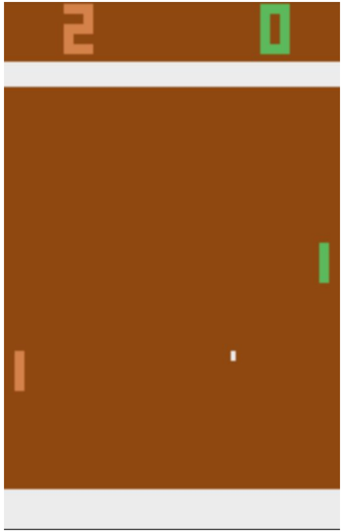
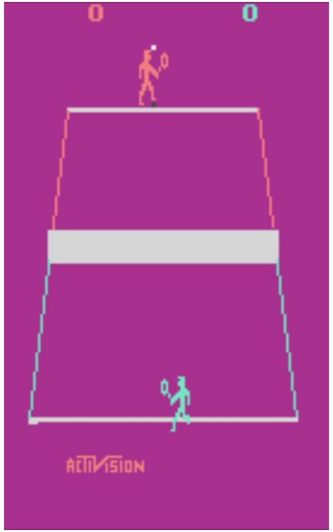
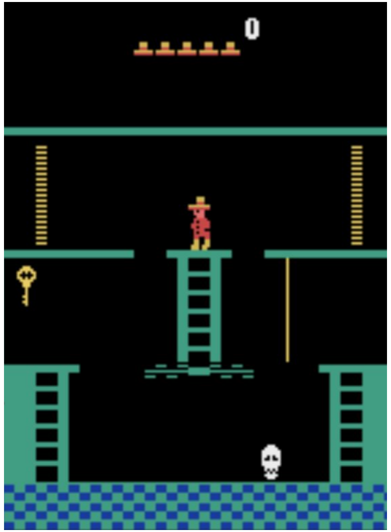
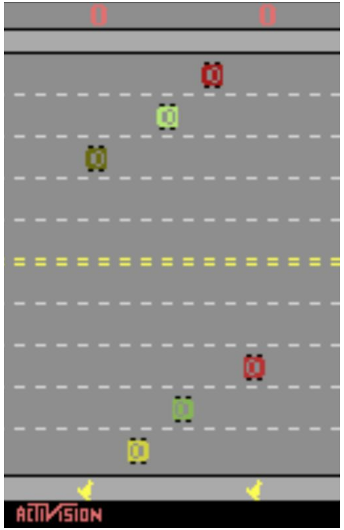
Deep RL in continuous action spaces

