CS 4/5789 14 March 2022 Prof Sarah Dean

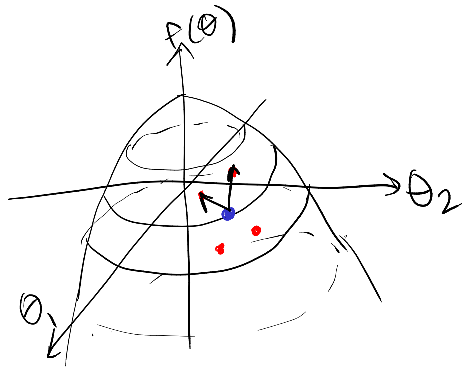
Lecture 14: Policy Gradients

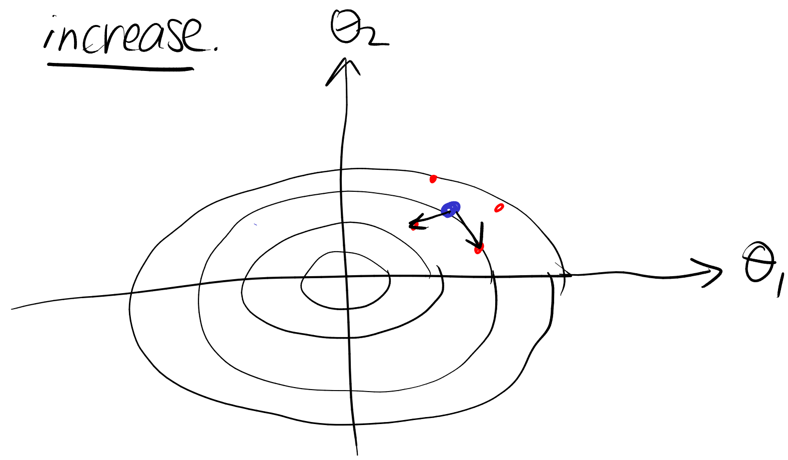
1) Derivative-Free Optimization

How can we find maxima only using function evaluation? I.e. we can query $f(\theta): \mathbb{R}^d \to \mathbb{R}$ but not $\nabla f(\theta)$.

Goal: find a descent direction

Simple idea: randomly test a few directions & see which lead to increase.





There are many variations of this simple idea: simulated annealing, cross-entropy method, genetic algorithms, evolutionary strategies. They differ in how random samples are aggregated into update step.

We will cover methods that use samples to construct gradient estimates.

A) Random Search

Recall when we discussed iLQR the finite difference approximation:

$$f'(x) \approx \frac{f(x+\delta) - f(x-\delta)}{2\delta}$$

This idea can help us build an approximation of the gradient based only on function evaluation.

← direction of steepest ascent

For vector functions: $\theta \in \mathbb{R}^d$

$$\langle \nabla f(\theta), v \rangle \approx \frac{f(\theta+\delta v) - f(\theta - \delta v)}{2\delta}$$

Alg: Random Search

initialize θ_0

for $t = 0, 1, \ldots$

 sample $v_1, \ldots v_N \sim N(0, I)$

 update $\theta_{t+1} = \theta_t + \frac{\alpha}{2\delta N} \sum_{k=1}^{N} \left(f(\theta_t + \delta v_k) - f(\theta_t - \delta v_k) \right) v_k$

We can understand this as stochastic gradient ascent:

$$\mathbb{E}\left(\left(f(\theta + \delta v_k) - f(\theta - \delta v_k) \right) v_k \right) \approx \mathbb{E}\left(2\delta \nabla f(\theta)^T v_k \cdot v_k \right)$$

$$= 2\delta \, \mathbb{E}\left[v_k v_k^T \right] \nabla f(\theta)$$

$$= 2\delta \nabla f(\theta)$$

This method samples/searches in <u>parameter</u> space. (θ)

B) Importance Weighting

Distribution trick: in general, we can write:

$$f(\theta) = \mathop{\mathbb{E}}_{x \sim P_\theta} \left[h(x) \right]$$

for some class of distributions P_θ

(In RL setting, P_θ could represent the distribution over trajectories induced by π_θ.)

