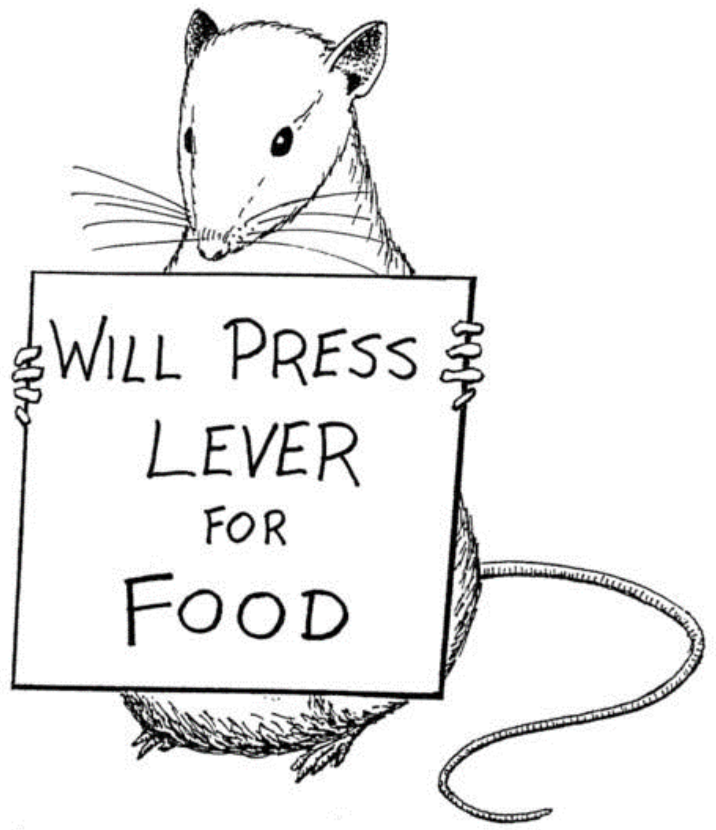
UCI University of California, Irvine

CS 277: Control and Reinforcement Learning

Winter 2022

Lecture 13: Inverse RL

Roy Fox
Department of Computer Science
Bren School of Information and Computer Sciences
University of California, Irvine



Logistics

assignments

- Assignment 3 is due today
- Assignment 4 to be published soon

quizzes

- Quiz 5 is due Friday

Today's lecture

Belief-state MDPs

RNNs

IRL, Feature Matching

MaxEnt IRL

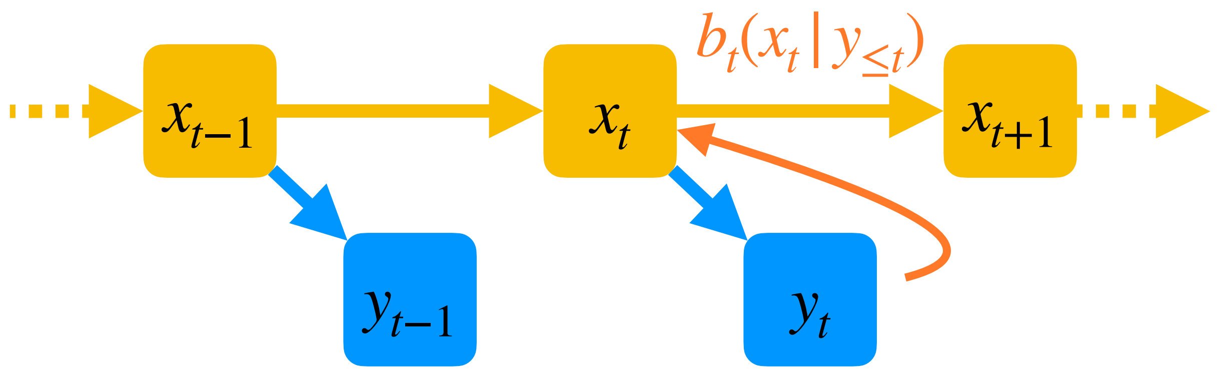
Tiger domain

- 2 states: which door leads to a tiger (-100 reward) and which to $$$ (+10)
- You can stop and listen: $p(o_t = s_t | s_t) = 0.8$

