Review

What we covered this semester:

1. Basics of Markov Decision Process

2. Planning in MDP: VI and PI

3. Learning: Model-based RL, Policy Optimization, Bandit

4. Imitation

Basics of MDP

Understanding those widely used notations and terminologies:

$$\pi, V^{\pi}, Q^{\pi}, \pi^{\star}, V^{\star}, Q^{\star}$$

$$\mathbb{P}_h^{\pi}(s;\mu), d_{\mu}^{\pi}(s)$$

Basics of MDP

Bellman Equation and Bellman Optimality:

$$\forall s, a: \quad Q^{\pi}(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} V^{\pi}(s')$$

policy evaluation

$$Q^{\star}(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \max_{a'} Q^{\star}(s', a'), \forall s, a$$

DP

Planning in MDP

Q: When P and r are known, we can compute $\pi^\star$ via VI or PI

Algorithm 1: Value Iteration

$$\forall s, a : Q^{t+1}(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \max_{a'} Q^t(s', a')$$

$$Q^{t+1} := \mathcal{T} Q^t$$

Planning in MDP

Q: When P and r are known, we can compute $\pi^\star$ via VI or PI

Algorithm 1: Value Iteration

$$\forall s, a : Q^{t+1}(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \max_{a'} Q^t(s', a')$$

Why it works?

Contraction + $Q^\star$ being a fixed point

Planning in MDP

Q: When P and r are known, we can compute $\pi^\star$ via VI or PI

Algorithm 1: Value Iteration

$$\forall s, a : Q^{t+1}(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \max_{a'} Q^t(s', a')$$

Why it works?

Contraction + $Q^\star$ being a fixed point

$Q^\star = \mathcal{T} Q^\star$

$$\|Q^{t+1} - Q^\star\|_\infty = \|\mathcal{T} Q^t - \mathcal{T} Q^\star\|_\infty \leq \gamma \|Q^t - Q^\star\|_\infty$$

Planning in MDP

Q: When P and r are known, we can compute $\pi^\star$ via VI or PI

Algorithm 2: Policy Iteration

Planning in MDP

Q: When P and r are known, we can compute $\pi^\star$ via VI or PI

Algorithm 2: Policy Iteration

1. Policy Evaluation: $Q^{\pi^t}(s, a), \forall s, a$

Planning in MDP

Q: When P and r are known, we can compute $\pi^\star$ via VI or PI

Algorithm 2: Policy Iteration

1. Policy Evaluation: $Q^{\pi^t}(s, a), \forall s, a$

2. Policy Improvement $\pi^{t+1}(s) := \arg\max_a Q^{\pi^t}(s, a), \forall s;$

Planning in MDP

Q: When P and r are known, we can compute $\pi^\star$ via VI or PI

Algorithm 2: Policy Iteration

1. Policy Evaluation: $Q^{\pi^t}(s,a), \forall s,a$

2. Policy Improvement $\pi^{t+1}(s) := \arg\max_a Q^{\pi^t}(s,a), \forall s;$

Key Properties:

PDL

Monotonic improvement + hit $\pi^\star$ in at most A^S many iterations (hw1)

What we covered this semester:

1. Basics of Markov Decision Process

2. Planning in MDP: VI and PI

3. Learning: Model-based RL, Policy Optimization, Bandit

4. Imitation

Learning:

Q: What we do when (P, r) are not known?

Learning:

Q: What we do when (P, r) are not known?

We considered two learning settings:

Learning:

Q: What we do when (P, r) are not known?

We considered two learning settings:

1. Generative model, i.e., we can reset to any (s, a)

Learning:

Q: What we do when (P, r) are not known?

We considered two learning settings:

1. Generative model, i.e., we can reset to any (s, a)

2. Reset from fixed initial state distribution μ;

Learning:

Q: What we do when (P, r) are not known?

Under generative model setting, we learned a simple model-based RL alg:

Learning:

Q: What we do when (P, r) are not known?

