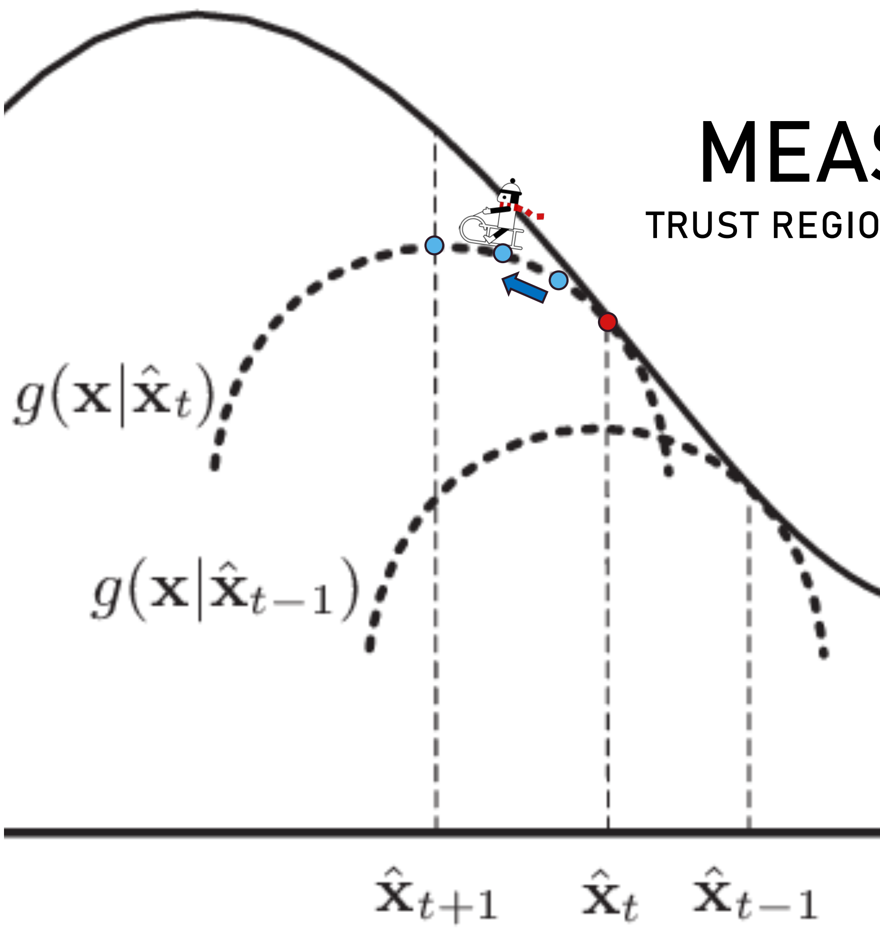


MEASURE ONCE, OPTIMIZE TWICE

TRUST REGIONS & STEP-SIZE RESTRICTIONS IN POLICY GRADIENT METHODS

William L. Ruys

CS395: Foundations of Machine Learning for Systems Researchers

POLICY GRADIENT METHODS

- **Reinforcement learning (RL) seeks policy π_θ to maximize the expected return:**

Expected Return (under policy π)

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[R(\tau)\right]$$

$$\tau = (s_0, a_0, s_1, a_1, \ldots)$$

Sampled Trajectory (following π)

Return of Trajectory

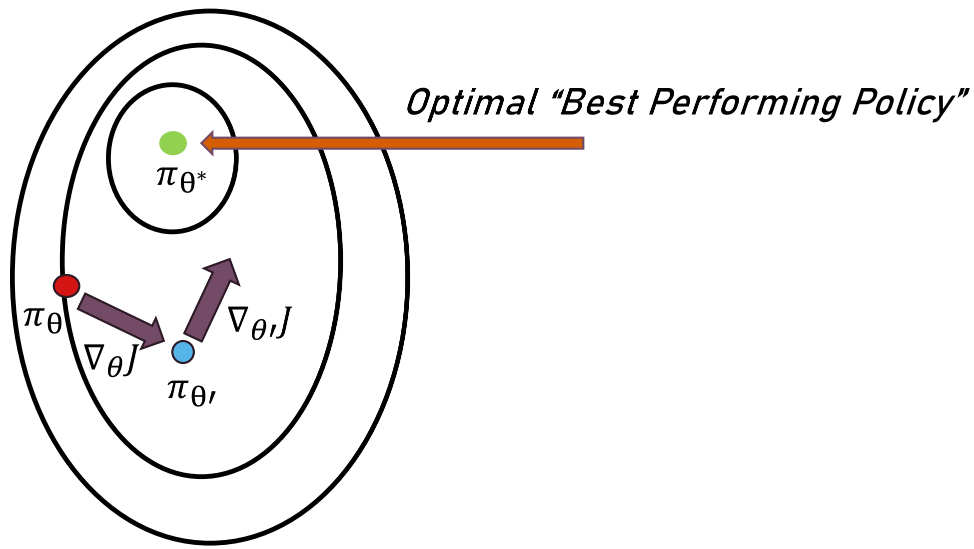
$$R(\tau) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$$

Discounted per-transition reward

$$(s_t, a_t) \in \tau$$

POLICY GRADIENT METHODS

- **Simplest solution: Follow the gradient!**



2D representation of "policy space"

HOW DO WE LOOK AT J?

- **Expected return can be factored:**

Expected Return (under policy π)

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[R(\tau)\right]$$

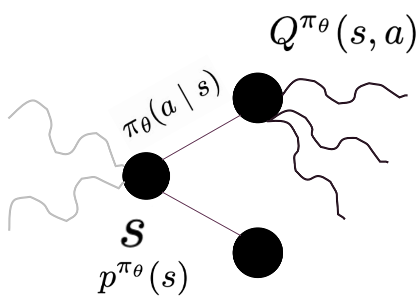
Expected Value of Action (downstream following π_θ)

$$Q^{\pi_\theta}(s,a) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a\right]$$

$$J(\theta) = \sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s,a)$$

State Visitation

Value of Action



HOW DO WE LOOK AT J?

- **Expected return can be factored:**

Expected Return (under policy π)

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[R(\tau)\right]$$

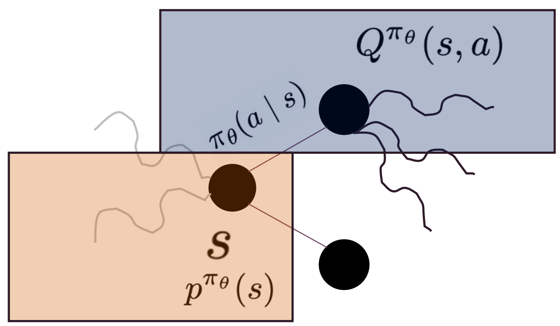
Expected Value of Action (downstream following π_θ)

$$Q^{\pi_\theta}(s,a) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a\right]$$

$$J(\theta) = \sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s,a)$$

State Visitation

Value of Action



Expected remaining value in trajectory given action

Probability of being in state s

WHAT IS $\nabla_\theta J(\theta)$?

$$J(\theta) = \sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a)$$

WHAT IS $\nabla_\theta J(\theta)$?

$$J(\theta) = \sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a)$$

$$\nabla_\theta J(\theta) = \nabla_\theta \left[\sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a) \right]$$

WHAT IS $\nabla_\theta J(\theta)$?

$$J(\theta) = \sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a)$$

$$\nabla_\theta J(\theta) = \nabla_\theta \left[\sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a) \right]$$

$$\nabla_\theta J(\theta) = \sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \nabla_\theta \left[\pi_\theta(a \mid s) \right] Q^{\pi_\theta}(s, a)$$

Policy Gradient Theorem: $\nabla_\theta J(\theta)$ ***does not depend on gradient of state-visitations!***

WHAT IS $\nabla_\theta J(\theta)$?

$$J(\theta) = \sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a)$$

$$\nabla_\theta J(\theta) = \nabla_\theta \left[\sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a) \right]$$

$$\nabla_\theta J(\theta) = \sum_{s \in \mathcal{S}} p^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \nabla_\theta \left[\pi_\theta(a \mid s) \right] Q^{\pi_\theta}(s, a)$$

Policy Gradient Theorem: $\nabla_\theta J(\theta)$ ***does not depend on gradient of state-visitations!***

$$\nabla_\theta J(\theta) \propto \mathbb{E}_{a \sim \pi_\theta} \nabla_\theta \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a)$$

WHAT IS $\nabla_\theta J(\theta)$?

