Trust Region Policy Optimization (TRPO)

Insoon Yang

Department of Electrical and Computer Engineering
Seoul National University

The RL problem

- Policy optimization:

$$\max_{\pi} \; J(\pi) := \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$$

The RL problem

- Policy optimization:

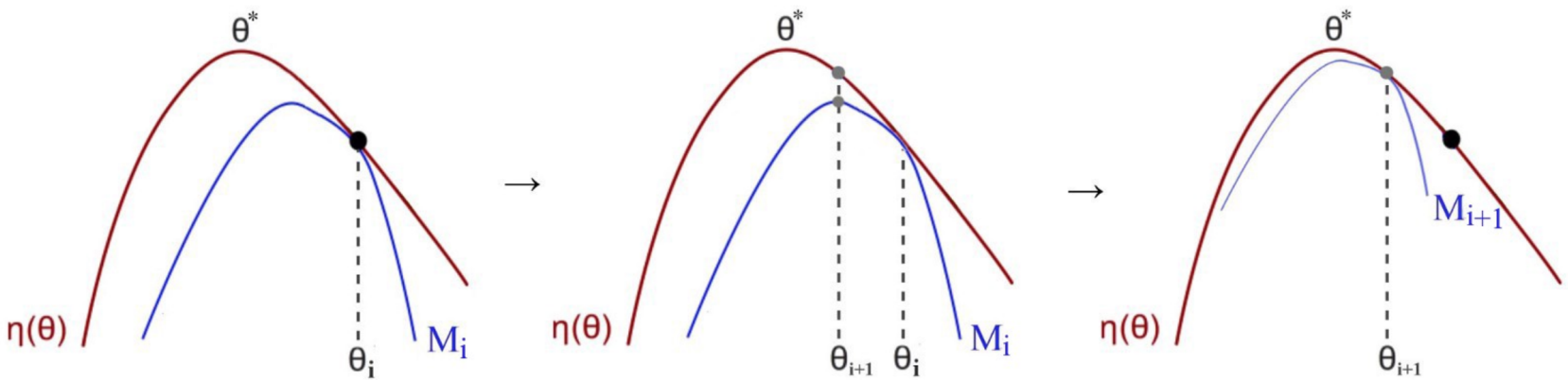
$$\max_{\pi} \; J(\pi) := \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$$

- It's the same as

$$\max_{\pi} \; J(\pi) - J(\pi_{old})$$

$\Longrightarrow$ Maximizing the reward relative to an old policy

Step I: Application of Minorize-Maximization algorithm



- Need a surrogate M_i of $\eta(\pi) := J(\pi) - J(\pi_{old})$

Effect of policy update: $\pi_{old} \rightarrow \pi$?

Effect of policy update: $\pi_{old} \rightarrow \pi$?

$$
\begin{aligned}
J(\pi) - J(\pi_{old}) &= J(\pi) - \mathbb{E}_{\tau \sim p^{\pi}}\left[v^{\pi_{old}}(s_0)\right] \\
&= J(\pi) - \mathbb{E}_{\tau \sim p^{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t v^{\pi_{old}}(s_t) - \sum_{t=1}^{\infty} \gamma^t v^{\pi_{old}}(s_t)\right] \\
&= \mathbb{E}_{\tau \sim p^{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right] + \mathbb{E}_{\tau \sim p^{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t (\gamma v^{\pi_{old}}(s_{t+1}) - v^{\pi_{old}}(s_t))\right] \\
&= \mathbb{E}_{\tau \sim p^{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + \gamma v^{\pi_{old}}(s_{t+1}) - v^{\pi_{old}}(s_t))\right] \\
&= \mathbb{E}_{\tau \sim p^{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t A^{\pi_{old}}(s_t, a_t)\right]
\end{aligned}
$$

Effect of policy update: $\pi_{old} \rightarrow \pi$

So far we have

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

- Let's use importance sampling!

$$J(\pi) - J(\pi_{old}) = \sum_t \mathbb{E}_{s_t \sim p^{\pi}} \left[\mathbb{E}_{a_t \sim \pi_{old}(\cdot|s_t)} \left[\frac{\pi(a_t|s_t)}{\pi_{old}(a_t|s_t)} \gamma^t A^{\pi_{old}}(s_t, a_t) \right] \right]$$

Local approximation: Ignoring distribution mismatch

Importance sampling:

$$J(\pi) - J(\pi_{old}) = \sum_t \mathbb{E}_{s_t \sim p^{\pi}} \left[\mathbb{E}_{a_t \sim \pi_{old}(\cdot|s_t)} \left[\frac{\pi(a_t|s_t)}{\pi_{old}(a_t|s_t)} \gamma^t A^{\pi_{old}}(s_t, a_t) \right] \right]$$

Local approximation: Ignoring distribution mismatch

Importance sampling:

$$J(\pi) - J(\pi_{old}) = \sum_t \mathbb{E}_{s_t \sim p^{\pi}} \left[\mathbb{E}_{a_t \sim \pi_{old}(\cdot|s_t)} \left[\frac{\pi(a_t|s_t)}{\pi_{old}(a_t|s_t)} \gamma^t A^{\pi_{old}}(s_t, a_t) \right] \right]$$

Local approximation:

$$L_{\pi_{old}}(\pi) := \sum_t \mathbb{E}_{s_t \sim p^{\pi_{old}}} \left[\mathbb{E}_{a_t \sim \pi_{old}(\cdot|s_t)} \left[\frac{\pi(a_t|s_t)}{\pi_{old}(a_t|s_t)} \gamma^t A^{\pi_{old}}(s_t, a_t) \right] \right]$$

Local approximation: Ignoring distribution mismatch

Importance sampling:

$$J(\pi) - J(\pi_{old}) = \sum_t \mathbb{E}_{s_t \sim p^{\pi}} \left[\mathbb{E}_{a_t \sim \pi_{old}(\cdot|s_t)} \left[\frac{\pi(a_t|s_t)}{\pi_{old}(a_t|s_t)} \gamma^t A^{\pi_{old}}(s_t, a_t) \right] \right]$$

Local approximation:

$$L_{\pi_{old}}(\pi) := \sum_t \mathbb{E}_{s_t \sim p^{\pi_{old}}} \left[\mathbb{E}_{a_t \sim \pi_{old}(\cdot|s_t)} \left[\frac{\pi(a_t|s_t)}{\pi_{old}(a_t|s_t)} \gamma^t A^{\pi_{old}}(s_t, a_t) \right] \right]$$

$\Longrightarrow$ Now all samples can be generated using π_{old}

Use the local approximation to get surrogate

- KL divergence for the two distributions p and q:
$$D_{KL}(p \parallel q) = \sum_{i=1}^{N} p_i \log \frac{p_i}{q_i}$$