Under generative model setting, we learned a simple model-based RL alg:

1. Model fitting:

$\forall s, a$: collect N next states, $s_i' \sim P(\cdot \mid s, a), i \in [N]$; set

$$\widehat{P}(s' \mid s, a) = \frac{\sum_{i=1}^{N} \mathbf{1}\{s_i' = s'\}}{N};$$

Learning:

Q: What we do when (P, r) are not known?

Under generative model setting, we learned a simple model-based RL alg:

1. Model fitting:

$\forall s, a$: collect N next states, $s'_i \sim P(\cdot \mid s, a), i \in [N]$; set

$$\widehat{P}(s' \mid s, a) = \frac{\sum_{i=1}^{N} \mathbf{1}\{s'_i = s'\}}{N};$$

$\left(\hat{r}(s,a) = \frac{\sum r_i}{N} \right)$

2. Planning w/ the learned model:

$$\widehat{\pi}^\star = \textbf{PI}\left(\widehat{P}, r\right)$$

Assume r is known

Learning:

Q: What we do when (P, r) are not known?

An Important Lemma that is widely used in model-based approach

Simulation Lemma:

$$\underset{\widehat{P}}{\widehat{V}^{\pi}}(s_0) - \underset{P}{V^{\pi}}(s_0) = \frac{\gamma}{1-\gamma}\mathbb{E}_{s,a\sim d^{\pi}_{s_0}}\left[\mathbb{E}_{s'\sim \widehat{P}(\cdot|s,a)}\widehat{V}^{\pi}(s') - \mathbb{E}_{s'\sim P(\cdot|s,a)}\widehat{V}^{\pi}(s')\right]$$

$$\leq \frac{\gamma}{(1-\gamma)^2}\mathbb{E}_{s,a\sim d^{\pi}_{s_0}}\left\| \widehat{P}(\cdot\,|\,s,a) - P(\cdot\,|\,s,a) \right\|_1$$

Learning:

Q: What we do when (P, r) are not known?

Under resetting from μ, we learned policy gradient algorithms

Initial Distribution

Learning:

Q: What we do when (P, r) are not known?

Under resetting from μ, we learned policy gradient algorithms

Given a differentiable parameterized policy $\pi_\theta(a \mid s)$, w/ $\theta \in \mathbb{R}^d$:

Learning:

Q: What we do when (P, r) are not known?

Under resetting from μ, we learned policy gradient algorithms

Given a differentiable parameterized policy $\pi_\theta(a \mid s)$, w/ $\theta \in \mathbb{R}^d$:

REINFORCE:

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

Learning:

Q: What we do when (P, r) are not known?

Under resetting from μ, we learned policy gradient algorithms

Given a differentiable parameterized policy $\pi_\theta(a \mid s)$, w/ $\theta \in \mathbb{R}^d$:

REINFORCE: ✓

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

starting from h

Learning:

Q: What we do when (P, r) are not known?

Under resetting from μ, we learned policy gradient algorithms

Given a differentiable parameterized policy $\pi_\theta(a \mid s)$, w/ $\theta \in \mathbb{R}^d$:

The Q-version:

π: Actor

Q_w : critic → Approximate $Q^{\pi_\theta}(s,a)$

$$\nabla_\theta J(\pi_\theta) = \frac{1}{1-\gamma} \mathbb{E}_{s,a \sim d_\mu^{\pi_\theta}} \left[\nabla_\theta \ln \pi_\theta(a \mid s) \left(Q^{\pi_\theta}(s,a) - b(s) \right) \right]$$

Learning:

Q: What we do when (P, r) are not known?

Under resetting from μ, we learned policy gradient algorithms

Given a differentiable parameterized policy $\pi_\theta(a \mid s)$, w/ $\theta \in \mathbb{R}^d$:

The Natural Policy Gradient:

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

Fisher Info-Matrix

$KL(\rho^{\pi_{\theta_a}} \| \rho^{\pi_{\theta_t}})$

$\approx (\theta - \theta_t)^T F_\theta (\theta - \theta_t)$

PSD

if $\max_{s,a} \frac{d^{\pi^*}_{\mu}(s,a)}{\mu(s,a)} \le C$

Learning:

Q: What we do when (P, r) are not known?