Now suppose a sampling distribution ρ where $\frac{P_\theta(x)}{\rho(x)} < \infty$.

$$\mathop{\mathbb{E}}_{x \sim P_\theta} \left[h(x) \right] = \sum_x h(x) P_\theta(x) \cdot \frac{\rho(x)}{\rho(x)} = \mathop{\mathbb{E}}_{x \sim \rho} \left[\underbrace{\frac{P_\theta(x)}{\rho(x)}}_{\text{"importance weights"}} h(x) \right].$$

This allows us to write the gradient:

$$\nabla f(\theta) = \mathop{\mathbb{E}}_{x \sim p} \left[\frac{\nabla_\theta P_\theta(x)}{p(x)} h(x) \right]$$

This is true for <u>any</u> $p(x)$. If we pick $p(x) = P_\theta(x)$ then

$$\nabla_\theta f(\theta) = \mathop{\mathbb{E}}_{x \sim P_\theta(x)} \left[\frac{\nabla_\theta P_\theta(x)}{P(x)} h(x) \right] = \mathop{\mathbb{E}}_{x \sim P_\theta(x)} \left[\nabla_\theta \log(P_\theta(x)) \, h(x) \right]$$

Now if $P_\theta(x)$ factors, $\log(P_\theta(x))$ will be sum of factors, and the gradient will depend only on factors which depend on optimization variable (This is very useful for policy optimization)

Therefore, our stochastic maximization algorithm:

<u>Alg: sampling-DFO</u>

initialize θ_0

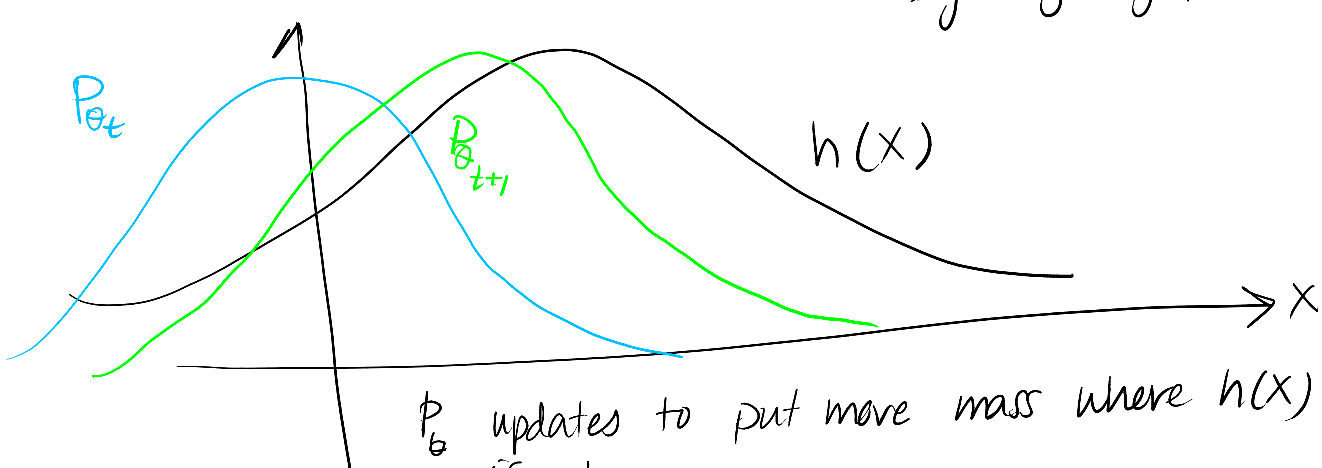
for $t = 0, 1, \ldots$

sample $x_i \sim P_{\theta_t}$ and observe $h(x)$ $i = 1, \ldots, N$

$$\theta_{t+1} = \theta_t + \frac{\alpha}{N} \sum_{i=1}^{N} \nabla_{\theta_t} \log(P_{\theta_t}(x_i)) \, h(x_i)$$

This method samples in x-space rather than parameter space.

e.g. trajectory space



P_θ updates to put more mass where $h(x)$ is large

2) Policy Optimization via Simple Random Search

Recall the RL setting:

MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, P, r, \gamma\}$ with P, r unknown

parametrized policy π_θ, $\theta \in \mathbb{R}^d$ (e.g. weights of neural networks)

objective function: $J(\theta) = \mathbb{E}_{s_0 \sim \mu_0}\left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| \begin{matrix} P, r, \\ \pi_\theta \end{matrix}\right]$

observe "rollout" of π_θ: trajectory $\tau = (s_0, a_0, s_1, \dots)$ and rewards $(r_0, r_1, \dots)$

means we can "sample" $J(\theta) = \mathbb{E}_\tau(R(\tau))$ & observe τ, $R(\tau)$

Meta-Algorithm: Derivative-Free SGA

initialize θ_0

for $t = 0, 1, \dots$

1) Collect rollouts using θ_t
2) Compute estimate g_t of $\nabla_{\theta_t} J(\theta)$ using rollouts
3) $\theta_{t+1} = \theta_t + \alpha g_t$

The rest of lecture: 3 ways to estimate $\nabla J(\theta)$ using rollouts.

Simple Random Search

Based on the "random search" idea.

1) Collect rollouts: τ^+ & τ^- with $\pi_{\theta_t + \delta v}$ & $\pi_{\theta_t - \delta v}$ for $v \sim \mathcal{N}(0, I)$, small $\delta > 0$
2) Compute estimate: $g_t = \frac{1}{2\delta}(R(\tau^+) - R(\tau^-))\, v$

3) Policy Gradient (PG) from Trajectories (REINFORCE)

Another approach based on "importance weighting" derivation.

$\tau = (s_0, a_0, s_1, \dots)$ and $p_\theta(\tau) = \mu_0(s_0)\pi_\theta(a_0|s_0)P(s_1|s_0,a_0)\pi_\theta(a_1|s_1)\cdots$

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

<u>Claim:</u> for $\tau \sim p_\theta(\tau)$ (i.e. τ observed from rollout with π_θ)

$$g = \sum_{t=0}^{\infty} \nabla_\theta[\log(\pi_\theta(a_t|s_t))]R(\tau)$$ is an unbiased estimate of $\nabla J(\theta)$.

<u>Proof:</u> Using derivation from earlier in lecture with $f \leftarrow J$, $h \leftarrow R$, $x \leftarrow \tau$:

$$\nabla J(\theta) = \mathbb{E}_{\tau\sim p_\theta}\left[\nabla_\theta[\log(p_\theta(\tau))]R(\tau)\right]$$

$$\begin{aligned}\nabla_\theta \log(p_\theta(\tau)) &= \nabla_\theta\left[\log(\mu_0(s_0)\pi_\theta(a_0|s_0)P(s_1|s_0,a_0)\pi(a_1|s_1)\dots)\right] \\ &= \nabla_\theta\left[\cancel{\log(\mu_0(s_0))} + \sum_{t=0}^{\infty}\log(\pi_\theta(a_t|s_t)) + \cancel{\log(P(s_{t+1}|s_t,a_t))}\right] \\ &= \sum_{t=0}^{\infty}\nabla_\theta \log(\pi_\theta(a_t|s_t))\end{aligned}$$

no θ dependence

$\square$

$\nabla_\theta \log p_\theta(\tau)$ ends up not depending at all on unknown transition function P!

<u>REINFORCE:</u>
1) collect rollout τ with π_θ
2) compute estimate

$$g = \sum_{t=0}^{\infty}\nabla_\theta \log(\pi_\theta(a_t|s_t))R(\tau)$$

4) Policy Gradient with value functions

PG w/ trajectories often has high variance. An alternative commonly used in practice uses an alternative estimate using Q functions.

