Lecture 12: Modern DRL

Ziyu Shao

School of Information Science and Technology
ShanghaiTech University

May 09, 2025

Outline

State of the Art for DRL

- Policy Gradient $\rightarrow$ TRPO $\rightarrow$ PPO
  - TRPO: "Trust region policy optimization", Schulman et al. 2015
  - PPO: "Proximal policy optimization algorithms", Schulman et al. 2017
- Q-learning $\rightarrow$ DDPG $\rightarrow$ TD3 $\rightarrow$ SAC
  - DDPG: "Deterministic Policy Gradient Algorithms", Silver et al 2014
  - TD3: "Addressing Function Approximation Error in Actor-Critic Methods", Fujimoto et al. 2018
  - SAC: "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor", Haarnoja et al. 2018

Outline

Trust Region Optimization Method

- Optimizing high-dimensional functions
  - break the problem up into a series of smaller, easier tasks
  - leading to a sequence of successive approximations to the optimizer

Trust Region Optimization Method

- Suppose we want to maximize a function $f(x)$ when $x \in \mathcal{X}$
- Gradient ascend method: a search direction and step length

$$x_{k+1} = x_k + \alpha_k \nabla_x f(x)|_{x=x_k}$$

- Trust-region method:
  - approximates the function f with some simpler function m_k (called the model function or surrogate function) in a neighborhood of x_k
  - $\mathcal{N}(x_k)$ denotes the neighborhood of x_k, ~~minimizing~~ maximize m_k within $\mathcal{N}(x_k)$ and set x_{k+1} equal to this minimizer

$$x_{k+1} = \arg \max_{x \in \mathcal{N}(x_k)} m_k(x)$$

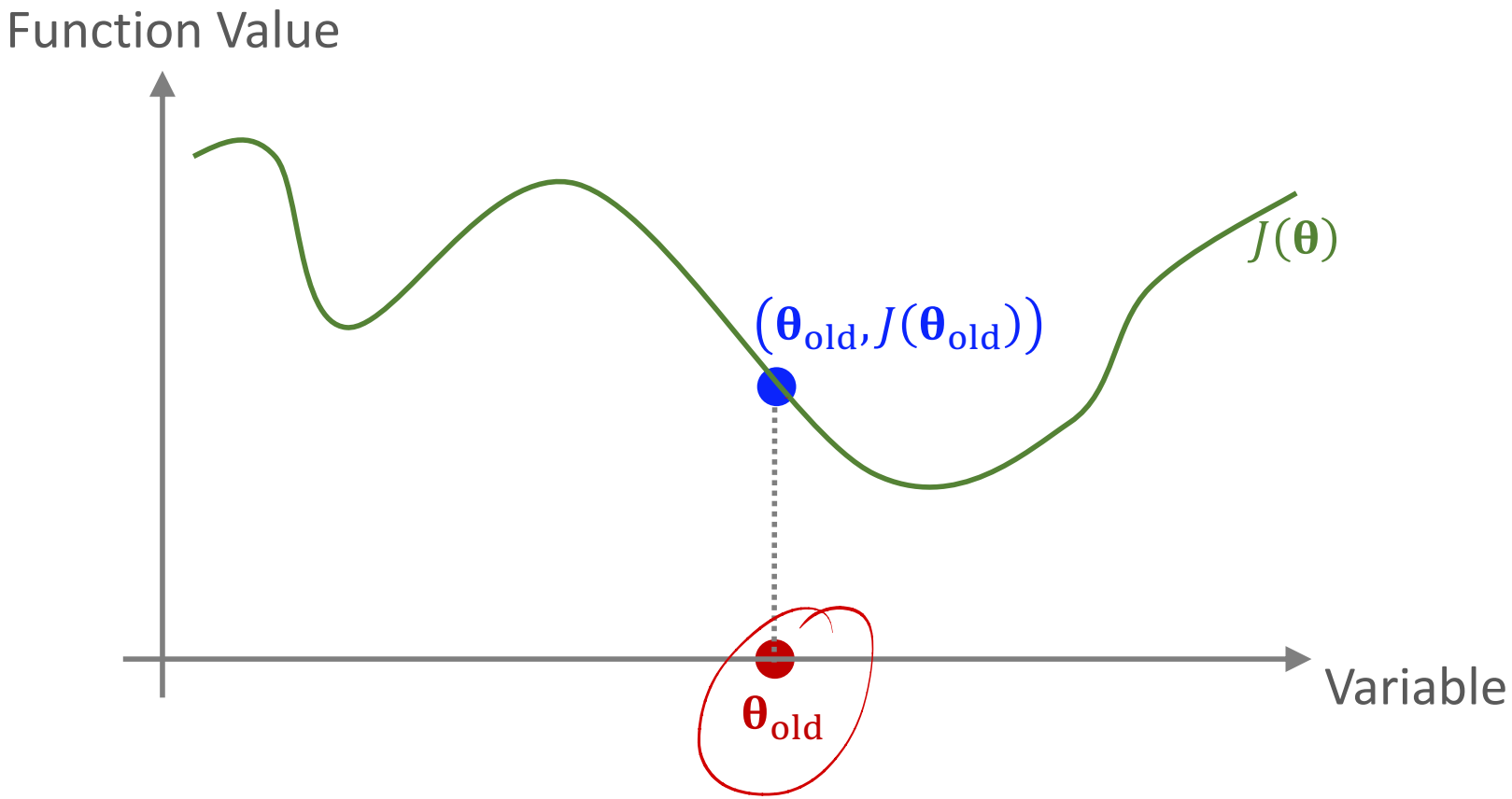
  - $\mathcal{N}(x_k)$ is called " trust region" because we trust that the surrogate function m_k gives a reasonably accurate approximation of f on this region