Under resetting from μ, we learned policy gradient algorithms

Given a differentiable parameterized policy $\pi_\theta(a \mid s)$, w/ $\theta \in \mathbb{R}^d$:

The Natural Policy Gradient:

Θ

θ_t

$d(\theta, \theta_t) = (\theta - \theta_t) F_{\theta_t} (\theta - \theta_t)$

Regular GD

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

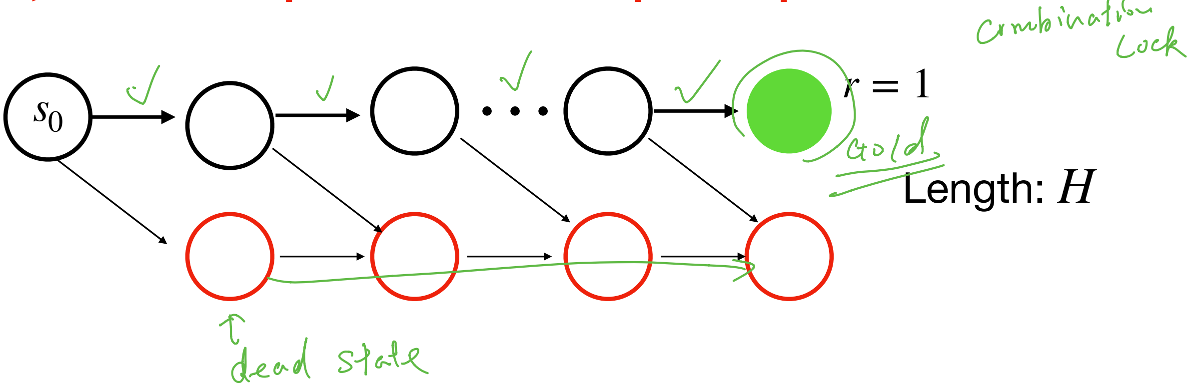
Instead of using Euclidean distance metric, we use local geometry metric

$$d(\theta, \theta_t) := (\theta - \theta_t)^\top F_{\theta_t} (\theta - \theta_t)$$

Learning:

Q: What we do when (P, r) are not known?

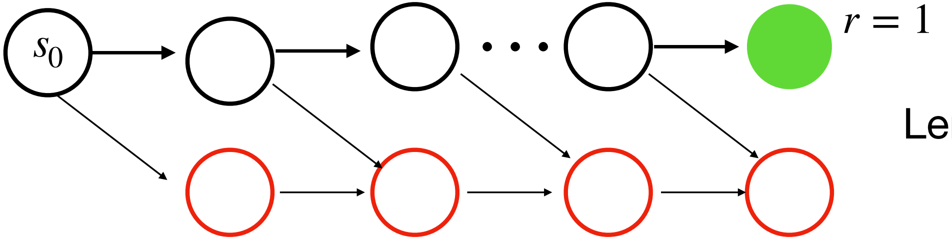
However, PG fails on problems that require exploration..



Learning:

Q: What we do when (P, r) are not known?

However, PG fails on problems that require exploration..



Length: H

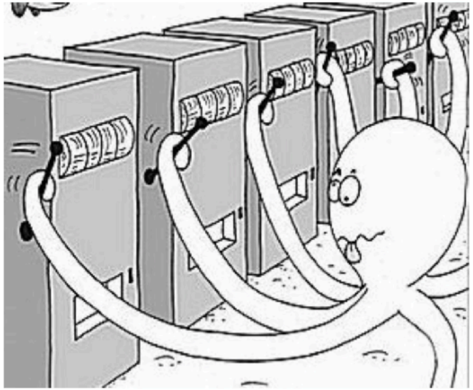
What is the probability of a random policy generating a trajectory that hits the goal?