Use the local approximation to get surrogate

- KL divergence for the two distributions p and q:
  $$D_{KL}(p \parallel q) = \sum_{i=1}^{N} p_i \log \frac{p_i}{q_i}$$
- Surrogate:

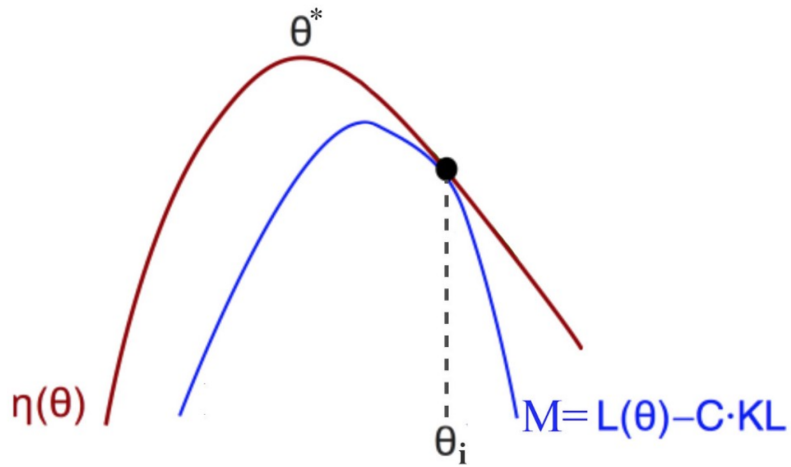
$$\underbrace{J(\pi) - J(\pi_{old})}_{=:\eta \text{ (objective to maximize)}} \geq \underbrace{L_{\pi_{old}}(\pi) + C \max_s D_{KL}(\pi_{old}(\cdot|s) \parallel \pi(\cdot|s))}_{=:M(\pi) \text{ (surrogate)}}$$

Use the local approximation to get surrogate

- KL divergence for the two distributions p and q:
$$D_{KL}(p \parallel q) = \sum_{i=1}^{N} p_i \log \frac{p_i}{q_i}$$
- Surrogate:
$$\underbrace{J(\pi) - J(\pi_{old})}_{=:\eta \text{ (objective to maximize)}} \geq \underbrace{L_{\pi_{old}}(\pi) + C \max_s D_{KL}(\pi_{old}(\cdot|s) \parallel \pi(\cdot|s))}_{=:M(\pi) \text{ (surrogate)}}$$
- Also,
$$\eta(\pi_{old}) = M(\pi_{old})$$

Use the local approximation to get surrogate

- KL divergence for the two distributions p and q:
$$D_{KL}(p \parallel q) = \sum_{i=1}^{N} p_i \log \frac{p_i}{q_i}$$
- Surrogate:
$$\underbrace{J(\pi) - J(\pi_{old})}_{=:\eta \text{ (objective to maximize)}} \geq \underbrace{L_{\pi_{old}}(\pi) + C \max_s D_{KL}(\pi_{old}(\cdot|s) \parallel \pi(\cdot|s))}_{=:M(\pi) \text{ (surrogate)}}$$
- Also,
$$\eta(\pi_{old}) = M(\pi_{old})$$



MM algorithm

Original RL problem:

$$\max_{\pi} \; J(\pi) - J(\pi_{old})$$

MM algorithm

Original RL problem:

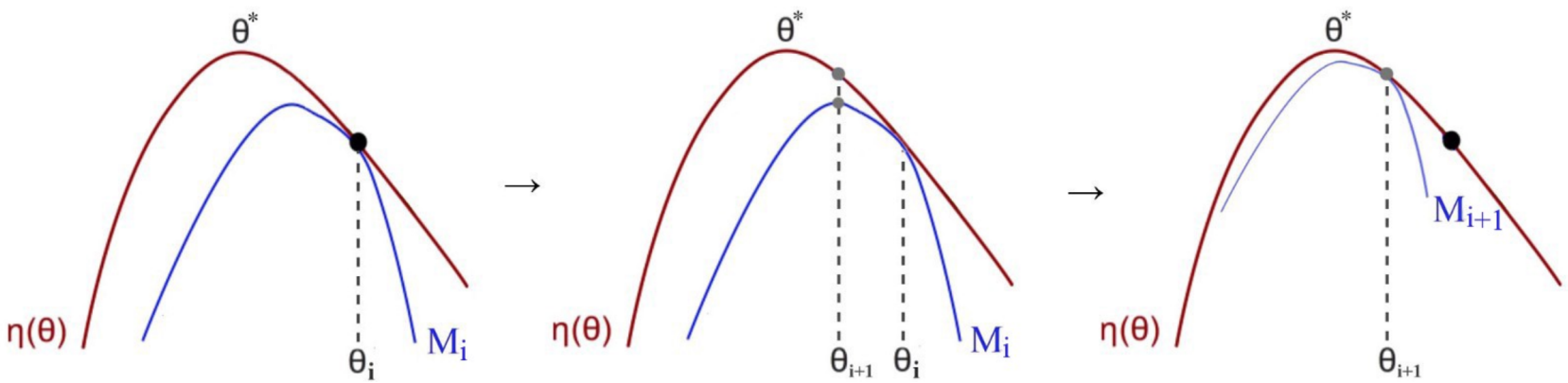
$$\max_{\pi} \; J(\pi) - J(\pi_{old})$$

MM algorithm:

1. Initialize π_0;
2. for $k = 0, 1, \ldots$ until convergence, do

   Solve

$$\pi_{k+1} \in \arg\max_{\pi} \Big\{ \underbrace{L_{\pi_k}(\pi) - C \max_{s} D_{KL}(\pi_k(\cdot|s) \parallel \pi(\cdot|s))}_{\text{surrogate}} \Big\}$$



Advantages and Disadvantages

Advantage:

- Monotonic improvement (Why?)

Advantages and Disadvantages

Advantage:

- Monotonic improvement (Why?)

$$
\begin{aligned}
J(\pi_{k+1}) - J(\pi_k) &\geq \max_{\pi} \left\{ L_{\pi_k}(\pi) - C \max_{s} D_{KL}(\pi_k(\cdot|s) \parallel \pi(\cdot|s)) \right\} \\
&\geq L_{\pi_k}(\pi_k) - C \max_{s} D_{KL}(\pi_k(\cdot|s) \parallel \pi_k(\cdot|s)) \\
&= 0
\end{aligned}
$$

Advantages and Disadvantages

Advantage:

- Monotonic improvement (Why?)

$$
\begin{aligned}
J(\pi_{k+1}) - J(\pi_k) &\geq \max_{\pi} \left\{ L_{\pi_k}(\pi) - C \max_{s} D_{KL}(\pi_k(\cdot|s) \parallel \pi(\cdot|s)) \right\} \\
&\geq L_{\pi_k}(\pi_k) - C \max_{s} D_{KL}(\pi_k(\cdot|s) \parallel \pi_k(\cdot|s)) \\
&= 0
\end{aligned}
$$

