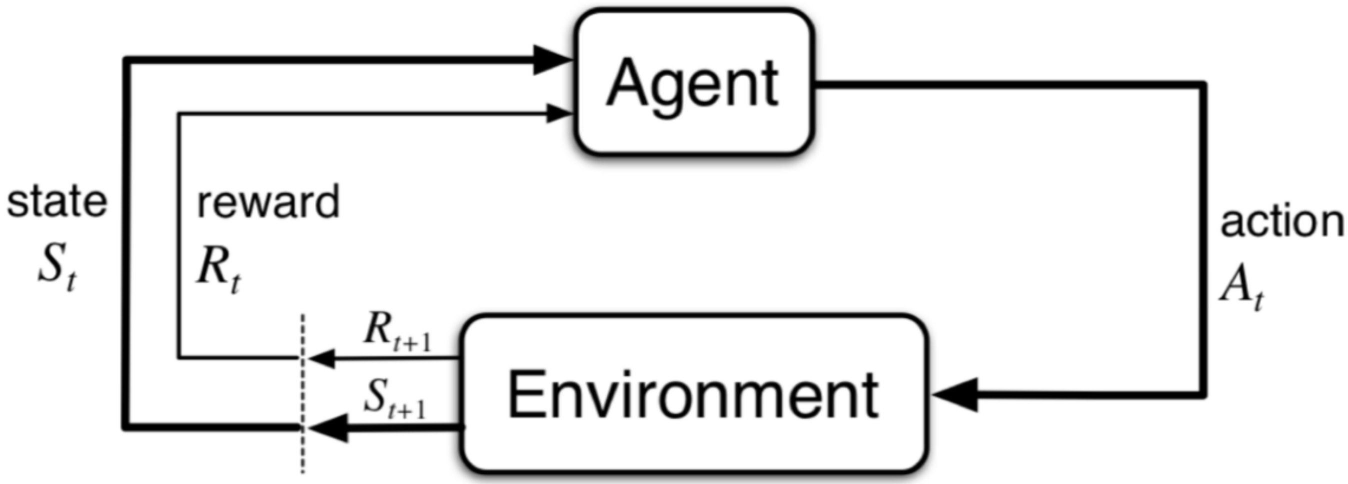



**Figure 3.1:** The agent–environment interaction in a Markov decision process.

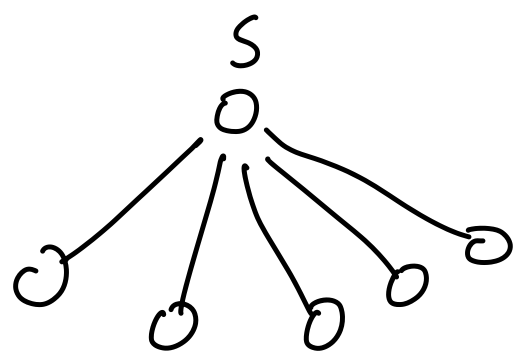
↳ Sutton, Barto; Reinforcement Learning

Four-argument function:

S - state set   A - action set   R - reward

$$p(s, a, s', r)$$

$$\sum_{\substack{s' \in S \\ r \in R}} p(s, a, s', r) = 1.$$



# Reward Hypothesis

That all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward).

