Policy Gradient Descent

Lucas Janson and Sham Kakade

CS/Stat 184: Introduction to Reinforcement Learning

Fall 2022

Today

- Recap

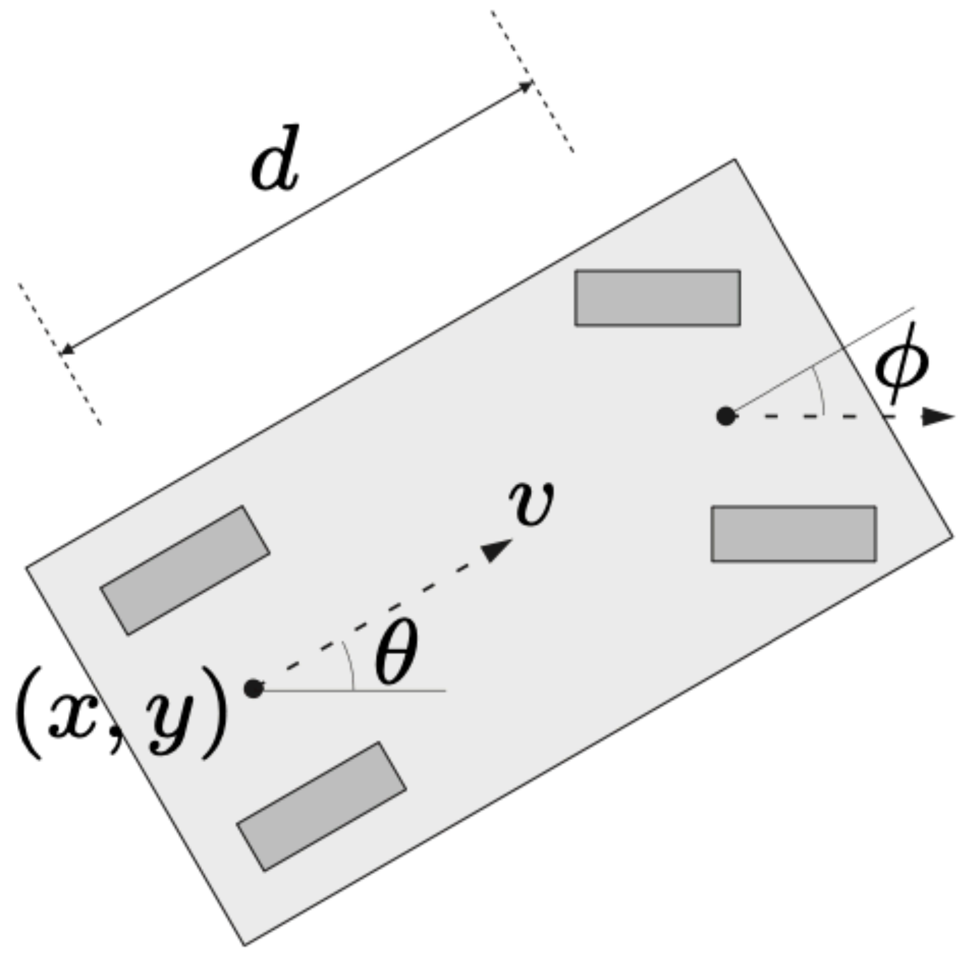
- Today:
  - How do we learn/compute a good policy in an intractably large MDP?
    - Policy gradient descent is one of the most effective methods.

