Deep Reinforcement Learning

Lecture 8 — Advanced Policy Gradient

國立清華大學
NATIONAL TSING HUA UNIVERSITY

ational Tsing Hua University
epartment of Computer Science

Prof. Chun-Yi Lee

Outline

- **Advanced Policy Gradient Techniques**
  - **Trust Region Policy Optimization (TRPO)**
  - **Proximal Policy Optimization (PPO)**

Trust Region Policy Optimization (TRPO)

The formulation of policy gradient

- The basic principle uses gradient ascent to follow policies with the steepest increase in rewards
- However, the first-order optimizer is not very accurate for curved areas

$$\max_{\theta} J(\pi_\theta) = \mathop{\mathrm{E}}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right]$$

policy gradient (steepest direction to maximize rewards)

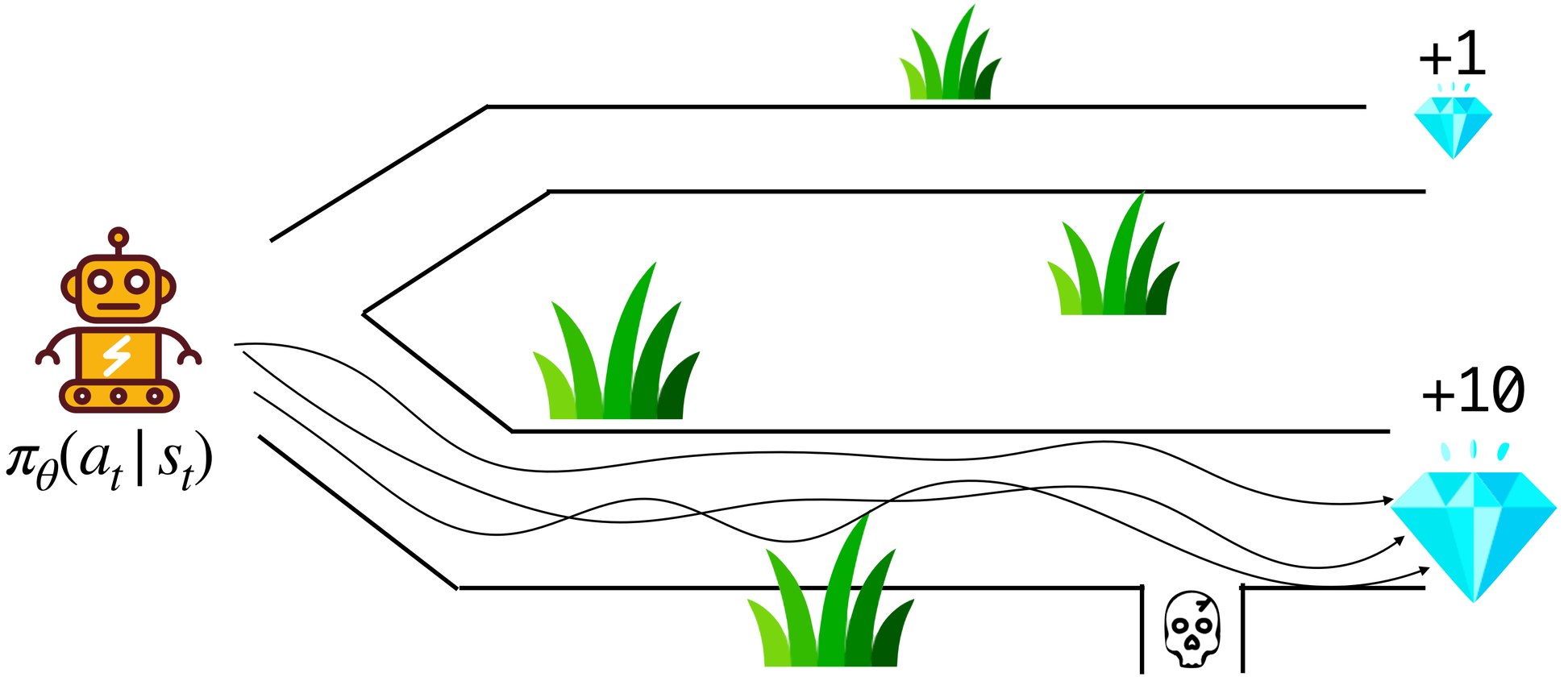
$$g = \nabla_\theta J(\pi_\theta) = \mathop{\mathrm{E}}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) A^{\pi_\theta}(s_t, a_t) \right]$$

$$\theta_{k+1} = \theta_k + \alpha g$$

take a gradient step in updating the policy

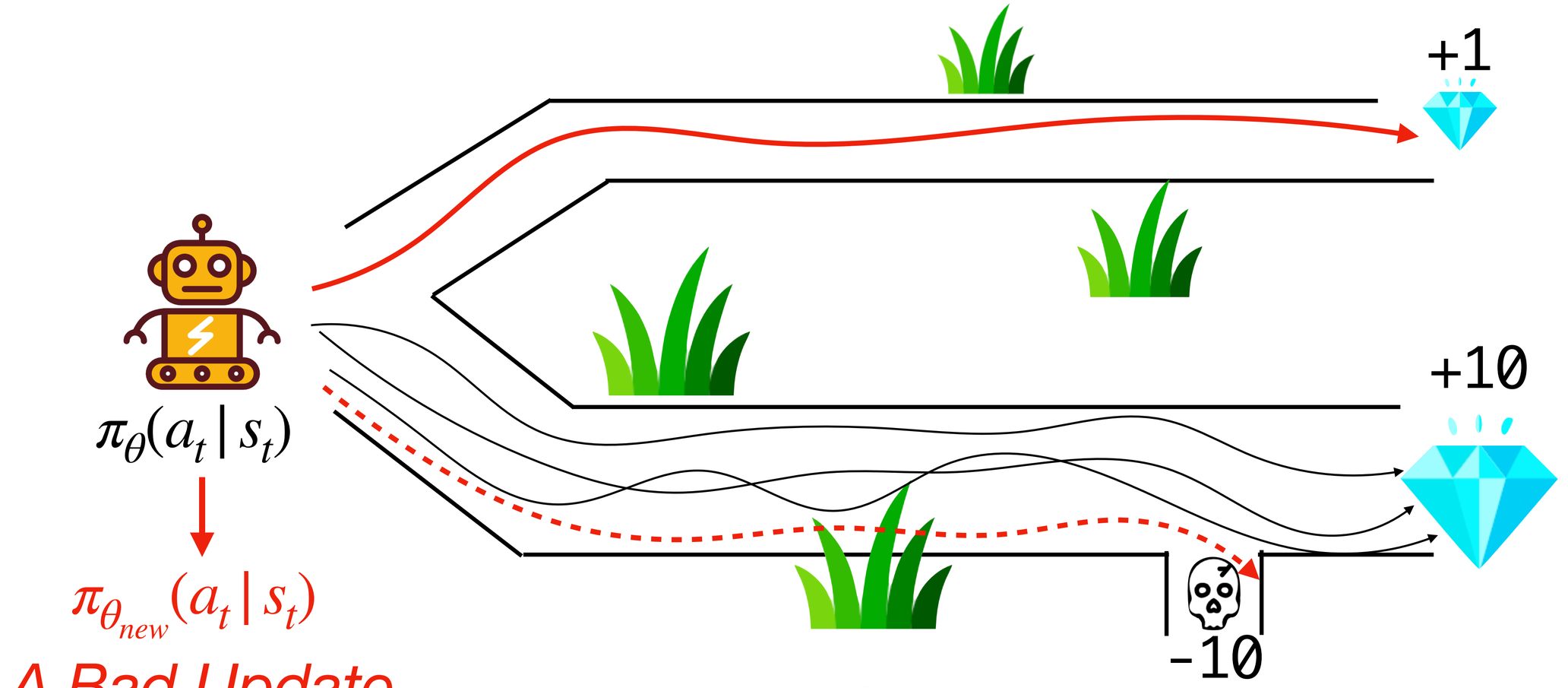
Trust Region Policy Optimization (TRPO)

