Trust Region Policy Optimization, Schulman et al, 2015.

옥찬호
utilForever@gmail.com

Introduction

Trust Region Policy Optimization, Schulman et al, 2015.

- Most algorithms for policy optimization can be classified…
  - (1) Policy iteration methods (Alternate between estimating the value function under the current policy and improving the policy)
  - (2) Policy gradient methods (Use an estimator of the gradient of the expected return)
  - (3) Derivative-free optimization methods (Treat the return as a black box function to be optimized in terms of the policy parameters) such as
    - The Cross-Entropy Method (CEM)
    - The Covariance Matrix Adaptation (CMA)

Introduction

- General derivative-free stochastic optimization methods such as CEM and CMA are preferred on many problems,
  - They achieve good results while being simple to understand and implement.
- For example, while Tetris is a classic benchmark problem for approximate dynamic programming (ADP) methods, stochastic optimization methods are difficult to beat on this task.

Introduction

Trust Region Policy Optimization, Schulman et al, 2015.

- For continuous control problems, methods like CMA have been successful at learning control policies for challenging tasks like locomotion when provided with hand-engineered policy classes with low-dimensional parameterizations.
- The inability of ADP and gradient-based methods to consistently beat gradient-free random search is unsatisfying,
  - Gradient-based optimization algorithms enjoy much better sample complexity guarantees than gradient-free methods.

Introduction

Trust Region Policy Optimization, Schulman et al, 2015.

- Continuous gradient-based optimization has been very successful at learning function approximators for supervised learning tasks with huge numbers of parameters, and extending their success to reinforcement learning would allow for efficient training of complex and powerful policies.

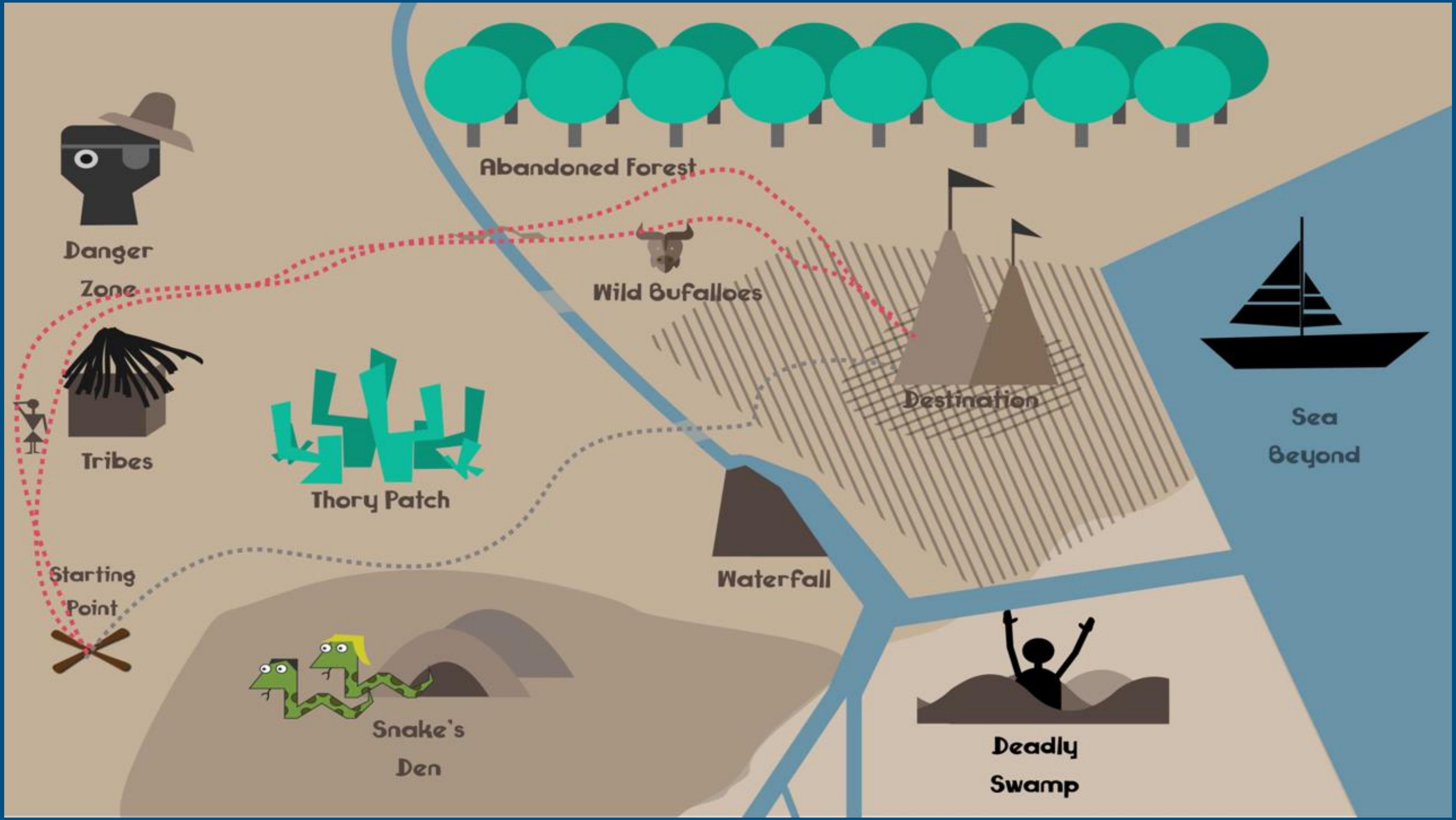
Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

- Markov Decision Process (MDP) : $(\mathcal{S}, \mathcal{A}, P, r, \rho_0, \gamma)$
  - $\mathcal{S}$ is a finite set of states
  - $\mathcal{A}$ is a finite set of actions
  - $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the transition probability
  - $r : \mathcal{S} \to \mathbb{R}$ is the reward function
  - $\rho_0 : \mathcal{S} \to \mathbb{R}$ is the distribution of the initial state s_0
  - $\gamma \in (0, 1)$ is the discount factor
- Policy
  - $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$

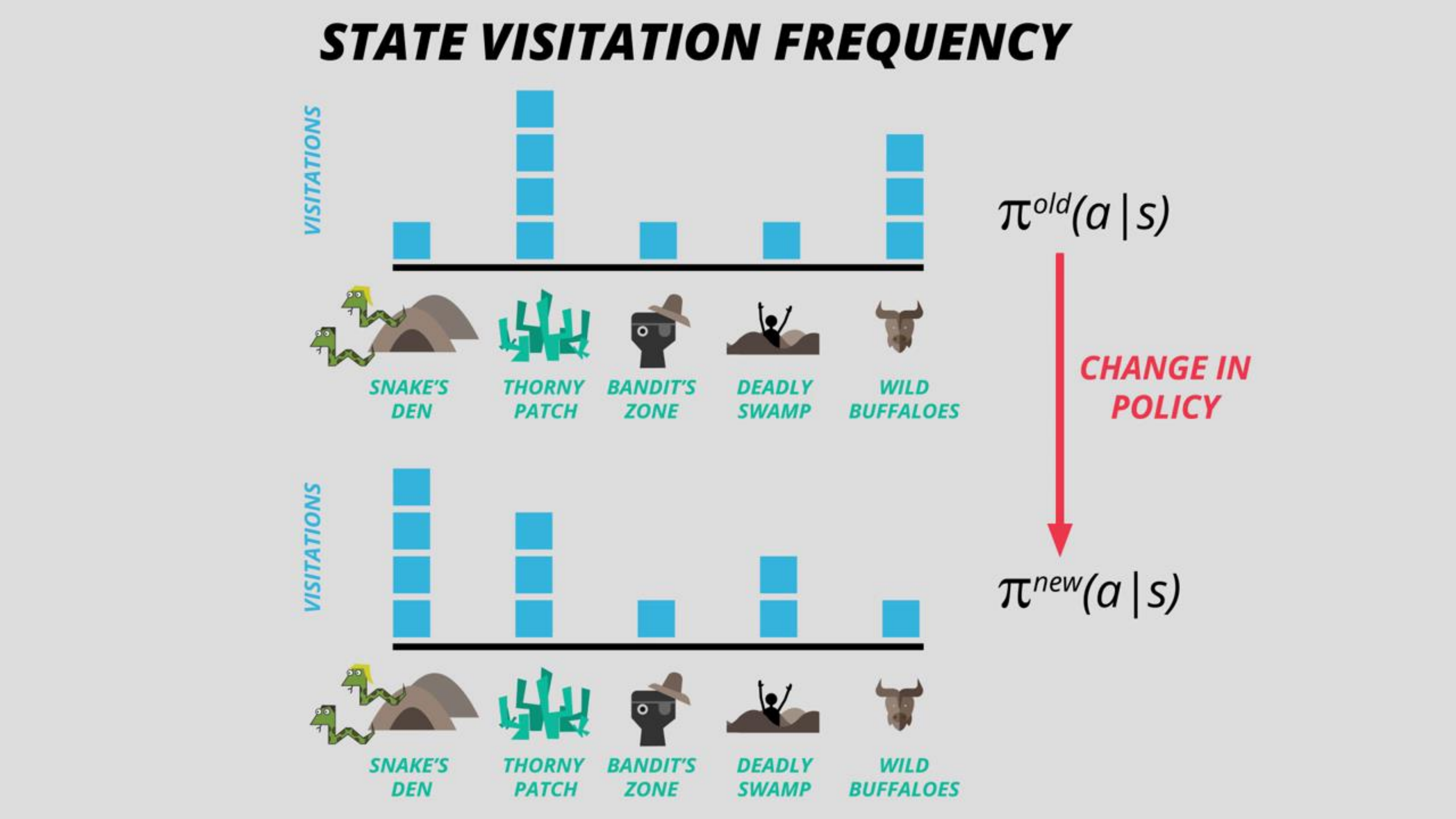
Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.



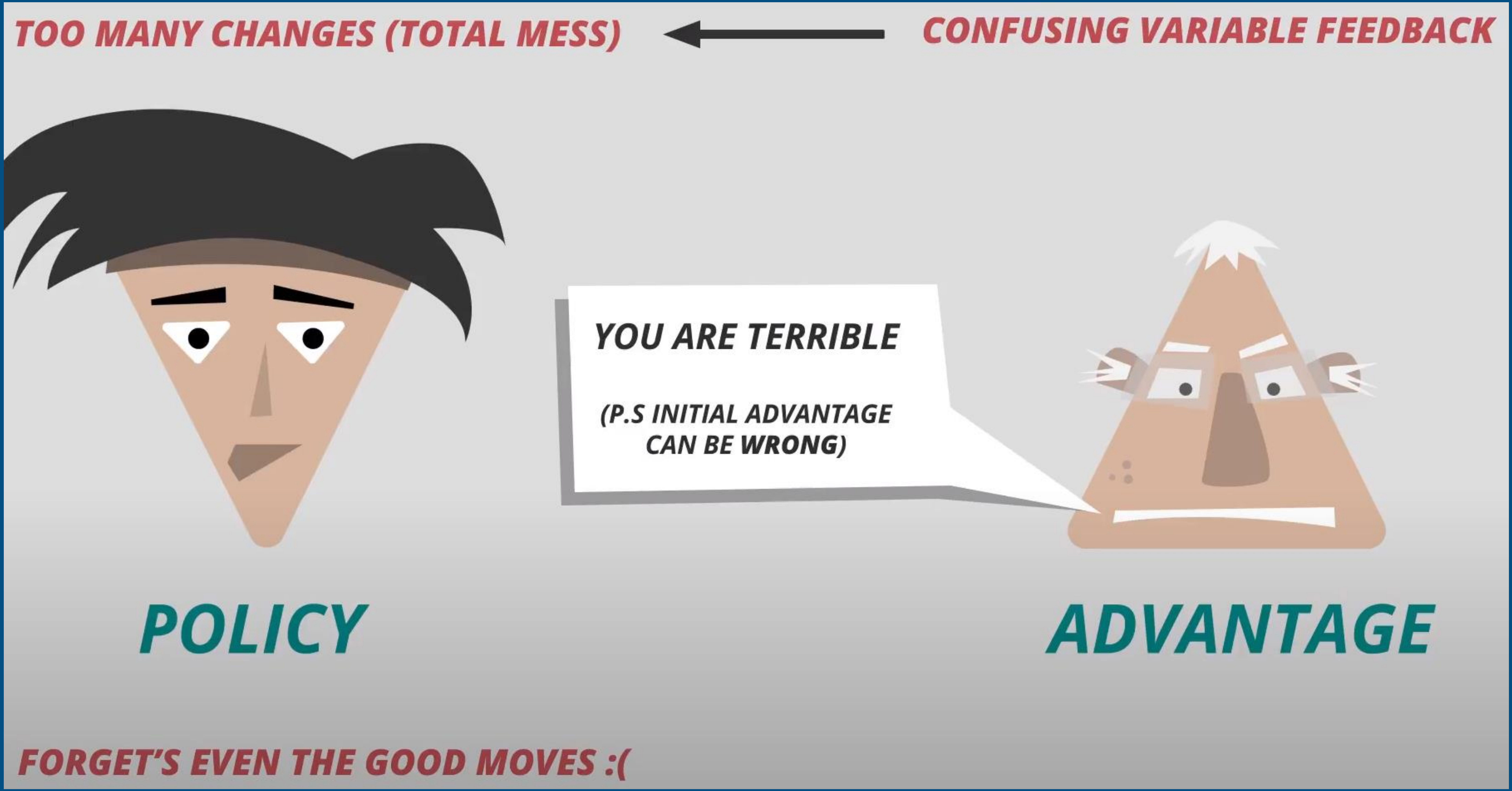
Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.