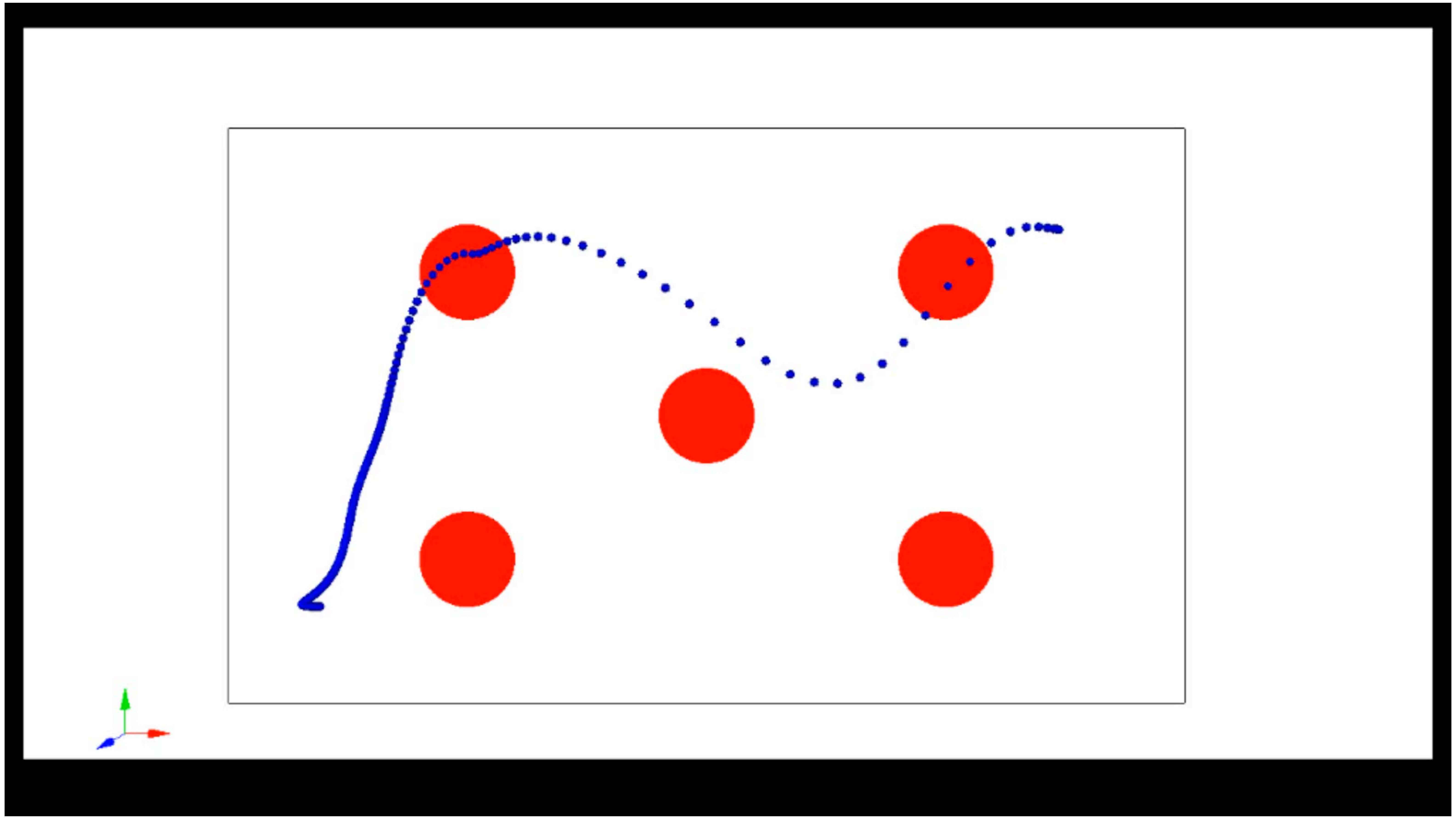
Recap

Example:

2-d car navigation

Cost function is designed such that it gets to the goal without colliding with obstacles (in red)





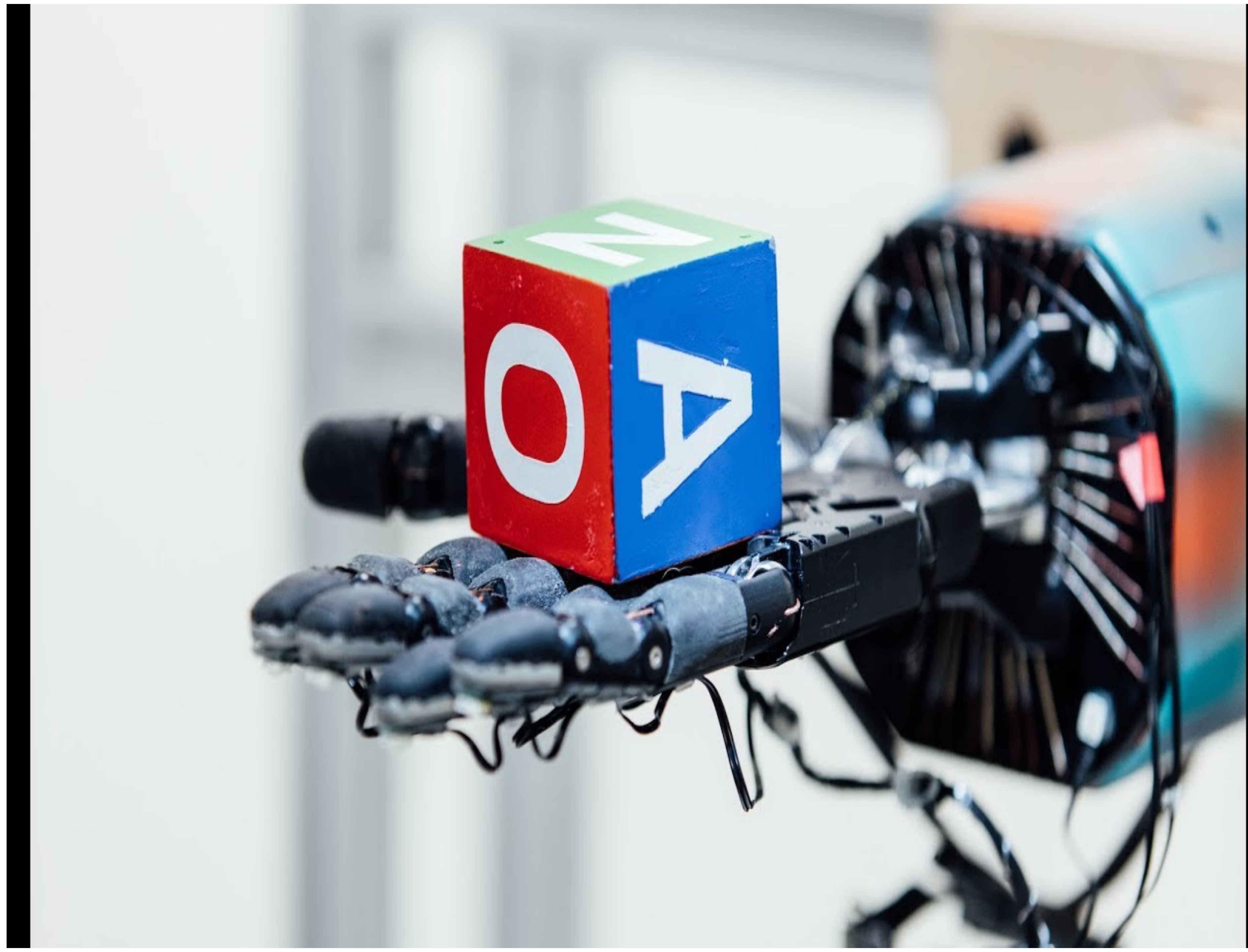
Today:

Policy Gradient Descent

Policy Optimization

[AlphaZero, Silver et.al, 17]

[OpenAI Five, 18]



[OpenAI,19]

Today: Policy Gradient Deriviation

Consider parameterized policy:

$$\pi_\theta(a \mid s) = \pi(a \mid s; \theta)$$

$$J(\theta) := E_{s_0 \sim \mu_0}\left[V^{\pi_\theta}(s_0)\right]$$

$$= E\left[\sum_{h=0}^{\infty} \gamma^h r_h \,\middle|\, \mu_0, \pi_\theta\right]$$

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

Main question for today's lecture: how to compute the gradient?

Outline for today

1. Recap on Gradient Descent (GD) and Stochastic Gradient Descent (SGD)

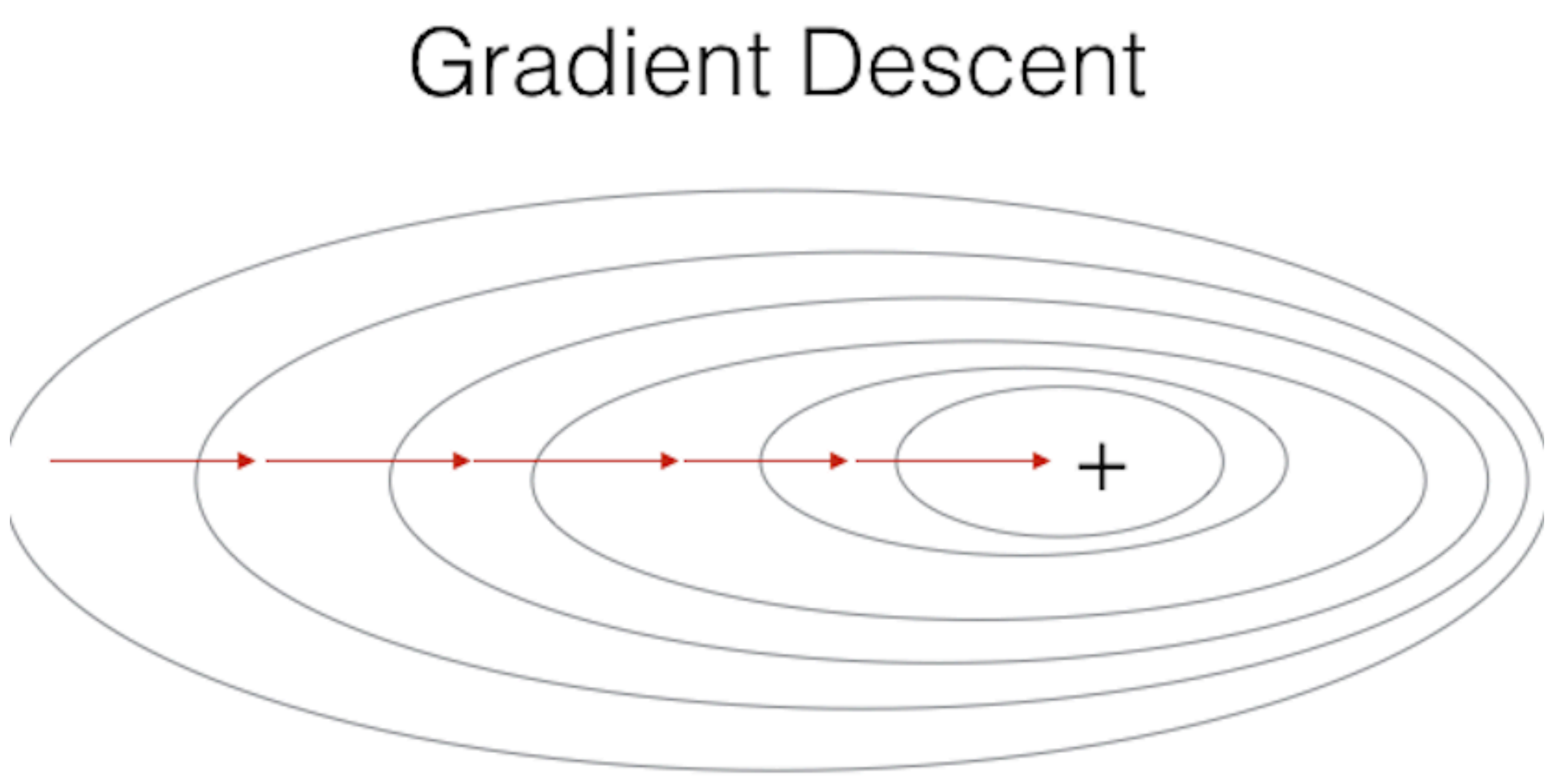
2. Warm up: computing gradient using importance weighting

