NPG and PPO

Annoucements

1. We will release HW3 w/ solution — it is optional, but do take a look

2. Prelim scope: first lecture to (and include) next Monday’s lecture

3. We will release a prelim from last year (but don’t overfit to it)

Recap on NPG:

At iteration t:

$$\max_{\pi_\theta} \mathbb{E}_{s\sim d_\mu^{\pi_{\theta_t}}}\left[\mathbb{E}_{a\sim\pi_\theta(s)} A^{\pi_{\theta_t}}(s,a)\right]$$ → First-order Taylor expansion at θ_t

s.t., $KL\left(\rho_{\pi_{\theta_t}} | \rho_{\pi_\theta}\right) \leq \delta$ → second-order Taylor expansion at θ_t

Intuition: maximize local adv subject to being incremental (in KL);

$$\max_{\theta} \nabla_\theta J(\pi_{\theta_t})^\top (\theta - \theta_t)$$

s.t. $(\theta - \theta_t)^\top F_{\theta_t} (\theta - \theta_t) \leq \delta$

$$F_{\theta_t} := \mathbb{E}_{s,a\sim d_\mu^{\pi_{\theta_t}}}\left[\nabla_\theta \ln \pi_{\theta_t}(a|s)\left(\nabla_\theta \ln \pi_{\theta_t}(a|s)\right)^\top\right] \in \mathbb{R}^{dim_\theta \times dim_\theta}$$

←

$$\theta_{t+1} = \theta_t + \eta F_{\theta_t}^{-1} \nabla_\theta J(\pi_{\theta_t})$$

NPG

Outline for Today:

1. More Explanation of Natural (Policy) Gradient

2. Proximal Policy Optimization (PPO)

NPG update: $\theta_1 = \theta_0 + \eta F_{\theta_0}^{-1} \nabla_{\theta_0}$

$$KL\left(\rho_{\pi_{\theta_0}} | \rho_{\pi_\theta}\right) \leq \delta \Rightarrow (\theta - \theta_0)^\top F_{\theta_0}(\theta - \theta_0) \leq \delta$$

Our goal is to make sure two distributions do not change to much,
but parameters θ could potential change a lot!

Consider special case where F_{θ_0} is a diagonal matrix: $F_{\theta_0} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix}$

$$\forall i : \ \theta_1[i] = \theta_0[i] + \left(\eta \sigma_i^{-1}\right) \nabla_{\theta_0}[i]$$

