COS435 / ECE433
Introduction to Reinforcement Learning

Instructors: Benjamin Eysenbach and Mengdi Wang

TAs: Yulai Zhao, Kurtland Chua, Zihao Li
UCAs: Alex Zhang, Ananya Parashar

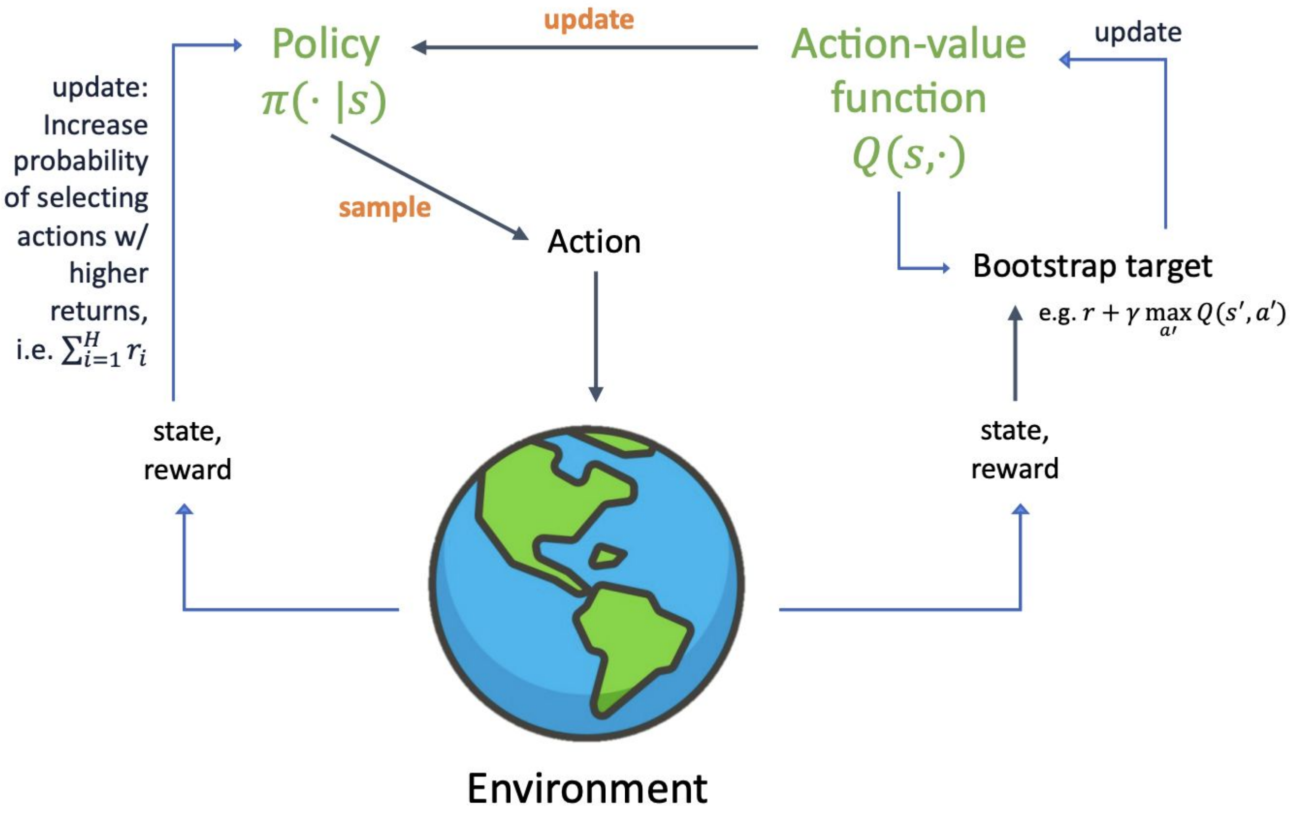
Lecture 19: Advanced Policy Gradient Methods

In previous lectures, we have seen that policy gradient methods, featured by actor-critic, have become the backbone of modern DRL.

