Deep Reinforcement Learning and Control

Natural Policy Gradients

CMU 10-403

Katerina Fragkiadaki
Revision
Policy Gradient

- Let $U(\theta)$ be any policy **objective function**

- Policy gradient algorithms search for a **local** maximum in $U(\theta)$ by ascending the gradient of the policy, w.r.t. parameters $\theta$

\[
\theta_{\text{new}} = \theta_{\text{old}} + \Delta \theta
\]

\[
\Delta \theta = \alpha \nabla_\theta U(\theta)
\]

$\alpha$ is a step-size parameter (learning rate)

is the **policy gradient**

\[
\nabla_\theta U(\theta) = \left( \begin{array}{c} \frac{\partial U(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial U(\theta)}{\partial \theta_n} \end{array} \right)
\]

Policy gradient: the gradient of the policy objective w.r.t. the parameters of the policy
Computing the policy gradient

**Likelihood ratio gradient estimator**

\[
\max_{\theta} \cdot U(\theta) = \mathbb{E}_{x \sim P_{\theta}(x)} f(x)
\]

\[
\nabla U(\theta) = \mathbb{E}_{x \sim P_{\theta}(x)} \nabla_{\theta} \log P_{\theta}(x) f(x)
\]

\[
\max_{\theta} \cdot U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [R(\tau)]
\]

\[
\nabla U(\theta) = \mathbb{E}_{\tau \sim P_{\theta}(\tau)} [\nabla_{\theta} \log P_{\theta}(\tau) R(\tau)]
\]
Derivatives of expectations

\[ \nabla_{\theta} \mathbb{E}_x f(x) = \nabla_{\theta} \mathbb{E}_{x \sim P_{\theta}(x)} [f(x)] \]

\[ = \nabla_{\theta} \sum_x P_{\theta}(x)f(x) \]

\[ = \sum_x \nabla_{\theta} P_{\theta}(x)f(x) \]

\[ = \sum_x P_{\theta}(x) \frac{\nabla_{\theta} P_{\theta}(x)}{P_{\theta}(x)} f(x) \]

\[ = \sum_x P_{\theta}(x) \nabla_{\theta} \log P_{\theta}(x)f(x) \]

\[ = \mathbb{E}_{x \sim P_{\theta}(x)} \left[ \nabla_{\theta} \log P_{\theta}(x)f(x) \right] \]

\[ \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_{\theta} \log P_{\theta}(x^{(i)})f(x^{(i)}) \]

From the law of large numbers, I can obtain an unbiased estimator for the gradient by sampling!
Computing the policy gradient

Likelihood ratio gradient estimator

\[
\max_{\theta} \quad U(\theta) = \mathbb{E}_{x \sim P_\theta(x)} f(x)
\]

\[
\nabla U(\theta) = \mathbb{E}_{x \sim P_\theta(x)} \nabla \log P_\theta(x) f(x)
\]

Chain rule of derivatives

\[
y = P_\theta(x)
\]

\[
\max_{\theta} \quad U(\theta) = f(P_\theta(x))
\]

\[
\nabla U(\theta) = \frac{df(P_\theta(x))}{d\theta} = \frac{df(y)}{dy} \frac{dy}{d\theta}
\]

\[
\max_{\theta} \quad U(\theta) = \mathbb{E}_{\tau \sim P_\theta(\tau)} \left[ R(\tau) \right]
\]

\[
\nabla U(\theta) = \mathbb{E}_{\tau \sim P_\theta(\tau)} \left[ \nabla \log P_\theta(\tau) R(\tau) \right]
\]

\[
a = \pi_\theta(s)
\]

\[
\max_{\theta} \quad U(\theta) = \mathbb{E} \sum_t Q^\pi(S_t, \pi_\theta(S_t))
\]

\[
\nabla U(\theta) = \mathbb{E} \sum_t \frac{d\mathbb{E} \sum_t Q^\pi(S_t, \pi_\theta(S_t))}{d\theta} = \mathbb{E} \sum_t \frac{dQ^\pi(S_t, a)}{da} \frac{d\pi_\theta(S_t)}{d\theta}
\]

\[
a = \pi_\theta(s)
\]

\[
\max_{\theta, \phi} \quad U(\theta, \phi) = \mathbb{E} \sum_t Q^\phi(S_t, \pi_\theta(S_t))
\]

\[
\frac{\partial U(\theta, \phi)}{\partial \theta} = \frac{\partial \mathbb{E} \sum_t Q^\phi(S_t, \pi_\theta(S_t))}{\partial \theta} = \mathbb{E} \sum_t \frac{\partial Q^\phi(S_t, a)}{da} \frac{d\pi_\theta(S_t)}{d\theta}
\]

\[
\frac{\partial U(\theta, \phi)}{\partial \phi} = \frac{\partial \mathbb{E} \sum_t Q^\phi(S_t, \pi_\theta(S_t))}{\partial \phi} = \mathbb{E} \sum_t \frac{\partial Q^\phi(S_t, a)}{\partial \phi}
\]
Deep Deterministic Policy Gradients

$$\frac{\partial U(\theta, \phi)}{\partial \theta} = \frac{\partial \mathbb{E} \sum_t Q_\phi(S_t, \pi_\theta(S_t))}{\partial \theta} = \mathbb{E} \sum_t \frac{\partial Q_\phi(S_t, a)}{\partial a} \frac{d\pi_\theta(S_t)}{d\theta}$$

$$\frac{\partial U(\theta, \phi)}{\partial \phi} = \frac{\partial \mathbb{E} \sum_t Q_\phi(S_t, \pi_\theta(S_t))}{\partial \phi} = \mathbb{E} \sum_t \frac{\partial Q_\phi(S_t, a)}{\partial \phi}$$
Computing the policy gradient

\[ \max_{\theta} . \ U(\theta) = \mathbb{E}_{x \sim P_\theta(x)} f(x) \]

\[ \nabla U(\theta) = \mathbb{E}_{x \sim P_\theta(x)} \nabla \log P_\theta(x) f(x) \]

**Likelihood ratio gradient estimator**

\[ \max_{\theta} . \ U(\theta) = \mathbb{E}_{\tau \sim P_\theta(\tau)} [R(\tau)] \]

\[ \nabla U(\theta) = \mathbb{E}_{\tau \sim P_\theta(\tau)} [\nabla \log P_\theta(\tau) R(\tau)] \]

**Chain rule of derivatives**

\[ y = P_\theta(x) \]

\[ \max_{\theta} . \ U(\theta) = f(P_\theta(x)) \]

\[ \nabla U(\theta) = \frac{df(P_\theta(x))}{d\theta} = \frac{df(y)}{dy} \frac{dy}{d\theta} \]

\[ a = \pi_\theta(s) \]

\[ \max_{\theta} . \ U(\theta) = \mathbb{E} \sum_t Q(S_t, \pi_\theta(S_t)) \]

\[ \nabla U(\theta) = \mathbb{E} \sum_t \frac{d}{d\theta} \frac{dQ(S_t, \pi_\theta(S_t))}{da} \frac{da}{d\theta} \]