Policy Gradient Theorem: $\nabla_\theta J(\theta)$ *does not depend on gradient of state-visitations*

$$\nabla_\theta J(\theta) \propto \boxed{\mathbb{E}_{a\sim\pi_\theta} \nabla_\theta \pi_\theta(a \mid s)}\boxed{Q^{\pi_\theta}(s,a)}$$

This is an *on-policy estimator:*

- **Actions taken by following** π_θ
- **Value estimated assuming** π_θ
- **Computes gradient at** θ

WHAT IS $\nabla_\theta J(\theta)$?

Policy Gradient Theorem: $\nabla_\theta J(\theta)$ *does not depend on gradient of state-visitations*

$$\nabla_\theta J(\theta) \propto \boxed{\mathbb{E}_{a\sim\pi_\theta} \nabla_\theta \pi_\theta(a \mid s)}\boxed{Q^{\pi_\theta}(s,a)}$$

This is an *on-policy estimator:*

- Actions taken by following π_θ
- Value estimated assuming π_θ
- Computes gradient at θ

High Level Idea:

- What happens when we break these assumptions?
- Why would we want to?
- How do we solve the new problem?

WHAT IS $\nabla_\theta J(\theta)$?

Policy Gradient Theorem: $\nabla_\theta J(\theta)$ *does not depend on gradient of state-visitations*

$$\nabla_\theta J(\theta) \propto \boxed{\mathbb{E}_{a\sim\pi_\theta} \nabla_\theta \pi_\theta(a \mid s)}\boxed{Q^{\pi_\theta}(s,a)}$$

This is an *on-policy estimator*:

- Actions taken by following π_θ
- Value estimated assuming π_θ
- Computes gradient at θ

High Level Idea:

- What happens when we break these assumptions?
- Why would we want to?
- How do we solve the new problem?

$p^{\pi_\theta}(s)$ starts to matter!

$$J(\theta) = \boxed{\sum_{s\in\mathcal{S}} p^{\pi_\theta}(s)} \sum_{a\in\mathcal{A}} \pi_\theta(a \mid s) Q^{\pi_\theta}(s,a)$$

WHY WOULD WE WANT TO BREAK THESE?

Motivation: Make on-policy RL *slightly* more like supervised learning

POLICY GRADIENT - REINFORCEMENT LEARNING

- Dataset collected inside the loop
- Each collected sample is used ***once***
- ***All current samples*** compute the loss

Algorithm 1: Advantage Actor-Critic (A2C) Training Loop

Input: Initial policy parameters θ, value parameters ϕ

while *not converged* **do**

$\boxed{\text{Collect } N \text{ trajectories } \{\tau_i : \{(s_t, a_t)\}_{t=0}^{m_i}\}_{i=1}^{N} \text{ using current policy } \pi_\theta;}$

Estimate advantages: $A_t^\tau = R_t^\tau - V^\phi(s_t)$;

`// Update policy (actor)`

$\boxed{\theta \leftarrow \theta + \alpha_\theta \nabla_\theta \mathbb{E}_{s_t \sim p^{\pi_\theta}, a_t \sim \pi_\theta}[\log \pi_\theta(a_t \mid s_t) A_t^\tau];}$

`// Update value function (critic)`

$\phi \leftarrow \phi - \alpha_\phi \nabla_\phi \mathbb{E}_{s_t \sim p^{\pi_\theta}, a_t \sim \pi_\theta}\left[(V^\phi(s_t) - R_t^\tau)^2\right]$;

ACTION/VALUE PREDICTION - SUPERVISED LEARNING

- Dataset exists outside the loop
- Each sample used ***many times*** (epochs)
- ***Loss is computed on mini-batches***
    - Controls compute / memory costs
    - Stochasticity sometimes helpful

Algorithm 2: Supervised Learning Training Loop

Input: Initial model parameters θ
Input: $\boxed{\text{Labeled dataset } \mathcal{D} = \{(a_t^\tau, s_t^\tau, R_\tau)\}_{t,\tau}}$
while *not converged* **do**
    $\boxed{\text{Sample minibatch } \mathcal{B} = \{(a_j, s_j, R_j)\}_{j=1}^{B} \text{ from } \mathcal{D};}$
    `// Compute model predictions`
    $\hat{a}_t, \hat{v}_t = f_\theta(s_j)$ for all $(a_j, s_j, R_j) \in \mathcal{B}$;
    `// Compute average loss`
    $\boxed{\mathcal{L}(\theta) = \frac{1}{B}\sum_{j=1}^{B} \ell(\hat{a}_j, a_j, \hat{v}_j, R_j)} \approx \mathbb{E}_{(a_j, s_j, R_j)\sim\mathcal{D}}[\ell(\hat{a}_j, a_j, \hat{v}_j, R_j)]$;
    `// Update model parameters`
    $\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta)$;

HOW TO ADJUST

- Gradient approximated by minibatches
- Each (state, action, reward) sample used multiple times
    - Dataset size “separate” from optimization steps
- Dataset does not depend on θ and optimization
    - If π_θ shifts too much, some states never resampled
    - RL has stability concerns

HOW TO ADJUST

- Gradient approximated by minibatches
- Each (state, action, reward) sample used multiple times
  - Dataset size "separate" from optimization steps
- Dataset does not depend on θ and optimization
  - If π_θ shifts too much, some states never resampled
  - RL has stability concerns

Optimize *current* policy w.r.t samples collected on *older* policy

HOW TO ADJUST

- Gradient approximated by minibatches
- Each (state, action, reward) sample used multiple times
  - Dataset size "separate" from optimization steps
- Dataset does not depend on θ and optimization
  - If π_θ shifts too much, some states never resampled
  - RL has stability concerns

Optimize *current* policy w.r.t samples collected on *older* policy

Requires small policy updates

HOW TO ADJUST

- Gradient approximated by minibatches
- Each (state, action, reward) sample used multiple times
  - Dataset size "separate" from optimization steps

Optimize *current* policy w.r.t samples collected on *older* policy

Requires small policy updates

- Dataset does not depend on θ and optimization
  - If π_θ shifts too much, some states never resampled
  - RL has stability concerns

Small policy updates help

PROBLEM SCOPE!

✓ Optimize ***current*** policy w.r.t samples collected on ***older*** policy
- Break the on-policy assumption *a little bit*

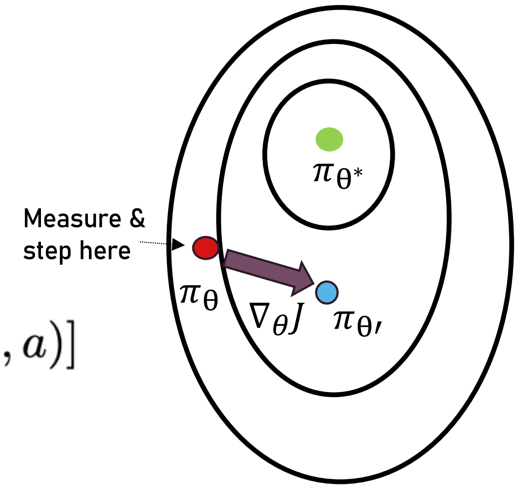
✗ Optimize ***current*** policy w.r.t samples collected on ***external (possibly expert)*** policy
- *No (or even weaker) on-policy assumption*

For this see:
- off-policy RL,
- behavioral cloning,
- offline RL
- "An operator view of policy gradient methods" (Ghosh, 2020)

WHAT DOES SAMPLE REUSE MEAN IN RL?

The usual setting:

- Samples collected by policy π_θ, optimize π_θ toward π_{θ^*}:
- Objective: $J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[R^{\pi_\theta}(\tau)\right]$
- Gradient expectation w.r.t π_θ: $\nabla_\theta J(\theta) \propto \mathbb{E}_{a \sim \pi_\theta}[\nabla_\theta \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a)]$
- Once we step policy parameters ($\theta' = \theta + \alpha \nabla_\theta J(\theta)$), no longer at π_θ



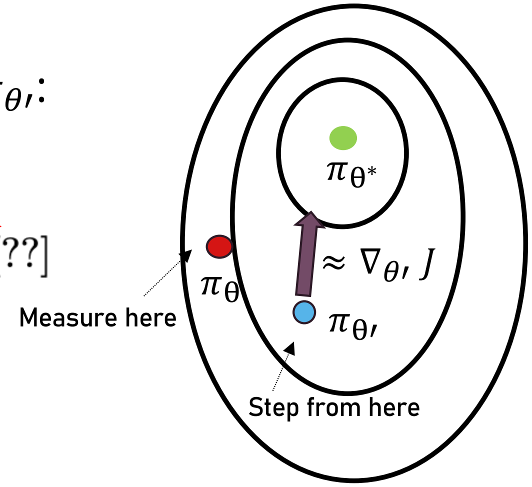
- If we want to step **AGAIN** ***without collecting new data***
  - Need new objective and estimator

WHAT DOES SAMPLE REUSE MEAN IN RL?

