2) > V^*(1) > 0$

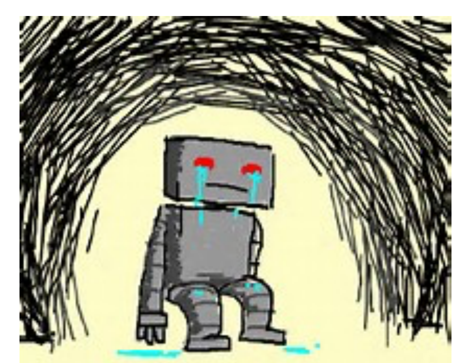
let $\hat{V}(s) = w\phi(s)$, to ensure $\hat{V}(2) > \hat{V}(1)$, $w < 0$

as value function based method minimizes $\|\hat{V} - V^*\|$

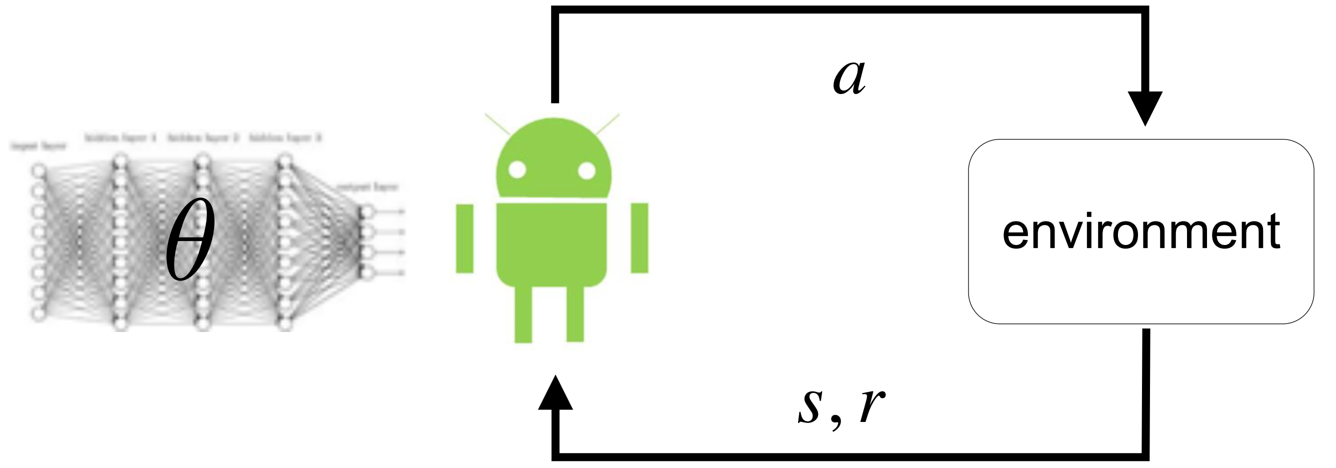
results in $w > 0$

sub-optimal policy, better value ≠ better policy

Policy Search



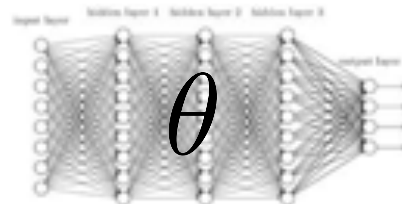
Policy search



recall in Lecture 3, we use black box search

we are to use the MDP structure

Policy space

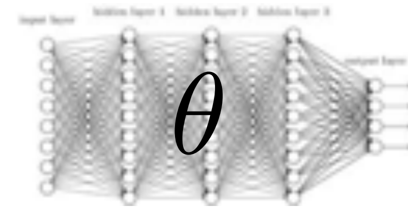


for parameterized differentiable models, we consider parameter space

for nonparametric / non-differentiable models, we consider function space

Parameterized policy

parameterized model $f(s;\theta)$



θ

Discrete actions: Gibbs policy (logistic regression)

