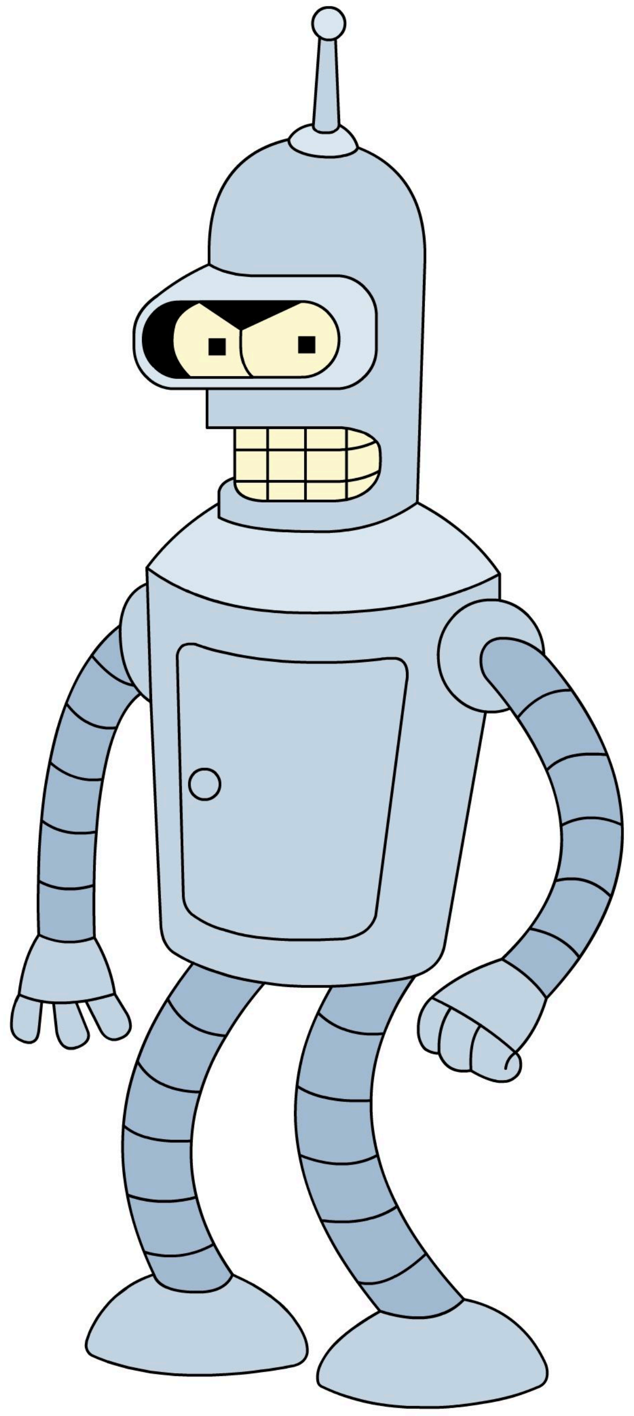
Reinforcement Learning

HSE, autumn - winter 2022

Lecture 6: Advanced Policy Optimisation



Sergei Laktionov
slaktionov@hse.ru
LinkedIn

Recap: Policy Gradient

$$\nabla J(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \,|\, s_t)\Psi_t\Big],$$

where Ψ_t may be one of the following:

- $\sum_{t=0}^{T} \gamma^t R_t$: total reward of the trajectory
- $\sum_{k=t}^{T} \gamma^{k-t} R_k$: reward following action a_t
- $\sum_{k=t}^{T} \gamma^{k-t} R_k - b(s_t)$: baseline version of previous formula.
- $Q^\pi(s_t, a_t)$: action value function
- $A^\pi(s_t, a_t)$: advantage function
- $\sum_{t=0}^{N-1} \gamma^t r_t + \gamma^N V^\pi(s_{t+N}) - V^\pi(s_t)$: TD(N) residual

Recap: A2C

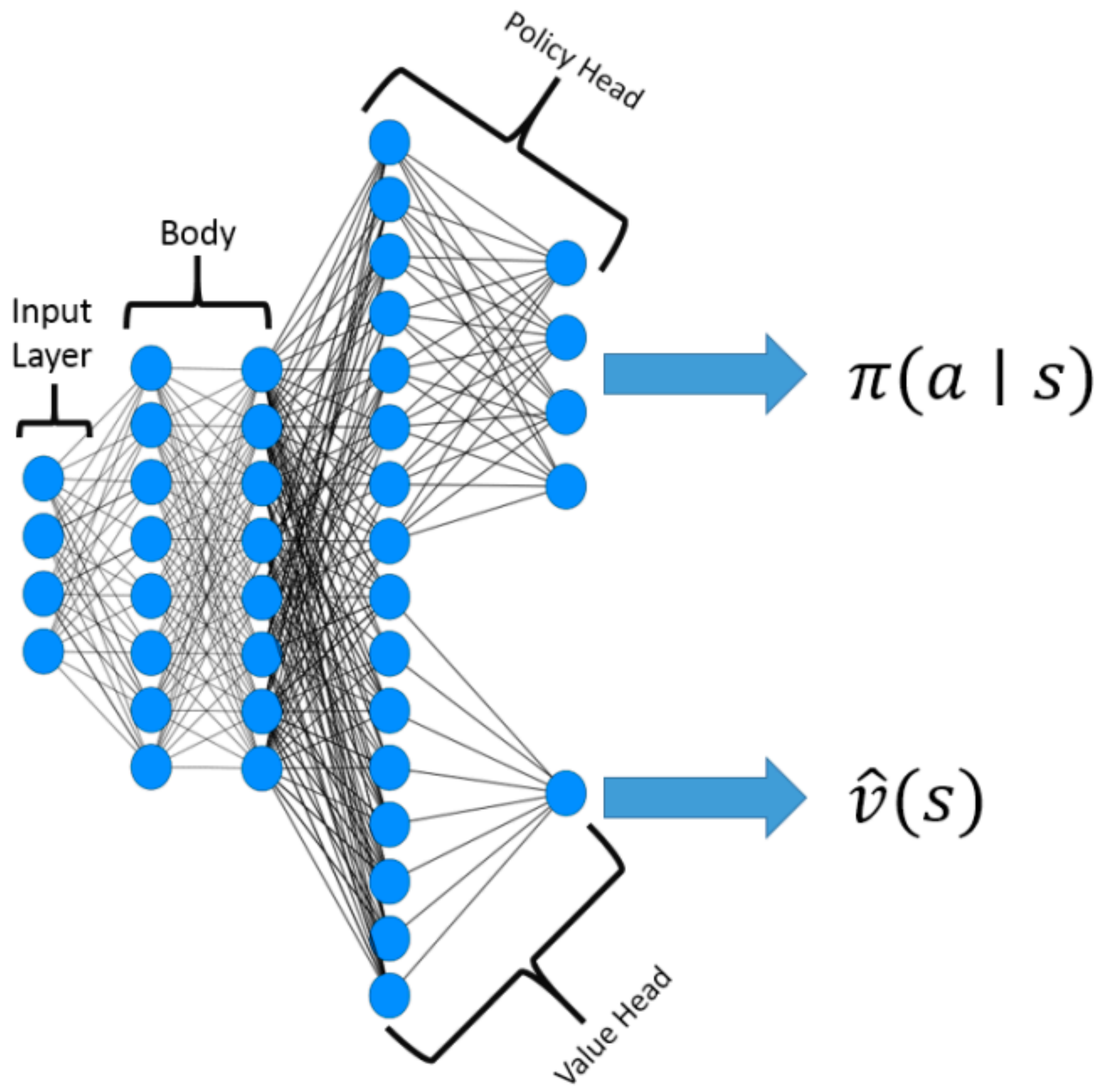
- Generate trajectories $\{\tau_i\}$ following $\pi_\theta(a \mid s)$ in parallel
- Policy improvement:
  - $A^\phi(s_{i,t}, a_{i,t}) = r_{i,t} + \gamma(1 - done_{i,t})V(s_{i,t+1}) - V(s_{i,t})$
  - Estimate gradient and make gradient ascent step:

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_{i,t} \mid s_{i,t}) A^\phi(s_{i,t}, a_{i,t})\Big]$$

- Policy evaluation:
- Estimate gradient and make gradient descent step:

$$\nabla_\phi L(\phi) \approx \frac{1}{N}\sum_{i=1}^{N}\Big[\sum_{t=0}^{T} \nabla_\phi (r_{i,t} + \gamma V_{\boxed{\phi^-}}(s_{i,t+1}) - V_\phi(s_{i,t}))^2\Big]$$

Not target network, just frozen parameters



Source

Policy Gradient

$$\nabla J(\theta) = \mathbb{E}_{p_\theta(\tau)}\Big[\sum_{t=0}^{T} \nabla \log \pi_\theta(a_t \,|\, s_t)\Psi_t\Big]$$

The choice $\Psi_t = A^\pi(s_t, a_t)$ yields almost the lowest possible variance, though in practice, the advantage function is not known and must be estimated.

Advantage Estimator

- Let V be an approximate value function
- $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$
- $A_t^{(1)} = r_t + \gamma V(s_{t+1}) - V(s_t) = \delta_t$
- $A_t^{(2)} = r_t + \gamma r_{t+1} + \gamma^2 V(s_{t+2}) - V(s_t) =$
  $= r_t \; + \gamma V(s_{t+1}) - V(s_t) + \gamma r_{t+1} + \gamma^2 V(s_{t+2}) \; - \gamma V(s_{t+1}) = \delta_t + \gamma \delta_{t+1}$
- …
- $A_t^{(N)} = r_t + \gamma r_{t+1} + \ldots + \gamma^{N-1} r_{t+N-1} + \gamma^N V(s_{t+N}) - V(s_t) = \sum_{k=0}^{N-1} \gamma^k \delta_{t+k}$
- $A_t^{(\infty)} = \sum_{k=0}^{\infty} \gamma^k \delta_{t+k}$

Advantage Estimator

- Let V be an approximate value function
- $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$
- $A_t^{(1)} = r_t + \gamma V(s_{t+1}) - V(s_t) = \delta_t$
- $A_t^{(2)} = r_t + \gamma r_{t+1} + \gamma^2 V(s_{t+2}) - V(s_t) =$
  $= r_t + \gamma V(s_{t+1}) - V(s_t) + \gamma r_{t+1} + \gamma^2 V(s_{t+2}) - \gamma V(s_{t+1}) = \delta_t + \gamma \delta_{t+1}$

Low variance
High bias

- …
- $A_t^{(N)} = r_t + \gamma r_{t+1} + \ldots + \gamma^{N-1} r_{t+N-1} + \gamma^N V(s_{t+N}) - V(s_t) = \sum_{k=0}^{N-1} \gamma^k \delta_{t+k}$
- $A_t^{(\infty)} = \sum_{k=0}^{\infty} \gamma^k \delta_{t+k}$

High variance
Low bias

Generalised Advantage Estimator

- $$A_t^{(N)} = \sum_{k=0}^{N-1} \gamma^k \delta_{t+k}$$
- GAE is defined as the exponentially-weighted average of these N-step estimators:

$$A_t^{GAE(\gamma,\lambda)} = (1 - \lambda)(A_t^{(1)} + \lambda A_t^{(2)} + \ldots) =$$