Trust Region Optimization Method for RL

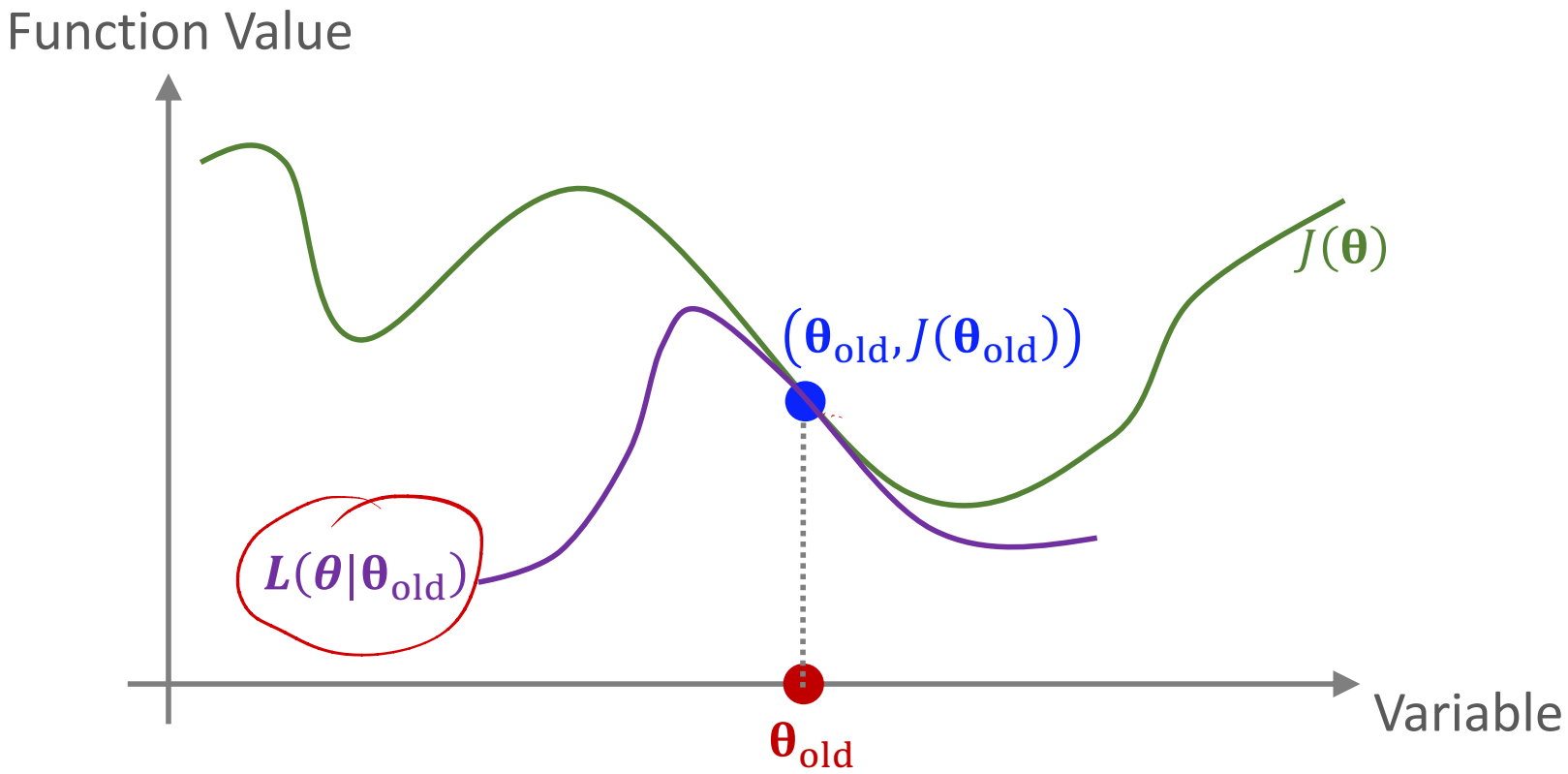
- Object function $J(\theta)$: find $\theta^* = \arg\max_\theta J(\theta)$
- Trust region algorithms repeat
  1. Approximation: given θ_{old}, construct surrogate function $L(\theta|\theta_{\text{old}})$, which is an approximation to $J(\theta)$ in $\mathcal{N}(\theta_{\text{old}})$. Usually L could be the linear or quadratic approximation of J based on its Taylor Series expansion about the point θ_{old}, or Monte Carlo approximation.
  2. Maximization: In the trust region $\mathcal{N}(\theta_{\text{old}})$, find θ_{new} by

$$\theta_{\text{new}} \leftarrow \arg \max_{\theta \in \mathcal{N}(\theta_{\text{old}})} L(\theta|\theta_{\text{old}})$$