Disadvantages:

Advantages and Disadvantages

Advantage:

- Monotonic improvement (Why?)

$$\begin{aligned} J(\pi_{k+1}) - J(\pi_k) &\geq \max_{\pi} \left\{ L_{\pi_k}(\pi) - C \max_{s} D_{KL}(\pi_k(\cdot|s) \parallel \pi(\cdot|s)) \right\} \\ &\geq L_{\pi_k}(\pi_k) - C \max_{s} D_{KL}(\pi_k(\cdot|s) \parallel \pi_k(\cdot|s)) \\ &= 0 \end{aligned}$$

Disadvantages:

- Still small stepsizes

Advantages and Disadvantages

Advantage:

- Monotonic improvement (Why?)

$$\begin{aligned} J(\pi_{k+1}) - J(\pi_k) &\geq \max_{\pi} \left\{ L_{\pi_k}(\pi) - C \max_{s} D_{KL}(\pi_k(\cdot|s) \parallel \pi(\cdot|s)) \right\} \\ &\geq L_{\pi_k}(\pi_k) - C \max_{s} D_{KL}(\pi_k(\cdot|s) \parallel \pi_k(\cdot|s)) \\ &= 0 \end{aligned}$$

Disadvantages:

- Still small stepsizes
- Difficult to solve

Step II: Trust region method

Optimization of the surrogate objective (parameterized version):

$$\max_{\theta} L_{\theta_{old}}(\theta) - C \max_{s} D_{KL}(\theta_{old} \parallel \theta)$$

Step II: Trust region method

Optimization of the surrogate objective (parameterized version):

$$\max_{\theta} L_{\theta_{old}}(\theta) - C \max_{s} D_{KL}(\theta_{old} \parallel \theta)$$

Q) How can we take a larger step?

Step II: Trust region method

Optimization of the surrogate objective (parameterized version):

$$\max_{\theta} L_{\theta_{old}}(\theta) - C \max_{s} D_{KL}(\theta_{old} \parallel \theta)$$

Q) How can we take a larger step?

Step II: Trust region method

Optimization of the surrogate objective (parameterized version):

$$\max_{\theta} L_{\theta_{old}}(\theta) - C \max_{s} D_{KL}(\theta_{old} \parallel \theta)$$

Q) How can we take a larger step?

- Trust region constraint:

Step II: Trust region method

Optimization of the surrogate objective (parameterized version):

$$\max_{\theta} L_{\theta_{old}}(\theta) - C \max_{s} D_{KL}(\theta_{old} \parallel \theta)$$

Q) How can we take a larger step?

- Trust region constraint:

$$\begin{aligned} \max_{\theta} \quad & L_{\theta_{old}}(\theta) \\ \text{s.t.} \quad & \underbrace{\max_{s} D_{KL}(\theta_{old} \parallel \theta) \leq \delta}_{\text{trust region}} \end{aligned}$$

Approximation of trust region constraint

Approximation of trust region constraint

- Max KL divergence $\rightarrow$ Average KL divergence

Approximation of trust region constraint

- Max KL divergence $\rightarrow$ Average KL divergence

$$\begin{aligned} \max_{\theta} \quad & L_{\theta_{old}}(\theta) \\ \text{s.t.} \quad & \underbrace{\mathbb{E}^{\pi_{\theta_{old}}}[D_{KL}(\pi_{\theta_{old}} \parallel \pi_{\theta})] \leq \delta}_{\text{approximate trust region}} \end{aligned}$$

Using samples...

Using samples...

Using samples generated by using $\pi_{\theta_{old}}$ (importance sampling),

Using samples...

Using samples generated by using $\pi_{\theta_{old}}$ (importance sampling),

$$\max_{\theta} \quad \underbrace{\hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t \right]}_{\approx L_{\theta_{old}}(\theta)}$$

$$\text{s.t.} \quad \hat{\mathbb{E}}_t[D_{KL}(\pi_{\theta_{old}}(\cdot|s_t) \parallel \pi_\theta(\cdot|s_t))] \leq \delta$$

Using samples...

Using samples generated by using $\pi_{\theta_{old}}$ (importance sampling),

$$\max_{\theta} \quad \underbrace{\hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t \right]}_{\approx L_{\theta_{old}}(\theta)}$$

$$\text{s.t.} \quad \hat{\mathbb{E}}_t[D_{KL}(\pi_{\theta_{old}}(\cdot|s_t) \parallel \pi_\theta(\cdot|s_t))] \leq \delta$$

- Sample efficiency (off-policy)

TRPO pseudocode

For iteration $= 1, 2, \ldots$ until convergence, do

1. Run old policy for T timesteps (can also use multiple trajectories)
2. Estimate advantage function $\hat{A}_t \approx A(s_t, a_t)$ at all timesteps
3. Solve the (approximate) surrogate problem:

$$\max_{\theta} \quad \sum_{t=1}^{T} \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t$$

$$\text{s.t.} \quad \frac{1}{T} \sum_{t=1}^{T} D_{KL}(\pi_{\theta_{old}}(\cdot|s_t) \parallel \pi_\theta(\cdot|s_t)) \leq \delta$$

Step III: Further approximation – Natural policy gradient

Step III: Further approximation – Natural policy gradient

So far we have

$$\begin{aligned} \max_{\theta} \quad & L_{\theta_{old}}(\theta) \\ \text{s.t.} \quad & \overline{D}_{KL}(\theta_{old} \,\|\, \theta) \leq \delta \end{aligned}$$

Step III: Further approximation – Natural policy gradient

So far we have

$$\begin{aligned} \max_{\theta} \quad & L_{\theta_{old}}(\theta) \\ \text{s.t.} \quad & \overline{D}_{KL}(\theta_{old} \,\|\, \theta) \leq \delta \end{aligned}$$

Q) Efficient solution?

Step III: Further approximation – Natural policy gradient

So far we have

$$\begin{aligned} \max_{\theta} \quad & L_{\theta_{old}}(\theta) \\ \text{s.t.} \quad & \overline{D}_{KL}(\theta_{old} \,\|\, \theta) \leq \delta \end{aligned}$$

Q) Efficient solution?

- Approximation via Taylor expansion:

$$L_{\theta_{old}}(\theta) \approx L_{\theta_{old}}(\theta_{old}) + \underbrace{\nabla_\theta L_{\theta_{old}}(\theta)|_{\theta=\theta_{old}}}_{=:g} \cdot(\theta - \theta_{old})$$

$$\overline{D}_{KL}(\theta_{old} \,\|\, \theta) \approx \frac{1}{2}(\theta - \theta_{old})^\top \underbrace{\nabla^2_\theta \overline{D}_{KL}(\theta_{old} \,\|\, \theta)|_{\theta=\theta_{old}}}_{=:H}(\theta - \theta_{old})$$

Step III: Further approximation – Natural policy gradient

So far we have

$$\begin{aligned} \max_{\theta} \quad & L_{\theta_{old}}(\theta) \\ \text{s.t.} \quad & \overline{D}_{KL}(\theta_{old} \,\|\, \theta) \leq \delta \end{aligned}$$

Q) Efficient solution?