On-Policy set up

-- Samriddhi Jain

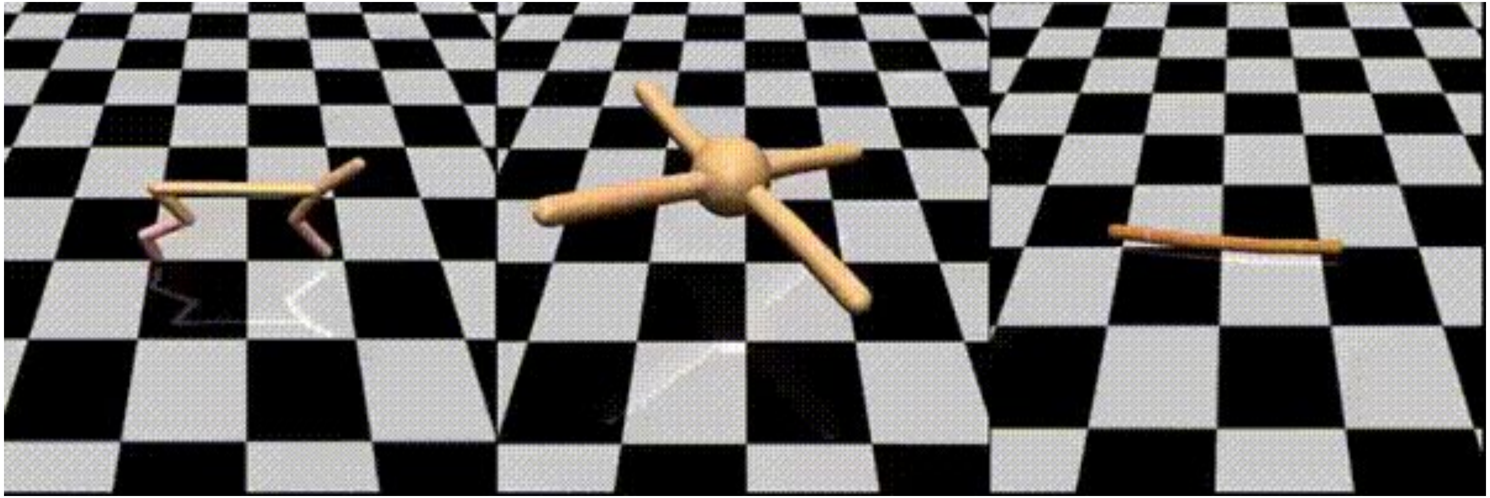
Discrete action environments

- Atari, Board games

Continuous action environments

- MuJoCo (Multi-Joint dynamics with Contact) environments

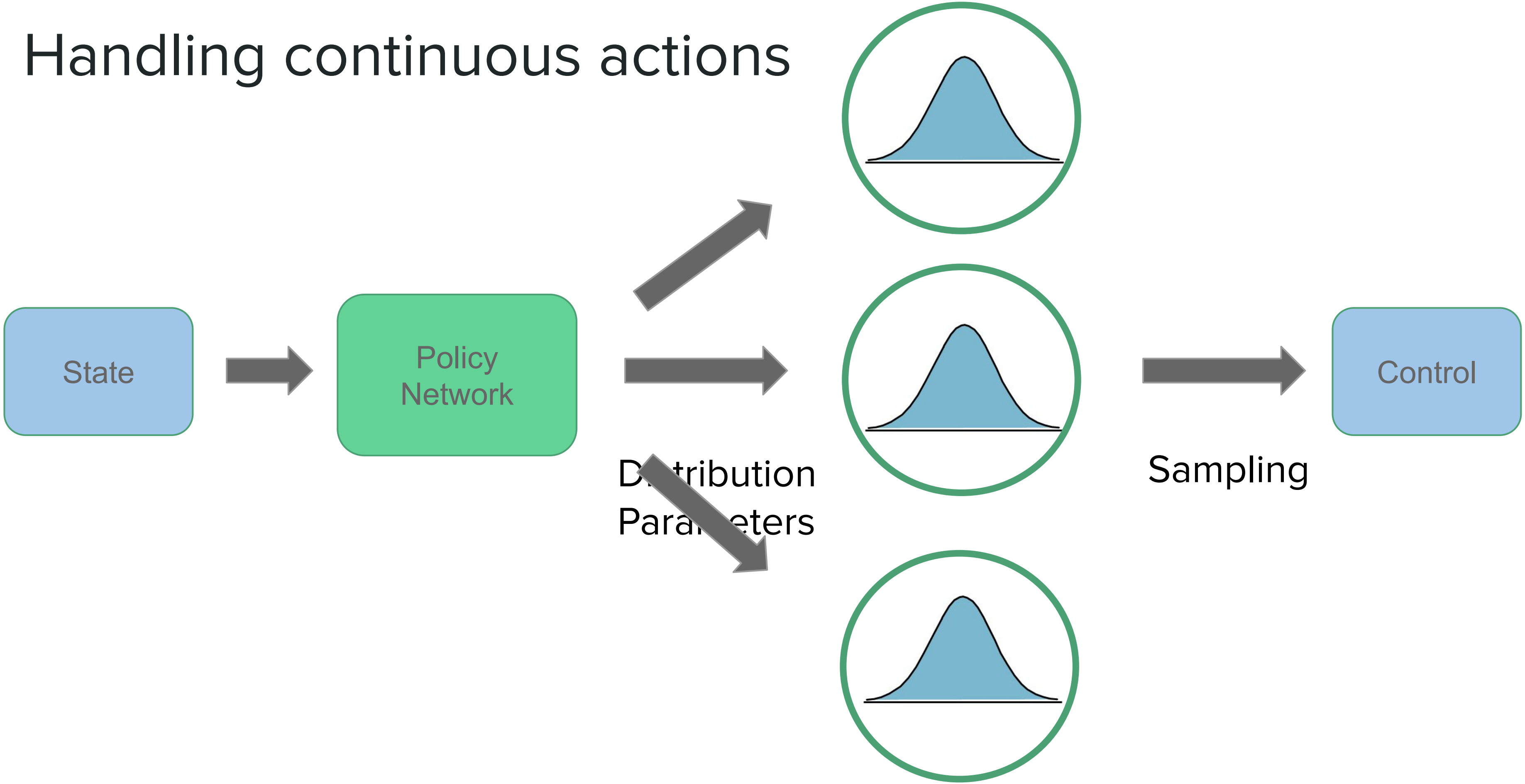


Continuous action environments

- Simulated goal-based tasks for the Fetch and ShadowHand robots

https://openai.com/blog/ingredients-for-robotics-research/

Handling continuous actions



Policy Gradients: review

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t\left[\log \pi_\theta(a_t \mid s_t)\hat{A}_t\right]$$

Pitfalls:

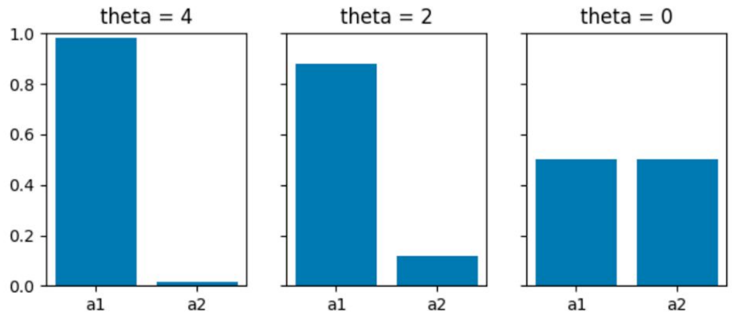
- Sample inefficiency

Policy Gradients: step size and convergence

Policy Gradients: pitfalls

- Relation between policy space and model parameter space?

$$\pi_\theta(a) = \begin{cases} \sigma(\theta) & a = 1 \\ 1 - \sigma(\theta) & a = 2 \end{cases}$$



Research works covered

- Approximately Optimal Approximate Reinforcement Learning, Kakade and Langford 2002
- Trust Region Policy Optimization, Schulman et al. 2015
- Proximal Policy Optimization Algorithms, Schulman et al. 2017