An example of the drawback of policy gradient methods



Trust Region Policy Optimization (TRPO)

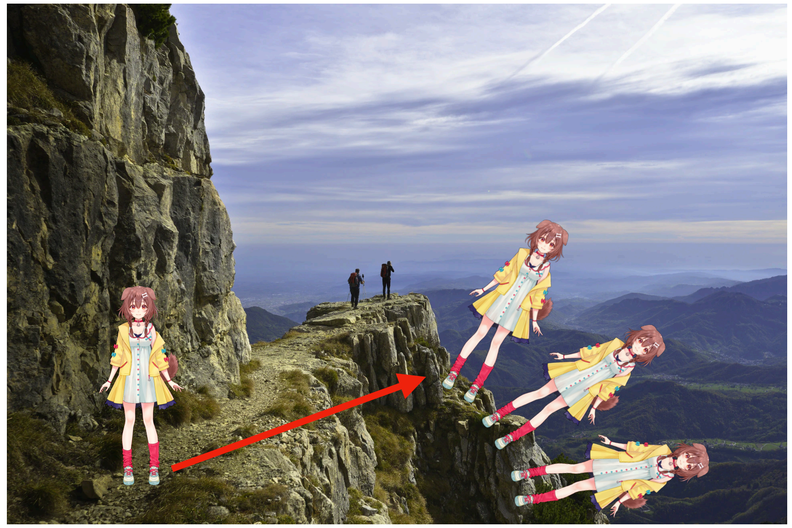
An example of the drawback of policy gradient methods



Trust Region Policy Optimization (TRPO)

Issues of policy gradient (1)

- There exists several issues with policy gradient methods discussed before
  - It is difficult to derive a proper step size for each policy gradient update
  - Too large a step leads to a disaster
  - To small a step may cause the learning process become too slow



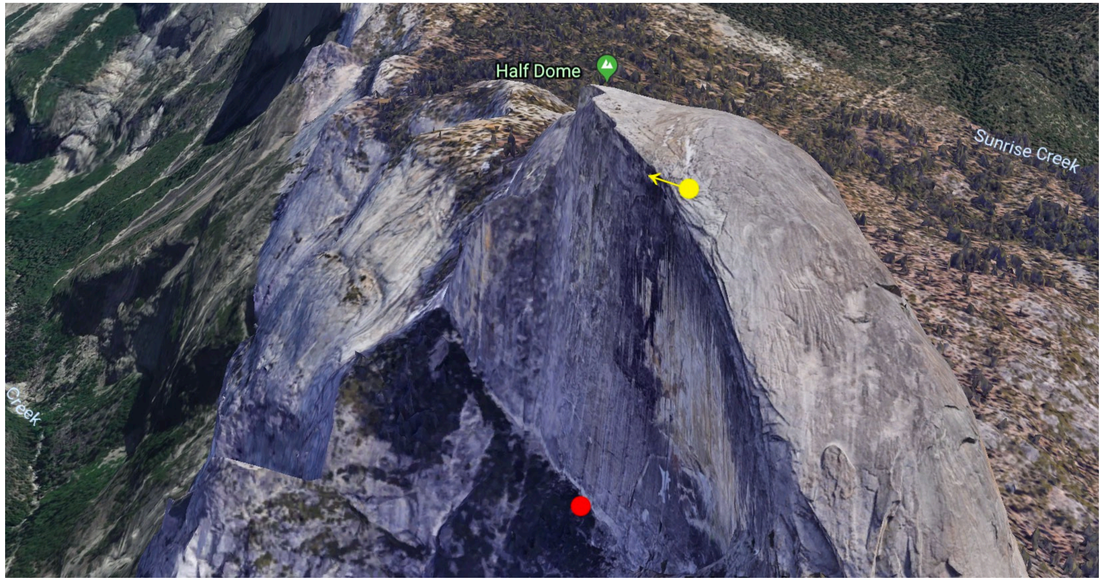
Too large of a step leads to a disaster.

Hard to get an appropriate step size

Trust Region Policy Optimization (TRPO)

Issues of policy gradient (2)

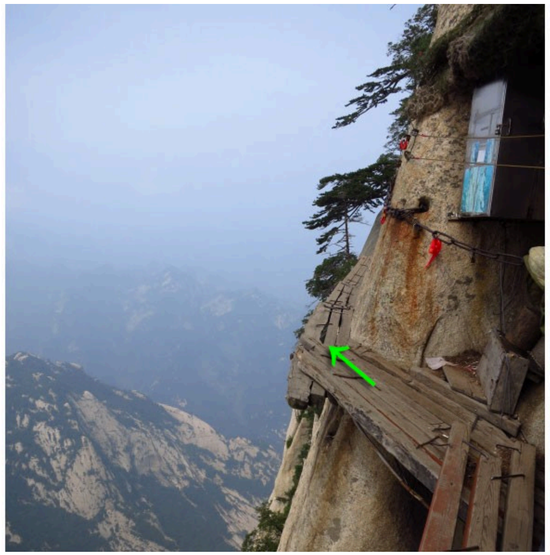
- Difficult to derive a proper learning rate
  - The learning rate at the yellow point should be larger so as to learn faster
  - The learning rate at the red spot should be small to avoid exploding policy update
  - Since the learning rate is not adaptive, policy gradient usually suffers from convergence problem severely

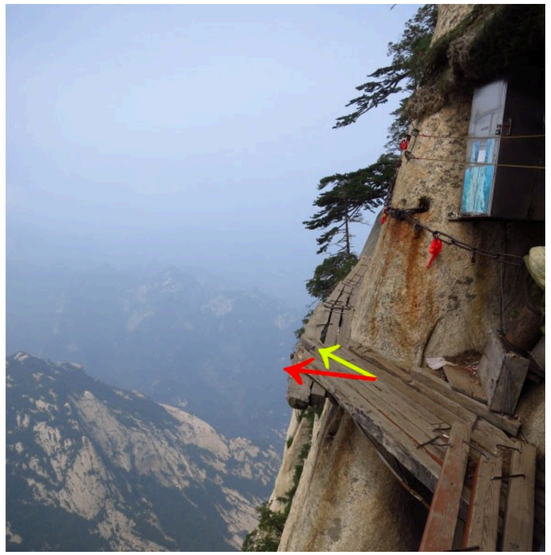


Trust Region Policy Optimization (TRPO)