RL is terrible, everything else is much worse. Andrej Karpathy

# Cumulative Reward

$0 < \gamma < 1$, discount factor

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1},$$

Note: For $0 < \gamma < 1$

$$1 + \gamma + \gamma^2 + \gamma^3 + \cdots = \frac{1}{1-\gamma}$$

Exercise: Show $G_t = R_{t+1} + \gamma G_{t+1}$

Policy

$\pi : S \to A$

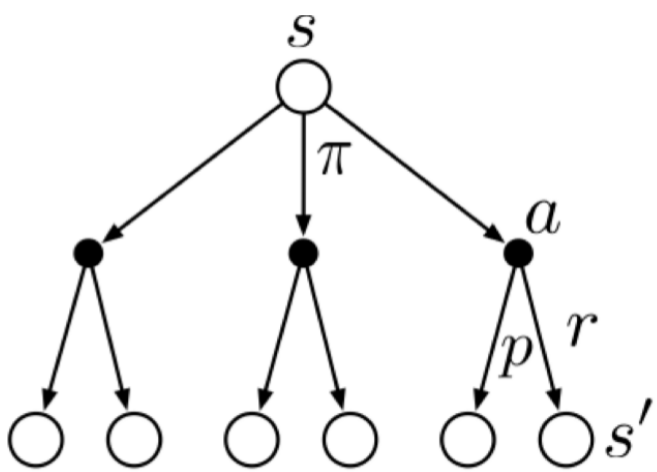
$$\sum_{a \in A} \pi(s,a) = 1$$

Value of a state when policy $\pi$ is fixed

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right], \text{ for all } s \in \mathcal{S},$$

Value of a state-action pair when π is fixed

$$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a\right]. \quad (3.13)$$




Backup diagram for $v_\pi$

Bellman Equations

$$v_\pi(s) = \sum_{a \in A} \pi(a, s) \cdot q_\pi(s, a)$$

$$v_\pi(s) = \sum_{a \in A} \pi(a, s) \cdot \left( \sum_{s'} p(s', r) \cdot r + \gamma v_\pi(s') \right)$$

Dfn: If $v_{\pi'}(s) \geq v_{\pi}(s)$ for all states $s \in S$ then $\pi'$ is a better policy than $\pi$.

Thm: If the dynamics is given by a finite MDP then there is a policy that assign maximal value to all states.

- This guy is usually denoted by $\pi^*$
- There is a unique maximum value for all $s \in S$ But there may be many optimal policies.

# Estimation and Control

Estimation Question: Given a policy $\pi$, can you estimate the value of each state $s \in S$?

Bellman Eqn: $v_\pi(s) = \sum_a \pi(a, s) \left( \sum_{s', r} p(s', r) \left( r + \gamma v_\pi(s') \right) \right)$

$$
\begin{aligned}
v_{k+1}(s) &\doteq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r \mid s, a) \Big[ r + \gamma v_k(s') \Big],
\end{aligned}
$$

Iterate until you can't :)

**Iterative Policy Evaluation, for estimating $V \approx v_\pi$**

Input $\pi$, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$ arbitrarily, for $s \in \mathcal{S}$, and $V(terminal)$ to 0

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r \,|\, s, a)\big[r + \gamma V(s')\big]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Control Question: Given a policy $\pi$ and value estimates $v_\pi(s)$ for all $s \in S$, how can you find a better policy $\pi'$?

Wishful Thinking: Suppose I found $\pi'$ such that

$$q_\pi(s, \pi'(s)) \geq v_\pi(s)$$

for all $s \in S$. Does this mean $\pi'$ is better than $\pi$?

Thm (Policy Improvement) Yes, $v_{\pi'}(s) \geq v_\pi(s)$ for all $s \in S$.

I'll go greedy then

$$\begin{aligned}
\pi'(s) &\doteq \arg\max_a q_\pi(s, a) \\
&= \arg\max_a \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a] \qquad (4.9) \\
&= \arg\max_a \sum_{s', r} p(s', r \mid s, a)\Big[r + \gamma v_\pi(s')\Big],
\end{aligned}$$

If your greedy iteration stops, then

$$
\begin{aligned}
v_{\pi'}(s) &= \max_a \mathbb{E}[R_{t+1} + \gamma v_{\pi'}(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \max_a \sum_{s',r} p(s', r \mid s, a)\Big[r + \gamma v_{\pi'}(s')\Big].
\end{aligned}
$$

**Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$**

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$; $V(terminal) \doteq 0$

2. Policy Evaluation
   Loop:
   $\quad \Delta \leftarrow 0$
   $\quad$ Loop for each $s \in \mathcal{S}$:
   $\quad\quad v \leftarrow V(s)$
   $\quad\quad V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s))\big[r + \gamma V(s')\big]$
   $\quad\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   *policy-stable* $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $\quad$ *old-action* $\leftarrow \pi(s)$
   $\quad \pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s', r \mid s, a)\big[r + \gamma V(s')\big]$
   $\quad$ If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

- Do we have to do the full estimation of values?
- Do we have to do full policy iteration?
- If there is more than one optimal policy, does the iteration converge?