Old samples collected by policy π_θ, but we want to keep optimizing $\pi_{\theta'}$:

Optimize: $J(\theta') = \mathbb{E}_{\tau \sim \pi_{\theta'}}[R^{\pi_{\theta'}}(\tau)] \approx \mathbb{E}_{\tau \sim \pi_\theta}[??]$ ← Want to find

Gradient: $\nabla_\theta J(\theta') = \mathbb{E}_{s \sim p^{\pi_{\theta'}}, a \sim \pi_{\theta'}}[\nabla_{\theta'} \pi_{\theta'}(a \mid s) Q^{\pi_{\theta'}}(s,a)] \approx \mathbb{E}_{s \sim p^{\pi_\theta}, a \sim \pi_\theta}[??]$



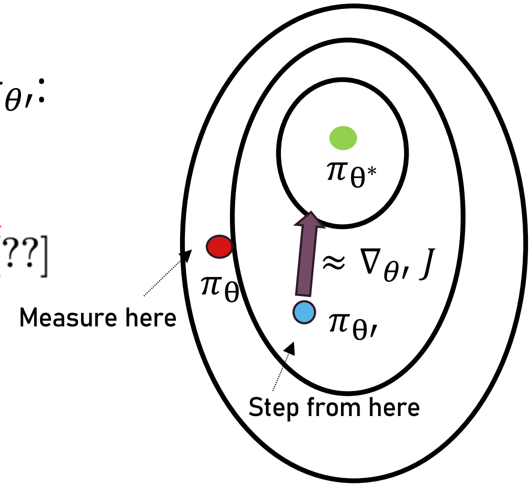
Requires a change of variable between probability densities

WHAT DOES SAMPLE REUSE MEAN IN RL?

Old samples collected by policy π_θ, but we want to keep optimizing $\pi_{\theta'}$:

Optimize: $J(\theta') = \mathbb{E}_{\tau\sim\pi_{\theta'}}\left[R^{\pi_{\theta'}}(\tau)\right] \approx \mathbb{E}_{\boxed{\tau\sim\pi_\theta}}[??]$ Want to find

Gradient: $\nabla_\theta J(\theta') = \mathbb{E}_{s\sim p^{\pi_{\theta'}},a\sim\pi_{\theta'}}\left[\nabla_{\theta'}\pi_{\theta'}(a \mid s) Q^{\pi_{\theta'}}(s,a)\right] \approx \mathbb{E}_{\boxed{s\sim p^{\pi_\theta},a\sim\pi_\theta}}[??]$



Requires a change of variable between probability densities

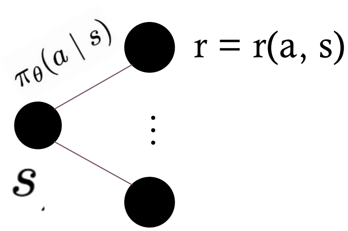
CHANGE OF VARIABLE – IMPORTANCE SAMPLING

Given two densities p and q:

$$\mathbb{E}_{x\sim p}[f(x)] = \sum_{x\in\mathcal{X}} p(x)f(x) = \sum_{x\in\mathcal{X}} q(x)\frac{p(x)}{q(x)}f(x) = \mathbb{E}_{\boxed{x\sim q}}\left[\frac{p(x)}{q(x)}f(x)\right]$$

Example: Single transition following $\pi_{\theta'}$, but sampled from π_θ:

$$J(\pi_{\theta'}) = \mathbb{E}_{(s,a,r)\sim\pi_{\theta'}}[r(\cdot,s)|s] = \sum_{a\in\mathcal{A}} \pi_{\theta'}(a|s)r(a,s) = \sum_{a\in\mathcal{A}} \pi_\theta(a|s)\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)}r(a,s) = \mathbb{E}_{\boxed{(s,a,r)\sim\pi_\theta}}[\frac{\pi_{\theta'}(\cdot|s)}{\pi_\theta(\cdot|s)}r(\cdot,s)|s]$$

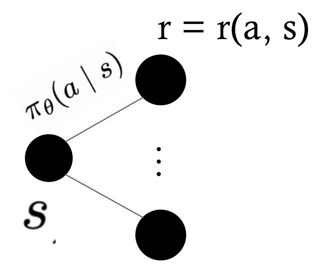


CHANGE OF VARIABLE – IMPORTANCE SAMPLING

The new re-centered estimator is unbiased (have the same true mean).

*But they are **different estimators**, at finite number of samples they have **DIFFERENT ERROR***

True Mean $$\mu = \mathbb{E}_{(s,a,r)\sim\pi_\theta}[\boxed{\frac{\pi_{\theta'}(\cdot|s)}{\pi_\theta(\cdot|s)} r(\cdot, s)}] = \mathbb{E}_{(s,a,r)\sim\pi_{\theta'}}[\boxed{r(\cdot, s)}]$$



Sample Mean $$\hat{\mu}_{\pi_\theta} = \sum_{(s,a,r)\in\mathcal{B}_{\pi_\theta}} \frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)} r(a, s) \neq \sum_{(s,a,r)\in\mathcal{B}_{\pi_{\theta'}}} r(a, s) = \hat{\mu}_{\pi_{\theta'}}$$

Sample batch under π_θ Sample batch under $\pi_{\theta'}$

IMPORTANCE SAMPLING – ERROR

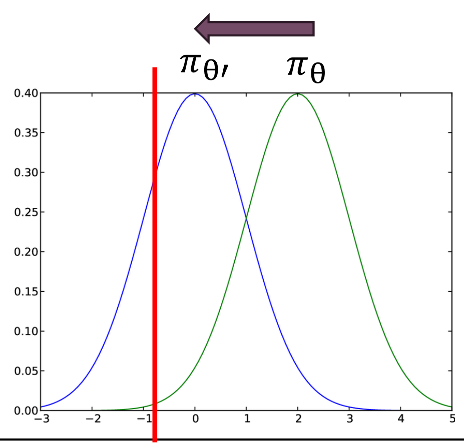
Error of re-centered IS estimator: $\mathrm{Var}_{(s,a,r)\sim\pi_\theta}[\hat{\mu}_{\pi_\theta}] = \int_{\mathcal{A}} \frac{(r(s,a)\pi_{\theta'}(s,a) - \mu\pi_\theta(s,a))^2}{\pi_\theta(s,a)} da$

In many (non-RL) applications: **You pick π_θ** so IS is *more* accurate than the original

In RL, the opposite is true. **Optimization picks π_θ**. We do IS bc we have to, **not to reduce error**

Error is typically (much) worse when: $\pi_\theta << \pi_{\theta'}$

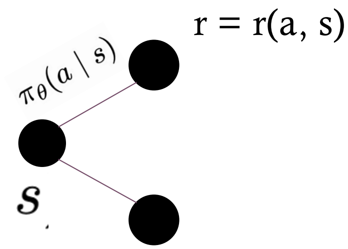
$|\pi_\theta - \pi_{\theta'}| >$ large (and r(s,a) nearly constant)

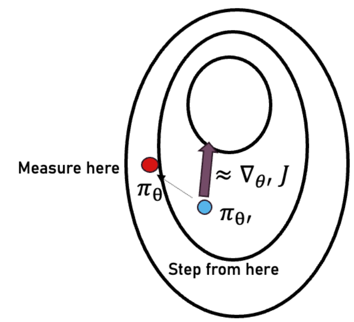


A policy that shifts too far can lead to terrible approximations

WHAT IS OUR NEW ESTIMATOR?