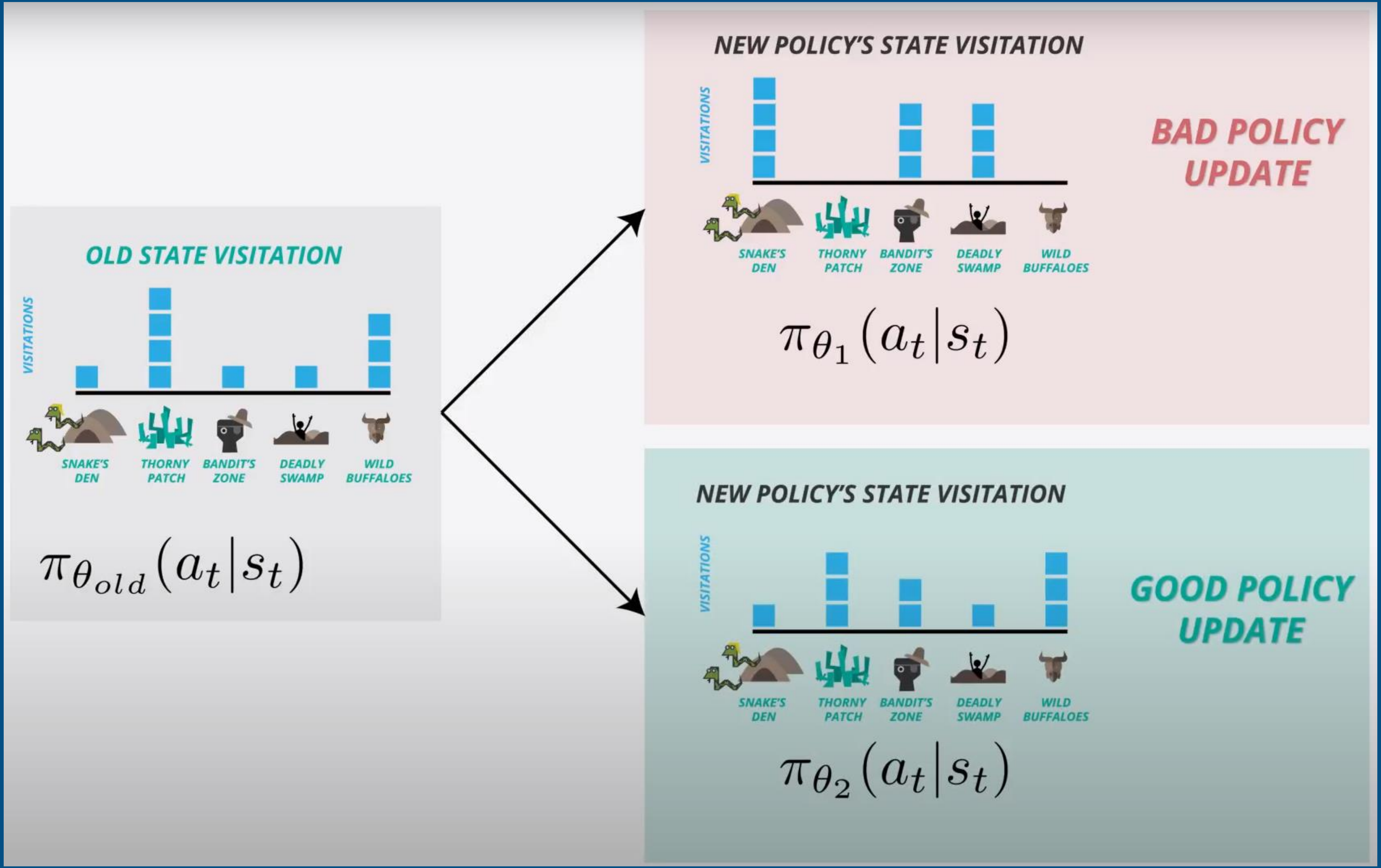
Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.



Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.



Preliminaries

How to measure policy's performance?

Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

- Expected cumulative discounted reward

$$\eta(\pi) = \mathbb{E}_{s_0, a_0, \ldots}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)\right], \text{ where}$$

$$s_0 \sim \rho_0(s_0),\ a_t \sim \pi(a_t|s_t),\ s_{t+1} \sim P(s_{t+1}|s_t, a_t).$$

- Action value, Value/Advantage functions

$$Q_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}, a_{t+1}, \ldots}\left[\sum_{l=0}^{\infty} \gamma^l r(s_{t+l})\right],$$

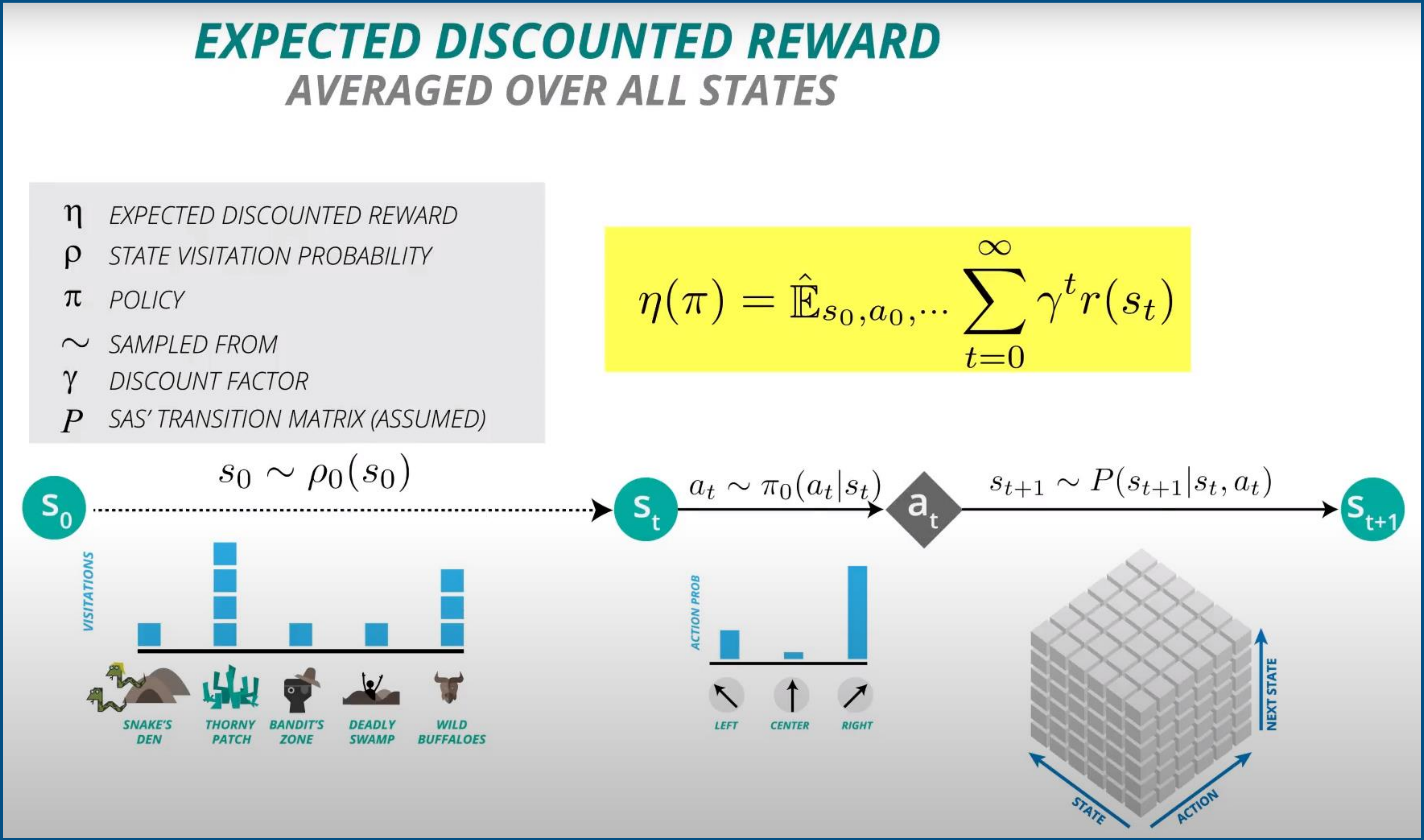
$$V_\pi(s_t) = \mathbb{E}_{a_t, s_{t+1}, \ldots}\left[\sum_{l=0}^{\infty} \gamma^l r(s_{t+l})\right],$$

$$A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s), \text{ where}$$

$$a_t \sim \pi(a_t|s_t), s_{t+1} \sim P(s_{t+1}|s_t, a_t) \text{ for } t \geq 0.$$

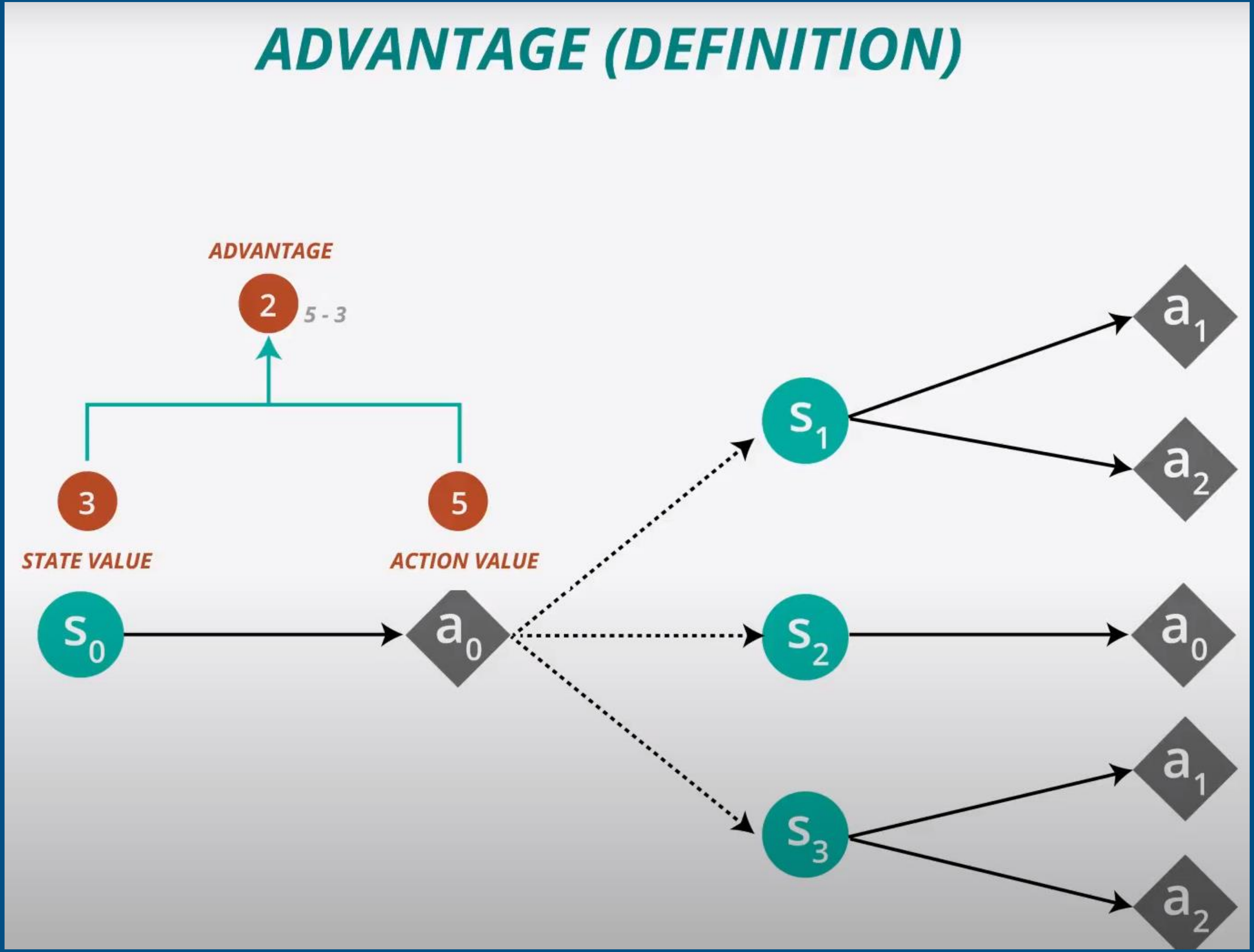
Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.



Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.



Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

- Lemma 1. Kakade & Langford (2002)

$$\eta(\tilde{\pi}) = \eta(\pi) + \mathbb{E}_{s_0, a_0, \cdots \sim \tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t A_\pi(s_t, a_t)\right]$$

- Proof

$$\begin{aligned}
&\mathbb{E}_{\tau|\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t A_\pi(s_t, a_t)\right] \\
&= \mathbb{E}_{\tau|\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t (r(s_t) + \gamma V_\pi(s_{t+1}) - V_\pi(s_t))\right] \\
&= \mathbb{E}_{\tau|\tilde{\pi}}\left[-V_\pi(s_0) + \sum_{t=0}^{\infty} \gamma^t r(s_t)\right] \\
&= -\mathbb{E}_{s_0}\left[V_\pi(s_0)\right] + \mathbb{E}_{\tau|\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)\right] \\
&= -\eta(\pi) + \eta(\tilde{\pi})
\end{aligned}$$

$$\begin{aligned}
&E_{\tau|\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t A_\pi(s_t, a_t)\right] \\
&= E_{\tau|\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t (r(s_t) + \gamma V_\pi(s_{t+1}) - V_\pi(s_t))\right] \\
&= E_{\tau|\tilde{\pi}}\left[\left(\sum_{t=0}^{\infty} \gamma^t r(s_t)\right) + \gamma V_\pi(s_1) - V_\pi(s_0) + \gamma^2 V_\pi(s_2) - \gamma V_\pi(s_1) + \gamma^3 V_\pi(s_3) - \gamma^2 V_\pi(s_2) + \cdots\right] \\
&= E_{\tau|\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t) + \gamma V_\pi(s_1) - V_\pi(s_0) + \gamma^2 V_\pi(s_2) - \gamma V_\pi(s_1) + \cdots\right] \\
&= E_{\tau|\tilde{\pi}}\left[-V_\pi(s_0) + \sum_{t=0}^{\infty} \gamma^t r(s_t)\right] \\
&\overset{(a)}{=} -E_{s_0}\left[V_\pi(s_0)\right] + E_{\tau|\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)\right] \\
&= -\eta(\pi) + \eta(\tilde{\pi}) \\
&\therefore \eta(\tilde{\pi}) = \eta(\pi) + E_{s_0, a_0, \ldots \sim \tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t A_\pi(s_t, a_t)\right]
\end{aligned}$$

Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

- Discounted (unnormalized) visitation frequencies

$$\rho_\pi(s) = P(s_0 = s) + \gamma P(s_1 = s) + \gamma^2 P(s_2 = s) + \dots,$$

- Rewrite Lemma 1

$$\begin{aligned}
\eta(\tilde{\pi}) &= \eta(\pi) + \sum_{t=0}^{\infty} \sum_s P(s_t = s|\tilde{\pi}) \sum_a \tilde{\pi}(a|s)\gamma^t A_\pi(s,a) \\
&= \eta(\pi) + \sum_s \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\tilde{\pi}) \sum_a \tilde{\pi}(a|s) A_\pi(s,a) \\
&= \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a).
\end{aligned}$$

Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

- This equation implies that any policy update $\pi \rightarrow \tilde{\pi}$ that has a nonnegative expected advantage at every state s, i.e., $\sum_a \tilde{\pi}(a|s) A_\pi(s,a) \geq 0$, is guaranteed to increase the policy performance η, or leave it constant in the case that the expected advantage is zero everywhere.

Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

OLD EXPECTED RETURN

+ve

$$\eta(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$

NEW EXPECTED RETURN

Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

- However, in the approximate setting, it will typically be unavoidable, due to estimation and approximation error, that there will be some states s for which the expected advantage is negative, that is, $\sum_a \tilde{\pi}(a|s) A_\pi(s,a) < 0$.
- The complex dependency of $\rho_{\tilde{\pi}}(s)$ on $\tilde{\pi}$ makes equation difficult to optimize directly.

Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

- Instead, we introduce the following local approximation to η:

$$L_\pi(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a).$$

- Note that L_π uses the visitation frequency ρ_π rather than $\rho_{\tilde{\pi}}$, ignoring changes in state visitation density due to changes in the policy.

Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

ACTUAL UPDATE

$$\eta(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$

APPROXIMATION USED

$$L_\pi(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$

Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

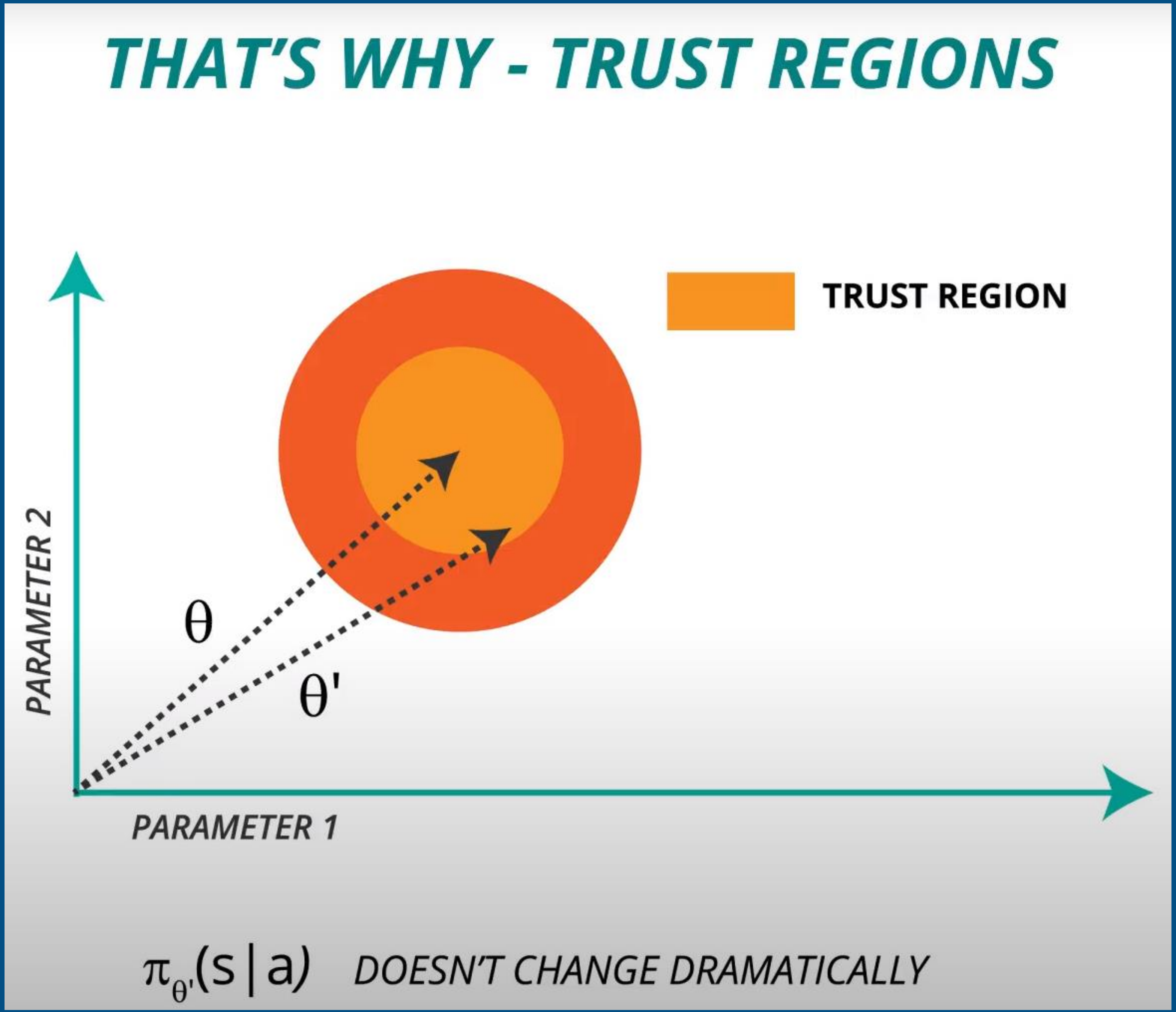
- However, if we have a parameterized policy π_θ, where $\pi_\theta(a|s)$ is a differentiable function of the parameter vector θ, then L_π matches η to first order.

$$L_{\pi_{\theta_0}}(\pi_{\theta_0}) = \eta(\pi_{\theta_0}),$$
$$\nabla_\theta L_{\pi_{\theta_0}}(\pi_\theta)\big|_{\theta=\theta_0} = \nabla_\theta \eta(\pi_\theta)\big|_{\theta=\theta_0}.$$

- This equation implies that a sufficiently small step $\pi_{\theta_0} \rightarrow \tilde{\pi}$ that improves $L_{\pi_{\theta_{old}}}$ will also improve η, but does not give us any guidance on how big of a step to take.

Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

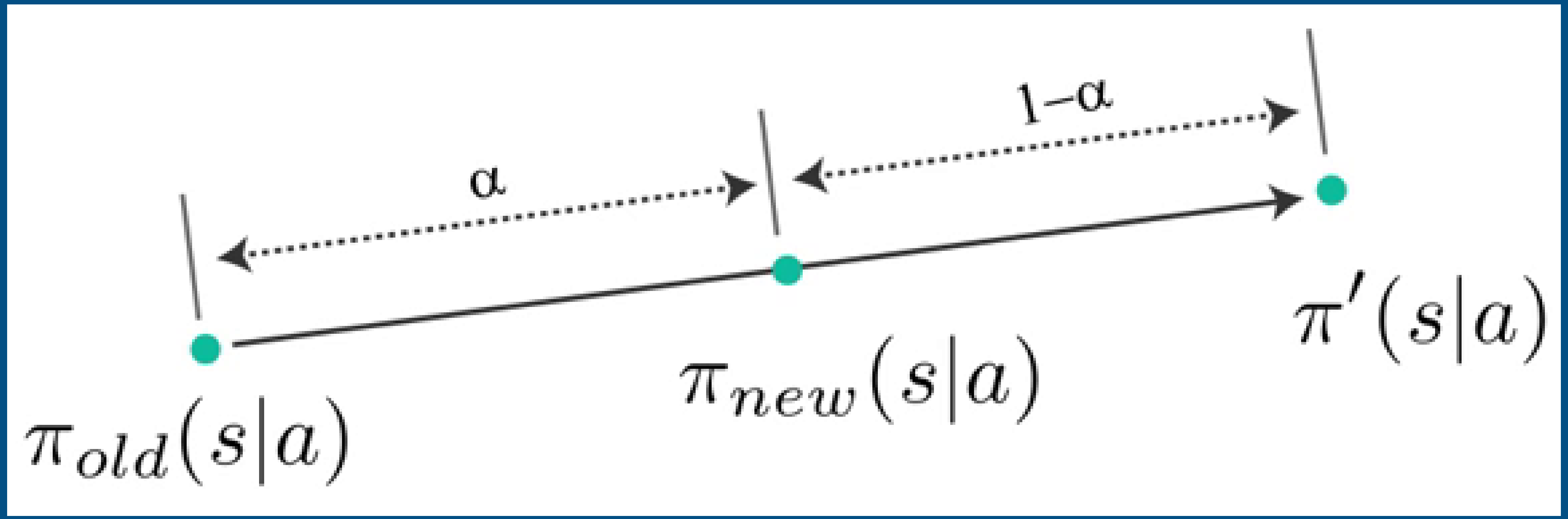


Preliminaries

Trust Region Policy Optimization, Schulman et al, 2015.

- Conservative Policy Iteration
  - Provide explicit lower bounds on the improvement of η.
  - Let π_{old} denote the current policy, and let $\pi' = \text{argmax}_{\pi'} L_{\pi_{old}}(\pi')$.
    The new policy π_{new} was defined to be the following texture:

$$\pi_{\text{new}}(a|s) = (1 - \alpha)\pi_{\text{old}}(a|s) + \alpha\pi'(a|s).$$



Preliminaries

- Conservative Policy Iteration
  - The following lower bound:

$$\eta(\pi_{\text{new}}) \geq L_{\pi_{\text{old}}}(\pi_{\text{new}}) - \frac{2\epsilon\gamma}{(1-\gamma)^2}\alpha^2$$
$$\text{where } \epsilon = \max_s \left| \mathbb{E}_{a\sim\pi'(a|s)}\left[A_\pi(s,a)\right] \right|.$$

  - However, this policy class is unwieldy and restrictive in practice, and it is desirable for a practical policy update scheme to be applicable to all general stochastic policy classes.