Issues of policy gradient (3)

- Lack of a link between the policy space and the model parameter space
  - Vanilla policy gradient method does not constraint the policy change
  - Since the changes in policy is directly resulted from the changes in the model parameters, there should be a way to restrict the changes of the latter to avoid aggressive moves





Trust Region Policy Optimization (TRPO)

Issues of policy gradient (4)

- Sample efficiency issue
  - Policy gradient samples the whole trajectory for just one policy update
  - It cannot update the policy on every time step
    - Since the states within a trajectory are similar, updating at every time step effectively increases the probability of the policy at a small region

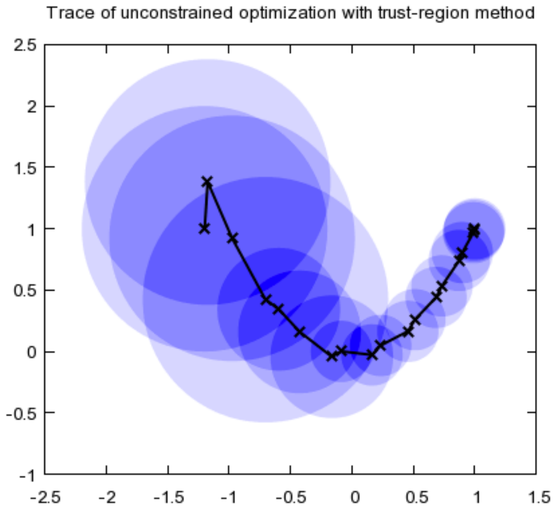
$$g = \nabla_\theta J(\pi_\theta) = \mathop{\mathrm{E}}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) A^{\pi_\theta}(s_t, a_t) \right]$$

one policy update per trajectory

Trust Region Policy Optimization (TRPO)

Trust Region Policy Optimization

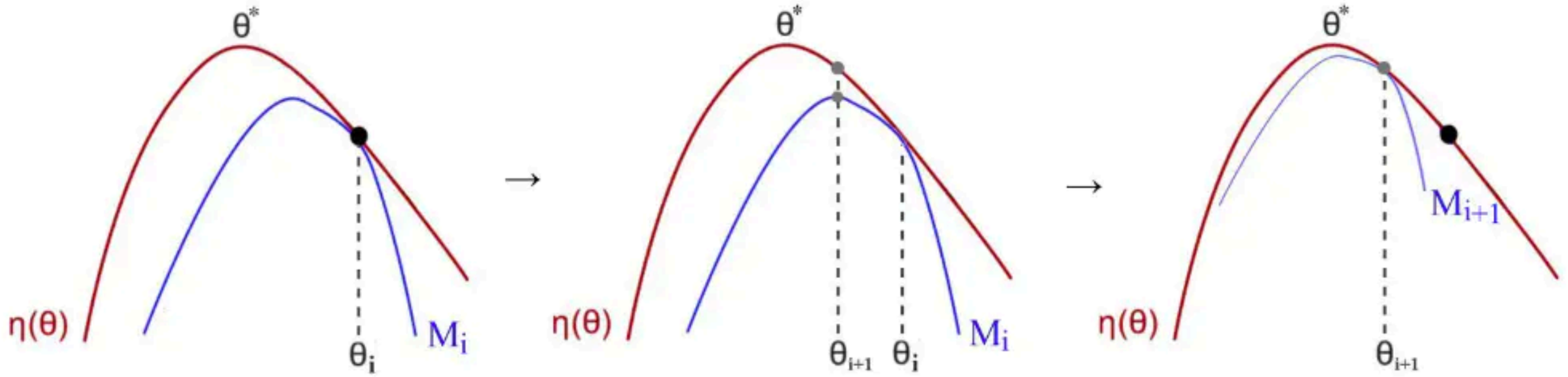
- What is TRPO?
- TRPO guarantees monotonic improvement for policy updates



Trust Region Policy Optimization (TRPO)

Minorize-Maximization (MM) algorithm

- The MM algorithm achieves this iteratively by maximizing a lower bound function (the blue line below) approximating the expected reward locally
  - $\eta(\theta)$ is the expected return, and M is the lower bound estimate of η at θ
  - θ is iteratively updated based on the optimal point of M
  - The process repeats and eventually θ will approach θ^*, the optimal point



Trust Region Policy Optimization (TRPO)

The objective of introducing the MM algorithm

- The MM algorithm can help us to derive a **trust region**
  - The trust region means that within a specific range of the model parameter space, the policy can be safely updated
  - The trust region is denoted as δ
  - Within the trust region, the policy can be updated by maximizing the lower bound estimate M of $\eta(\theta)$ through updating θ

$$\max_{s \in \mathbb{R}^n} \; \mathrm{m}_k(s)$$
$$\text{s.t.} \quad \|s\| \leq \delta$$

Trust Region Policy Optimization (TRPO)

Formulation of the expected discounted return

- Assume that the policy π is modeled as a deep neural network parameterized by θ
  - We replace $\eta(\theta)$ as $\eta(\pi_\theta)$ and drops θ for simplicity
- The expected discounted return of policy π is represented as follows:

$$\eta(\pi) = \mathbb{E}_{s_0, a_0, \ldots \sim \pi} \sum_{t=0} \gamma^t R_t$$

- If we can derive another policy $\hat{\pi}$ such that η improves as follows:

$$\eta(\hat{\pi}) = \eta(\pi) + \mathbb{E}_{s_0, a_0, \ldots \sim \hat{\pi}} \sum_{t=0} \gamma^t A_\pi(s_t, a_t)$$

the policy may be gradually improved by iteratively updating θ

Derivation of the left equation

$$\begin{aligned}
&\mathop{\mathrm{E}}_{\tau \sim \pi'} \left[\sum_{t=0}^{\infty} \gamma^t A^\pi(s_t, a_t) \right] \\
&= \mathop{\mathrm{E}}_{\tau \sim \pi'} \left[\sum_{t=0}^{\infty} \gamma^t \left(R(s_t, a_t, s_{t+1}) + \gamma V^\pi(s_{t+1}) - V^\pi(s_t) \right) \right] \\
&= \eta(\pi') + \mathop{\mathrm{E}}_{\tau \sim \pi'} \left[\sum_{t=0}^{\infty} \gamma^{t+1} V^\pi(s_{t+1}) - \sum_{t=0}^{\infty} \gamma^t V^\pi(s_t) \right] \\
&= \eta(\pi') + \mathop{\mathrm{E}}_{\tau \sim \pi'} \left[\sum_{t=1}^{\infty} \gamma^t V^\pi(s_t) - \sum_{t=0}^{\infty} \gamma^t V^\pi(s_t) \right] \\
&= \eta(\pi') - \mathop{\mathrm{E}}_{\tau \sim \pi'} [V^\pi(s_0)] \\
&= \eta(\pi') - \eta(\pi)
\end{aligned}$$