$f(s;\theta)$ has $|A|$ output heads $f(a|s;\theta)$ is the output of head a

$$\pi_\theta(a|s) = \frac{\exp(f(a|s;\theta))}{\sum_{a'} \exp(f(a'|s;\theta))}$$

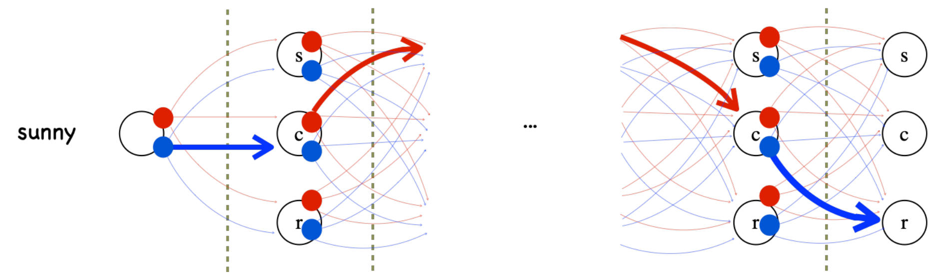
Continuous action: Gaussian policy

$f(s;\theta)$ has 2 output heads

$$\pi_\theta(a|s) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(f(s;\theta)-a)^2}{\sigma^2}\right)$$

Direct objective function — Trajectory-wise

episodic environments



trajectory space:

all possible trajectories

$$Tra = \begin{matrix} s,\ s,\ s & s,\ c,\ s & s,\ r,\ s \\ s,\ s,\ c & s,\ c,\ c & s,\ r,\ c \\ s,\ s,\ r & s,\ c,\ r & s,\ r,\ r \end{matrix}$$

combination with actions

probability of generating a trajectory by policy

trajectory $\tau = s_0, a_1, s_1, a_2, \ldots, s_T$

probability $p_\theta(\tau) = p(s_0) \prod_{i=1}^{T} p(s_i | a_i, s_{i-1}) \pi_\theta(a_i | s_{i-1})$

expected total reward

$$J(\theta) = \int_{Tra} p_\theta(\tau) R(\tau)\ \mathrm{d}\tau$$

From trajectories to stationary distribution

ignoring actions and consider 3 steps

s, s, s	s, c, s	s, r, s
s, s, c	s, c, c	s, r, c
s, s, r	s, c, r	s, r, r

⇒ probability of all state *s*

$$d^{\pi_\theta}(s)$$

occupancy measure

$$p^\pi(s) = \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi)$$

note $\sum_{t=0}^{\infty} \gamma^t = \frac{1}{1-\gamma}$

stationary distribution (state)

$$d^\pi(s) = (1-\gamma) \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi)$$

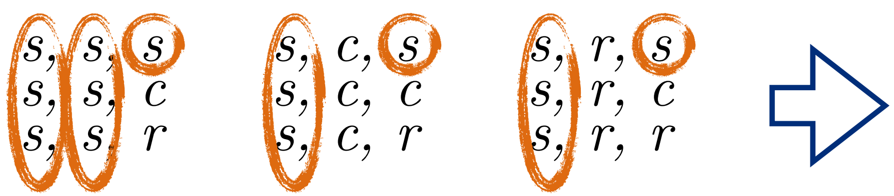
stationary distribution (state, action) $d^\pi(s,a) = (1-\gamma) \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi)\pi(a|s)$

Direct objective function — Stationary dist.

continuing environments: one-step MDPs

d^{π_θ} is the stationary distribution

e.g. ignoring actions and consider 3 steps



probability of all state *s*

$$d^{\pi_\theta}(s)$$

expected total reward

$$J(\theta) = \int_S d^{\pi_\theta}(s) \int_A \pi_\theta(a|s) r(s,a) \ \mathrm{ds} \ \mathrm{da}$$

assume r is stationary

$$V^\pi(s) = E[d^\pi(s,a) r(s,a)]$$

Analytical optimization

$$J(\theta) = \int_{Tra} p_\theta(\tau) R(\tau) \ \mathrm{d}\tau$$

$$\nabla_\theta J(\theta) = \int_{Tra} \nabla_\theta p_\theta(\tau) R(\tau) \ \mathrm{d}\tau$$

$$\nabla_\theta J(\theta) = \int_{Tra} p_\theta(\tau) \nabla_\theta \log p_\theta(\tau) R(\tau) \ \mathrm{d}\tau$$

logarithm trick

$$\nabla_\theta p_\theta = p_\theta \nabla_\theta \log p_\theta$$

$$p_\theta(\tau) = p(s_0) \prod_{i=1}^{T} p(s_i | a_i, s_{i-1}) \pi_\theta(a_i | s_{i-1})$$

structure information

$$\nabla_\theta \log p_\theta(\tau) = \sum_{i=1}^{T} \nabla_\theta \log \pi_\theta(a_i | s_{i-1}) + \mathrm{const}$$

