Trust Region Policy Optimization

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, Pieter Abbeel

@ICML 2015

Presenter: Shivam Kalra

Shivam.kalra@uwaterloo.ca

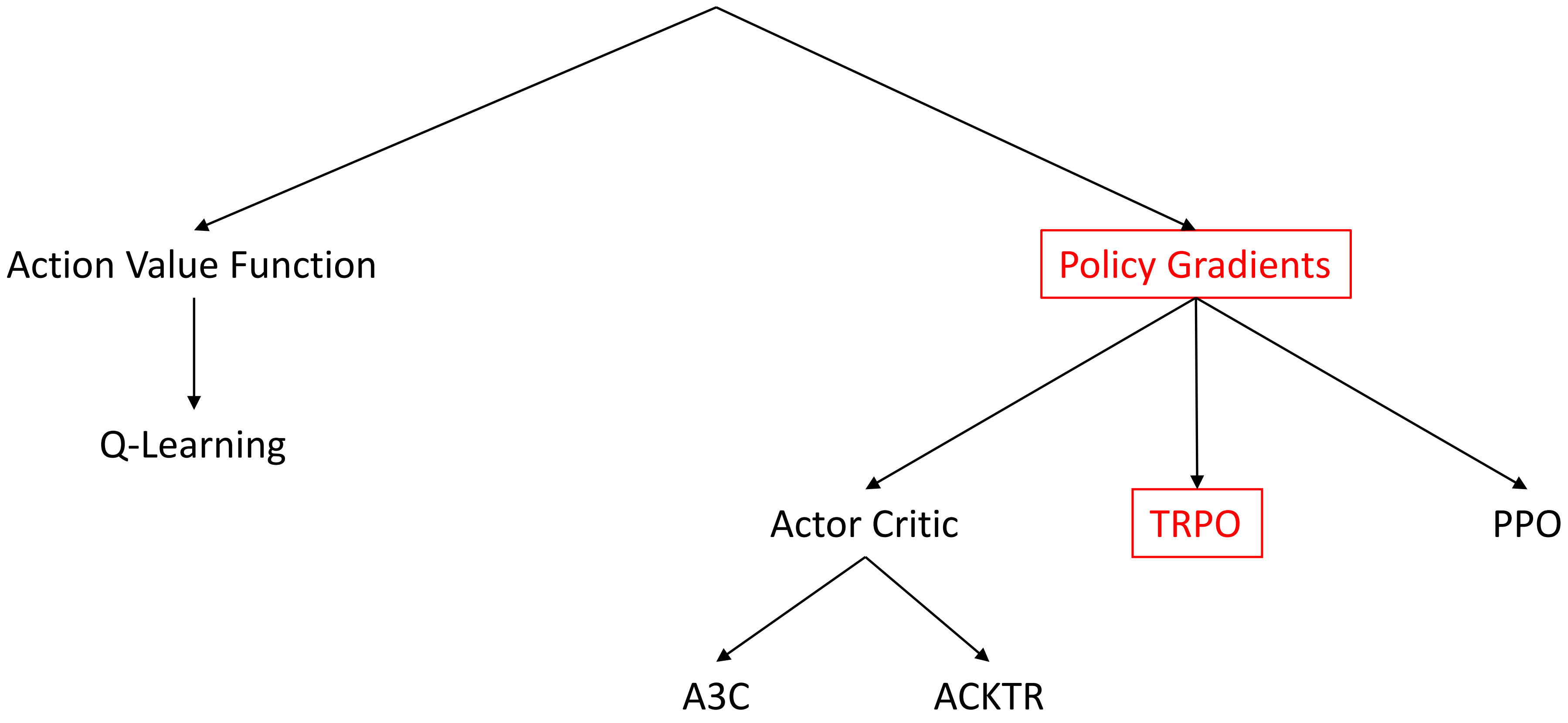
CS 885 (Reinforcement Learning)