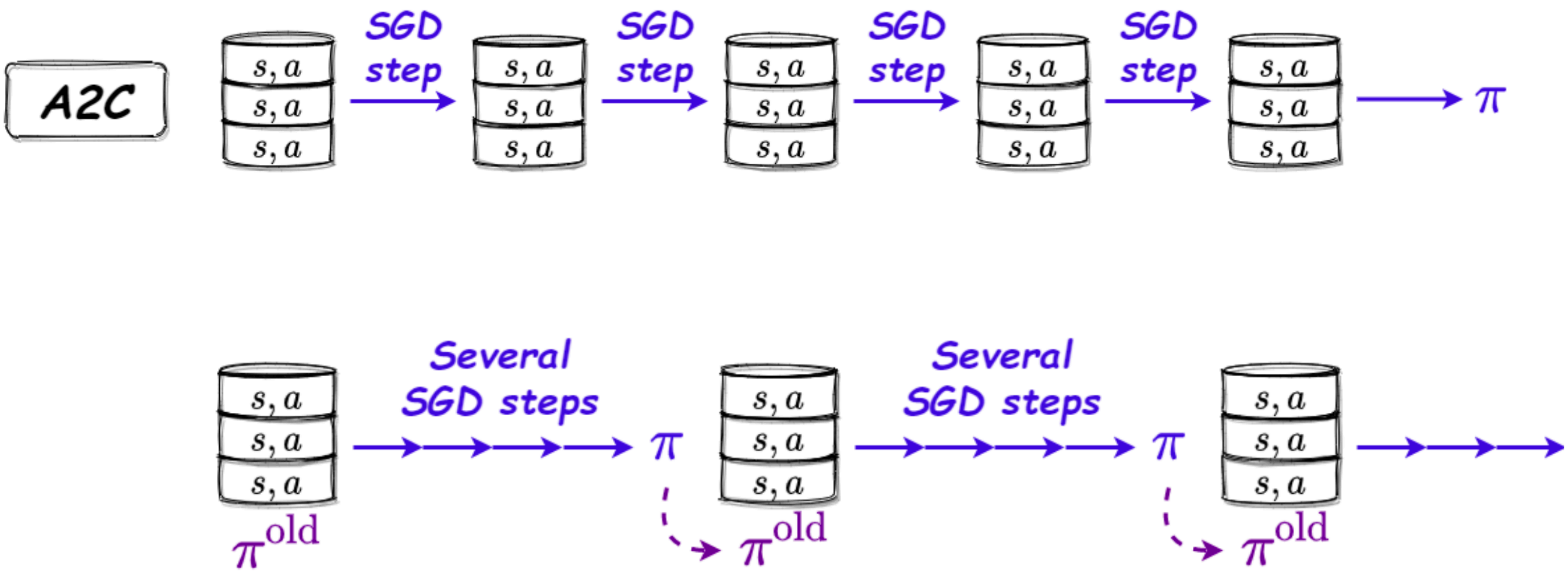
Generalised Advantage Estimator

- $$A_t^{(N)} = \sum_{k=0}^{N-1} \gamma^k \delta_{t+k}$$
- GAE is defined as the exponentially-weighted average of these N-step estimators:

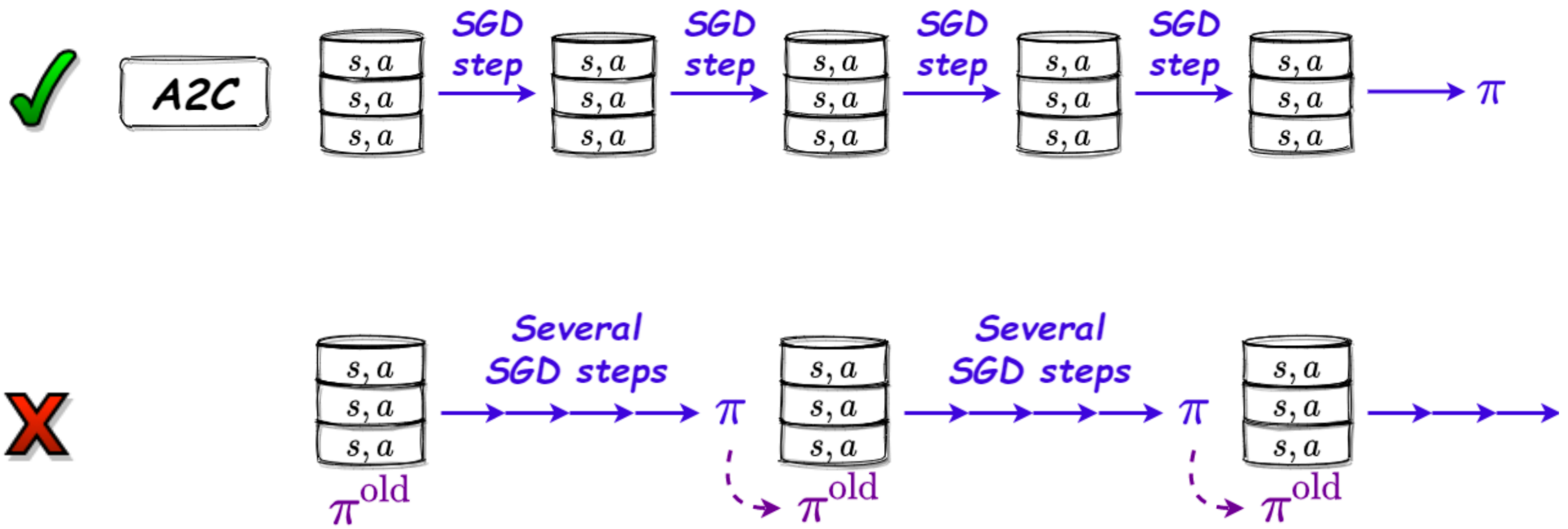
$$A_t^{GAE(\gamma,\lambda)} = (1-\lambda)(A_t^{(1)} + \lambda A_t^{(2)} + \ldots) = \sum_{k=0}^{\infty} (\gamma\lambda)^k \delta_{t+k}$$

Optimisation

Optimisation



Optimisation



Policy Optimisation via Gradient Ascent

Several issues:

- We make gradient step in the space of parameters, get new parameters θ and policy π_θ from θ_{old} and old policy $\pi_{\theta_{old}}$. However, it's difficult to measure the impact of change in parameters on change in policy.
- Apply only first-order optimisation methods
- Low sample efficiency

$$\theta = \theta_{old} + \alpha \nabla J(\theta_{old})$$

Optimisation

$$J(\theta) \approx J(\theta_{old}) + \nabla J(\theta_{old})(\theta - \theta_{old})$$

$$J(\theta) \to \max_{\theta} \quad \Longleftrightarrow \quad \begin{aligned} &\nabla J(\theta_{old})(\theta - \theta_{old}) \to \max_{\theta} \\ &\text{s.t. } (\theta - \theta_{old})^T(\theta - \theta_{old}) \leq \delta \end{aligned}$$

Let’s $d = \theta - \theta_{old}$, then $d^* \propto \nabla J(\theta_{old})$

$$\theta = \theta_{old} + \alpha \nabla J(\theta_{old})$$

Optimisation

$$J(\theta_{old})(\theta - \theta_{old}) \to \max_{\theta} \text{ s.t.}$$

$$(\theta - \theta_{old})^T K(\theta - \theta_{old}) \leq \delta$$

K is symmetric, positive-definite matrix

Let's $d = \theta - \theta_{old}$, then $d^* \propto K^{-1} \nabla J(\theta_{old})$

$$\theta = \theta_{old} + \alpha K^{-1} \nabla J(\theta_{old})$$

Natural Gradient

$KL(\pi_{\theta_{old}}||\pi_\theta) \approx \frac{1}{2}(\theta - \theta_{old})^T K(\theta_{old})(\theta - \theta_{old})$, where $K(\theta_{old}) = \nabla^2_\theta KL(\pi_{old}||\pi_\theta)|_{\theta_{old}}$

$\theta = \theta_{old} + \alpha s$, where $s = K^{-1}\nabla J(\theta_{old})$, α is a step size.

Natural Gradient

$KL(\pi_{\theta_{old}} || \pi_\theta) \approx \frac{1}{2}(\theta - \theta_{old})^T K(\theta_{old})(\theta - \theta_{old})$, where $K(\theta_{old}) = \nabla^2_\theta KL(\pi_{old} || \pi_\theta)|_{\theta_{old}}$

$\theta = \theta_{old} + \alpha s$, where $s = K^{-1} \nabla J(\theta_{old})$, α is a step size.

Choose the largest step:

$$\frac{1}{2}(\theta - \theta_{old})^T K(\theta_{old})(\theta - \theta_{old}) = \delta \iff \alpha = \sqrt{\frac{2\delta}{s^T K s}}$$

$$\theta = \theta_{old} + \alpha K^{-1} \nabla J(\theta_{old})$$

Natural Gradient

$KL(\pi_{\theta_{old}}||\pi_\theta) \approx \frac{1}{2}(\theta - \theta_{old})^T K(\theta_{old})(\theta - \theta_{old})$, where $K(\theta_{old}) = \nabla^2_\theta KL(\pi_{old}||\pi_\theta)|_{\theta_{old}}$

$\theta = \theta_{old} + \alpha s$, where $s = K^{-1}\nabla J(\theta_{old})$, α is a step size.

Choose the largest step:

$$\frac{1}{2}(\theta - \theta_{old})^T K(\theta_{old})(\theta - \theta_{old}) = \delta \iff \alpha = \sqrt{\frac{2\delta}{s^T K s}}$$

$$\theta = \theta_{old} + \alpha K^{-1}\nabla J(\theta_{old})$$

$K \in \mathbb{R}^{|\theta| \times |\theta|}$, K^{-1} computation takes $O(|\theta|^3)$

Conjugate Gradient Method

Paper

K is symmetric, positive-definite matrix

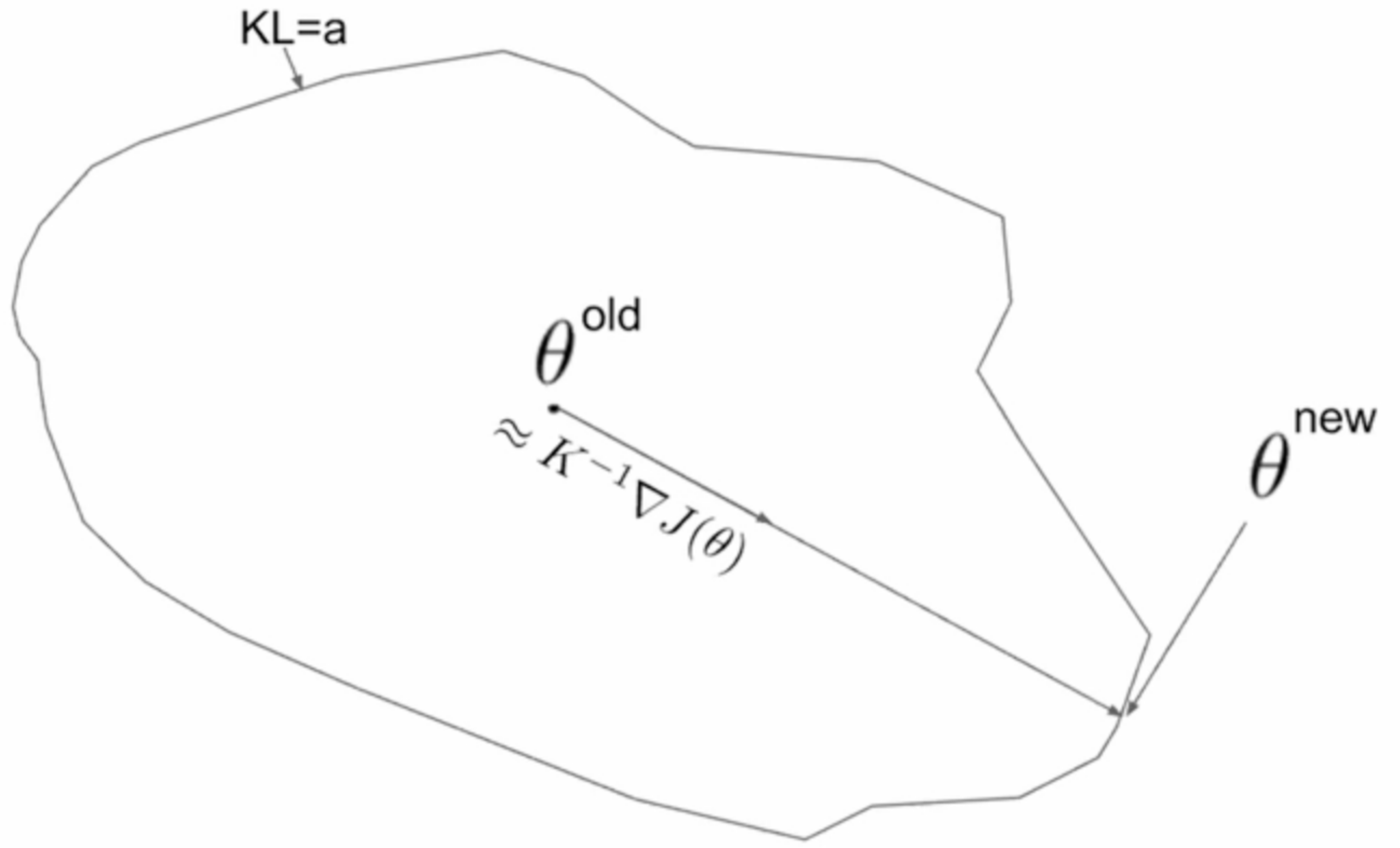
In order to find $K^{-1}\nabla J(\theta_{old})$ we can solve system $Ks = \nabla J(\theta)$ iteratively.