3. Policy Gradient formulations

Gradient Descent

Given an objective function $J(\theta) : \mathbb{R}^d \mapsto \mathbb{R}$, (e.g., $J(\theta) = \mathbb{E}_{x,y}(f_\theta(x) - y)^2$)

GD minimizes the above objective function as follows:



Initialize θ_0, for t = 0, … :

$$\theta_{t+1} = \theta_t - \eta \nabla J(\theta_t)$$

Stochastic Gradient Descent

Given an objective function $J(\theta) : \mathbb{R}^d \mapsto \mathbb{R}$, (e.g., $J(\theta) = \mathbb{E}_{x,y}(f_\theta(x) - y)^2$)

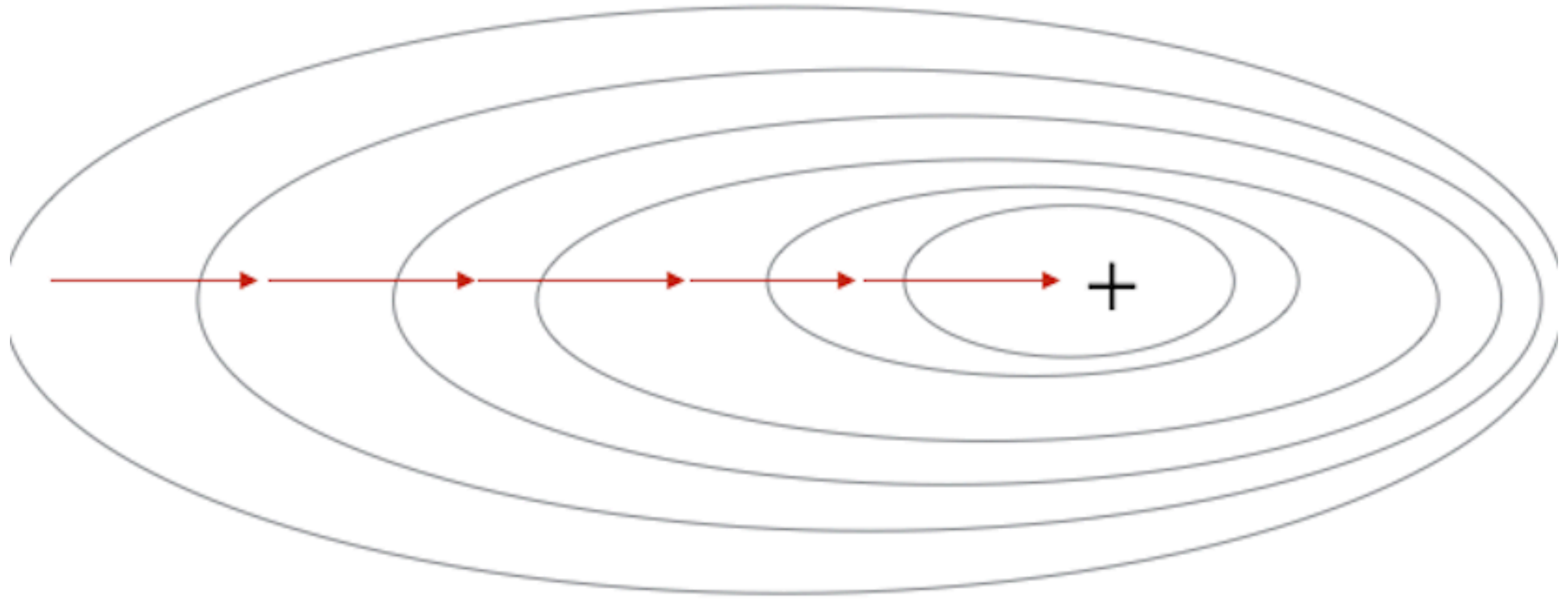
SGD minimizes the above objective function as follows:

Initialize θ_0, for t = 0, … :

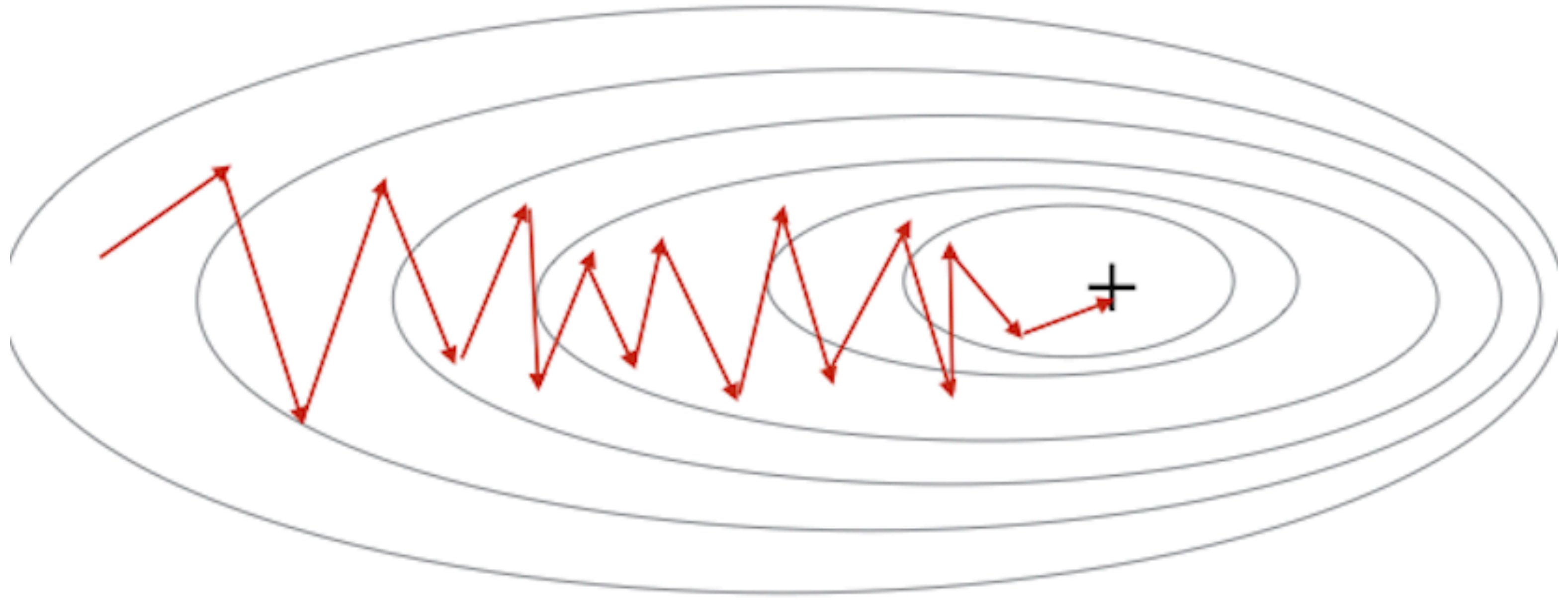
$$\theta_{t+1} = \theta_t - \eta_t g_t$$

where $\mathbb{E}[g_t] = \nabla_\theta J(\theta_t)$

Gradient Descent



Stochastic Gradient Descent



Brief overview of GD/SGD:

- Global optima, local optima, and saddle points (by picture)

- For convex functions (with certain regularity conditions, such as “smoothness”),
  - GD (with an appropriate constant learning rate) converges to the global optima.
  - SGD (with an appropriately decaying learning rate) converges to the global optima.

- For non-convex functions, we hope to find a local minima.

- What we can prove (under mild regularity conditions) is a little weaker:
  - GD (with an appropriate constant learning rate) converges to a saddle point.
  - SGD (with an appropriately decaying learning rate) converges to a saddle point.

SGD: Convergence to a Stationary Point for Nonconvex Functions

- Def of β-smooth: $\|\nabla_\theta J(\theta) - \nabla_\theta J(\theta_0)\|_2 \le \beta \|\theta - \theta_0\|_2$

- [**Theorem**] Suppose we run SGD: $\theta_{t+1} = \theta_t - \eta \widetilde{\nabla}_\theta J(\theta_t)$, for T steps, where $\mathbb{E}\left[\widetilde{\nabla}_\theta J(\theta_t)\right] = \nabla_\theta J(\theta_t)$ with $\eta = O(1/\sqrt{T})$. Assume:
  - $J(\theta)$ is β-smooth.
  - $J(\theta)$ is bounded: $|J(\theta)| \le M, \quad \forall \theta$.
  - $\widetilde{\nabla}_\theta J(\theta)$ has "bounded second moment": $\mathbb{E}\left[\|\widetilde{\nabla}_\theta J(\theta_t)\|_2^2\right] \le \sigma^2$,

then, in T steps, SGD will find a θ such that: $\|\nabla_\theta J(\theta)\|^2 \le O\left(\sqrt{M\beta\sigma^2/T}\right)$.