Monotonic Improvement Guarantee

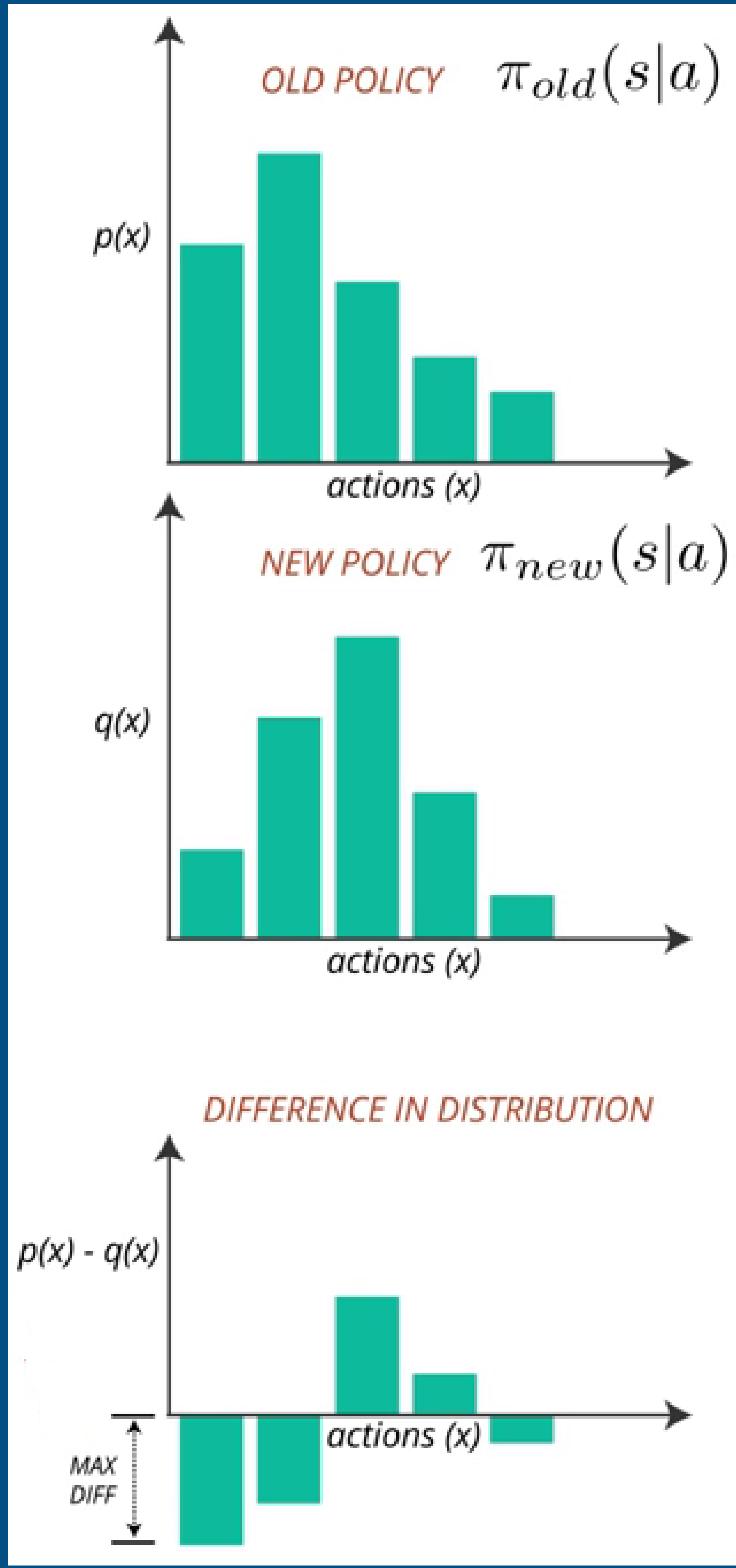
- The policy improvement bound in above equation can be extended to general stochastic policies, by
  - (1) Replacing α with a distance measure between π and $\tilde{\pi}$
  - (2) Changing the constant ϵ appropriately

Monotonic Improvement Guarantee

- The particular distance measure we use is the total variation divergence for discrete probability distributions p, q:

$$D_{\mathrm{TV}}(p \parallel q) = \frac{1}{2}\sum_i |p_i - q_i|$$

- $D_{\mathrm{TV}}^{\max}(\pi, \tilde{\pi}) = \max_s D_{TV}(\pi(\cdot|s) \parallel \tilde{\pi}(\cdot|s)).$



Monotonic Improvement Guarantee

- Theorem 1. Let $\alpha = D_{\mathrm{TV}}^{\max}(\pi_{\mathrm{old}}, \pi_{\mathrm{new}})$.
  Then the following bound holds:

$$\eta(\pi_{\mathrm{new}}) \geq L_{\pi_{\mathrm{old}}}(\pi_{\mathrm{new}}) - \frac{4\epsilon\gamma}{(1-\gamma)^2}\alpha^2$$

$$\textit{where } \epsilon = \max_{s,a} |A_\pi(s,a)|$$

Monotonic Improvement Guarantee

- We note the following relationship between the total variation divergence and the KL divergence.

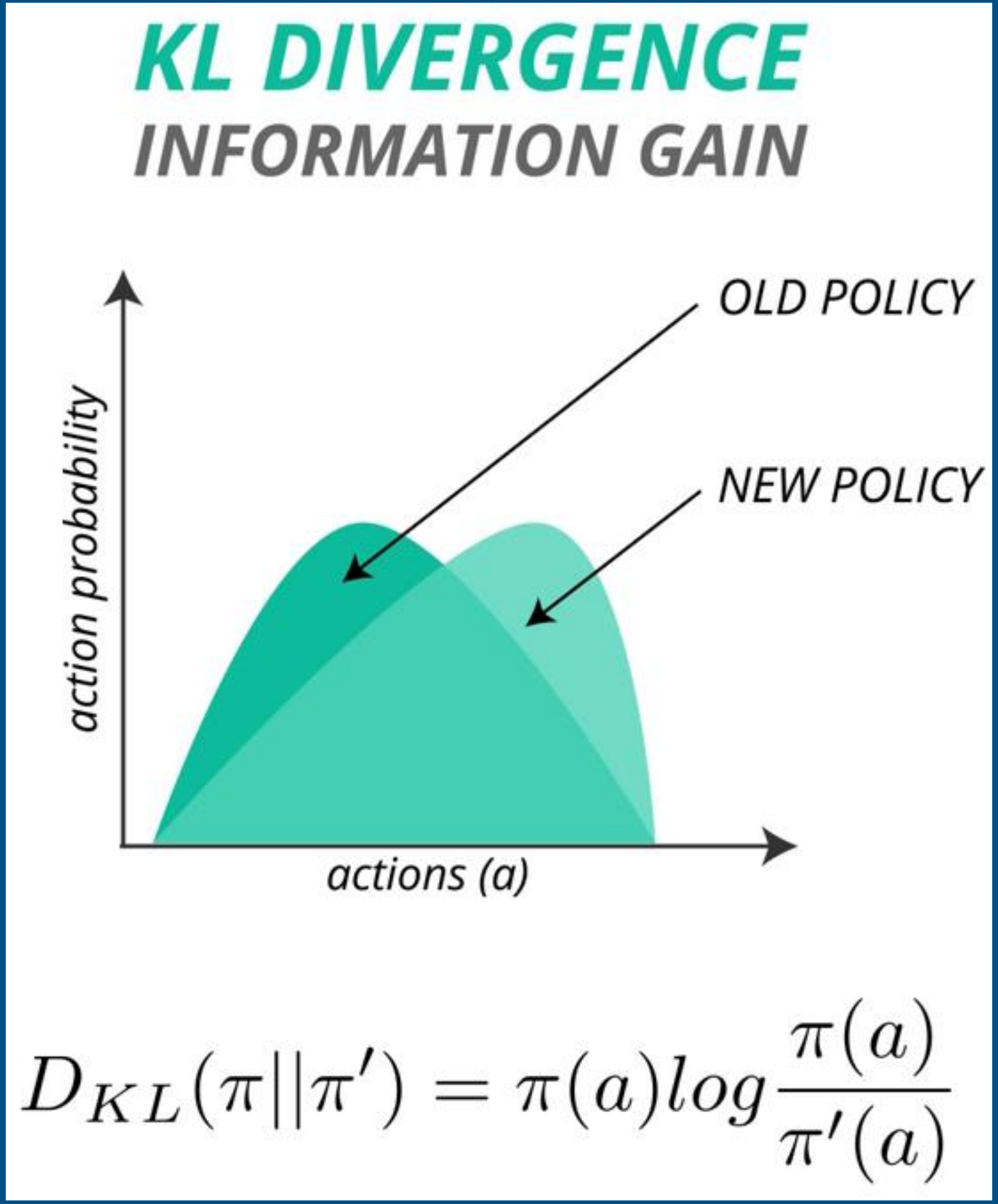
$$D_{\mathrm{TV}}(p \parallel q)^2 \le D_{\mathrm{KL}}(p \parallel q)$$

- Let $D_{\mathrm{KL}}^{\max}(\pi, \tilde{\pi}) = \max_s D_{\mathrm{KL}}\big(\pi(\cdot\,|s) \parallel \tilde{\pi}(\cdot\,|s)\big)$.

  The following bound then follows directly from Theorem 1:

$$\eta(\tilde{\pi}) \ge L_\pi(\tilde{\pi}) - C D_{\mathrm{KL}}^{\max}(\pi, \tilde{\pi}),$$
$$\text{where } C = \frac{4\epsilon\gamma}{(1-\gamma)^2}.$$

Monotonic Improvement Guarantee



Monotonic Improvement Guarantee

- Algorithm 1

Algorithm 1 Policy iteration algorithm guaranteeing non-decreasing expected return η

Initialize π_0.

for $i = 0, 1, 2, \ldots$ until convergence **do**

Compute all advantage values $A_{\pi_i}(s, a)$.

Solve the constrained optimization problem

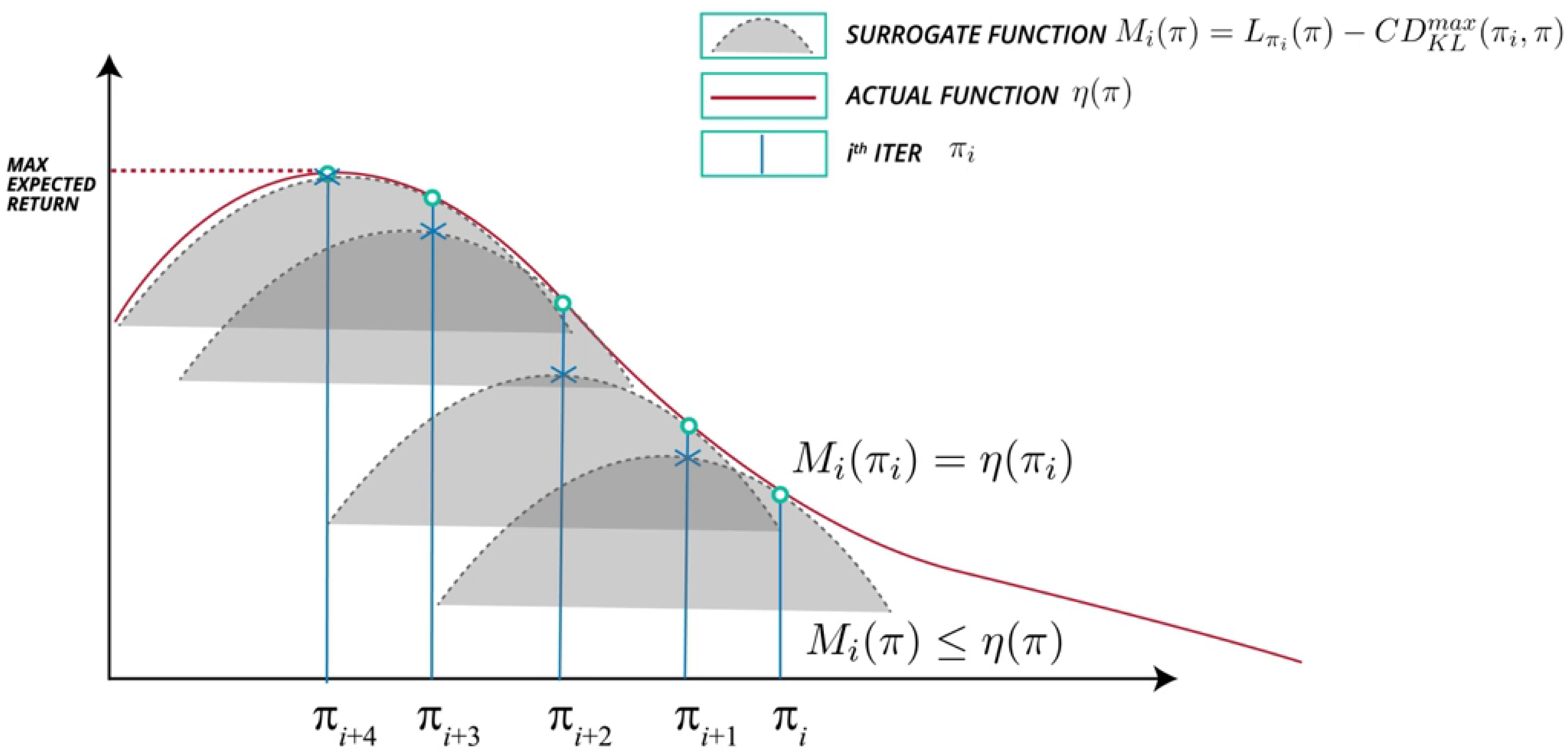
$$\pi_{i+1} = \arg\max_{\pi} \left[L_{\pi_i}(\pi) - C D_{\mathrm{KL}}^{\max}(\pi_i, \pi)\right]$$

where $C = 4\epsilon\gamma/(1-\gamma)^2$

and $L_{\pi_i}(\pi) = \eta(\pi_i) + \sum_s \rho_{\pi_i}(s) \sum_a \pi(a|s) A_{\pi_i}(s, a)$

end for

Monotonic Improvement Guarantee



Monotonic Improvement Guarantee

- Algorithm 1 is guaranteed to generate a monotonically improving sequence of policies $\eta(\pi_0) \leq \eta(\pi_1) \leq \eta(\pi_2) \leq \cdots$.
  - Let $M_i(\pi) = L_{\pi_i} - C D_{\mathrm{KL}}^{\max}(\pi_i, \pi)$. Then,

