Basic RL Settings

Monte-Carlo policy evaluation

- Given π, estimate $v^\pi := \mathbb{E}_{s \sim \mu}[V^\pi(s)]$

Monte-Carlo policy evaluation

- Given π, estimate $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$
- Alg outputs v; evaluated by $|v - v^\pi|$

Monte-Carlo policy evaluation

- Given π, estimate $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$
- Alg outputs v; evaluated by $|v - v^\pi|$
- Data: trajectories starting from $s_1 \sim \mu$ using π (i.e., $a_h = \pi(s_h)$)

$$\{(s_1^{(i)}, a_1^{(i)}, r_1^{(i)}, s_2^{(i)}, \ldots, s_H^{(i)}, a_H^{(i)}, r_H^{(i)})\}_{i=1}^n$$

(for simplicity, assume process terminates in H time steps)

Monte-Carlo policy evaluation

- Given π, estimate $v^\pi := \mathbb{E}_{s \sim \mu}[V^\pi(s)]$
- Alg outputs v; evaluated by $|v - v^\pi|$
- Data: trajectories starting from $s_1 \sim \mu$ using π (i.e., $a_h = \pi(s_h)$)

$$\{(s_1^{(i)}, a_1^{(i)}, r_1^{(i)}, s_2^{(i)}, \ldots, s_H^{(i)}, a_H^{(i)}, r_H^{(i)})\}_{i=1}^n$$

(for simplicity, assume process terminates in H time steps)

- Algorithm: output $\frac{1}{n} \sum_{i=1}^n \sum_{h=1}^H \gamma^{h-1} r_h^{(i)}$

Monte-Carlo policy evaluation

- Given π, estimate $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$
- Alg outputs v; evaluated by $|v - v^\pi|$
- Data: trajectories starting from $s_1 \sim \mu$ using π (i.e., $a_h = \pi(s_h)$)

$$\{(s_1^{(i)}, a_1^{(i)}, r_1^{(i)}, s_2^{(i)}, \ldots, s_H^{(i)}, a_H^{(i)}, r_H^{(i)})\}_{i=1}^n$$

(for simplicity, assume process terminates in H time steps)

- Algorithm: output $\frac{1}{n}\sum_{i=1}^n \sum_{h=1}^H \gamma^{h-1} r_h^{(i)}$
- Guarantee: w.p. at least $1 - \delta$, $|v - v^\pi| \leq \frac{R_{\max}}{1-\gamma}\sqrt{\frac{1}{2n}\ln\frac{2}{\delta}}$
    - Depends on value range & sample size
    - No dependence on anything else, e.g., state/action spaces

Monte-Carlo optimization

- Want to optimize $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$; have parameterized π_θ.

Monte-Carlo optimization

- Want to optimize $v^{\pi} := \mathbb{E}_{s \sim \mu}[V^{\pi}(s)]$; have parameterized π_θ.
- For each θ, roll-out trajectories using π_θ to estimate v^{π_θ}

Monte-Carlo optimization

- Want to optimize $v^{\pi} := \mathbb{E}_{s \sim \mu}[V^{\pi}(s)]$; have parameterized π_θ.
- For each θ, roll-out trajectories using π_θ to estimate v^{π_θ}
- Pick a bunch of θ, estimate return, pick the best

Monte-Carlo optimization

- Want to optimize $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$; have parameterized π_θ.
- For each θ, roll-out trajectories using π_θ to estimate v^{π_θ}
- Pick a bunch of θ, estimate return, pick the best
- In general, reduce to 0-th order optimization
  - e.g., CMA-ES for RL

Monte-Carlo optimization

- Want to optimize $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$; have parameterized π_θ.
- For each θ, roll-out trajectories using π_θ to estimate v^{π_θ}
- Pick a bunch of θ, estimate return, pick the best
- In general, reduce to 0-th order optimization
  - e.g., CMA-ES for RL
- Guarantee: depends on optimization

Model-based RL w/ sampling oracle

- Want to optimize $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$, tabular state space

Model-based RL w/ sampling oracle

- Want to optimize $v^\pi := \mathbb{E}_{s \sim \mu}[V^\pi(s)]$, tabular state space
- Sampling oracle: can generate $s' \sim P(s, a)$ for any (s, a)

Model-based RL w/ sampling oracle

- Want to optimize $v^\pi := \mathbb{E}_{s \sim \mu}[V^\pi(s)]$, tabular state space
- Sampling oracle: can generate $s' \sim P(s, a)$ for any (s, a)
- Use a total of $n|S \times A|$ samples: n samples per (s, a)

Model-based RL w/ sampling oracle

- Want to optimize $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$, tabular state space
- Sampling oracle: can generate $s' \sim P(s, a)$ for any (s, a)
- Use a total of $n|S \times A|$ samples: n samples per (s, a)
- Estimate transition probabilities

Model-based RL w/ sampling oracle

- Want to optimize $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$, tabular state space
- Sampling oracle: can generate $s' \sim P(s, a)$ for any (s, a)
- Use a total of $n|S \times A|$ samples: n samples per (s, a)
- Estimate transition probabilities
- Plan in the estimated model

Model-based RL w/ sampling oracle

- Want to optimize $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$, tabular state space
- Sampling oracle: can generate $s' \sim P(s, a)$ for any (s, a)
- Use a total of $n|S\times A|$ samples: n samples per (s, a)
- Estimate transition probabilities
- Plan in the estimated model
- Guarantee (will analyze later) depends on:
  - Sample size, value range
  - Horizon (error compounding)
  - $|S\times A|$ (“curse of dimensionality”)

Categorization of RL settings

What's the goal?