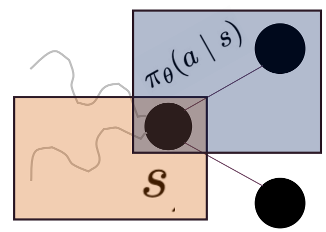
- With 1-transition: $\dfrac{\pi_{\theta'}(\cdot|s)}{\pi_\theta(\cdot|s)} r(\cdot, s)$





- But we got to state S from the old policy

$$p^{\pi_\theta}(s) \neq p^{\pi_{\theta'}}(s)$$



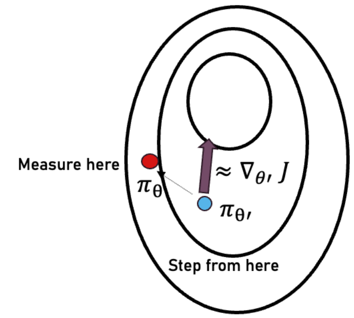
- Need importance sampling along the whole trajectory:

$$J(\pi) = \mathbb{E}\left[\sum_{h=1}^{H} \gamma^{h-1} r_h \;\middle|\; a_{1:H} \sim \pi\right] = \mathbb{E}_{\tau \sim p}\left[\sum_{h=1}^{H} \gamma^{h-1} r_h\right] = \mathbb{E}_{\tau \sim q}\left[\frac{p(\tau)}{q(\tau)} \sum_{h=1}^{H} \gamma^{h-1} r_h\right]$$

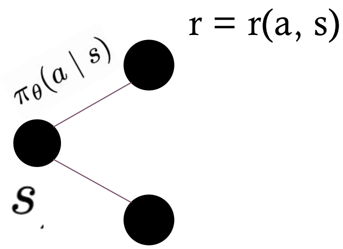
$$= \mathbb{E}_{\tau \sim q}\left[\frac{d_0(s_1)\pi(a_1|s_1)R(r_1|s_1,a_1)P(s_2|s_1,a_1)\cdots\pi(a_H|s_H)R(r_H|s_H,a_H)}{d_0(s_1)\pi_b(a_1|s_1)R(r_1|s_1,a_1)P(s_2|s_1,a_1)\cdots\pi_b(a_H|s_H)R(r_H|s_H,a_H)} \sum_{h=1}^{H} \gamma^{h-1} r_h\right]$$

Notes on Importance Sampling and Policy Gradient (Jiang, 2023)

WHAT IS OUR NEW ESTIMATOR?

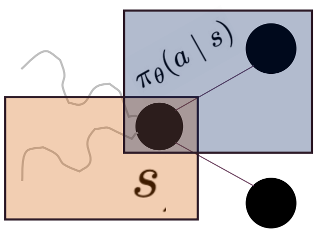


- With 1-transition: $\dfrac{\pi_{\theta'}(\cdot|s)}{\pi_\theta(\cdot|s)} r(\cdot, s)$



- But we got to state S from the old policy

$$p^{\pi_\theta}(s) \neq p^{\pi_{\theta'}}(s)$$

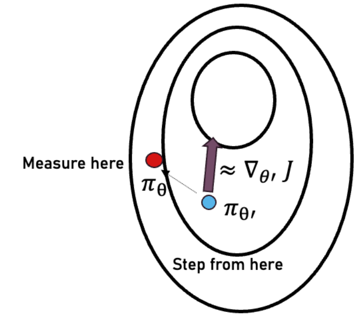


- Need importance sampling reweighting for every step in trajectory so far:

$$J(\pi) = \mathbb{E}\left[\sum_{h=1}^{H} \gamma^{h-1} r_h \;\middle|\; a_{1:H} \sim \pi\right] = \mathbb{E}_{\tau \sim p}\left[\sum_{h=1}^{H} \gamma^{h-1} r_h\right] = \mathbb{E}_{\tau \sim q}\left[\frac{p(\tau)}{q(\tau)} \sum_{h=1}^{H} \gamma^{h-1} r_h\right]$$

$$= \mathbb{E}_{\tau \sim q}\left[\frac{d_0(s_1)\boxed{\pi(a_1|s_1)}R(r_1|s_1,a_1)P(s_2|s_1,a_1)\cdots\boxed{\pi(a_H|s_H)}R(r_H|s_H,a_H)}{d_0(s_1)\boxed{\pi_b(a_1|s_1)}R(r_1|s_1,a_1)P(s_2|s_1,a_1)\cdots\boxed{\pi_b(a_H|s_H)}R(r_H|s_H,a_H)} \sum_{h=1}^{H} \gamma^{h-1} r_h\right]$$

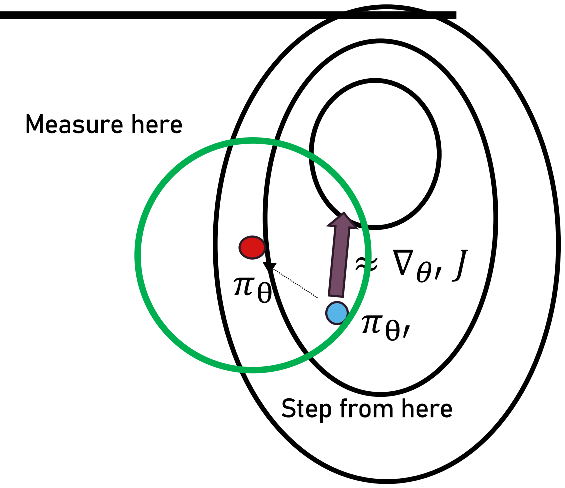
WHAT IS OUR NEW ESTIMATOR?



- Using the full path has problems:
  - Quadratic compute cost (in trajectory length) to reweight each sample
  - Variance & Error explodes: multiplying many small or large ratios

- Idea: Use an approximation based on the 1-step update as a proxy
  - Control the error by keeping the two policies close

WHAT IS OUR NEW ESTIMATOR?



- Claim: $p^{\pi_\theta}(s)$ is close to $p^{\pi_{\theta'}}(s)$, when π_θ is close to $\pi_{\theta'}$
- Assume $|\pi_\theta(a_t, s_t) - \pi_{\theta'}(a_t, s_t)| < \epsilon$,

Different action at each decision chosen with prob at most ϵ

$$|p_\theta(s_t) - p_{\theta'}(s_t)| = (1 - (1-\epsilon)^t)|p_{\theta'}(s_t) - p_\theta(s_t)| < 2\epsilon t$$

The approximate expected return using off-centered measurements at π_θ

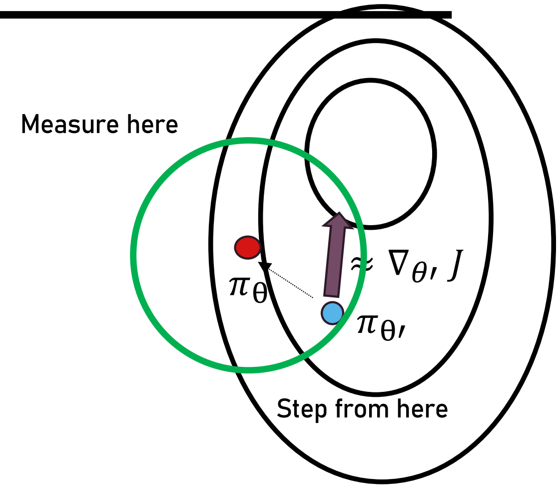
$$J(\theta')|_\theta \approx \mathbb{E}_{s_t \sim p_\theta, a_t \sim \pi_\theta(a_t|s_t)}\left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)}\gamma^t A^{\pi_\theta}(s_t, a_t)\right] + O\left(\frac{2t\epsilon r_{max}}{1-\gamma}\right)$$

This proxy objective is also a *lower bound*, optimizing this optimizes the original objective.

Advanced Policy Gradients (Levine, 2020)
Trust Region Policy Optimization (Schulman, 2015)

WHAT IS OUR NEW ESTIMATOR?