**Re-parametrization for Gaussian policies**

\[ \max_{\theta} . \ U(\theta) = \mathbb{E}_{x \sim \mathcal{N}(\mu_\theta, \Sigma_\theta)} f(x) \]

\[ \max_{\theta} . \ U(\theta) = \mathbb{E}_{A_t \sim \mathcal{N}(\mu_\theta(S_t), \sigma_\theta(S_t))} \sum_t Q^\pi(S_t, A_t) \]

\[ \max_{\theta} . \ U(\theta) = \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t Q^\pi (S_t, \mu_\theta(S_t) + z * \sigma_\theta(S_t)) \]
Instead of:  \( a \sim \mathcal{N}(\mu_\theta(s), \Sigma_\theta(s)) \)

We can write:  \( a = \mu_\theta(s) + z \odot \sigma_\theta(s) \)  \( z \sim \mathcal{N}(0, I) \)

Why?

\[
\mathbb{E}_z(\mu_\theta(s) + z\sigma_\theta(s)) = \mu_\theta(s)
\]

Because:

\[
\text{Var}_z(\mu_\theta(s) + z\sigma_\theta(s)) = \sigma_\theta(s)^2
\]
Computing the policy gradient

Likelihood ratio gradient estimator

$$\max_\theta . \quad U(\theta) = \mathbb{E}_{x \sim P_\theta(x)} f(x)$$

$$\nabla U(\theta) = \mathbb{E}_{x \sim P_\theta(x)} \nabla \theta \log P_\theta(x) f(x)$$

Chain rule of derivatives

$$y = P_\theta(x)$$

$$\max_\theta . \quad U(\theta) = f(P_\theta(x))$$

$$\nabla U(\theta) = \frac{df(P_\theta(x))}{d\theta} = \frac{df(y)}{dy} \frac{dy}{d\theta}$$

Re-parametrization for Gaussian policies

$$\max_\theta . \quad U(\theta) = \mathbb{E}_{x \sim \mathcal{N}(\mu_\theta, \Sigma_\theta)} f(x)$$

$$\max_\theta . \quad U(\theta) = \mathbb{E}_{\tau \sim P_\theta(\tau)} [R(\tau)]$$

$$\nabla U(\theta) = \mathbb{E}_{\tau \sim P_\theta(\tau)} [\nabla \theta \log P_\theta(\tau) R(\tau)]$$

$$a = \pi_\theta(s)$$

$$\max_\theta . \quad U(\theta) = \mathbb{E} \sum_t Q(S_t, \pi_\theta(S_t))$$

$$\nabla U(\theta) = \frac{d\mathbb{E} \sum_t Q(S_t, \pi_\theta(S_t))}{d\theta} = \mathbb{E} \sum_t \frac{dQ(S_t, a)}{da} \frac{d\pi_\theta(S_t)}{d\theta}$$

$$\max_{\theta, \phi} . \quad U(\theta, \phi) = \mathbb{E}_{A_t \sim \mathcal{N}(\mu_\theta(S_t), \Sigma_\theta(S_t))} \sum_t Q_\phi(S_t, A_t)$$

$$\max_{\theta, \phi} . \quad U(\theta, \phi) = \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t Q_\phi(S_t, \mu_\theta(S_t) + z\sigma_\theta(s))$$

$$\frac{\partial U(\theta, \phi)}{\partial \theta} = \frac{\partial \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t Q_\phi(S_t, \mu_\theta(S_t) + z\sigma_\theta(s))}{\partial \theta} = \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t \frac{\partial Q_\phi(S_t, a)}{da} \frac{d(\mu_\theta(S_t) + z\sigma_\theta(s))}{d\theta}$$

$$\frac{\partial U(\theta, \phi)}{\partial \phi} = \frac{\partial \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t Q_\phi(S_t, \mu_\theta(S_t) + z\sigma_\theta(s))}{\partial \phi} = \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t \frac{\partial Q_\phi(S_t, \mu_\theta(S_t) + z\sigma_\theta(s))}{\partial \phi}$$
Stochastic Value Gradients

\[ z \sim \mathcal{N}(0, 1) \]

\[ s \xrightarrow{\theta} a \]

\[ s \xrightarrow{\phi} Q_\phi(s, \pi_\theta(a)) \]

\[
\frac{\partial U(\theta, \phi)}{\partial \theta} = \frac{\partial \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t Q_\phi(S_t, \mu_\theta(S_t) + z\sigma_\theta(s))}{\partial \theta} = \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t \frac{\partial Q_\phi(S_t, a)}{\partial a} \frac{d(\mu_\theta(S_t) + z\sigma_\theta(s))}{d\theta}
\]

\[
\frac{\partial U(\theta, \phi)}{\partial \phi} = \frac{\partial \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t Q_\phi(S_t, \mu_\theta(S_t) + z\sigma_\theta(s))}{\partial \phi} = \mathbb{E}_{z \sim \mathcal{N}(0, I)} \sum_t \frac{\partial Q_\phi(S_t, \mu_\theta(S_t) + z\sigma_\theta(s))}{\partial \phi}
\]

*Learning continuous control by stochastic value gradients*, Hees et al.
Actor-critic

1. Sample trajectories \( \{s^i_t, a^i_t\}^T_{i=0} \) by running the current policy \( a \sim \pi_\theta(s) \)
2. Fit value function \( V^\pi_\phi(s) \) by MC or TD estimation (update \( \phi \))
3. Compute advantages \( A^\pi_i(s^i_t, a^i_t) = R(s^i_t, a^i_t) + \gamma V^\pi_\phi(s^i_{t+1}) - V^\pi_\phi(s^i_t) \)
4. \( \nabla_\theta U(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\alpha^i_t | s^i_t) A^\pi_i(s^i_t, a^i_t) \)
5. \( \theta' = \theta + \alpha \nabla_\theta U(\theta) \)
Actor-critic

1. Sample trajectories \( \{s_i^t, a_i^t\}_{i=0}^T \) by running the current policy \( a \sim \pi_\theta(s) \)
2. Fit value function \( V_\phi^\pi(s) \) by MC or TD estimation (update \( \phi \))
3. Compute advantages \( A_\pi^\pi(s_i^t, a_i^t) = R(s_i^t, a_i^t) + \gamma V_\phi^\pi(s_{i+1}^t) - V_\phi^\pi(s_i^t) \)
4. \( \nabla_\theta U(\theta) \approx \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_\theta \log \pi_\theta(a_i^t | s_i^t) A_\pi^\pi(s_i^t, a_i^t) \)
5. \( \theta' = \theta + \alpha \nabla_\theta U(\theta) \)