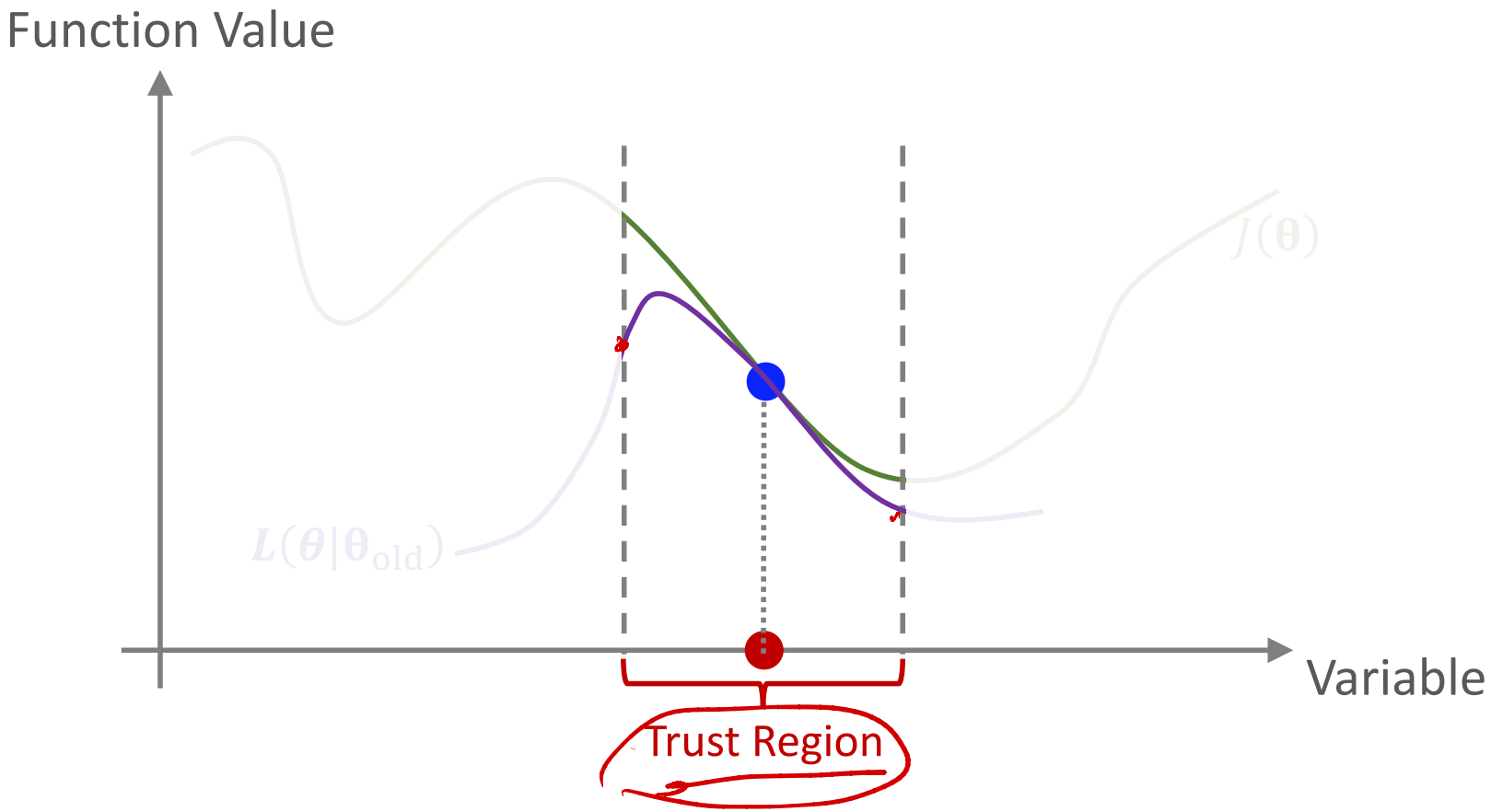
Trust Region Optimization Method for RL



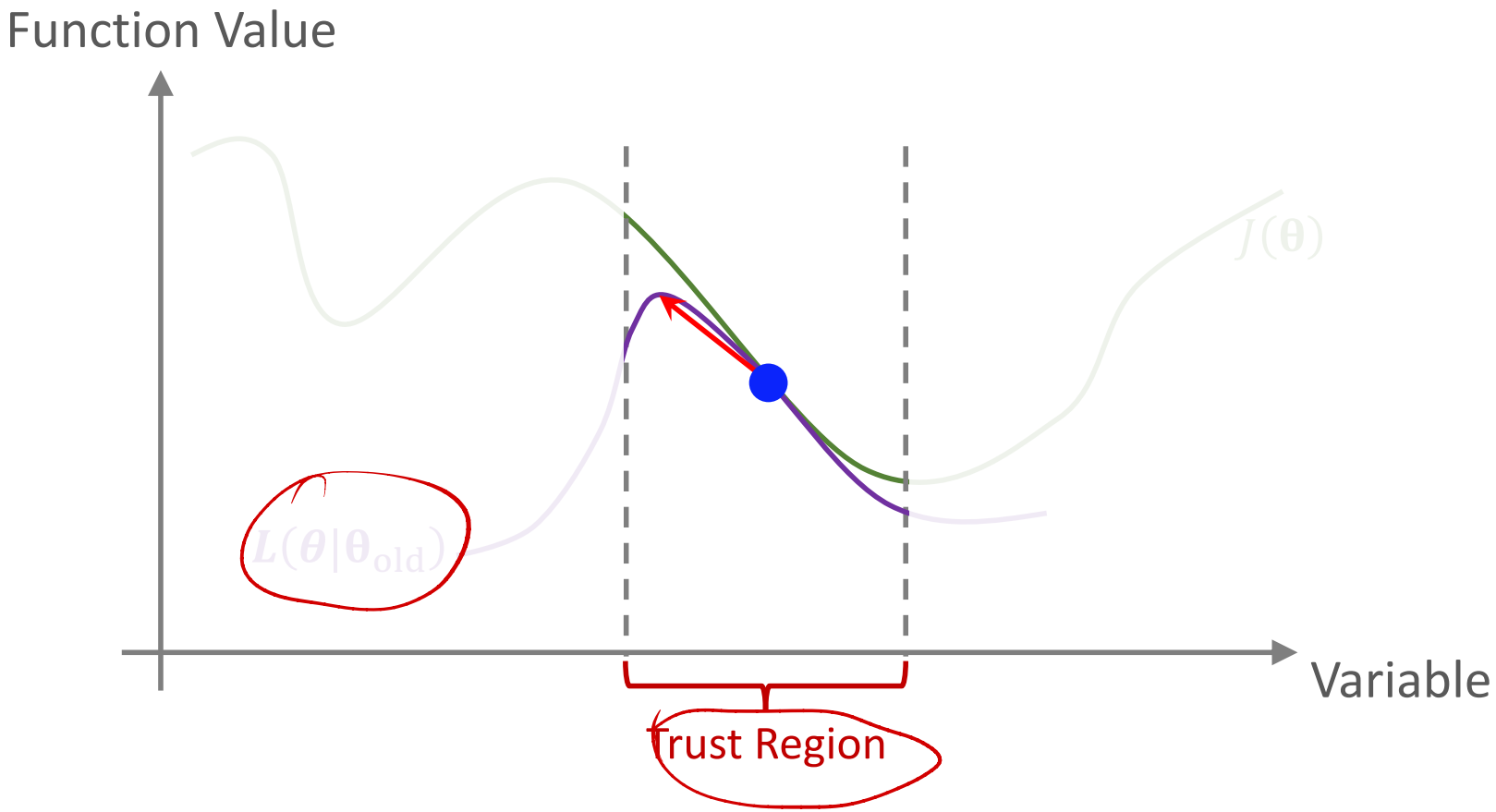
Trust Region Optimization Method for RL



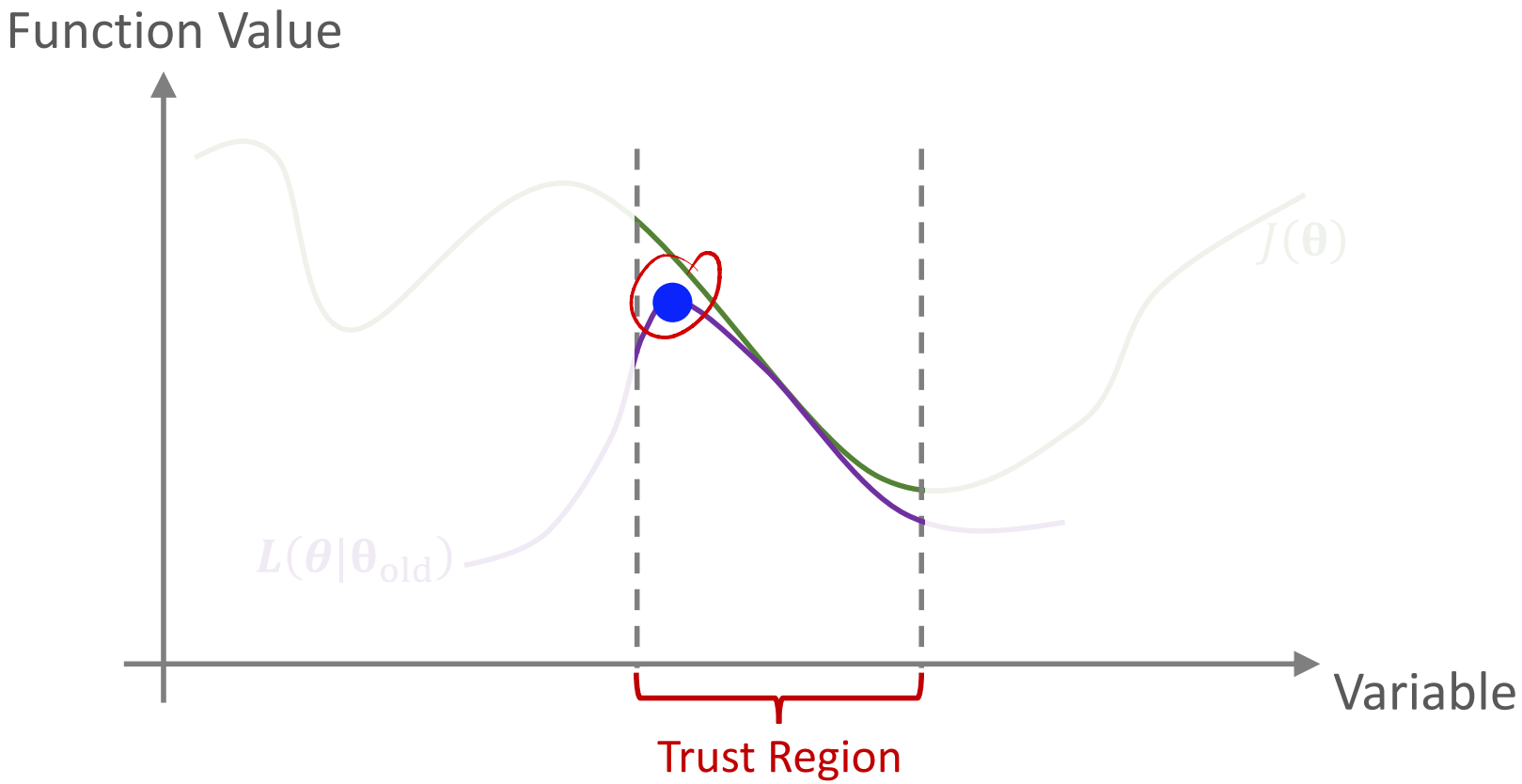
Trust Region Optimization Method for RL



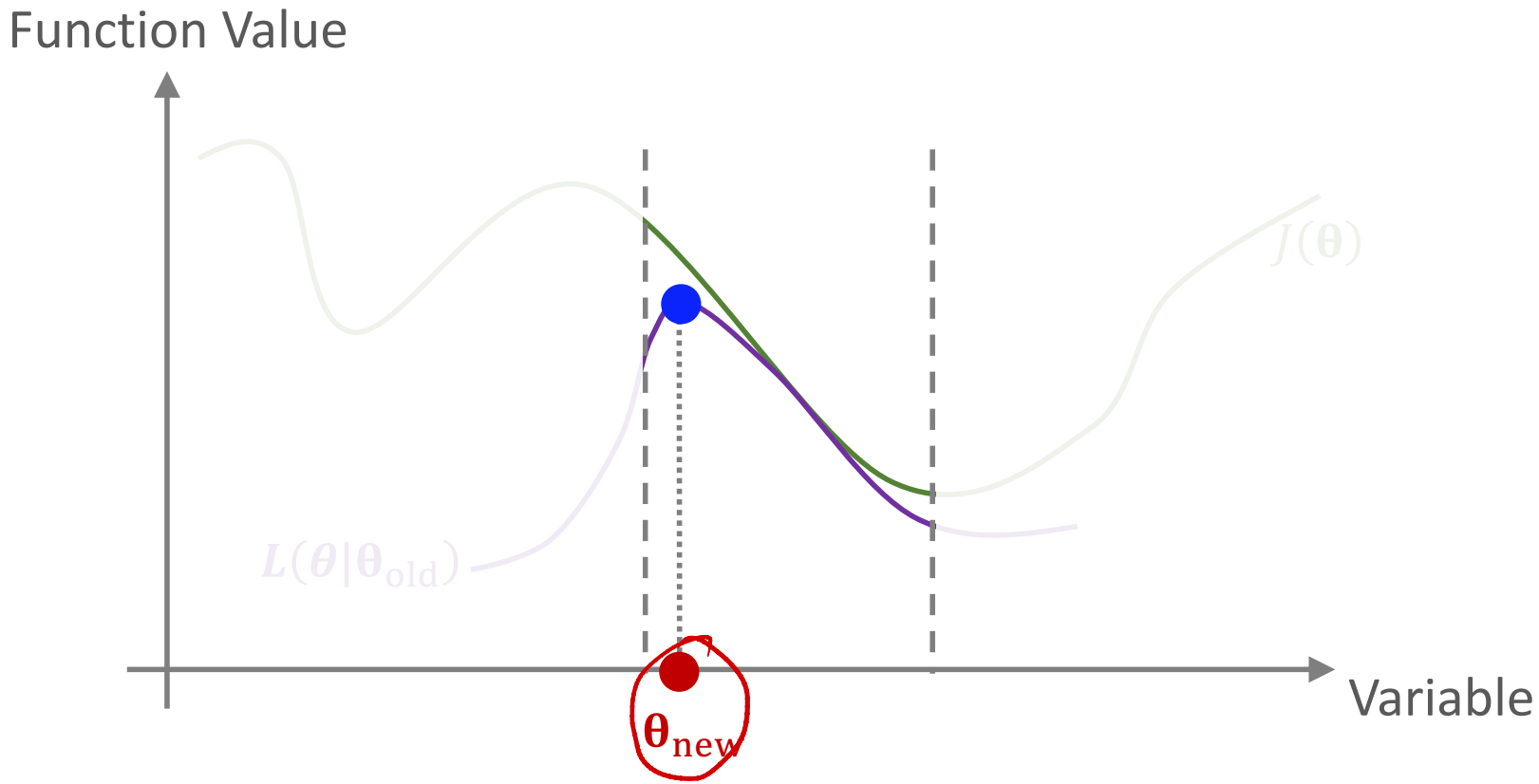
Trust Region Optimization Method for RL



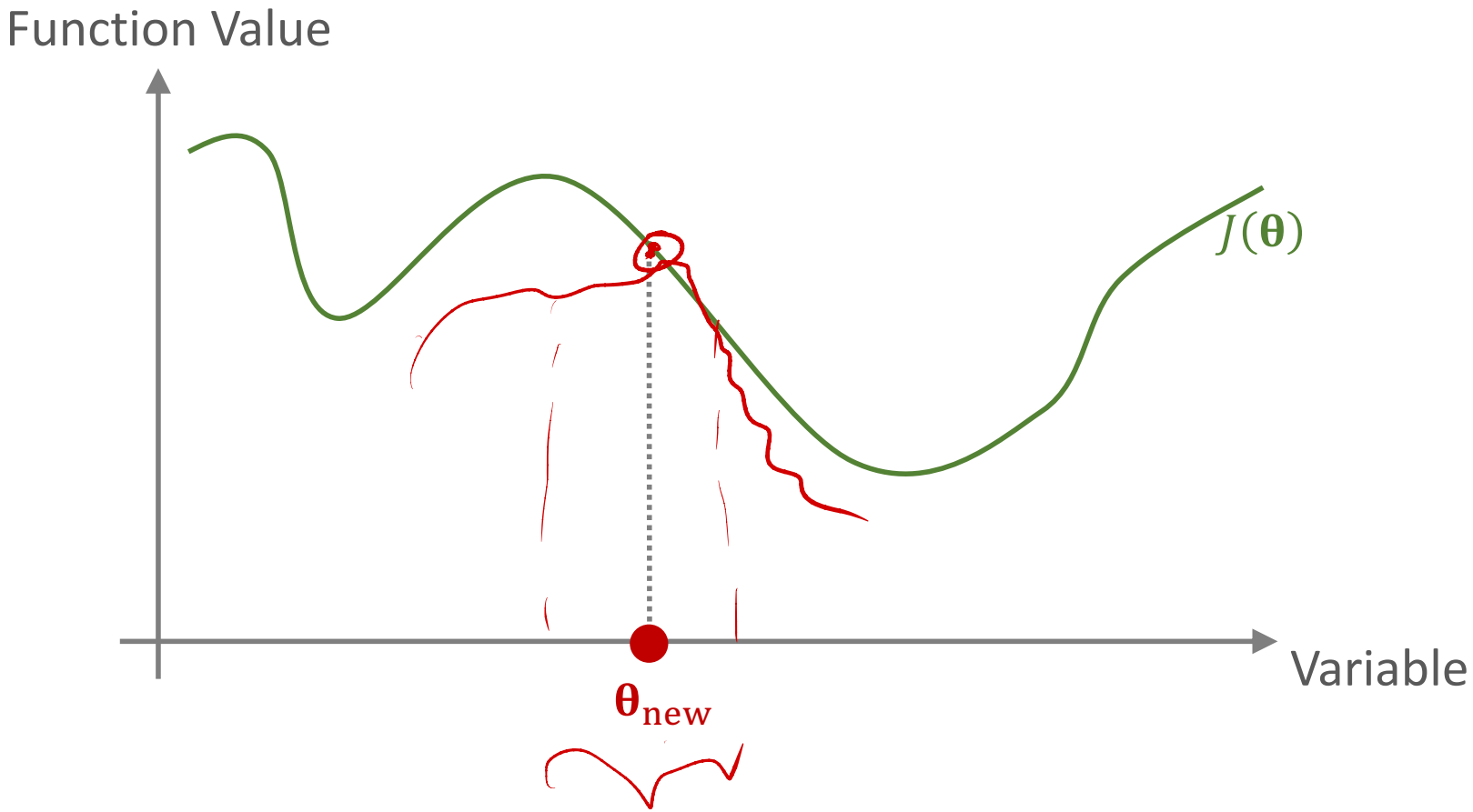
Trust Region Optimization Method for RL



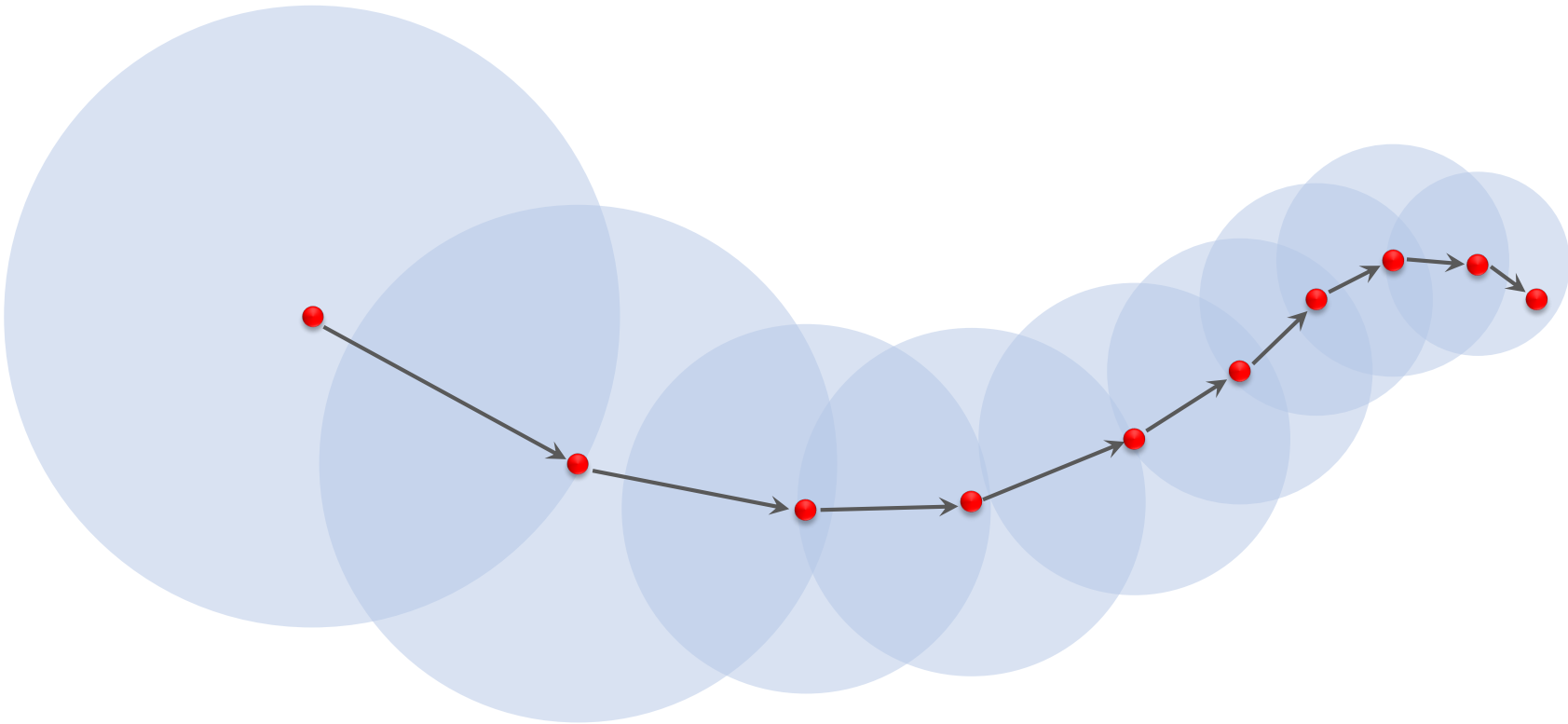
Trust Region Optimization Method for RL



Trust Region Optimization Method for RL



Trust Region Optimization Method for RL



Surrogate Function

- From the off-policy learning using importance sampling

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{S\sim d^\beta, A\sim\beta(\cdot|S)}\left[\frac{\pi_\theta(A|S)}{\beta(A|S)} A^{\pi_\theta}(S,A)\nabla_\theta \ln \pi_\theta(A|S)\right]$$

- $\beta(a|s)$: behavior policy for collecting samples (a known policy)