2^{-H}

Learning:

Q: How to learn efficient (i.e., balance explore and exploit) in Multi-armed Bandit setting:

We have K many arms (or actions): $a_1, \ldots, a_K$



Learning:

Q: How to learn efficient (i.e., balance explore and exploit) in Multi-armed Bandit setting:

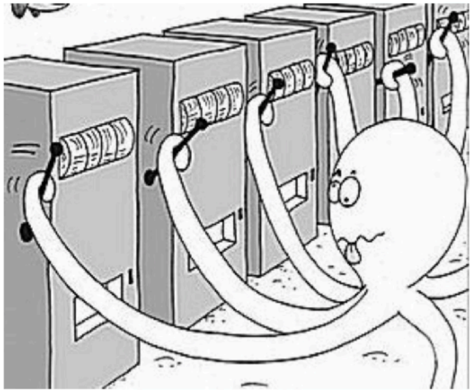
We have K many arms (or actions): $a_1, \ldots, a_K$

Each arm has a unknown reward distribution,

i.e., $\nu_i \in \Delta([0,1])$,

w/ mean $\mu_i = \mathbb{E}_{r \sim \nu_i}[r]$

$I^* = \arg\max_{i \in [K]} \mu_i$



Learning:

Q: How to learn efficient in Multi-armed Bandit setting:

1. Explore and Committee algorithm:

Learning:

Q: How to learn efficient in Multi-armed Bandit setting:

1. Explore and Committee algorithm:

1. For the first NK rounds, try each arm N times, compute its average mean $\hat{\mu}_i$

Learning:

Q: How to learn efficient in Multi-armed Bandit setting:

1. Explore and Committee algorithm:

1. For the first NK rounds, try each arm N times, compute its average mean $\hat{\mu}_i$

2. For all future T-KN rounds, play the best empirical arm $\hat{I} = \arg\max_{i \in [K]} \hat{\mu}_i$

$\text{Regret}_T = O\left(T^{2/3} K^{1/3}\right)$

Learning:

Q: How to learn efficient in Multi-armed Bandit setting:

2. The Upper Confidence Bound Algorithm

Learning:

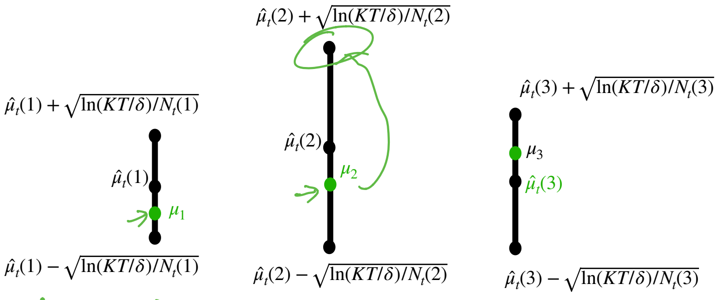
Q: How to learn efficient in Multi-armed Bandit setting:

2. The Upper Confidence Bound Algorithm

For $t = 0 \to T-1$:

$$I_t = \arg\max_{i \in [K]} \left(\hat{\mu}_t(i) + \sqrt{\frac{\ln(KT/\delta)}{N_t(i)}} \right)$$

(# Upper-conf-bound of arm i)



$\text{Regret}_T = O(\sqrt{KT})$

Learning:

Q: How to learn efficient in Contextual Bandit setting:

1. Explore and Commit (or ϵ-greedy)

Importance weighting + Reward-sensitive Classification

Learning:

Q: How to learn efficient in Contextual Bandit setting:

1. Explore and Commit (or ϵ-greedy)

Importance weighting + Reward-sensitive Classification

1. For the first N rounds, randomly try actions to construct a classification dataset:

$$\{x_i, \hat{\mathbf{r}}_i\}_{i=0}^{N-1}$$

t

$x_t \sim \mu$

IW $\rightarrow$

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \dots \\ r_t / p(a_t) \\ 0, \\ \dots \\ 0 \end{bmatrix}$$

$E[\hat{r}] = r$

Learning:

Q: How to learn efficient in <u>Contextual</u> Bandit setting:

1. Explore and Commit (or ϵ-greedy)

Importance weighting + Reward-sensitive Classification

1. For the first N rounds, randomly try actions to construct a classification dataset:

$$\{x_i, \hat{\mathbf{r}}_i\}_{i=0}^{N-1}$$

$$\hat{\mathbf{r}} := \begin{bmatrix} 0 \\ 0 \\ \dots \\ r_t / p(a_t) \\ 0, \\ \dots \\ 0 \end{bmatrix}$$