$p(s_0 = s_{\text{left}}) = 0.5$	$\mathbb{E}[r(s_0, a_{\text{left}})] = -45$	$o_1 = o_{\text{right}}$
$p(s_1 = s_{\text{left}}) = 0.2$	$\mathbb{E}[r(s_1, a_{\text{left}})] = -12$	$o_2 = o_{\text{left}}$
$p(s_2 = s_{\text{left}}) = 0.5$	$\mathbb{E}[r(s_2, a_{\text{left}})] = -45$	$o_3 = o_{\text{right}}$
$p(s_3 = s_{\text{left}}) = 0.2$	$\mathbb{E}[r(s_3, a_{\text{left}})] = -12$	$o_4 = o_{\text{right}}$
$p(s_4 = s_{\text{left}}) = \frac{0.04}{0.04 + 0.64} \approx 0.06$	$\mathbb{E}[r(s_4, a_{\text{left}})] = -3.5$	$o_5 = o_{\text{right}}$
$p(s_5 = s_{\text{left}}) \approx 0.015$	$\mathbb{E}[r(s_4, a_{\text{left}})] = -8.3$	

Belief

- Belief = distribution over the state $b(s)$
  - If the agent reaches belief b after history h, that does not imply $s \sim b$
- Bayesian belief $b_h(s) = p(s \mid h)$: a sufficient statistic of h for s
  - For a Bayesian belief: $s \sim b_h$ after history h
- In the linear–Gaussian case: the Kalman filter



  - Bayesian belief is Gaussian $p(x_t \mid h_t = y_{\le t}) = \mathcal{N}(x_t; \hat{x}_t, \Sigma_t)$
  - Covariance can be precomputed $\mathbb{V}(x_t \mid h_t) = \Sigma_t$ (independent of h_t)
  - Mean can be updated linearly: $\hat{x}'_t = A\hat{x}_{t-1} + Bu_{t-1} \qquad e_t = y_t - C\hat{x}'_t \qquad \hat{x}_t = \hat{x}'_t + K_t e_t$

Computing the Bayesian belief

- Predict s_{t+1} from $h_t = (o_0, a_0, o_1, a_1, \ldots, o_t)$ and a_t:

$$b'_t(s_{t+1} | h_t, a_t) = \sum_{s_t} p(s_t | h_t) p(s_{t+1} | s_t, a_t) = \sum_{s_t} b_t(s_t) p(s_{t+1} | s_t, a_t)$$

total probability over s_t

previous belief b_t

dynamics needs to be known

- Update belief of s_t after seeing $h_t = (h_{t-1}, a_{t-1}, o_t)$:

previous prediction

observation model

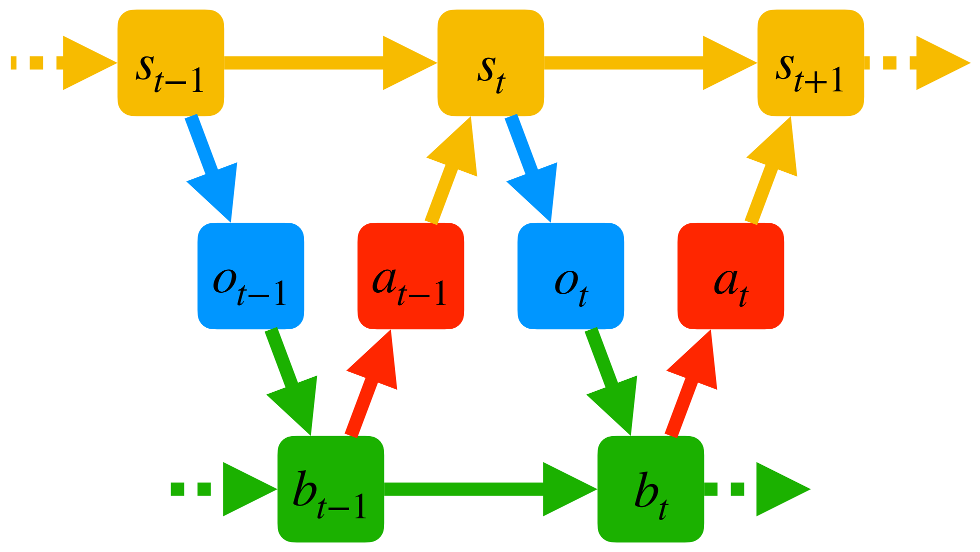
$$b_t(s_t | h_t) = \frac{p(s_t | h_{t-1}, a_{t-1}) p(o_t | s_t)}{p(o_t | h_{t-1}, a_{t-1})} = \frac{b'_{t-1}(s_t) p(o_t | s_t)}{\sum_{\bar{s}_t} b'_{t-1}(\bar{s}_t) p(o_t | \bar{s}_t)}$$