- Claim: $p^{\pi_\theta}(s)$ is close to $p^{\pi_{\theta'}}(s)$, when π_θ is close to $\pi_{\theta'}$
- Assume $|\,\pi_\theta(a_t, s_t) - \pi_{\theta'}(a_t, s_t)| < \epsilon$,



Different action at each decision chosen with prob at most ϵ

$$|p_\theta(s_t) - p_{\theta'}(s_t)| = (1 - (1-\epsilon)^t)|p_{\theta'}(s_t) - p_\theta(s_t)| < 2\epsilon t$$

The approximate expected return using off-centered measurements at π_θ

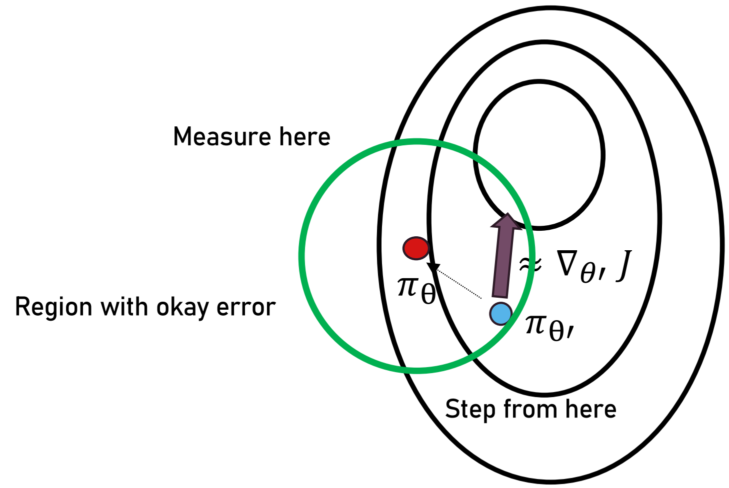
$$\boxed{J(\theta')}\Big|_\theta \approx \mathbb{E}_{\overset{\text{Old actions}}{\boxed{s_t\sim p_\theta,\, a_t\sim\pi_\theta(a_t|s_t)}}}\left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)}\gamma^t A^{\overset{\text{Old advantages}}{\boxed{\pi_\theta}}}(s_t, a_t)\right] + O\left(\frac{2t\epsilon r_{max}}{1-\gamma}\right)$$

This proxy objective is also a *lower bound*, optimizing this optimizes the original objective.

Advanced Policy Gradients (Levine, 2020)
Trust Region Policy Optimization (Schulman, 2015)

KEEPING POLICY UPDATES CLOSE

- If we keep π_θ and $\pi_{\theta'}$ close we can use the proxy

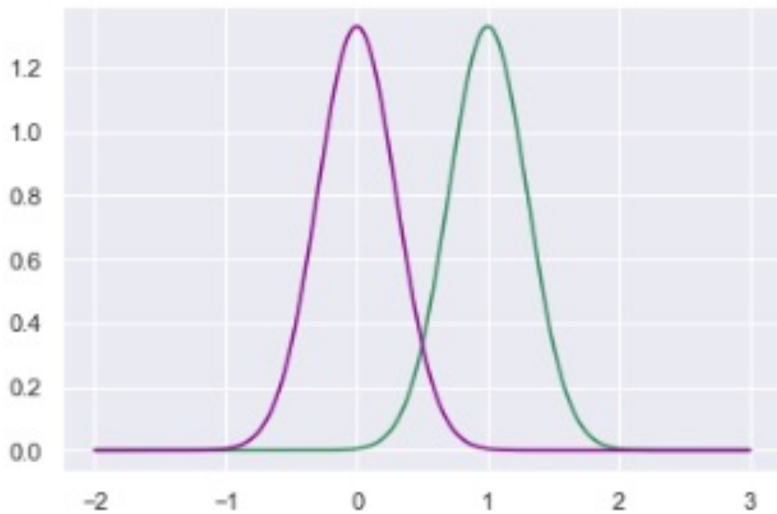


SMALL PARAMETER STEPS ! = SMALL POLICY UPDATES.

Example:

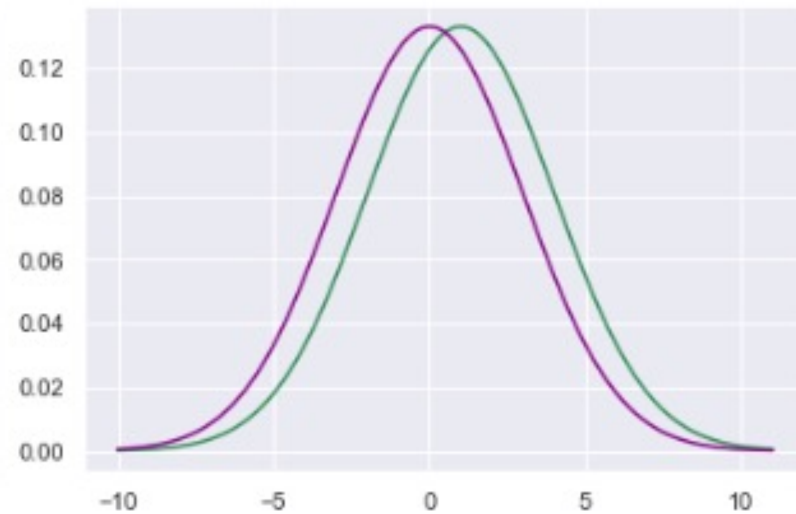
Gaussian w/ mean (θ_1) and std (θ_2): $\|\theta - \theta'\|= 1$ for both cases

$\Delta\mu = 1, \sigma_1 = \sigma_2 = 0.3$



Ratio between $\pi_{\theta'}(x)/\pi_\theta(x)$ large

$\Delta\mu = 1, \sigma_1 = \sigma_2 = 3$

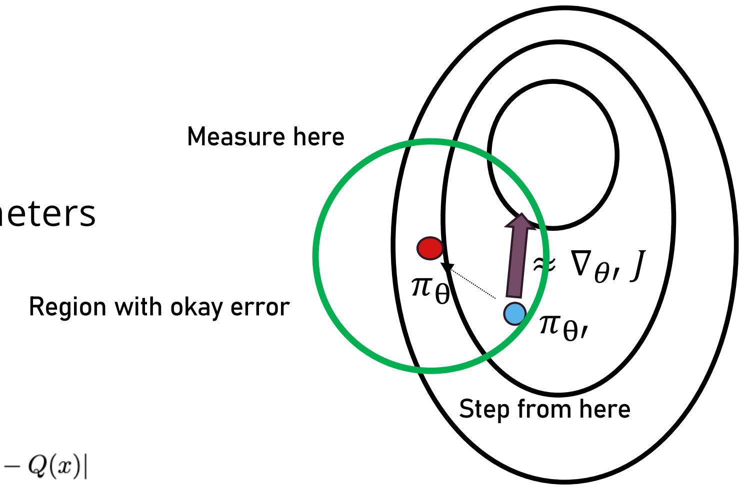


Ratio between $\pi_{\theta'}(x)/\pi_\theta(x)$ small

Natural Policy Gradients in Reinforcement Learning Explained (Heeswijk, 2022)

KEEPING POLICY UPDATES CLOSE

- If we keep π_θ and $\pi_{\theta'}$ close we can use the proxy
- Need to measure 'close' w.r.t. the distribution not the parameters



$$\max_{\theta'} \mathbb{E}_{s\sim p_\theta, a\sim\pi_\theta(a|s)}\left[\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)} A^{\pi_\theta}(a,s)\right]$$

$$\text{subject to} \quad ||\pi_{\theta'} - \pi_\theta||_{TV} < \epsilon \qquad ||P-Q||_{TV} = \frac{1}{2}\sum_x |P(x) - Q(x)|$$

- In practice, Total Variation (TV) is hard to optimize;
- KL-Divergence is easier. Pinsker's Inequality: $||\pi_{\theta'} - \pi_\theta||_{TV} \leq \sqrt{2D_{\mathrm{KL}}(\pi_{\theta'}||\pi_\theta)}$

$$\max_{\theta'} \mathbb{E}_{s\sim p_\theta, a\sim\pi_\theta(a|s)}\left[\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)} A^{\pi_\theta}(a,s)\right]$$

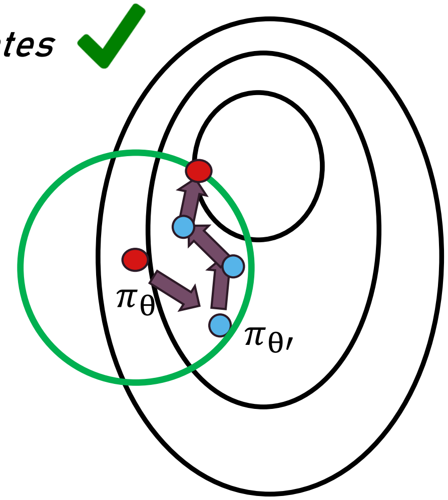
$$\text{subject to} \quad D_{\mathrm{KL}}(\pi_{\theta'}||\pi_\theta) < \delta \qquad D_{\mathrm{KL}}(P \parallel Q) = \sum_{x\in\mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}.$$

TRUST REGION POLICY OPTIMIZATION

- Gradient steps can be split by minibatches
- Each sample can be used multiple times
- If π_θ shifts too much, some states never resampled

Optimize current policy w.r.t samples collected on old policy

Require small policy updates



$$\max_{\theta'} \mathbb{E}_{s\sim p_\theta, a\sim\pi_\theta(a|s)}\left[\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)}A^{\pi_\theta}(a,s)\right]$$

$$\text{subject to} \quad D_{\text{KL}}(\pi_{\theta'}||\pi_\theta) < \delta$$

TRUST REGION POLICY OPTIMIZATION

- Gradient steps can be split by minibatches
- Each sample can be used multiple times