2. **Call RSC oracle:** $\hat{\pi} = \arg\max_{\pi \in \Pi} \sum_{i=0}^{N-1} \hat{\mathbf{r}}_i[\pi(x_i)]$

$O(T^{2/3} K^{1/3})$

What we covered this semester:

1. Basics of Markov Decision Process

2. Planning in MDP: VI and PI

3. Learning: Model-based RL, Policy Optimization, Bandit

4. Imitation

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Offline IL: only expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$ is available (no other interaction)

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Offline IL: only expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$ is available (no other interaction)

BC: a Reduction to Supervised Learning:

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Offline IL: only expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$ is available (no other interaction)

BC: a Reduction to Supervised Learning:

$$\hat{\pi} = \arg\min_{\pi \in \Pi} \sum_{i=1}^{M} \ell\left(\pi, s^\star, a^\star\right)$$

loss function for classification/Regression

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Offline IL: only expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$ is available (no other interaction)

BC: a Reduction to Supervised Learning:

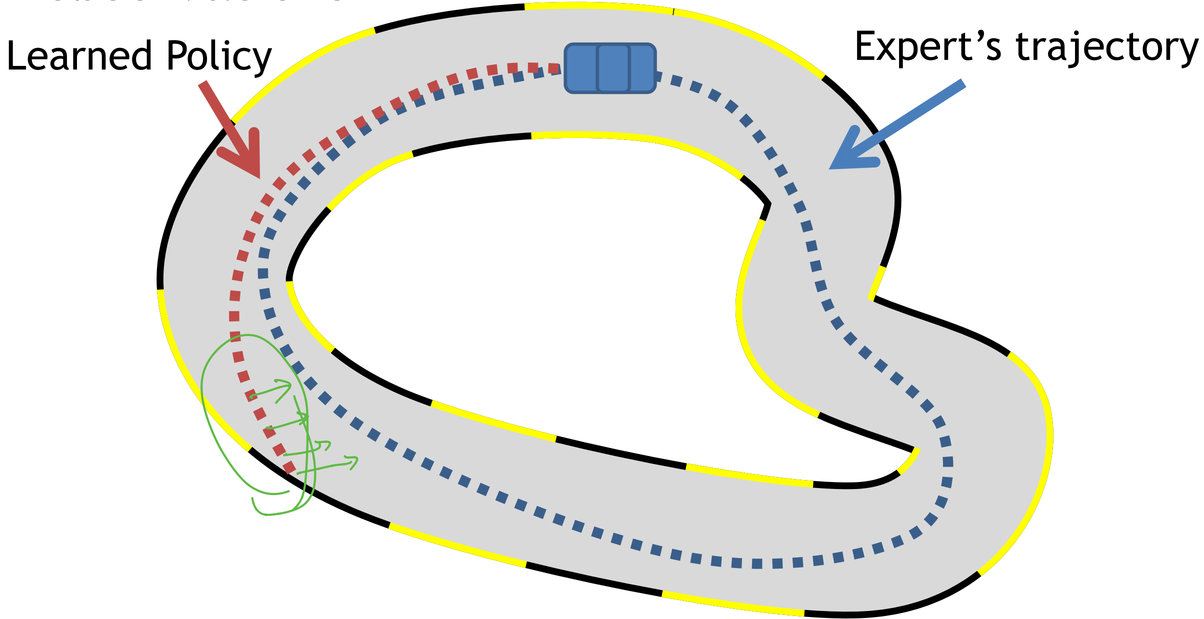
$$\hat{\pi} = \arg\min_{\pi \in \Pi} \sum_{i=1}^{M} \ell\left(\pi, s^\star, a^\star\right)$$

e.g., Negative log-likelihood (NLL): $\ell(\pi, s, a^\star) = -\ln \pi(a^\star \mid s^\star)$ (used in AlphaGo)

Distribution shift!