+ simple & stable
- no shared features between actor & critic
Actor-critic

1. Sample trajectories \( \{s_t^{(i)}, a_t^{(i)}\}_{i=0}^T \) by running the current policy \( a \sim \pi_\theta(s) \)
2. Fit value function \( V_\phi^\pi(s) \) by MC or TD estimation (update \( \phi \))
3. Compute advantages \( A^\pi(s_t^{(i)}, a_t^{(i)}) = R(s_t^{(i)}, a_t^{(i)}) + \gamma V_\phi^\pi(s_{t+1}^{(i)}) - V_\phi^\pi(s_t^{(i)}) \)
4. \( \nabla_\theta U(\theta) \approx \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_\theta \log \pi_\theta(\alpha_t^{(i)} | s_t^{(i)}) A^\pi(s_t^{(i)}, a_t^{(i)}) \)
5. \( \theta' = \theta + \alpha \nabla_\theta U(\theta) \)

Figure from Sergey Levine
Critics are state dependent baselines

\[ \hat{g} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T} \nabla_{\theta} \log \pi_\theta(a_t^{(i)} | s_t^{(i)})(G_t^{i} - b) \]

+ no bias
- higher variance (because single-sample estimate)

\[ \hat{g} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T} \nabla_{\theta} \log \pi_\theta(a_t^{(i)} | s_t^{(i)})(G_t^{i} - b(s_t^{(i)})) \]

+ no bias
+ lower variance (baseline is closer to rewards)

\[ \hat{g} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_{\theta} \log \pi_\theta(\alpha_t^{(i)} | s_t^{(i)}) \left( R(s_t^{(i)}, a_t^{(i)}) + \gamma V^\pi(s_{t+1}^{(i)}) - V^\pi(s_t^{(i)}) \right) \]

+ lower variance (due to critic)
- not unbiased (if the critic is not perfect)
Policy Gradient

- Let $U(\theta)$ be any policy **objective function**

- Policy gradient algorithms search for a **local** maximum in $U(\theta)$ by ascending the gradient of the policy, w.r.t. parameters $\theta$

$$
\theta_{\text{new}} = \theta_{\text{old}} + \Delta \theta
$$

$$
\Delta \theta = \alpha \nabla_\theta U(\theta)
$$

$\alpha$ is a step-size parameter (learning rate)

This lecture is all about the stepsizes

is the **policy gradient**

$$
\nabla_\theta U(\theta) = \left( \begin{array}{c}
\frac{\partial U(\theta)}{\partial \theta_1} \\
\vdots \\
\frac{\partial U(\theta)}{\partial \theta_n}
\end{array} \right)
$$

Policy gradient: the gradient of the policy objective w.r.t. the parameters of the policy
1. Collect trajectories for policy $\pi_\theta$
2. Estimate advantages $A$
3. Compute policy gradient $\hat{g}$
4. Update policy parameters $\theta_{new} = \theta + \epsilon \cdot \hat{g}$
5. GOTO 1

This lecture is all about the stepsize.
What is the underlying objective function?

Policy gradients:
\[
\hat{g} \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_{\theta} \log \pi_{\theta}(\alpha_{i}^{(i)} | s_{t}^{(i)}) A(s_{t}^{(i)}, a_{t}^{(i)}), \quad \tau_{i} \sim \pi_{\theta}
\]

This result from differentiating the following objective function:
\[
U^{PG}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \log \pi_{\theta}(\alpha_{t}^{(i)} | s_{t}^{(i)}) A(s_{t}^{(i)}, a_{t}^{(i)}) \quad \tau_{i} \sim \pi_{\theta}
\]

Compare this to supervised learning using expert actions \( \tilde{a} \sim \pi^{*} \) and a maximum likelihood objective:
\[
U^{SL}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \log \pi_{\theta}(\tilde{\alpha}_{t}^{(i)} | s_{t}^{(i)}), \quad \tau_{i} \sim \pi^{*} \quad (+\text{regularization})
\]

This maximizes the probability of expert actions in the training set.

We want to optimize both objectives using gradient descent
\[
\theta' = \theta + \alpha \nabla_{\theta} U(\theta)
\]

Choosing stepsize \( \alpha \) is more critical for RL than for SL.

Why?

Because we cannot optimize it too far, our advantage estimates come from \( \pi_{\theta_{old}} \)
Policy Gradients

1. Collect trajectories for policy $\pi_\theta$
2. Estimate advantages $A$
3. Compute policy gradient $\hat{g}$
4. Update policy parameters $\theta_{\text{new}} = \theta + \epsilon \cdot \hat{g}$
5. GOTO 1

Two problems:
1. Hard to choose stepwise $\epsilon$
2. Sample inefficient: we cannot use data collected with policies of previous iterations
Hard to choose stepsizes

1. Collect trajectories for policy $\pi_\theta$
2. Estimate advantages $A$
3. Compute policy gradient $\hat{g}$
4. Update policy parameters $\theta_{\text{new}} = \theta + \epsilon \cdot \hat{g}$
5. GOTO 1

- Step too big
  Bad policy -> data collected under bad policy -> we cannot recover
  (in Supervised Learning, data does not depend on neural network weights)
- Step too small
  Not efficient use of experience
  (in Supervised Learning, data can be trivially re-used)

Gradient descent in parameter space does not take into account the resulting distance in the (output) policy space between $\pi_{\theta_{\text{old}}}(s)$ and $\pi_{\theta_{\text{new}}}(s)$
Hard to choose stepsizes

1. Collect trajectories for policy $\pi_\theta$
2. Estimate advantages $A$
3. Compute policy gradient $\hat{g}$
4. Update policy parameters $\theta_{new} = \theta + \epsilon \cdot \hat{g}$
5. GOTO 1

Consider a family of policies with parametrization:

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

The same parameter step $\Delta \theta = -2$ changes the policy distribution more or less dramatically depending on where in the parameter space we are.
Two Limitations of "Vanilla" Policy Gradient Methods

- Hard to choose stepsizes
- Input data is nonstationary due to changing policy: observation and reward distributions change
- Bad step is more damaging than in supervised learning, since it affects visitation distribution
- Step too far!
- Next batch: collected under bad policy
- Can't recover—collapse in performance

Sample efficiency
- Only one gradient step per environment sample
- Dependent on scaling of coordinates

Notation

We will use the following to denote values of parameters and corresponding policies before and after an update:

\[ \theta_{old} \rightarrow \theta_{new} \]
\[ \pi_{old} \rightarrow \pi_{new} \]
\[ \theta \rightarrow \theta' \]
\[ \pi \rightarrow \pi' \]
Gradient Descent in Parameter Space

The stepwise in gradient descent results from solving the following optimization problem, e.g., using line search:

\[ d^* = \arg \max_{\|d\| \leq \varepsilon} J(\theta + d) \]

SGD: \[ \theta_{new} = \theta_{old} + d^* \]

Euclidean distance in parameter space

It is hard to predict the result on the parameterized distribution. Hard to pick the threshold epsilon.
Gradient Descent in Distribution Space

