Policy Search Methods
(CSC2547: Introduction to Reinforcement Learning)

Amir-massoud Farahmand

University of Toronto & Vector Institute
## Table of Contents

1. Introduction
   - Policy Parametrization
   - Performance Measure

2. Zero-Order Methods
   - Finite Policy Parameter Space
   - Random Search
   - Evolutionary Algorithms

3. First-Order Methods and the Policy Gradient
   - Finite Difference Approximation of the Policy Gradient
   - Policy Gradient for the Immediate Reward Problem
     - Variance Reduction – Randomness of States
     - Variance Reduction – Randomness of Actions
   - Policy Gradient for Continuing Tasks
So far, we have described methods for computing the optimal policy based on the computation of the value function. Only the value function was explicitly represented.

The policy could be computed based on it.

There are methods based on explicit representation of the policy and optimizing the performance of the agent by searching in the space of policies.

We call them policy search methods.

Hybrid methods: explicit representation of both value and policy.
Consider a stochastic policy $\pi_\theta : \mathcal{X} \to \mathcal{M}(\mathcal{A})$ that is parameterized by a $\theta \in \Theta$.

- The set $\Theta$ is the parameter space, e.g., a subset of $\mathbb{R}^p$.
- The space of all parameterized policies:

$$\Pi_\Theta = \{ \pi_\theta : \mathcal{X} \to \mathcal{M}(\mathcal{A}) : \theta \in \Theta \} . \quad (1)$$

- This space depends on the mapping $\pi_\theta$ and $\Theta$. 

Many choices for how we can parameterize a policy $\pi_\theta$.

A generic example is based on the Boltzmann (or softmax) distribution.

Given a function $f_\theta : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{R}$ (e.g., a DNN or decision tree parameterized by $\theta$), the density of choosing action $a$ at state $x$ is

$$
\pi_\theta(a|x) = \frac{\exp(f_\theta(x, a))}{\int \exp(f_\theta(x, a')) da'}.
$$
Policy Parametrization: Examples

- A special case would be when \( f_{\theta}(x, a) = \phi(x, a)^\top \theta \) for some features \( \phi : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{R}^p \) and \( \theta \in \mathbb{R}^p \):

  \[
  \pi_{\theta}(a|x) = \frac{\exp(\phi(x, a)^\top \theta)}{\int \exp(\phi(x, a')^\top \theta) da'}.
  \]

- When the action space \( \mathcal{A} \) is discrete, \( \pi_{\theta}(a|x) \) denotes the probability of choosing action \( a \) at state \( x \) (instead of its density):

  \[
  \pi_{\theta}(a|x) = \frac{\exp(\phi(x, a)^\top \theta)}{\sum_{a' \in \mathcal{A}} \exp(\phi(x, a')^\top \theta)}.
  \]
Another example: $\pi_\theta(\cdot|x)$ defining a Normal distribution over action space with $\theta$ parameterization its mean and covariance:

$$\pi_\theta(\cdot|x) = \mathcal{N}(\mu_\theta(x), \Sigma_\theta(x)).$$

If the action space is $d_\mathcal{A}$-dimensional:

- **Mean**: $\mu_\theta : \mathcal{X} \rightarrow \mathbb{R}^{d_\mathcal{A}}$
- **Covariance**: $\Sigma_\theta : \mathcal{X} \rightarrow S^{d_\mathcal{A}}_{++}$. Here $S^{d_\mathcal{A}}_{++}$ refers to the set of $d_\mathcal{A} \times d_\mathcal{A}$ positive semi-definite matrices.
Ease of Work in Continuous Action Spaces

- Explicit parameterization of policy allows us to easily choose a continuous action.
- For value-based methods, this can be challenging:
  - Even if we know $Q^*$, computing the optimal policy $\pi^*(x) = \pi_g(x; Q^*)$ requires an optimization problem in $A$.
  - This is challenging if $A$ is a high-dimensional space.
  - VI and PI requires repeated calculation of the greedy policy.
- Sure, action selection might be easy!
- **Question:** How can we optimize the performance of a parametrized policy?
The performance can be measured in various ways.

We focus on the expected return of following $\pi_\theta$, the value function.

- We can also incorporate the variance or some other risk measures, relatively easily.

**Goal:** Find a policy that maximizes this performance measure.

We are restricted to choosing policies within $\Pi_\Theta$ (1).
Assume that we only care about the performance at state $x \in X$.

The goal of policy search:

$$\arg\max_{\pi \in \Pi_\Theta} V^\pi(x) = \arg\max_{\theta \in \Theta} V^{\pi_\theta}(x).$$

**Interpretation:** We are interested in finding a policy that if the agent starts at this particular state $x$, its performance, measured according to its expected return, is maximized.
Performance Measure on a Single State

\[
\arg\max_{\pi \in \Pi_\Theta} V^\pi(x) = \arg\max_{\theta \in \Theta} V^{\pi_\theta}(x).
\]

- The optimal policy \( \pi^* \) not only maximizes the value function at this particular \( x \), but also at any other \( x' \in \mathcal{X} \).
- But the optimal policy may not be in \( \Pi_\Theta \).
- If \( \pi^* \notin \Pi_\Theta \), we will not be able to find a policy that maximizes the value at all states.
  - In that case, we may want to find a policy that is only good at our starting state \( x \), and ignore the performance at other states.
  - The obtained policy is going to be initial-state-dependent. If we change \( x \) to another state \( x' \neq x \), the optimal policy within \( \Pi_\Theta \) might change.
Performance Measure with $X_1 \sim \rho$

- Instead of the extreme case of considering a single initial state $x$, we can consider when the initial state is distributed according to some distribution $\rho \in \mathcal{M}(\mathcal{X})$.

- The performance measure would be the average of following $\pi_\theta$ with the initial state $X_1 \sim \rho$.

$$J(\pi_\theta) = J_\rho(\pi_\theta) \triangleq \int V^{\pi_\theta}(x)d\rho(x) = \mathbb{E}_{X \sim \rho}[V^{\pi_\theta}(X)]. \quad (3)$$
Performance Measure with $X_1 \sim \rho$

$$J(\pi_\theta) = J_\rho(\pi_\theta) \triangleq \int V^{\pi_\theta}(x) d\rho(x) = \mathbb{E}_{X \sim \rho} [V^{\pi_\theta}(X)].$$

- The optimal policy maximizes $J_\rho$.
- $J_\rho(\pi^*) \geq J_\rho(\pi_\theta)$ for any $\pi_\theta \in \Pi_\Theta$.
- If $\pi^* \notin \Pi_\Theta$, the inequality is strict.
Performance Measure with $X_1 \sim \rho$

$$J(\pi_\theta) = J_\rho(\pi_\theta) \triangleq \int V^{\pi_\theta}(x) d\rho(x) = \mathbb{E}_{X \sim \rho} [V^{\pi_\theta}(X)].$$

- In policy search methods, we aim to find the maximizer of the performance measure within $\Pi_\Theta$.
  $$\bar{\pi} \leftarrow \arg\max_{\pi_\theta \in \Pi_\Theta} J_\rho(\pi_\theta). \quad (4)$$

- The corresponding policy is denoted by $\bar{\theta}$, i.e., $\bar{\pi} = \pi_{\bar{\theta}}$.