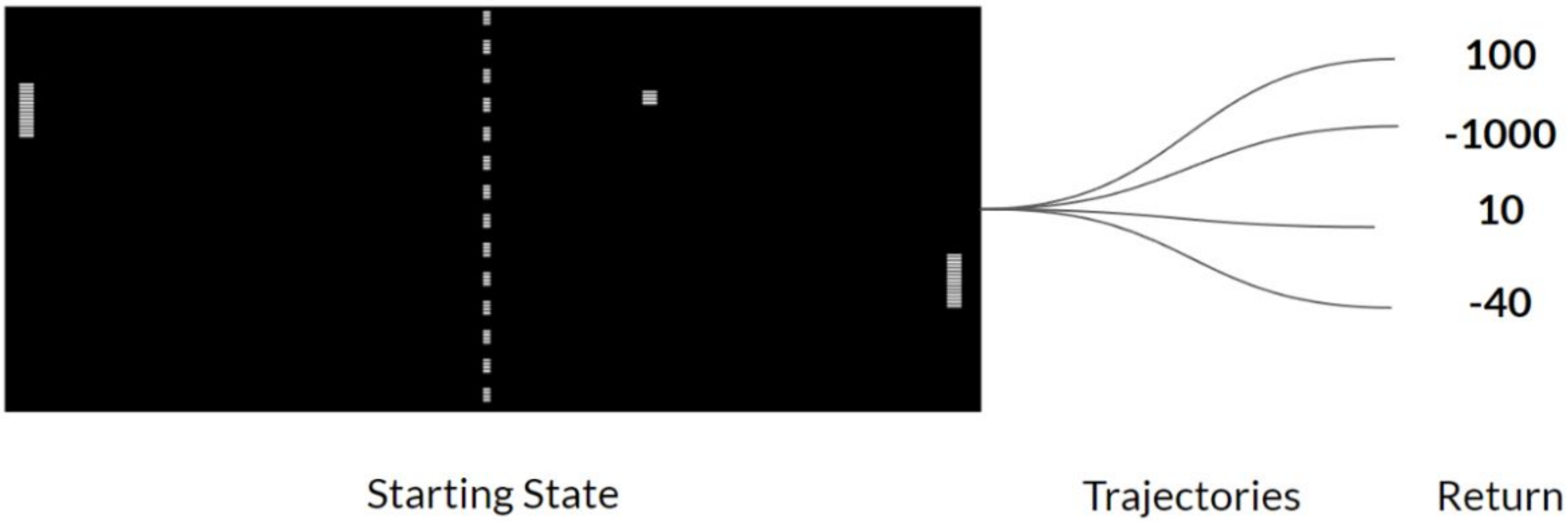
Today: **Can we improve actor-critic?**

- **Reduce Variance**: Advantage estimation and A2C
- **Leverage the Geometry**: Nature Policy Gradient and TRPO
- **Scale Up Further**: Proximal Policy Optimization

Actor-Critic



Policy Gradient Estimates have High Variances



Add a Baseline: Advantage Actor Critic (A2C)

$$A(s,a) = \underbrace{Q(s,a)}_{\text{q value for action a in state s}} - \underbrace{V(s)}_{\text{average value of that state}}$$

```python
# suppose everything have the correct type
# the term 'done' is important because for the end of the episode we only want
# the reward, without the discounted next state value.
advantage = reward + (1.0 - done) * gamma * critic(next_state) - critic(state)
```

Add a Baseline: Advantage Actor Critic (A2C)

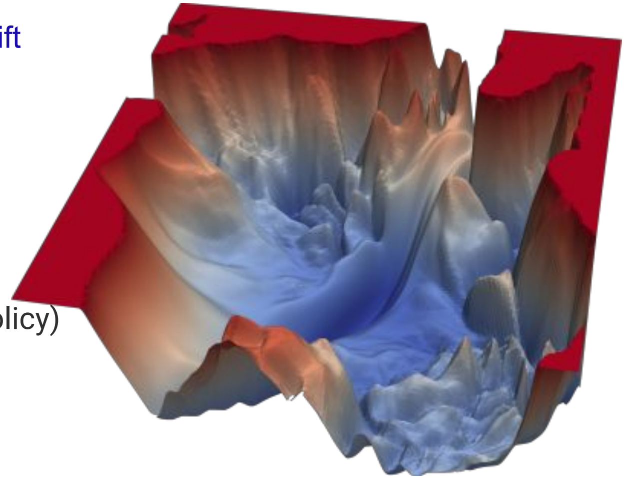
```
# it's just the MSE of the advantage
# the target value is: reward + critic(next_state)
# the predicted value is: critic(state)
critic_loss = advantage.pow(2)
```

```
# supposing the actions are categorical
probs = actor(state)
dist = torch.distributions.Categorical(probs=probs)
action = dist.sample()

# ...

actor_loss = -dist.log_prob(action) * advantage.detach()
```

Convergence theory for policy gradient

Policy optimization is **highly nonconvex**!