The stepwise in gradient descent results from solving the following optimization problem, e.g., using line search:

\[ d^* = \arg \max_{d, \text{s.t.} \ KL(\pi_\theta \| \pi_\theta + d) \leq \epsilon} J(\theta + d) \]

\[ d^* = \arg \max_{\|d\| \leq \epsilon} J(\theta + d) \]

**SGD:** \[ \theta_{\text{new}} = \theta_{\text{old}} + d^* \]

**Euclidean distance in parameter space**

It is hard to predict the result on the parameterized distribution.. hard to pick the threshold epsilon

**Natural gradient descent:** the stepwise in parameter space is determined by considering the KL divergence in the distributions before and after the update:

\[ d^* = \arg \max_{d, \text{s.t.} \ KL(\pi_\theta \| \pi_\theta + d) \leq \epsilon} J(\theta + d) \]

**KL divergence in distribution space**

Easier to pick the distance threshold!!!

\[ D_{KL}(P \| Q) = \sum_i P(i) \log \left( \frac{P(i)}{Q(i)} \right) \]
\[ D_{KL}(P \| Q) = \int_{-\infty}^{\infty} p(x) \log \left( \frac{p(x)}{q(x)} \right) dx \]
Solving the KL Constrained Problem

Unconstrained penalized objective:

\[ d^* = \arg \max_d U(\theta + d) - \lambda (D_{KL} [\pi_\theta || \pi_{\theta+d}] - \epsilon) \]

First order Taylor expansion for the loss and second order for the KL:

\[ \approx \arg \max_d U(\theta_{old}) + \nabla_\theta U(\theta) \big|_{\theta=\theta_{old}} \cdot d - \frac{1}{2} \lambda (d^T \nabla^2_{KL} [\pi_{\theta_{old}} || \pi_\theta] \big|_{\theta=\theta_{old}} d) + \lambda \epsilon \]
Taylor expansion of KL

\[
D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \approx D_{KL}(p_{\theta_{old}} \mid p_{\theta_{old}}) + d^T \nabla_{\theta} D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} + \frac{1}{2} d^T \nabla^2_{\theta} D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} d
\]
Taylor expansion of KL

\[ D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \approx D_{KL}(p_{\theta_{old}} \mid p_{\theta_{old}}) + d^T \nabla_{\theta} D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} + \frac{1}{2} d^T \nabla^2_{\theta} D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} d \]

\[ D_{KL}(p_{\theta_{old}} \mid p_{\theta}) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{p_{\theta_{old}}(x)}{p_{\theta}(x)} \right) \]
Taylor expansion of KL

\[ D_{KL}(p_{\theta_{old}} | p_{\theta}) \approx D_{KL}(p_{\theta_{old}} | p_{\theta_{old}}) + d^T \nabla_{\theta} D_{KL}(p_{\theta_{old}} | p_{\theta}) |_{\theta = \theta_{old}} + \frac{1}{2} d^T \nabla^2_{\theta} D_{KL}(p_{\theta_{old}} | p_{\theta}) |_{\theta = \theta_{old}} d \]

\[ \nabla_{\theta} D_{KL}(p_{\theta_{old}} | p_{\theta}) |_{\theta = \theta_{old}} = - \nabla_{\theta} \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_{\theta}(x) |_{\theta = \theta_{old}} + \nabla_{\theta} \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_{\theta_{old}}(x) |_{\theta = \theta_{old}} \]

\[ D_{KL}(p_{\theta_{old}} | p_{\theta}) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{P_{\theta_{old}}(x)}{P_{\theta}(x)} \right) \]
Taylor expansion of KL

\[ D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \approx D_{KL}(p_{\theta_{old}} \mid p_{\theta_{old}}) + d^T \nabla_{\theta} D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} + \frac{1}{2} d^T \nabla_{\theta}^2 D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} d \]

\[ \nabla_{\theta} D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} = - \nabla_{\theta} \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_{\theta}(x) \mid_{\theta=\theta_{old}} + \nabla_{\theta} \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_{\theta_{old}}(x) \mid_{\theta=\theta_{old}} \]

\[ = - \mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_{\theta} \log P_{\theta}(x) \mid_{\theta=\theta_{old}} \]

\[ D_{KL}(p_{\theta_{old}} \mid p_{\theta}) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{P_{\theta_{old}}(x)}{P_{\theta}(x)} \right) \]
Taylor expansion of KL

\[ \begin{align*}
\text{D}_{\text{KL}}(p_{\theta_{\text{old}}} \mid p_{\theta}) & \approx \text{D}_{\text{KL}}(p_{\theta_{\text{old}}} \mid p_{\theta_{\text{old}}}) + d^T \nabla_{\theta} \text{D}_{\text{KL}}(p_{\theta_{\text{old}}} \mid p_{\theta}) \bigg|_{\theta = \theta_{\text{old}}} + \frac{1}{2} d^T \nabla^2_{\theta} \text{D}_{\text{KL}}(p_{\theta_{\text{old}}} \mid p_{\theta}) \bigg|_{\theta = \theta_{\text{old}}} d \\
\n\nabla_{\theta} \text{D}_{\text{KL}}(p_{\theta_{\text{old}}} \mid p_{\theta}) \bigg|_{\theta = \theta_{\text{old}}} & = -\nabla_{\theta} \mathbb{E}_{x \sim p_{\theta_{\text{old}}}} \log P_{\theta}(x) \bigg|_{\theta = \theta_{\text{old}}} + \nabla_{\theta} \mathbb{E}_{x \sim p_{\theta_{\text{old}}}} \log P_{\theta_{\text{old}}}(x) \bigg|_{\theta = \theta_{\text{old}}} \\
&= -\mathbb{E}_{x \sim p_{\theta_{\text{old}}}} \nabla_{\theta} \log P_{\theta}(x) \bigg|_{\theta = \theta_{\text{old}}} \\
&= -\mathbb{E}_{x \sim p_{\theta_{\text{old}}}} \frac{1}{P_{\theta_{\text{old}}}(x)} \nabla_{\theta} P_{\theta}(x) \bigg|_{\theta = \theta_{\text{old}}} \\
\text{D}_{\text{KL}}(p_{\theta_{\text{old}}} \mid p_{\theta}) & = \mathbb{E}_{x \sim p_{\theta_{\text{old}}}} \log \left( \frac{P_{\theta_{\text{old}}}(x)}{P_{\theta}(x)} \right) 
\end{align*} \]
Taylor expansion of KL

\[ D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \approx D_{KL}(p_{\theta_{old}} \mid p_{\theta_{old}}) + d^T \nabla_\theta D_{KL}(p_{\theta_{old}} \mid p_{\theta})_{\theta=\theta_{old}} + \frac{1}{2} d^T \nabla^2_\theta D_{KL}(p_{\theta_{old}} \mid p_{\theta})_{\theta=\theta_{old}} d \]