<u>Claim:</u> for $s, a \sim d^{\pi_\theta}_{\mu_0}$,

$$g = \frac{1}{1-\gamma} \nabla_\theta \log(\pi_\theta(a|s))\, Q^{\pi_\theta}(s,a)$$

is an unbiased estimate of $\nabla J(\theta)$

<u>Proof:</u>

$$\nabla J(\theta) = \nabla_\theta \mathop{\mathbb{E}}_{s_0 \sim \mu}\left[V^{\pi_\theta}(s_0)\right] \qquad \text{value fn. def.}$$

$$= \mathop{\mathbb{E}}_{s_0 \sim \mu}\left[\nabla_\theta \mathop{\mathbb{E}}_{a_0 \sim \pi_\theta(s_0)}\left[Q^{\pi_\theta}(s_0, a_0)\right]\right] \qquad s_0 \text{ doesn't depend on } \theta \text{ \& } Q \text{ fn. def.}$$

$$\nabla_\theta \mathop{\mathbb{E}}_{a_0 \sim \pi_\theta(s_0)}\left[Q^{\pi_\theta}(s_0,a_0)\right] = \sum_{a_0 \in \mathcal{A}} \nabla_\theta \left[\pi(a_0|s_0)\, Q^{\pi_\theta}(s_0,a_0)\right] \qquad \text{defn. of expectation}$$

product rule

$$= \sum_{a_0 \in \mathcal{A}} (\nabla_\theta \pi(a_0|s_0))\, Q^{\pi_\theta}(s_0,a_0) + \pi(a_0|s_0)\, \nabla_\theta Q^{\pi_\theta}(s_0,a_0)$$

importance weighting trick

$r(s_0, a_0)$ doesn't depend on θ

$$= \mathop{\mathbb{E}}_{a_0 \sim \pi_\theta(s_0)}\left[(\nabla_\theta \log \pi_\theta(a_0|s_0))\, Q^{\pi_\theta}(s_0,a_0)\right] + \gamma \mathop{\mathbb{E}}_{\substack{a_0 \sim \pi_\theta(s_0) \\ s_1 \sim P(s_0,a_0)}}\left[\nabla_\theta V^{\pi_\theta}(s_1)\right]$$

$$\nabla J(\theta) = \mathop{\mathbb{E}}_{\substack{s_0 \sim \mu_0 \\ a_0 \sim \pi_\theta(s_0)}}\left[\nabla_\theta \log \pi_\theta(a_0|s_0)\, Q^{\pi_\theta}(s_0,a_0)\right] + \gamma \mathop{\mathbb{E}}_{s_1 \sim \mathbb{P}_1^{\pi_\theta}}\left[\nabla_\theta V^{\pi_\theta}(s_1)\right]$$

$$\mathop{\mathbb{E}}_{s_0 \sim \mu_0}\left[\nabla_\theta V^{\pi_\theta}(s_0)\right]$$

We can iterate!

$$\nabla J(\theta) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{s_t, a_t \sim \mathbb{P}_t^{\pi_\theta}} \left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot Q^{\pi_\theta}(s_t, a_t) \right]$$

expanding expectation
$$= \sum_{t=0}^{\infty} \sum_{\substack{s \in \mathcal{S} \\ a \in \mathcal{A}}} \mathbb{P}_t^{\pi_\theta}(s, a; \mu_0)\, \gamma^t \cdot \nabla_\theta \log \pi_\theta(a|s) \cdot Q^{\pi_\theta}(s, a)$$

definition of $d_{\mu_0}^{\pi_\theta}$
$$= \frac{1}{1-\gamma} \mathbb{E}_{s, a \sim d_{\mu_0}^{\pi_\theta}} \left[\nabla_\theta \log \pi_\theta(a|s) \cdot Q^{\pi_\theta}(s, a) \right]$$

$\square$

One final gradient estimate: $s, a \sim d_{\mu_0}^{\pi_\theta}$

$$\frac{1}{1-\gamma} \nabla_\theta \log \pi_\theta(a|s) \cdot \left(Q^{\pi_\theta}(s, a) - b(s) \right)$$

Baseline function $b(s)$ further helps in variance reduction. Most common $b(s) = V^{\pi_\theta}(s)$ resuts in <u>advantage function</u>-<u>based PG</u>

$$A^{\pi_\theta}(s, a) = Q^{\pi_\theta}(s, a) - V^{\pi_\theta}(s).$$

Exercise: show that $\mathbb{E}_{a \sim \pi_\theta(s)} \left[\nabla_\theta \log \pi_\theta(a|s) \cdot b(s) \right] = 0$

for any action-independent baseline.