# Value Iteration

- One step estimation + One step control

$$
\begin{aligned}
v_{k+1}(s) &\doteq \max_a \mathbb{E}[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \max_a \sum_{s',r} p(s', r \mid s, a)\Big[r + \gamma v_k(s')\Big],
\end{aligned}
$$

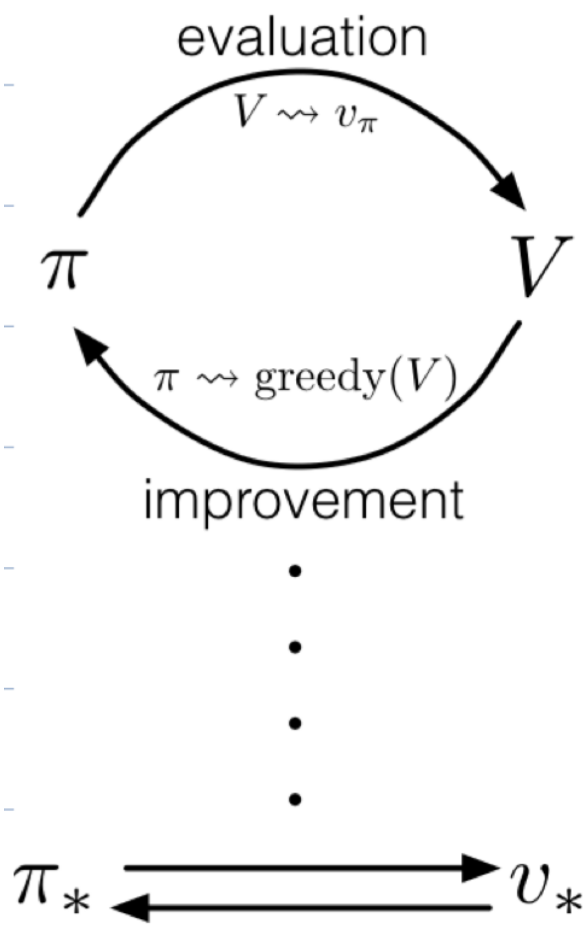
**Value Iteration, for estimating $\pi \approx \pi_*$**

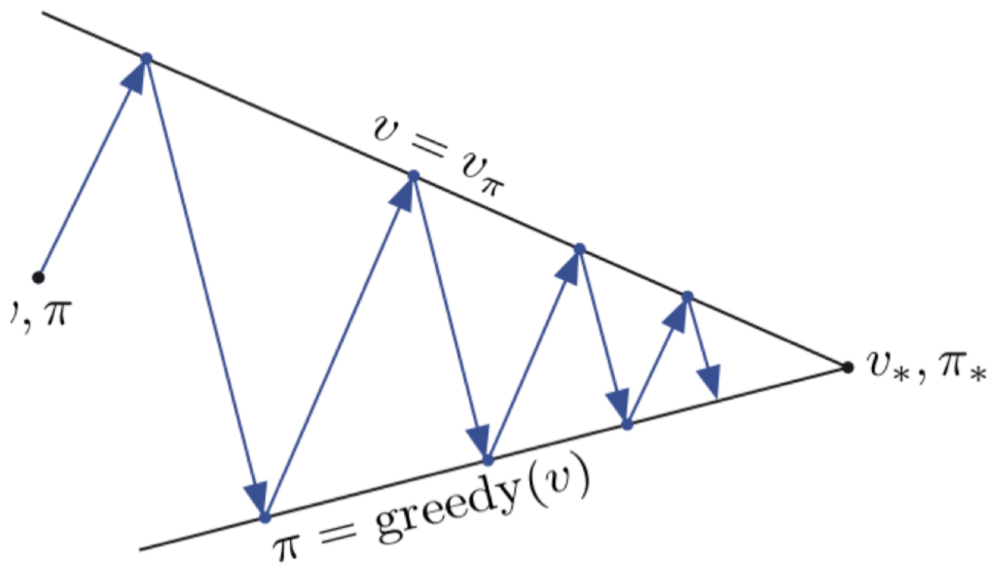
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\textit{terminal}) = 0$