Surrogate Loss

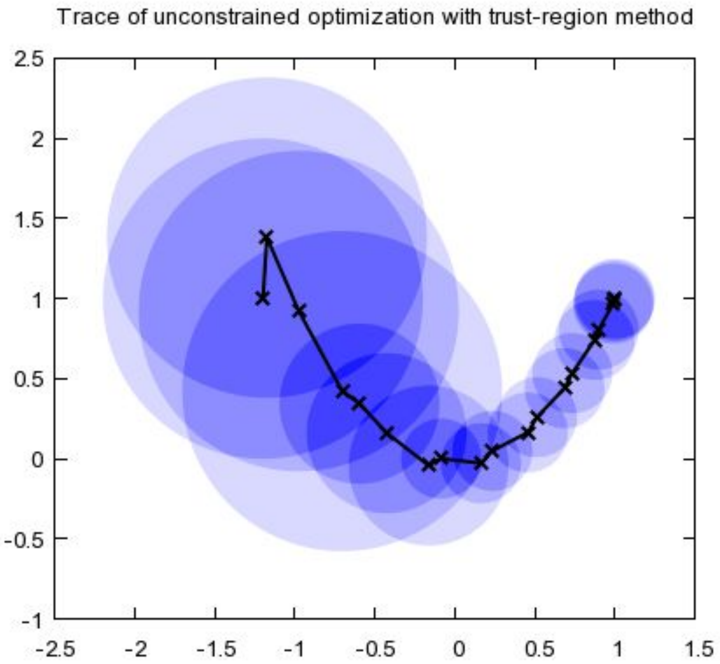
- How to improve sample efficiency?
  - Use trajectories from other policies with importance sampling

$$\underset{\theta}{\text{maximize}} \quad \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t \right]$$

- Gradients are same

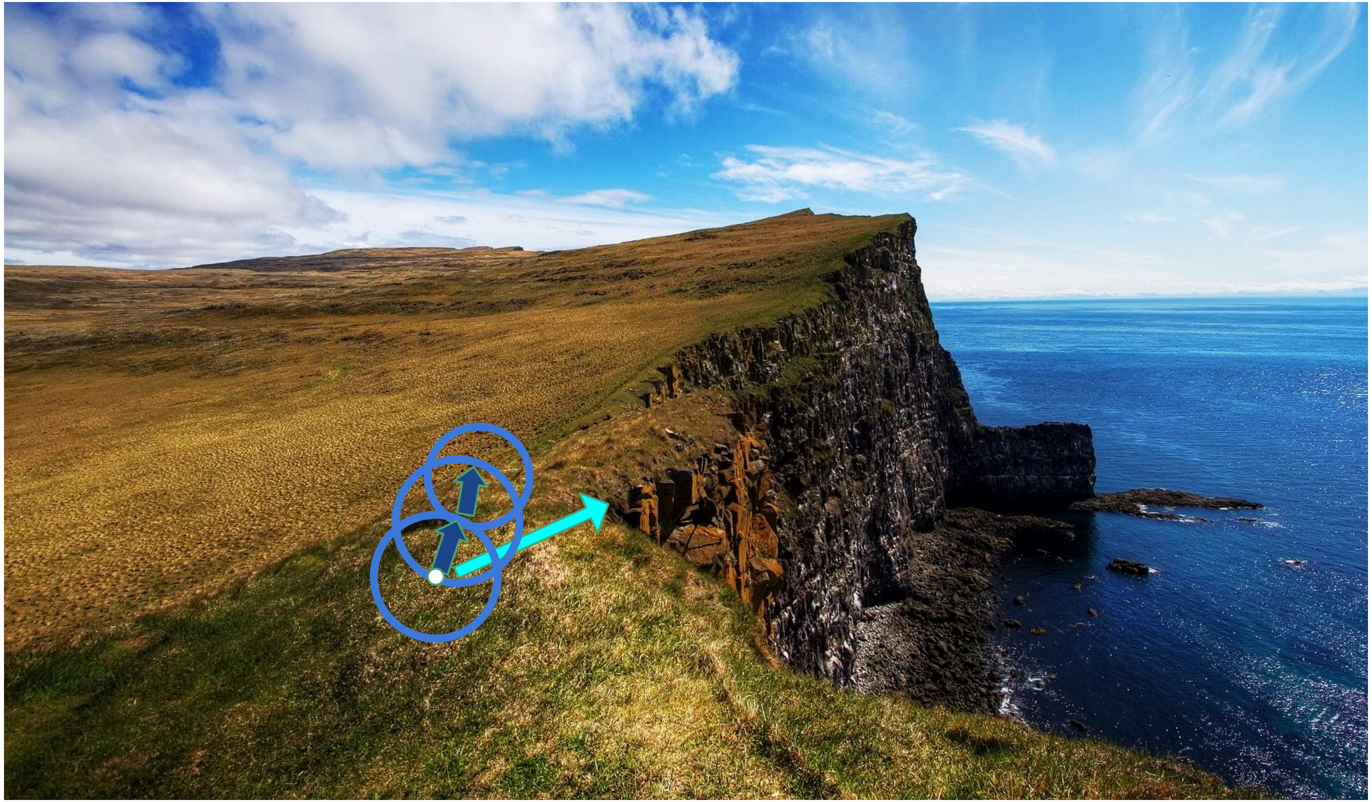
Trust Region Methods

Search in a **trusted** region



https://reference.wolfram.com/language/tutorial/UnconstrainedOptimizationTrustRegionMethods.html
https://optimization.mccormick.northwestern.edu/index.php/Trust-region_methods