- $d^\beta(s) = \lim_{t\to\infty} P(S_t = s|S_0, \beta)$ is the stationary distribution of the behavior policy $\beta(a|s)$
- Now set the behavior policy $\beta(a|s)$ as the policy $\pi_{\theta_{\text{old}}}(a|s)$, then

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{S\sim d^{\pi_{\theta_{\text{old}}}}, A\sim\pi_{\theta_{\text{old}}}(\cdot|S)}\left[\frac{\pi_\theta(A|S)}{\pi_{\theta_{\text{old}}}(A|S)} A^{\pi_\theta}(S,A)\nabla_\theta \ln \pi_\theta(A|S)\right]$$

Surrogate Function

- Then we have

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{S\sim d^{\pi_{\theta_{\text{old}}}}, A\sim\pi_{\theta_{\text{old}}}(\cdot|S)}\left[\frac{\pi_\theta(A|S)}{\pi_{\theta_{\text{old}}}(A|S)} A^{\pi_\theta}(S,A)\nabla_\theta \ln \pi_\theta(A|S)\right]$$

$$= \mathbb{E}_{S\sim d^{\pi_{\theta_{\text{old}}}}, A\sim\pi_{\theta_{\text{old}}}(\cdot|S)}\left[\frac{\nabla_\theta \pi_\theta(A|S)}{\pi_{\theta_{\text{old}}}(A|S)} A^{\pi_\theta}(S,A)\right]$$