    $\eta(\pi_{i+1}) \geq M_i(\pi_{i+1})$ by Equation (9)

    $\eta(\pi_i) = M_i(\pi_i)$, therefore,

    $\eta(\pi_{i+1}) - \eta(\pi_i) \geq M_i(\pi_{i+1}) - M(\pi_i)$.
  - Thus, by maximizing M_i at each iteration, we guarantee that the true objective η is non-decreasing.
  - This algorithm is a type of minorization-maximization (MM) algorithm.

Optimization of Parameterized Policies

- Overload our previous notation to use functions of θ.
  - $\eta(\theta) \coloneqq \eta(\pi_\theta)$
  - $L_\theta(\tilde{\theta}) \coloneqq L_{\pi_\theta}(\pi_{\tilde{\theta}})$
  - $D_{\mathrm{KL}}(\theta \parallel \tilde{\theta}) \coloneqq D_{\mathrm{KL}}(\pi_\theta \parallel \pi_{\tilde{\theta}})$
  - Use θ_{old} to denote the previous policy parameters

Optimization of Parameterized Policies

- The preceding section showed that

$$\eta(\theta) \geq L_{\theta_{\text{old}}}(\theta) - C D_{\text{KL}}^{\max}(\theta_{\text{old}}, \theta)$$

- Thus, by performing the following maximization, we are guaranteed to improve the true objective η:

$$\text{maximize}_{\theta}\left[L_{\theta_{\text{old}}}(\theta) - C D_{\text{KL}}^{\max}(\theta_{\text{old}}, \theta)\right]$$

Optimization of Parameterized Policies

- In practice, if we used the penalty coefficient C recommended by the theory above, the step sizes would be very small.
- One way to take largest steps in a robust way is to use a constraint on the KL divergence between the new policy and the old policy, i.e., a trust region constraint:

$$\begin{aligned}&\underset{\theta}{\text{maximize}}\, L_{\theta_{\text{old}}}(\theta)\\&\text{subject to } D_{\text{KL}}^{\max}(\theta_{\text{old}}, \theta) \leq \delta.\end{aligned}$$

Optimization of Parameterized Policies

- This problem imposes a constraint that the KL divergence is bounded at every point in the state space. While it is motivated by the theory, this problem is impractical to solve due to the large number of constraints.
- Instead, we can use a heuristic approximation which considers the average KL divergence:

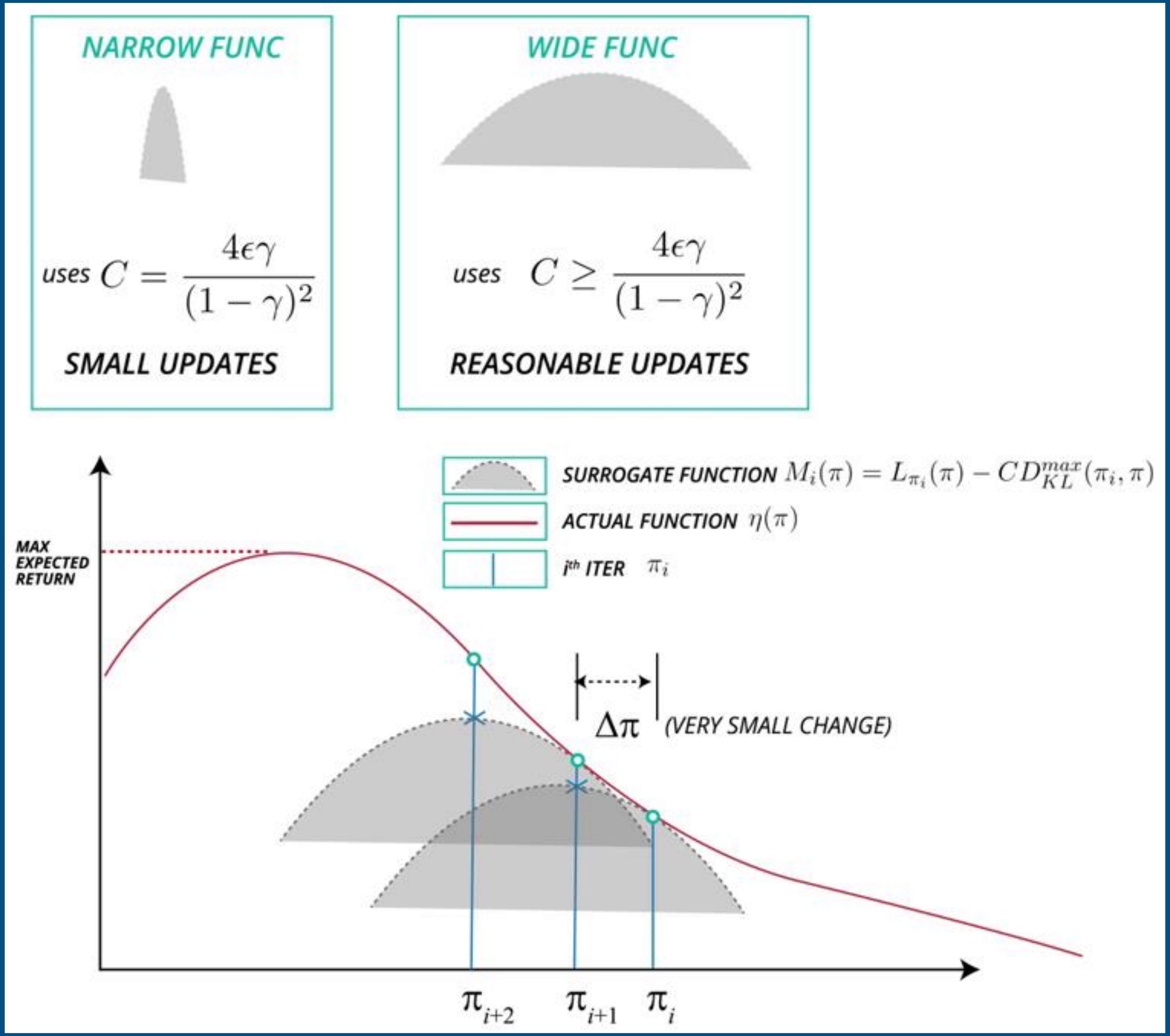
$$\overline{D}_{\mathrm{KL}}^{\rho}(\theta_1, \theta_2) := \mathbb{E}_{s \sim \rho}\left[D_{\mathrm{KL}}(\pi_{\theta_1}(\cdot|s) \parallel \pi_{\theta_2}(\cdot|s))\right].$$

Optimization of Parameterized Policies

- We therefore propose solving the following optimization problem to generate a policy update:

$$\begin{aligned} &\underset{\theta}{\text{maximize}}\, L_{\theta_{\text{old}}}(\theta) \\ &\text{subject to } \overline{D}_{\text{KL}}^{\rho_{\theta_{\text{old}}}}(\theta_{\text{old}}, \theta) \le \delta. \end{aligned}$$

Optimization of Parameterized Policies



Sample-Based Estimation

- How the objective and constraint functions can be approximated using Monte Carlo simulation?
- We seek to solve the following optimization problem, obtained by expanding $L_{\theta_{\text{old}}}$ in previous equation:

$$\begin{aligned} &\underset{\theta}{\text{maximize}} \sum_s \rho_{\theta_{\text{old}}}(s) \sum_a \pi_\theta(a|s) A_{\theta_{\text{old}}}(s,a) \\ &\qquad\qquad \text{subject to } \overline{D}_{\text{KL}}^{\rho_{\theta_{\text{old}}}}(\theta_{\text{old}}, \theta) \leq \delta. \end{aligned}$$

Sample-Based Estimation

- We replace
  - $\sum_s \rho_{\theta_{\text{old}}}(s)[\dots] \rightarrow \frac{1}{1-\gamma} E_{s\sim\rho_{\theta_{\text{old}}}}[\dots]$
  - $A_{\theta_{\text{old}}} \rightarrow Q_{\theta_{\text{old}}}$
  - $\sum_a \pi_{\theta_{\text{old}}}(a|s) A_{\theta_{\text{old}}} \rightarrow E_{a\sim q}\left[\frac{\pi_\theta(a|s)}{q(a|s)} A_{\theta_{\text{old}}}\right]$

    (We replace the sum over the actions by an importance sampling estimator.)

Sample-Based Estimation

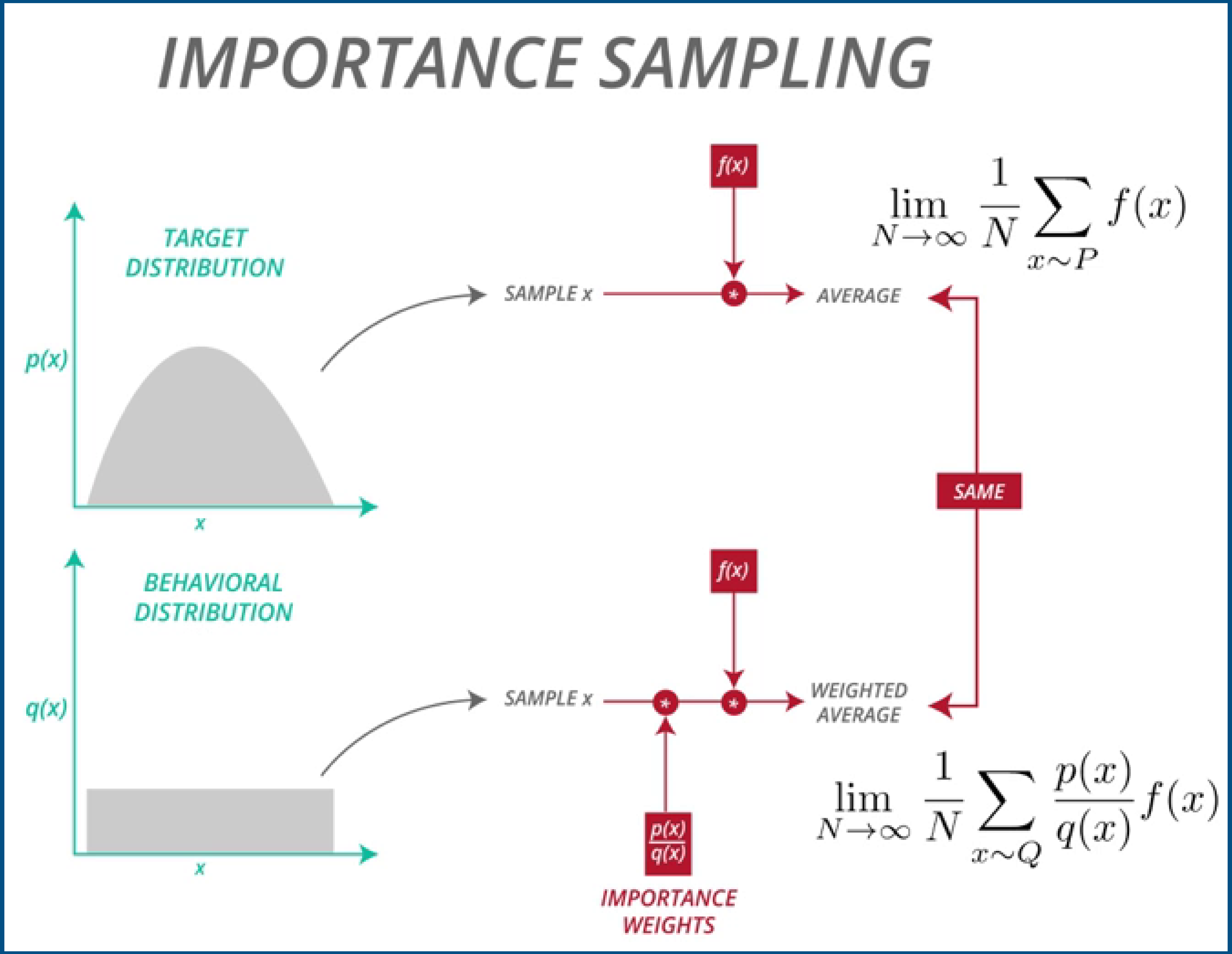
$$\underset{\theta}{\text{maximize}} \sum_s \rho_{\theta_{\text{old}}}(s) \sum_a \pi_\theta(a|s) A_{\theta_{\text{old}}}(s,a)$$