Line search vs Trust region search



New loss

- Constrained:

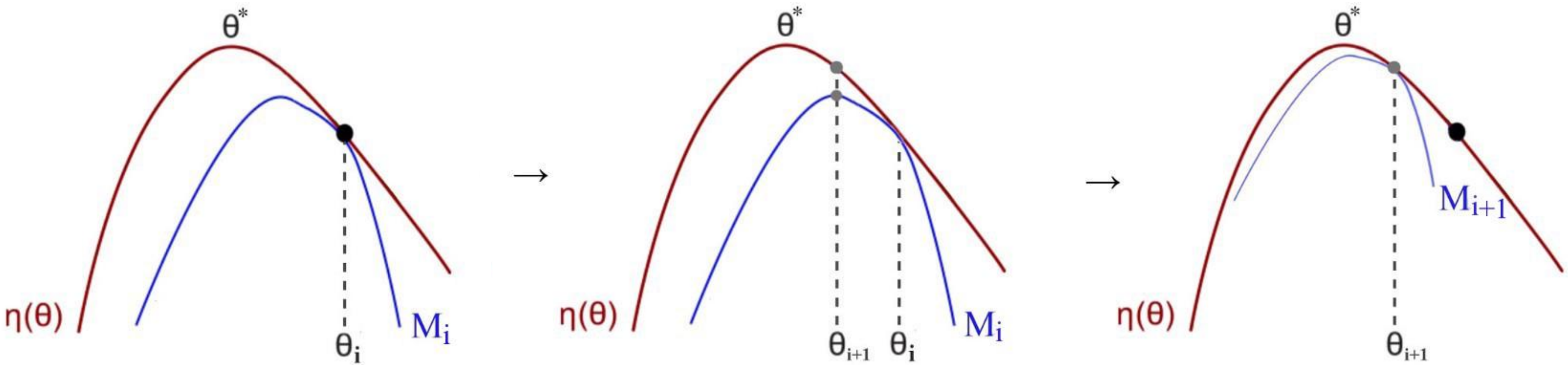
$$\underset{\theta}{\text{maximize}} \quad \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t\right]$$
$$\text{subject to} \quad \hat{\mathbb{E}}_t[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]] \le \delta.$$

- Penalty:

$$\underset{\theta}{\text{maximize}} \qquad \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t\right] - \beta\hat{\mathbb{E}}_t[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]]$$

In practise, β is harder to tune than □.

Minorize Maximization theorem



$$\eta(\pi) = \mathbb{E}_{s_0, a_0, \ldots}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)\right]$$

$$\rho_\pi(s) = P(s_0 = s) + \gamma P(s_1 = s) + \gamma^2 P(s_2 = s) + \ldots,$$

$$\eta(\tilde{\pi}) = \eta(\pi) + \mathbb{E}_{s_0, a_0, \cdots \sim \tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t A_\pi(s_t, a_t)\right]$$

By Kakade and Langford (2002)

$$\eta(\tilde{\pi}) = \eta(\pi) + \sum_{t=0}^{\infty} \sum_s P(s_t = s | \tilde{\pi}) \sum_a \tilde{\pi}(a|s) \gamma^t A_\pi(s, a)$$

$$= \eta(\pi) + \sum_s \sum_{t=0}^{\infty} \gamma^t P(s_t = s | \tilde{\pi}) \sum_a \tilde{\pi}(a|s) A_\pi(s, a)$$

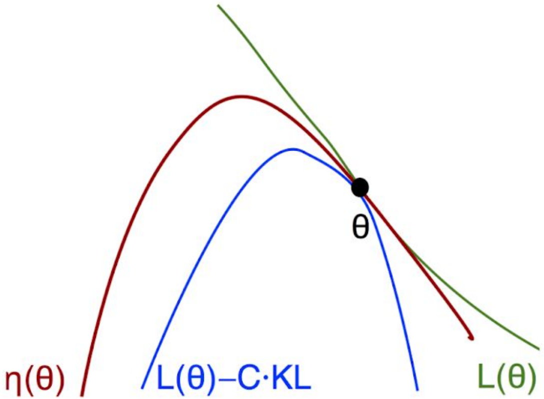
$$\eta(\tilde{\pi}) = \eta(\pi) + \sum_s \underline{\rho_{\tilde{\pi}}(s)} \sum_a \tilde{\pi}(a|s) A_\pi(s, a)$$

Local approximation

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

$$\eta(\widetilde{\pi}) \geq L_{\pi}(\widetilde{\pi}) - C D_{KL}^{max}(\pi, \widetilde{\pi})$$

$$C = \frac{4\epsilon\gamma}{(1-\gamma)^2}, \epsilon = \max_{s,a}|A_{\pi}(s,a)|$$



$$\eta(\theta) \geq L_{\theta_{old}}(\theta) - C D_{KL}^{max}(\theta_{old}, \theta)$$