Trust Region Policy Optimization (TRPO)

Formulation of the expected discounted return

- Expansion of the expression of $\eta(\hat{\pi})$ is represented as follows:

$$\eta(\hat{\pi}) = \eta(\pi) + \sum_{t=0}^{\infty} \sum_{s} P(s_t = s \mid \hat{\pi}) \sum_{a} \hat{\pi}(a \mid s) \gamma^t A_{\pi}(s, a)$$

$$= \eta(\pi) + \sum_{t=0}^{\infty} \sum_{s} \gamma^t P(s_t = s \mid \hat{\pi}) \sum_{a} \hat{\pi}(a \mid s) A_{\pi}(s, a)$$

$$= \eta(\pi) + \sum_{s} \rho_{\hat{\pi}}(s) \sum_{a} \hat{\pi}(a \mid s) A_{\pi}(s, a)$$

Trust Region Policy Optimization (TRPO)

Formulation of the expected discounted return

- In the equation, $\rho_{\hat{\pi}}(s)$ denotes the discounted state visitation frequencies:

$$\rho_{\hat{\pi}}(s) = P(s_0 = s) + \gamma P(s_1 = s) + \dots$$

- If a nonnegative expected advantage at every 's'

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

- This equation implies that any policy update $\pi \rightarrow \hat{\pi}$ that has a non-negative expected advantage at every state s, i.e., $\sum_a \hat{\pi}(a \mid s) A_\pi(s, a) \geq 0$, is guaranteed to increase the policy performance η,

  or leave it constance in the case that the expected advantage is zero everywhere

Trust Region Policy Optimization (TRPO)

Approximation of the visitation frequency

- However, the state visitation frequency from the new policy $\hat{\pi}$ is hard to estimate, a local approximation of $\rho_{\hat{\pi}}(s)$ is thus necessary

$$\eta(\hat{\pi}) = \eta(\pi) + \sum_{s} \rho_{\hat{\pi}}(s) \sum_{a} \hat{\pi}(a \mid s) A_{\pi}(s, a)$$

$\hat{\pi}$: the new policy

π : the old policy

Trust Region Policy Optimization (TRPO)

Approximation of the visitation frequency

- In practice, it is possible to use the $\rho_\pi(s)$ to approximate the new state visitation frequency $\rho_{\hat{\pi}}(s)$ (given that $\hat{\pi} \simeq \pi$), expressed as follows:

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

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

Trust Region Policy Optimization (TRPO)

Approximation of the policy gradient

- Use θ to parameterized $\pi \rightarrow \pi_\theta$

  To make $L_{\pi_\theta}(\pi_\theta)$ match the first order $\eta(\pi_\theta)$,

$$L_{\pi_\theta}(\pi_\theta) = \eta(\pi_\theta) + \sum_s \rho_{\pi_\theta}(s) \sum_a \pi_\theta(a \mid s) A_{\pi_\theta}(s, a)$$

  = 0

  then, $L_{\pi_\theta}(\pi_\theta) = \eta(\pi_\theta)$

  This allows us to derive $\nabla_\theta L_{\pi_\theta}(\pi_\theta) = \nabla_\theta \eta(\pi_\theta)$

Trust Region Policy Optimization (TRPO)

Insights from the formulation

- $L_{\pi_\theta}(\pi_\theta)$, and thus $\nabla_\theta L_{\pi_\theta}(\pi_\theta)$, provide a direction (i.e. policy gradient) for finding a good policy (if given that $\hat{\pi}_\theta \simeq \pi_\theta$)
- If a good policy $\hat{\pi}_\theta$ can be found to improve $L_{\pi_\theta}(\hat{\pi}_\theta)$, then it can also improve $\eta(\hat{\pi}_\theta)$
- However, the formulation still doesn't provide any guidance on how big of a step size should be taken

Trust Region Policy Optimization (TRPO)

The KL divergence constraint

- The original paper of TRPO derives a lower bound, expressed as:

$$\eta(\hat{\pi}) \geq L_{\pi}(\hat{\pi}) - CD_{KL}^{\max}(\pi, \hat{\pi}), \;\; where, C = \frac{4\epsilon\gamma}{(1-\gamma)^2}$$

- TRPO uses the distance measure method between π (the old policy) and $\hat{\pi}$ (the new policy) for constraining the two policy distributions:

$$D_{KL}^{\max}(\pi, \hat{\pi}) = \max_{s} D_{KL}(\pi(\,.\,|\,s)\,||\,\hat{\pi}(\,.\,|\,s))$$

Trust Region Policy Optimization (TRPO)

The lower bound function of TRPO

- To show an improving $\eta(\pi_0) \leq \eta(\pi_1) \leq \eta(\pi_2) \leq \ldots,$ TRPO defines the lower bound (surrogate function) function M, defined as:

$$M_i(\pi) = L_{\pi_i}(\pi) - CD_{KL}^{\max}(\pi_i, \pi),$$

- Therefore, $\eta(\pi_{i+1}) \geq M_i(\pi_{i+1})$
- The equation requires that the new policy should not be too different from the old one
- If π_i is applied as M_i's input, the following equality holds:

$$M_i(\pi_i) = \underbrace{L_{\pi_i}(\pi_i)}_{\text{By page. 11 def.}} - \underbrace{CD_{KL}^{\max}(\pi_i, \pi_i)}_{0} = \eta(\pi_i)$$

$$= \eta(\pi_i)$$

Trust Region Policy Optimization (TRPO)

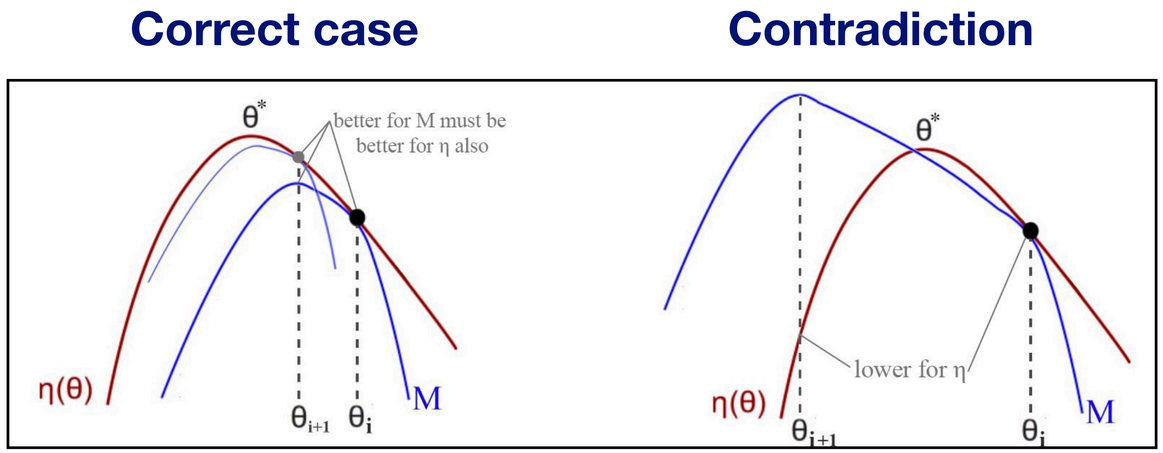
The lower bound function of TRPO

- According to the above derivation:

$$\eta(\pi_{i+1}) \geq M_i(\pi_{i+1})$$
$$\eta(\pi_i) = M_i(\pi_i)$$

- It can then be inferred that:

$$\eta(\pi_{i+1}) - \eta(\pi_i) \geq M_i(\pi_{i+1}) - M_i(\pi_i)$$



If M is the lower bound, it will never cross the red line η

Trust Region Policy Optimization (TRPO)