Surprisingly, policy gradient method enjoy global convergence:



On the theory of policy gradient methods: Optimality, approximation, and distribution shift
A Agarwal, SM Kakade, JD Lee, G Mahajan
Journal of Machine Learning Research, 2021 • jmlr.org (Unparameterized polic)

On the global convergence rates of softmax policy gradient methods
J Mei, C Xiao, C Szepesvari, D Schuurmans
International conference on machine learning, 2020 • proceedings.mlr.press (Softmax policy)

Variational policy gradient method for reinforcement learning with general utilities
J Zhang, A Koppel, AS Bedi, C Szepesvari, M Wang
Advances in Neural Information Processing Systems 2020 (General policy networks)

Policy optimization as a distribution optimization problem

Each policy maps to a distribution over state-action pairs:

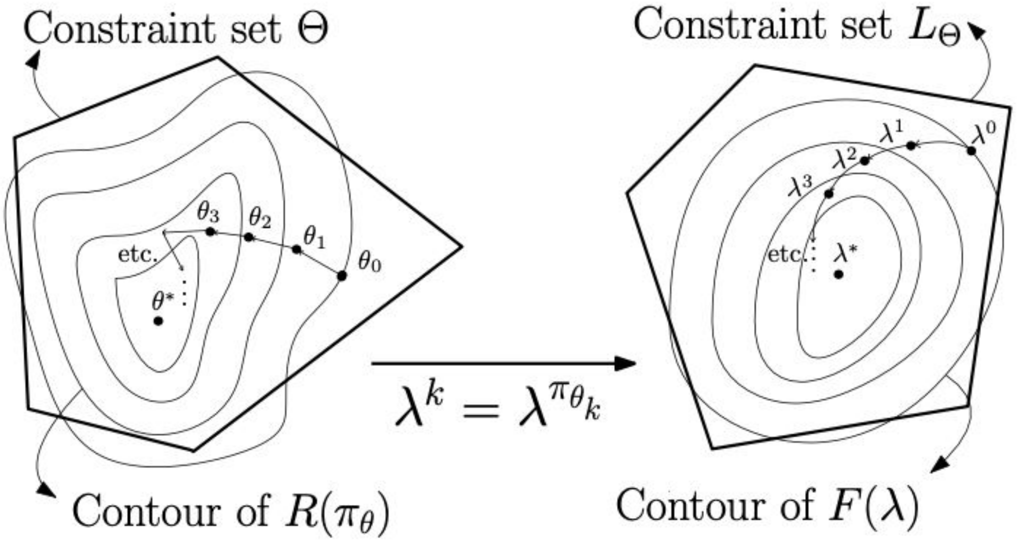
λ^π the unnormalized state-action occupancy measure.

$$\lambda^\pi_{sa} := \sum_{t=0}^{\infty} \gamma^t \cdot \mathbb{P}\left(s_t = s, a_t = a \,\middle|\, \pi, s_0 \sim \xi\right).$$

Policy optimization for standard MDP is a linear program in the distribution space

PG Convergence due to hidden convexity in the distribution space

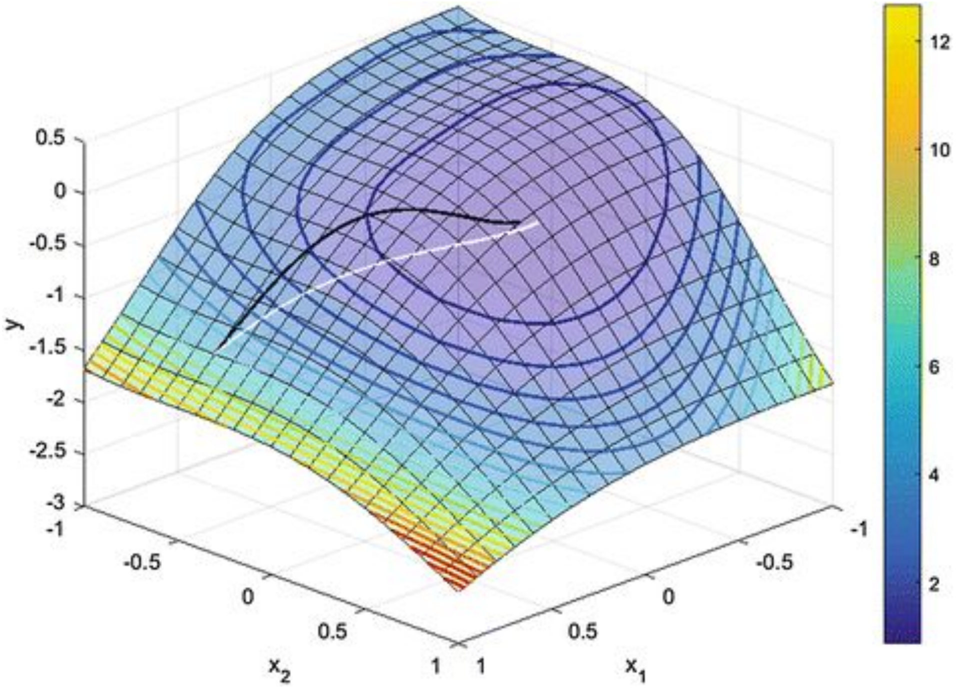
- Gradient flow in θ space $\Longleftrightarrow$ "gradient flow" in λ space.