- For different $\rho$, we may get different optimizers.

- To emphasize the dependence of the maximizer on $\rho$, we may use $\bar{\pi}_\rho$.

- We may sometimes denote $J(\pi_\theta)$ or $J_\rho(\pi_\theta)$ simply by $J_\rho(\theta)$. 
Question: How can we solve the optimization problem (4) to find $\pi_\theta$ that maximizes the performance measure $J_\rho$?

This is an optimization problem, so we can benefit from the arsenal of optimization algorithms.

Being an RL problem, however, brings both challenges and opportunities.

- **Challenge**: The value of $J_\rho$ is not readily available, and has to be estimated through interaction with the environment.
- **Opportunity**: The special structure of the RL problem, such as the value function satisfying the Bellman equation.
Policy Search Methods

Introduction

Performance Measure

Policy Search as an Optimization Problem

- Optimization methods, broadly speaking, can be categorized based on the information they need about their objective.
- **Zero-order** methods only use the value of the objective at various query points.
  - They compute $J_\rho(\theta)$ at various $\theta$s in order to guide the optimization process.
- **First-order** methods use the derivative of the objective instead of, or in addition to, the value of the objective.
  - They use $\nabla_\theta J_\rho(\theta)$ in order to guide the search.
  - The quintessential first-order optimization method is the gradient descent (and its stochastic variant).
Zero-Order Methods

- We first consider the case when the policy parameter space $\Theta$ is finite.
- This helps us understand some of the challenges.
- We then extend our discussion to case when $\Theta$ is continuous function space.
Zero-Order Methods: Finite Policy Parameter Space

- Assume that we are given a finite $\Theta = \{\theta_1, \ldots, \theta_m\}$ policy parameters.
- This defines the finite policy space $\Pi_\Theta = \{\pi_\theta : \theta \in \Theta\}$.
- Find the policy $\pi_\theta \in \Pi_\Theta$ such that $J_\rho(\pi_\theta)$ is maximized (4).
- If we can easily compute $J_\rho(\pi_\theta)$ for each $\theta \in \Theta$, this is an easy problem, at least in principle.
- So the main issue is to compute $J_\rho(\pi_\theta)$. 
Zero-Order Methods: Finite Policy Parameter Space

- The performance measure $J_\rho(\pi_\theta) = \mathbb{E}_{X \sim \rho} [V^{\pi_\theta}(X)]$, i.e., the expectation of $V^{\pi_\theta}(X)$ w.r.t. $X \sim \rho$.

- We can try to compute $V^{\pi_\theta}(x)$ for all $x \in \mathcal{X}$, using any of the PE methods that we have developed, and take the weighted average according to $\rho$.

- If $\mathcal{X}$ is discrete, this would be

  $$J_\rho(\pi_\theta) = \sum_{x \in \mathcal{X}} \rho(x) V^{\pi_\theta}(x).$$

- If $\mathcal{X}$ is large:
  - Computing $V^{\pi_\theta}$ itself is not going to be easy.
  - Computing the integral $\int V^{\pi_\theta}(x) d\rho(x)$ is going to be challenging.
Alternative: Computing an unbiased estimate of $J_\rho(\pi_\theta)$ instead, using MC estimation.

We derive this in two steps.

- Assume that we know $V^{\pi_\theta}$, estimate $J_\rho(\pi_\theta)$.
- Replace $V^{\pi_\theta}(x)$ with the return $G^{\pi_\theta}(x)$. 
Zero-Order Methods: Finite Policy Parameter Space

- We assume that we know $V^{\pi_\theta}$, and we want to estimate $J_\rho(\pi_\theta)$.

- If we sample $X \sim \rho$, we have that $V^{\pi_\theta}(X)$ is an unbiased estimate of $J_\rho(\rho_\pi)$ as

$$\mathbb{E} [V^{\pi_\theta}] = \int V^{\pi_\theta}(x) d\rho(x) = J_\rho(\pi_\theta).$$

- If we draw $n$ independent samples $X_1, \ldots, X_n \sim \rho$, the estimator

$$\frac{1}{n} \sum_{i=1}^{n} V^{\pi_\theta}(X_i)$$

would be an unbiased as well.
Zero-Order Methods: Finite Policy Parameter Space

\[ \frac{1}{n} \sum_{i=1}^{n} V^{\pi_\theta}(X_i) \]

- Variance:
  \[
  \text{Var} \left[ V^{\pi_\theta}(X) \right] \cdot \frac{n}{n}
  \]

  - This variance goes to 0 as \( n \) increases.
  - The variance \( \text{Var} \left[ V^{\pi_\theta}(X) \right] \) is a measure of dispersion of the value function across states samples according to \( \rho \).
    - If the value function is constant, it will be zero.
    - If it is changing slowly as a function of the state, it would be small.
    - If the value function varies greatly, the variance is large.
  - The variance is a function of the policy \( \pi_\theta \), so for each \( \pi_\theta \in \Pi_\theta \), we get a different variance.
The second step is to replace $V^{\pi_\theta}(x)$ with the return $G^{\pi_\theta}(x)$.

The return $G^{\pi_\theta}(x)$ is an unbiased estimate of $V^{\pi_\theta}(x)$.

Computation of $G^{\pi_\theta}(x)$ requires starting the agent from state $x$ and following $\pi_\theta$ (i.e., performing a rollout from $x$) until the end of episode for episodic tasks, or until infinity for continual tasks.

If $X \sim \rho$, $G^{\pi_\theta}(X)$ is an unbiased estimate of $J_\rho(\pi_\theta)$ as

$$
\mathbb{E}_{X \sim \rho} [G^{\pi_\theta}(X)] = \mathbb{E}_{X \sim \rho} [\mathbb{E} [G^{\pi_\theta}(X) | X]] \\
= \mathbb{E}_{X \sim \rho} [V^{\pi_\theta}(X)] = J_\rho(\pi_\theta).
$$
If we draw \( n \) independently selected \( X_1, \ldots, X_n \sim \rho \), we can form

\[
\hat{J}_n(\pi_\theta) = \frac{1}{n} \sum_{i=1}^{n} G^{\pi_\theta}(X_i),
\]

as an unbiased estimate of \( J_\rho(\pi_\theta) \).
Zero-Order Methods: Finite Policy Parameter Space