Analytical optimization

Gibbs policy $\pi_\theta(i|s) = \dfrac{\exp(\theta_i^\top \phi(s))}{\sum_j \exp(\theta_j^\top \phi(s))}$

$$\nabla_{\theta_j} \log \pi_\theta(a_i|s_i) = \begin{cases} \phi(s_i, a_i)(1 - \pi_\theta(a_i|s_i)), & i = j \\ -\phi(s_i, a_i)\pi_\theta(a_i|s_i) & i \neq j \end{cases}$$

Gaussian policy $\pi_\theta(a|s) = \dfrac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\dfrac{(\theta^\top \phi(s) - a)^2}{\sigma^2}\right)$

$$\nabla_{\theta_j} \log \pi_\theta(a_i|s_i) = -2\frac{(\theta^\top \phi(s) - a)\phi(s)}{\sigma^2} + \text{const}$$

Analytical optimization

$$J(\theta) = \int_{Tra} p_\theta(\tau) R(\tau)\ \mathrm{d}\tau$$

gradient: $$\nabla_\theta J(\theta) = \int_{Tra} p_\theta(\tau) \sum_{i=1}^{T} \nabla_\theta \log \pi_\theta(a_i|s_{i-1}) R(\tau)\ \mathrm{d}\tau$$

$$= E[\sum_{i=1}^{T} \nabla_\theta \log \pi_\theta(a_i|s_{i-1}) r(s_i, a_i)]$$

use samples to estimate the gradient (unbiased estimation)

Analytical optimization: One-step MDPs

$$J(\theta) = \int_S d^{\pi_\theta}(s) \int_A \pi_\theta(a|s) r(s,a) \text{ ds da}$$

logarithm trick $\nabla_\theta \pi_\theta = \pi_\theta \nabla_\theta \log \pi_\theta$

$$\nabla_\theta J(\theta) = \int_S d^{\pi_\theta}(s) \int_A \pi_\theta(a|s) \nabla_\theta \log \pi_\theta(a|s) r(s,a) \text{ ds da}$$

$$= E[\nabla_\theta \log \pi_\theta(a|s) r(s,a)]$$

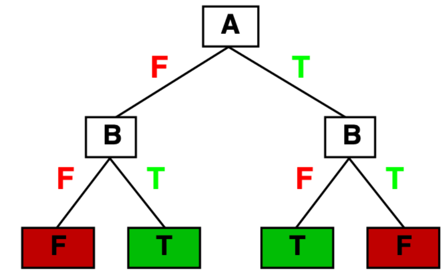
equivalent to $E[\sum_{i=1}^{T} \nabla_\theta \log \pi_\theta(a_i|s_i) R(s_i, a_i)]$

use samples to estimate the gradient (unbiased estimation)

Nonparametric/nondifferential models

base model $f(s;\theta)$

aggregated model $F(s) = \sum_{i=1}^{N} f_i(s)$



a decision-tree model

Discrete actions: Gibbs policy (logistic regression)

$$\pi_F(a|s) = \frac{\exp(F(a|s))}{\sum_{a'} \exp(F(a'|s))}$$

Continuous action: Gaussian policy

$$\pi_F(a|s) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(F(s)-a)^2}{\sigma^2}\right)$$

Nonparametric/nondifferential models

$$J(F) = \int_S d^{\pi_F}(s) \int_A \pi_F(a|s) r(s,a) \text{ ds da}$$

aggregated model $F_N(s) = \sum_{i=1}^{N} f_i(s)$

functional gradient update rule

$$F_{N+1} = F_N + \alpha \nabla_F J(F)$$

$$f_{N+1} = \nabla_F J(F)$$

solve the next base model

$$\text{solve } \min_f \sum_{s,a} \| f(a|s) - \nabla_F J(F(a|s)) |^2$$

Nonparametric/nondifferential models

functional gradient

$$\nabla_F J(F) = E[\nabla_F \log \pi_F(a|s) r(s,a)]$$

Then for discrete action space, we have

$$\nabla_{\Psi(\boldsymbol{s},a)} \pi(a \mid \boldsymbol{s}) = \pi_\Psi(a \mid \boldsymbol{s})(1 - \pi_\Psi(a \mid \boldsymbol{s}))$$

and for continuous action space,

$$\nabla_{\Psi(\boldsymbol{s},a)} \pi(a \mid \boldsymbol{s}) = 2\pi_\Psi(a \mid \boldsymbol{s})(a - \Psi(\boldsymbol{s}))/\sigma^2.$$