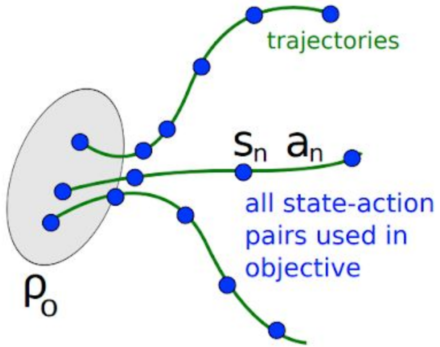
$$\underset{\theta}{maximize} \;\; L_{\theta_{old}}(\theta)$$
$$subject\ to\ D_{KL}^{max}(\theta_{old}, \theta) \leq \delta$$

$$\underset{\theta}{maximize} \;\; L_{\theta_{old}}(\theta)$$
$$subject\ to\ \overline{D}_{KL}(\theta_{old}, \theta) \leq \delta$$

Sample based estimation

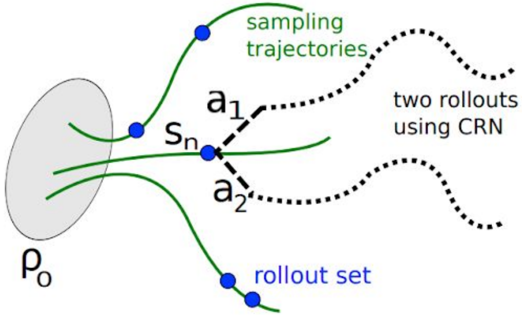
$$\underset{\theta}{maximize} \sum_s \rho_{\theta_{old}}(s) \sum_a \pi_\theta(a|s) A_{\theta_{old}}(s,a)$$

$$subject\ to\ \overline{D}_{KL}(\theta_{old}, \theta) \leq \delta$$



$$\underset{\theta}{maximize}\ \mathbb{E}_{s\sim\rho_{\theta old}, a\sim\pi_{\theta_{old}}}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)} A_{\theta_{old}}(s,a)\right]$$

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



Natural Policy Gradients

$$\theta_{k+1} = \arg\max_{\theta} \ \mathcal{L}_{\theta_k}(\theta) \text{ s.t. } \bar{D}_{KL}(\theta||\theta_k) \leq \delta$$

- Approximate using taylor expansion:

$$\mathcal{L}_{\theta_k}(\theta) \approx \mathcal{L}_{\theta_k}(\theta_k) + g^T(\theta - \theta_k) + \ldots$$

$$\bar{D}_{KL}(\theta||\theta_k) \approx \bar{D}_{KL}(\theta_k||\theta_k) + \nabla_\theta \bar{D}_{KL}(\theta||\theta_k)|_{\theta_k}(\theta - \theta_k) + \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) + \ldots$$

Loss reduces to the first order term, constraint to second order term

Natural Policy Gradients

$$\theta_{k+1} = \arg\max_{\theta} \; g^T(\theta - \theta_k)$$

$$\text{s.t. } \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \leq \delta$$

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

$\xrightarrow{O(N^3)}$

Computationally (very?) expensive

Truncated Natural Policy Gradient

- Solutions:
  - Use kronecker vector product: **ACKTR**
  - Use Conjugate gradients to compute $H^{-1}g$ without actually inverting H:

    Truncated Natural Policy Gradient

Line search for TRPO

- With the quadratic approximations, the constraint may be violated

- Solution: Enforce KL constraint, backtracking line search with exponential decay

---

Algorithm 2 Line Search for TRPO

Compute proposed policy step $\Delta_k = \sqrt{\frac{2\delta}{\hat{g}_k^T \hat{H}_k^{-1} \hat{g}_k}} \hat{H}_k^{-1} \hat{g}_k$

for $j = 0, 1, 2, ..., L$ **do**

  Compute proposed update $\theta = \theta_k + \alpha^j \Delta_k$

  if $\mathcal{L}_{\theta_k}(\theta) \geq 0$ and $\bar{D}_{KL}(\theta || \theta_k) \leq \delta$ **then**

    accept the update and set $\theta_{k+1} = \theta_k + \alpha^j \Delta_k$

    break

  end if

end for

Trust Region Policy Optimization

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 $x_k \approx \hat{H}_k^{-1} \hat{g}_k$

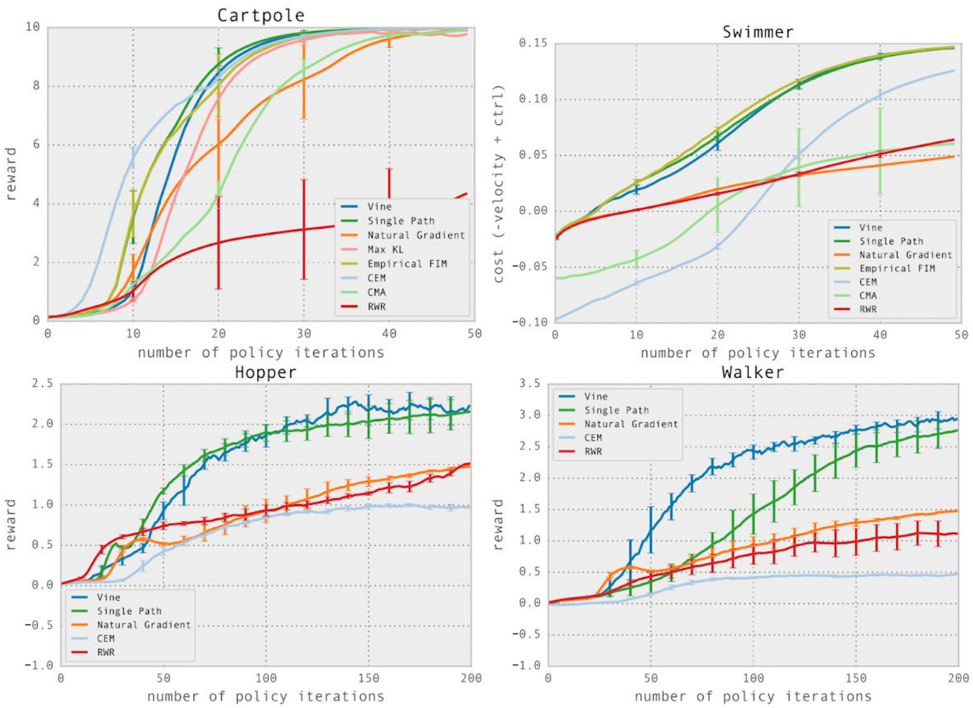
Estimate proposed step $\Delta_k \approx \sqrt{\frac{2\delta}{x_k^T \hat{H}_k x_k}} x_k$