APPROXIMATE STATE VISITIATION
VIA SAMPLING

$$\underset{\theta}{\text{maximize}} \, \mathbb{E}_{s\sim\rho_{\theta_{\text{old}}}, a\sim q}\left[\frac{\pi_\theta(a|s)}{q(a|s)} Q_{\theta_{\text{old}}}(s,a)\right]$$

$$\sum_a \pi_\theta(a|s_n) A_{\theta_{\text{old}}}(s_n,a) = \mathbb{E}_{a\sim q}\left[\frac{\pi_\theta(a|s_n)}{q(a|s_n)} A_{\theta_{\text{old}}}(s_n,a)\right]$$

Sample-Based Estimation



Sample-Based Estimation

- Now, our optimization problem in previous equation is exactly equivalent to the following one, written in terms of expectations:

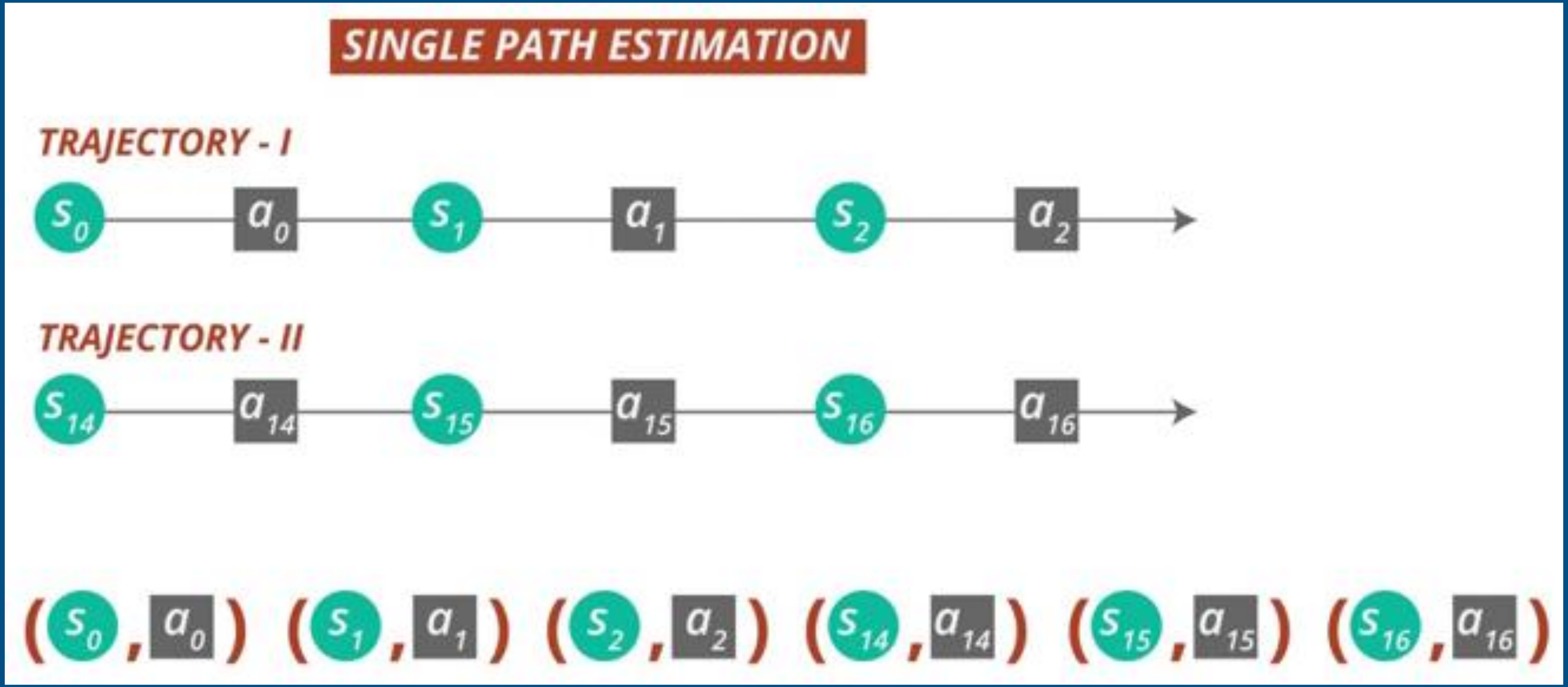
$$\underset{\theta}{\text{maximize}}\, \mathbb{E}_{s\sim\rho_{\theta_{\text{old}}}, a\sim q}\left[\frac{\pi_\theta(a|s)}{q(a|s)} Q_{\theta_{\text{old}}}(s,a)\right]$$
$$\text{subject to } \mathbb{E}_{s\sim\rho_{\theta_{\text{old}}}}\left[D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot|s) \,\|\, \pi_\theta(\cdot|s))\right] \leq \delta.$$

- Constrained Optimization → Sample-based Estimation

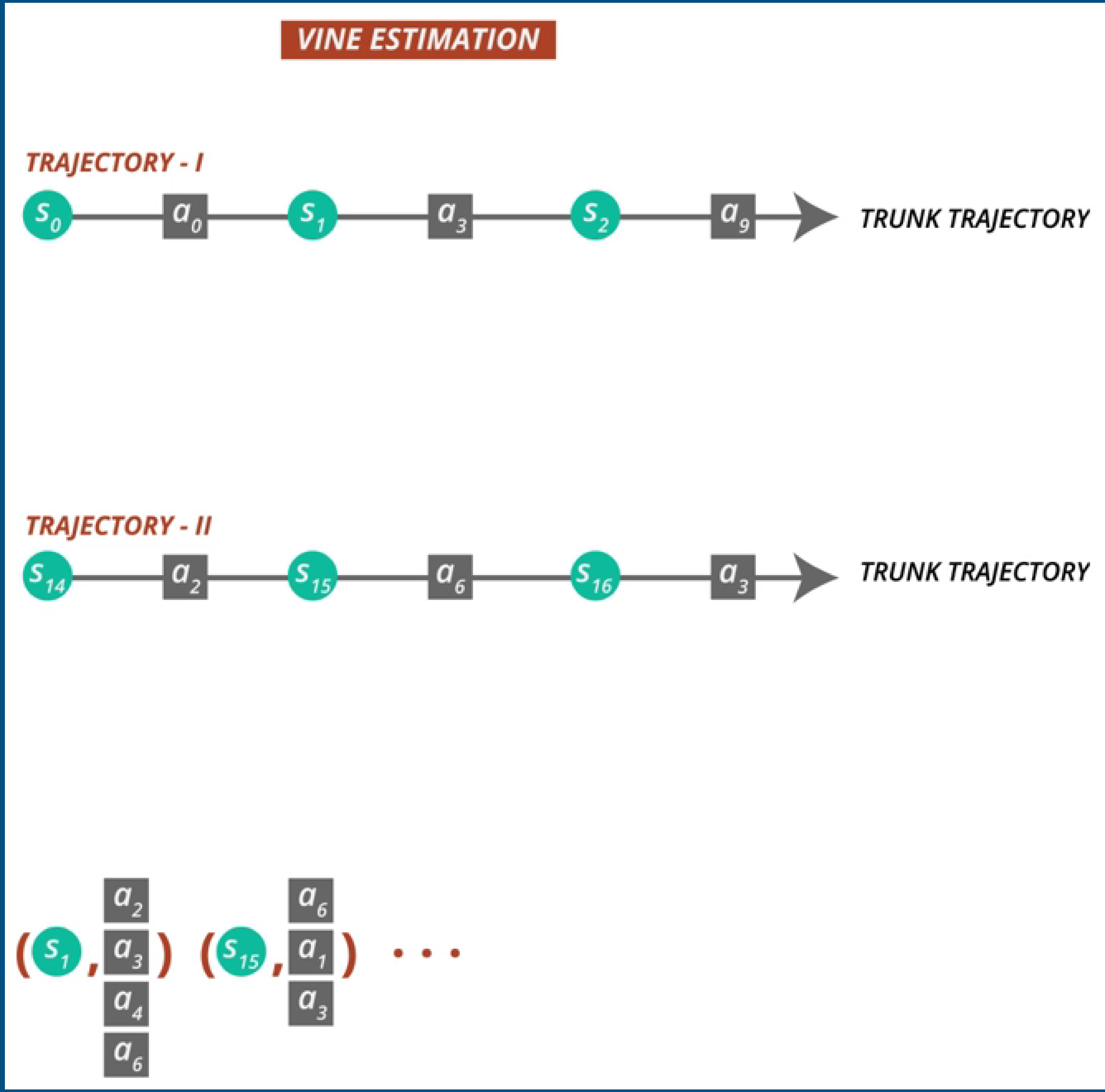
Sample-Based Estimation

- All that remains is to replace the expectations by sample averages and replace the Q value by an empirical estimate. The following sections describe two different schemes for performing this estimation.
  - Single Path: Typically used for policy gradient estimation (Bartlett & Baxter, 2011), and is based on sampling individual trajectories.
  - Vine: Constructing a rollout set and then performing multiple actions from each state in the rollout set.

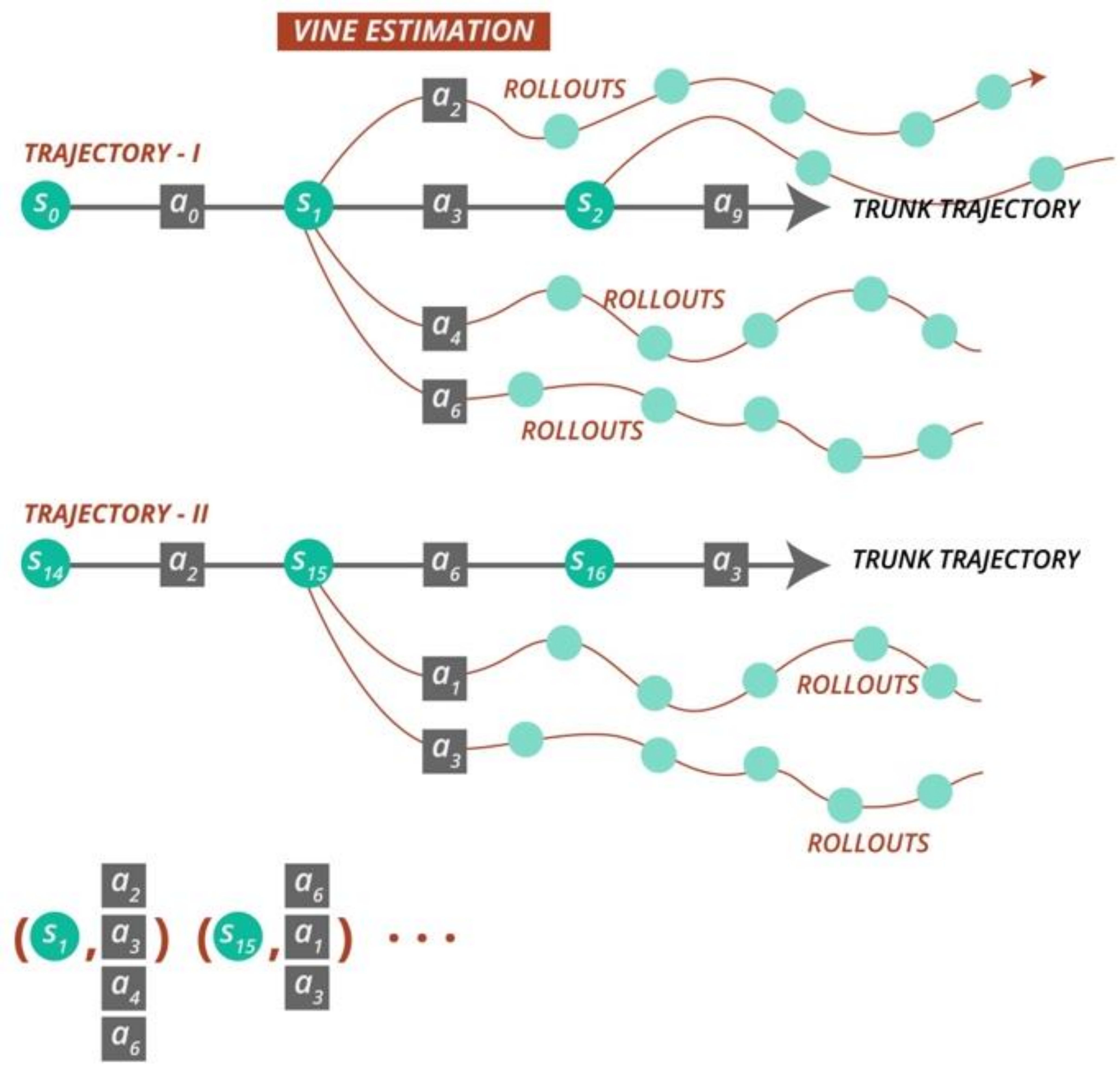
Sample-Based Estimation



Sample-Based Estimation



Sample-Based Estimation



Practical Algorithm

Trust Region Policy Optimization, Schulman et al, 2015.