The lower bound function of TRPO

- If a π_{i+1} can be found that improves M_i, then

$$M_i(\pi_{i+1}) - M_i(\pi_i) \geq 0, \text{ and}$$
$$\eta(\pi_{i+1}) - \eta(\pi_i) \geq M_i(\pi_{i+1}) - M_i(\pi_i) \geq 0,$$

- η will monotonically improve.
- For notation simplicity, let's denote $\theta := \pi_\theta, L_{\theta_{old}}(\theta) := L_{\pi_{\theta_{old}}}(\pi_\theta)$
- Then, $\max_\theta [L_{\theta_{old}}(\theta) - CD_{KL}^{\max}(\theta_{old}, \theta)]$ is our main optimization objective

Trust Region Policy Optimization (TRPO)

The pseudo-code for the optimization objective in the previous page

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

Trust Region Policy Optimization (TRPO)

Optimization of a parameterized policy

- The above derivation is under the assumption that the policy can be evaluated at all states
  - Impossible for both the advantage function and the KL-divergence
- As a result, it is necessary to convert the method to a practical algorithm, under finite sample counts and arbitrary parameterization setups

$$\max_{\theta}[L_{\theta_{old}}(\theta) - CD_{KL}^{\max}(\theta_{old}, \theta)]$$

Trust Region Policy Optimization (TRPO)

Optimization of a parameterized policy

- Optimizing the equation $\max_{\theta}[L_{\theta_{old}}(\theta) - CD_{KL}^{\max}(\theta_{old}, \theta)]$, is guaranteed to improve true objective η
- However, if we use constant C defined before, the step size would be too small
- Therefore, we convert the penalty to a constraint format as follows:

$$\max_{\theta}[L_{\theta_{old}}(\theta) - CD_{KL}^{\max}(\theta_{old}, \theta)] \rightarrow$$

$$\max_{\theta} L_{\theta_{old}}(\theta) \text{ , subject to } D_{KL}^{\max}(\theta_{old}, \theta) \leq \delta$$

Trust Region Policy Optimization (TRPO)

Optimization of a parameterized policy

- $\max_{\theta} L_{\theta_{old}}(\theta)$ subject to $D_{KL}^{\max}(\theta_{old}, \theta) \leq \delta$, calculate KL-divergence for **all states** and compare them are impractical
  - In addition, it is not possible to obtain max $D_{KL}^{\max}(\theta_{old}, \theta)$
- TRPO uses a heuristic approximation averaged KL-divergence as a replacement

$$\bar{D}_{KL}^{\rho}(\theta_1, \theta_2) := \mathbb{E}_{s\sim\rho}[D_{KL}(\pi_{\theta_1}(\,.\,|s)\,||\,\pi_{\theta_2}(\,.\,|s))]$$

$$D_{KL}^{\max}(\theta_1, \theta_2) = \max_{s} D_{KL}(\pi_{\theta_1}(\,.\,|s)\,||\,\pi_{\theta_2}(\,.\,|s))$$

Trust Region Policy Optimization (TRPO)

Approximate the objective and the constraint function

$$\max_{\theta} L_{\theta_{old}}(\theta) \text{ subject to } D_{KL}^{\rho_{\theta_{old}}}(\theta_{old}, \theta) \le \delta$$

Expand

$$\max_{\theta} \sum_{s} \rho_{\theta_{old}}(s) \sum_{a} \pi_{\theta}(a \mid s) A_{\theta_{old}}(s, a)$$

By def.

$$\mathbb{E}_{s \sim \rho_{\theta_{old}}}[\dots]$$

1.By an important sampling estimator

2.Replace Advantage by Q-value

$$\mathbb{E}_{a \sim q}[\frac{\pi_{\theta}(a \mid s)}{q(a \mid s)} Q_{\theta_{old}}(s, a)]$$

* q(a|s) is the sample distribution

Trust Region Policy Optimization (TRPO)

Approximate the objective and the constraint function

- Finally, the objective and constraint function can be rewrite as :

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

$$\text{subject to } \mathbb{E}_{s\sim\rho_{\theta_{old}}}[\bar{D}_{KL}^{\rho_{\theta_{old}}}(\pi_{\theta_{old}}(\,.\,|\,s)\,||\,\pi_\theta(\,.\,|\,s))] \leq \delta$$

Trust Region

Trust Region Policy Optimization (TRPO)

Approximate the objective and the constraint function

- Finally, the objective and constraint function can be rewrite as :

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

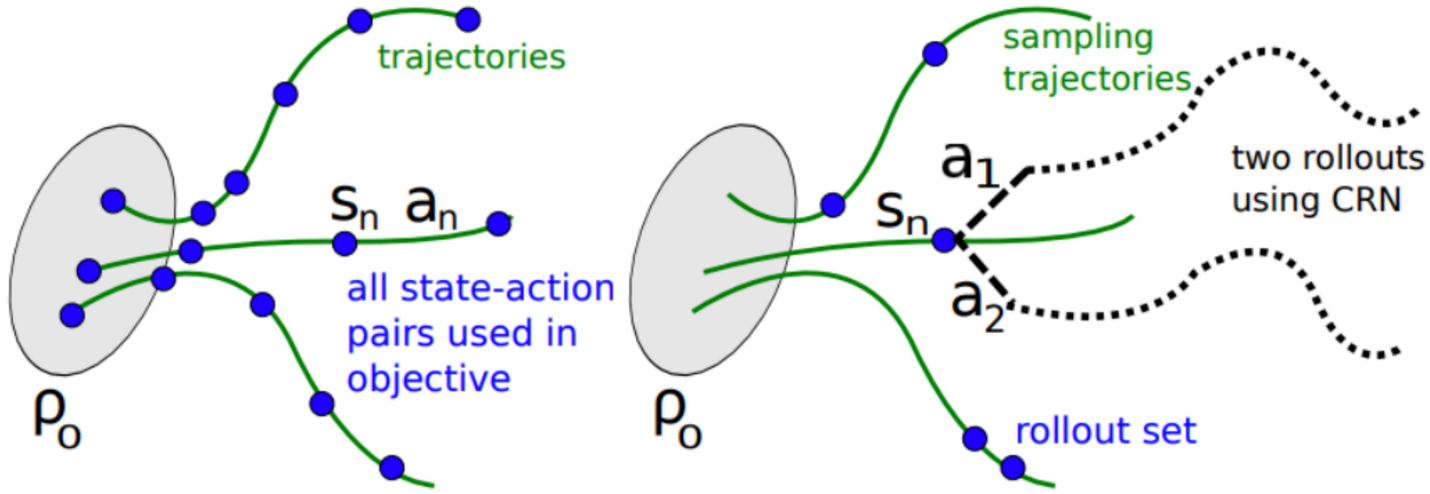
$$\text{subject to } \mathbb{E}_{s\sim\rho_{\theta_{old}}}[\bar{D}_{KL}^{\rho_{\theta_{old}}}(\pi_{\theta_{old}}(\,.\,|\,s)\,||\,\pi_\theta(\,.\,|\,s))] \leq \delta$$