- Approximation via Taylor expansion:

$$L_{\theta_{old}}(\theta) \approx L_{\theta_{old}}(\theta_{old}) + \underbrace{\nabla_\theta L_{\theta_{old}}(\theta)|_{\theta=\theta_{old}}}_{=:g} \cdot(\theta - \theta_{old})$$

$$\overline{D}_{KL}(\theta_{old} \,\|\, \theta) \approx \frac{1}{2}(\theta - \theta_{old})^\top \underbrace{\nabla^2_\theta \overline{D}_{KL}(\theta_{old} \,\|\, \theta)|_{\theta=\theta_{old}}}_{=:H}(\theta - \theta_{old})$$

Approximate problem:

$$\begin{aligned} \max_{\theta} \quad & g^\top(\theta - \theta_{old}) \\ \text{s.t.} \quad & \frac{1}{2}(\theta - \theta_{old})^\top H(\theta - \theta_{old}) \leq \delta \end{aligned}$$

Connection to policy gradient

- Approximate problem:

$$\begin{aligned} \max_{\theta} \quad & g^{\top}(\theta - \theta_{old}) \\ \text{s.t.} \quad & \frac{1}{2}(\theta - \theta_{old})^{\top} H (\theta - \theta_{old}) \leq \delta \end{aligned}$$

Connection to policy gradient

- Approximate problem:

$$\begin{aligned} \max_{\theta} \quad & g^{\top}(\theta - \theta_{old}) \\ \text{s.t.} \quad & \frac{1}{2}(\theta - \theta_{old})^{\top} H (\theta - \theta_{old}) \leq \delta \end{aligned}$$

- Q) Solution?

Connection to policy gradient

- Approximate problem:

$$\begin{aligned} \max_{\theta} \quad & g^\top(\theta - \theta_{old}) \\ \text{s.t.} \quad & \frac{1}{2}(\theta - \theta_{old})^\top H(\theta - \theta_{old}) \leq \delta \end{aligned}$$

- Q) Solution?

$$\theta_{new} = \theta_{old} + \sqrt{\frac{2\delta}{g^\top H^{-1} g}} H^{-1} g$$

Connection to policy gradient

- Approximate problem:

$$\begin{aligned} \max_{\theta} \quad & g^\top(\theta - \theta_{old}) \\ \text{s.t.} \quad & \frac{1}{2}(\theta - \theta_{old})^\top H(\theta - \theta_{old}) \leq \delta \end{aligned}$$

- Q) Solution?

$$\theta_{new} = \theta_{old} + \sqrt{\frac{2\delta}{g^\top H^{-1} g}} H^{-1} g$$

- Q) Looks familiar?

Connection to policy gradient

- Approximate problem:

$$\begin{aligned} \max_{\theta} \quad & g^\top(\theta - \theta_{old}) \\ \text{s.t.} \quad & \frac{1}{2}(\theta - \theta_{old})^\top H(\theta - \theta_{old}) \leq \delta \end{aligned}$$

- Q) Solution?

$$\theta_{new} = \theta_{old} + \sqrt{\frac{2\delta}{g^\top H^{-1} g}} H^{-1} g$$

- Q) Looks familiar?

$$\theta_{new} = \theta_{old} + \underbrace{\sqrt{\frac{2\delta}{g^\top H^{-1} g}}}_{=:\alpha \text{ (stepsize)}} H^{-1} \underbrace{\nabla_\theta L_{\theta_{old}}(\theta_{old})}_{=g \text{ (policy gradient)}}$$

Natural policy gradient

- Approximate solution:

$$\theta_{new} = \theta_{old} + \underbrace{\sqrt{\frac{2\delta}{g^\top H^{-1} g}}}_{=:\alpha \text{ (stepsize)}} H^{-1} \underbrace{\nabla_\theta L_{\theta_{old}}(\theta_{old})}_{=g \text{ (policy gradient)}}$$

Natural policy gradient

- Approximate solution:

$$\theta_{new} = \theta_{old} + \underbrace{\sqrt{\frac{2\delta}{g^\top H^{-1} g}}}_{=:\alpha \text{ (stepsize)}} H^{-1} \underbrace{\nabla_\theta L_{\theta_{old}}(\theta_{old})}_{=g \text{ (policy gradient)}}$$

- Natural policy gradient: $H^{-1} \nabla_\theta L_\theta(\theta)$

Natural policy gradient

- Approximate solution:

$$\theta_{new} = \theta_{old} + \underbrace{\sqrt{\frac{2\delta}{g^\top H^{-1} g}}}_{=:\alpha \text{ (stepsize)}} H^{-1} \underbrace{\nabla_\theta L_{\theta_{old}}(\theta_{old})}_{=g \text{ (policy gradient)}}$$

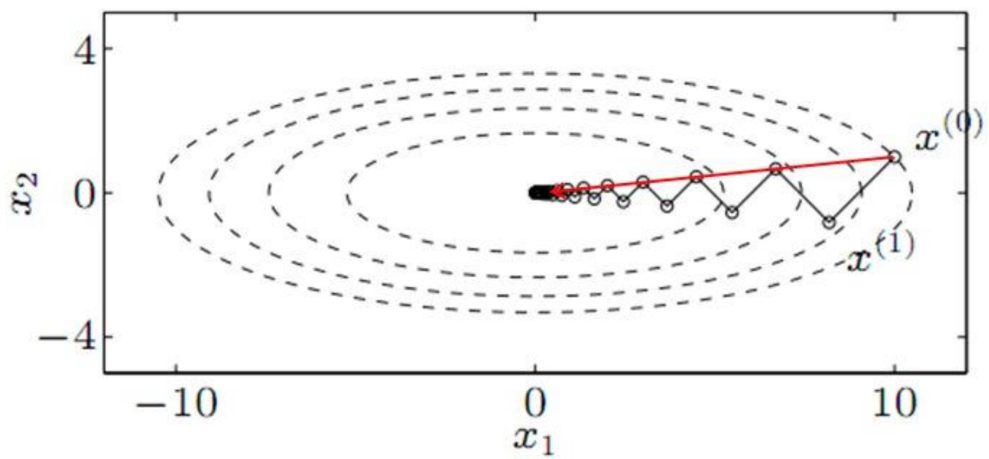
- Natural policy gradient: $H^{-1}\nabla_\theta L_\theta(\theta)$
  - More reasonable direction to move than PG