Conjugate Gradient Method

Paper

K is symmetric, positive-definite matrix

In order to find $K^{-1}\nabla J(\theta_{old})$ we can solve system $Ks = \nabla J(\theta)$ iteratively.



Source

Policy Improvement

On each step we would like to have a positive difference $J(\pi) - J(\pi_{old})$

Note that $J(\pi) - J(\pi_{old}) = J(\pi) - \mathbb{E}_{\tau\sim\pi_{old}}[V^{\pi_{old}}(s_0)] =$

$$= \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t r_t] + \mathbb{E}_{\tau\sim\pi_{old}}[\sum_{t=0}^{\infty} \gamma V^{\pi_{old}}(s_{t+1}) - \sum_{t=0}^{\infty} \gamma V^{\pi_{old}}(s_t)] =$$

$$= \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t (r_t + \gamma V^{\pi_{old}}(s_{t+1}) - V^{\pi_{old}}(s_t))] =$$

$$= \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t A^{\pi_{old}}(s_t, a_t)]$$

Alternative Form

$$J(\pi) - J(\pi_{old}) = \mathbb{E}_{\tau \sim \pi}[\sum_{t=0}^{\infty} \gamma^t A^{\pi_{old}}(s_t, a_t)]$$

<u>Lemma</u>:

- Define state-visitation distribution: $d_\pi(s) = (1-\gamma)\sum_{t=0}^{T} \gamma^t \mathbb{P}(s_t = s \,|\, \pi)$
- Then for all $f(s,a)$: $\mathbb{E}_{\tau \sim \pi}[\sum_{t=0}^{\infty} \gamma^t f(s_t, a_t)] = \frac{1}{1-\gamma}\mathbb{E}_{s \sim d_\pi}\mathbb{E}_{a \sim \pi(.|s)}[f(s,a)]$

$$J(\pi_\theta) - J(\pi_{old}) = \frac{1}{1-\gamma}\mathbb{E}_{s \sim d_{\pi_\theta}}\mathbb{E}_{a \sim \pi(.|s)}[A^{\pi_{old}}(s,a)]$$

Alternative Form

$$J(\pi_\theta) - J(\pi_{old}) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_{\pi_\theta}}\mathbb{E}_{a\sim\pi(.|s)}[A^{\pi_{old}}(s,a)]$$

Alternative Form

$$J(\pi_\theta) - J(\pi_{old}) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_{\pi_\theta}}\mathbb{E}_{a\sim\pi(.|s)}[A^{\pi_{old}}(s,a)]$$

$$J(\pi_\theta) - J(\pi_{old}) = \frac{1}{1-\gamma}\mathbb{E}_{\boxed{s\sim d_{\pi_\theta}}}\mathbb{E}_{a\sim\pi_{old}(.|s)}[\frac{\pi_\theta(a\,|\,s)}{\pi_{old}(a\,|\,s)}A^{\pi_{old}}(s,a)]$$

Alternative Form

$$J(\pi_\theta) - J(\pi_{old}) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_{\pi_\theta}}\mathbb{E}_{a\sim\pi(.|s)}[A^{\pi_{old}}(s,a)]$$

$$J(\pi_\theta) - J(\pi_{old}) = \frac{1}{1-\gamma}\mathbb{E}_{\boxed{s\sim d_{\pi_\theta}}}\mathbb{E}_{a\sim\pi_{old}(.|s)}[\frac{\pi_\theta(a|s)}{\pi_{old}(a|s)}A^{\pi_{old}}(s,a)]$$

Define surrogate objective:

$$L_{\pi_{old}}(\theta) = \frac{1}{1-\gamma}\mathbb{E}_{\boxed{s\sim d_{\pi_{old}}}}\mathbb{E}_{a\sim\pi_{old}(.|s)}[\frac{\pi_\theta(a|s)}{\pi_{old}(a|s)}A^{\pi_{old}}(s,a)]$$

Optimisation in Policy Space

Let $D_{KL}(\pi_{old} || \pi_\theta) = \mathbb{E}_{s \sim d_{\pi_{old}}}[D_{KL}(\pi_{old}(\,.\,|s) || \pi_\theta(\,.\,|s))]$

Performance Lower Bound:

$$J(\pi_\theta) - J(\pi_{old}) \geq L_{\pi_{old}}(\theta) - C\, D_{KL}(\pi_{old} || \pi_\theta),$$

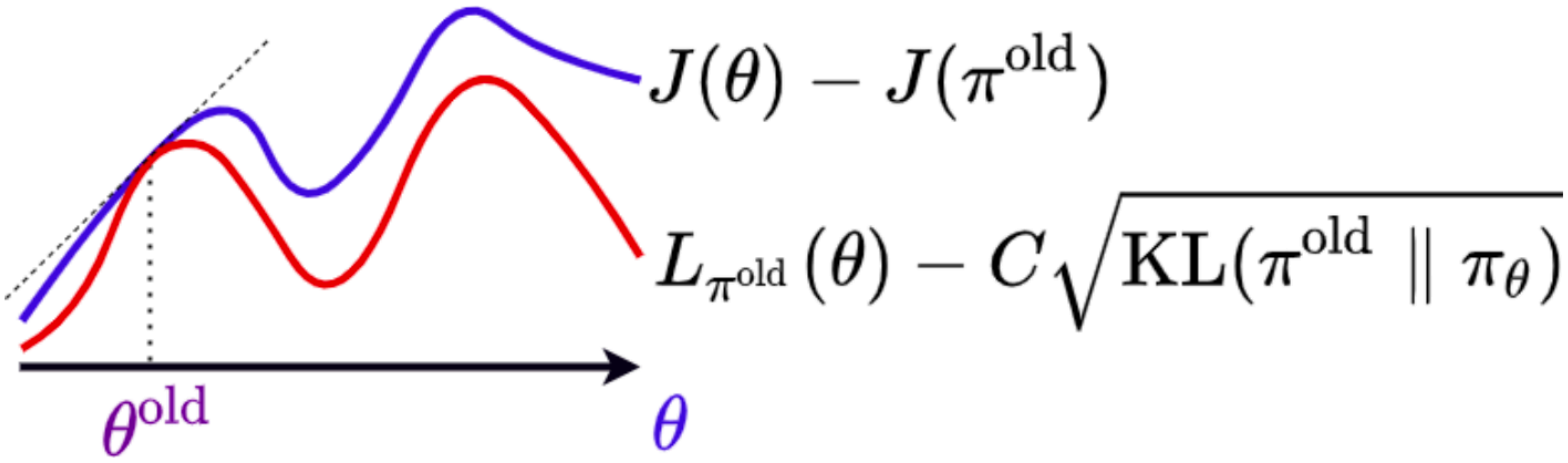
Where $C = \dfrac{\sqrt{2}\gamma}{(1-\gamma)^2} \max_{s,a} |A^{\pi_{old}}(s,a)|$

Optimisation in Policy Space

Performance Lower Bound:

$$J(\pi_\theta) - J(\pi_{old}) \geq L_{\pi_{old}}(\theta) - C\sqrt{D_{KL}(\pi_{old}\,||\,\pi_\theta)},$$

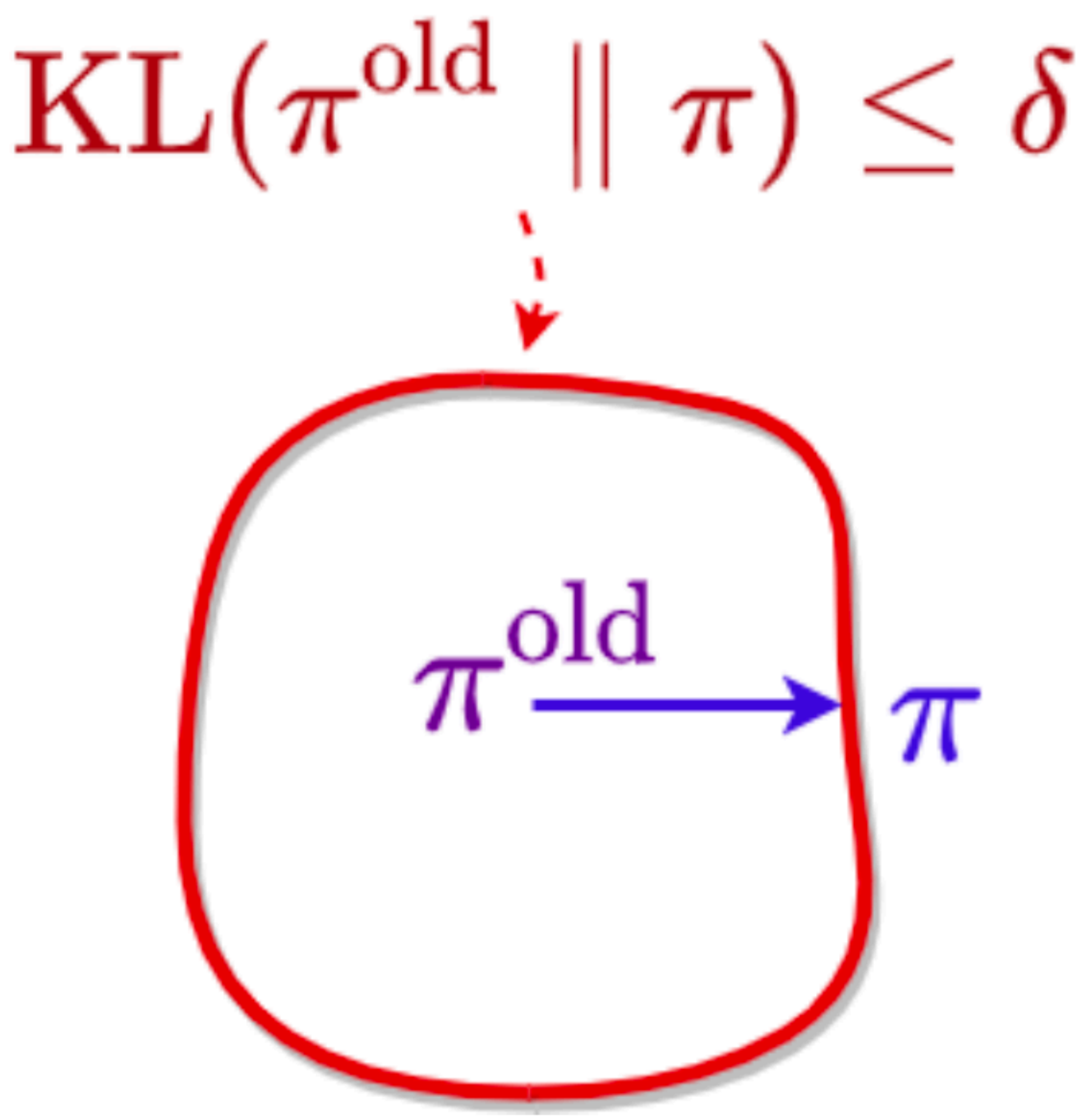
Where $C = \dfrac{\sqrt{2}\gamma}{(1-\gamma)^2} \max_{s,a} |A^{\pi_{old}}(s,a)|$