\[ \nabla_\theta D_{KL}(p_{\theta_{old}} \mid p_{\theta})_{\theta=\theta_{old}} = -\nabla_\theta \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_\theta(x)_{\theta=\theta_{old}} + \nabla_\theta \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_{\theta_{old}}(x)_{\theta=\theta_{old}} \]

\[ = -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_\theta \log P_\theta(x)_{\theta=\theta_{old}} \]

\[ = -\mathbb{E}_{x \sim p_{\theta_{old}}} \frac{1}{P_{\theta_{old}}(x)} \nabla_\theta P_\theta(x)_{\theta=\theta_{old}} \]

\[ = \int_x P_{\theta_{old}}(x) \frac{1}{P_{\theta_{old}}(x)} \nabla_\theta P_\theta(x)_{\theta=\theta_{old}} \]

\[ D_{KL}(p_{\theta_{old}} \mid p_{\theta}) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{P_{\theta_{old}}(x)}{P_\theta(x)} \right) \]
Taylor expansion of KL

\[
D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \approx D_{KL}(p_{\theta_{old}} \mid p_{\theta_{old}}) + d^T \nabla_\theta D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \bigg|_{\theta = \theta_{old}} + \frac{1}{2} d^T \nabla_\theta^2 D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \bigg|_{\theta = \theta_{old}} d
\]

\[
\nabla_\theta D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \bigg|_{\theta = \theta_{old}} = -\nabla_\theta \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_\theta(x) \bigg|_{\theta = \theta_{old}} + \nabla_\theta \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_{\theta_{old}}(x) \bigg|_{\theta = \theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_\theta \log P_\theta(x) \bigg|_{\theta = \theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \frac{1}{P_{\theta_{old}}(x)} \nabla_\theta P_\theta(x) \bigg|_{\theta = \theta_{old}}
\]

\[
= \int_x P_{\theta_{old}}(x) \frac{1}{P_{\theta_{old}}(x)} \nabla_\theta P_\theta(x) \bigg|_{\theta = \theta_{old}}
\]

\[
= \int_x \nabla_\theta P_\theta(x) \bigg|_{\theta = \theta_{old}}
\]

\[
D_{KL}(p_{\theta_{old}} \mid p_{\theta}) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{P_{\theta_{old}}(x)}{P_\theta(x)} \right)
\]
Taylor expansion of KL

\[
D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \approx D_{KL}(p_{\theta_{old}} \mid p_{\theta_{old}}) + d^T \nabla_{\theta} D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} + \frac{1}{2} d^T \nabla^2_{\theta} D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} d
\]

\[
\nabla_{\theta} D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \mid_{\theta=\theta_{old}} = -\nabla_{\theta} \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_{\theta}(x) \mid_{\theta=\theta_{old}} + \nabla_{\theta} \mathbb{E}_{x \sim p_{\theta_{old}}} \log P_{\theta_{old}}(x) \mid_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_{\theta} \log P_{\theta}(x) \mid_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \frac{1}{P_{\theta_{old}}(x)} \nabla_{\theta} P_{\theta}(x) \mid_{\theta=\theta_{old}}
\]

\[
= \int_x P_{\theta_{old}}(x) \frac{1}{P_{\theta_{old}}(x)} \nabla_{\theta} P_{\theta}(x) \mid_{\theta=\theta_{old}}
\]

\[
= \int_x \nabla_{\theta} P_{\theta}(x) \mid_{\theta=\theta_{old}}
\]

\[
= \nabla_{\theta} \int_x P_{\theta}(x) \mid_{\theta=\theta_{old}}.
\]

\[
D_{KL}(p_{\theta_{old}} \mid p_{\theta}) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{P_{\theta_{old}}(x)}{P_{\theta}(x)} \right)
\]

\[
= 0
\]
Taylor expansion of KL

\[ D_{KL}(p_{\theta_{old}} | p_{\theta}) \approx D_{KL}(p_{\theta_{old}} | p_{\theta_{old}}) + d^T \nabla_{\theta} KL(p_{\theta_{old}} | p_{\theta}) |_{\theta=\theta_{old}} + \frac{1}{2} d^T \nabla^2_{\theta} D_{KL}(p_{\theta_{old}} | p_{\theta}) |_{\theta=\theta_{old}} d \]

\[ \nabla^2_{\theta} D_{KL}(p_{\theta_{old}} | p_{\theta}) |_{\theta=\theta_{old}} = -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla^2_{\theta} \log P_{\theta}(x) |_{\theta=\theta_{old}} \]

\[ D_{KL}(p_{\theta_{old}} | p_{\theta}) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{P_{\theta_{old}}(x)}{P_{\theta}(x)} \right) \]
Taylor expansion of KL

\[ D_{KL}(p_{\theta_{old}} \mid p_\theta) \approx D_{KL}(p_{\theta_{old}} \mid p_{\theta_{old}}) + d^T \nabla_\theta KL(p_{\theta_{old}} \mid p_\theta) \mid_{\theta = \theta_{old}} + \frac{1}{2} d^T \nabla^2_\theta D_{KL}(p_{\theta_{old}} \mid p_\theta) \mid_{\theta = \theta_{old}} d \]

\[ \nabla^2_\theta D_{KL}(p_{\theta_{old}} \mid p_\theta) \mid_{\theta = \theta_{old}} = -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla^2_\theta \log P_\theta(x) \mid_{\theta = \theta_{old}} \]

\[ = -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_\theta \left( \frac{\nabla_\theta P_\theta(x)}{P_\theta(x)} \right) \mid_{\theta = \theta_{old}} \]

\[ D_{KL}(p_{\theta_{old}} \mid p_\theta) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{P_{\theta_{old}}(x)}{P_\theta(x)} \right) \]
Taylor expansion of KL

\[
D_{KL}(p_{\theta_{old}} | p_{\theta}) \approx D_{KL}(p_{\theta_{old}} | p_{\theta_{old}}) + d^T \nabla_{\theta} KL(p_{\theta_{old}} | p_{\theta}) |_{\theta=\theta_{old}} + \frac{1}{2} d^T \nabla^2_{\theta} D_{KL}(p_{\theta_{old}} | p_{\theta}) |_{\theta=\theta_{old}} d
\]

\[
\nabla^2_{\theta} D_{KL}(p_{\theta_{old}} | p_{\theta}) |_{\theta=\theta_{old}} = -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla^2_{\theta} \log P_{\theta}(x) |_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_{\theta} \left( \frac{\nabla_{\theta} P_{\theta}(x)}{P_{\theta}(x)} \right) |_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \left( \frac{\nabla^2_{\theta} P_{\theta}(x) P_{\theta}(x) - \nabla_{\theta} P_{\theta}(x) \nabla_{\theta} P_{\theta}(x)^T}{P_{\theta}(x)^2} \right) |_{\theta=\theta_{old}}
\]

\[
D_{KL}(p_{\theta_{old}} | p_{\theta}) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{P_{\theta_{old}}(x)}{P_{\theta}(x)} \right)
\]
Taylor expansion of KL