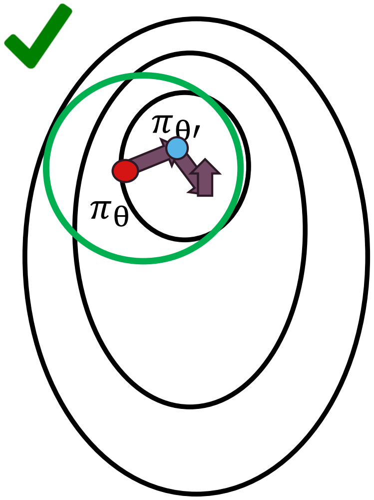
Optimize current policy w.r.t samples collected on old policy

- If π_θ shifts too much, some states never resampled

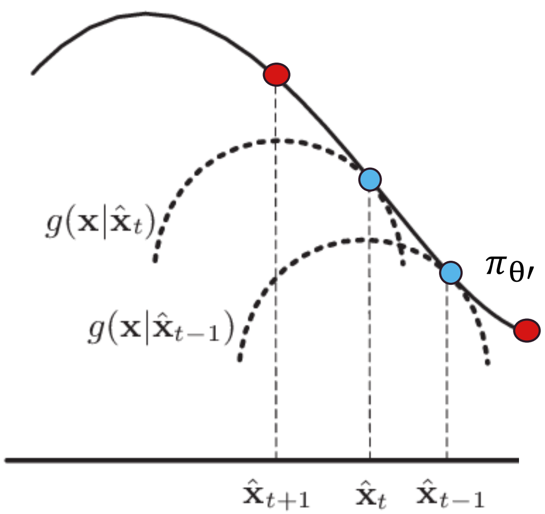
Require small policy updates

$$\max_{\theta'} \mathbb{E}_{s \sim p_\theta, a \sim \pi_\theta(a|s)} \left[\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)} A^{\pi_\theta}(a, s) \right]$$

$$\text{subject to} \quad D_{\mathrm{KL}}(\pi_{\theta'} || \pi_\theta) < \delta$$



HOW DO WE SOLVE THIS? TRUST REGION OPTIMIZATION



$$\max_{\theta'} \mathbb{E}_{s \sim p_\theta, a \sim \pi_\theta(a|s)} \left[\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)} A^{\pi_\theta}(a, s) \right]$$

$$\text{subject to } D_{\mathrm{KL}}(\pi_{\theta'}||\pi_\theta) \approx \underbrace{||\pi_{\theta'} - \pi_\theta||^2_{F_\theta}}_{\text{Fisher information metric}} < \delta$$

Solve each epoch (or mini-batch) step with constrained Newton's method

1. Form a local quadratic approximation to objective
2. Step as far as possible towards solution of quadratic
3. Backtrack if step overshoots constraint

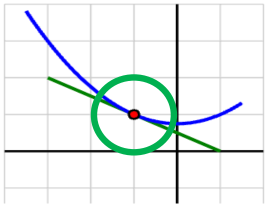
CONSTRAINED LINEAR UPDATE - EUCLIDEAN

$$\min_{x, \|x - x_0\| \leq \Delta} f(x)$$

Linear approximation:

$$\approx \min_{x \in U_k} m_k(x, x_k) = f(x_k) + \nabla f(x)^T (x - x_k)$$

$$U_k := \{ x : \|x - x_k\|_k \leq \Delta_k \}.$$



Solve approximation exactly within the ball:

$$x_{\min} \approx x^\star \in \arg\min_{x \in U_k} m_k(x).$$

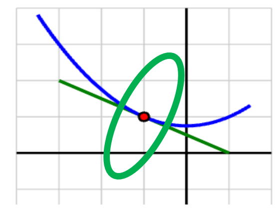
Farthest step down the line that stays in the radius

$$x^* = x_k - \Delta_k \frac{\nabla f(x)}{\|f(x)\|},$$

Update new center to x^* and iterate

CONSTRAINED LINEAR UPDATE – KL-DIVERGENCE

Ball is warped locally by F



$$\min_{D_{\mathrm{KL}}(x|x_k)\leq\Delta} f(x)$$

Linear approximation:

$$\approx \min_{x\in U_k} m_k(x, x_k) = f(x_k) + \nabla f(x)^T (x - x_k)$$

$$U_k := \{\, x : D_{\mathrm{KL}}(x|x_k) \approx ||x - x_k||_F = (x - x_k)^T F (x - x_k) \leq \Delta \,\},$$

Solve approximation exactly within the ball:

$$x_{\min} \approx x^\star \in \arg\min_{x\in U_k} m_k(x).$$

Farthest step down the line that stays in the radius

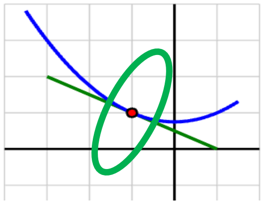
$$x^* = x_k - D_k \nabla f(x)$$

$$D_k = \sqrt{\frac{\Delta}{\nabla f(x_k)^T F^{-1}(x_k) \nabla f(x_k)}}\, F^{-1}(x_k)$$

Update new center to x^* and iterate

CONSTRAINED LINEAR UPDATE – KL-DIVERGENCE

Ball is warped locally by F



Solve approximation exactly within the ball:

$$x_{\min} \approx x^{\star} \in \arg\min_{x \in U_k} m_k(x).$$

Farthest step down the line that stays in the radius

$$x^{*} = x_k - D_k \nabla f(x)$$

$$D_k = \sqrt{\frac{\Delta}{\nabla f(x_k)^T F^{-1}(x_k) \nabla f(x_k)}} F^{-1}(x_k)$$

This is derived from Lagrange multipliers and KKT conditions, assuming solution is on the boundary of constraints

$$\mathcal{L}(s, \lambda) = g^{\top} s + \lambda \left(\frac{1}{2} s^{\top} F s - \Delta^2\right)$$

$$s = (x - x_k) \qquad := \nabla f(x_k),\ F := F(x_k) \succ 0$$

Stationarity $\nabla_s \mathcal{L} = 0 \;\Rightarrow\; s^{*} = -\frac{1}{\lambda} F^{-1} g,$

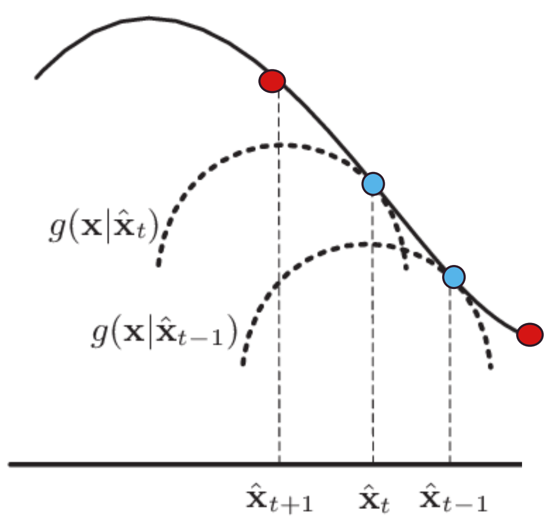
Boundary $s^{*\top} F s^{*} = \Delta^2 \;\Rightarrow\; \frac{1}{\lambda} = \frac{\Delta}{\sqrt{g^{\top} F^{-1} g}}.$

$$x_{k+1} = x_k + s^{*} = x_k - \frac{\Delta}{\sqrt{\nabla f(x_k)^{\top} F(x_k)^{-1} \nabla f(x_k)}} F(x_k)^{-1} \nabla f(x_k).$$

HOW DO WE SOLVE THIS?

$$\max_{\theta'} \mathbb{E}_{s\sim p_\theta, a\sim\pi_\theta(a|s)}\left[\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)} A^{\pi_\theta}(a,s)\right]$$

$$\text{subject to } D_{\mathrm{KL}}(\pi_{\theta'}||\pi_\theta) \approx ||\pi_{\theta'} - \pi_\theta||^2_{F_\theta} < \delta$$