Variational policy gradient method for reinforcement learning with general utilities
J Zhang, A Koppel, AS Bedi, C Szepesvari, M Wang
Advances in Neural Information Processing Systems 2020

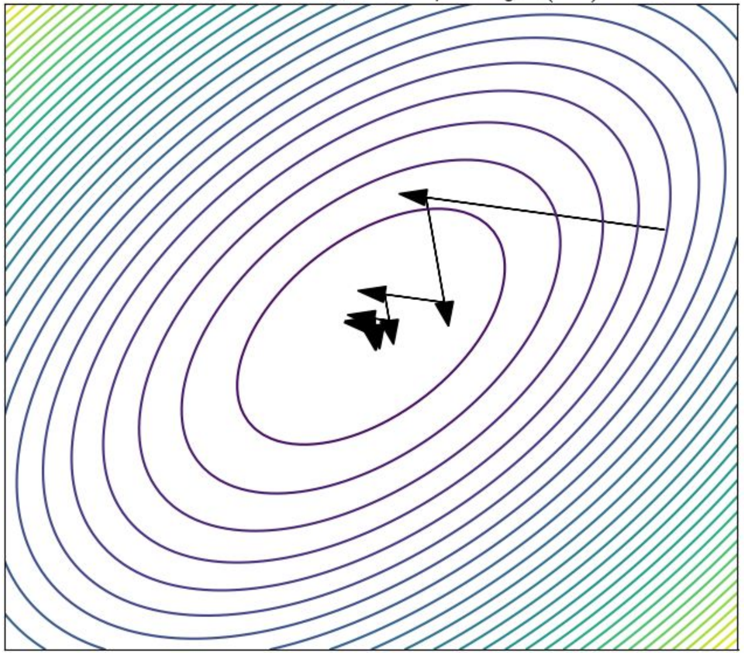
Can we leverage the geometry for better algorithms?

Gradient flow on Riemannian manifold can be faster than steepest ascent:

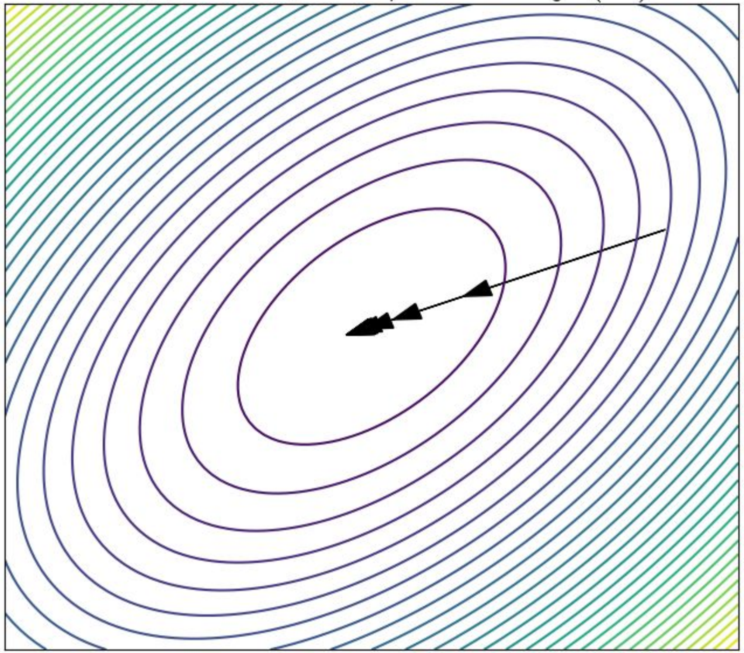


Steepest Descent vs Conditioned Gradient

$$x_{t+1} = x_t - \gamma \nabla f(x)$$



$$x_{t+1} = x_t - \gamma A^{-1} \nabla f(x)$$

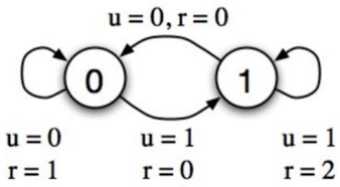


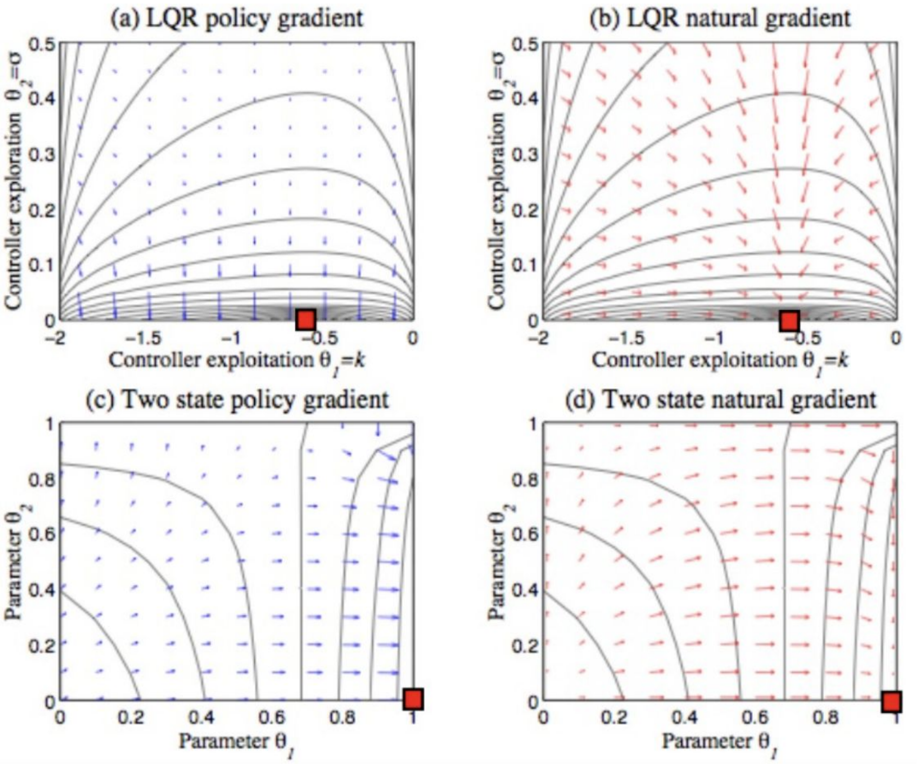
PG vs Natural PG

Linear Quadratic Regulation

$$x_{t+1} = Ax_t + Bu_t$$
$$u_t \sim \pi(u|x_t) = \mathcal{N}(u|kx_t, \sigma)$$
$$r_t = -x_t^T Q x_t - u_t^T R u_t$$