Trust Region Policy Optimisation (TRPO)

$$L_{\pi_{old}}(\theta) \rightarrow \max_{\theta}$$

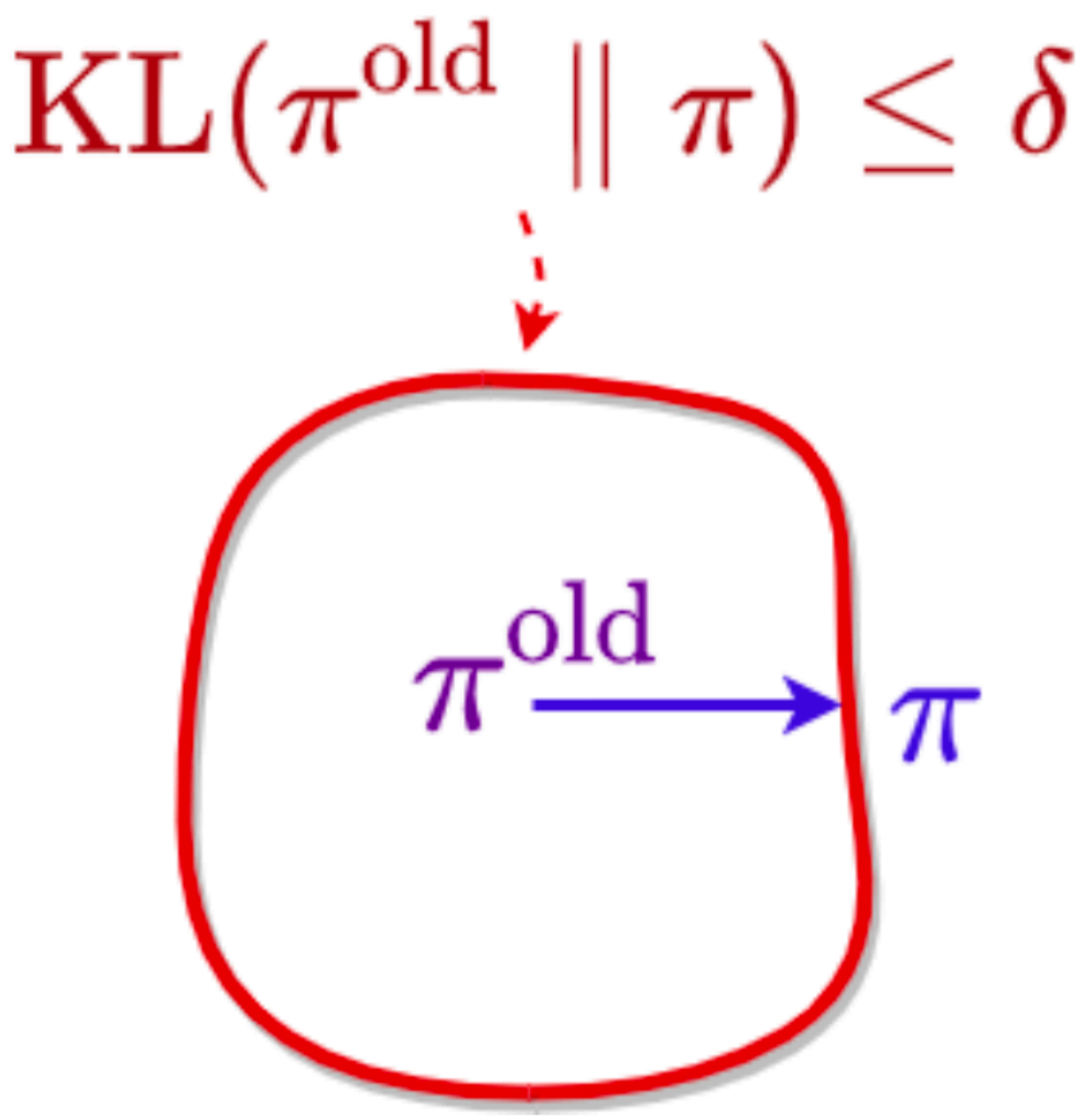
s.t. $D_{KL}(\pi_{old} || \pi_\theta) \leq \delta$



Trust Region Policy Optimisation (TRPO)

$$L_{\pi_{old}}(\theta) \rightarrow \max_{\theta}$$

s.t. $D_{KL}(\pi_{old} || \pi_{\theta}) \leq \delta$



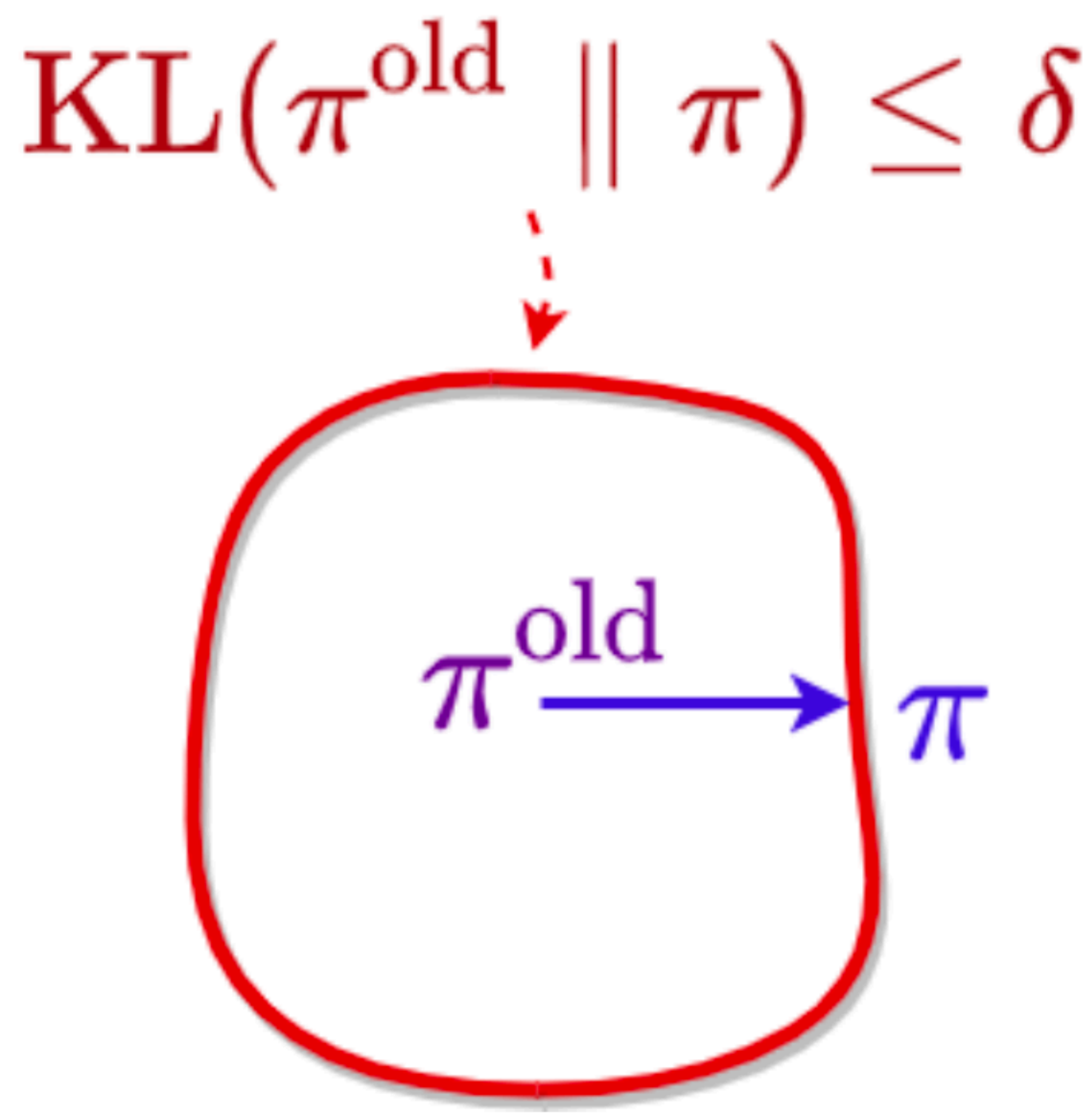
$L_{\pi_{old}}(\theta) \approx g(\theta - \theta_{old})$, where $g = \nabla_{\theta} L_{\pi_{old}}(\theta)|_{\theta_{old}}$

$D_{KL}(\pi_{\theta_{old}} || \pi_{\theta}) \approx \frac{1}{2}(\theta - \theta_{old})^T K(\theta - \theta_{old})$, where $K = \nabla^2_{\theta} D_{KL}(\pi_{old} || \pi_{\theta})|_{\theta_{old}}$

Trust Region Policy Optimisation (TRPO)

$$L_{\pi_{old}}(\theta) \to \max_{\theta}$$

s.t. $D_{KL}(\pi_{old} || \pi_\theta) \leq \delta$



$L_{\pi_{old}}(\theta) \approx g(\theta - \theta_{old})$, where $g = \nabla_\theta L_{\pi_{old}}(\theta)|_{\theta_{old}}$

$D_{KL}(\pi_{\theta_{old}} || \pi_\theta) \approx \frac{1}{2}(\theta - \theta_{old})^T K(\theta - \theta_{old})$, where $K = \nabla^2_\theta D_{KL}(\pi_{old} || \pi_\theta)|_{\theta_{old}}$

$\theta = \theta_{old} + \alpha K^{-1} g$, where $\alpha = \sqrt{\frac{2\delta}{g^T K^{-1} g}}$