Natural policy gradient

- Approximate solution:

$$\theta_{new} = \theta_{old} + \underbrace{\sqrt{\frac{2\delta}{g^\top H^{-1} g}}}_{=:\alpha \text{ (stepsize)}} H^{-1} \underbrace{\nabla_\theta L_{\theta_{old}}(\theta_{old})}_{=g \text{ (policy gradient)}}$$

- Natural policy gradient: $H^{-1}\nabla_\theta L_\theta(\theta)$
  - More reasonable direction to move than PG
  - Intuition:



Example: Policy gradient vs Natural policy gradient

Neural Networks 21 (2008) 682–697

Contents lists available at ScienceDirect

Neural Networks

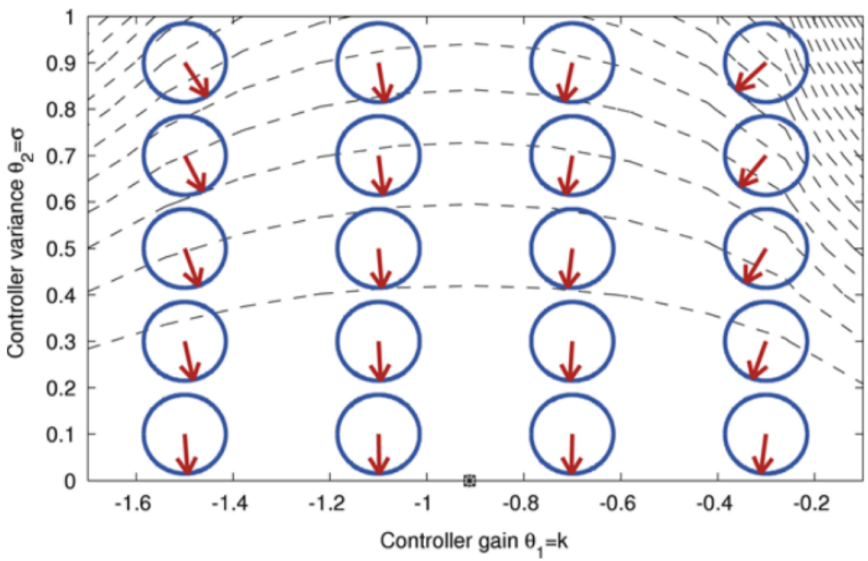
journal homepage: www.elsevier.com/locate/neunet

2008 Special Issue

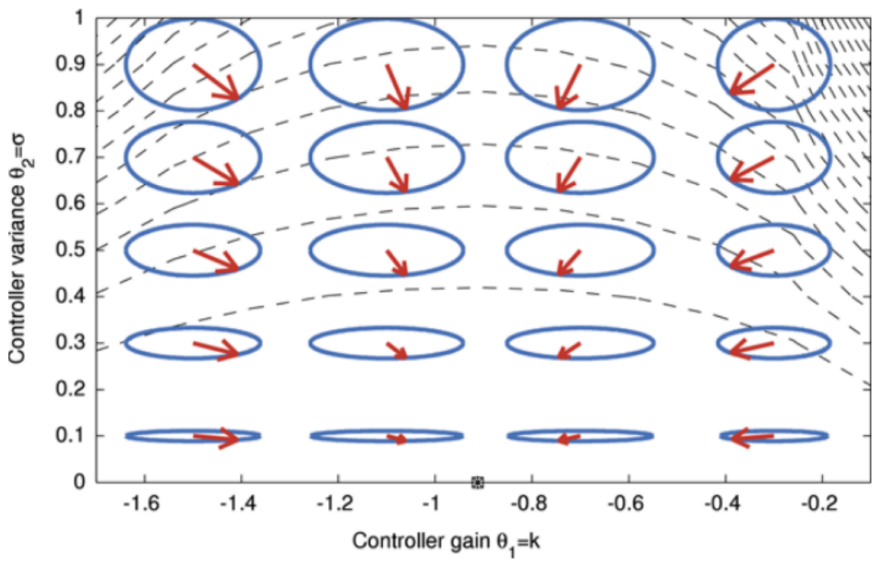
Reinforcement learning of motor skills with policy gradients

Jan Peters [a,b,*], Stefan Schaal [b,c]

[a] Max Planck Institute for Biological Cybernetics, Spemannstr. 38, 72076 Tübingen, Germany
[b] University of Southern California, 3710 S. McClintoch Ave – RTH401, Los Angeles, CA 90089-2905, USA
[c] ATR Computational Neuroscience Laboratory, 2-2-2 Hikaridai, Seika-cho, Soraku-gun Kyoto 619-0288, Japan



(a) Vanilla policy gradient.



(b) Natural policy gradient.

Natural policy gradient (NPG) algorithm

For iteration $k = 0, 1, \ldots$ until convergence, do

1. Collect set of trajectories using π_{θ_k};
2. Estimate $\hat{A}_t$;
3. Estimate
   1. policy gradient $\hat{g}_k$;
   2. KL-divergence Hessian $\hat{H}_k$;
4. Update:

$$\theta_{k+1} := \theta_k + \sqrt{\frac{2\delta}{\hat{g}_k^\top \hat{H}^{-1} \hat{g}_k}} \hat{H}_k^{-1} \hat{g}_k$$

Issues in NPG algorithm

Issues in NPG algorithm

Because of approximation:

Issues in NPG algorithm

Because of approximation:

- At some iteration trust region size δ may be too large
  $\Longrightarrow$ performance can degrade

Issues in NPG algorithm

Because of approximation:

- At some iteration trust region size δ may be too large
  $\Longrightarrow$ performance can degrade
- KL-divergence constraint may be violated

Issues in NPG algorithm

Because of approximation:

- At some iteration trust region size δ may be too large
  $\Longrightarrow$ performance can degrade
- KL-divergence constraint may be violated

Q) How to resolve these issues?

Issues in NPG algorithm

Because of approximation:

- At some iteration trust region size δ may be too large
  $\Longrightarrow$ performance can degrade
- KL-divergence constraint may be violated

Q) How to resolve these issues?

- Require improvement in surrogate:
  Make sure $L_{\theta_k}(\theta_{k+1}) \geq 0$

Issues in NPG algorithm

Because of approximation:

- At some iteration trust region size δ may be too large
  $\Longrightarrow$ performance can degrade
- KL-divergence constraint may be violated

Q) How to resolve these issues?

- Require improvement in surrogate:
  Make sure $L_{\theta_k}(\theta_{k+1}) \geq 0$
- Enforce KL-constraint

Step IV: How? Line search!

Step IV: How? Line search!