Bayes' rule on o_t

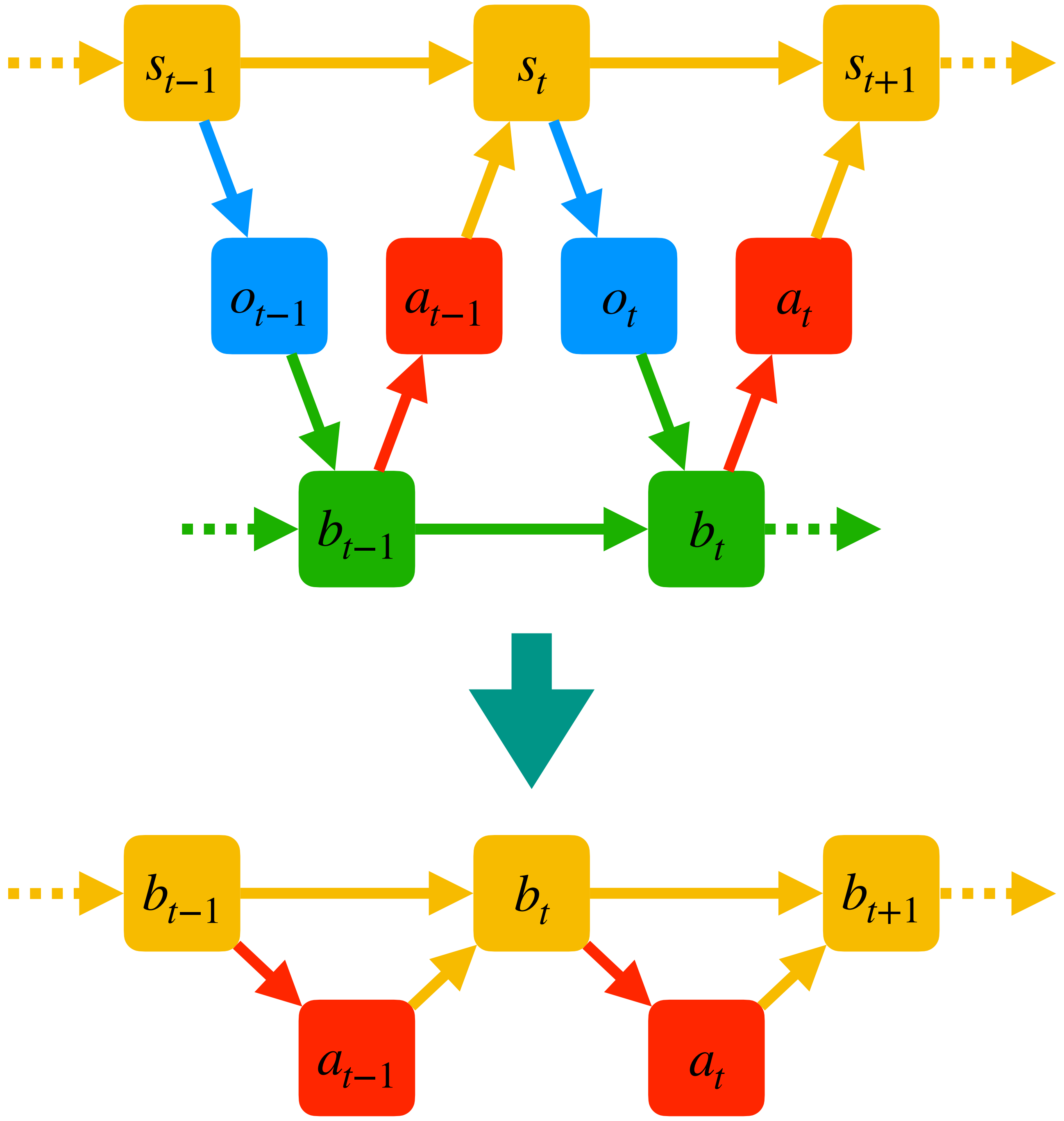
$o_t - s_t - (h_{t-1}, a_{t-1})$

normalizer

- A deterministic, model-based update:
  - $b_{t-1}(s_{t-1})$ → use a_{t-1} to predict $b'_{t-1}(s_t)$ → use o_t to update $b_t(s_t)$