[Pomerleau89,Daume09]

- Predictions affect future inputs/ observations



Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Interactive IL: expert is available for query during the learning process

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Interactive IL: expert is available for query during the learning process

The DAgger Algorithm (Data Aggregation):

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Interactive IL: expert is available for query during the learning process

The DAgger Algorithm (Data Aggregation):

Expert actions

1. W/ π^t, generate dataset $\mathscr{D}^t = \{s_i, a_i^\star\}, s_i \sim d_\mu^{\pi^t}, a_i^\star = \pi^\star(s_i)$

from our policy π^t

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Interactive IL: expert is available for query during the learning process

The DAgger Algorithm (Data Aggregation):

1. W/ π^t, generate dataset $\mathscr{D}^t = \{s_i, a_i^\star\}, s_i \sim d_\mu^{\pi^t}, a_i^\star = \pi^\star(s_i)$
2. **Data aggregation**: $\mathscr{D} = \mathscr{D} + \mathscr{D}^t$

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Interactive IL: expert is available for query during the learning process

The DAgger Algorithm (Data Aggregation):

1. W/ π^t, generate dataset $\mathscr{D}^t = \{s_i, a_i^\star\}, s_i \sim d_\mu^{\pi^t}, a_i^\star = \pi^\star(s_i)$
2. **Data aggregation**: $\mathscr{D} = \mathscr{D} + \mathscr{D}^t$
3. **Update policy via Supervised-Learning**: $\pi^{t+1} = \text{SL}\left(\mathscr{D}\right)$

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

1. Interactive IL: expert is available for query during the learning process

The DAgger Algorithm (Data Aggregation):

1. W/ π^t, generate dataset $\mathscr{D}^t = \{s_i, a_i^\star\}, s_i \sim d_\mu^{\pi^t}, a_i^\star = \pi^\star(s_i)$
2. **Data aggregation**: $\mathscr{D} = \mathscr{D} + \mathscr{D}^t$
3. **Update policy via Supervised-Learning**: $\pi^{t+1} = \mathrm{SL}\left(\mathscr{D}\right)$

(Recap the connection to online learning and how it avoids distribution shift..)

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

3. Hybrid IL: expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$, and access to real transition $P(\,.\,|\,s, a)$

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

3. Hybrid IL: expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$, and access to real transition $P(\,.\,|s, a)$

Formulation of Maximum Entropy Inverse RL:

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

3. Hybrid IL: expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$, and access to real transition $P(\,.\,|s,a)$

Formulation of Maximum Entropy Inverse RL:

$$\arg\min_\pi \mathbb{E}_{s,a\sim d_\mu^\pi} \ln \pi(a\,|\,s) \iff \max_\pi \mathbb{E}_{s\sim d^\pi} \text{entropy}\left(\pi(\cdot|s)\right)$$

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

3. Hybrid IL: expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$, and access to real transition $P(\,.\,|s,a)$

Formulation of Maximum Entropy Inverse RL:

$$\arg\min_\pi \mathbb{E}_{s,a\sim d_\mu^\pi} \ln \pi(a\,|\,s)$$

$$s\,.\,t, \mathbb{E}_{s,a\sim d_\mu^\pi}\phi(s,a) = \mathbb{E}_{s,a\sim d_\mu^{\pi^\star}}\phi(s,a)$$

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

3. Hybrid IL: expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$, and access to real transition $P(\,.\,|s,a)$

Formulation of Maximum Entropy Inverse RL:

$$\arg\min_\pi \mathbb{E}_{s,a\sim d_\mu^\pi} \ln \pi(a\,|\,s)$$

$$s.t, \mathbb{E}_{s,a\sim d_\mu^\pi}\phi(s,a) = \mathbb{E}_{s,a\sim d_\mu^{\pi^\star}}\phi(s,a)$$

Assume the ground truth reward $r(s,a) = (\theta^\star)^\top \phi(s,a)$

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

3. Hybrid IL: expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$, and access to real transition $P(\,.\,|s,a)$

Formulation of Maximum Entropy Inverse RL:

Lagrangian multipliers

$$\max_{w \in \mathbb{R}^d} \min_{\pi} \underbrace{\mathbb{E}_{s,a\sim d_\mu^\pi} \ln \pi(a|s) + w^\top \left(\mathbb{E}_{s,a\sim d_\mu^\pi} \phi(s,a) - \mathbb{E}_{s,a\sim d_\mu^{\pi^\star}} \phi(s,a) \right)}_{:=\ell(\pi,w)}$$

Imitation:

Q: what we do when r is not available but we have an expert $\pi^e(\approx \pi^\star)$?