Proposition

The estimator \( \hat{J}_n(\pi_\theta) \) (5) is an unbiased estimator for \( J_\rho(\pi_\theta) \) and has the variance of

\[
\text{Var} \left[ \hat{J}_n(\pi_\theta) \right] = \frac{1}{n} \left( \mathbb{E} \left[ \text{Var} \left[ G^{\pi_\theta}(X) \mid X \right] \right] + \text{Var} \left[ V^{\pi_\theta}(X) \right] \right).
\]

- If we have a finite number of parameters in \( \Theta \), we can estimate \( J_\rho(\pi_{\theta_i}) \approx \hat{J}_n(\pi_{\theta_i}) \pm O_P\left(\frac{1}{\sqrt{n}}\right) \) for each \( \theta_i \in \Theta \).
- We can use these estimates to select the best among them:

\[
\hat{\pi} = \pi_{\hat{\theta}} \leftarrow \arg\max_{\theta \in \Theta} \hat{J}_n(\pi_\theta). \quad (6)
\]
As there is an $O_P\left(\frac{1}{\sqrt{n}}\right)$ error in estimation of each $J_\rho(\pi_\theta)$, the selected policy $\hat{\pi}$ may not be the same as the maximizer $\bar{\pi}$ of (4).

The error can happen if $\hat{J}_n(\hat{\pi}) > \hat{J}_n(\bar{\pi})$ (which leads to preferring $\hat{\pi}$ to $\bar{\pi}$ according to the empirical performance measure) even though $J_\rho(\hat{\pi}) < J_\rho(\bar{\pi})$. 
Even if we make an error in selecting the best policy, the gap in their performance is within $O_P\left(\frac{1}{\sqrt{n}}\right)$.

As we increase $n$, the error in estimating $J_\rho(\pi_\theta)$ decreases and the probability of selecting an optimal policy increases.

This increased accuracy, however, is at the cost of increased sample and computational complexity, which would be $n|\Theta|$ rollouts.
Zero-Order Methods: Finite Policy Parameter Space

Proposition

Consider $\hat{\pi} = \pi_{\hat{\theta}} \leftarrow \arg\max_{\theta \in \Theta} \hat{J}_n(\pi_\theta)$ (6). Assume that the returns $G^{\pi_\theta}(x)$ are all $Q_{\text{max}}$-bounded for any $\theta \in \Theta$ and $x \in \mathcal{X}$. Furthermore, suppose that $|\Theta| < \infty$. For any $\delta > 0$, we have that

$$J_\rho(\hat{\theta}) \geq \max_{\theta \in \Theta} J_\rho(\theta) - 2Q_{\text{max}} \sqrt{\frac{2 \ln \left( \frac{2|\Theta|}{\delta} \right)}{n}},$$

with probability at least $1 - \delta$. 
Zero-Order Methods: Random Search

- If $\Theta$ is not finite, we cannot evaluate $\hat{J}_n(\pi_\theta)$ for all $\theta \in \Theta$.
- There are several generic methods for searching in a large parameter space:
  - Random Search (RS)
  - Simulated Annealing
  - Various evolutionary algorithms
Zero-Order Methods: Random Search

- We randomly pick $m$ policy parameters $\theta_1, \ldots, \theta_m \in \Theta$.
- Evaluate $\hat{J}_n(\pi_{\theta_i})$
- Pick the one with the highest value.
- Intuition of why this works:
  - With large enough $m$, one of $\theta_i$ might hit close to the optimal
    \[
    \hat{\theta} \leftarrow \arg\max_{\theta \in \Theta} \hat{J}_n(\pi_{\theta}).
    \]
  - If $n$ is large enough, the difference between $\hat{J}_n(\theta)$ and $J_\rho(\theta)$ would be small for all randomly selected $\theta$. 
Zero-Order Methods: Random Search

**Require:** Distribution $\nu \in \mathcal{M}(\Theta)$; Number of rollouts $n$; Maximum number of iterations $K$

1: Draw a parameter $\theta_1 = \theta'_1 \sim \nu$
2: Evaluate $\hat{J}_n(\pi_{\theta_1})$
3: for $k = 2, 3, \ldots, K$ do
4: Draw a parameter $\theta'_k \sim \nu$
5: Evaluate $\hat{J}_n(\pi_{\theta'_k})$
6: if $\hat{J}_n(\pi_{\theta'_k}) > \hat{J}_n(\pi_{\theta_k})$ then
7: $\theta_k \leftarrow \theta'_k$
8: else
9: $\theta_k \leftarrow \theta_{k-1}$
10: end if
11: end for
12: return $\pi_{\theta_K}$
Zero-Order Methods: Random Search

- We can provide guarantee that RS finds the optimal point, asymptotically.
- RS is not the most efficient way to search a parameter space.
- The way it is presented here does not benefit from all previous evaluation of the function when suggesting a new $\theta'_{k}$.
- That knowledge can be useful by helping us focus on more promising regions of the search space, instead of blindly sampling from the same distribution $\nu$.
- This can be achieved by adaptively changing the distribution $\nu_{k}$ to be a function of previous evaluations.
A large class of optimization methods are inspired by the process of evolution.

Heritable characteristics of individuals in a population change over generations due to processes such as natural selection.

The evolution leads to the adaptation of individuals, which means that they become better to live in their habitat.
Evolutionary Process for Optimization

- Identifying a solution to an optimization problem as an individual in a population
- The value of the function to be optimized for a particular solution as the fitness of that individual
- Emulate the evolution:
  - Mutation
  - Reproduction (cross-over)
  - Selection
- There are many variations in how we can do this:
  - Genetic Algorithms
  - Genetic Programming
  - Evolutionary Strategy
Evolutionary Strategy (ES) \((1 + 1)\)

**Require:** Initial point \(\theta_0 \in \Theta\); Rollouts \(n\); Iterations \(K\)

**Require:** Initial standard deviation of mutation operator: \(\sigma_1 > 0\)

**Require:** Adaptation parameters: \(c_+ > 0\) and \(c_- < 0\).

1: Evaluate \(\hat{J}_n(\pi_{\theta_0})\)
2: for \(k = 1, 2, \ldots, K\) do
3: Draw a perturbation \(\eta \sim \mathcal{N}(0, I)\)
4: \(\theta'_k \leftarrow \theta_k + \sigma_k \eta\) \hspace{1cm} ▷ Mutation
5: Evaluate \(\hat{J}_n(\pi_{\theta'_k})\)
6: if \(\hat{J}_n(\pi_{\theta'_k}) > \hat{J}_n(\pi_{\theta_k})\) then \hspace{1cm} ▷ Selection
7: \(\theta_{k+1} \leftarrow \theta'_k\)
8: \(\sigma_{k+1} \leftarrow \sigma_k e^{c+}\)
9: else
10: \(\theta_{k+1} \leftarrow \theta_{k-1}\)
11: \(\sigma_{k+1} \leftarrow \sigma_k e^{c-}\)
12: end if
13: end for
14: return \(\pi_{\theta_K}\)
Evolutionary Strategy (ES) \((1 + 1)\) and Beyond

- **Evolutionary Strategy (ES) \((1 + 1)\)** is one of the simplest evolutionary algorithms
  - Similar to RS, but guided choice of randomness
  - Has some theoretical analysis

- A modification of this algorithm is called ES\((1, \lambda)\) with \(\lambda > 1\) being an integer number.
  - The parent \(\theta_k\) generates \(\lambda\) offsprings:
    \[
    \theta'_{k, j} = \theta_k + \sigma_k \eta_j, \quad j = 1, \ldots, \lambda.
    \]
  - The competition would only be between the offsprings \(\{\theta'_{k, j}\}_{j=1}^{\lambda}\), and not with the parent. Only one of the \(\lambda\) offsprings gets to the next generation.
ES does not have any sexual reproduction.