Belief-state MDP



- In the linear–quadratic–Gaussian case: certainty equivalence
  - Plan using $\hat{x}_t$ as if it was x_t
- More generally (though vastly less useful): belief-state MDP
  - States: $\Delta(\mathscr{S})$ Actions: $\mathscr{A}$ Rewards: $r(b_t, a_t) = \sum_{s_t} b_t(s_t) r(s_t, a_t)$
- Transitions: each possible observation o_{t+1} contributes its probability

$$p(o_{t+1} | b_t, a_t) = \sum_{s_t, s_{t+1}} b_t(s_t) p(s_{t+1} | s_t, a_t) p(o_{t+1} | s_{t+1})$$

to the total probability that the belief that follows (b_t, a_t, o_{t+1}) is the Bayesian belief

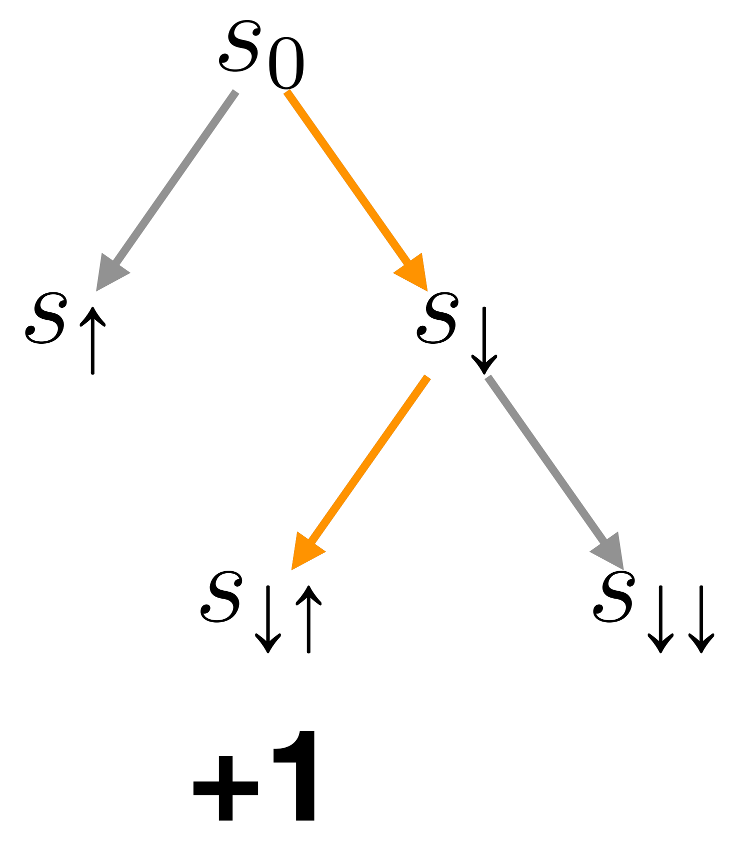
$$b_{t+1}(s_{t+1}) = p(s_{t+1} | b_t, a_t, o_{t+1}) = \frac{\sum_{s_t} b_t(s_t) p(s_{t+1} | s_t, a_t) p(o_{t+1} | s_{t+1})}{p(o_{t+1} | b_t, a_t)}$$

Learning to use memory is hard

- Belief space $b(s_t)$ is continuous and high-dimensional (dimension $|\mathcal{S}|$)
  - Curse of dimensionality
  - Beliefs are naturally multi-modal — how do we even represent them?
- The number of reachable beliefs may grow exponentially in t (one per h_t)
  - Curse of history
- Belief-value function can be very complex, hard to approximate
- There may not be optimal stationary deterministic policy ⇒ instability

Stationary deterministic policy counterexample

- Assume no observability
- Stationary deterministic policies gets no reward
- Non-stationary policy: ↓, ↑; expected return: +1
  - But non-stationary = observability of a clock t
- Stationary stochastic policy: ↓ / ↑ with equal prob.; expected return: +0.25