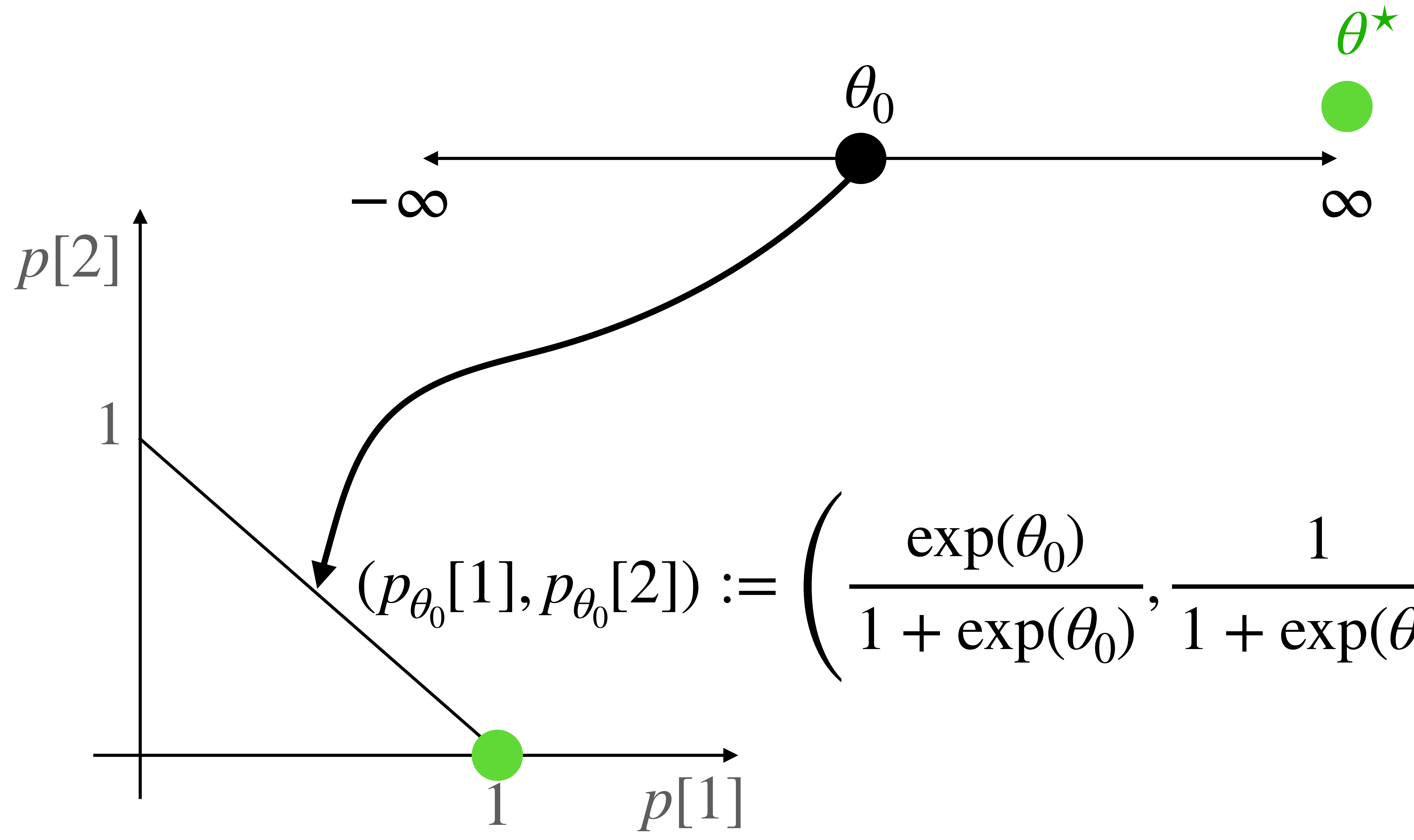
For tiny σ_i, we indeed have a **huge** learning rate, i.e., $\eta \sigma_i^{-1}$, at coordinate i !

In other words, NPG **allows a big jump** on some coordinates which do not affect KL-div too much

Example of Natural Gradient on 1-d problem:

$$p_\theta = \left(\frac{\exp(\theta)}{1+\exp(\theta)}, \frac{1}{1+\exp(\theta)} \right)$$

$$g(\theta) = 100 \cdot p_\theta[1] + 1 \cdot p_\theta[2]$$



$$(p_{\theta_0}[1], p_{\theta_0}[2]) := \left(\frac{\exp(\theta_0)}{1+\exp(\theta_0)}, \frac{1}{1+\exp(\theta_0)} \right)$$

Fisher information scalar: $f_{\theta_0} = \dfrac{\exp(\theta_0)}{(1+\exp(\theta_0))^2}$

Hence: $f_{\theta_0} \to 0^+$, as $\theta_0 \to \infty$

NPG: $\theta_1 = \theta_0 + \eta \dfrac{g'(\theta_0)}{f_{\theta_0}}$

GA: $\theta_1 = \theta_0 + \eta g'(\theta_0)$

i.e., Natural GA can speed up learning when θ gets larger

Outline for Today:

1. More Explanation of Natural (Policy) Gradient

2. Proximal Policy Optimization (PPO)

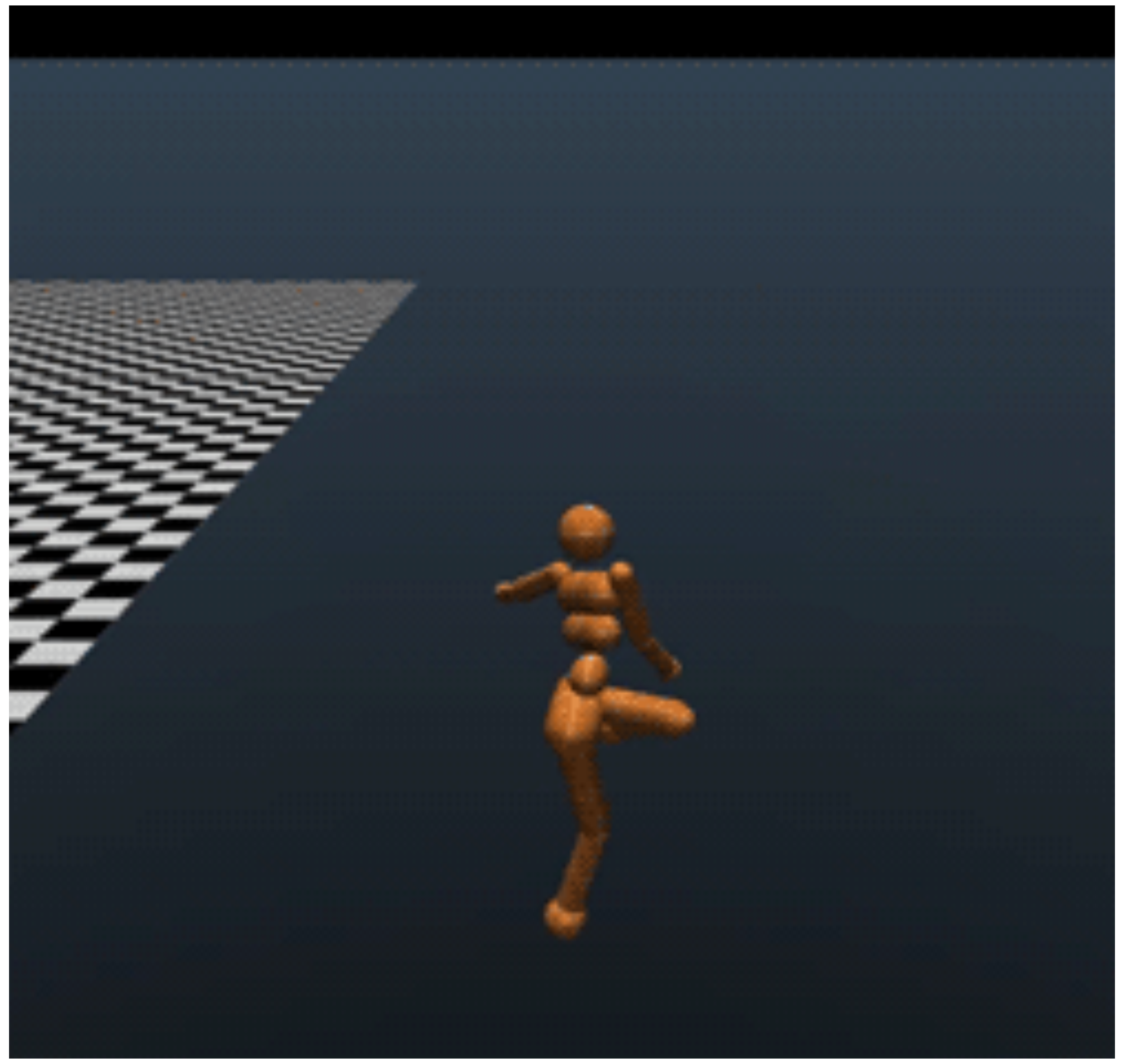
Policy Gradient (e.g., REINFORCE) can unstable and slow