- Policy evaluation: given π, estimate its value.

What's the goal?

- Policy evaluation: given π, estimate its value.
  - Estimate $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$
    e.g., alg outputs v; evaluated by $|v - v^\pi|$

What's the goal?

- Policy evaluation: given π, estimate its value.
  - Estimate $v^\pi := \mathbb{E}_{s \sim \mu}[V^\pi(s)]$
    e.g., alg outputs v; evaluated by $|v - v^\pi|$
  - Estimate the entire value function
    e.g., alg outputs V; evaluated by $\|V - V^\pi\|_\infty$

What's the goal?

- Policy evaluation: given π, estimate its value.
  - Estimate $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$
    e.g., alg outputs v; evaluated by $|v - v^\pi|$
  - Estimate the entire value function
    e.g., alg outputs V; evaluated by $||V - V^\pi||_\infty$
- Policy optimization

What's the goal?

- Policy evaluation: given π, estimate its value.
  - Estimate $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$
    e.g., alg outputs v; evaluated by $|v - v^\pi|$
  - Estimate the entire value function
    e.g., alg outputs V; evaluated by $\|V - V^\pi\|_\infty$
- Policy optimization
  - Optimize $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$
    e.g., alg outputs π; evaluated by $v^* - v^\pi$

What's the goal?

- Policy evaluation: given π, estimate its value.
  - Estimate $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$
    e.g., alg outputs v; evaluated by $|v - v^\pi|$
  - Estimate the entire value function
    e.g., alg outputs V; evaluated by $\|V - V^\pi\|_\infty$
- Policy optimization
  - Optimize $v^\pi := \mathbb{E}_{s\sim\mu}[V^\pi(s)]$
    e.g., alg outputs π; evaluated by $v^* - v^\pi$
  - No particular initial state
    e.g., alg outputs π; evaluated by $\|V^* - V^\pi\|_\infty$

Where do data come from?

- Passively given

Where do data come from?

- Passively given
  - For policy optimization: usually needs to be exploratory
    e.g., fixed number of samples from each (s,a)

Where do data come from?

- Passively given
    - For policy optimization: usually needs to be exploratory e.g., fixed number of samples from each (s,a)
    - For policy evaluation of π
        - Data generated w/ π: on-policy (somewhat easy)
        - Data generated w/ π': off-policy (requires counter-factual reasoning)

Where do data come from?

- Passively given
  - For policy optimization: usually needs to be exploratory e.g., fixed number of samples from each (s,a)
  - For policy evaluation of π
    - Data generated w/ π: on-policy (somewhat easy)
    - Data generated w/ π': off-policy (requires counter-factual reasoning)
- Collect own data

Where do data come from?

- Passively given
  - For policy optimization: usually needs to be exploratory e.g., fixed number of samples from each (s,a)
  - For policy evaluation of π
    - Data generated w/ π: on-policy (somewhat easy)
    - Data generated w/ π': off-policy (requires counter-factual reasoning)
- Collect own data
  - Can ask for sample from any (s,a): powerful, unrealistic

Where do data come from?

- Passively given
  - For policy optimization: usually needs to be exploratory e.g., fixed number of samples from each (s,a)
  - For policy evaluation of π
    - Data generated w/ π: on-policy (somewhat easy)
    - Data generated w/ π': off-policy (requires counter-factual reasoning)
- Collect own data
  - Can ask for sample from any (s,a): powerful, unrealistic
  - Can only roll-out trajectories: exploration!

What’s the basic principle?

Monte-Carlo

What’s the basic principle?

Monte-Carlo

- e.g., roll-out trajectories to estimate return
- e.g., policy gradient, evolutionary strategies
- e.g., Monte-Carlo tree search

What's the basic principle?

Monte-Carlo

- e.g., roll-out trajectories to estimate return
- e.g., policy gradient, evolutionary strategies
- e.g., Monte-Carlo tree search

- Often no dependence on state space
- No compounding error
- Learning signals can be sparse
- Local optimal (sometimes)

What’s the basic principle?

Dynamic programming

What’s the basic principle?

Dynamic programming

- e.g., use data to approximately solve Bellman Equation
- e.g., estimate a transition model, then do planning

What’s the basic principle?

Dynamic programming

- e.g., use data to approximately solve Bellman Equation
- e.g., estimate a transition model, then do planning

- “Curse of dimensionality” (dependence on state space)
- Error compounds over time
- Leverage immediate signals to learn
- Global optimality (sometimes)

Manageable state spaces?

Yes: tabular RL

Manageable state spaces?

Yes: tabular RL

No: function approximation. Approximate what?