\[
D_{KL}(p_{\theta_{old}} \mid p_{\theta}) \approx D_{KL}(p_{\theta_{old}} \mid p_{\theta_{old}}) + d^T \nabla_\theta KL(p_{\theta_{old}} \mid p_{\theta})|_{\theta=\theta_{old}} + \frac{1}{2} d^T \nabla^2_\theta D_{KL}(p_{\theta_{old}} \mid p_{\theta})|_{\theta=\theta_{old}} d
\]

\[
\nabla^2_\theta D_{KL}(p_{\theta_{old}} \mid p_{\theta})|_{\theta=\theta_{old}} = -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla^2_\theta \log P_\theta(x) \mid_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_\theta \left( \frac{\nabla_\theta P_\theta(x)}{P_\theta(x)} \right) \mid_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \left( \frac{\nabla^2_\theta P_\theta(x) P_\theta(x) - \nabla_\theta P_\theta(x) \nabla_\theta P_\theta(x)^T}{P_\theta(x)^2} \right) \mid_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \frac{\nabla^2_\theta P_\theta(x) \mid_{\theta=\theta_{old}}}{P_{\theta_{old}}(x)} + \mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_\theta \log P_\theta(x) \nabla_\theta \log P_\theta(x)^T \mid_{\theta=\theta_{old}}
\]

\[
D_{KL}(p_{\theta_{old}} \mid p_{\theta}) = \mathbb{E}_{x \sim p_{\theta_{old}}} \log \left( \frac{P_{\theta_{old}}(x)}{P_\theta(x)} \right)
\]
Taylor expansion of KL

\[
D_{KL}(p_{\theta_{old}} | p_\theta) \approx D_{KL}(p_{\theta_{old}} | p_{\theta_{old}}) + d^T \nabla_\theta KL(p_{\theta_{old}} | p_\theta) |_{\theta=\theta_{old}} + \frac{1}{2} d^T \nabla^2_\theta D_{KL}(p_{\theta_{old}} | p_\theta) |_{\theta=\theta_{old}} d
\]

\[
\nabla^2_\theta D_{KL}(p_{\theta_{old}} | p_\theta) |_{\theta=\theta_{old}} = -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla^2_\theta \log P_\theta(x) |_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_\theta \left( \frac{\nabla_\theta P_\theta(x)}{P_\theta(x)} \right) |_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \left( \frac{\nabla^2_\theta P_\theta(x) P_\theta(x) - \nabla_\theta P_\theta(x) \nabla_\theta P_\theta(x)^T}{P_\theta(x)^2} \right) |_{\theta=\theta_{old}}
\]

\[
= -\mathbb{E}_{x \sim p_{\theta_{old}}} \frac{\nabla^2_\theta P_\theta(x) |_{\theta=\theta_{old}}}{P_{\theta_{old}}(x)} + \mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_\theta \log P_\theta(x) \nabla_\theta \log P_\theta(x)^T |_{\theta=\theta_{old}}
\]

\[
= \mathbb{E}_{x \sim p_{\theta_{old}}} \nabla_\theta \log P_\theta(x) \nabla_\theta \log P_\theta(x)^T |_{\theta=\theta_{old}}
\]
Fisher Information Matrix

Exactly equivalent to the Hessian of KL divergence!

\[
\mathbf{F}(\theta) = \mathbb{E}_\theta \left[ \nabla_\theta \log p_\theta(x) \nabla_\theta \log p_\theta(x)^\top \right]
\]

\[
\mathbf{F}(\theta_{\text{old}}) = \nabla^2_\theta \mathcal{D}_{\text{KL}}(p_{\theta_{\text{old}}} | p_\theta) |_{\theta=\theta_{\text{old}}}
\]

\[
\mathcal{D}_{\text{KL}}(p_{\theta_{\text{old}}} | p_\theta) \approx \mathcal{D}_{\text{KL}}(p_{\theta_{\text{old}}} | p_{\theta_{\text{old}}}) + d^\top \nabla_\theta \mathcal{D}_{\text{KL}}(p_{\theta_{\text{old}}} | p_\theta) |_{\theta=\theta_{\text{old}}} + \frac{1}{2} d^\top \nabla^2_\theta \mathcal{D}_{\text{KL}}(p_{\theta_{\text{old}}} | p_\theta) |_{\theta=\theta_{\text{old}}} d
\]

\[
= \frac{1}{2} d^\top \mathbf{F}(\theta_{\text{old}}) d
\]

\[
= \frac{1}{2} (\theta - \theta_{\text{old}})^\top \mathbf{F}(\theta_{\text{old}})(\theta - \theta_{\text{old}})
\]

Since KL divergence is roughly analogous to a distance measure between distributions, Fisher information serves as a local distance metric between distributions: how much you change the distribution if you move the parameters a little bit in a given direction.
Unconstrained penalized objective:

\[ d^\ast = \arg \max_d U(\theta + d) - \lambda (D_{KL}[\pi_\theta || \pi_{\theta+d}] - \epsilon) \]

First order Taylor expansion for the loss and second order for the KL:

\[
\approx \arg \max_d U(\theta_{old}) + \nabla_\theta U(\theta) \big|_{\theta=\theta_{old}} \cdot d - \frac{1}{2} \lambda (d^\top \nabla^2_{\theta} D_{KL} [\pi_{\theta_{old}} || \pi_\theta] \big|_{\theta=\theta_{old}} d) + \lambda \epsilon
\]

Substitute for the information matrix:

\[
= \arg \max_d \nabla_\theta U(\theta) \big|_{\theta=\theta_{old}} \cdot d - \frac{1}{2} \lambda (d^\top F(\theta_{old}) d)
\]

\[
= \arg \min_d - \nabla_\theta U(\theta) \big|_{\theta=\theta_{old}} \cdot d + \frac{1}{2} \lambda (d^\top F(\theta_{old}) d)
\]
Natural Gradient Descent

Setting the gradient to zero:

\[
0 = \frac{\partial}{\partial d} \left( -\nabla_{\theta} U(\theta) \big|_{\theta=\theta_{old}} \cdot d + \frac{1}{2} \lambda (d^T F(\theta_{old}) d) \right)
\]

\[
= -\nabla_{\theta} U(\theta) \big|_{\theta=\theta_{old}} + \frac{1}{2} \lambda (F(\theta_{old})) d
\]

\[
d = \frac{2}{\lambda} F^{-1}(\theta_{old}) \nabla_{\theta} U(\theta) \big|_{\theta=\theta_{old}}
\]