The potential high-variance in PG can make learning very unstable

Natural Policy gradient is computational expensive

Even compute fisher information matrix is slow

These methods do not take advantage of GPUs well

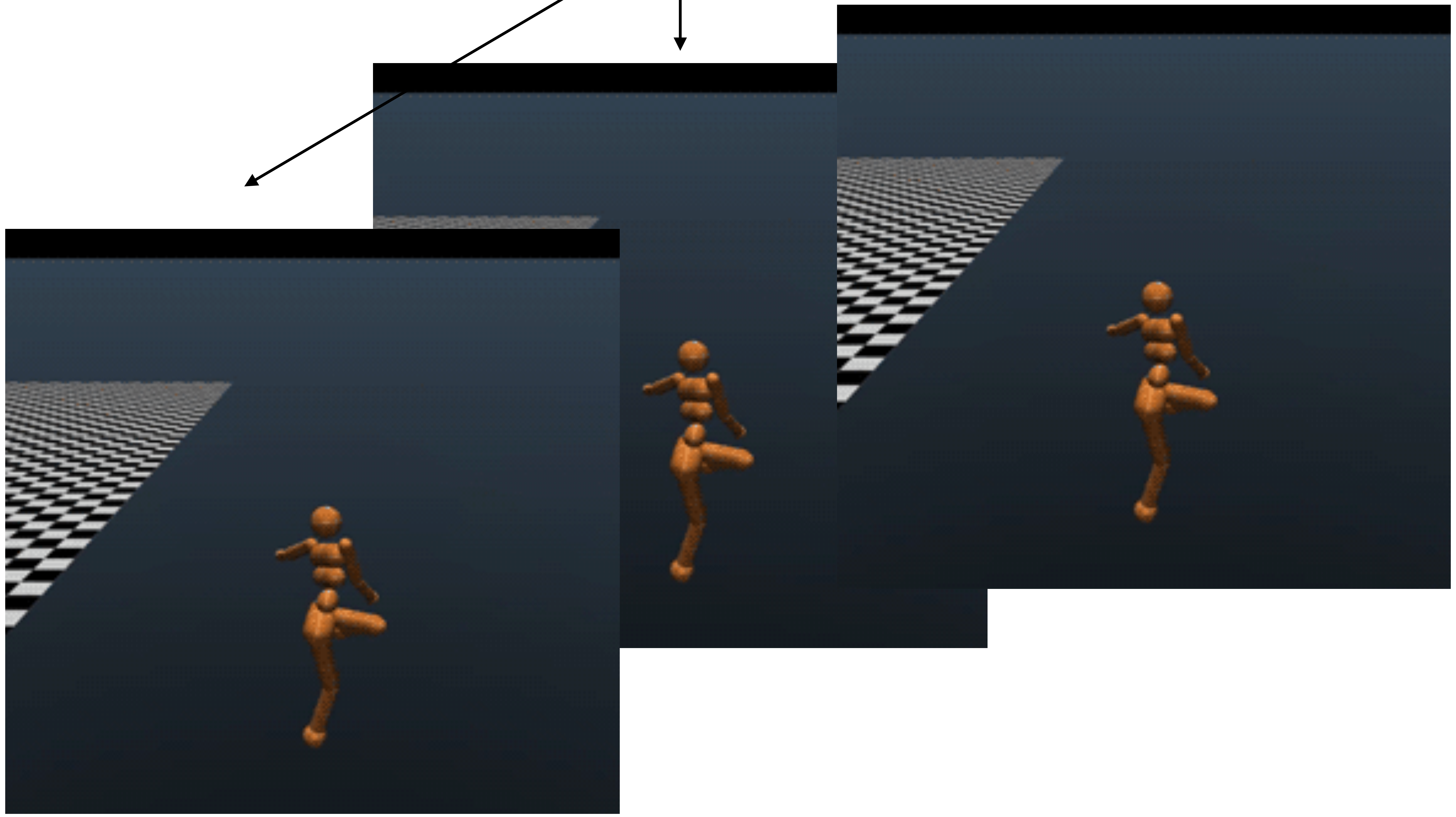


Too frequent!