Proof of Convergence to Stationary Point (optional)

If J is β-smooth, then $\left|J(\theta) - J(\theta_0) - \nabla_\theta J(\theta_0)^\top(\theta - \theta_0)\right| \le \frac{\beta}{2}\|\theta - \theta_0\|_2^2, \ \forall \theta, \theta_0$

$$\left|J(\theta_{t+1}) - J(\theta_t) - \nabla_\theta J(\theta_t)^\top(\theta_{t+1} - \theta_t)\right| \le \frac{\beta}{2}\|\theta_{t+1} - \theta_t\|_2^2$$

$$\Rightarrow \left|J(\theta_{t+1}) - J(\theta_t) + \eta \nabla_\theta J(\theta_t)^\top \widetilde{\nabla}_\theta J(\theta_t)\right| \le \frac{\beta}{2}\eta^2\|\widetilde{\nabla}_\theta J(\theta_t)\|_2^2$$

$$\Rightarrow \eta \nabla_\theta J(\theta_t)^\top \widetilde{\nabla}_\theta J(\theta_t) \le -J(\theta_{t+1}) + J(\theta_t) + \frac{\beta}{2}\eta^2\|\widetilde{\nabla}_\theta J(\theta_t)\|_2^2$$

$$\Rightarrow \mathbb{E}\left[\eta \nabla_\theta J(\theta_t)^\top \nabla_\theta J(\theta_t)\right] \le \mathbb{E}\left[J(\theta_t) - J(\theta_{t+1})\right] + \frac{\beta}{2}\eta^2\sigma^2$$

Set $\eta = \sqrt{M/(\beta\sigma^2 T)}$

$$\Rightarrow \eta\mathbb{E}\left[\sum_t \|\nabla_\theta J(\theta_t)\|_2^2\right] \le \sum_t \mathbb{E}\left[J(\theta_t) - J(\theta_{t+1})\right] + \frac{\beta T}{2}\eta^2\sigma^2 \ \Rightarrow \frac{1}{T}\sum_t \|\nabla_\theta J(\theta_t)\|_2^2 \le \frac{1}{\eta T}M + \frac{\beta}{2}\eta\sigma^2$$

Outline for today

1. Recap on Gradient descent and stochastic gradient descent

2. Warm up: computing gradient using importance weighting

3. Policy Gradient formulations

Importance Sampling (and the Likelihood Ratio Method)

For $J(\theta) = \mathbb{E}_{x \sim P_\theta}\left[f(x)\right]$, our goal is to accurately compute $\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_{x \sim P_\theta} f(x)$.

- Suppose:
  - $J(\theta)$ is "difficult" to compute.
  - P_θ is "easy" to compute.
  - We have a distribution ρ, that is easy to sample from and where $\max_x P_\theta(x)/\rho(x) < \infty$

$$\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_{x \sim P_\theta} f(x) = \nabla_\theta \mathbb{E}_{x \sim \rho} \frac{P_\theta(x)}{\rho(x)} f(x) \ = \mathbb{E}_{x \sim \rho} \frac{\nabla_\theta P_\theta(x)}{\rho(x)} f(x) \ \approx \frac{1}{N} \sum_{i=1}^{N} \frac{\nabla_\theta P_\theta(x_i)}{\rho(x_i)} f(x_i)$$

To compute gradient at θ_0: $\nabla_\theta J(\theta_0)$ (in short of $\nabla_\theta J(\theta)\,|_{\theta=\theta_0}$)

By setting the sampling distribution $\rho = P_{\theta_0}$

$$\nabla_\theta J(\theta_0) = \mathbb{E}_{x \sim P_{\theta_0}}\left[\nabla_\theta \ln P_{\theta_0}(x) \cdot f(x)\right]$$

Importance Sampling (and the Likelihood Ratio Method)

To compute gradient at θ_0: $\nabla_\theta J(\theta_0)$ (in short of $\nabla_\theta J(\theta)\,|_{\theta=\theta_0}$)