Perform backtracking line search with exponential decay to obtain final update

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

end for

Results



Cartpole
reward
number of policy iterations
Vine
Single Path
Natural Gradient
Max KL
Empirical FIM
CEM
CMA
RWR
Swimmer
cost (-velocity + ctrl)
number of policy iterations
Vine
Single Path
Natural Gradient
Empirical FIM
CEM
CMA
RWR
Hopper
reward
number of policy iterations
Vine
Single Path
Natural Gradient
CEM
RWR
Walker
reward
number of policy iterations
Vine
Single Path
Natural Gradient
CEM
RWR

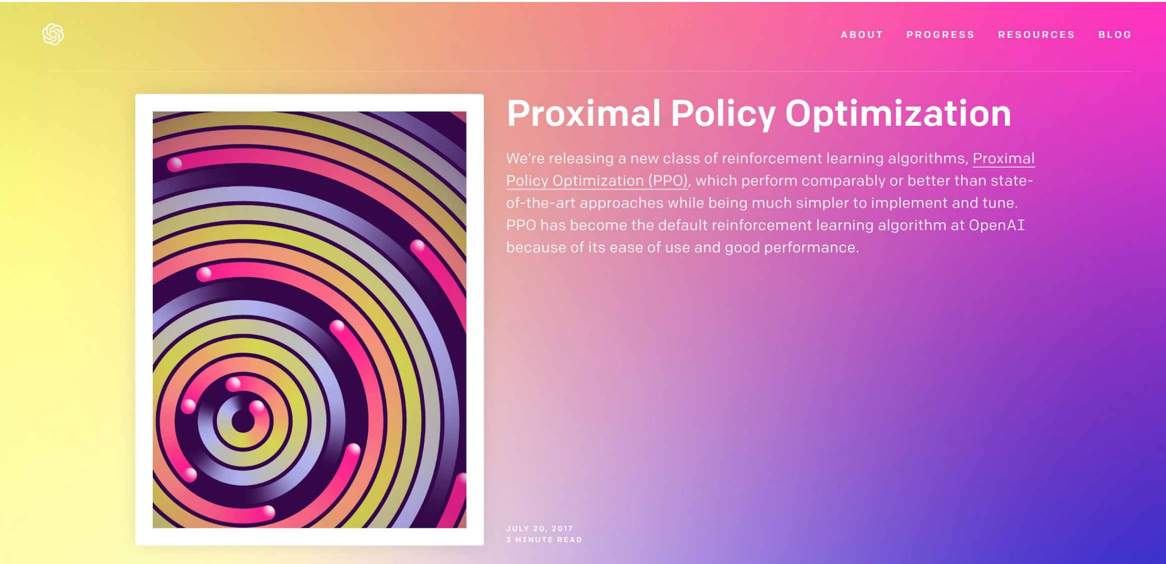
Issues with TRPO

- Doesn’t work well with CNNs and RNNs
- Scalability
- Complexity

Proximal Policy Optimization (PPO)

- Motivation is same as that of TRPO
- Uses first order derivative solutions
- Designed to be simpler to implement
- Two versions:
  - PPO Penalty
  - PPO Clip

https://openai.com/blog/openai-baselines-ppo/



PPO adaptive KL penalty

- Considers the unconstrained objective

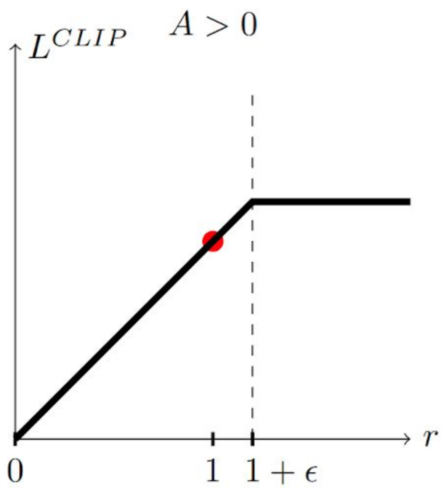
$$\theta_{k+1} = \arg\max_{\theta} \; \mathcal{L}_{\theta_k}(\theta) - \beta_k \bar{D}_{KL}(\theta || \theta_k)$$