The natural gradient:

\[
\tilde{\nabla} J(\theta) = F^{-1}(\theta_{old}) \nabla_{\theta} J(\theta)
\]

\[
\theta_{\text{new}} = \theta_{\text{old}} + \alpha \cdot F^{-1}(\theta_{old}) \tilde{g}
\]

\[
D_{\text{KL}}(\pi_{\theta_{\text{old}}} \mid \pi_{\theta}) \approx \frac{1}{2} (\theta - \theta_{\text{old}})^T F(\theta_{\text{old}})(\theta - \theta_{\text{old}})
\]

\[
\frac{1}{2} (\alpha g_N)^T F(\alpha g_N) = \epsilon
\]

\[
\alpha = \sqrt{\frac{2\epsilon}{g_N^T F g_N}}
\]
Both use samples from the current policy $\pi_k = \pi(\theta_k)$

---

**Algorithm 1** Natural Policy Gradient

Input: initial policy parameters $\theta_0$

for $k = 0, 1, 2, ...$ do

- Collect set of trajectories $D_k$ on policy $\pi_k = \pi(\theta_k)$
- Estimate advantages $\hat{A}_{\pi_k}^t$ using any advantage estimation algorithm
- Form sample estimates for:
  - policy gradient $\hat{g}_k$ (using advantage estimates)
  - and KL-divergence Hessian / Fisher Information Matrix $\hat{H}_k$

Compute Natural Policy Gradient update:

$$\theta_{k+1} = \theta_k + \sqrt{\frac{2\mathcal{E}}{\hat{g}_k^T \hat{H}_k^{-1} \hat{g}_k}} \hat{H}_k^{-1} \hat{g}_k$$

end for
Algorithm 1 Natural Policy Gradient

**Input:** initial policy parameters \( \theta_0 \)

**for** \( k = 0, 1, 2, \ldots \) **do**

- Collect set of trajectories \( D_k \) on policy \( \pi_k = \pi(\theta_k) \)
- Estimate advantages \( \hat{A}_{\pi_k}^t \) using any advantage estimation algorithm
- Form sample estimates for
  - policy gradient \( \hat{g}_k \) (using advantage estimates)
  - and KL-divergence Hessian / Fisher Information Matrix \( \hat{H}_k \)

**Compute Natural Policy Gradient update:**

\[
\theta_{k+1} = \theta_k + \sqrt{\frac{2 \epsilon}{\hat{g}_k^T \hat{H}_k^{-1} \hat{g}_k}} \hat{H}_k^{-1} \hat{g}_k
\]

**end for**

very expensive to compute for a large number of parameters!
Monte Carlo Policy Gradients (REINFORCE), gradient direction: \[ \hat{g} = \hat{E}_t \left[ \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{A}_t \right] \]

Actor-Critic Policy Gradient: \[ \hat{g} = \hat{E}_t \left[ \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) A_w(s_t) \right] \]

1. Collect trajectories for policy \( \pi_{\theta_{\text{old}}} \)
2. Estimate advantages \( A \)
3. Compute policy gradient \( \hat{g} \)
4. Update policy parameters \( \theta_{\text{new}} = \theta_{\text{old}} + \epsilon \cdot \hat{g} \)
5. GOTO 1
1. Collect trajectories for policy $\pi_{\theta_{old}}$
2. Estimate advantages $A$
3. Compute policy gradient $\hat{g}$
4. Update policy parameters $\theta_{new} = \theta_{old} + \epsilon \cdot \hat{g}$
5. GOTO 1

- On policy learning can be extremely inefficient
- The policy changes only a little bit with each gradient step
- I want to be able to use earlier data..how to do that?
Off-policy learning with Importance Sampling

\[ U(\theta) = \mathbb{E}_{\tau \sim \pi_\theta(\tau)} [R(\tau)] \]
\[ = \sum_{\tau} \pi_\theta(\tau)R(\tau) \]
\[ = \sum_{\tau} \pi_{\theta_{old}}(\tau) \frac{\pi_\theta(\tau)}{\pi_{\theta_{old}}(\tau)} R(\tau) \]
\[ = \mathbb{E}_{\tau \sim \pi_{\theta_{old}}} \frac{\pi_\theta(\tau)}{\pi_{\theta_{old}}(\tau)} R(\tau) \]

\[ \nabla_\theta U(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta_{old}}} \frac{\nabla_\theta \pi_\theta(\tau)}{\pi_{\theta_{old}}(\tau)} R(\tau) \]

\[ \nabla_\theta U(\theta) \big|_{\theta=\theta_{old}} = \mathbb{E}_{\tau \sim \pi_{\theta_{old}}} \nabla_\theta \log \pi_\theta(\tau) \big|_{\theta=\theta_{old}} R(\tau) \]

<-Gradient evaluated at theta_old is unchanged
Off policy learning with Importance Sampling

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

\[ = \sum_{\tau} \pi_{\theta}(\tau) R(\tau) \]

\[ = \sum_{\tau} \frac{\pi_{\theta}(\tau)}{\pi_{\theta_{old}}(\tau)} \pi_{\theta_{old}}(\tau) R(\tau) \]

\[ = \sum_{\tau \sim \pi_{\theta_{old}}} \pi_{\theta}(\tau) \pi_{\theta_{old}}(\tau) \]

\[ = \mathbb{E}_{\tau \sim \pi_{\theta_{old}}} \frac{\pi_{\theta}(\tau)}{\pi_{\theta_{old}}(\tau)} R(\tau) \]

\[ \nabla_{\theta} U(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta_{old}}} \frac{\nabla_{\theta} \pi_{\theta}(\tau)}{\pi_{\theta_{old}}(\tau)} R(\tau) \]

\[ \nabla_{\theta} U(\theta) \big|_{\theta = \theta_{old}} = \mathbb{E}_{\tau \sim \pi_{\theta_{old}}} \nabla_{\theta} \log \pi_{\theta}(\tau) \big|_{\theta = \theta_{old}} R(\tau) \]
Define the constrained objective:

\[
\begin{align*}
\text{maximize} & \quad \hat{E}_t \left[ \frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)} \hat{A}_t \right] \\
\text{subject to} & \quad \hat{E}_t [\text{KL}[\pi_{\theta_{\text{old}}} (\cdot | s_t), \pi_\theta (\cdot | s_t)]] \leq \delta.
\end{align*}
\]

- Also worth considering using a penalty instead of a constraint

\[
\begin{align*}
\text{maximize} & \quad \hat{E}_t \left[ \frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)} \hat{A}_t \right] - \beta \hat{E}_t [\text{KL}[\pi_{\theta_{\text{old}}} (\cdot | s_t), \pi_\theta (\cdot | s_t)]]
\end{align*}
\]

Again the KL penalized problem!


Further Reading
- S. Kakade. “A Natural Policy Gradient.” NIPS. 2001
- blog.openai.com: recent post sobase line sr releases

Again the KL penalized problem!
Police gradients: have a function approximation for the policy \( \pi_\theta(u|x) \) and optimize using SGD. SGD is sufficient to learn great object object detectors for example. What is different in RL?

- Non-stationarity in RL: Each time the policy changes the state visitation distribution changes. And this can cause the policy to diverge!