Loop:
| $\Delta \leftarrow 0$
| Loop for each $s \in \mathcal{S}$:
| $v \leftarrow V(s)$
| $V(s) \leftarrow \max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma V(s')]$
| $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that
$\pi(s) = \arg\max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma V(s')]$

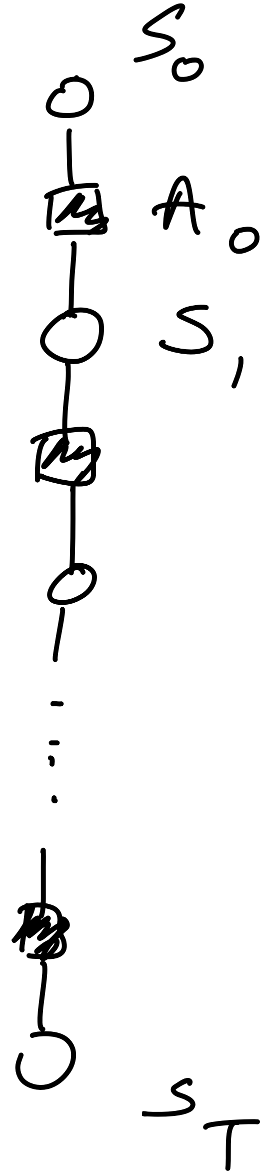
# Generalized Policy Iteration



# Monte-Carlo

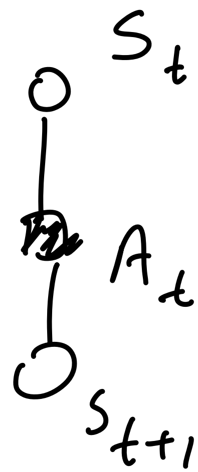
$S_0, A_0, R_0, S_1, A_1, R_1, \ldots, R_T, S_T$




- roll out the policy until termination
- for $t = T, T-1, \ldots, 0$

  $G = G + \gamma R_t$ returns
- $Q(S_0, A_0)$ = average returns
- $\pi(S_0) = \arg\max_a Q(S_0, a)$