GPU usage can be very low…

Proximal Policy Optimization (PPO)

π_{θ_t}



Collect a large dataset

$$\mathscr{D} = \left\{ s, a, A^{\pi_{\theta_t}}(s, a) \right\}$$

Now let’s do multiple epoches of mini-batch gradient update on the dataset

Proximal Policy Optimization (PPO)

Construct a **batch Supervised Learning** style objective using $\mathscr{D} = \{s, a, A^{\pi_{\theta_t}}(s, a)\}$

$$\max_\theta \ell(\theta) = \max_\theta \mathbb{E}_{s \sim d^{\pi_{\theta_t}}} \mathbb{E}_{a \sim \pi_\theta(\cdot|s)} \cdot A^{\pi_{\theta_t}}(s, a)$$

IW trick $\rightarrow \mathbb{E}_{s \sim d^{\pi_{\theta_t}}} \mathbb{E}_{a \sim \pi_{\theta_t}(\cdot|s)} \dfrac{\pi_\theta(a \mid s)}{\pi_{\theta_t}(a \mid s)} \cdot A^{\pi_{\theta_t}}(s, a)$

$$\approx \sum_{s,a} \frac{\pi_\theta(a \mid s)}{\pi_{\theta_t}(a \mid s)} \cdot A^{\pi_{\theta_t}}(s, a)$$

Proximal Policy Optimization (PPO)

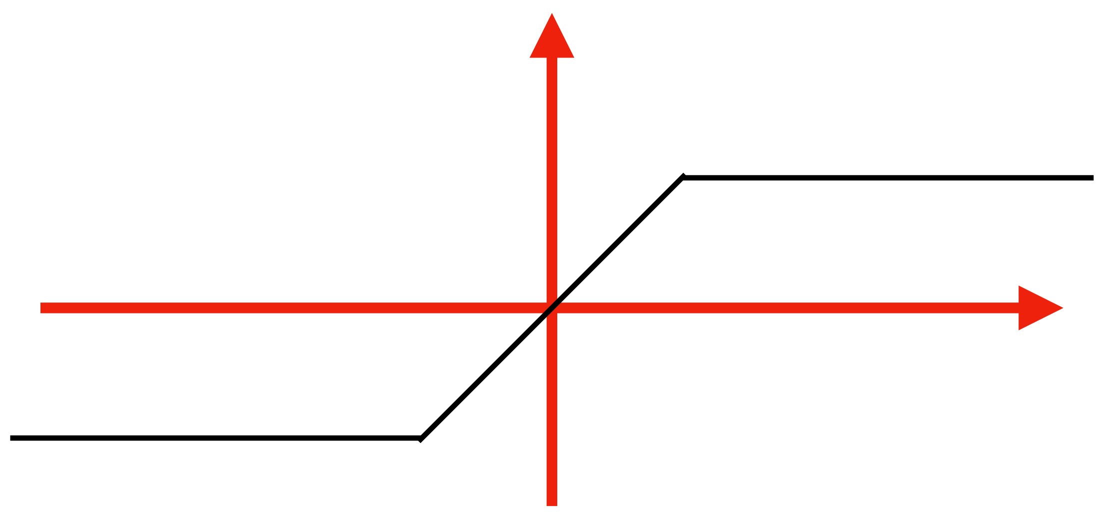
Construct a **batch Supervised Learning** style objective using $\mathcal{D} = \{s, a, A^{\pi_{\theta_t}}(s, a)\}$

$$\hat{\ell}(\theta) = \sum_{s,a} \frac{\pi_\theta(a \mid s)}{\pi_{\theta_t}(a \mid s)} \cdot A^{\pi_{\theta_t}}(s, a)$$