- Replace the expectations by some sample scheme
- Replace the Q-value by an empirical estimate

Trust Region Policy Optimization (TRPO)

Sample scheme

- Sample data to estimate expectations in the objective function

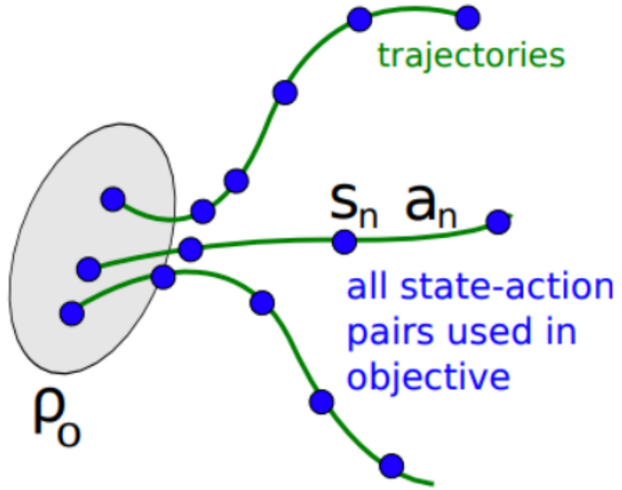


Single path Vine

Trust Region Policy Optimization (TRPO)

Sample scheme

- Single path
  - Sample a trajectory from the old policy $\pi_{\theta_{old}}$
  - Thus, $q(a \mid s) = \pi_{\theta_{old}}(a \mid s)$



Trust Region Policy Optimization (TRPO)

Sample scheme

- Vine
    - Sample a number of trajectories from the policy π_{θ_i}
    - Choose **N** states $s_1, s_2, \ldots s_N$ from these trajectories
    - $s_1, s_2, \ldots s_N$ — ‘Rollout Set’

Trust Region Policy Optimization (TRPO)

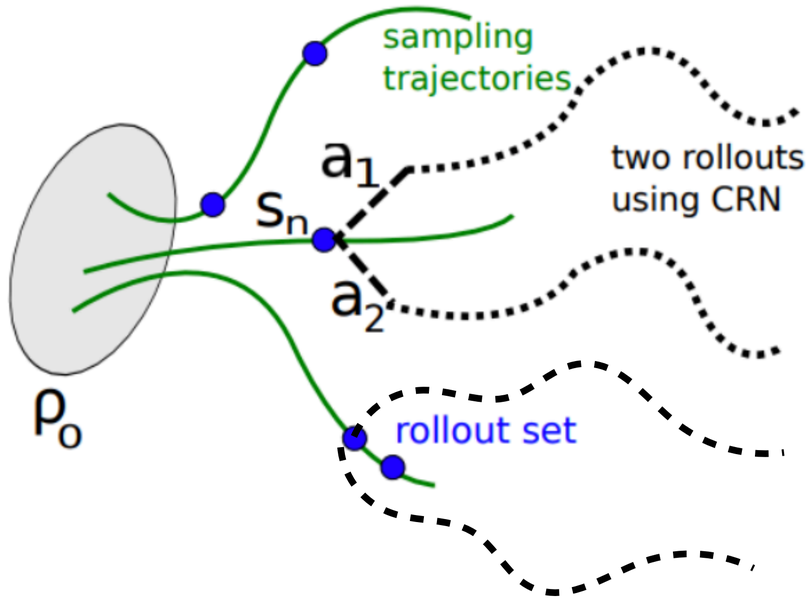
Sample scheme

- Vine
  - For each state s_n , sample K actions from $a_{n,k} \sim q(\ .\ |\ s_n)$
  - For continuous case, we use $q(\ .\ |\ s_n) = \pi_{\theta_i}(\ .\ |\ s_n)$
  - For discrete case, we use $q(\ .\ |\ s_n) =$ uniform distribution

Trust Region Policy Optimization (TRPO)

Sample scheme

- Vine
  - Estimate $\hat{Q}_{\theta_i}(s_n, a_{n,k})$ by the rollouts to reduce variance



Trust Region Policy Optimization (TRPO)

Parameter update method

- TRPO uses either **single path** or **vine** to collect data for estimating Q-values with Monte Carlo method

- Construct the estimated objective and the constraint of the following

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

$$\text{subject to } \mathbb{E}_{s\sim\rho_{\theta_{old}}}[\bar{D}_{KL}^{\rho_{\theta_{old}}}(\pi_{\theta_{old}}(\,.\,|\,s)\,||\,\pi_\theta(\,.\,|\,s))] \leq \delta$$

Trust Region Policy Optimization (TRPO)

Parameter update method

- TRPO needs a lot of approximations for both L_θ and $\bar{D}_{KL}$

  Such that $\bar{D}_{KL} \approx \frac{1}{2}(\theta - \theta_i)^T H(\theta - \theta_i)$, where H is Fisher Information matrix,

$$\mathcal{L}_{\theta_k}(\theta) \approx g^T(\theta - \theta_k) \qquad g \doteq \nabla_\theta \mathcal{L}_{\theta_k}(\theta)\,|_{\theta_k}$$

$$\bar{D}_{KL}(\theta||\theta_k) \approx \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \qquad H \doteq \nabla^2_\theta \bar{D}_{KL}(\theta||\theta_k)\,|_{\theta_k}$$

$$H = \nabla^2 f = \begin{pmatrix} \frac{\partial^2 f_1}{\partial x_1^2} & \frac{\partial^2 f_1}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f_1}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f_2}{\partial x_1^2} & \frac{\partial^2 f_2}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f_2}{\partial x_1 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f_m}{\partial x_1^2} & \frac{\partial^2 f_m}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f_m}{\partial x_1 \partial x_n} \end{pmatrix}$$

- It then uses conjugate gradient algorithm to update the policy (search a best gradient in the constraint region
  - However, conjugate gradient to update policy under calculate such constraint is still very hard.