1. Use the single path or vine procedures to collect a set of state-action pairs along with Monte Carlo estimates of their Q-values.
2. By averaging over samples, construct the estimated objective and constraint in Equation $\underset{\theta}{\text{maximize}}\, \mathbb{E}_{s\sim\rho_{\theta_{\text{old}}},a\sim q}\left[\frac{\pi_\theta(a|s)}{q(a|s)}Q_{\theta_{\text{old}}}(s,a)\right]$.
3. Approximately solve this constrained optimization problem to update the policy's parameter vector θ. We use the conjugate gradient algorithm followed by a line search, which is altogether only slightly more expensive than computing the gradient itself. See Appendix C for details.

Practical Algorithm

Trust Region Policy Optimization, Schulman et al, 2015.

- With regard to (3), we construct the Fisher information matrix (FIM) by analytically computing the Hessian of KL divergence, rather than using the covariance matrix of the gradients.
- That is, we estimate A_{ij} as $\frac{1}{N}\sum_{n=1}^{N}\frac{\partial^2}{\partial\theta_i\partial\theta_j}D_{\mathrm{KL}}(\pi_{\theta_{\mathrm{old}}}(\cdot|s_n)\,\|\,\pi_\theta(\cdot|s_n))$, rather than $\frac{1}{N}\sum_{n=1}^{N}\frac{\partial}{\partial\theta_i}\log\pi_\theta(a_n|s_n)\frac{\partial}{\partial\theta_j}\log\pi_\theta(a_n|s_n)$.
- This analytic estimator has computational benefits in the large-scale setting, since it removes the need to store a dense Hessian or all policy gradients from a batch of trajectories.

Experiments

Trust Region Policy Optimization, Schulman et al, 2015.

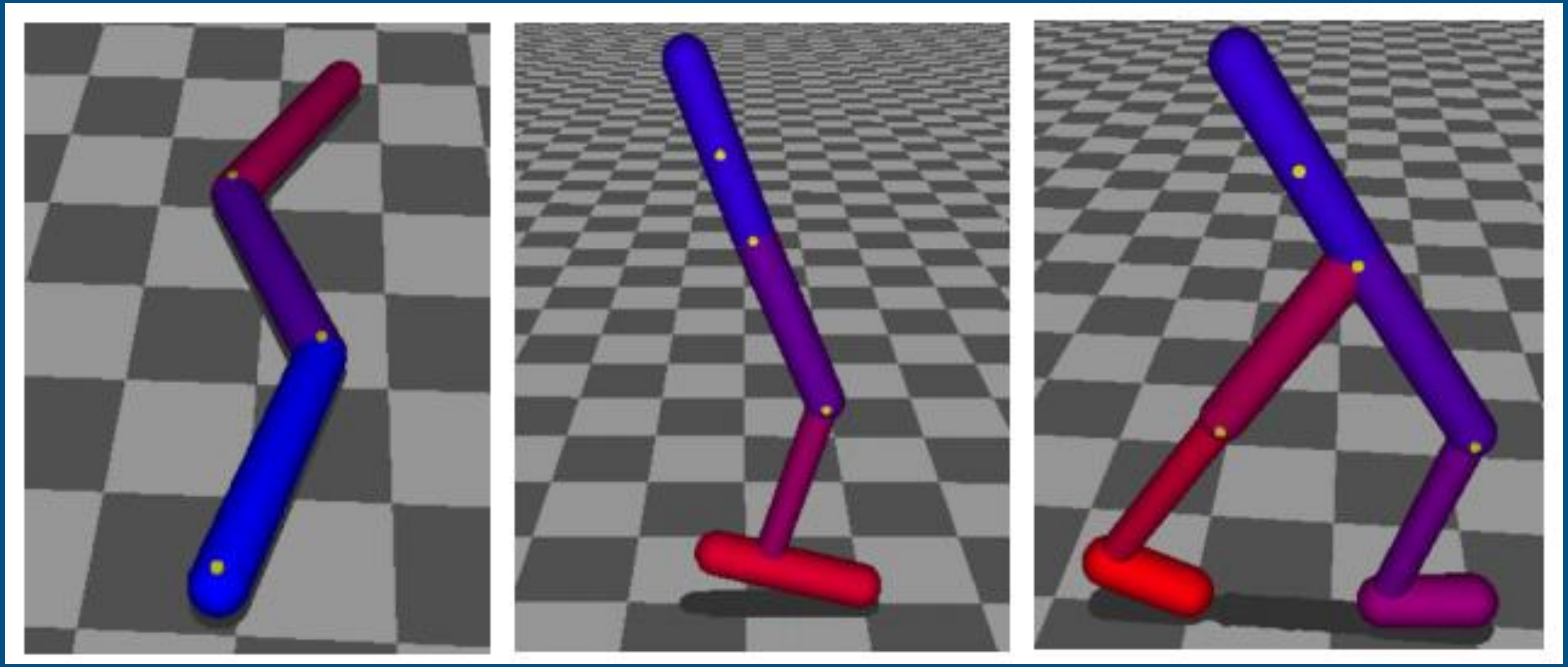
We designed experiments to investigate the following questions:

1. What are the performance characteristics of the single path and vine sampling procedures?
2. TRPO is related to prior methods but makes several changes, most notably by using a fixed KL divergence rather than a fixed penalty coefficient. How does this affect the performance of the algorithm?
3. Can TRPO be used to solve challenging large-scale problems? How does TRPO compare with other methods when applied to large-scale problems, with regard to final performance, computation time, and sample complexity?

Experiments

Trust Region Policy Optimization, Schulman et al, 2015.

- Simulated Robotic Locomotion



Experiments

Trust Region Policy Optimization, Schulman et al, 2015.

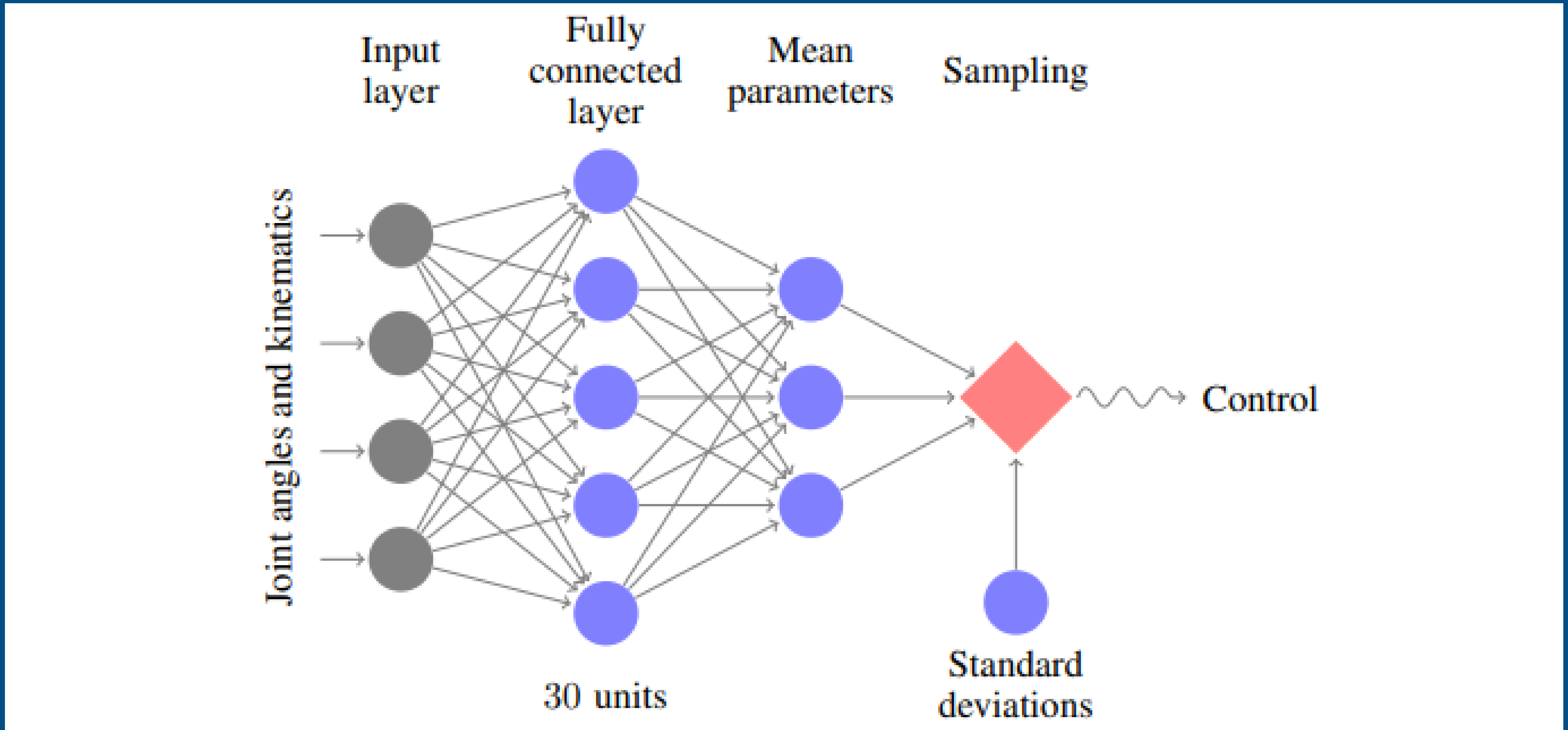
- Simulated Robotic Locomotion

	Swimmer	Hopper	Walker
State space dim.	10	12	20
Control space dim.	2	3	6
Total num. policy params	364	4806	8206
Sim. steps per iter.	50K	1M	1M
Policy iter.	200	200	200
Stepsize ($\overline{D}_{\mathrm{KL}}$)	0.01	0.01	0.01
Hidden layer size	30	50	50
Discount (γ)	0.99	0.99	0.99
Vine: rollout length	50	100	100
Vine: rollouts per state	4	4	4
Vine: Q-values per batch	500	2500	2500
Vine: num. rollouts for sampling	16	16	16
Vine: len. rollouts for sampling	1000	1000	1000
Vine: computation time (minutes)	2	14	40
SP: num. path	50	1000	10000
SP: path len.	1000	1000	1000
SP: computation time	5	35	100

Experiments

Trust Region Policy Optimization, Schulman et al, 2015.