Trick 1: clipping to make sure π_θ stay close to π_{θ_t} (ensuring stability in training)

$$\hat{\ell}_{clip}(\theta) = \sum_{s,a} \text{clip}\left(\frac{\pi_\theta(a \mid s)}{\pi_{\theta_t}(a \mid s)}, 1 - \epsilon, 1 + \epsilon \right) \cdot A^{\pi_{\theta_t}}(s, a)$$

Stop updating $\pi_\theta(a \mid s)$ if it is too different from $\pi_{\theta_t}(a \mid s)$



$\text{clip}(x, 1 - \epsilon, 1 + \epsilon)$

Proximal Policy Optimization (PPO)

Trick 2, take the min of the clipped and uncipped (original) obj

$$\hat{\ell}_{final}(\theta) = \sum_{s,a} \min \left\{ \frac{\pi_\theta(a\,|\,s)}{\pi_{\theta_t}(a\,|\,s)} \cdot A^{\pi_{\theta_t}}(s,a), \quad \text{clip}\left(\frac{\pi_\theta(a\,|\,s)}{\pi_{\theta_t}(a\,|\,s)}, 1-\epsilon, 1+\epsilon \right) \cdot A^{\pi_{\theta_t}}(s,a) \right\}$$

Original obj

clipped obj which ensures no abrupt change in action probabilities

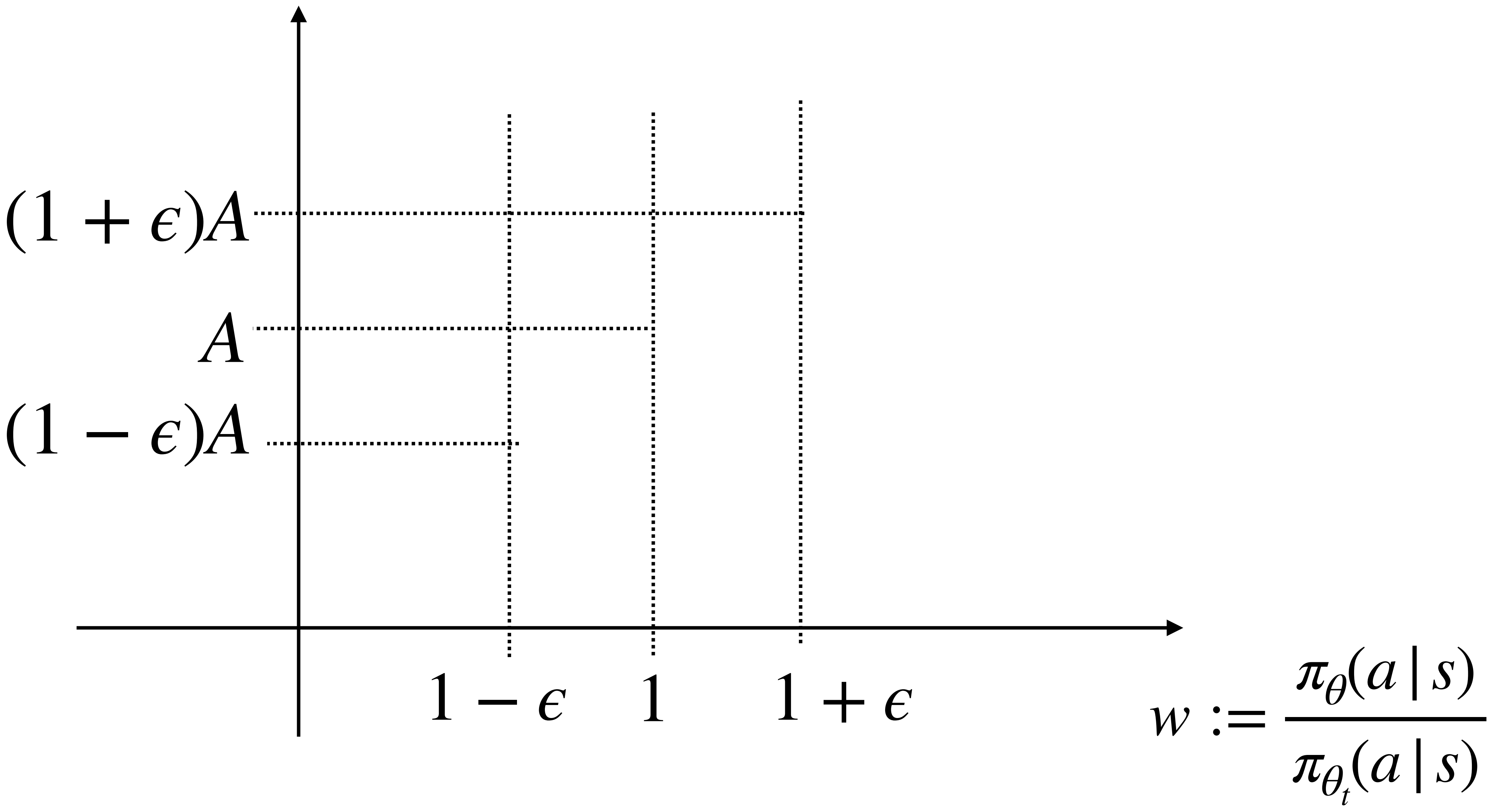
Proximal Policy Optimization (PPO)