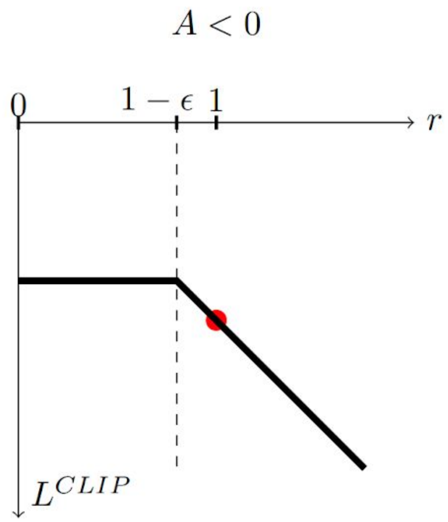
- Updates β_k between iterations
    - If $d < d_{targ}/1.5$, $\beta_k \leftarrow \beta_k / 2$
    - If $d > d_{targ} \times 1.5$, $\beta_k \leftarrow \beta_k \times 2$

PPO Clip objective

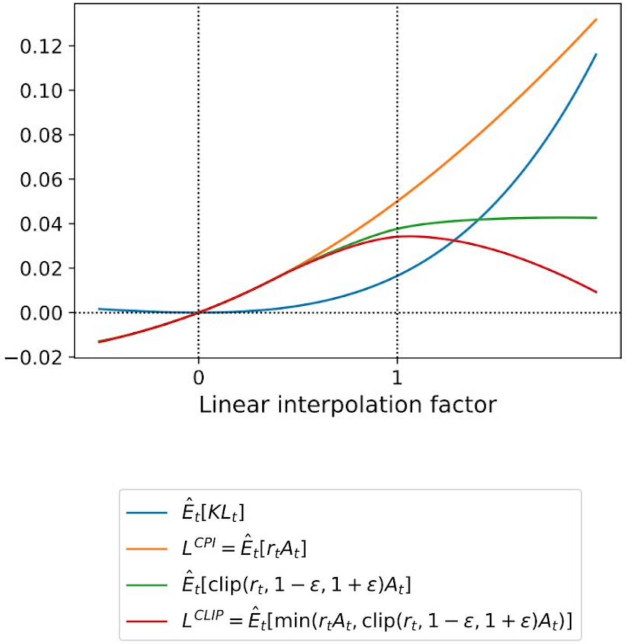
$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}$$

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





Surrogate objectives



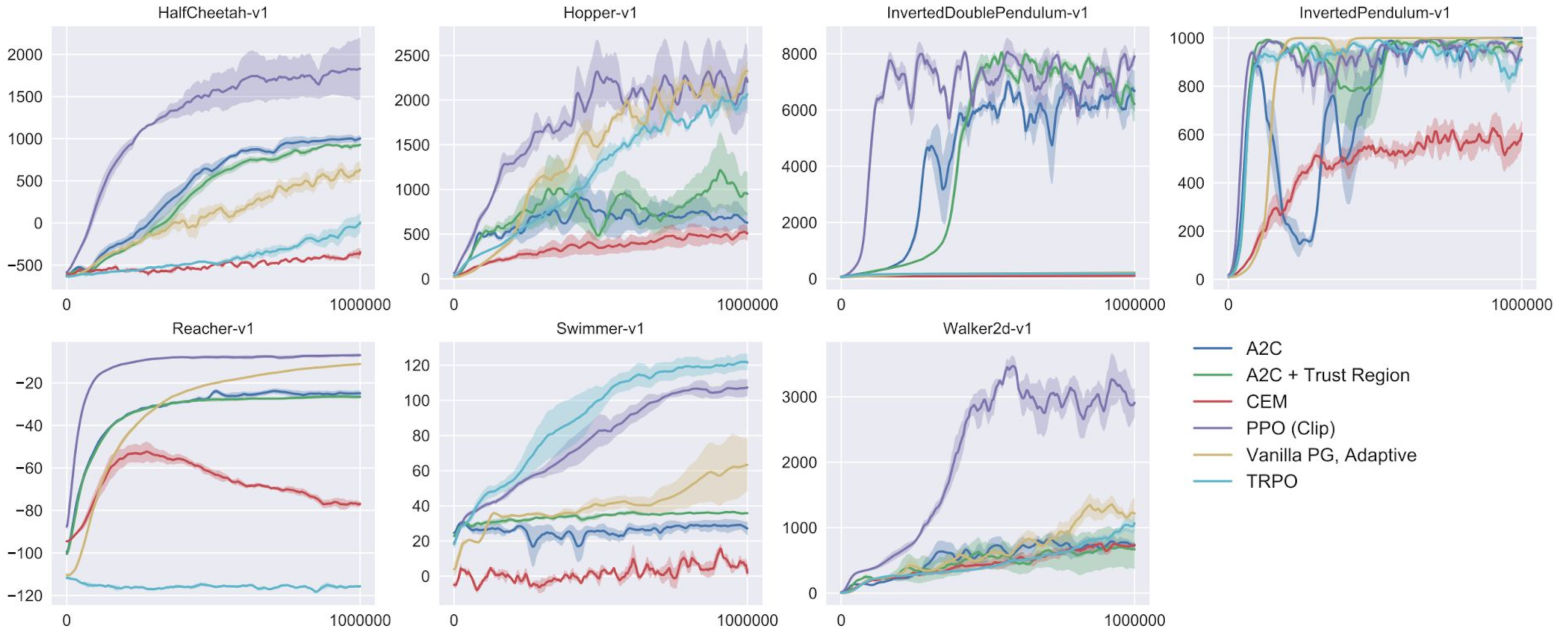
algorithm	avg. normalized score
No clipping or penalty	-0.39
Clipping, $\epsilon = 0.1$	0.76
Clipping, $\epsilon = 0.2$	**0.82**
Clipping, $\epsilon = 0.3$	0.70
Adaptive KL $d_{\text{targ}} = 0.003$	0.68
Adaptive KL $d_{\text{targ}} = 0.01$	0.74
Adaptive KL $d_{\text{targ}} = 0.03$	0.71
Fixed KL, $\beta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