Compute the **full stepsize**: $\Delta_k := \sqrt{\frac{2\delta}{\hat{g}_k^\top \hat{H}^{-1} \hat{g}_k}} \hat{H}_k^{-1} \hat{g}_k$;

For $j = 0, 1, 2, \ldots, L$, do

1. Compute the proposed update $\theta := \theta_k + \underbrace{\alpha^j \Delta_k}_{\text{shrink stepsize}}$;
2. if $L_{\theta_k}(\theta) \geq 0$ and $\overline{D}_{KL}(\theta \,\|\, \theta_k) \leq \delta$, then
   accept the update and
   set $\theta_{k+1} := \theta_k + \alpha^j \Delta_k$;
   break;

Putting everything together: Full TRPO algorithm

For $k = 0, 1, \ldots$ until convergence, do

1. Collect set of trajectories using π_{θ_k};
2. Estimate $\hat{A}_t$;
3. Estimate
   1. policy gradient $\hat{g}_k$;
   2. KL-divergence Hessian $\hat{H}_k$;
4. Use conjugate gradient algorithm to obtain $x_k \approx \underbrace{\hat{H}_k^{-1}\hat{g}_k}_{\text{natural PG}}$ without inverting $\hat{H}_k$;
5. Estimate the proposed step $\Delta_k \approx \sqrt{\frac{2\delta}{\hat{g}_k^\top \hat{H}^{-1}\hat{g}_k}} x_k$;
6. Perform line search with exponential decay to obtain the final update

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

TRPO results

Application of TRPO: Legged robots

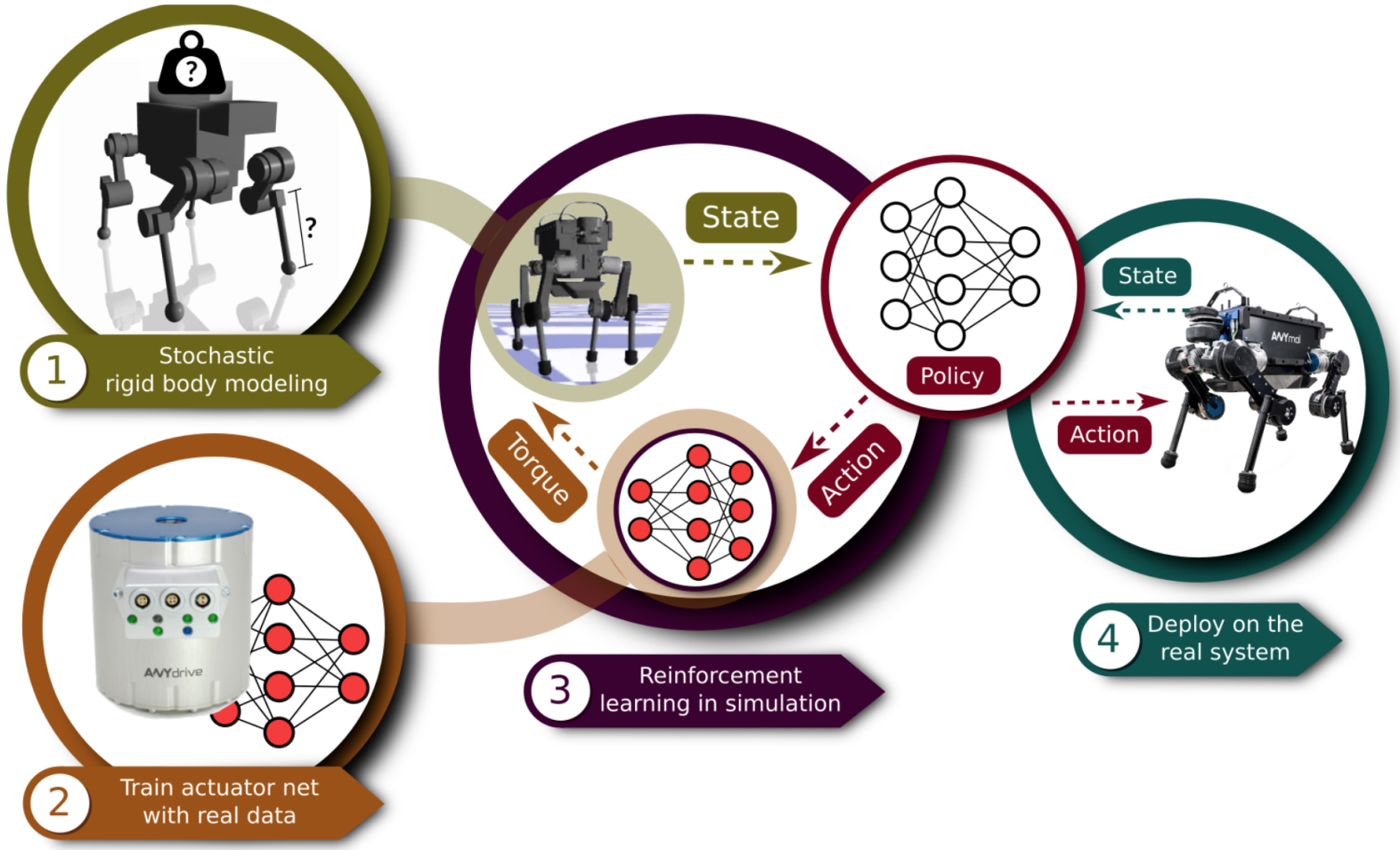
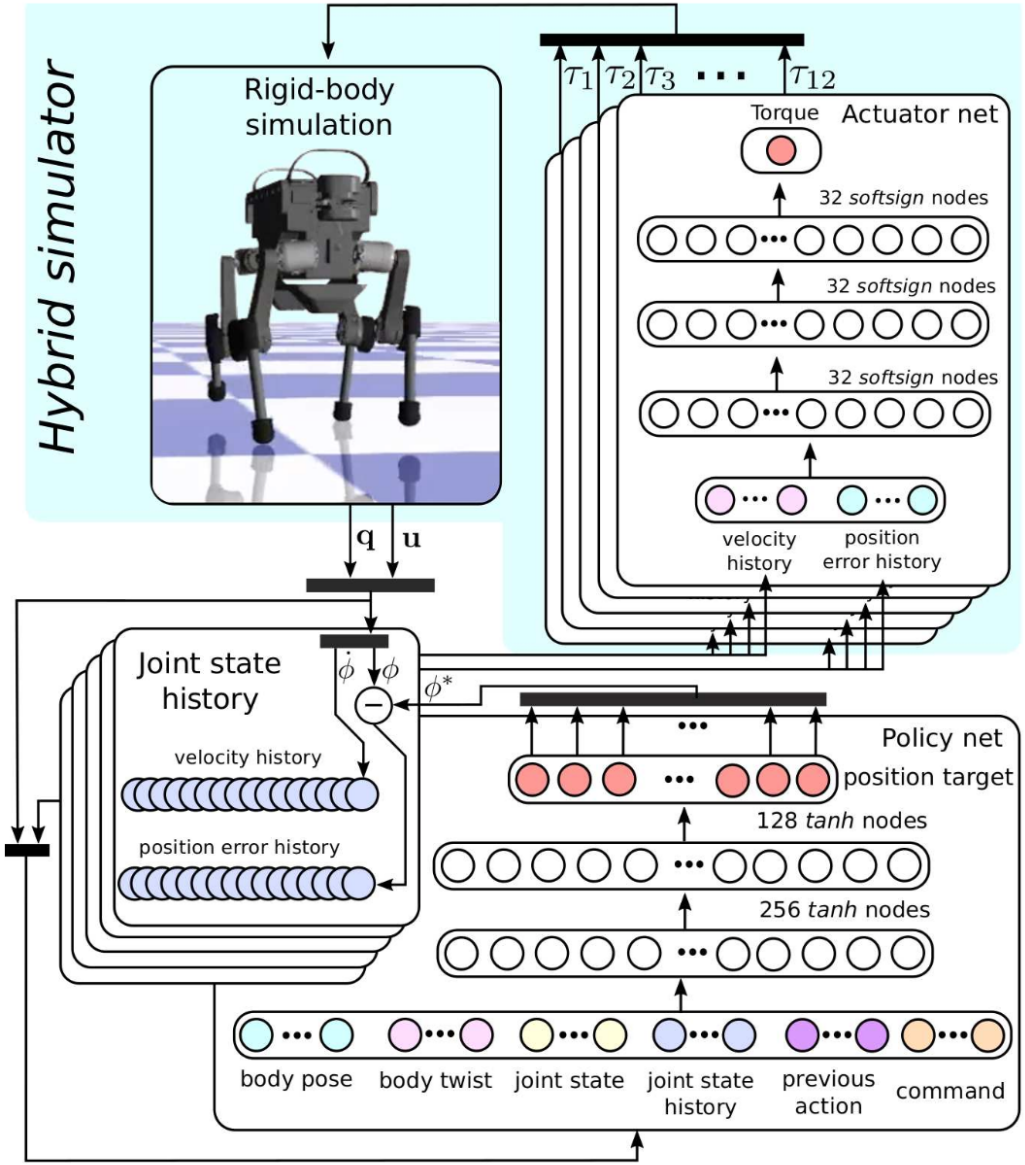