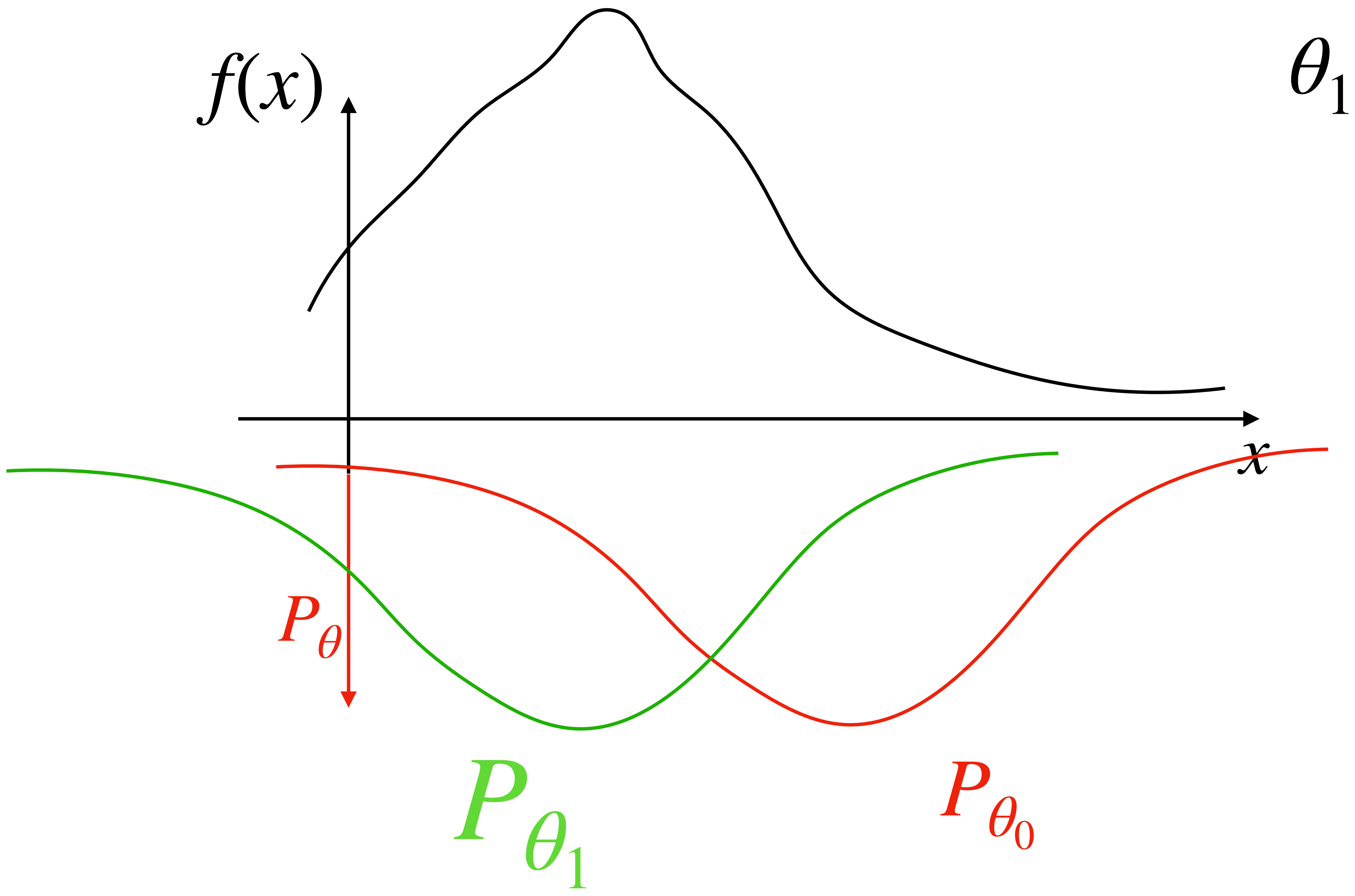
By setting the sampling distribution $\rho = P_{\theta_0}$

$$\nabla_\theta J(\theta_0) = \mathbb{E}_{x\sim\rho}\frac{\nabla_\theta P_{\theta_0}(x)}{\rho(x)}f(x) = \mathbb{E}_{x\sim P_{\theta_0}}\left[\nabla_\theta \ln P_{\theta_0}(x)\cdot f(x)\right]$$

Example and Intuition

$$\nabla_\theta J(\theta)\,|_{\theta=\theta_0} = \mathbb{E}_{x\sim P_{\theta_0}} \nabla_\theta \ln P_{\theta_0}(x) \cdot f(x)$$

$$\theta_1 = \theta_0 + \eta \nabla_\theta J(\theta_0)$$



Update distribution (via updating θ) such that P_θ has high probability mass at regions where $f(x)$ is large

Using same idea, now let's move on to RL…

Outline for today

1. Recap on Gradient descent and stochastic gradient descent

2. Warm up: computing gradient using importance weighting

3. Policy Gradient formulations

Policy Gradient: Examples of Policy Parameterization (discrete actions)

Recall that we consider parameterized policy $\pi_\theta(\cdot \mid s) \in \Delta(A), \forall s$

1. Softmax Policy for discrete MDPs:

$$\theta_{s,a} \in \mathbb{R}, \forall s, a \in S \times A$$

$$\pi_\theta(a \mid s) = \frac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})}$$

2. Softmax linear Policy (We will try this in HW2)

Feature vector $\phi(s, a) \in \mathbb{R}^d$, and parameter $\theta \in \mathbb{R}^d$

$$\pi_\theta(a \mid s) = \frac{\exp(\theta^\top \phi(s, a))}{\sum_{a'} \exp(\theta^\top \phi(s, a'))}$$