There are other evolutionary algorithms that have the reproduction component too, e.g., Genetic Algorithm (GA).

Evolutionary algorithms can be quite complicated algorithms.

- Many heuristics, inspired by nature.
- Their performance is often evaluated only empirically.
- Analyzing them theoretically can be quite complicated.
- Current available results are limited to simple algorithms, such as ES(1 + 1), which may not be the best performing ones in practice.
Evolutionary Algorithms and RL

- Studying evolutionary algorithms to solve RL problems is a niche area in the RL community.
- Sometimes (often?), they are not considered as a part of the RL research proper.
- Knowing about them is useful!
- Both evolution and learning have been crucial adaptation mechanisms to get to the point where we have relatively smart species.
- Building AI agents with comparable capabilities to animals may require borrowing ideas from both learning and evolution.

- Learning: Within the lifespan of the agent
- Evolution: Across generations of the agents
First-Order Methods and the Policy Gradient

- The gradient of $J_\rho(\pi_\theta)$ w.r.t. $\theta$ allows us to design first-order optimization methods.
- Potentially more efficient in finding an optimum of the performance compared to zero-order methods.
- Not obvious how to compute the gradient:
  - The performance $J_\rho(\pi_\theta)$ depends on $V^{\pi_\theta}$.
  - Not a simple function of $\pi_\theta$.
  - The value function is a complicated function of the policy, reward distribution $\mathcal{R}$ and the transition dynamics $\mathcal{P}$.
Use Finite Difference (FD) approximation of the policy gradient.

Can be computed using the value of the performance objective itself.

Recall that given a function $f : \mathbb{R} \rightarrow \mathbb{R}$, the FD approximation of the derivative $f'(x) = \frac{df}{dx}(x)$ is

$$f'_{\text{FD}}(x) = \frac{f(x + \Delta x) - f(x)}{\Delta x}, \quad (7)$$

where $\Delta x$ is a small number. This is called the forward difference approximation.
By the Taylor’s theorem, assuming twice differentiability, we have
\[ f(x + \Delta x) = f(x) + f'(x)\Delta x + f''(z)\bigg|_{x<z<x+\Delta x} \frac{(\Delta x)^2}{2!}. \]
Therefore,
\[ f'(x) = \frac{f(x + \Delta x) - f(x)}{\Delta x} - f''(z)\bigg|_{x<z<x+\Delta x} \frac{(\Delta x)^2}{2!}. \]
The error between the FD approximation (7) and \( f'(x) \) is
\[ \left| f''(z)\bigg|_{x<z<x+\Delta x} \frac{(\Delta x)^2}{2!} \right|, \]
that is, \( O((\Delta x)^2) \).
Central difference approximation:

\[ f'_{FD}(x) = \frac{f(x + \Delta x) - f(x - \Delta x)}{2\Delta x}. \]

Error is \( O((\Delta x)^3) \).
Finite Difference Approximation of the Policy Gradient

To compute the gradient of $J_\rho(\pi_{\theta})$ w.r.t. $\theta \in \mathbb{R}^p$, we need to compute $2p$ evaluations of $J_\rho$:

$$\nabla_\theta J_\rho(\pi_{\theta}) \approx \nabla^{(\text{FD})}_\theta J_\rho(\pi_{\theta}) = \left[ \begin{array}{c} \frac{J_\rho(\theta+\varepsilon e_1)-J_\rho(\theta-\varepsilon e_1)}{2\varepsilon} \\ \vdots \\ \frac{J_\rho(\theta+\varepsilon e_i)-J_\rho(\theta-\varepsilon e_i)}{2\varepsilon} \\ \vdots \\ \frac{J_\rho(\theta+\varepsilon e_p)-J_\rho(\theta-\varepsilon e_p)}{2\varepsilon} \end{array} \right],$$

where $e_i$ is a unit vector along dimension $i$ of $\mathbb{R}^p$. 
We cannot directly compute $J_\rho(\pi_\theta)$. We can only compute $\hat{J}_n(\pi_\theta)$ using rollouts. Replace each $J_\rho$ above with their corresponding $\hat{J}_n$. This requires $2pn$ rollouts in total.

Given the approximated gradient, which has error caused by both the FD approximation and using $\hat{J}_n$ instead of $J_\rho$, we may use the gradient ascent to move towards higher value of $J_\rho(\pi_\theta)$:

$$\theta_{k+1} \leftarrow \theta_k + \alpha_k \nabla_{\theta}^{(\text{FD})} \hat{J}_n(\pi_{\theta_k}).$$

(8)

Even though this is a feasible approach, we can compute the gradient more elegantly.
Suppose that we want to find a policy $\pi_\theta : X \rightarrow M(A)$ with $\theta \in \mathbb{R}^p$ that maximizes the performance for the immediate reward problem.

Recall that the interaction protocol is

- At episode $t$, $X_t \sim \rho \sim M(X)$
- The agent chooses action $A_t \sim \pi_\theta (\cdot | X_t)$
- The agent receives reward $R_t \sim \mathcal{R}(\cdot | X_t, A_t)$.
- The agent starts the new (independent) episode $t + 1$.

This is an RL setting as $\rho$ and $\mathcal{R}$ are not directly available to the agent, but only through samples.
Performance Measure

The performance measure is

\[ J_\rho(\pi_\theta) = \int V^{\pi_\theta}(x)d\rho(x) = \int r^{\pi_\theta}(x)d\rho(x) \]

\[ = \int r(x, a)\pi_\theta(a|x)d\rho(x)da, \]

as the value function \( V^{\pi_\theta} \) for the immediate reward problem is the same as \( r^{\pi_\theta} \).

Considered the action space to be continuous and we assume that \( \pi_\theta(\cdot|x) \) provides a density over the state space.