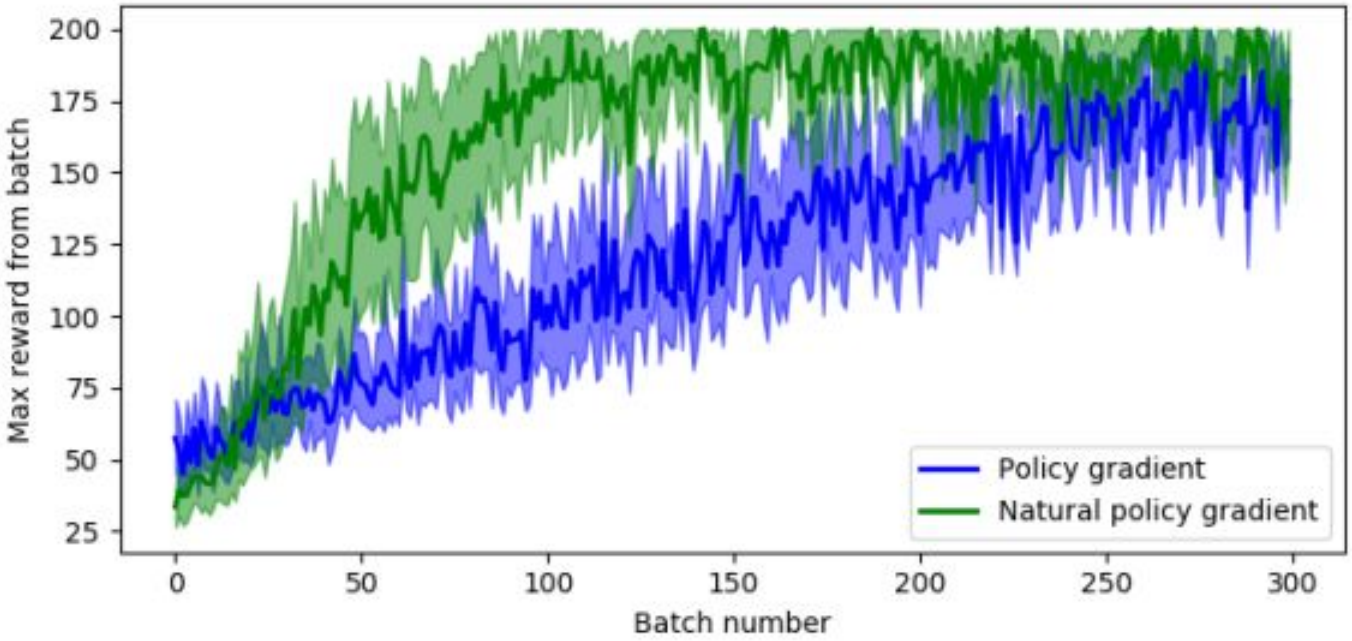
Two-State Problem



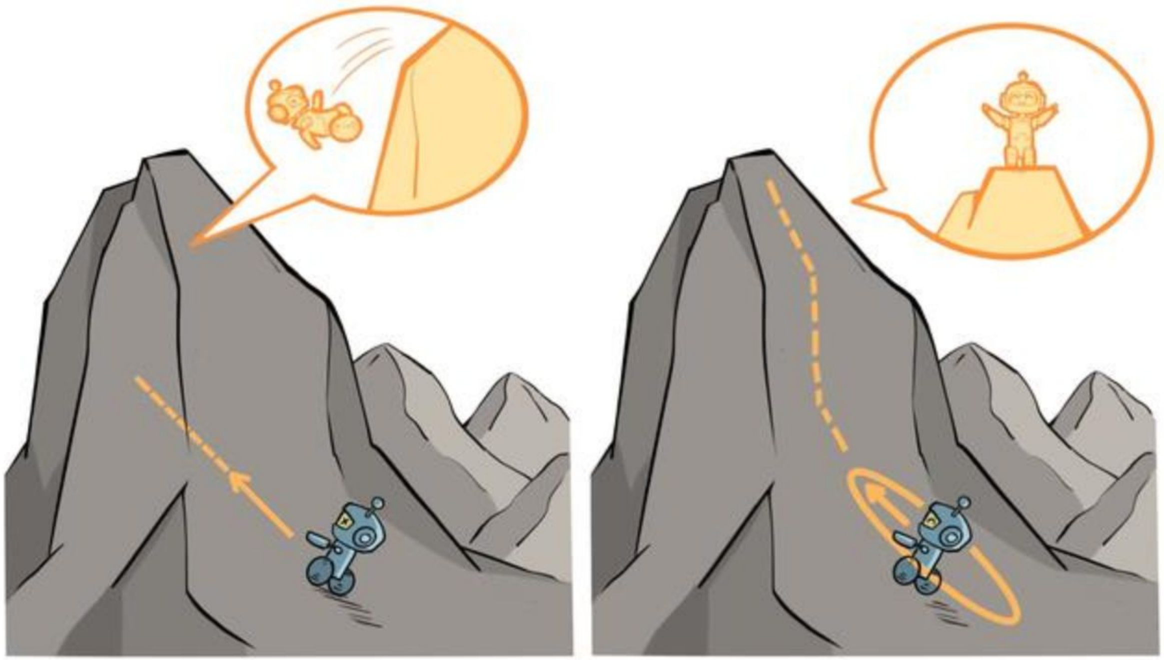


[Peters et al. 2003, 2005]

PG vs Natural PG



Trust-Region Policy Optimization (TRPO)



Trust-Region Policy Optimization (TRPO)

Surrogate loss: $\max_{\pi} L(\pi) = \mathbb{E}_{\pi_{\text{old}}}\left[\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)} A^{\pi_{\text{old}}}(s,a)\right]$

Constraint: $\mathbb{E}_{\pi_{\text{old}}}\left[KL(\pi||\pi_{\text{old}})\right] \leq \epsilon$

for iteration$=1, 2, \ldots$ **do**
  Run policy for T timesteps or N trajectories
  Estimate advantage function at all timesteps
  Compute policy gradient g
  Use CG (with Hessian-vector products) to compute $F^{-1}g$
  Do line search on surrogate loss and KL constraint
end for

But *how* does clipping keep policy close? By making objective as pessimistic as possible about performance far away from θ_k:

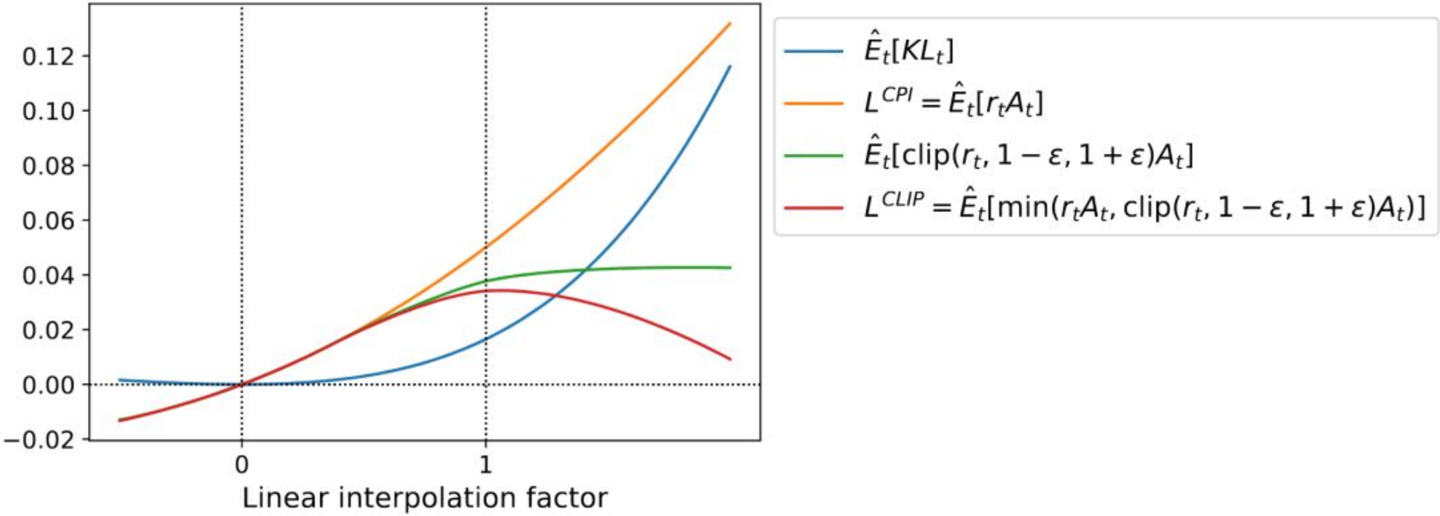
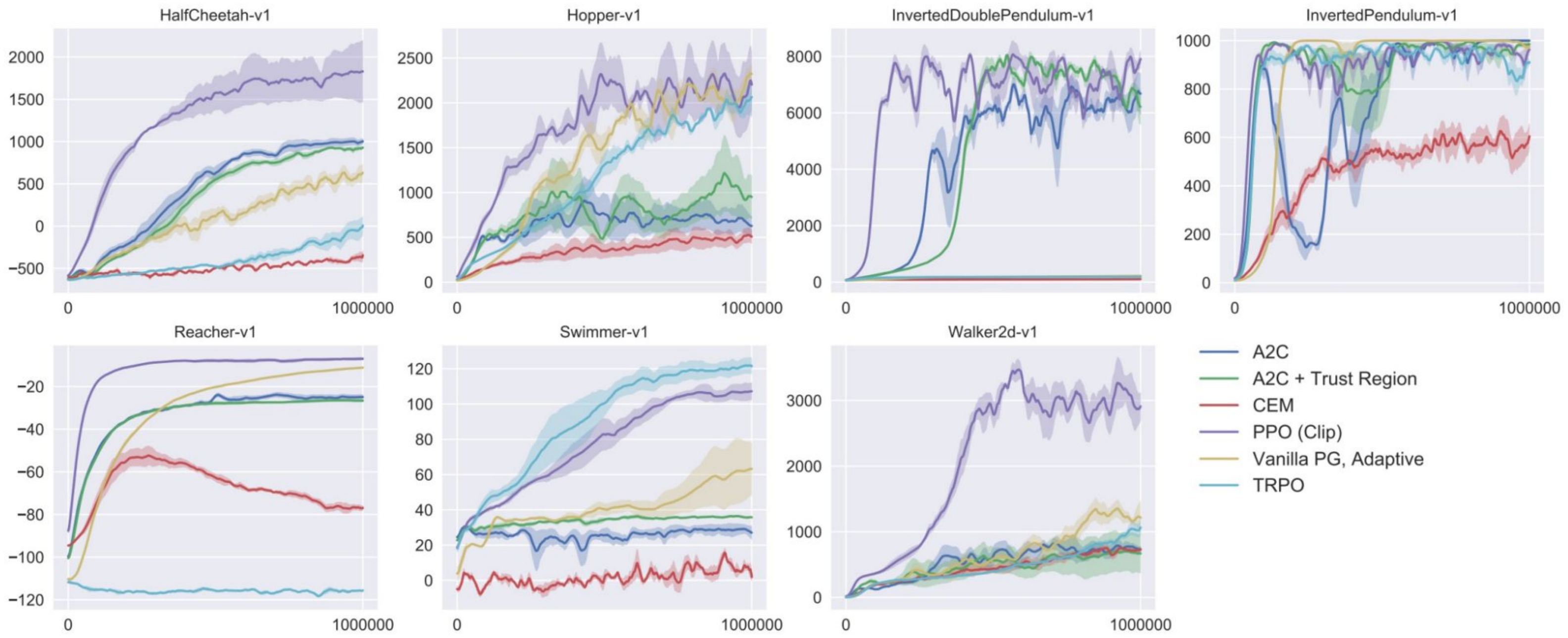