# Temporal Difference

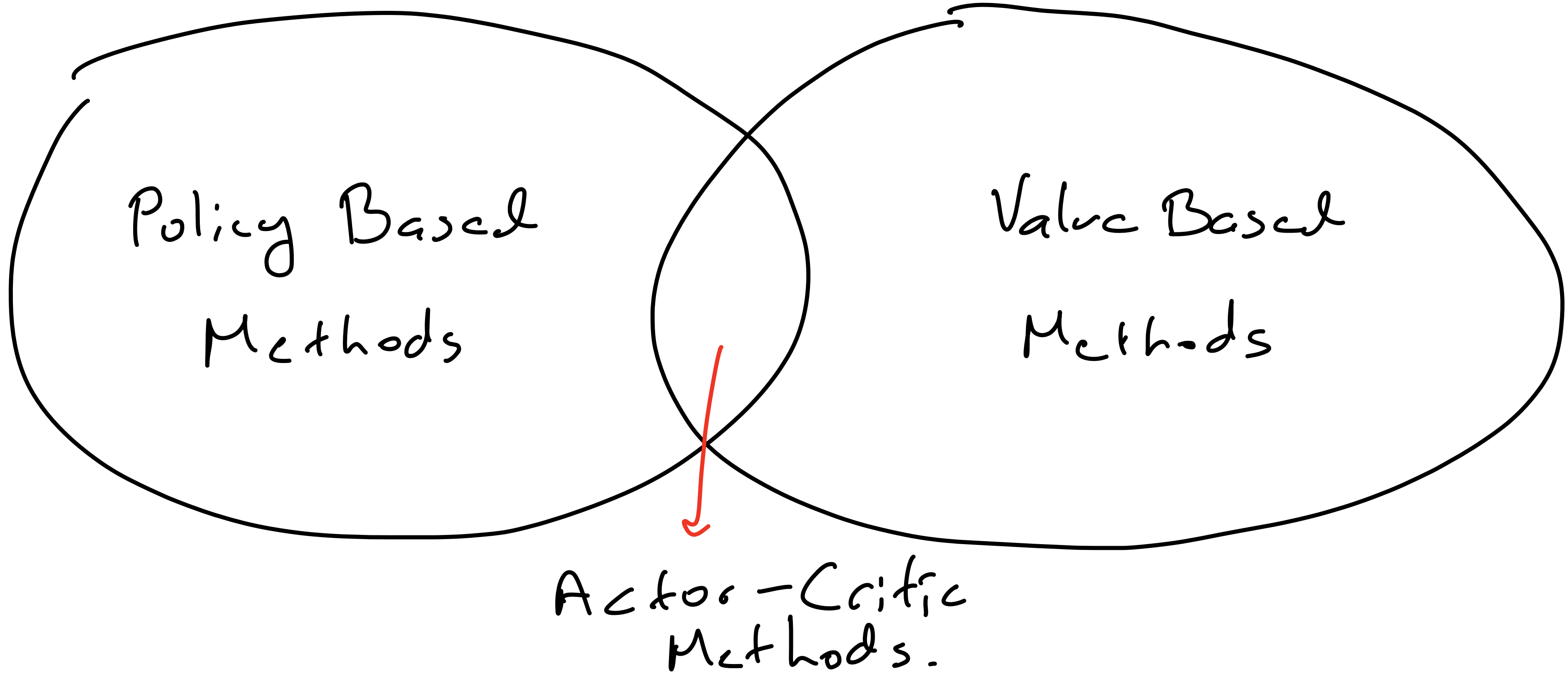



TD(0)

$$G_t = R_t + \gamma \cdot G_{t+1}$$

$$V(s_t) = V(s_t) + \alpha_t \left[ R_t + \gamma G_{t+1} - G_t \right]$$

- There is a more general framework that interpolates between MC to TD(0) called TD($\lambda$).




Intuition: Can we optimize the policy directly without learning the value function?

- $\theta$ parameter of policy n-net
- Assume a probability measure on state-space
- Average return of policy $\pi_\theta$ is

$$J(\theta) = \mathbb{E}_s \left[ \text{cumulative reward of } s \mid \pi_\theta \right]$$

- Goal :

$$\underset{\theta}{\text{arg-max}}\ J(\theta)$$

# Policy Gradient Theorem

- A trajectory $\tau$ : $s_0, a_0, R_0, s_1, a_1, R_1, \ldots, R_T$

$\pi_\theta(s, a) \rightarrow$ probability of taking action $a$ at state $s$ w.r.t $\pi_\theta$

$p(s, a, s', r) \rightarrow$ when action $a$ is taken the probability of going to $s'$ and reward $r$

$$\mathbb{P}(\tau) = \prod_{i=0}^{T} \pi_\theta(a_i) \cdot p(s_i, a_i, s_{i+1}, r_i)$$

# Policy Gradient Theorem

$$\log P(\tau) = \sum \log \pi_\theta(a_i) + \sum \log p(s_i, a_i, s_{i+1}, r_i)$$

$$\nabla_\theta \log P(\tau) = \underbrace{\sum}_{} \nabla_\theta \log \pi_\theta(a_i)$$

↳ no model of environment needed!