Prof. Pascal Poupart

June 20th 2018

UNIVERSITY OF WATERLOO

Reinforcement Learning



Ref: https://www.youtube.com/watch?v=CKaN5PgkSBc

Policy Gradient

```
For i=1,2,…
    Collect N trajectories for policy π_θ
    Estimate advantage function A
    Compute policy gradient g
    Update policy parameter θ = θ_old + αg
```

Problems of Policy Gradient

```
For i=1,2,…
        Collect N trajectories for policy π_θ
        Estimate advantage function A
        Compute policy gradient g
        Update policy parameter θ = θ_old + αg
```

Non stationary input data due to changing policy and reward distributions change

Problems of Policy Gradient

For i=1,2,…

Collect N trajectories for policy π_θ

Estimate advantage function A

Compute policy gradient g

Update policy parameter $\theta = \theta_{old} + \alpha g$

Advantage is very random initially

Problems of Policy Gradient

For i=1,2,…

Collect N trajectories for policy π_θ

Estimate advantage function A

Compute policy gradient g

Update policy parameter $\theta = \theta_{old} + \alpha g$

We need more carefully crafted policy update

We want improvement and not degradation

Idea: We can update old policy π_{old} to a new policy $\tilde{\pi}$ such that they are "trusted" distance apart. Such conservative policy update allows improvement instead of degradation.

RL to Optimization

- Most of ML is optimization
  - Supervised learning is reducing training loss

- RL: what is policy gradient optimizing?
  - Favoring (s, a) that gave more advantage A.
  - Can we write down optimization problem that allows to do small update on a policy π based on data sampled from π (on-policy data)

Ref: https://www.youtube.com/watch?v=xvRrgxcpaHY (6:40)

What loss to optimize?

- Optimize $\eta(\pi)$ i.e., expected return of a policy π

$$\eta(\pi) = \mathbb{E}_{s_0 \sim \rho_0, a^t \sim \pi(.|s_t)} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right]$$

- We collect data with π_{old} and optimize the objective to get a new policy $\tilde{\pi}$.

What loss to optimize?

- We can express $\eta(\tilde{\pi})$ in terms of the advantage over the original policy[1].

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

Expected return of old policy

Expected return of new policy

Sample from new policy

[1] Kakade, Sham, and John Langford. "Approximately optimal approximate reinforcement learning." ICML. Vol. 2. 2002.

What loss to optimize?

- Previous equation can be rewritten as[1]:

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

$\eta(\tilde{\pi})$: Expected return of new policy

$\eta(\pi_{old})$: Expected return of old policy

$\rho_{\tilde{\pi}}(s)$: Discounted visitation frequency

$$\rho_\pi(s) = P(s_0 = s) + \gamma P(s_1 = s) + \gamma^2 P + \cdots$$

[1] Schulman, John, et al. "Trust region policy optimization." International Conference on Machine Learning. 2015.

What loss to optimize?

Old Expected Return

$$\eta(\tilde{\pi}) = \eta(\pi_{old}) + \sum_s \rho_{\tilde{\pi}}(s) \boxed{\sum_a \tilde{\pi}(a|s) A_{\pi}(s,a)} \geq \mathbf{0}$$

New Expected Return

What loss to optimize?

$$\eta(\tilde{\pi}) = \eta(\pi_{old}) + \sum_{s} \rho_{\tilde{\pi}}(s) \boxed{\sum_{a} \tilde{\pi}(a|s) A_{\pi}(s,a)} \geq \mathbf{0}$$

New Expected Return > **Old Expected Return**

Guaranteed Improvement from $\pi_{old} \rightarrow \tilde{\pi}$

New State Visitation is Difficult

State visitation based on new policy

$$\eta(\tilde{\pi}) = \eta(\pi_{old}) + \sum_{s} \rho_{\tilde{\pi}}(s) \sum_{a} \tilde{\pi}(a|s) A_{\pi}(s,a)$$

New policy

"Complex dependency of $\rho_{\tilde{\pi}}(s)$ on $\tilde{\pi}$ makes the equation difficult to optimize directly." [1]

[1] Schulman, John, et al. "Trust region policy optimization." International Conference on Machine Learning. 2015.