3. Hybrid IL: expert data $\{s_i^\star, a_i^\star\}_{i=1}^N$, and access to real transition $P(\,.\,|s,a)$

Incremental update

Formulation of Maximum Entropy Inverse RL:

Best Response

$$\max_{w \in \mathbb{R}^d} \min_{\pi} \underbrace{\mathbb{E}_{s,a\sim d_\mu^\pi} \ln \pi(a|s) + w^\top \left(\mathbb{E}_{s,a\sim d_\mu^\pi} \phi(s,a) - \mathbb{E}_{s,a\sim d_\mu^{\pi^\star}} \phi(s,a) \right)}_{:=\ell(\pi,w)}$$

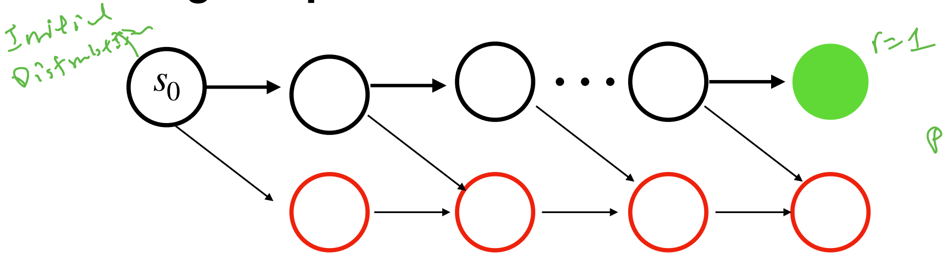
Iterate: $w_{t+1} = w_t + \eta \nabla_w \ell(w_t, \pi_t)$, $\pi^{t+1} = \arg\min_{\pi} \ell(w_{t+1}, \pi)$ via Soft-VI

Gradient Ascent on W

Best Response on π

What we did not cover:

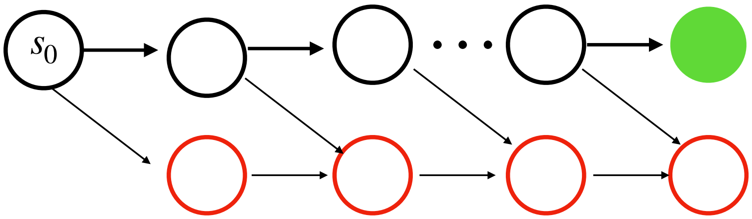
1. How to do strategic exploration in RL? Can we do it in poly time?



poly(|S|, |A|, H)

What we did not cover:

1. How to do strategic exploration in RL? Can we do it in poly time?

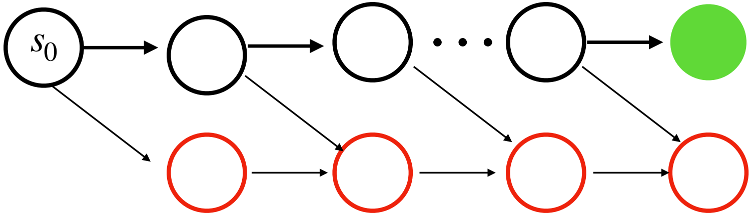


2. When does Policy Gradient guarantee Global optimality?

Though the RL objective function is non-convex wrt policy, under some cases, PG provably converges to global optimal policies!

What we did not cover:

1. How to do strategic exploration in RL? Can we do it in poly time?



2. When does Policy Gradient guarantee Global optimality?

Though the RL objective function is non-convex wrt policy, under some cases, PG provably converges to global optimal policies!

3. Deep Reinforcement Learning

Most of the time, it is Deep nets (e.g., policies) + RL