Trust Region Policy Optimization (TRPO)

The full TRPO algorithm

- However, if we done all approximation, we can get

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

- policy gradient $\hat{g}_k$ (using advantage estimates)
- and KL-divergence Hessian-vector product function $f(v) = \hat{H}_k v$

Use CG with n_{cg} iterations to obtain $x_k \approx \hat{H}_k^{-1} \hat{g}_k$
Estimate proposed step $\Delta_k \approx \sqrt{\frac{2\delta}{x_k^T \hat{H}_k x_k}} x_k$
Perform backtracking line search with exponential decay to obtain final update

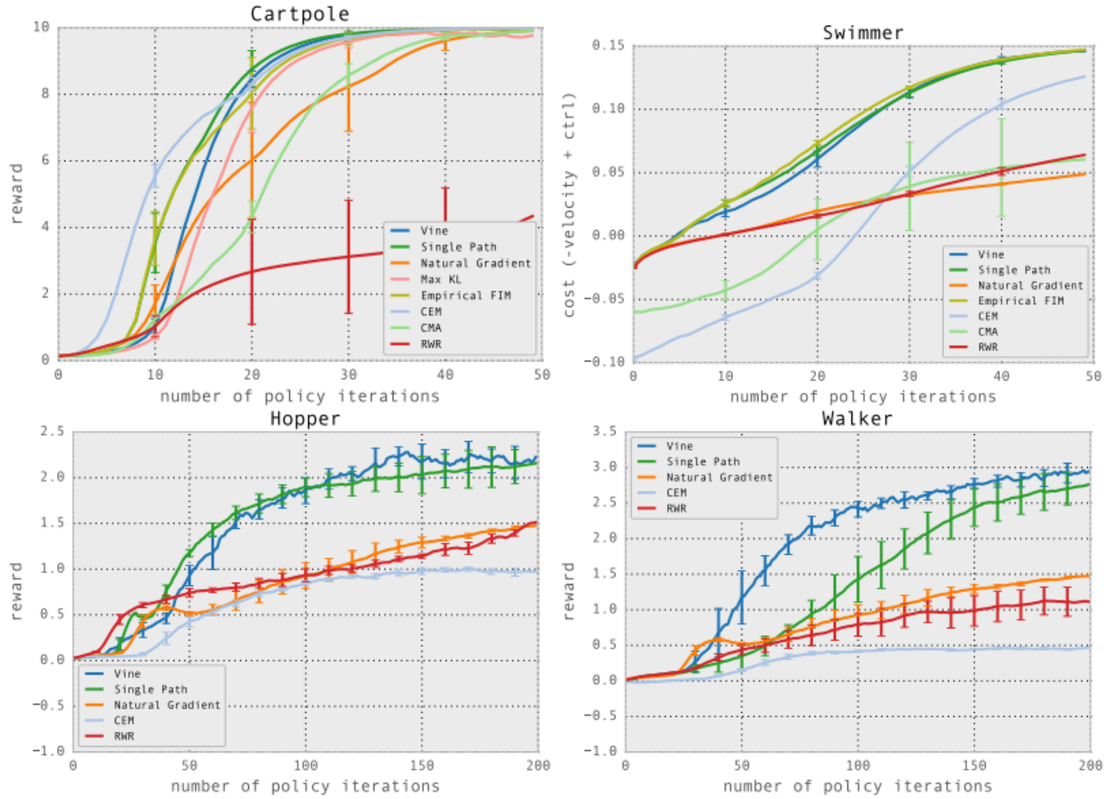
$$\theta_{k+1} = \theta_k + \alpha^j \Delta_k$$

end for

Trust Region Policy Optimization (TRPO)

Experimental results of TRPO

- Experimental results of TRPO on MuJoCo



Outline

- **Advanced Policy Gradient Techniques**
  - **Trust Region Policy Optimization (TRPO)**
  - **Proximal Policy Optimization (PPO)**

Proximal Policy Optimization (PPO)

The difficulties of TRPO

- TRPO involves the calculation of the second-order derivative and its inverse, a very expensive operation
  - $\max_{\theta} \mathbb{E}_{s\sim\rho_{\theta_{old}}, a\sim q}[\frac{\pi_\theta(a\,|\,s)}{q(a\,|\,s)} Q_{\theta_{old}}(s,a)]$,

    subject to $\mathbb{E}_{s\sim\rho_{\theta_{old}}}[D_{KL}(\pi_{\theta_{old}}(\,.\,|\,s)\,||\,\pi_\theta(\,.\,|\,s))] \le \delta$,

    this objective is the main constraint of TRPO

- It is difficult to conduct this calculation. The approximation consumes lots of time

$$\theta_{k+1} = \theta_k + \underbrace{\sqrt{\frac{2\delta}{g^T F^{-1} g}} F^{-1} g}_{\text{natural policy gradient}}$$

$$F = \nabla^2 f = \begin{pmatrix} \frac{\partial^2 f_1}{\partial x_1^2} & \frac{\partial^2 f_1}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f_1}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f_2}{\partial x_1^2} & \frac{\partial^2 f_2}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f_2}{\partial x_1 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f_m}{\partial x_1^2} & \frac{\partial^2 f_m}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f_m}{\partial x_1 \partial x_n} \end{pmatrix}$$

Proximal Policy Optimization (PPO)

The advantages of PPO

- PPO modified this constraint of TRPO
- It does not directly calculates the second-order derivatives
- Instead, PPO sets simple constraints on the objective function, providing several benefits
    - PPO is easily to implement
    - PPO is able to generate good results
    - It is faster

Proximal Policy Optimization (PPO)

PPO with Adaptive KL Penalty

- The first method proposed by PPO is called **Adaptive KL Penalty Coefficient**
  - The expression is formulated as:
    - $L^{KLPEN}(\theta) = \hat{\mathbb{E}}_t[r_t(\theta)\hat{A}_t - \beta KL[\pi_{\theta_{old}}(\,.\,|\,s_t), \pi_\theta(\,.\,|\,s_t)]]$
  - This method changes the constraint adaptively by adjusting a parameter β

Proximal Policy Optimization (PPO)

PPO with Adaptive KL Penalty

- β control the KL-divergence penalty term
- β is adjusted according to the some ratios between $d = \mathbb{E}[KL[\pi_{\theta_{old}}(. \mid s_t), \pi_\theta(. \mid s_t)]]]$ and a pre-defined hyper-parameter d_{targ}
    - If $d < d_{targ}/1.5, \beta \leftarrow \beta/2$
    - If $d > d_{targ} * 1.5, \beta \leftarrow \beta * 2,$

      where d_{targ} is an hyper-parameter
- If d is large, penalty become larger, limiting policy updates to be small

Proximal Policy Optimization (PPO)

PPO with clipped objective

- Since TRPO simply intends to constraint the ratio between the new and the old policies such that it does not change a lot, PPO propose to directly limits L
- This modification is the second and also the **main** objective proposed by PPO, expressed as:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t[\min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)],$$

$$\text{which } r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{old}}(a_t \mid s_t)}$$