Derivation of result:

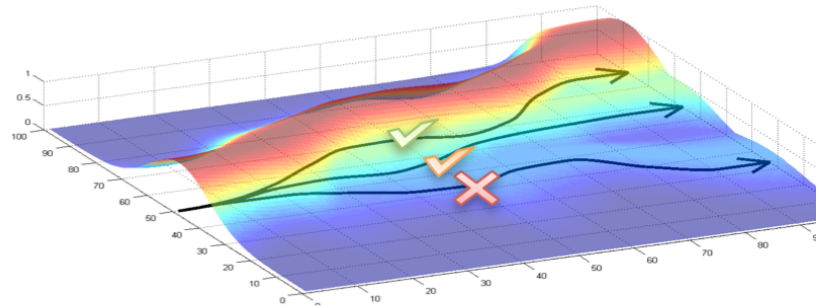
$$\nabla_\theta J(\theta) = \nabla_\theta \left( \sum_\tau \mathbb{P}(\tau) \cdot R(\tau) \right)$$

$$\nabla_\theta J(\theta) = \sum_\tau R(\tau) \cdot P(\tau) \cdot \nabla_\theta \log P(\tau)$$

So, we can estimate $\nabla_\theta J(\theta)$ directly.

- Sample $m$ paths $\tau_1, \tau_2, \ldots, \tau_m$
- $\nabla_\theta J(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} R(\tau_i) \cdot P(\tau_i) \cdot \nabla_\theta \log P(\tau_i)$
- $\theta = \theta + \alpha \nabla_\theta J(\theta)$

- Gradient tries to:
  - Increase probability of paths with positive R
  - Decrease probability of paths with negative R



**!** Likelihood ratio changes probabilities of experienced paths, does not try to change the paths (<-> Path Derivative)

Pieter Abbel

# REINFORCE

- Initialize $\theta$ randomly
- Sample and compute average gradient for $t = 1, 2, .., T$
- $\theta = \theta + \alpha_t \cdot \nabla J_t$

  learning rate

This is great, actually. The only problem is that it is too noisy.

# Baseline Trick

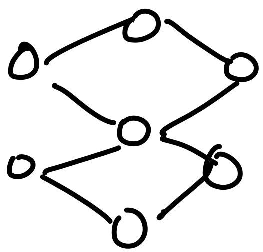
$$J(\theta) = \sum P(\tau) \cdot \left( R(\tau) - \underbrace{b}_{\text{anything except a function of } \theta} \right)$$

$$\nabla_\theta J = \sum P(\tau) \cdot R(\tau) \cdot \nabla_\theta \log P(\tau)$$

Options

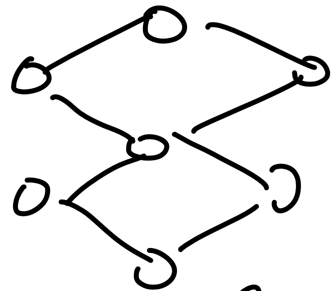
- sample, collect rewards, put b as average.
- Estimate $V_{\pi_\theta}(s)$ and use it for b.

# Actor-Critic Style



Actor

Policy brain



Valuefunction estimator

Critic of the action suggested by policy n-net

Policy Gradient + Generalized Advantage Estimation:

- Init $\pi_{\theta_0}$ $V^{\pi}_{\phi_0}$
- Collect roll-outs {s, u, s', r} and $\hat{Q}_i(s,u)$
- Update: $\phi_{i+1} \leftarrow \min_{\phi} \sum_{(s,u,s',r)} \|\hat{Q}_i(s,u) - V^{\pi}_{\phi}(s)\|_2^2 + \kappa\|\phi - \phi_i\|_2^2$

$$\theta_{i+1} \leftarrow \theta_i + \alpha \frac{1}{m} \sum_{k=1}^{m} \sum_{t=0}^{H-1} \nabla_\theta \log \pi_{\theta_i}(u_t^{(k)}|s_t^{(k)}) \left(\hat{Q}_i(s_t^{(k)}, u_t^{(k)}) - V^{\pi}_{\phi_i}(s_t^{(k)})\right)$$

DDPG → this works even with deterministic policies

TD3 → it works but too noisy, here is how to stabilize it.

GRPO → I'll go back to the days of REINFORCE