Fig. 1. Creating a control policy. In the first step, we identify the physical parameters of the robot and estimate uncertainties in the identification. In the second step, we train an actuator net that models complex actuator/software dynamics. In the third step, we train a control policy using the models produced in the first two steps. In the fourth step, we deploy the trained policy directly on the physical system.

Application of TRPO: Legged robots



Computational issue: Inverse of KL divergence Hessian

Optimization problem used in TRPO:

$$\begin{aligned} \max_{\theta} \quad & L_{\theta_{old}}(\theta) \\ \text{s.t.} \quad & \overline{D}_{KL}(\theta_{old} \,\|\, \theta) \leq \delta \end{aligned}$$

- Approximation via Taylor expansion:

$$L_{\theta_{old}}(\theta) \approx L_{\theta_{old}}(\theta_{old}) + \underbrace{\nabla_\theta L_{\theta_{old}}(\theta)|_{\theta=\theta_{old}}}_{=:g} \cdot (\theta - \theta_{old})$$

$$\overline{D}_{KL}(\theta_{old} \,\|\, \theta) \approx \frac{1}{2}(\theta - \theta_{old})^\top \underbrace{\nabla_\theta^2 \overline{D}_{KL}(\theta_{old} \,\|\, \theta)|_{\theta=\theta_{old}}}_{=:H} (\theta - \theta_{old})$$

Approximate problem:

$$\begin{aligned} \max_{\theta} \quad & g^\top (\theta - \theta_{old}) \\ \text{s.t.} \quad & \frac{1}{2}(\theta - \theta_{old})^\top H (\theta - \theta_{old}) \leq \delta \end{aligned}$$

How to improve computational efficiency?

- Hessian (second order information) demands a significant computational effort
- Can we only use first order information modifying TRPO?
  $\Longrightarrow$ Proximal policy optimization (PPO)

Proximal Policy Optimization Algorithms

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov
OpenAI
{joschu, filip, prafulla, alec, oleg}@openai.com

Abstract

We propose a new family of policy gradient methods for reinforcement learning, which alternate between sampling data through interaction with the environment, and optimizing a "surrogate" objective function using stochastic gradient ascent. Whereas standard policy gradient methods perform one gradient update per data sample, we propose a novel objective function that enables multiple epochs of minibatch updates. The new methods, which we call proximal policy optimization (PPO), have some of the benefits of trust region policy optimization (TRPO), but they are much simpler to implement, more general, and have better sample complexity (empirically). Our experiments test PPO on a collection of benchmark tasks, including simulated robotic locomotion and Atari game playing, and we show that PPO outperforms other online policy gradient methods, and overall strikes a favorable balance between sample complexity, simplicity, and wall-time.

Idea I: KL penalty

Instead of using KL constraint, add KL penalty to the objective function:

$$\max_{\theta} \quad L_{\theta_{old}}(\theta) - \underbrace{\beta \overline{D}_{KL}(\theta_{old} \parallel \theta)}_{\text{KL penalty}}$$

- It penalizes the objective if the new policy is different from the old policy (effect of trust region).
- Can dynamically adjust β: If the KL penalty is higher than a target value, we shrink β.

Proximal policy optimization (PPO) with adaptive KL penalty

For $k = 0, 1, \ldots$ until convergence, do

1. Collect set of trajectories using π_{θ_k};
2. Estimate $\hat{A}_t$;
3. Compute policy parameter update

$$\theta_{k+1} \in \arg\max_{\theta} L_{\theta_k}(\theta) - \beta_k \overline{D}_{KL}(\theta_k \parallel \theta)$$

4. If $\overline{D}_{KL}(\theta_k \parallel \theta_{k+1}) \geq 1.5\delta$, then
   $\beta_{k+1} \leftarrow 2\beta_k$; (shrink trust region)
   else if $\overline{D}_{KL}(\theta_k \parallel \theta_{k+1}) \leq \delta/1.5$, then
   $\beta_{k+1} \leftarrow \beta_k/2$; (expand trust region)

Idea II: Clipped objective

Recall the surrogate objective function:

$$L_{\theta_{old}}(\theta) = \hat{\mathbb{E}}_t\left[\underbrace{\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}}_{:=r_t(\theta) \text{ prob. ratio}} \hat{A}_t \right] = \hat{\mathbb{E}}_t[r_t(\theta)\hat{A}_t]$$

- Without a trust region constraint or penalty, maximization of $L_{\theta_{old}}(\theta)$ would lead to an excessively large policy.
- Idea: Penalize changes to the policy that move $r_t(\theta)$ away from 1.
- How?
  Clip $r_t(\theta)$ if $r_t(\theta)$ falls outside the interval $[1-\epsilon, 1+\epsilon]$.