$$\approx \mathbb{E}_{S\sim d^{\pi_{\theta_{\text{old}}}}, A\sim\pi_{\theta_{\text{old}}}(\cdot|S)}\left[\frac{\nabla_\theta \pi_\theta(A|S)}{\pi_{\theta_{\text{old}}}(A|S)} A^{\pi_{\theta_{\text{old}}}}(S,A)\right]$$

- Thus

$$J(\theta) \approx \mathbb{E}_{S\sim d^{\pi_{\theta_{\text{old}}}}, A\sim\pi_{\theta_{\text{old}}}(\cdot|S)}\left[\frac{\pi_\theta(A|S)}{\pi_{\theta_{\text{old}}}(A|S)} A^{\pi_{\theta_{\text{old}}}}(S,A)\right]$$

Surrogate Function

- Use $\hat{A}_{\theta_{\text{old}}}(S, A)$ to estimate $A^{\pi_{\theta_{\text{old}}}}$, then the surrogate function $L(\theta|\theta_{\text{old}}) \approx J(\theta)$

$$L(\theta|\theta_{\text{old}}) = \mathbb{E}_{S\sim d^{\pi_{\theta_{\text{old}}}}, A\sim\pi_{\theta_{\text{old}}}(\cdot|S)}\left[\frac{\pi_\theta(A|S)}{\pi_{\theta_{\text{old}}}(A|S)}\hat{A}_{\theta_{\text{old}}}(S, A)\right]$$