Issue of policy gradient

supervised gradient

$$J(\theta) = \int_x p(x)\text{loss}_\theta(x)\ \mathrm{dx}$$

policy gradient

$$J(\theta) = \int_{Tra} p_\theta(\tau)R(\tau)\ \mathrm{d}\tau \qquad \int_{Tra} p_\theta(\tau)\ \mathrm{d}\tau = 1$$

sampling

$$J(\theta) = \sum_{i=1}^{m} \text{loss}_\theta(x)$$

$$J(\theta) = \sum_{i=1}^{m} p_\theta(\tau_i)R(\tau_i)$$

$$\nabla_\theta J(\theta) = \sum_{(x,y)\in D} \nabla_\theta \text{loss}_\theta(x)$$

$$\nabla_\theta J(\theta) = \sum_{(s,a)\in D} \nabla_\theta \log \pi_\theta(a|s) r(s,a)$$

Issue of policy gradient

policy gradient

$$J(\theta) = \int_{Tra} p_\theta(\tau) R(\tau)\ \mathrm{d}\tau$$

black-box optimization with differentiable model

$$\mu, \sigma = \arg\max_{\mu,\sigma} E_{\theta \sim \mathcal{N}(\mu,\sigma)} J(\pi_\theta) = \arg\max_{\mu,\sigma} \int p(\theta; \mu, \sigma^2) J(\pi_\theta)\ \mathrm{d}\theta$$

policy gradient is more close to black-box optimization with a differentiable model, only with an MDP structure

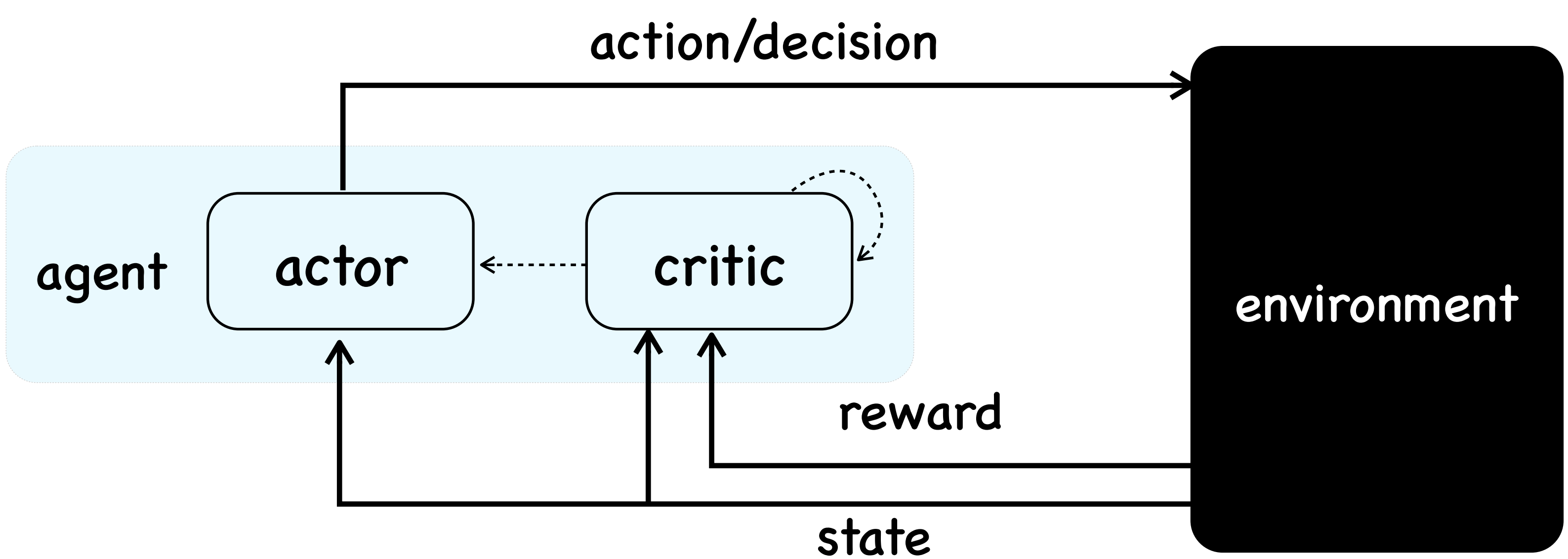
Reduce variance by critic: Actor-Critic

learn policy from trajectories high var. -- actor only

learn value functions low var. -- critic only

combine the two for the good of both:

use critic to stably estimate the return



[Grondman, et al. Bartlett. A Survey of Actor-Critic Reinforcement Learning:Standard and Natural Policy Gradients. IEEE Trans. SMC-C, 2012]
[Konda & Tsitsiklis. Actor-Critic Algorithms. NIPS'97]

Reduce variance by critic: Actor-Critic

Maintain another parameter vector w

$$Q_w(s,a) = w^\top \phi(s,a) \approx Q^\pi(s,a)$$

value-based function approximated methods to update Q_w

MC, TD, TD(λ), LSPI

Multi-step MDPs: $J(\theta) = \int_S d^{\pi_\theta}(s) \int_A \pi_\theta(a|s) Q^{\pi_\theta}(s,a)\ \mathrm{d}s\ \mathrm{d}a$

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

Policy Gradient Theorem

equivalent gradient for all objectives

[Sutton et al. Policy gradient methods for reinforcement learning with function approximation. NIPS'00]

$$\nabla_\theta J(\theta) \approx E[\nabla_\theta \log \pi_\theta(a|s) Q_w(s,a)]$$

if w is a minimizer of $E[(Q^{\pi_\theta}(s,a) - Q_w(s,a))^2]$

Learn policy (actor) and Q-value (critic) simultaneously

Example

initial state s

for i=0, 1, ...

$a = \pi_\epsilon(s)$

s', r = do action a

$a' = \pi_\epsilon(s')$

$\delta = r + \gamma Q_w(s', a') - Q_w(s, a)$

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

$w = w + \alpha \delta \phi(s, a)$

$s = s'$, $a = a'$

end for

Control variance by introducing a bias term

for any bias term b(s)

$$\int_S d^{\pi_\theta}(s) \nabla_\theta \int_A \pi_\theta(a|s) \qquad b(s)\, \mathrm{d}s\mathrm{d}a = 0$$

gradient with a bias term

$$\nabla_\theta J(\theta) = E[\nabla_\theta \log \pi_\theta(a|s)(Q^\pi(s,a) - b(s))]$$

obtain the bias by minimizing variance

obtain the bias by V(s)

advantage function: $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$

$$\nabla_\theta J(\theta) = E[\nabla_\theta \log \pi_\theta(a|s) A^\pi(s,a)]$$

learn policy, Q and V simultaneously

Policy search v.s. value function based

Policy search advantages:

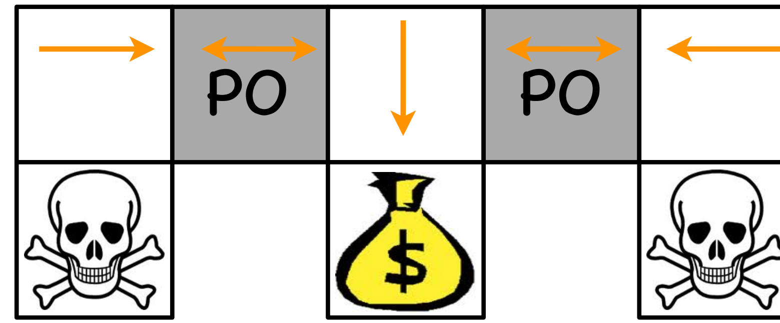
- effective in high-dimensional and continuous action space
- learn stochastic policies directly
- avoid policy degradation

disadvantages:

- converge only to a local optimum
- high variance

Example: Aliased gridworld

state PO cannot be distinguished
=> same action distribution



deterministic policy: stuck at one side

value function based policy is mostly deterministic

stochastic policy: either direction with prob. 0.5

policy search derives stochastic policies

adversarial games commonly require stochastic (mixed) policy