Clipped surrogate objective function

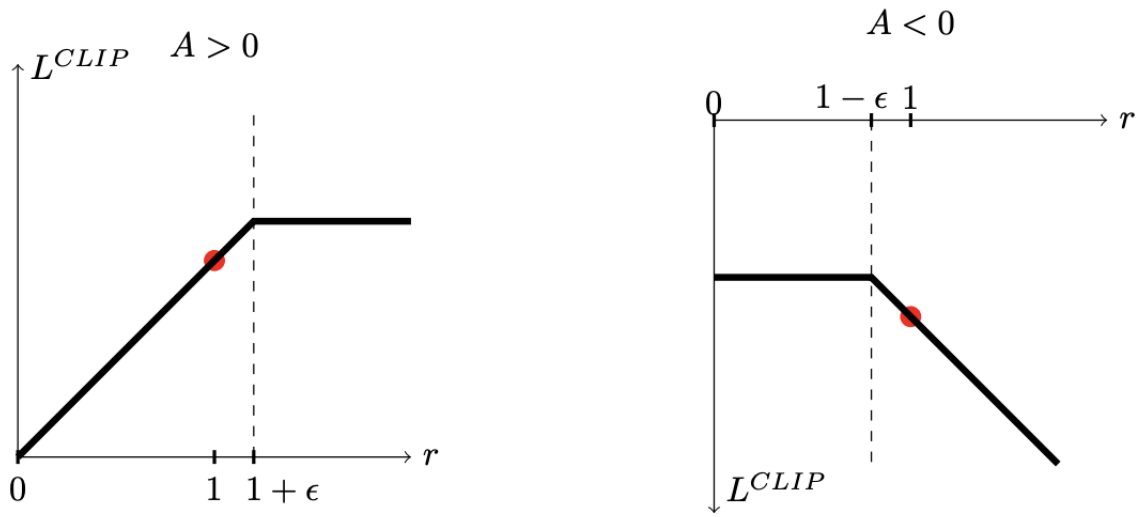
Idea: Penalize changes to the policy that move $r_t(\theta)$ away from 1.

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

- The second term modifies the surrogate objective by clipping the probability ratio, which removes the incentive for moving r_t outside of the interval $[1-\epsilon, 1+\epsilon]$.
- Take the minimum of the clipped and unclipped objective, so the final objective is a lower bound (i.e., a pessimistic bound) on the unclipped objective
  $\Longrightarrow$ Ignore the change in probability ratio when it would make the objective improve, and include it when it makes the objective worse!

Illustration of clipped objective

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



- One term (i.e., a single timestep) of the surrogate 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 there is no incentive move outside the interval $[1-\epsilon, 1+\epsilon]$ due to the clipping.

PPO with clipped objective

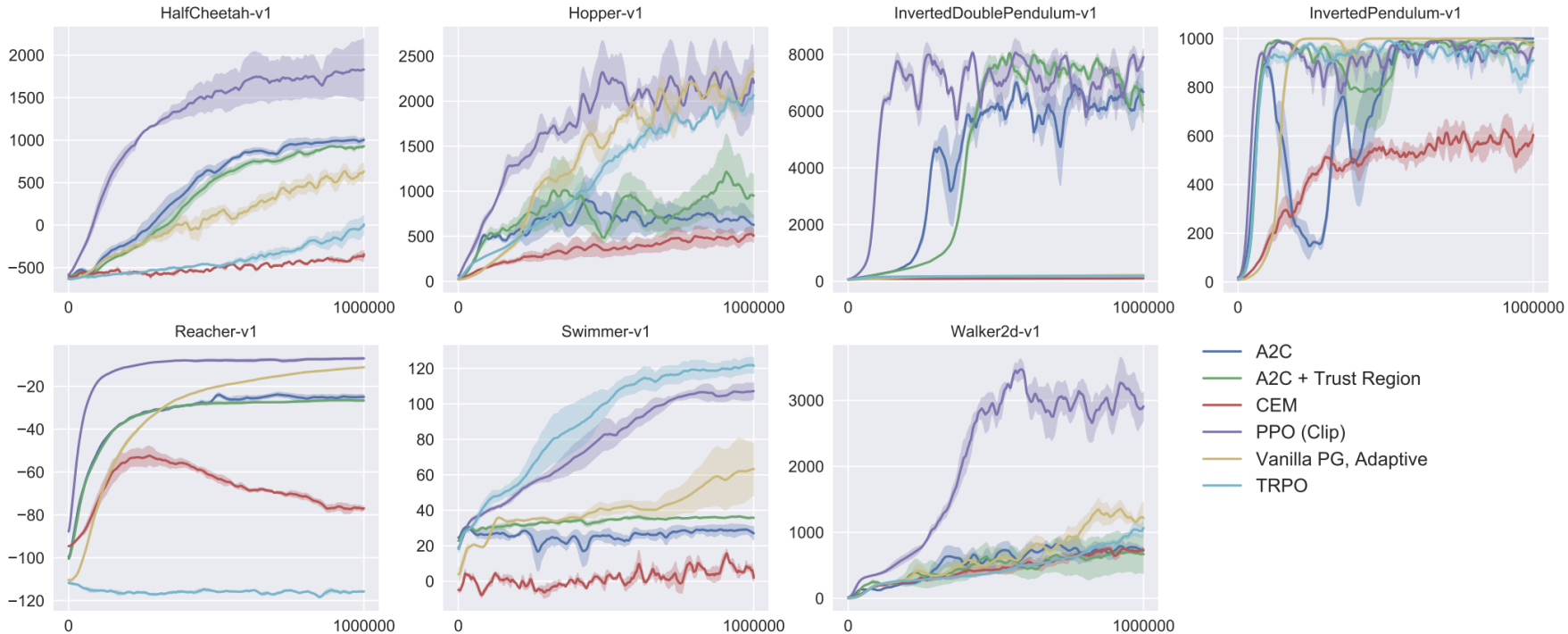
For $k = 0, 1, \ldots$ until convergence, do

1. Collect set of trajectories using π_{θ_k};
2. Estimate $\hat{A}_t$;
3. Compute policy parameter update

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

- The PPO optimization problem can be solved by various first-order algorithms suitable for large-scale problems (No need to compute Hessian unlike TRPO!)

PPO results


HalfCheetah-v1
Hopper-v1
InvertedDoublePendulum-v1
InvertedPendulum-v1
Reacher-v1
Swimmer-v1
Walker2d-v1
A2C
A2C + Trust Region
CEM
PPO (Clip)
Vanilla PG, Adaptive
TRPO