- The policy $\pi_{\theta_{\text{old}}}$ and policy π_θ should be close to each other.
- We add a trust region constraint which enforces the distance between old and new policies measured by KL-divergence to be small enough, within a parameter δ:

$$\mathbb{E}_{S\sim d^{\pi_{\theta_{\text{old}}}}}\left[D_{\text{KL}}(\pi_{\theta_{\text{old}}}(.|S)\|\pi_\theta(.|S))\right] \leq \delta$$

where given $S = s$

$$D_{\text{KL}}(\pi_{\theta_{\text{old}}}(.|s)\|\pi_\theta(.|s)) = \sum_{a\in\mathcal{A}} \pi_{\theta_{\text{old}}}(a|s) \log\left(\frac{\pi_{\theta_{\text{old}}}(a|s)}{\pi_\theta(a|s)}\right)$$

Core of TRPO

- Maximization part in trust-region method of RL:

$$\begin{aligned} \text{maximize}_\theta \quad & L(\theta|\theta_{\text{old}}) = \mathbb{E}_{S,A}\left[\frac{\pi_\theta(A|S)}{\pi_{\theta_{\text{old}}}(A|S)}\hat{A}_{\theta_{\text{old}}}(S,A)\right] \\ \text{subject to} \quad & \mathbb{E}_S[D_{\text{KL}}(\pi_{\theta_{\text{old}}}(.|S)\|\pi_\theta(.|S))] \leq \delta \end{aligned}$$

- Collecting samples from policy $\pi_{\theta_{\text{old}}}$, and let $\hat{E}_t$ denote the empirical average over a finite batch of samples, then we have sample-version maximization problem

$$\begin{aligned} \text{maximize}_\theta \quad & \hat{E}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}A_t\right] \\ \text{subject to} \quad & \hat{E}_t[D_{KL}(\pi_{\theta_{\text{old}}}(\cdot|s_t),\pi_\theta(\cdot|s_t))] \leq \delta \end{aligned}$$

- Given a mini-batch of N sample trajectories or N sample time-steps,

$$\hat{E}_t[f(t)] \approx \frac{1}{N}\sum_{i=1}^{N} f(t^{(i)}).$$

Increasing the Robustness with Trust Regions

- In the trust region, we limit our parameter search within a region controlled by δ

Gradient ascend

Trust region

Trust Region Optimization

- Following Taylors series expansion on both terms above up to the second-order
- After some derivations we can have

$$L_{\theta_k}(\theta) \approx g^T(\theta - \theta_k)$$

$$D_{KL}(\theta \| \theta_k) \approx \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k)$$

where $g = \nabla_\theta L_{\theta_k}(\theta)$ and $H = \nabla^2_\theta D_{KL}(\theta \| \theta_k)$ and $\theta_k = \theta_{\text{old}}$ is the old policy parameter

- Then the objective turns to:

$$\theta_{k+1} = \arg\max_\theta g^T(\theta - \theta_k) \text{ s.t. } \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \leq \delta$$

- This is a quadratic equation and can be solved analytically:

$$\theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{g^T H^{-1} g}} H^{-1} g$$

Limitations of TRPO

- Scalability issue for TRPO
  - H is the Fisher Information Matrix (FIM). Computing H every time for the current policy model is expensive
  - It requires a large batch of rollouts to approximate H

$$H = E_{X\sim\pi_\theta}\left[(\nabla_\theta \log \pi_\theta(X))^T(\nabla_\theta \log \pi_\theta(X))\right]$$

  - Conjugate Gradient(CG) makes implementation more complicated
- TRPO does not work well for deep networks

Outline

Proximal Policy Optimization (PPO)

- Simplification of TRPO: better performance with must faster speed and much higher robustness
- Avoid using second order information (e.g. Fisher Information Matrix)
- Pure PPO has two variants
  1. PPO1: put the hard KL constraint into the surrogate function(object function)
  2. PPO2: clipping the surrogate function directly without the hard KL constraint
- PPO default refers to PPO2.

PPO1

- Surrogate function with adaptive KL Penality
- Corresponding Unconstrained optimization problem:

$$\text{maximize}_\theta \ \mathbb{E}_{S,A}\left[\frac{\pi_\theta(A|S)}{\pi_{\theta_{\text{old}}}(A|S)}\hat{A}_{\theta_{\text{old}}}(S,A)\right] - \beta\mathbb{E}_S[D_{\text{KL}}(\pi_{\theta_{\text{old}}}(.|S)\|\pi_\theta(.|S))]$$

- Sample Version:

$$\text{maximize}_\theta \ \mathbb{E}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}A_t\right] - \beta\mathbb{E}_t[D_{KL}[\pi_{\theta_{\text{old}}}(\cdot|s_t), \pi_\theta(\cdot|s_t)]]$$

- Increase β when KL value is large, decrease β when KL value is small

PPO1

Algorithm 4 PPO with Adaptive KL Penalty

Input: initial policy parameters θ_0, initial KL penalty β_0, target KL-divergence δ
for $k = 0, 1, 2, \ldots$ **do**
  Collect set of partial trajectories $\mathcal{D}_k$ on policy $\pi_k = \pi(\theta_k)$
  Estimate advantages $\hat{A}_t^{\pi_k}$ using any advantage estimation algorithm
  Compute policy update

$$\theta_{k+1} = \arg\max_{\theta} \mathcal{L}_{\theta_k}(\theta) - \beta_k \bar{D}_{KL}(\theta||\theta_k)$$

  by taking K steps of minibatch SGD (via Adam)
  if $\bar{D}_{KL}(\theta_{k+1}||\theta_k) \geq 1.5\delta$ **then**
    $\beta_{k+1} = 2\beta_k$
  else if $\bar{D}_{KL}(\theta_{k+1}||\theta_k) \leq \delta/1.5$ **then**
    $\beta_{k+1} = \beta_k/2$
  end if
end for

- Same performance as TRPO, but solved much faster by first-order optimization (SGD)

PPO2: Clipping

- Let $r(\theta)$ denote the probability ratio

$$r(\theta) = r(\theta)(a|s) = \frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}$$

- Now surrogate function is:

$$L(\theta|\theta_{\text{old}}) = \mathbb{E}_{S,A}\left[r(\theta)\hat{A}_{\theta_{\text{old}}}(S,A)\right]$$

- Without a limitation on the distance between θ_{old} and θ, maximize surrogate function would lead to instability with extremely large parameter updates and big policy ratios.
- PPO2 imposes the constraint by forcing $r(\theta)$ to stay within a small interval around 1, precisely $[1-\epsilon, 1+\epsilon]$, where ϵ is a hyperparameter.

PPO2: Clipping

- A new surrogate function to clip the estimated advantage function if the new policy is far away from the old policy ($r(\theta)$ is too large)

$$L^{\text{CLIP}}(\theta|\theta_{\text{old}}) = \mathbb{E}[\min(r(\theta)\hat{A}_{\theta_{\text{old}}}, \text{clip}(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{\theta_{\text{old}}})]$$

- The function $\text{clip}(r(\theta), 1-\epsilon, 1+\epsilon)$ clips the ratio to be no more than $1+\epsilon$ and no less than $1-\epsilon$.
- Such new surrogate function of PPO2 takes the minimum one between the original value and the clipped version
- If the probability ratio between the new policy and the old policy falls outside the range $(1-\epsilon)$ and $(1+\epsilon)$, the advantage function will be clipped.
- We lose the motivation for increasing the policy update to extremes for better rewards.
- ϵ is set to 0.2.