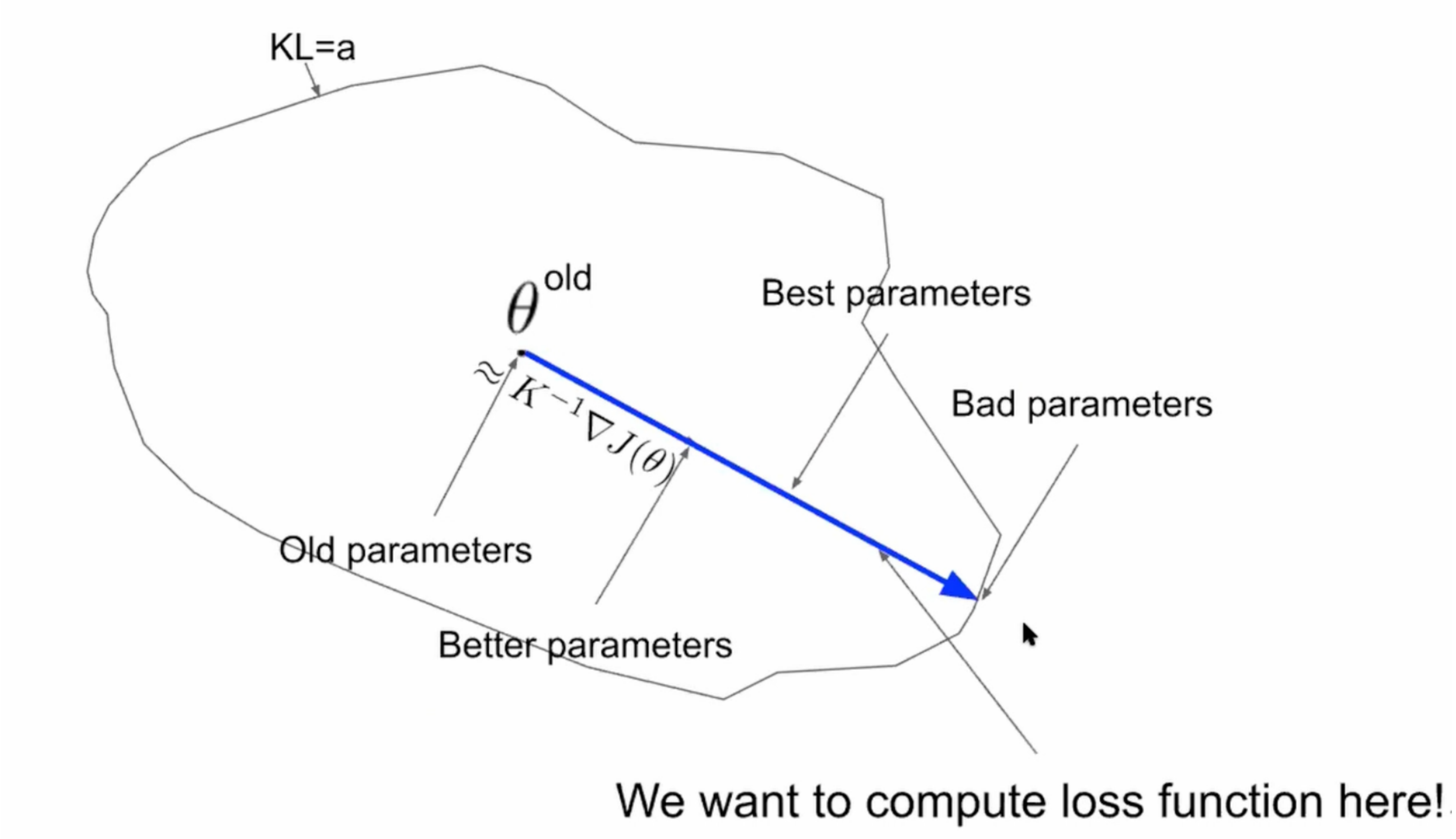
$K \in \mathbb{R}^{|\theta| \times |\theta|}$, K^{-1} computation takes $O(|\theta|^3)$

Conjugate Gradient Method

K is symmetric, positive-definite matrix

In order to find $K^{-1} \nabla J(\theta_{old})$ we can solve system $Ks = \nabla J(\theta)$ iteratively.

Visualisation



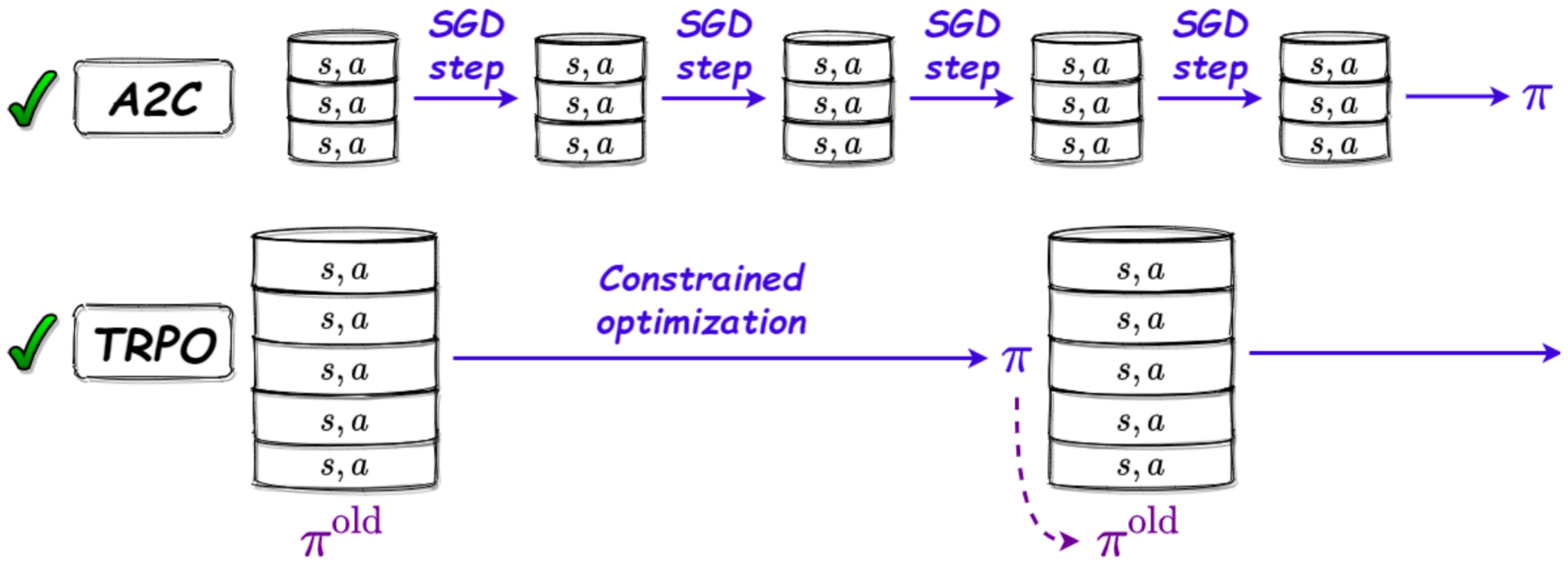
Source

TRPO Algorithm

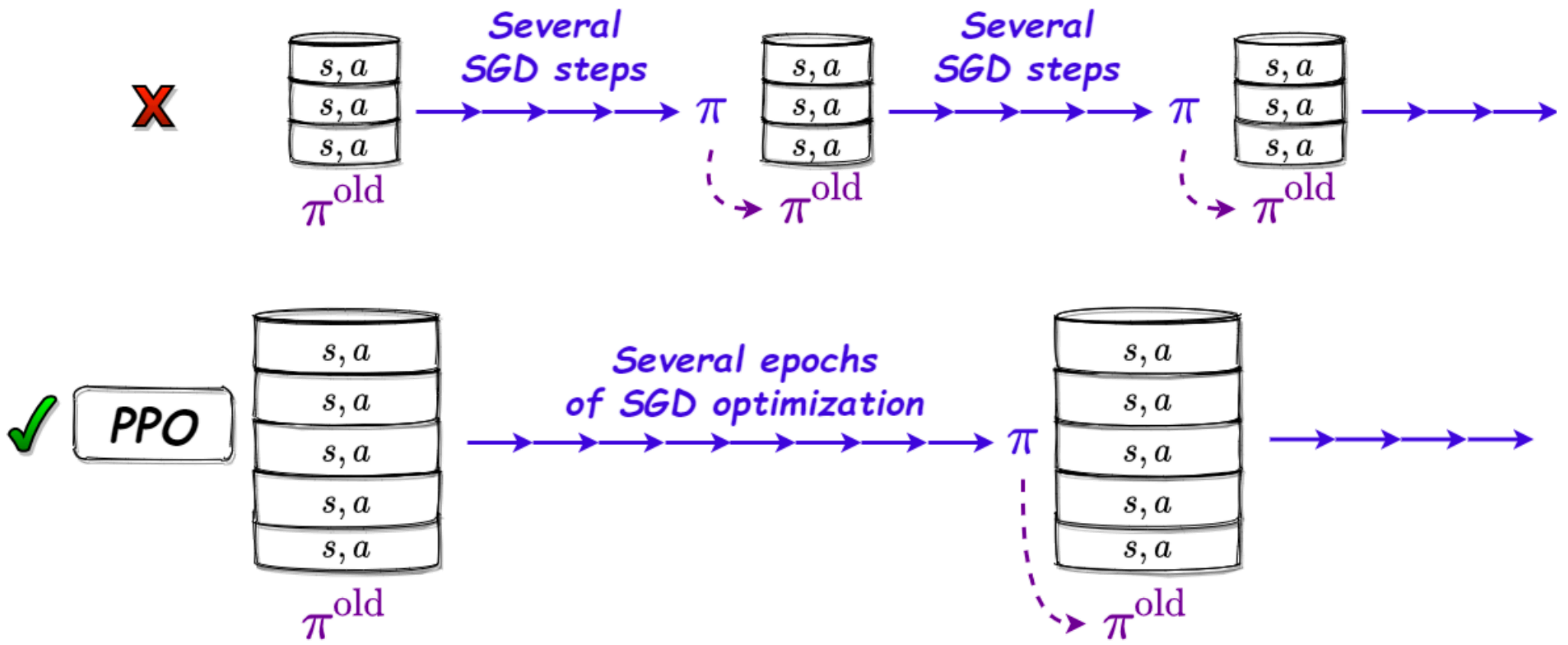
Repeat until convergence:

1. Collect trajectories following current policy $\pi_{\theta_{old}}$
2. Compute $g = \nabla_\theta \frac{1}{N} \sum_{i=1}^{N} \frac{\pi_\theta(a_i \mid s_i)}{\pi_{\theta_{old}}(a_i \mid s_i)} A^{\pi_{\theta_{old}}}(s_i, a_i)$
3. Compute $K = \nabla_\theta^2 \frac{1}{N} \sum_{i=1}^{N} D_{KL}(\pi_{\theta_{old}}(. \mid s_i) \mid\mid \pi_\theta(. \mid s_i))$
4. Find optimal direction via Conjugate Gradients Method (find $s = K^{-1}g$)
5. Do linear search in optimal direction checking the KL constraint and objective value for each new parameter: $\theta_j = \theta_{old} + \alpha_j \sqrt{\frac{2\delta}{g^T s}} s$

Comparison



Beyond the Second-order Optimisation



Problem Formulation

$$L_{\pi_{old}}(\theta) = \mathbb{E}_{s \sim d_{\pi_{old}}} \mathbb{E}_{a \sim \pi_{old}(.|s)} [\frac{\pi_\theta(a\,|\,s)}{\pi_{old}(a\,|\,s)} A^{\pi_{old}}(s, a)]$$