If we had a discrete action space, we would have

\[ \int_X d\rho(x) \sum_{a \in A} r(x, a)\pi_\theta(a|x). \]

We may switch back and forth between continuous and discrete action spaces.
The gradient of $J_\rho(\pi_\theta)$ w.r.t. $\theta$:

$$\nabla_\theta J_\rho(\pi_\theta) = \int r(x, a) \nabla_\theta \pi_\theta(a|x) d\rho(x) da$$

$$= \int d\rho(x) \int r(x, a) \nabla_\theta \pi_\theta(a|x) da$$

$$= \mathbb{E}_{X \sim \rho} \left[ \int r(X, a) \nabla_\theta \pi_\theta(a|X) da \right]. \quad (9)$$

For discrete action spaces, the inner integral becomes

$$\sum_{a \in A} r(x, a) \nabla_\theta \pi_\theta(a|x).$$

We call $\nabla_\theta J_\rho(\pi_\theta)$ the Policy Gradient (PG).
If we can compute PG, we can update the policy parameters, using a gradient ascent method:

$$\theta_{k+1} \leftarrow \theta_k + \alpha_k \nabla_{\theta} J_{\rho}(\pi_{\theta_k}),$$

(10)

similar to what we have done using the FD approximation (8).
Computing the Policy Gradient

\[ \nabla_\theta J_\rho(\pi_\theta) = \int r(x, a) \nabla_\theta \pi_\theta(a|x) d\rho(x) da \]

- How can we compute this gradient?
- We build this gradually in several steps.
- At each step, we relax some assumptions until we get to a procedure that can use the data available by the interaction protocol above.
Computing the Policy Gradient – Known $\rho$ and $r$

\[
\nabla_\theta J_\rho(\pi_\theta) = \int r(x, a) \nabla_\theta \pi_\theta(a|x) d\rho(x) da
\]

- If we know $\rho$ and $r$, we have all the necessary information for computing the gradient.
  - For each $x \in X$, we compute the summation (or integral) over all $a \in A$ of $r(x, a) \nabla_\theta \pi_\theta(a|x)$.
  - We weigh that term proportional to $\rho(x)$.
  - Take average over all $x$.

- But this is not the RL setting described as the interaction protocol at the beginning of the section.
Computing the Policy Gradient – Known \( r \), unknown \( \rho \)

- Assume that \( r \) is known, but \( \rho \) can only be sampled.
- Approximately solve this problem by sampling \( X_i \sim \rho \) \((i = 1, \ldots, n)\) and computing
  \[
  \frac{1}{n} \sum_{i=1}^{n} \sum_{a \in A} r(X_i, a) \nabla_\theta \pi_\theta(a | X_i).
  \]
  \[
  \text{or}
  \frac{1}{n} \sum_{i=1}^{n} \int r(X_i, a) \nabla_\theta \pi_\theta(a | X_i) \, da.
  \]

As \( X_i \sim \rho \), this is an unbiased estimate of

\[
\nabla_\theta J_\rho(\pi_\theta) = \mathbb{E}_{X \sim \rho} \left[ \sum_{a \in A} r(X, a) \nabla_\theta \pi_\theta(a | X) \right]
\]

or \( \mathbb{E}_{X \sim \rho} \left[ \int r(x, a) \nabla_\theta \pi_\theta(a | x) \, da \right] \) (continuous).
Computing the Policy Gradient – Known $r$, unknown $\rho$

- As $X_i \sim \rho$, this is an unbiased estimate of

$$\nabla_\theta J_\rho(\pi_\theta) = \mathbb{E}_{X \sim \rho} \left[ \sum_{a \in A} r(X, a) \nabla_\theta \pi_\theta(a|X) \right]$$

or

$$\mathbb{E}_{X \sim \rho} \left[ \int r(x, a) \nabla_\theta \pi_\theta(a|x) da \right].$$

- This is still not feasible if $r$ is unknown in our interaction protocol:
  - the agent is initialized at state $x$
  - it has to choose its action according to $A \sim \pi_\theta(\cdot|x)$. 
Computing the Policy Gradient – Unknown \( r \) and \( \rho \)

- The term
  \[
  \sum_{a \in \mathcal{A}} r(x, a) \nabla_\theta \pi_\theta(a|x)
  \]
  can be interpreted as the expectation of
  \[
  r(x, A) \nabla_\theta \pi_\theta(A|x)
  \]
  when \( A \) is coming from a uniform distribution with \( q(a) = \frac{1}{|\mathcal{A}|} \) (for \( a \in \mathcal{A} \)).

- We have
  \[
  \sum_{a \in \mathcal{A}} r(x, a) \nabla_\theta \pi_\theta(a|x) = |\mathcal{A}| \sum_{a \in \mathcal{A}} q(a) r(x, a) \nabla_\theta \pi_\theta(a|x). \quad (12)
  \]

- Similar for the continuous case.
Computing the Policy Gradient – Unknown $r$ and $\rho$

- If the actions were coming from a uniform distribution, we could easily form an empirical estimate of these terms.

- But the actions in the interaction protocol comes from distribution $\pi_\theta(\cdot|x)$, which in general is different distribution than a uniform one.

- We have some form of off-policy sampling scenario in the distribution of actions.

- Some approaches to deal with it:
  - Estimate $\hat{r} \approx r$ using data (model-based approach).
  - Modify $r(x, A) \nabla_\theta \pi_\theta(A|x)$ to a quantity that can be estimated from data.
Computing the Policy Gradient – Unknown $r$ and $\rho$

- Observation: for a function $f : \mathbb{R} \to \mathbb{R}$, we have

$$\frac{d \log f(x)}{dx} = \frac{df(x)}{dx} \frac{1}{f(x)},$$

or more generally, for a function $f : \mathbb{R}^p \to \mathbb{R}$,

$$\nabla_x \log f(x) = \frac{\nabla_x f(x)}{f(x)}.$$
Computing the Policy Gradient – Unknown $r$ and $\rho$

- Using this observation, we get

$$\int r(x, a) \nabla_\theta \pi_\theta(a|x) da = \int r(x, a) \pi_\theta(a|x) \nabla_\theta \log \pi_\theta(a|x) da$$

$$= \mathbb{E}_{A \sim \pi_\theta(\cdot|x)} [r(x, A) \nabla_\theta \log \pi_\theta(A|x)].$$

- The desired quantity can be written as the expectation of

$$r(x, A) \nabla_\theta \log \pi_\theta(A|x)$$

when $A \sim \pi_\theta(\cdot|x)$.

- The sampling distribution is the same as the one agent uses to choose its actions.

- We are in the on-policy sampling scenario over the choice of actions.
If $X \sim \rho$ and $A \sim \pi_\theta(\cdot|X)$, the random variable
\[
r(X, A) \nabla_\theta \log \pi_\theta(A|X)
\]
is an unbiased estimate of $\nabla_\theta J_\rho(\pi_\theta)$, i.e.,
\[
\nabla_\theta J_\rho(\pi_\theta) = \mathbb{E} [r(X, A) \nabla_\theta \log \pi_\theta(A|X)]
\]
\[
= \mathbb{E}_{X \sim \rho} \left[ \mathbb{E}_{A \sim \pi_\theta(\cdot|X)} [r(X, A) \nabla_\theta \log \pi_\theta(A|X) | X] \right].
\]

We can estimate the gradient of the performance w.r.t. the parameters of the policy using data available through the interaction of the agent with its environment.