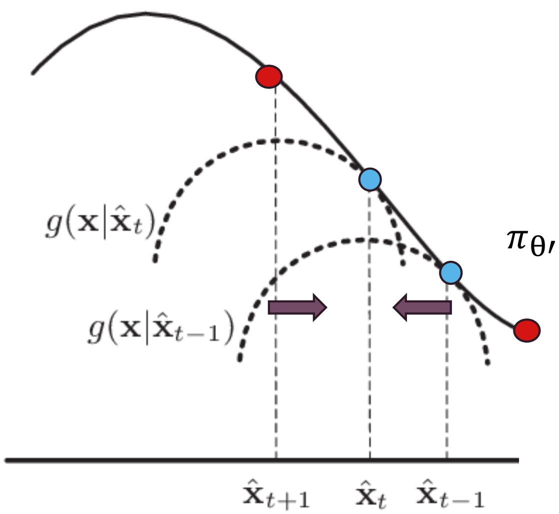
- Solve + backtracking

See also:
Newton's Method
Quasi-Newton / CG-Newton
Natural Gradient Methods

- Expensive! O(N^3) per step

SOFT CONSTRAINTS:

$$\max_{\theta'} \mathbb{E}_{s\sim p_\theta, a\sim\pi_\theta(a|s)}\left[\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)}A^{\pi_\theta}(a,s)\right] - \beta D_{\mathrm{KL}}(\pi_{\theta'}||\pi_\theta)$$



- Use first-order methods (SGD) on penalized problem
  - KL-penalty Proximal Policy Optimization (PPO-KL)
    - Adaptive β: x2 if constraint threshold violated, ÷2 if well within threshold
  - Mirror Descent Policy Optimization (MDPO)
    - Scheduled $\beta = \frac{c}{optimization\ step}$, (they also use reverse KL)
  - Very sensitive to choice of β

ARE THERE *EVEN* CHEAPER *ROBUST* APPROXIMATIONS?

- Clipped Proximal Policy Optimization (PPO)
  - Modify objective function to ”discourage” steps far from trust region

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

$$\max_\theta L_t^{\text{CLIP}}(\theta')|_\theta = \begin{cases} (1-\epsilon)A^{\pi_\theta} & \text{if } w(s,a) \leq 1-\epsilon \text{ and } A^{\pi_\theta} < 0 \\ (1+\epsilon)A^{\pi_\theta} & \text{if } w(s,a) \geq 1+\epsilon \text{ and } A^{\pi_\theta} > 0 \\ w(s,a)A^{\pi_\theta} & \text{otherwise} \end{cases}$$

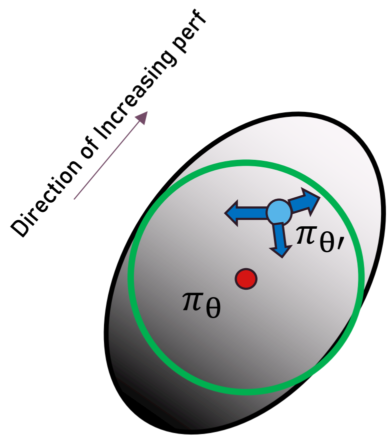
$$\nabla_{\theta'} L_t^{\text{CLIP}}(\theta')|_\theta = \begin{cases} 0 & \text{if } w(s,a) \leq 1-\epsilon \text{ and } A^{\pi_\theta} < 0 \\ 0 & \text{if } w(s,a) \geq 1+\epsilon \text{ and } A^{\pi_\theta} > 0 \\ \nabla_{\theta'} J(\theta')|_\theta & \text{otherwise} \end{cases}$$

ARE THERE *EVEN* CHEAPER APPROXIMATIONS?

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

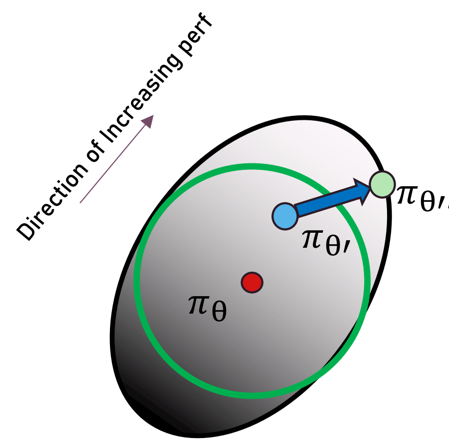
- Clipped Proximal Policy Optimization (PPO)

Per-Sample Gradient:
$$\nabla_{\theta'} L_t^{\text{CLIP}}(\theta')|_\theta = \begin{cases} 0 & \text{if } w(s,a) \leq 1-\epsilon \text{ and } A^{\pi_\theta} < 0 \\ 0 & \text{if } w(s,a) \geq 1+\epsilon \text{ and } A^{\pi_\theta} > 0 \\ \nabla_{\theta'} J(\theta')|_\theta & \text{otherwise} \end{cases}$$



Nearby ratio = step freely

$|w(s,a) - 1| < \epsilon$



Allows steps that bring policy OUTSIDE of threshold

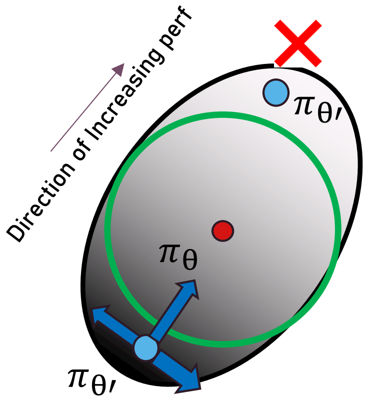
Restricts steps that can be taken once outside

UNDERSTANDING THE THRESHOLD

$$\nabla_{\theta'} L_t^{\text{CLIP}}(\theta')|_\theta = \begin{cases} 0 & \text{if } w(s,a) \leq 1-\epsilon \text{ and } A^{\pi_\theta} < 0 \\ 0 & \text{if } w(s,a) \geq 1+\epsilon \text{ and } A^{\pi_\theta} > 0 \\ \nabla_{\theta'} J(\theta')|_\theta & \text{otherwise} \end{cases}$$

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

Four cases:



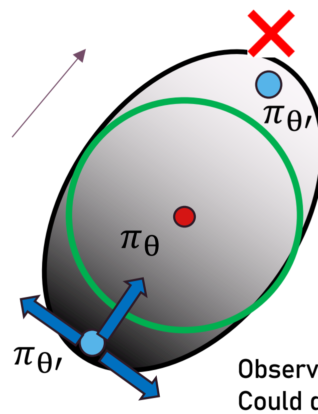
Observed a good action
Already have high likelihood
Shouldn't increase it too much

$w(s,a) > 1+\epsilon$

$A^{\pi_\theta} > 0$

Observed a good action, have low likelihood
Could increase likelihood more

Steps that:
- Increase w(s, a)
- Increase perf



Observed a bad action
Already have low likelihood
Shouldn't decrease it too much

$A^{\pi_\theta} < 0$

Observed a bad action, have high likelihood
Could decrease likelihood more

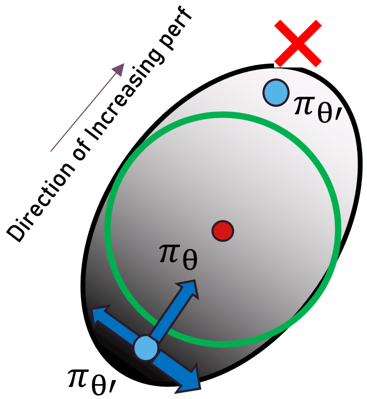
Steps that:
- Decrease w(s, a)
- Increase perf

UNDERSTANDING THE THRESHOLD

$$\nabla_{\theta'} L_t^{\text{CLIP}}(\theta')|_\theta = \begin{cases} 0 & \text{if } w(s,a) \leq 1-\epsilon \text{ and } A^{\pi_\theta} < 0 \\ 0 & \text{if } w(s,a) \geq 1+\epsilon \text{ and } A^{\pi_\theta} > 0 \\ \nabla_{\theta'} J(\theta')|_\theta & \text{otherwise} \end{cases}$$

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

Four cases:



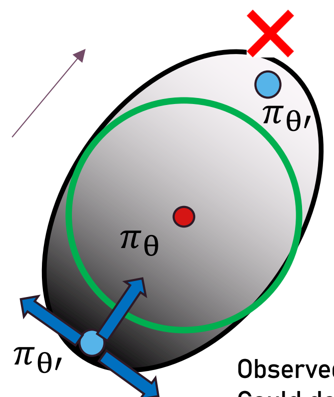
Observed a good action
Already have high likelihood
Shouldn't increase it too much