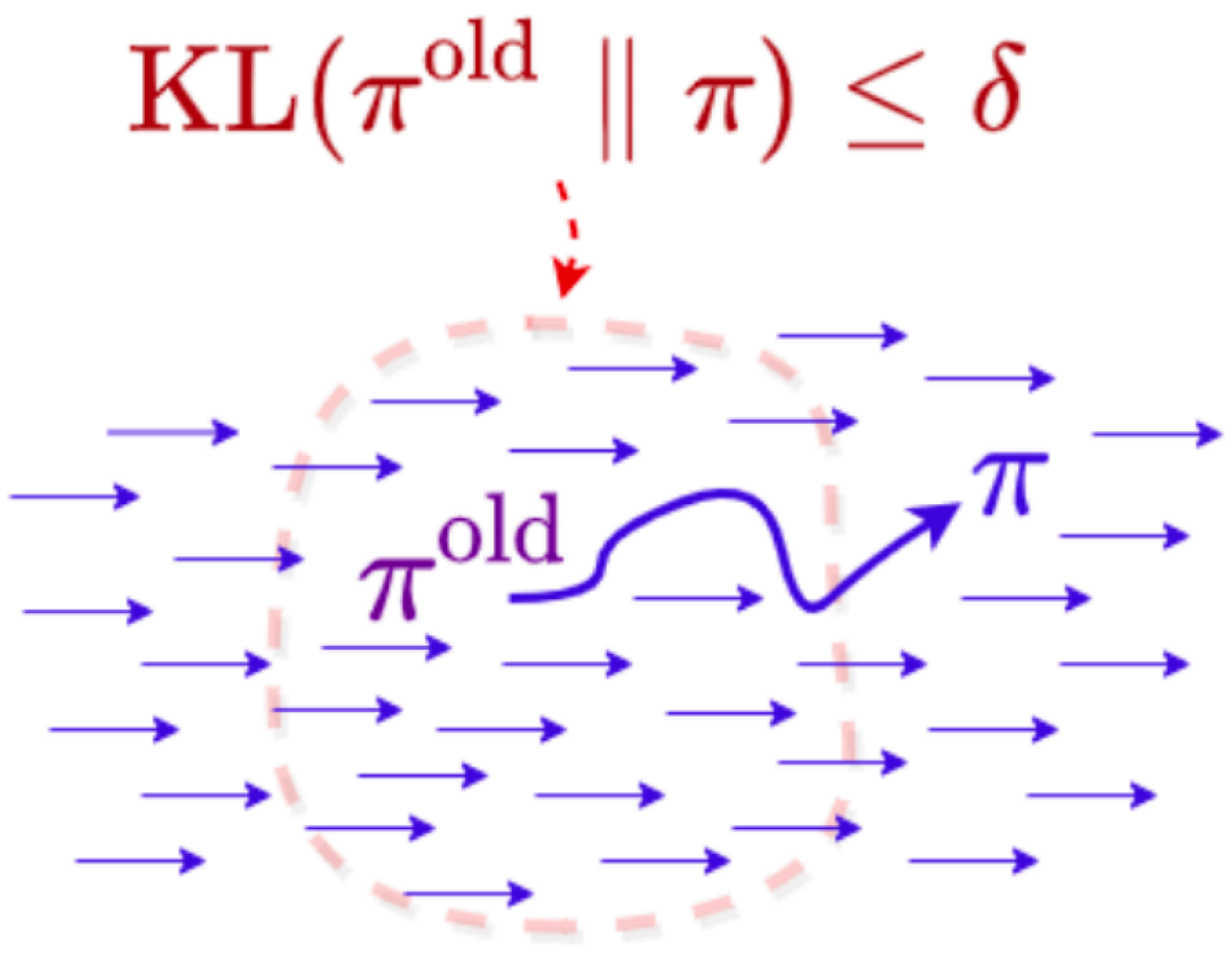
Constrained problem

$$L_{\pi_{old}}(\theta) \to \max_\theta$$

s.t. $D_{KL}(\pi_{old}\,||\,\pi_\theta) \leq \delta$

Unconstrained problem

$$L_{\pi_{old}}(\theta) - \beta D_{KL}(\pi_{old}\,||\,\pi_\theta) \to \max_\theta$$



PPO Objective

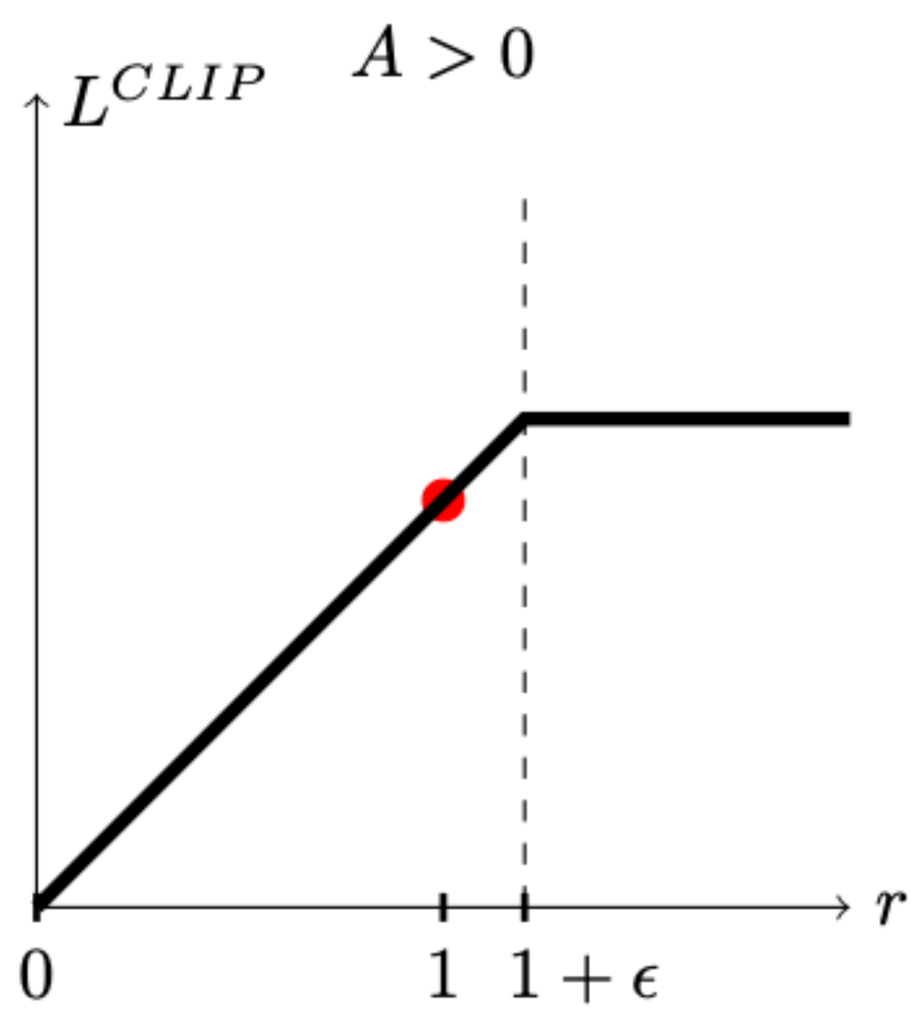
$$r(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)} \qquad r^{CLIP}(\theta) = clip(\frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)}, 1-\varepsilon, 1+\varepsilon)$$

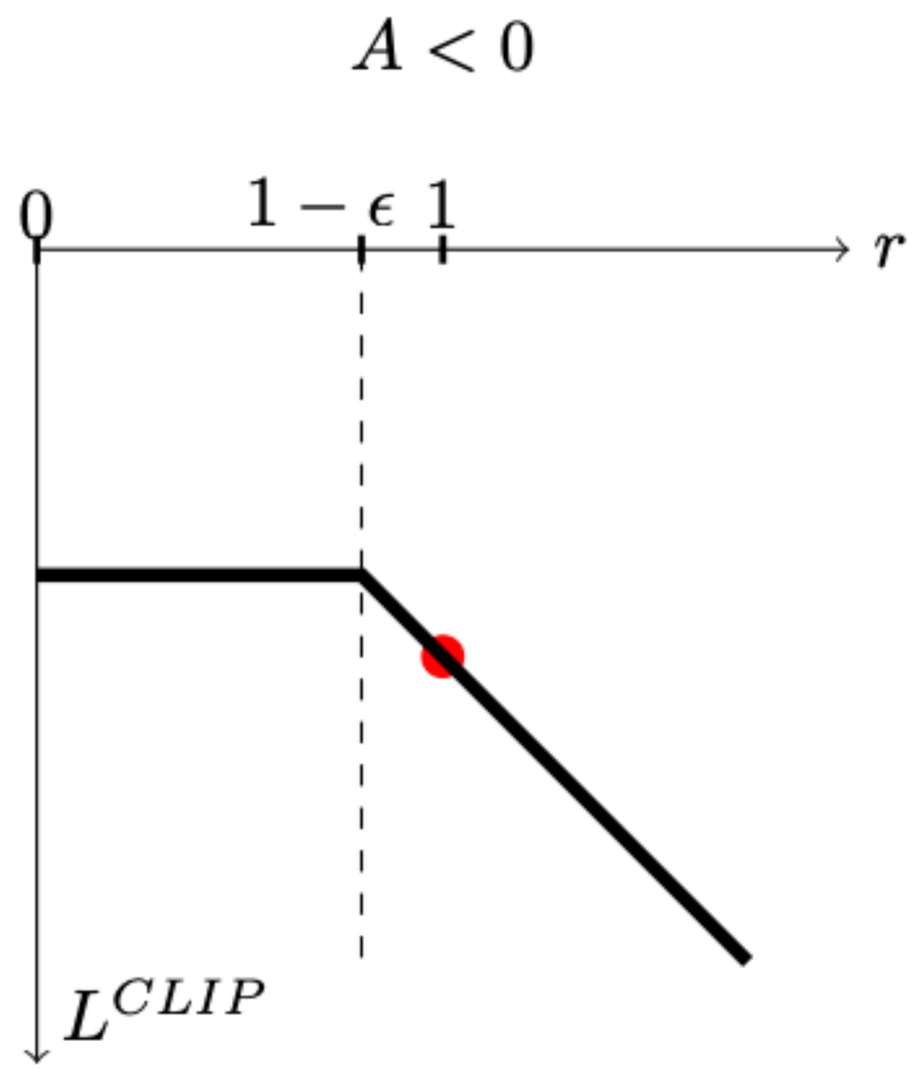
$$L_{\pi_{old}}(\theta) = \mathbb{E}_{s\sim d_{\pi_{old}}}\mathbb{E}_{a\sim\pi_{old}(.|s)}[r(\theta)A^{\pi_{old}}(s,a)]$$

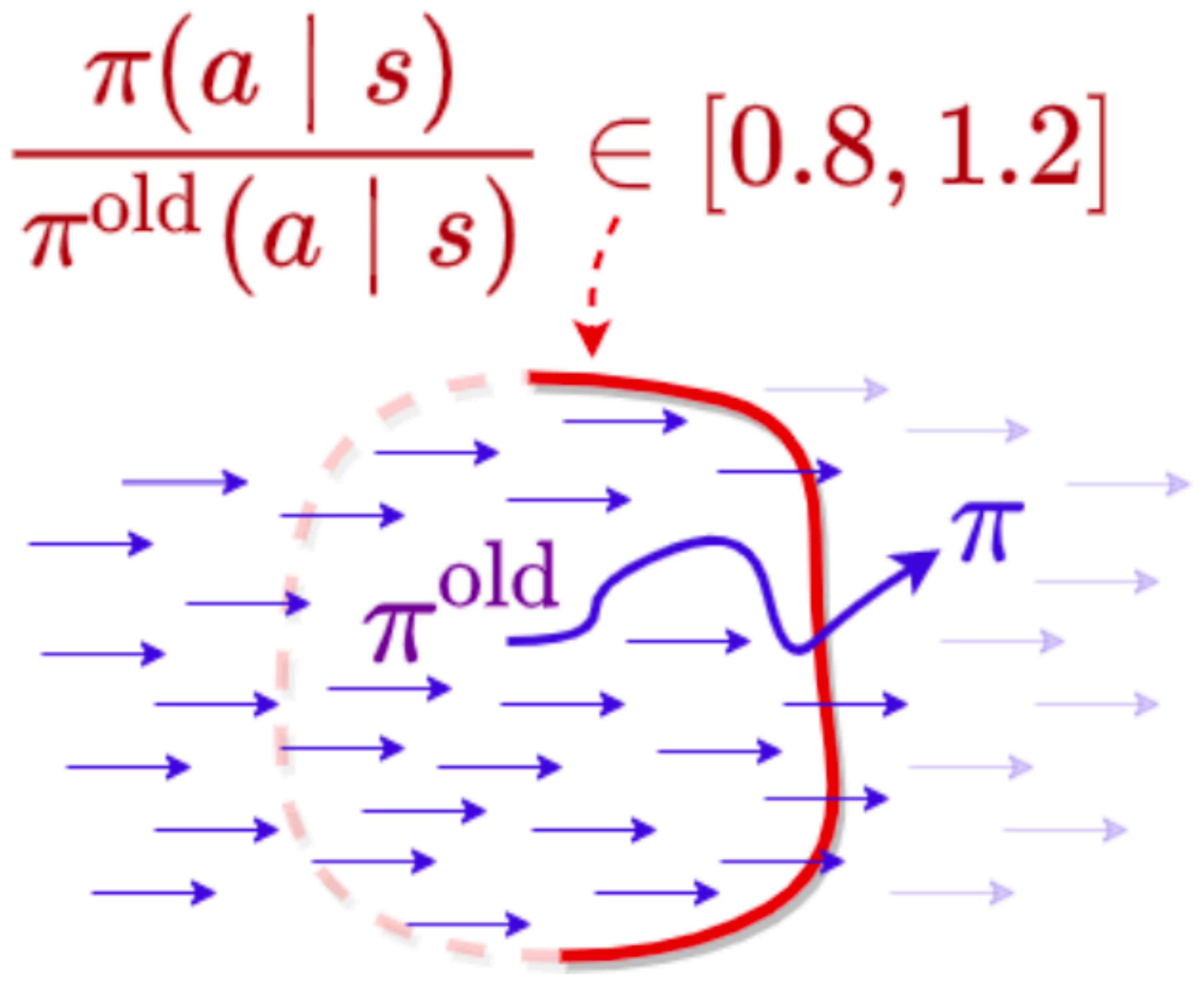
$$L^{CLIP}_{\pi_{old}}(\theta) = \mathbb{E}_{s\sim d_{\pi_{old}}}\mathbb{E}_{a\sim\pi_{old}(.|s)}[\min(r(\theta)A^{\pi_{old}}(s,a), r^{CLIP}(\theta)A^{\pi_{old}}(s,a))]$$