We may use this estimate in (10) to update the policy parameters using unbiased but noisy estimate of the gradient.

This makes it a stochastic gradient ascent.
Two Sources of Variance

\[ r(X, A) \nabla_{\theta} \log \pi_{\theta}(A|X) \]

- Unbiased estimate of the gradient
- But it has variance due to two sources of randomness:
  - Variance of estimating
    \[ g(x; \theta) \triangleq \mathbb{E}_{A \sim \pi_{\theta}(\cdot|X)} \left[ r(X, A) \nabla_{\theta} \log \pi_{\theta}(A|X) \mid X = x \right] \]
    with a single sample \( r(X, A) \nabla_{\theta} \log \pi_{\theta}(A|X) \).
  - Variance of estimating \( \mathbb{E}_{X \sim \rho} [g(X; \theta)] \) using a single sample.
Two Sources of Variance

\[ r(X, A) \nabla_\theta \log \pi_\theta(A|X) \]

One can show that the variance along the \( i \)-th dimension of this r.v. is

\[
\text{Var} \left[ r(X, A) \frac{\partial \log \pi_\theta(A|X)}{\partial \theta_i} \right] = \\
\mathbb{E}_{X \sim \rho} \left[ \text{Var} \left[ r(X, A) \frac{\partial \log \pi_\theta(A|X)}{\partial \theta_i} \mid X \right] \right] + \text{Var}_{X \sim \rho} \left[ g_i(X; \theta) \right].
\]

(15)
Two Sources of Variance

Let us define

\[ g(x; \theta) = \mathbb{E}_{A \sim \pi_\theta(\cdot|x)} [r(x, A) \nabla_\theta \log \pi_\theta(A|x)] . \]  

(16)

The function \( g : \mathcal{X} \times \Theta \rightarrow \mathbb{R}^p \) is the gradient of \( r^{\pi_\theta} \) w.r.t. \( \theta \) at state \( x \), and is a \( p \)-dimensional vector.

If we knew \( r(x, a) \) and we could compute \( g(x; \theta) \), we wouldn’t have the first source of variance, but we still would have the second one.

The variance would be

\[ \text{Var}_{X \sim \rho} [g_i(X; \theta)] . \]

These two sources of variance make our estimate of the gradient inaccurate.

There are ways to reduce them.
Variance Reduction – Randomness of States

- Suppose we can compute $g(x; \theta)$ exactly for any given $x \in \mathcal{X}$.
- Instead of a single sample $g(X_1; \theta)$, we use multiple independent samples $X_1, \ldots, X_n$, all distributed according to $\rho$, to estimate the PG:

\[
\nabla \theta J_\rho(\pi_\theta) \approx \frac{1}{n} \sum_{i=1}^{n} g(X_i; \theta)
\]

\[
= \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{A \sim \pi_\theta(\cdot | X_i)} [r(X_i, A) \nabla \theta \log \pi_\theta(A|X_i)].
\]

- The variance of this estimator, along dimension $i$, is

\[
\frac{1}{n} \text{Var}_{X \sim \rho}[g_i(X; \theta)].
\]

As $n \to \infty$, the variance goes to zero. This leads to more accurate estimate of the PG, hence more accurate update of the policy.
Variance Reduction – Randomness of Actions

Consider the variance of estimating $g(x; \theta)$ (16) using a single sample $r(x, A) \nabla_\theta \log \pi_\theta(A|x)$ with $A \sim \pi_\theta(\cdot|x)$.

For each dimension $i$, we have

$$
\mathbb{E} \left[ \frac{\partial \log \pi_\theta(A|x)}{\partial \theta_i} b(x) \mid x \right] = \int \pi_\theta(a|x) \frac{\partial \log \pi_\theta(a|x)}{\partial \theta_i} b(x) da
$$

$$
= \int \frac{\partial \pi_\theta(a|x)}{\partial \theta_i} b(x) da
$$

$$
= b(x) \int \frac{\partial \pi_\theta(a|x)}{\partial \theta_i} da
$$

$$
= b(x) \frac{\partial}{\partial \theta_i} \int \pi_\theta(a|x) da = 0.
$$

\begin{align*}
\text{=} & 1
\end{align*}
Variance Reduction – Randomness of Actions

- This shows that
\[
\mathbb{E} \left[ \frac{\partial \log \pi_\theta(A|x)}{\partial \theta_i} r(x, A) \mid x \right] = \mathbb{E} \left[ \frac{\partial \log \pi_\theta(A|x)}{\partial \theta_i} (r(x, A) + b(x)) \mid x \right].
\]

(17)

- Adding a state-dependent function \( b : \mathcal{X} \rightarrow \mathbb{R} \) to \( r(x, a) \) does not change the expectation.

- But it may change the variance!
Variance Reduction – Randomness of Actions

- For each dimension $i$ of the PG, we can use a different state-dependent function.
- For any state-dependent function $b : \mathcal{X} \rightarrow \mathbb{R}^p$, the PG (14) is

$$\nabla_\theta J_\rho(\pi_\theta) = \mathbb{E} \left[ r(X, A) \nabla_\theta \log \pi_\theta(A|X) \right] = \mathbb{E} \left[ (r(X, A)1 + b(X)) \odot \nabla_\theta \log \pi_\theta(A|X) \right],$$

where $1$ is a $p$-dimensional vector with all components equal to 1, and $\odot$ is a pointwise (Hadamard) product of two vectors, i.e., for $u, v \in \mathbb{R}^p$, $[u \odot v]_i = u_i v_i$.

- If we simply choose a scalar function $b$, which is often the case in practice, we have

$$\nabla_\theta J_\rho(\pi_\theta) = \mathbb{E} \left[ (r(X, A) + b(X)) \nabla_\theta \log \pi_\theta(A|X) \right].$$

- The function $b$ is called the baseline.
The baseline can be used in order to minimize the variation of \( p \)-dimensional random vector.

We use the variance for this purpose.