New State Visitation is Difficult

$$\eta(\tilde{\pi}) = \eta(\pi_{old}) + \sum_{s} \boxed{\rho_{\tilde{\pi}}(s)} \sum_{a} \tilde{\pi}(a|s) A_{\pi}(s,a)$$

$$L(\tilde{\pi}) = \eta(\pi_{old}) + \sum_{s} \boxed{\rho_{\pi}(s)} \sum_{a} \tilde{\pi}(a|s) A_{\pi}(s,a)$$

Local approximation of $\eta(\tilde{\pi})$

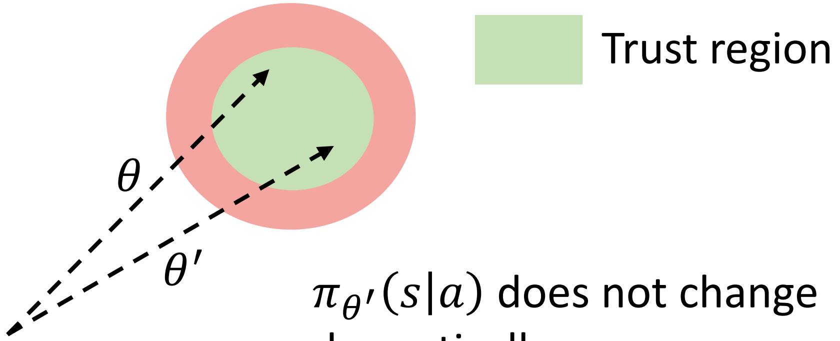
[1] Schulman, John, et al. "Trust region policy optimization." International Conference on Machine Learning. 2015.

Local approximation of $\eta(\tilde{\pi})$

$$L(\tilde{\pi}) = \eta(\pi_{old}) + \sum_{s} \rho_{\pi}(s) \sum_{a} \tilde{\pi}(a|s) A_{\pi_{old}}(s,a)$$

The approximation is accurate within step size δ (trust region)

Monotonic improvement guaranteed



$\pi_{\theta'}(s|a)$ does not change dramatically.

[1] Schulman, John, et al. "Trust region policy optimization." International Conference on Machine Learning. 2015.

Local approximation of $\eta(\tilde{\pi})$

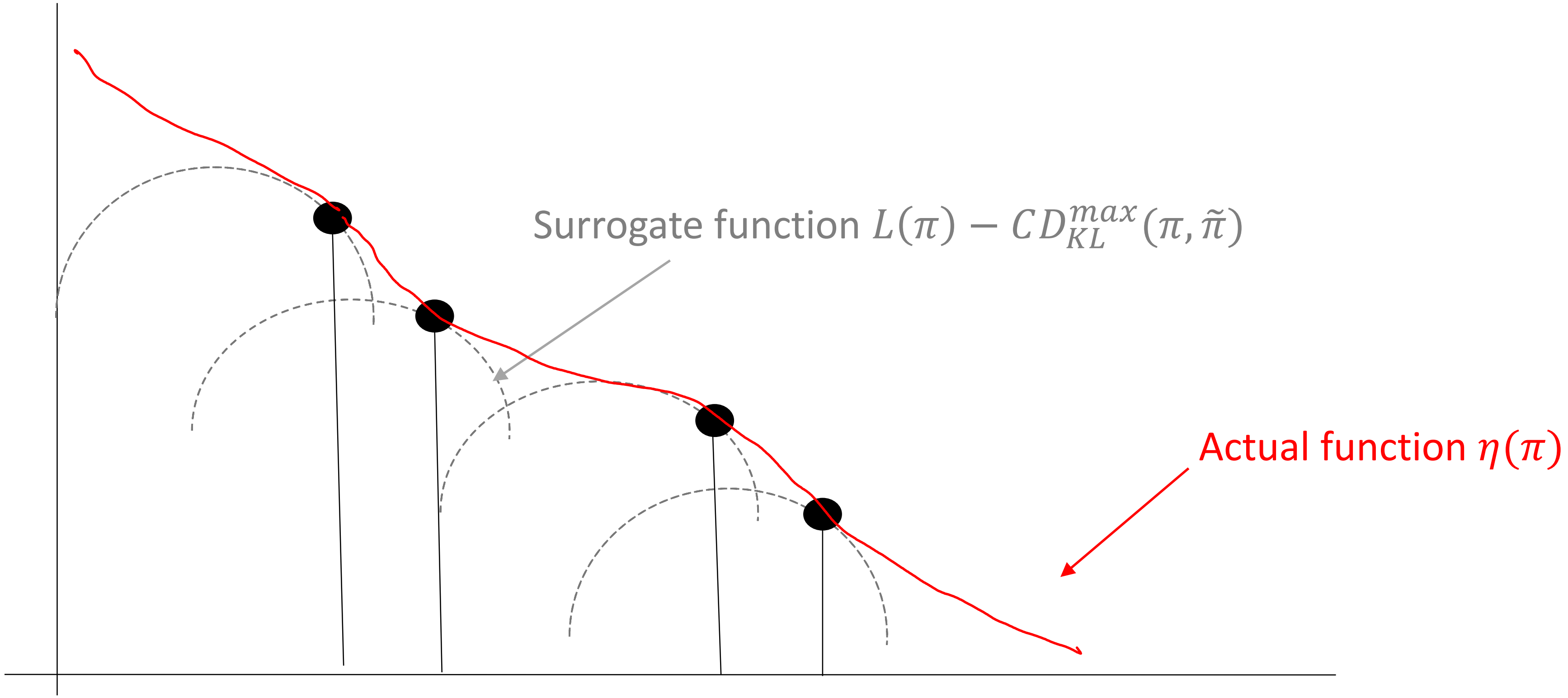
- The following bound holds:

$$\eta(\tilde{\pi}) \geq L(\tilde{\pi}) - CD_{KL}^{max}(\pi, \tilde{\pi})$$
$$\text{Where, } C = \frac{4\epsilon\gamma}{(1-\gamma)^2}$$

- Monotonically improving policies can be generated by:

$$\pi = \arg\max_{\pi}[L(\tilde{\pi}) - CD_{KL}^{max}(\pi, \tilde{\pi})]$$
$$Where, C = \frac{4\epsilon\gamma}{(1-\gamma)^2}$$

Minorization Maximization (MM) algorithm



Optimization of Parameterized Policies

- Now policies are parameterized $\pi_\theta(a|s)$ with parameters θ
- Accordingly surrogate function changes to

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

Optimization of Parameterized Policies

$$\arg\max_{\theta}[L(\theta) - \boxed{C} D_{KL}^{max}(\theta_{old}, \theta)]$$

In practice C results in very small step sizes

One way to take larger step size is to constraint KL divergence between the new policy and the old policy, i.e., a trust region constraint:

$$\underset{\theta}{\boldsymbol{maximize}}\ \boldsymbol{L_{\theta}(\theta)}$$

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

Solving KL-Penalized Problem

- $\text{max}imize_\theta \; L(\theta) - C.D_{KL}^{max}(\theta_{old}, \theta)$

- Use mean KL divergence instead of max.
  - i.e., $\underset{\theta}{\text{max}imize} \; L(\theta) - C.\overline{D_{KL}}(\theta_{old}, \theta)$

- Make linear approximation to L and quadratic to KL term:

$$\underset{\theta}{\text{max}imize} \; g \,.\, (\theta - \theta_{old}) - \frac{c}{2}(\theta - \theta_{old})^T F (\theta - \theta_{old})$$

where, $g = \frac{\partial}{\partial \theta} L(\theta)|_{\theta=\theta_{old}}$, $\quad F = \frac{\partial^2}{\partial^2 \theta} \overline{D_{KL}}(\theta_{old}, \theta)|_{\theta=\theta_{old}}$

Solving KL-Penalized Problem

- Make linear approximation to L and quadratic to KL term:

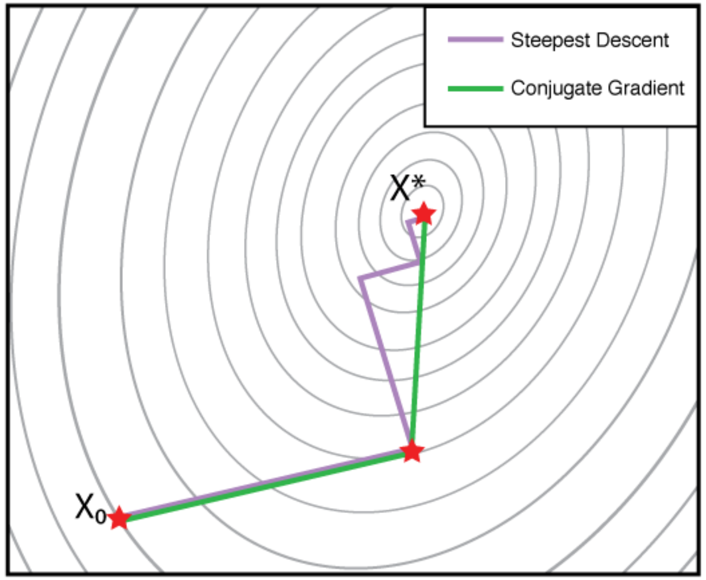
$$\underset{\theta}{\text{maximize}}\; g\,.\,(\theta-\theta_{old})-\frac{c}{2}(\theta-\theta_{old})^T F(\theta-\theta_{old})$$

$$\text{where, } g=\frac{\partial}{\partial\theta}L(\theta)\big|_{\theta=\theta_{old}},\qquad F=\frac{\partial^2}{\partial^2\theta}\overline{D_{KL}}(\theta_{old},\theta)\big|_{\theta=\theta_{old}}$$

- Solution: $\theta-\theta_{old}=\frac{1}{c}F^{-1}g$. Don't want to form full Hessian matrix $F=\frac{\partial^2}{\partial^2\theta}\overline{D_{KL}}(\theta_{old},\theta)|_{\theta=\theta_{old}}$.
- Can compute $F^{-1}g$ approximately using **conjugate gradient** algorithm without forming F explicitly.

Conjugate Gradient (CG)

- Conjugate gradient algorithm approximately solves for $x = A^{-1}b$ without explicitly forming matrix A
- After k iterations, CG has minimized $\frac{1}{2}x^T Ax - bx$



TRPO: KL-Constrained

- Unconstrained problem: $\underset{\theta}{\max imize}\, L(\theta) - C.\overline{D_{KL}}(\theta_{old}, \theta)$
- Constrained problem: $\underset{\theta}{\max imize}\, L(\theta)$ subject to $C.\overline{D_{KL}}(\theta_{old}, \theta) \leq \delta$
- δ is a hyper-parameter, remains fixed over whole learning process
- Solve constrained quadratic problem: compute $F^{-1}g$ and then rescale step to get correct KL
  - $\underset{\theta}{\max imize}\, g\ .\ (\theta - \theta_{old})$ subject to $\frac{1}{2}(\theta - \theta_{old})^T F(\theta - \theta_{old}) \leq \delta$
  - Lagrangian: $\mathcal{L}(\theta, \lambda) = g\ .\ (\theta - \theta_{old}) - \frac{\lambda}{2}[(\theta - \theta_{old})^T F(\theta - \theta_{old}) - \delta]$
  - Differentiate wrt θ and get $\theta - \theta_{old} = \frac{1}{\lambda}F^{-1}g$
  - We want $\frac{1}{2}s^T F s = \delta$
  - Given candidate step $s_{unscaled}$ rescale to $s = \sqrt{\frac{2\delta}{s_{unscaled}.(F s_{unscaled})}}\, s_{unscaled}$

TRPO Algorithm

```
For i=1,2,…
        Collect N trajectories for policy π_θ
        Estimate advantage function A
        Compute policy gradient g
        Use CG to compute H^{-1}g
        Compute rescaled step s = αH^{-1}g with rescaling and line search
        Apply update: θ = θ_old + αH^{-1}g
```

$\underset{\theta}{\text{maximize}}\ L(\theta)$ subject to $C.\overline{D_{KL}}(\theta_{old}, \theta) \leq \delta$

Questions?