PPO Gradient

$$\min(r(\theta)A^{\pi_{old}}(s,a), r^{CLIP}(\theta)A^{\pi_{old}}(s,a))$$

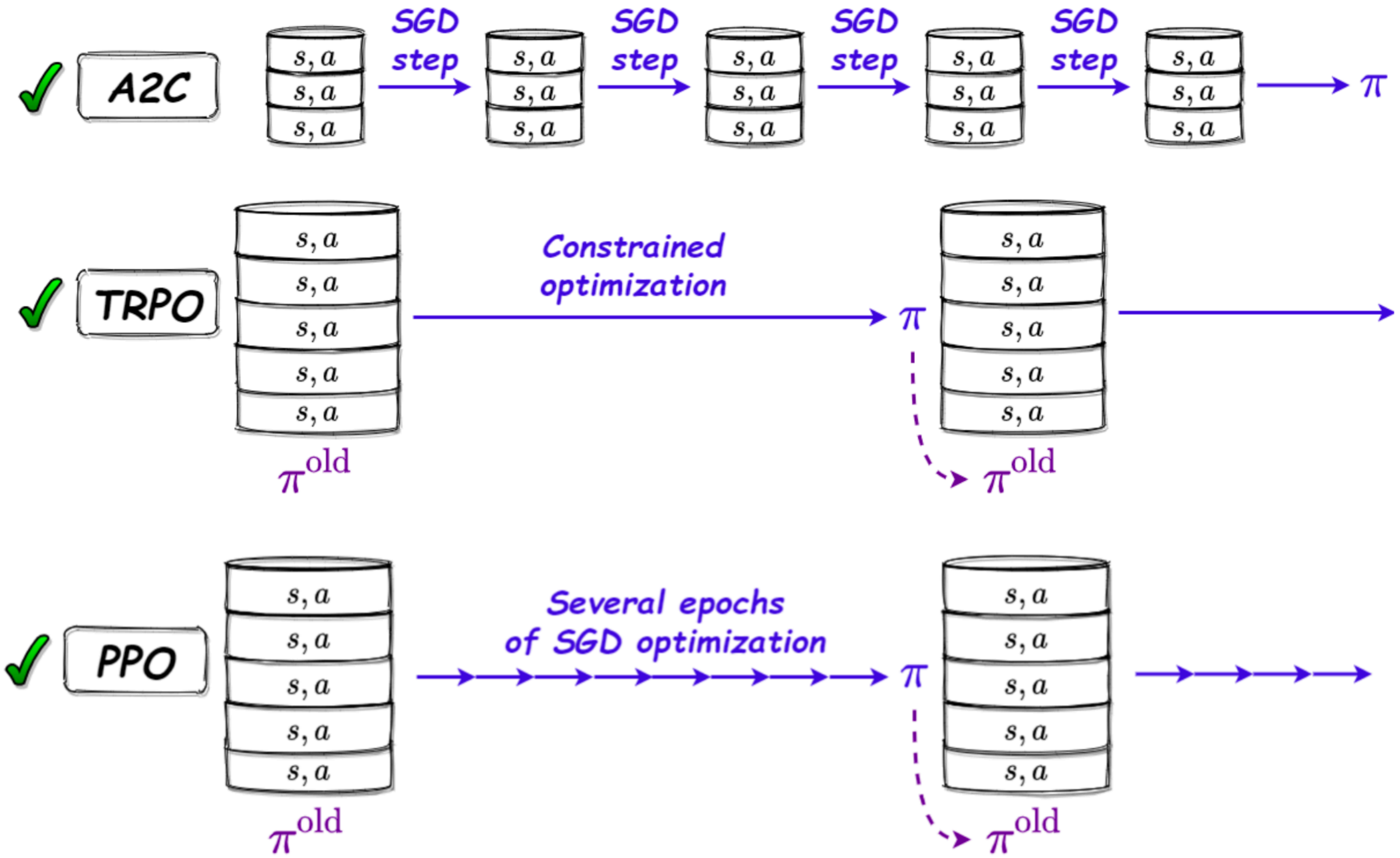






Source

Comparison



A2C
s, a
SGD step
π
TRPO
Constrained optimization
π^old
PPO
Several epochs of SGD optimization

TRPO vs PPO

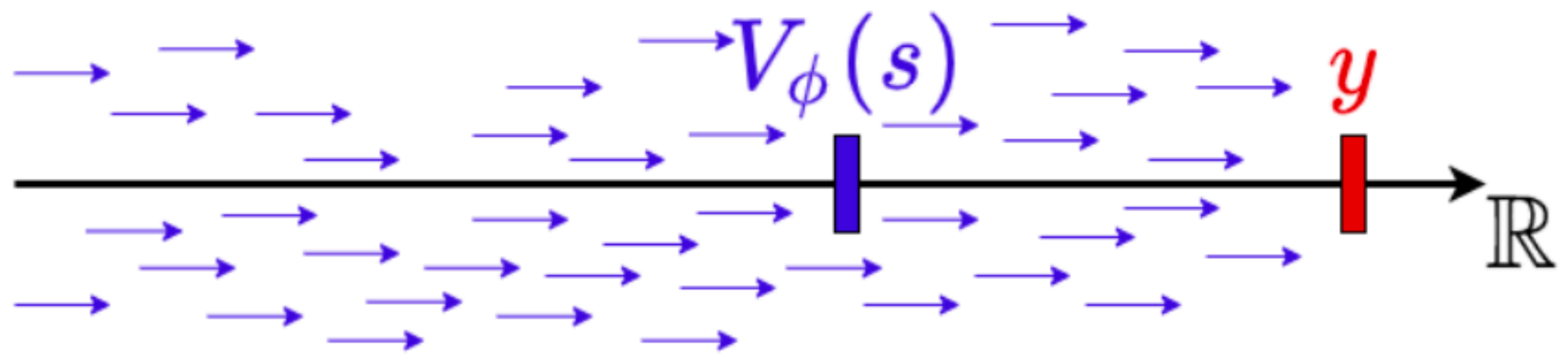
+ Very stable
- Works only for small models
- Hard to implement

+ Relatively easy to implement
+ Works for big models
+ Works better than TRPO
- Many code-level optimisations

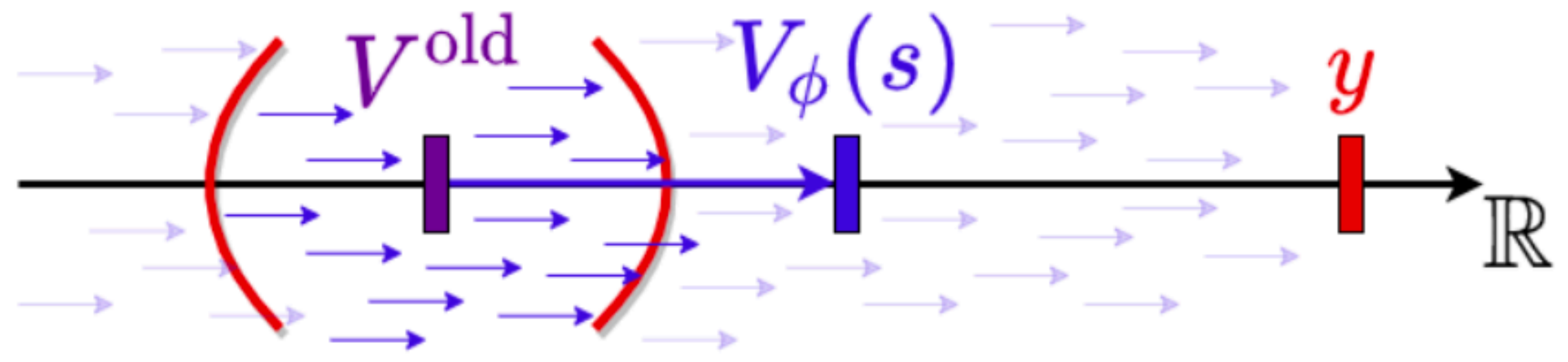
PPO Code-level Optimisations

- Value function clipping

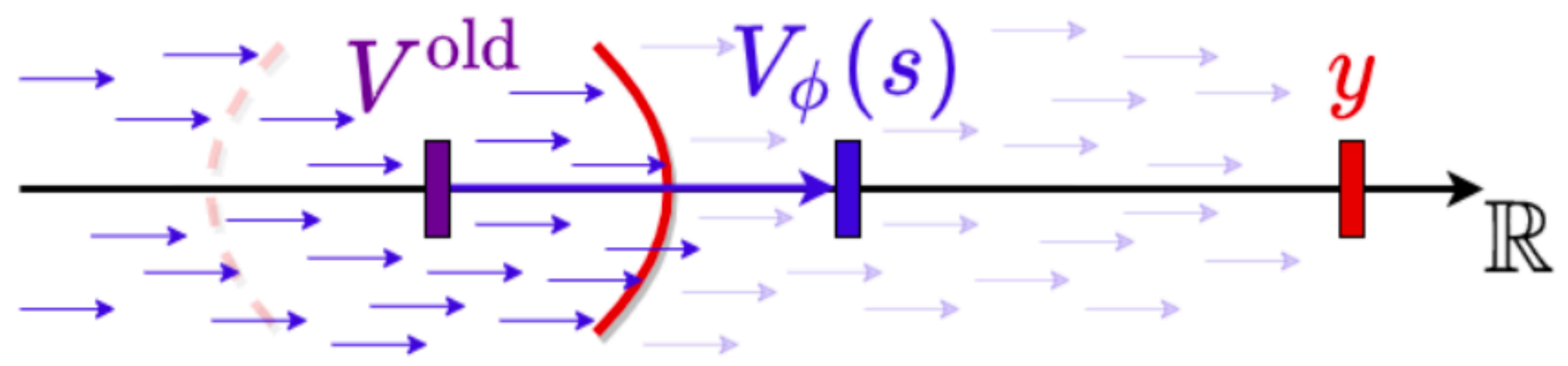
$$L^{Critic}(\theta) = \max[(V_\theta - y)^2, (clip(V_\theta - V_{\theta_{old}}, -\varepsilon, \varepsilon) - (y - V_{\theta_{old}}))^2]$$



$$(V_\theta - y)^2$$



$$clip(V_\theta - V_{\theta_{old}}, -\varepsilon, \varepsilon) - (y - V_{\theta_{old}}))^2$$