Trick 2, take the min of the clipped and uncipped (original) obj

$$\hat{\ell}_{final}(\theta) = \sum_{s,a} \min \left\{ \frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)} \cdot A^{\pi_{\theta_t}}(s,a), \quad \text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)}, 1-\epsilon, 1+\epsilon \right) \cdot A^{\pi_{\theta_t}}(s,a) \right\}$$

Just consider one term inside the summation:

When $A^{\pi_{\theta_t}}(s,a) > 0$



Proximal Policy Optimization (PPO)

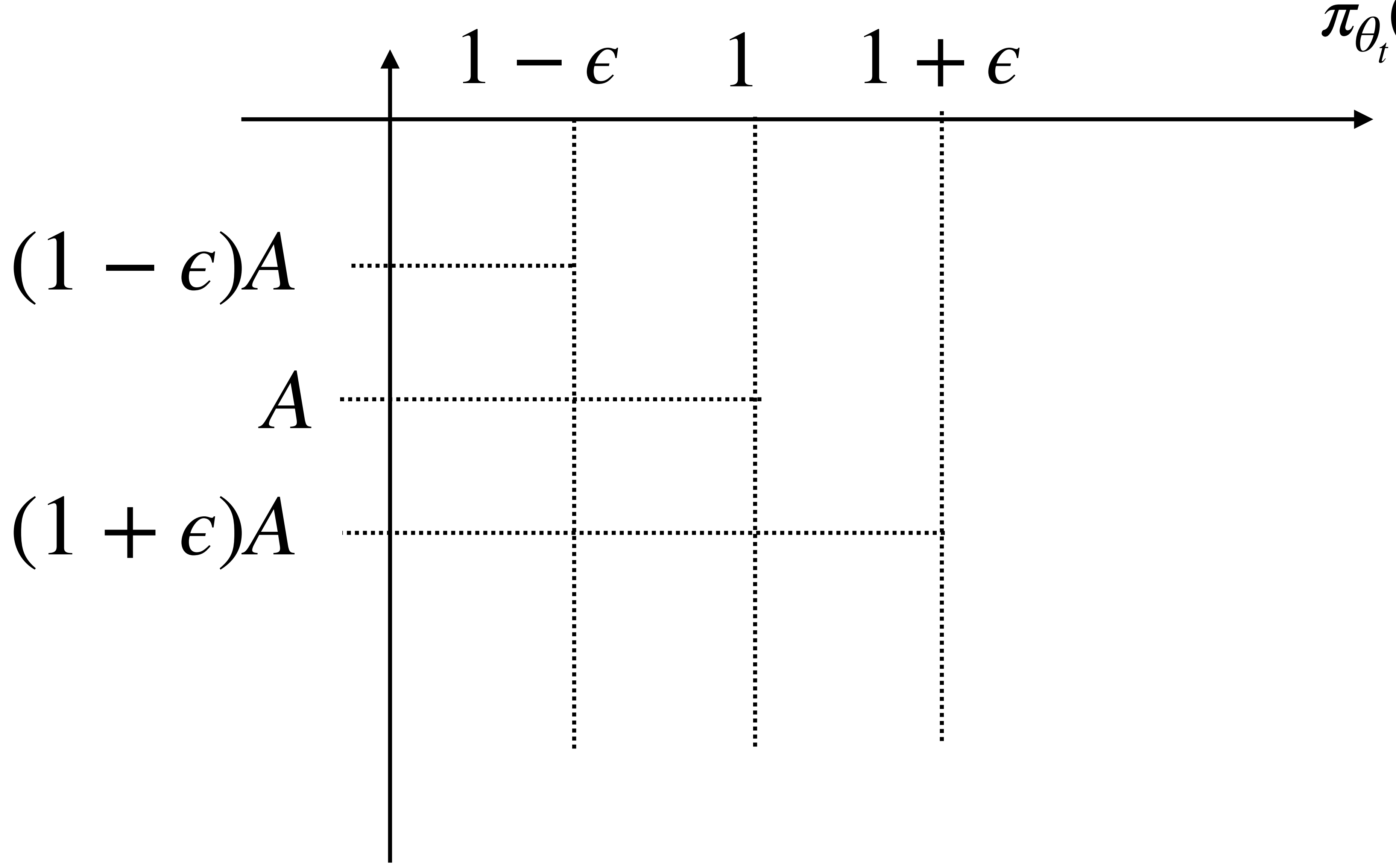
Trick 2, take the min of the clipped and uncipped (original) obj