Figure: Various objectives as a function of interpolation factor α between θ_{k+1} and θ_k after one update of PPO-Clip [9]

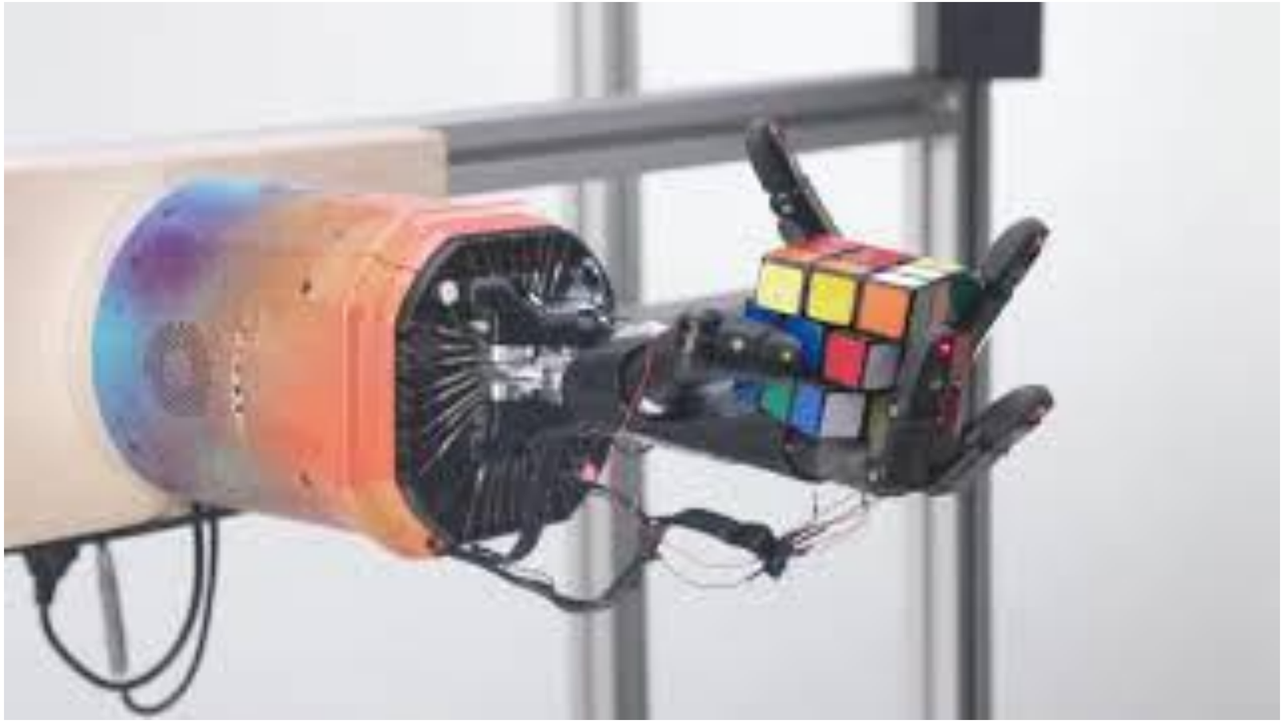
Implementation matters. What really makes PPO work?

See careful empirical studies:

- Engstrom, Logan, et al. "Implementation matters in deep policy gradients: A case study on PPO and TRPO." *arXiv preprint arXiv:2005.12729* (2020).
- Andrychowicz, Marcin, et al. "What matters in on-policy reinforcement learning? a large-scale empirical study." *arXiv preprint arXiv:2006.05990* (2020).



OpenAI solves Rubik's cube using PPO



Summary

Policy gradient methods are an and powerful class of RL methods.

Yes we can improve actor critic:

- **Reduce Variance:** Advantage estimation and A2C
- **Leverage the Geometry:** Nature Policy Gradient and TRPO
- **Scale Up Further:** Proximal Policy Optimization

Theory:

- Global convergence in the distribution space