- Contribution: theoretical and practical method of how big of a step our gradient can take.
maximize $\theta \ L_{\pi_{\theta_{old}}}(\pi_{\theta}) - \beta \cdot \overline{KL}_{\pi_{\theta_{old}}}(\pi_{\theta})$

Make linear approximation to $L_{\pi_{\theta_{old}}}$ and quadratic approximation to KL term:

$\max_{\theta} \ g \cdot (\theta - \theta_{old}) - \frac{\beta}{2}(\theta - \theta_{old})^T F(\theta - \theta_{old})$

where $g = \frac{\partial}{\partial \theta} L_{\pi_{\theta_{old}}}(\pi_{\theta})|_{\theta=\theta_{old}}$, $F = \frac{\partial^2}{\partial^2 \theta} \overline{KL}_{\pi_{\theta_{old}}}(\pi_{\theta})|_{\theta=\theta_{old}}$

Exactly what we saw with natural policy gradient!
One important detail!
Due to the quadratic approximation, the KL constraint may be violated! What if we just do a line search to find the best stepsize, making sure:

- I am improving my objective $J(\theta)$
- The KL constraint is not violated!

\[
\begin{align*}
\text{maximize} & \quad \mathbb{E}_t \left[ \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta_{old}}(a_t | s_t)} \hat{A}_t \right] \\
\text{subject to} & \quad \mathbb{E}_t [KL(\pi_{\theta_{old}}(\cdot | s_t), \pi_{\theta_{old}}(\cdot | s_t))] \leq \delta.
\end{align*}
\]

**Algorithm 2 Line Search for TRPO**

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

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

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

  if $L_{\theta_k}(\theta) \geq 0$ and $\tilde{D}_{KL}(\theta || \theta_k) \leq \delta$ then

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

    break

  end if

end for
Trust region Policy Optimization

TRPO= NPG +Linesearch

Algorithm 3 Trust Region Policy Optimization

Input: initial policy parameters $\theta_0$

for $k = 0, 1, 2, \ldots$ do

Collect set of trajectories $\mathcal{D}_k$ on policy $\pi_k = \pi(\theta_k)$

Estimate advantages $\hat{A}_{\pi_k}^t$ using any advantage estimation algorithm

Form sample estimates for

- policy gradient $\hat{g}_k$ (using advantage estimates)
- and KL-divergence Hessian-vector product function $f(\nu) = \hat{H}_k \nu$

Use CG with $n_{cg}$ iterations to obtain $x_k \approx \hat{H}^{-1}_k \hat{g}_k$

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
Algorithm 3 Trust Region Policy Optimization

Input: initial policy parameters $\theta_0$

for $k = 0, 1, 2, \ldots$ do
  Collect set of trajectories $D_k$ on policy $\pi_k = \pi(\theta_k)$
  Estimate advantages $\hat{A}^{\pi_k}_t$ 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
Proximal Policy Optimization

Can I achieve similar performance without second order information (no Fisher matrix!)

- **Adaptive KL Penalty**
  - Policy update solves unconstrained optimization problem
    \[ \theta_{k+1} = \arg\max_{\theta} \mathcal{L}_{\theta_k}(\theta) - \beta_k \tilde{D}_{KL}(\theta||\theta_k) \]
  - Penalty coefficient \( \beta_k \) changes between iterations to approximately enforce KL-divergence constraint

- **Clipped Objective**
  - New objective function: let \( r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_k}(a_t|s_t) \). Then
    \[
    \mathcal{L}^{CLIP}_{\theta_k}(\theta) = \mathbb{E}_{\tau \sim \pi_k} \left[ \sum_{t=0}^{T} \min(r_t(\theta)\hat{A}^{\pi_k}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}^{\pi_k}_t) \right]
    \]
  - where \( \epsilon \) is a hyperparameter (maybe \( \epsilon = 0.2 \))
  - Policy update is \( \theta_{k+1} = \arg\max_{\theta} \mathcal{L}^{CLIP}_{\theta_k}(\theta) \)

---

PPO: Adaptive KL Penalty

Input: initial policy parameters $\theta_0$, initial KL penalty $\beta_0$, target KL-divergence $\delta$

for $k = 0, 1, 2, \ldots$ do

Collect set of partial trajectories $\mathcal{D}_k$ on policy $\pi_k = \pi(\theta_k)$

Estimate advantages $\hat{A}^\pi_t$ using any advantage estimation algorithm

Compute policy update

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

by taking $K$ steps of minibatch SGD (via Adam)

if $\bar{D}_{KL}(\theta_{k+1}||\theta_k) \geq 1.5\delta$ then

$\beta_{k+1} = 2\beta_k$

else if $\bar{D}_{KL}(\theta_{k+1}||\theta_k) \leq \delta/1.5$ then

$\beta_{k+1} = \beta_k/2$

end if

end for

Don’t use second order approximation for KL which is expensive, use standard gradient descent
PPO: Clipped Objective

- Recall the surrogate objective

\[
L^{IS}(\theta) = \hat{\mathbb{E}}_t \left[ \frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{old}}(a_t | s_t)} \hat{A}_t \right] = \hat{\mathbb{E}}_t \left[ r_t(\theta) \hat{A}_t \right].
\]  

- Form a lower bound via clipped importance ratios

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

---

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

**Figure:** Various objectives as a function of interpolation factor $\alpha$ between $\theta_{k+1}$ and $\theta_k$ after one update of PPO-Clip \(^9\)
PPO: Clipped Objective

Input: initial policy parameters $\theta_0$, clipping threshold $\epsilon$

for $k = 0, 1, 2, \ldots$ do

Collect set of partial trajectories $D_k$ on policy $\pi_k = \pi(\theta_k)$
Estimate advantages $\hat{A}_{\pi_k}^t$ using any advantage estimation algorithm
Compute policy update

\[
\theta_{k+1} = \arg\max_{\theta} L_{\theta_k}^{CLIP}(\theta)
\]

by taking $K$ steps of minibatch SGD (via Adam), where

\[
L_{\theta_k}^{CLIP}(\theta) = \mathbb{E}_{\tau \sim \pi_k} \left[ \sum_{t=0}^{T} \min(r_t(\theta)\hat{A}_{\pi_k}^t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{\pi_k}^t) \right]
\]

end for

- Clipping prevents policy from having incentive to go far away from $\theta_{k+1}$
- Clipping seems to work at least as well as PPO with KL penalty, but is simpler to implement
PPO: Clipped Objective

Figure: Performance comparison between PPO with clipped objective and various other deep RL methods on a slate of MuJoCo tasks.  

Schulman, Wolski, Dhariwal, Radford, Klimov, 2017
Summary

• Gradient Descent in Parameter VS distribution space
• Natural gradients: we need to keep track of how the KL changes from iteration to iteration
• Natural policy gradients
• Clipped objective works well

Further Reading

I S. Kakade. “A Natural Policy Gradient.” NIPS. 2001