$$\hat{\ell}_{final}(\theta) = \sum_{s,a} \min \left\{ \frac{\pi_\theta(a\,|\,s)}{\pi_{\theta_t}(a\,|\,s)} \cdot A^{\pi_{\theta_t}}(s,a), \quad \text{clip}\left(\frac{\pi_\theta(a\,|\,s)}{\pi_{\theta_t}(a\,|\,s)}, 1-\epsilon, 1+\epsilon \right) \cdot A^{\pi_{\theta_t}}(s,a) \right\}$$

Just consider one term inside the summation:

When $A^{\pi_{\theta_t}}(s,a) < 0$

$$w := \frac{\pi_\theta(a\,|\,s)}{\pi_{\theta_t}(a\,|\,s)}$$



Proximal Policy Optimization (PPO)

Trick 2, take the min of the clipped and uncipped (original) obj

$$\hat{\ell}_{final}(\theta) = \sum_{s,a} \min \left\{ \frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)} \cdot A^{\pi_{\theta_t}}(s,a), \quad \text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)}, 1-\epsilon, 1+\epsilon \right) \cdot A^{\pi_{\theta_t}}(s,a) \right\}$$

Original obj

clipped obj which ensures no abrupt change in action probabilities

We compute $\theta_{t+1} \approx \arg\max_\theta \hat{\ell}_{final}(\theta)$, via performing a few epoches of minbatch SG ascent (or Adam/Adagrad) on $\hat{\ell}_{final}$

Proximal Policy Optimization (PPO)

Initialize θ_0 for the policy

For $t = 0 \rightarrow T$:

Run π_θ to collect multiple trajectories, and form the dataset $\{s, a, A^{\pi_{\theta_t}}(s, a)\}$

Construct the loss $\hat{\ell}_{final}(\theta)$ using the dataset

Perform a few steps of mini-batch gradient updates on $\hat{\ell}_{final}(\theta)$ to get θ_{t+1}

Summary

NPG controls the changes in the policy space (KL) directly

NPG allows one to have big jumps in parameter space, as long as the outcome (distribution) does not change too much

PPO is a more practical versions of NPG — making NPG really scalable while maintaing the high level idea of NPG