Understanding the clipping

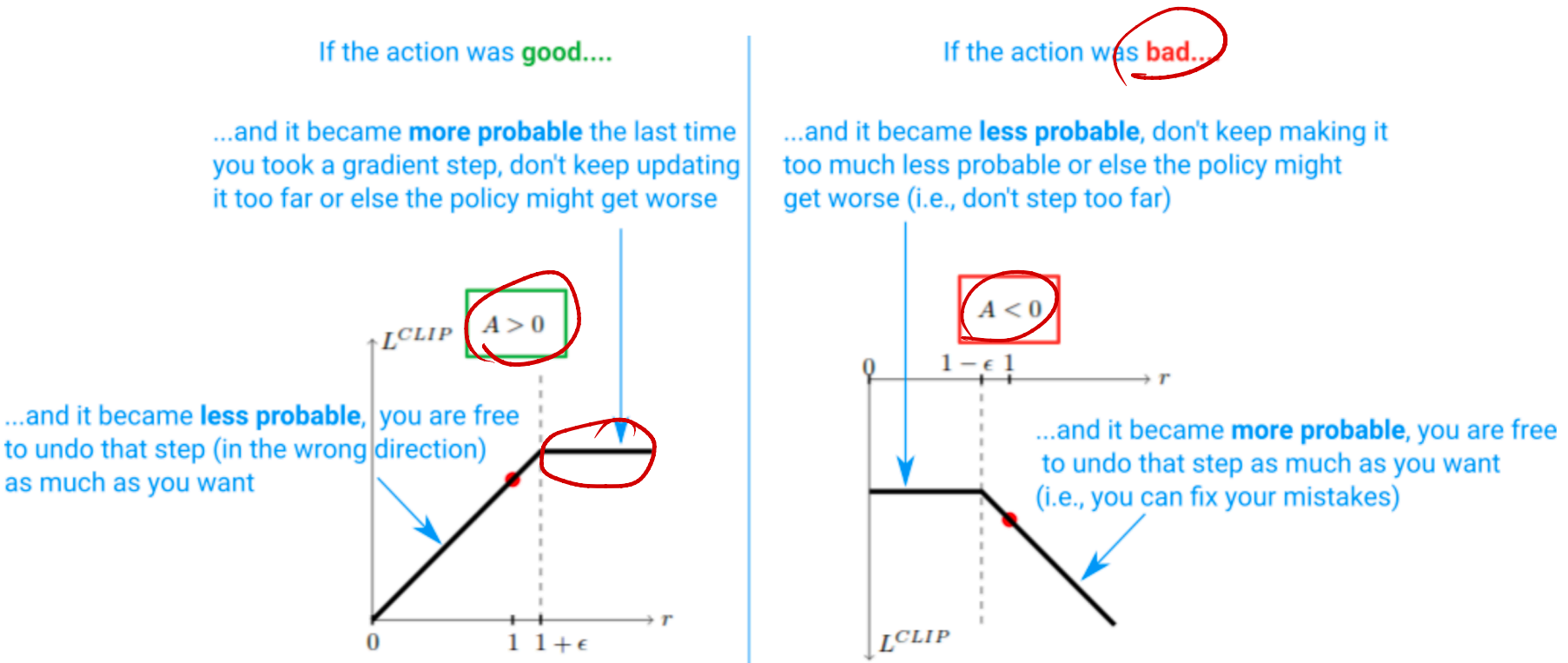


Figure 1: Plots showing one term (i.e., a single timestep) of the surrogate function L^{CLIP} as a function of the probability ratio r, for positive advantages (left) and negative advantages (right). The red circle on each plot shows the starting point for the optimization, i.e., $r = 1$. Note that L^{CLIP} sums many of these terms.

Understanding the clipping

- $\hat{A}(a|s) > 0$: action a is good, then increase $\pi_\theta(a|s)$, equivalently increasing $r(\theta)$, but no more than $1+\epsilon$
- $\hat{A}(a|s) < 0$: action a is bad, then decrease $\pi_\theta(a|s)$, equivalently decreasing $r(\theta)$, but no less than $1-\epsilon$
- Clipping surrogate function:

$$L^{\text{CLIP}}(\theta|\theta_{\text{old}}) = \mathbb{E}\left(\min(r(\theta)\hat{A}_{\theta_{\text{old}}}, \text{clip}(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{\theta_{\text{old}}})\right)$$

$r(\theta) > 0$	$\hat{A}$	Min	Clipped	Surrogate	Gradient
$r(\theta) \in [1-\epsilon, 1+\epsilon]$	$+$	$r(\theta)\hat{A}$	no	$+$	$\surd$
$r(\theta) \in [1-\epsilon, 1+\epsilon]$	$-$	$r(\theta)\hat{A}$	no	$-$	$\surd$
$r(\theta) < 1-\epsilon$	$+$	$r(\theta)\hat{A}$	no	$+$	$\surd$
$r(\theta) < 1-\epsilon$	$-$	$(1-\epsilon)\hat{A}$	yes	$-$	0
$r(\theta) > 1+\epsilon$	$+$	$(1+\epsilon)\hat{A}$	yes	$+$	0
$r(\theta) > 1+\epsilon$	$-$	$r(\theta)\hat{A}$	no	$-$	$\surd$

PPO2: Clipping

Algorithm 5 PPO with Clipped Objective

Input: initial policy parameters θ_0, clipping threshold ϵ
for $k = 0, 1, 2, \ldots$ **do**
  Collect set of partial trajectories $\mathcal{D}_k$ on policy $\pi_k = \pi(\theta_k)$
  Estimate advantages $\hat{A}_t^{\pi_k}$ using any advantage estimation algorithm
  Compute policy update

$$\theta_{k+1} = \arg\max_{\theta} \mathcal{L}_{\theta_k}^{CLIP}(\theta)$$

  by taking K steps of minibatch SGD (via Adam), where

$$\mathcal{L}_{\theta_k}^{CLIP}(\theta) = \mathop{\mathrm{E}}_{\tau \sim \pi_k} \left[\sum_{t=0}^{T} \left[\min(r_t(\theta)\hat{A}_t^{\pi_k}, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t^{\pi_k}) \right] \right]$$

end for

- PPOs have the stability and reliability of trust-region methods but are much simpler to implement, requiring only few lines of code change to a vanilla policy gradient implementation

PPO2: GAE

- TD residual $\delta_t^V = r_t + \gamma V(s_{t+1}) - V(s_t)$
- Then we have advantage function $\hat{A}_t^{(k)}$

$$
\begin{aligned}
\hat{A}_t^{(1)} &:= \delta_t^V && = -V(s_t) + r_t + \gamma V(s_{t+1}) \\
\hat{A}_t^{(2)} &:= \delta_t^V + \gamma \delta_{t+1}^V && = -V(s_t) + r_t + \gamma r_{t+1} + \gamma^2 V(s_{t+2}) \\
\hat{A}_t^{(3)} &:= \delta_t^V + \gamma \delta_{t+1}^V + \gamma^2 \delta_{t+2}^V && = -V(s_t) + r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 V(s_{t+3}) \\
&\vdots \\
\hat{A}_t^{(k)} &:= \sum_{l=0}^{k-1} \gamma^l \delta_{t+l}^V && = -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{k-1} r_{t+k-1} + \gamma^k V(s_{t+k}) \\
\hat{A}_t^{(\infty)} &:= \sum_{l=0}^{\infty} \gamma^l \delta_{t+l}^V && = -V(s_t) + \sum_{l=0}^{\infty} \gamma^l r_{t+l},
\end{aligned}
$$