$w(s,a) > 1+\epsilon$

$A^{\pi_\theta} > 0$

Observed a good action, have low likelihood
Could increase likelihood more

Steps that:
- Increase w(s, a)
- Increase perf



Observed a bad action
Already have low likelihood
Shouldn't decrease it too much

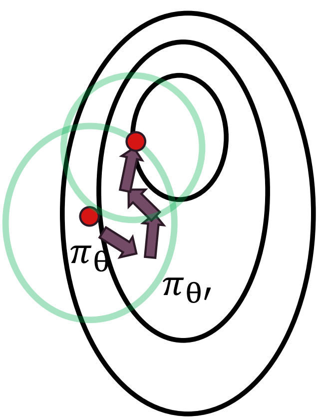
$A^{\pi_\theta} < 0$

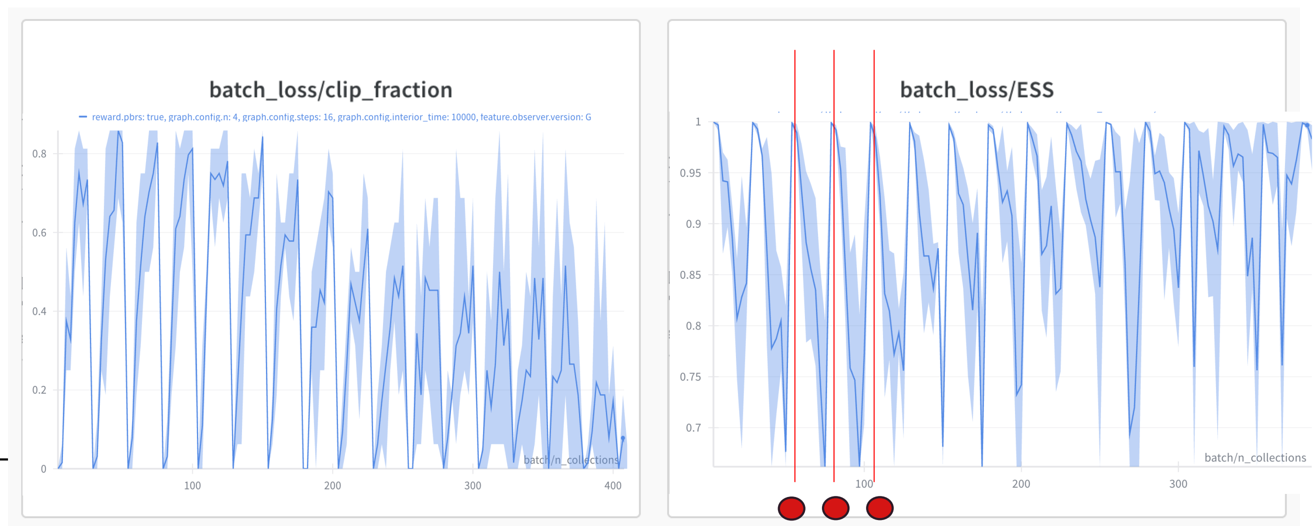
Observed a bad action, have high likelihood
Could decrease likelihood more

Steps that:
- Decrease w(s, a)
- Increase perf

PROXIMAL POLICY OPTIMIZATION: HEALTH MONITORING

- PPO is not even a “soft constraint” optimization -> important to monitor trust region and stability
- Two common metrics:
  - Clip Fraction -- what percentage of samples fall outside of the trust region after a step
  - Effective Sample Size (ESS) – IS estimator ratios, error behaves as if “ESS%” less samples were taken





π_θ is recentered every 16 batch steps

New data is collected

PPO OVERVIEW

Algorithm 2: Proximal Policy Optimization (PPO) Training Loop

Input: Initial policy parameters θ, value parameters ϕ
Input: Clip ϵ, epochs K, minibatch size M, entropy coef. c_s, value-loss coef. c_v, learning rates $\alpha_\theta, \alpha_\phi$

```
while not converged do
    // Rollout with the current policy
    Collect trajectories {τ_i}_{i=1}^N with π_θ to obtain {(s_t, a_t, r_t, s_{t+1})};
    Compute returns R_t (e.g., discounted) and advantages Â_t (e.g., GAE);
    Normalize advantages: Â_t ← (Â_t − μ_Â)/σ_Â;
    Store behavior log-probabilities log π_θold(a_t|s_t); set θ_old ← θ;
    for k = 1, 2, ..., K                                   // epochs do
        Shuffle the dataset into minibatches {B_b}_b of size M;
        foreach minibatch B_b do
            // Update policy (actor): gradient ascent on total objective
            θ ← θ + α_θ ∇_θ (L_clip(θ) + c_s L_ent(θ));
            // Update value function (critic): gradient descent on value loss
            φ ← φ − α_φ ∇_φ L_value(φ);
```

Key updates:

$$\hat{A}_t \leftarrow (\hat{A}_t - \mu_{\hat{A}})/\sigma_{\hat{A}}$$

$$\theta \leftarrow \theta + \alpha_\theta \nabla_\theta \Big(\mathcal{L}_{\text{clip}}(\theta) + c_s\, \mathcal{L}_{\text{ent}}(\theta)\Big)$$

$$\phi \leftarrow \phi - \alpha_\phi \nabla_\phi \mathcal{L}_{\text{value}}(\phi)$$

SO FAR

- Introduced "old sample" corrections to the policy network update
  - Requires taking small steps (in "policy space") to ensure low variance & accurate approximations

- What about corrections update to the Value/Critic/Baseline network?
  - In general, value network is "less sensitive" to using old samples than the policy network
    - Phasic Policy Gradient (Cobbe, Schulman, 2020) –different number of steps for each / different update frequency
  - But, yes, it requires a similar correction when samples may be "much" older
  - IMPALA and V-Trace are similar importance sampling updates to the critic!
    - See also:
      - Off-Policy Actor Critic (Degris, 2012), ReTrace (Munos, 2016),. SEED-RL (Espeholt, 2020)

IMPALA

- Samples may be several updates behind current policy
  - Need state distribution correction to N-step Temporal Difference (TD) Target

N-step Temporal Difference (TD) Value Target:

$$V(x_s) = V(x_s) + \sum_{t=s}^{s+n-1} \gamma^{t-s}(r_t + \gamma V(x_{t+1}) - V(x_s))$$

Network Parameter updates in direction of: $(V(x_s) - V_{\theta'}(x_s))\nabla V_{\theta'}(x_s)$

STATE DISTRIBUTION CORRECTION

N-step Temporal Difference (TD) Value Target:

$$V(x_s) = V(x_s) + \sum_{t=s}^{s+n-1} \gamma^{t-s}(r_t + \gamma V(x_{t+1}) - V(x_s))$$

Reweight future state probabilities

$$V(x_s) = V(x_s) + \sum_{t=s}^{s+n-1} \gamma^{t-s}(\prod_{i=s}^{t-1} \frac{\pi_{\theta'}(a_i|x_i)}{\pi_\theta(a_i|x_i)})\frac{\pi_{\theta'}(a_t|x_t)}{\pi_\theta(a_t|x_t)}(r_t + \gamma V(x_{t+1}) - V(x_s))$$

- To reduce variance blowup, IMPALA applies truncated IS estimators:

$$c_t = \min(\frac{\pi_{\theta'}(a_t|x_t)}{\pi_\theta(a_t|x_t)}, \tau_n)$$

(Ionides, 2008) - Original
(Liang, Fu, 2025) – New Bias Bounds

OTHER REFERENCES & EXTERNAL SOURCES

- Book on trust regions & optimization theory: Numerical Optimization by Nocedal & Wright

- Trust Region Policy Optimization (Schulman, 2015)
- Proximal Policy Optimization (Schulman, 2017)
- Phasic Policy Gradient (Schulman, 2020)
- New Insights and Perspectives on the Natural Gradient Method
- Natural, Trust Region and Proximal Policy Optimization (Blog Post)
- Natural Policy Gradients in Reinforcement Learning Explained (Heeswijk, 2022)

- Advanced Policy Gradients, Slides from Sergey Levine's CS285 Course.
- Trust Region Methods (Lectures 14/15), CS885 Pascal Poupart's Course.

- What Matters In On-Policy Reinforcement Learning? A Large-Scale Empirical Study
- Implementation Matters in Deep Policy Gradients: PPO and TRPO

- Notes on Importance Sampling and Policy Gradient (Jiang, 2023)

OTHER REFERENCES & EXTERNAL SOURCES

- Papers that tie Policy Gradient theory with other methods:
  - Monte-Carlo Tree Search as Regularized Policy Optimization
  - A Theory of Regularized Markov Decision Processes
  - An Operator View of Policy Gradient Methods