- Model?
- Value function?
- Policy?

Quick Recap of MDP results

Review of the lectures so far…

Notations

- $\forall f \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$, let $\pi_f = (s \mapsto \arg\max_{a\in\mathcal{A}} f(s,a))$, $V_f = (s \mapsto \max_{a\in\mathcal{A}} f(s,a))$

Review of the lectures so far…

Notations

- $\forall f \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$, let $\pi_f = (s \mapsto \arg\max_{a\in\mathcal{A}} f(s,a))$, $V_f = (s \mapsto \max_{a\in\mathcal{A}} f(s,a))$
- Bellman (optimality) op: $(\mathcal{T}f)(s,a) := R(s,a) + \gamma\langle P(s,a), V_f\rangle$

Review of the lectures so far…

Notations

- $\forall f \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$, let $\pi_f = (s \mapsto \arg\max_{a\in\mathcal{A}} f(s,a))$, $V_f = (s \mapsto \max_{a\in\mathcal{A}} f(s,a))$
- Bellman (optimality) op: $(\mathcal{T}f)(s,a) := R(s,a) + \gamma\langle P(s,a), V_f\rangle$

Results

- $\mathcal{T}$ is a γ-contraction under ℓ_∞ : $\|\mathcal{T}f_1 - \mathcal{T}f_2\|_\infty \le \gamma\|f_1 - f_2\|_\infty$ (therefore value iteration enjoys exponential convergence)

Review of the lectures so far…

Notations

- $\forall f \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$, let $\pi_f = (s \mapsto \arg\max_{a\in\mathcal{A}} f(s,a))$, $V_f = (s \mapsto \max_{a\in\mathcal{A}} f(s,a))$
- Bellman (optimality) op: $(\mathcal{T}f)(s,a) := R(s,a) + \gamma\langle P(s,a), V_f\rangle$

Results

- $\mathcal{T}$ is a γ-contraction under ℓ_∞ : $\|\mathcal{T}f_1 - \mathcal{T}f_2\|_\infty \le \gamma\|f_1 - f_2\|_\infty$ (therefore value iteration enjoys exponential convergence)
- Loss of acting greedily according to an approximate Q-value function f : $\|V^\star - V^{\pi_f}\|_\infty \le 2\frac{\|Q^\star - f\|_\infty}{1-\gamma}$

Review of the lectures so far…

Notations

- $\forall f \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$, let $\pi_f = (s \mapsto \arg\max_{a\in\mathcal{A}} f(s,a))$, $V_f = (s \mapsto \max_{a\in\mathcal{A}} f(s,a))$
- Bellman (optimality) op: $(\mathcal{T}f)(s,a) := R(s,a) + \gamma\langle P(s,a), V_f\rangle$

Results

- $\mathcal{T}$ is a γ-contraction under ℓ_∞ : $\|\mathcal{T}f_1 - \mathcal{T}f_2\|_\infty \le \gamma\|f_1 - f_2\|_\infty$ (therefore value iteration enjoys exponential convergence)
- Loss of acting greedily according to an approximate Q-value function f: $\|V^\star - V^{\pi_f}\|_\infty \le 2\frac{\|Q^\star - f\|_\infty}{1-\gamma}$
- Advantage decomposition of value difference:

$$V^{\pi'}(s) - V^{\pi}(s) = \frac{1}{1-\gamma}\mathbb{E}_{s'\sim\eta_s^{\pi'}}[A^{\pi}(s',\pi')]$$

$$Q^{\pi}(s',\pi'(s')) - V^{\pi}(s')$$

Useful tools

- Hoeffding's inequality: for independent r.v.'s $X_1, \ldots, X_n$ bounded in $[a, b]$, w.p. at least $1 - \delta$, the empirical average deviates from the true mean by at most $(b-a)\sqrt{\frac{1}{2n}\ln\frac{2}{\delta}}$.

Useful tools

- Hoeffding's inequality: for independent r.v.'s $X_1, \ldots, X_n$ bounded in $[a, b]$, w.p. at least $1 - \delta$, the empirical average deviates from the true mean by at most $(b-a)\sqrt{\frac{1}{2n}\ln\frac{2}{\delta}}$.
- Union bound: Pr[union of events] $\leq$ sum of Pr[event].

Useful tools

- Hoeffding's inequality: for independent r.v.'s $X_1, \ldots, X_n$ bounded in $[a, b]$, w.p. at least 1 - δ, the empirical average deviates from the true mean by at most $(b-a)\sqrt{\frac{1}{2n}\ln\frac{2}{\delta}}$.
- Union bound: Pr[union of events] ≤ sum of Pr[event].
- Hölder's inequality: for any $u, v \in \mathbb{R}^d$ and any norm, dual norm pair $\|.\|$ and $\|.\|_*$,

$$|\langle u, v\rangle| \leq \|u\| \cdot \|v\|_*$$

Special cases

  - $\|.\|_2$ and $\|.\|_2$
  - $\|.\|_1$ and $\|.\|_\infty$

$$\|x\|_* := \sup_{\|y\|=1} y^\top x$$