$$L^{Critic}(\theta)$$

PPO Code-level Optimisations

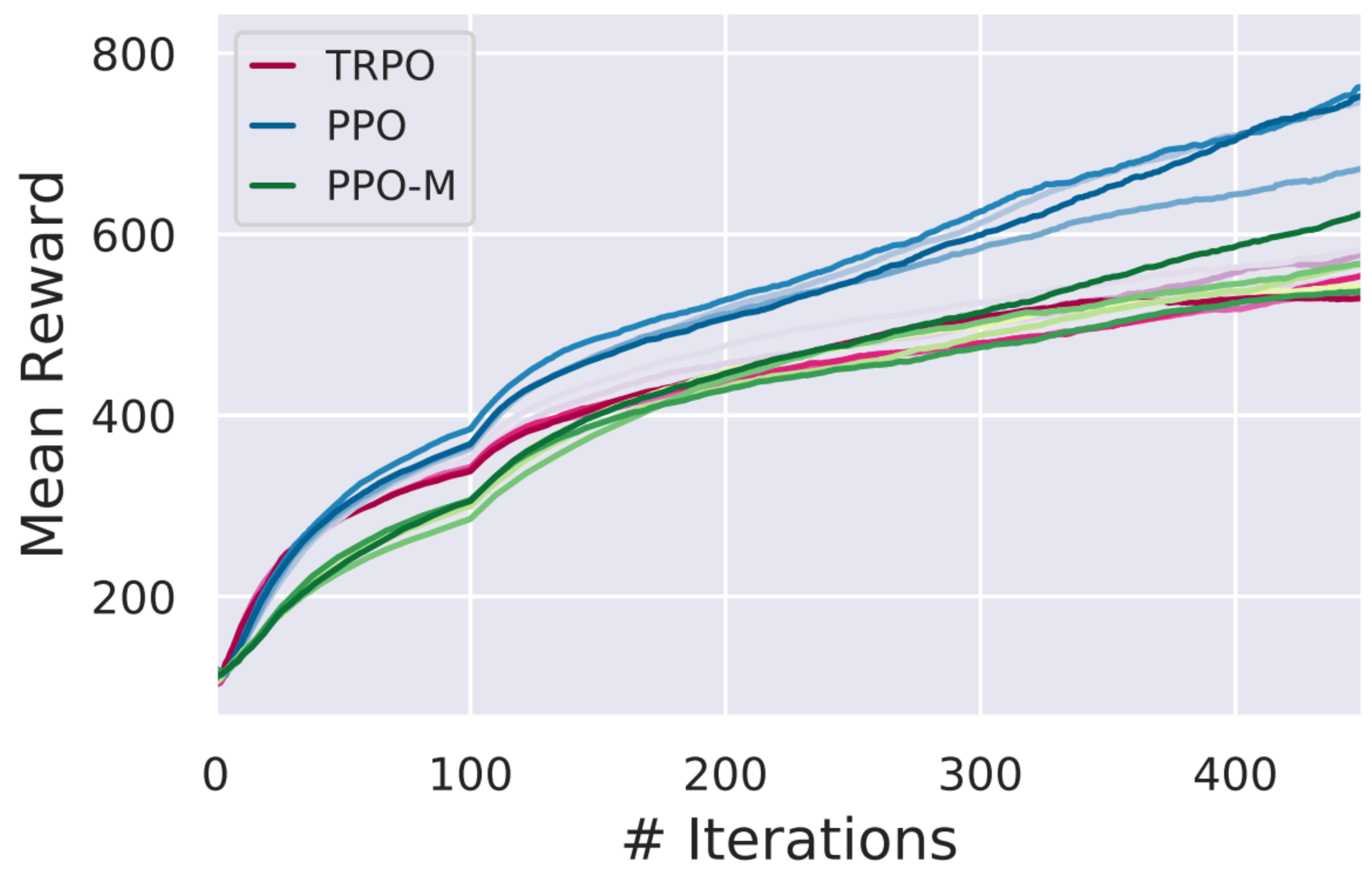
- Reward scaling
- Orthogonal initialization and layer scaling
- Adam learning rate annealing
- Reward Clipping
- Observation Normalization
- Hyperbolic tan activation
- Global Gradient Clipping

TRPO vs PPO

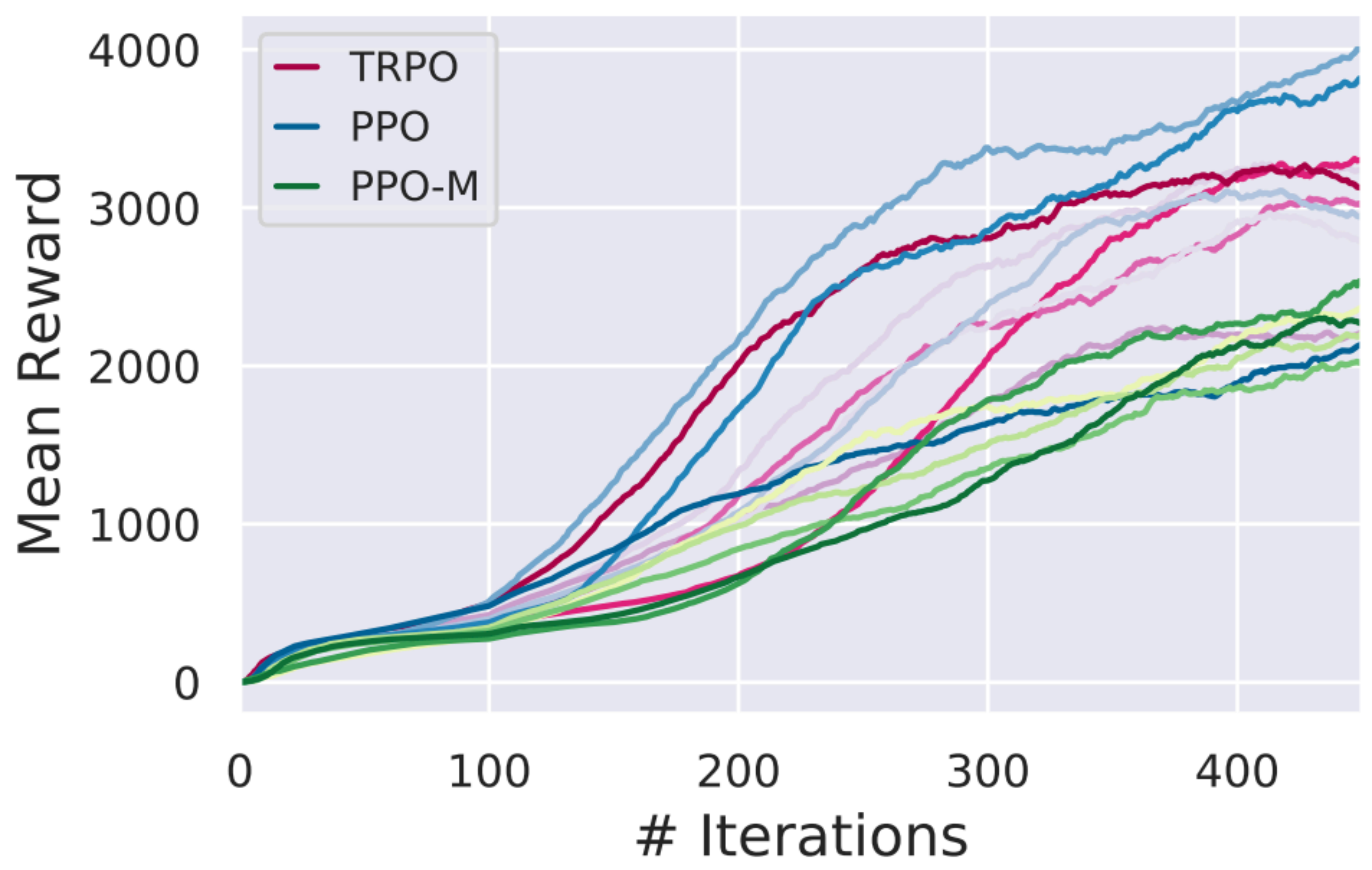
+ Very stable
- Works only for small models
- Hard to implement

+ Relatively easy to implement
+ Works for big models
+ Works better than TRPO
- Many code-level optimisations

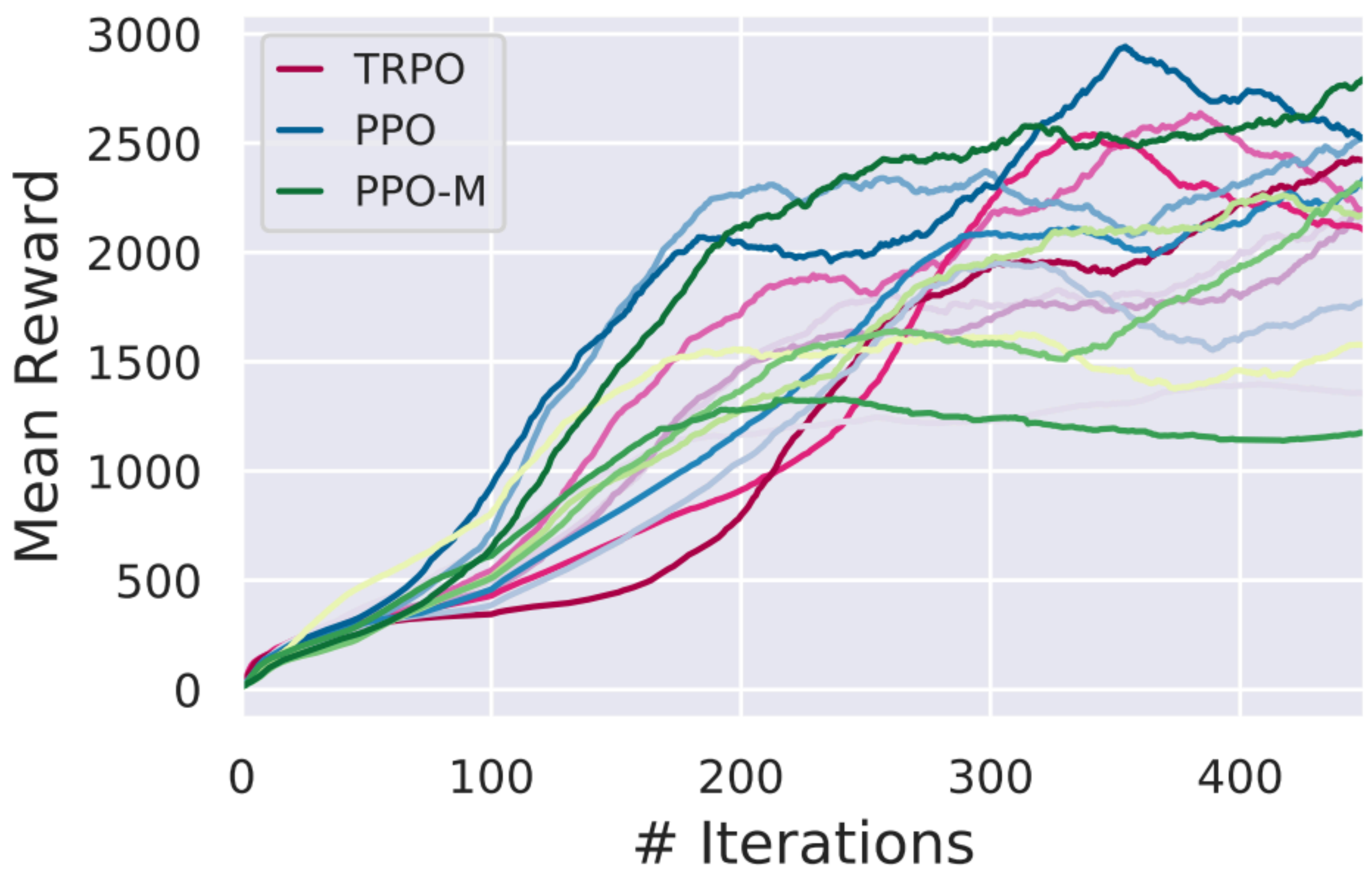
Humanoid-v2



Walker2d-v2



Hopper-v2



Background

1. Practical RL course by YSDA, week 9
2. Reinforcement Learning Textbook (in Russian): 5.3
3. https://spinningup.openai.com/en/latest/algorithms/trpo.html
4. https://spinningup.openai.com/en/latest/algorithms/ppo.html
5. Implementation Matters in Deep RL
6. What Matters In On-Policy Reinforcement Learning?
7. 37 implementation details of PPO

Thank you for your attention!