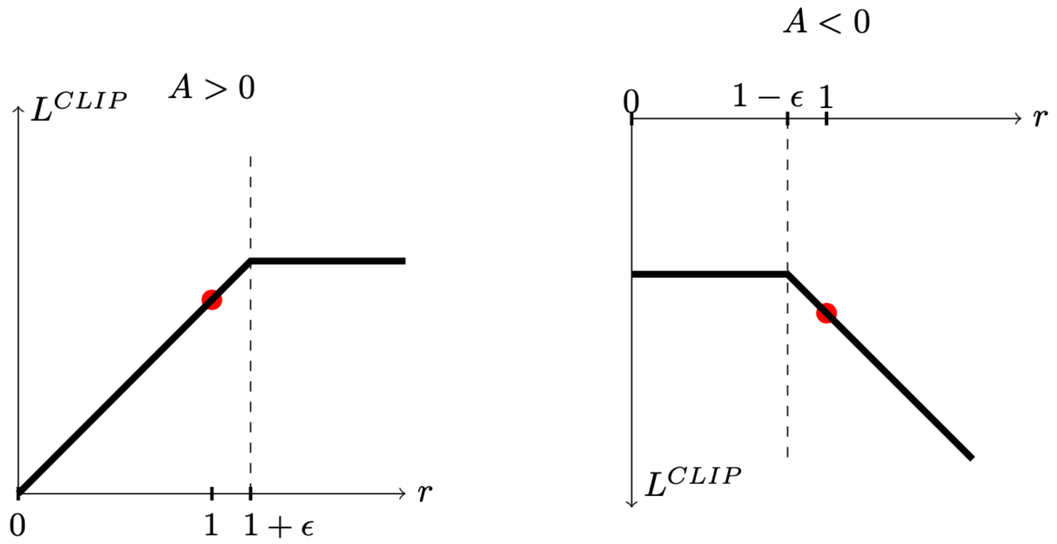
- The term $L^{CLIP}(\theta)$ is called the **clipped surrogate objective**

Proximal Policy Optimization (PPO)

PPO with clipped objective

- This modification allows $L^{CLIP}(\theta)$ to be limited
- An illustration of the objective function is plotted as follows:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t[\min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)]$$



Proximal Policy Optimization (PPO)

The overall objective function used in PPO

- The overall objective function adopted in PPO is expressed as:

$$L(\theta) = \mathbb{E}[L^{CLIP}(\theta) - c_1 L^{VF}(\theta) + c_2 S[\pi_\theta](s_t)],$$

where S is an entropy bonus (for exploration purposes), L^{VF} is a squared-error loss of the value function

Proximal Policy Optimization (PPO)

Different implementations of the objective function

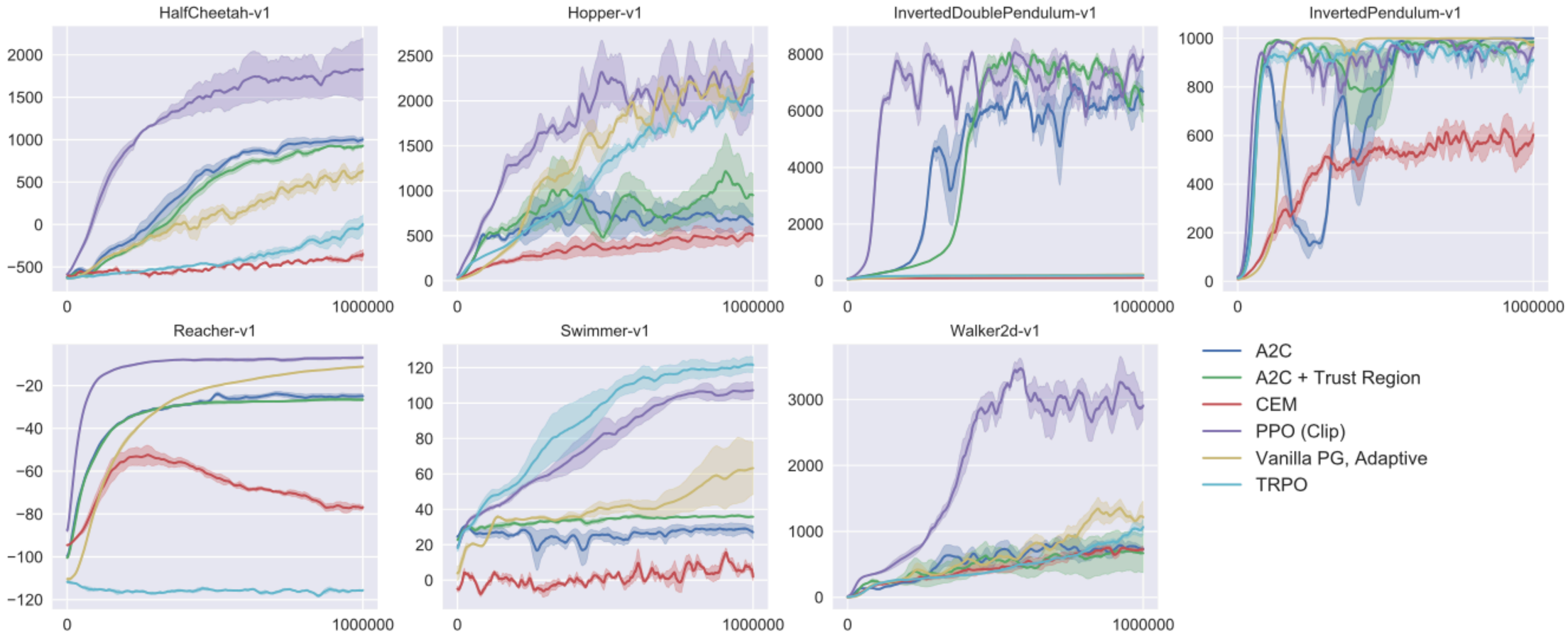
- PPO ablatively analyzed and compared different settings as follows:

$$\begin{aligned}
&\text{No clipping or penalty:} && L_t(\theta) = r_t(\theta)\hat{A}_t \\
&\text{Clipping:} && L_t(\theta) = \min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta)), 1-\epsilon, 1+\epsilon)\hat{A}_t \\
&\text{KL penalty (fixed or adaptive)} && L_t(\theta) = r_t(\theta)\hat{A}_t - \beta\,\text{KL}[\pi_{\theta_{\text{old}}}, \pi_\theta]
\end{aligned}$$

algorithm	avg. normalized score
No clipping or penalty	-0.39
Clipping, $\epsilon = 0.1$	0.76
Clipping, $\epsilon = 0.2$	**0.82**
Clipping, $\epsilon = 0.3$	0.70
Adaptive KL $d_{\text{targ}} = 0.003$	0.68
Adaptive KL $d_{\text{targ}} = 0.01$	0.74
Adaptive KL $d_{\text{targ}} = 0.03$	0.71
Fixed KL, $\beta = 0.3$	0.62
Fixed KL, $\beta = 1.$	0.71
Fixed KL, $\beta = 3.$	0.72
Fixed KL, $\beta = 10.$	0.69

Proximal Policy Optimization (PPO)

Experimental results of PPO on MuJoCo



ありがとう ございます