We would like to find a function \( b : \mathcal{X} \rightarrow \mathbb{R}^p \) such that for all \( x \in \mathcal{X} \),

\[
\min_b \sum_{i=1}^p \text{Var} \left[ (r(x, A) + b_i(x)) \frac{\partial \log \pi_\theta(A|x)}{\partial \theta_i} \mid x \right] = \\
\text{Tr Cov} \left( (r(X, A)1 + b(x)) \odot \nabla_\theta \log \pi_\theta(A|x) \mid x \right)
\]
Variance Reduction – Randomness of Actions – Baseline

\[ b_i(x) = \frac{-\mathbb{E} \left[ r(x, A) \left( \frac{\partial \log \pi_{\theta}(A|x)}{\partial \theta_i} \right)^2 \mid x \right]}{\mathbb{E} \left[ \left( \frac{\partial \log \pi_{\theta}(A|x)}{\partial \theta_i} \right)^2 \mid x \right]} \tag{18} \]

We could choose a single scalar function \( b : \mathcal{X} \rightarrow \mathbb{R} \) instead. In that case, the solution would be

\[ b(x) = \frac{-\mathbb{E} \left[ r(x, A) \Vert \nabla_\theta \log \pi_{\theta}(A|x) \Vert_2^2 \mid x \right]}{\mathbb{E} \left[ \Vert \nabla_\theta \log \pi_{\theta}(A|x) \Vert_2^2 \mid x \right]} \]
We derive the PG for continuing tasks.

The difference with the immediate reward case is that the performance $J_\rho(\pi_\theta)$ depends on the dynamics $\mathcal{P}^{\pi_\theta}$ too.

As we change $\theta$, the dynamics $\mathcal{P}^{\pi_\theta}$ changes as well.

This seems to complicate the gradient computation.

It turns out that despite this challenge, the PG can be written in an elegant, and relatively easy to compute, form.
Discounted Future-State Distribution

- New notations to present the results more compactly.
- Recall that $\mathcal{P}^\pi(\cdot|x; k) = \mathcal{P}^\pi(\cdot|x)^k$ is the probability distribution of following policy $\pi$ for $k \geq 0$ steps.
- We introduce **discounted future-state distribution** of starting from $x \in \mathcal{X}$ and following $\pi$ as

\[
\rho^\pi_\gamma(\cdot|x) = \rho_\gamma(\cdot|x; \mathcal{P}^\pi) \triangleq (1 - \gamma) \sum_{k \geq 0} \gamma^k \mathcal{P}^\pi(\cdot|x; k).
\]

- It is easy to verify that $\rho^\pi_\gamma(\cdot|x)$ is a valid probability distribution, e.g., $\rho^\pi_\gamma(\mathcal{X}|x) = 1$. 

\[19\]
Discounted Future-State Distribution

The relevant of this distribution becomes more clear if we note that

\[ V^\pi(x) = \mathbb{E} \left[ \sum_{t \geq 0} \gamma^t R_t \mid X_0 = x \right] \]

\[ = \sum_{t \geq 0} \gamma^t \mathbb{E} [R_t \mid X_0 = x] \]

\[ = \sum_{t \geq 0} \gamma^t \int \mathcal{P}^\pi(dx' \mid x; t)r(x') \]

\[ = \frac{1}{1 - \gamma} \int \rho^\pi_{\gamma}(dx' \mid x)r(x') = \frac{1}{1 - \gamma} \mathbb{E}_{X' \sim \rho^\pi_{\gamma}(\cdot \mid x)} [r(X')] . \]

The value function at a state \( x \) is the expected reward when \( X' \) is distributed according to \( \rho^\pi_{\gamma}(\cdot \mid x) \).
Discounted Future-State Distribution

- Interpretation: The agent starts from state \( x \) and at each time step, it decides to follow \( \pi \) with probability \( \gamma \) or terminates the episode with probability \( 1 - \gamma \).

- We can also define discounted future-state distribution of starting from \( \rho \) and following \( \pi \) as

\[
\rho_\gamma^\pi(\cdot) = \rho_\gamma(\cdot | \mathcal{P}^\pi) \triangleq \int \rho_\gamma(\cdot | x; \mathcal{P}^\pi) d\rho(x).
\]

- The performance measure \( J(\pi_\theta) \) (3) is

\[
J(\pi_\theta) = \mathbb{E}_{X \sim \rho} [V_{\pi_\theta}(X)] = \frac{1}{1 - \gamma} \mathbb{E}_{X \sim \rho_\gamma^\pi} [r(X)].
\]
Policy Gradient Theorem

Theorem (Policy Gradient Theorem – Sutton et al. 2000)

Assume that $\pi_\theta$ is differentiable w.r.t. $\theta \in \Theta$. We have

$$\nabla_\theta J_\rho(\pi_\theta) =$$

$$\sum_{k \geq 0} \gamma^k \int d\rho(x) \mathcal{P}^{\pi_\theta}(dx'|x; k) \int \nabla_\theta \pi_\theta(a'|x') Q^{\pi_\theta}(x', a') da' =$$

$$\frac{1}{1 - \gamma} \int \rho_\gamma^{\pi_\theta}(dx) \int \pi_\theta(a|x) \nabla_\theta \pi_\theta(a|x) Q^{\pi_\theta}(x, a) da =$$

$$\frac{1}{1 - \gamma} \mathbb{E} \left[ \nabla_\theta \log \pi_\theta(A|X) Q^{\pi_\theta}(X, A) \right],$$

with $X \sim \rho_\gamma^{\pi_\theta}, A \sim \pi_\theta(\cdot|X)$. 
We write the value function at state $x \in \mathcal{X}$ as the expected value of the action-value function, i.e.,

$$V^{\pi_\theta}(x) = \int \pi_\theta(a|x)Q^{\pi_\theta}(x,a) da.$$  

We take its derivative w.r.t. $\theta$ and use the product rule to get

$$\nabla_\theta V^{\pi_\theta}(x) = \int \left[ \nabla_\theta \pi_\theta(a|x)Q^{\pi_\theta}(x,a) + \pi_\theta(a|x)\nabla_\theta Q^{\pi_\theta}(x,a) \right] da.$$  

(20)
Policy Gradient Theorem – Proof

As \( Q^{\pi_\theta}(x, a) = r(x, a) + \gamma \int \mathcal{P}(dx'|x, a) V^{\pi_\theta}(x') \),

\[
\nabla_\theta Q^{\pi_\theta}(x, a) = \gamma \int \mathcal{P}(dx'|x, a) \nabla_\theta V^{\pi_\theta}(x').
\]

This alongside with (20) gives us the recursive Bellman-like equation for the gradient of \( V^{\pi_\theta}(x) \):

\[
\nabla_\theta V^{\pi_\theta}(x) = \int \nabla_\theta \pi_\theta(a|x) Q^{\pi_\theta}(x, a) da + \gamma \int \mathcal{P}^{\pi_\theta}(dx'|x) \nabla_\theta V^{\pi_\theta}(x').
\]

(21)
Policy Gradient Theorem – Proof

Expanding $\nabla_\theta V^{\pi_\theta}(x')$ likewise, we get that

$$\nabla_\theta V^{\pi_\theta}(x) = \int \nabla_\theta \pi_\theta(a|x) Q^{\pi_\theta}(x, a) da +$$

$$\gamma \int \mathcal{P}^{\pi_\theta}(dx'|x) \left[ \nabla_\theta \pi_\theta(a'|x') Q^{\pi_\theta}(x', a') da' + \right.$$

$$\gamma \int \mathcal{P}^{\pi_\theta}(dx''|x') \nabla_\theta V^{\pi_\theta}(x'') \right].$$