PPO actor pseudocode

Sample efficient version of PPO

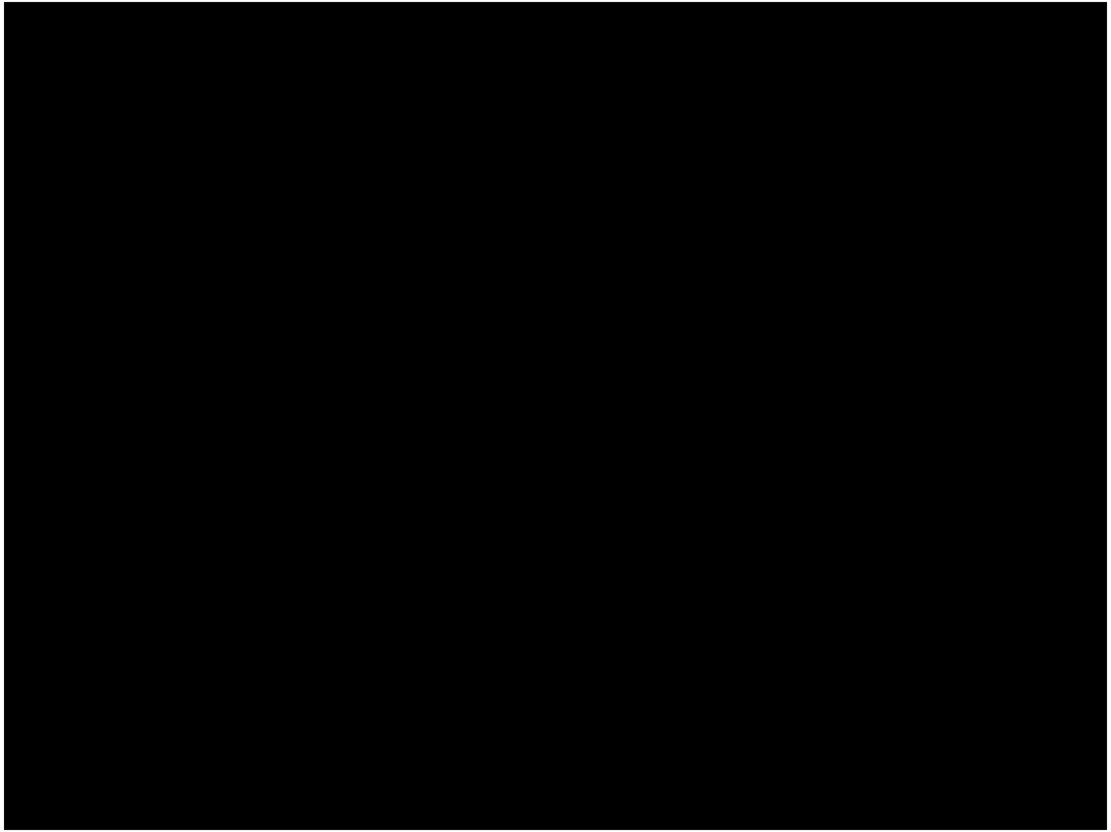
for iteration$=1, 2, \ldots$ **do**
  for actor$=1, 2, \ldots, N$ **do**
    Run policy $\pi_{\theta_{\text{old}}}$ in environment for T timesteps
    Compute advantage estimates $\hat{A}_1, \ldots, \hat{A}_T$
  end for
  Optimize surrogate L wrt θ, with K epochs and minibatch size $M \leq NT$
  $\theta_{\text{old}} \leftarrow \theta$
end for

Results



HalfCheetah-v1
Hopper-v1
InvertedDoublePendulum-v1
InvertedPendulum-v1
Reacher-v1
Swimmer-v1
Walker2d-v1
0
1000000
A2C
A2C + Trust Region
CEM
PPO (Clip)
Vanilla PG, Adaptive
TRPO

Results



Discussion

- Implementation matters

Implementation Matters in Deep RL: A Case Study on PPO and TRPO, https://openreview.net/forum?id=r1etN1rtPB

References

- Trust Region Policy Optimization (https://arxiv.org/pdf/1502.05477.pdf)
- Constrained Policy Optimization, Achiam et al. 2017 (https://arxiv.org/pdf/1705.10528.pdf)
- Proximal Policy Optimization Algorithms (https://arxiv.org/pdf/1707.06347.pdf)
- Spinning Up in Deep RL (https://spinningup.openai.com/en/latest/index.html)
- UC Berkeley Deep RL course (http://rail.eecs.berkeley.edu/deeprlcourse-fa17/f17docs/lecture_13_advanced_pg.pdf)
- Deep RL Bootcamp (https://sites.google.com/view/deep-rl-bootcamp/lectures)
- Deep RL Series (https://medium.com/@jonathan_hui/rl-trust-region-policy-optimization-trpo-part-2-f51e3b2e373a)
- https://towardsdatascience.com/the-pursuit-of-robotic-happiness-how-trpo-and-ppo-stabilize-policy-gradient-methods-545784094e3b

Questions?