PPO2: GAE

- Similar to construction of $TD(\lambda)$, we use λ to balance the tradeoff between bias and variance
- The smaller the λ, the smaller the variance, and the larger the bias
- The larger the λ, the larger the variance, and the smaller the bias
- GAE

$$\begin{aligned}
\hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} &:= (1-\lambda)\left(\hat{A}_t^{(1)} + \lambda \hat{A}_t^{(2)} + \lambda^2 \hat{A}_t^{(3)} + \ldots\right) \\
&= (1-\lambda)\left(\delta_t^V + \lambda(\delta_t^V + \gamma\delta_{t+1}^V) + \lambda^2(\delta_t^V + \gamma\delta_{t+1}^V + \gamma^2\delta_{t+2}^V) + \ldots\right) \\
&= (1-\lambda)\left(\delta_t^V(1+\lambda+\lambda^2+\ldots) + \gamma\delta_{t+1}^V(\lambda+\lambda^2+\lambda^3+\ldots)\right. \\
&\quad \left. + \gamma^2\delta_{t+2}^V(\lambda^2+\lambda^3+\lambda^4+\ldots) + \ldots\right) \\
&= (1-\lambda)\left(\delta_t^V\left(\frac{1}{1-\lambda}\right) + \gamma\delta_{t+1}^V\left(\frac{\lambda}{1-\lambda}\right) + \gamma^2\delta_{t+2}^V\left(\frac{\lambda^2}{1-\lambda}\right) + \ldots\right) \\
&= \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^V
\end{aligned}$$

- Simplify notation $\hat{A}_t^{GAE}$, we have

$$\hat{A}_t^{GAE} = \delta_t^V + \gamma\lambda \hat{A}_{t+1}^{GAE}.$$

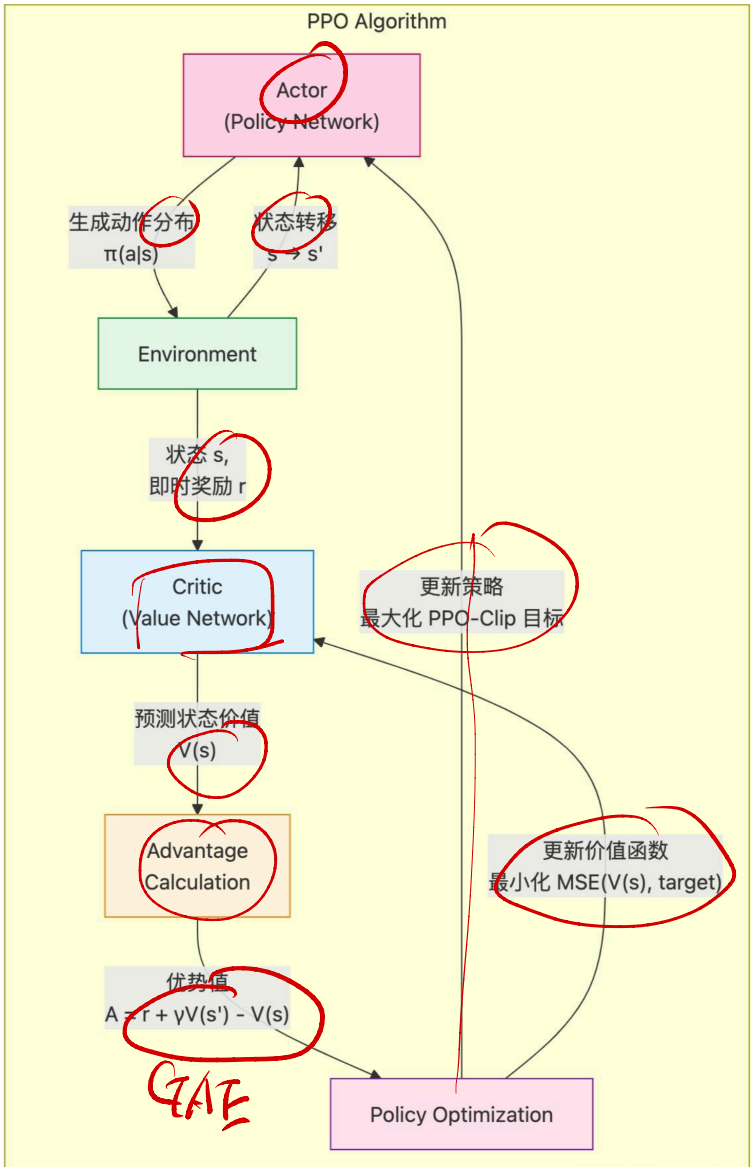
Actor-Critic PPO

- Applying PPO on the network architecture with shared parameters for both policy (actor) and value (critic) functions
- In addition to the clipped reward, the surrogate function is augmented with an error term on the value estimation (formula in red) and an entropy term (formula in blue) to encourage sufficient exploration.

$$
\begin{aligned}
& L^{\text{CLIP-AC}}(\theta|\theta_{\text{old}}) \\
&= L^{\text{CLIP}}(\theta|\theta_{\text{old}}) + \mathbb{E}[-c_1(V_\theta(S) - V_{\text{target}})^2 + c_2 H(S, \pi_\theta(.))]
\end{aligned}
$$

- Both c_1 and c_2 are two hyperparameter constants

Actor-Critic PPO



Failure Modes in PPO

- PPO has been tested on a set of benchmark tasks and proved to produce awesome results with much greater simplicity.
- Two common design choices in PPO are revisited in 2020:
  - clipped probability ratio for policy regularization
  - parameterize policy action space by continuous Gaussian or discrete softmax distribution

Failure Modes in PPO

- Three failure modes:
  1. On continuous action spaces, standard PPO is unstable when rewards vanish outside bounded support.
  2. On discrete action spaces with sparse high rewards, standard PPO often gets stuck at suboptimal actions.
  3. The policy is sensitive to initialization when there are locally optimal actions close to initialization.
- Discretizing the action space or use Beta distribution helps avoid failure mode 1 & 3 associated with Gaussian policy.
- Using KL regularization as an alternative surrogate model helps resolve failure mode 1 & 2

Rainbow PPO: Tricks to Improve PPO

- Advantage Normalization: compute the mean and std of advantages in a batch, then normalize them with (advantage-mean)/std
- State Normalization: incremental estimation of mean and std of all visited state-values, then normalize current state value with (state-value-mean)/std
- Reward Scaling: compute the standard deviation of a rolling discounted sum of the rewards, then scale current reward with (reward)/std
- Policy Entropy: loss function of actor with entropy regulation
- Learning Rate Decay
- Gradient Clip
- Orthogonal Initialization
- Adam Optimizer Epsilon Parameter
- Tanh Activation Function: use Tanh to replace ReLu in PPO

Outline

Main References

- Reinforcement Learning: An Introduction (second edition), R. Sutton & A. Barto, 2018.
- RL course slides from Richard Sutton, University of Alberta.
- RL course slides from David Silver, University College London.
- RL course slides from Sergey Levine, UC Berkeley
- RL course slides from Shusen Wang