Following this pattern recursively, we get that

$$\nabla_\theta V^{\pi_\theta}(x) = \sum_{k \geq 0} \gamma^k \int \mathcal{P}^{\pi_\theta}(dx'|x; k) \int \nabla_\theta \pi_\theta(a'|x') Q^{\pi_\theta}(x', a') da'$$

$$= \frac{1}{1 - \gamma} \int \rho^{\pi_\theta}(dx'|x) \int \nabla_\theta \pi_\theta(a'|x') Q^{\pi_\theta}(x', a') da'.$$
Also since $\nabla_\theta \pi_\theta (a'|x') = \pi_\theta (a'|x') \nabla_\theta \log \pi_\theta (a'|x')$, we can write the gradient as

$$\nabla_\theta V^{\pi_\theta} (x) = \frac{1}{1 - \gamma} \int \rho_\gamma^{\pi_\theta} (dx'|x) \int \pi_\theta (a'|x') \nabla_\theta \log \pi_\theta (a'|x') Q^{\pi_\theta} (x', a') da' = \frac{1}{1 - \gamma} \int \rho_\gamma^{\pi_\theta} (dx'|x) \mathbb{E}_{A' \sim \pi_\theta (\cdot|X')} \left[ \nabla_\theta \log \pi_\theta (A'|X') Q^{\pi_\theta} (X', A') \right].$$
Policy Gradient Theorem – Proof

As \( J_\rho(\pi_\theta) = \int V^{\pi_\theta}(x) d\rho(x) \), taking the average of \( x \) w.r.t. \( \rho \), we get that

\[
\nabla_\theta J_\rho(\pi_\theta) = \frac{1}{1 - \gamma} \int \rho^{\pi_\theta} (dx) \int \pi_\theta(a|x) \nabla_\theta \pi_\theta(a|x) Q^{\pi_\theta}(x, a) da
\]

\[
= \frac{1}{1 - \gamma} \mathbb{E}_{X \sim \rho^{\pi_\theta}} \left[ \nabla_\theta \log \pi_\theta(A|X) Q^{\pi_\theta}(X, A) \right],
\]

which is the desired result.
Policy Gradient Theorem

- This theorem provides an elegant formula for the PG.
- It relates the PG to the discounted future-state distribution $\rho^{\pi_\theta}$, the action-value function $Q^{\pi_\theta}(x, a)$, and the gradient of $\pi_\theta$.
- To compute the PG in the RL setting, we have to estimate it using samples. If we get
  - a state $X$ sampled from $\rho^{\pi_\theta}$,
  - an action $A$ sampled from $\pi_\theta(\cdot|X)$, and
  - know action-value $Q^{\pi_\theta}(X, A)$,
the random variables

$$\nabla_\theta \log \pi_\theta(A|X)Q^{\pi_\theta}(X, A)$$

is an unbiased estimate of the PG (cf. (13)).
- We can then use it in an SGD scheme to improve the policy.
Sampling $X$ from $\rho^{\pi\theta}_{\gamma}$

Sampling from $\rho^{\pi\theta}_{\gamma}$ is relatively straightforward in the on-policy sampling scenario when the agent follows $\pi_\theta$.

- The agent starts an episode from $X_0 \sim \rho$ and follows $\pi_\theta$.
- We get a sequence of states $X_0, X_1, \ldots$.
- These would be samples from $\int d\rho(x) \mathcal{P}^{\pi_\theta}(\cdot|x; k)$ for $k = 0, 1, \ldots$.
- The distribution $\rho^{\pi\theta}_{\gamma}$, however, has a $\gamma^k$ factor for the $k$-th time step, see (19).
- Its effect is that the contribution to the gradient from $X_k$, which is

$$\mathbb{E} [\nabla_\theta \log \pi_\theta(A|X)Q^{\pi_\theta}(X, A)] = \int \pi_\theta(a|x)\nabla_\theta \pi_\theta(a|x)Q^{\pi_\theta}(x, a)da,$$

should be weighted by $\gamma^k$. 
Sampling $X$ from $\rho_{\pi, \theta}$

Another way to directly sample from $\rho_{\gamma}^{\pi, \theta}$ is to follow $\pi$, but at each step terminate the episode with probability $1 - \gamma$. 

Sampling $A$ from $\pi_\theta(\cdot|X)$

An action $A$ sampled from $\pi_\theta(\cdot|X)$ is automatically generated when the agent follows policy $\pi_\theta$ (on-policy).
The remaining issue is the computation of $Q^{\pi \theta}(X, A)$ for $X \sim \rho^{\pi \theta}$ and $A \sim \pi_\theta(\cdot | X)$ using data.

This is essentially a PE problem, and we may use various action-value function estimators that we have developed so far.
Computation of $Q^{\pi_\theta}(X, A)$

- A simple approach: Use the MC estimate $Q^{\pi_\theta}(X, A)$. This would lead to what is known as the REINFORCE algorithm by Williams [1992].

- In the on-policy setting when the agent follows $\pi_\theta$, it generates the sequence $X_0, A_0, R_0, X_1, A_1, R_1, \ldots$ with $A_t \sim \pi_\theta(\cdot|X_t)$.

- The return $G^\pi_t = \sum_{k \geq t} \gamma^{k-t} R_k$ is an unbiased estimate of $Q^{\pi_\theta}(X_t, A_t)$.

- We replace the action-value function at that state-action with this return from time $t$ onward.

- The return, however, is a high variance estimate of the action-value function.

- One approach is to use a baseline in order to reduce the variance of this MC estimate.

$^1$REINFORCE stands for REward Increment × Nonnegative Factor × Offset Reinforcement × Characteristic Eligibility.
Computation of $Q^\pi_\theta(X,A)$ – Actor-Critic Methods

- Another approach is to use an action-value function estimator instead.
  - TD methods
  - LSTD
  - Fitted Value Iteration (for PE, and not for Control)
- Such a method is called **actor-critic** method
  - The actor refers to the policy (and often PG method to improve it)
  - The critic refers to the value function estimator used to criticize the policy (actor).
The use of a critic, however, may induce a bias as $\mathbb{E} \left[ \hat{Q}^{\pi_\theta}(X, A) | X, A \right]$ may be different from $Q^{\pi_\theta}(X, A)$, especially if we use a TD method (which introduces bias because of bootstrapping) or a function approximator (for large state-action spaces).

Such a method would explicitly represent both policy and value function.

Actor-critic methods bring together some of the benefits of both value-based and policy search methods.
Summary

- Policy Search Methods: Explicit representation of the policy and searching within the policy space
- The search might be guided by the zero-order or first-order methods
- We may sometimes constraint the change of the policy update
  - Policy gradient is only a local information. We should not make a large move.
References