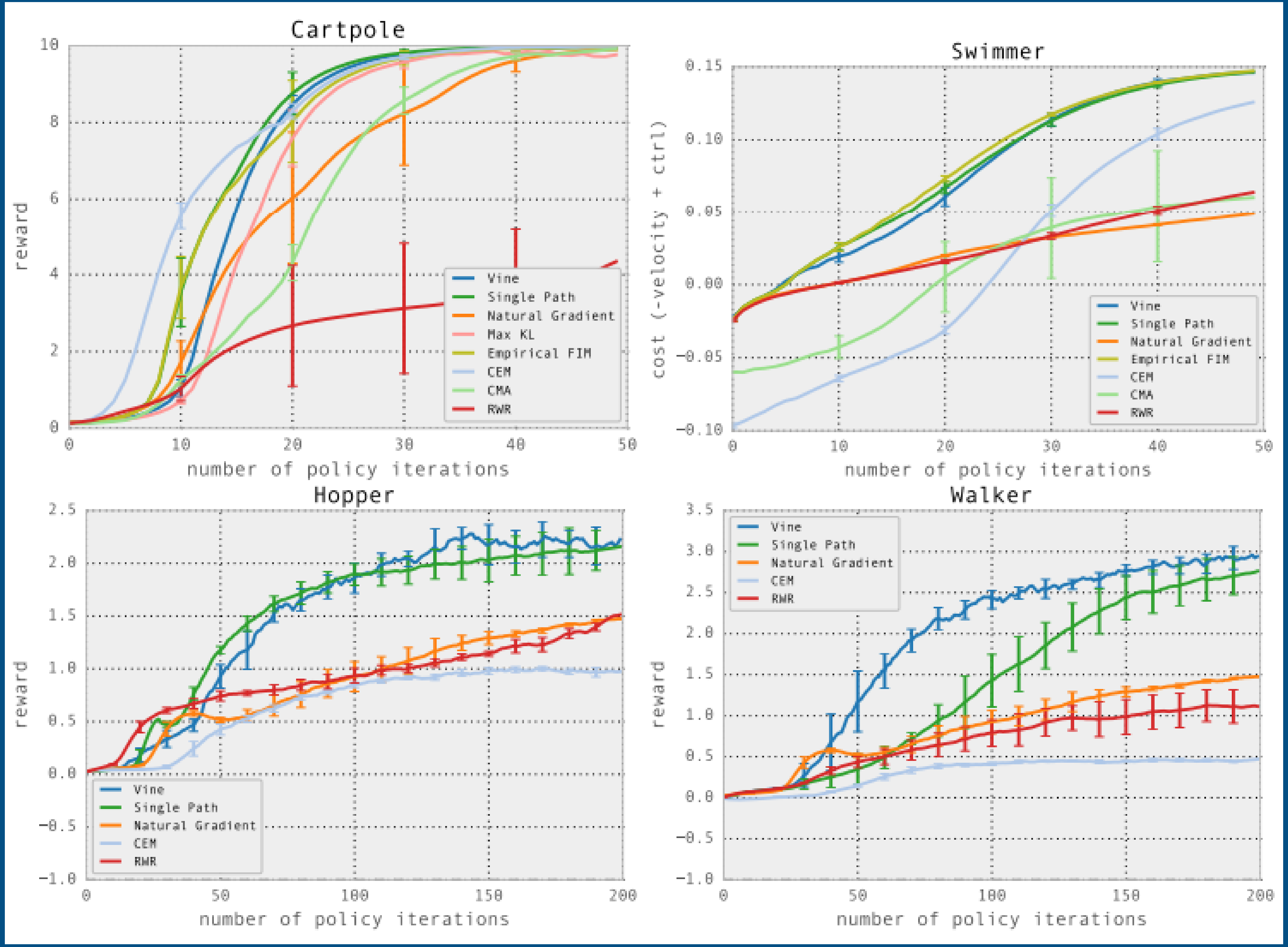
- Simulated Robotic Locomotion



Experiments

Trust Region Policy Optimization, Schulman et al, 2015.

- Simulated Robotic Locomotion



Experiments

Trust Region Policy Optimization, Schulman et al, 2015.

- Playing Games from Images
  - To evaluate TRPO on a partially observed task with complex observations, we trained policies for playing Atari games, using raw images as input.
  - Challenging points
    - The high dimensionality, challenging elements of these games include delayed rewards (no immediate penalty is incurred when a life is lost in Breakout or Space Invaders)
    - Complex sequences of behavior (Q*bert requires a character to hop on 21 different platforms)
    - Non-stationary image statistics (Enduro involves a changing and flickering background)

Experiments

Trust Region Policy Optimization, Schulman et al, 2015.

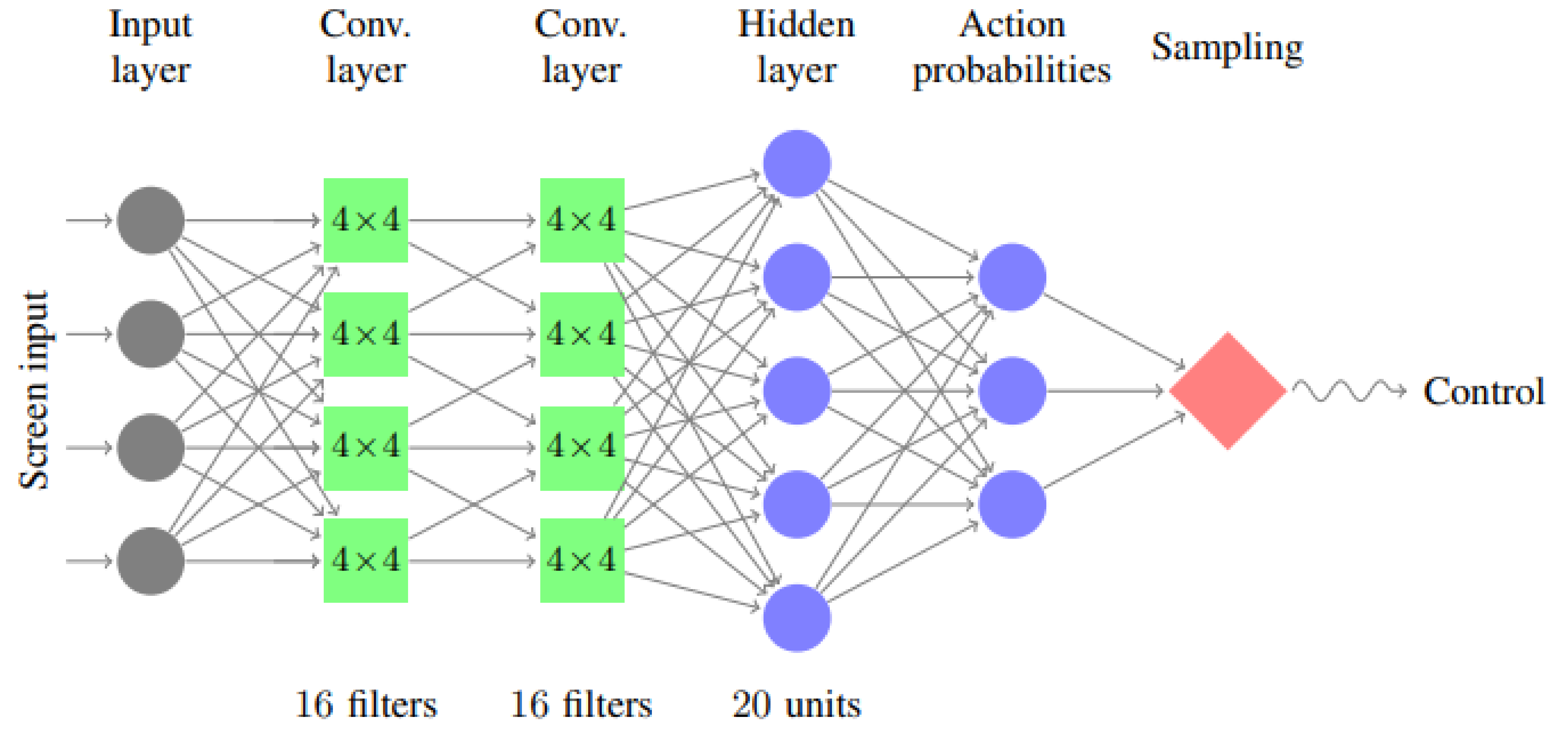
- Playing Games from Images

	All games
Total num. policy params	33500
Vine: Sim. steps per iter.	400K
SP: Sim. steps per iter.	100K
Policy iter.	500
Stepsize ($\overline{D}_{\mathrm{KL}}$)	0.01
Discount (γ)	0.99
Vine: rollouts per state	≈ 4
Vine: computation time	≈ 30 hrs
SP: computation time	≈ 30 hrs

Experiments

Trust Region Policy Optimization, Schulman et al, 2015.

- Playing Games from Images



Experiments

Trust Region Policy Optimization, Schulman et al, 2015.

- Playing Games from Images

	B. Rider	*Breakout*	*Enduro*	*Pong*	*Q*bert*	*Seaquest*	*S. Invaders*
Random	354	1.2	0	−20.4	157	110	179
Human (Mnih et al., 2013)	7456	31.0	368	−3.0	18900	28010	3690
Deep Q Learning (Mnih et al., 2013)	4092	168.0	470	20.0	1952	1705	581
UCC-I (Guo et al., 2014)	5702	380	741	21	20025	2995	692
TRPO - single path	1425.2	10.8	534.6	20.9	1973.5	1908.6	568.4
TRPO - vine	859.5	34.2	430.8	20.9	7732.5	788.4	450.2

Discussion

- We proposed and analyzed trust region methods for optimizing stochastic control policies.
- We proved monotonic improvement for an algorithm that repeatedly optimizes a local approximation to the expected return of the policy with a KL divergence penalty.

Discussion

Trust Region Policy Optimization, Schulman et al, 2015.

- In the domain of robotic locomotion, we successfully learned controllers for swimming, walking and hopping in a physics simulator, using general purpose neural networks and minimally informative rewards.
- At the intersection of the two experimental domains we explored, there is the possibility of learning robotic control policies that use vision and raw sensory data as input, providing a unified scheme for training robotic controllers that perform both perception and control.

Discussion

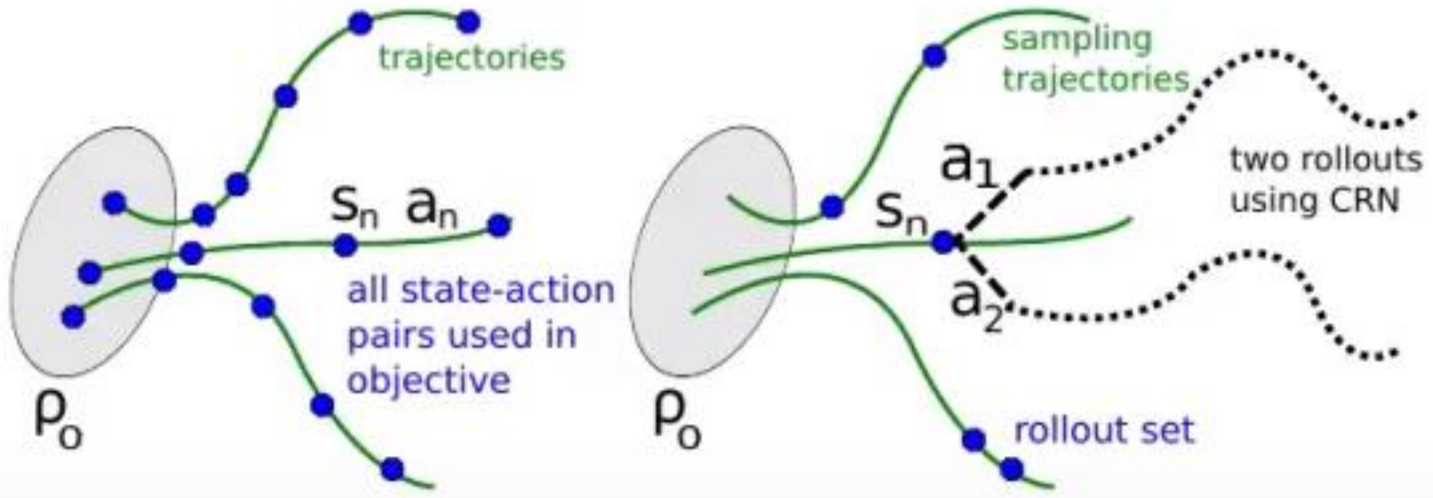
Trust Region Policy Optimization, Schulman et al, 2015.

- By combining our method with model learning, it would also be possible to substantially reduce its sample complexity, making it applicable to real-world settings where samples are expensive.

Summary

Trust Region Policy Optimization, Schulman et al, 2015.

1. Use the *single path* or *vine* procedures to collect a set of state-action pairs along with Monte Carlo estimates of their Q-values.



Find Monte Carlo Estimates of Q values for (s,a) samples

2. By averaging over samples, construct the estimated objective and constraint in Equation $\boxed{14}$.

$$\underset{\theta}{\text{maximize}}\, \mathbb{E}_{s\sim\rho_{\theta_{\text{old}}}, a\sim q}\left[\frac{\pi_\theta(a|s)}{q(a|s)} Q_{\theta_{\text{old}}}(s,a)\right] \tag{14}$$
$$\text{subject to } \mathbb{E}_{s\sim\rho_{\theta_{\text{old}}}}\left[D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot|s) \,\|\, \pi_\theta(\cdot|s))\right] \leq \delta.$$

Plug the calculated Q values

+

Plug old action prob for KL Div

3. Approximately solve this constrained optimization problem to update the policy's parameter vector θ. We use the conjugate gradient algorithm followed by a line search, which is altogether only slightly more expensive than computing the gradient itself. See Appendix $\boxed{C}$ for details.

$$\frac{1}{N}\sum_{n=1}^{N} \frac{\partial^2}{\partial\theta_i \partial\theta_j} D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot|s_n) \,\|\, \pi_\theta(\cdot|s_n))$$

Policy update directions are *conjugate* w.r.t F.I.M (Fisher Information Matrix)

References

- https://www.slideshare.net/WoongwonLee/trpo-87165690
- https://reinforcement-learning-kr.github.io/2018/06/24/5_trpo/
- https://www.youtube.com/watch?v=XBO4oPChMfI
- https://www.youtube.com/watch?v=CKaN5PgkSBc

Thank you!