3. Neural Policy:

Neural network $f_\theta : S \times A \mapsto \mathbb{R}$

$$\pi_\theta(a \mid s) = \frac{\exp(f_\theta(s, a))}{\sum_{a'} \exp(f_\theta(s, a'))}$$

Derivation of Policy Gradient: REINFORCE

$$\tau = \{s_0, a_0, s_1, a_1, \ldots\}$$

$$\rho_\theta(\tau) = \mu(s_0)\pi_\theta(a_0 \mid s_0)P(s_1 \mid s_0, a_0)\pi_\theta(a_1 \mid s_1)\ldots$$

$$J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \underbrace{\left[\sum_{h=0}^{\infty} \gamma^h r(s_h, a_h) \right]}_{R(\tau)}$$

Adjust policy such that larger reward traj has higher likelihood

$$\nabla_\theta J(\theta_0) = \mathbb{E}_{\tau \sim \rho_{\theta_0}(\tau)} \left[\nabla_\theta \ln \rho_{\theta_0}(\tau) R(\tau) \right]$$

$$= \mathbb{E}_{\tau \sim \mu_{\theta_0}(\tau)} \left[\nabla_\theta \left(\ln \rho(s_0) + \ln \pi_{\theta_0}(a_0 \mid s_0) + \ln P(s_1 \mid s_0, a_0) + \ldots \right) R(\tau) \right]$$

$$= \mathbb{E}_{\tau \sim \rho_{\theta_0}(\tau)} \left[\nabla_\theta \left(\ln \pi_{\theta_0}(a_0 \mid s_0) + \ln \pi_{\theta_0}(a_1 \mid s_1) \ldots \right) R(\tau) \right] = \mathbb{E}_{\tau \sim \rho_{\theta_0}(\tau)} \left[\left(\sum_{h=0}^{\infty} \nabla_\theta \ln \pi_{\theta_0}(a_h \mid s_h) \right) R(\tau) \right]$$

Summary so far for Policy Gradients

We derived the most basic PG formulation:

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \left[\left(\sum_{h=0}^{\infty} \nabla_\theta \ln \pi_\theta(a_h \mid s_h) \right) R(\tau) \right]$$

Increase the likelihood of sampling an trajectory with high total reward

Obtaining a sample $\widetilde{\nabla}_\theta J(\theta)$ for REINFORCE (for this approach)

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \left[\left(\sum_{h=0}^{\infty} \nabla_\theta \ln \pi_\theta(a_h \mid s_h) \right) R(\tau) \right], \text{ where } R(\tau) = \sum_{h=0}^{\infty} \gamma^h r(s_h, a_h)$$

For finite horizon MDP (sometimes used with PG):

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \left[\left(\sum_{h=0}^{H-1} \nabla_\theta \ln \pi_\theta(a_h \mid s_h) \right) R(\tau) \right]$$

where $R(\tau) = \sum_{h=0}^{H-1} r(s_h, a_h)$

Increase the likelihood of sampling an trajectory with high total reward

A improved PG formulation, for sampling (for the discounted setting)

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \left[\left(\sum_{h=0}^{\infty} \nabla_\theta \ln \pi_\theta(a_h \mid s_h) \right) R(\tau) \right]$$

$$= \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \left[\left(\sum_{h=0}^{\infty} \nabla_\theta \ln \pi_\theta(a_h \mid s_h) \sum_{t=h}^{\infty} \gamma^t r_t \right) \right]$$

$$= \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \left[\left(\sum_{h=0}^{\infty} \nabla_\theta \ln \pi_\theta(a_h \mid s_h) \gamma^h Q^{\pi_\theta}(s_h, a_h) \right) \right]$$

Intuition: Change action distribution at h only affects rewards later on…)

Exercise: Show this simplified version is equivalent to REINFORCE

A improved PG formulation, for sampling (for the discounted setting)

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \left[\left(\sum_{h=0}^{\infty} \nabla_\theta \ln \pi_\theta(a_h \mid s_h) \sum_{t=h}^{\infty} \gamma^t r_t \right) \right]$$

Further simplification on PG (e.g., for finite horizon)

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \left[\left(\sum_{h=0}^{H-1} \nabla_\theta \ln \pi_\theta(a_h \mid s_h) \cdot \sum_{\tau=h}^{H-1} r(s_\tau, a_\tau) \right) \right]$$

(Change action distribution at h only affects rewards later on…)

Exercise:

Show this simplified version is equivalent to REINFORCE

Summary for today

1. Importance Weighting (the likelihood ratio method)
2. The Policy Gradient:
   REINFORCE (a direct application of the likelihood ratio method)

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \rho_\theta(\tau)} \left[\left(\sum_{h=0}^{\infty} \nabla_\theta \ln \pi_\theta(a_h \mid s_h) \right) R(\tau) \right]$$

3. SGAscent With unbiased estimate of $\nabla_\theta J(\theta)$, SGA(hopefully) converges to a local optimal policy.

1-minute feedback form: https://bit.ly/3RHtlxy