- Open problem: Bellman optimality is inherently stationary and deterministic

no dependence on t

maximum achieved for some action

$$V(s) = \max_a r(s,a) + \gamma \mathbb{E}_{(s'|s,a)\sim p}[V(s')]$$

Today's lecture

Belief-state MDPs

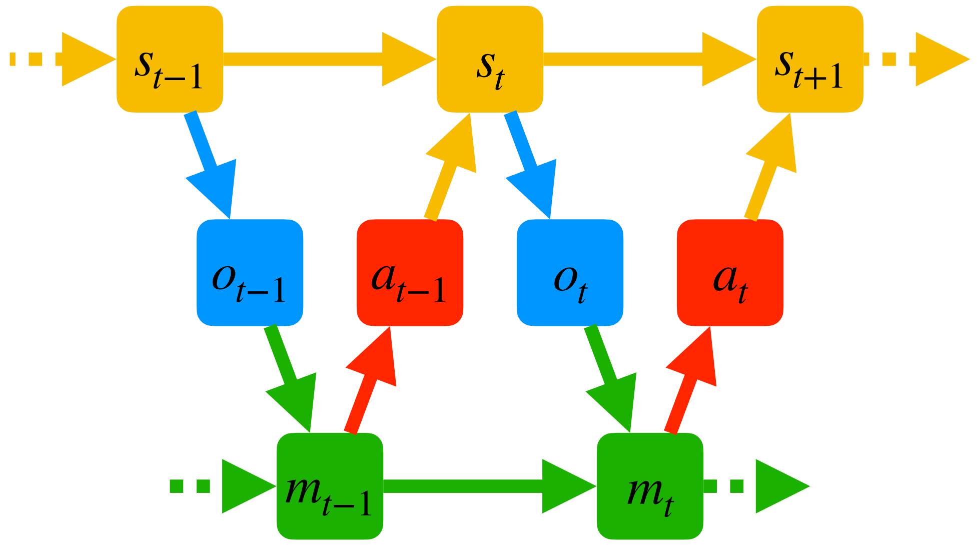
RNNs

IRL, Feature Matching

MaxEnt IRL

Filtering with function approximation

- Instead of Bayesian belief: memory update $m_t = f_\theta(m_{t-1}, o_t)$ (a_{t-1} optional)
  - Action policy: $\pi_\theta(a_t | m_t)$
  - Sequential structure = Recurrent Neural Network (RNN)



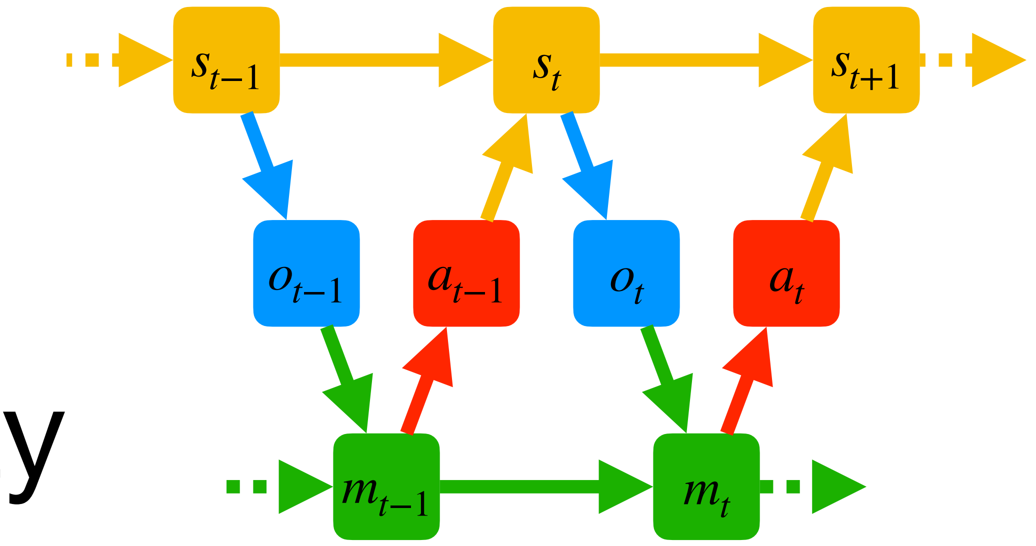
- Training: back-propagate gradients through the whole sequence
  - Back-propagation through time (BPTT)
- Unfortunately, gradients tend to vanish → 0 / explode → ∞
  - Long term coordination of memory updates + actions is challenging
  - RNN can't use information not remembered, but backup no gradient unless used

RNNs in on-policy methods

- Training RNNs with on-policy methods is straightforward (and backward)
  - Roll out policy: parameters of a_t distribution are determined by $\pi_\theta(m_t)$ with

$$m_t = f_\theta(\cdots f_\theta(f_\theta(o_0), o_1), \cdots o_t)$$

  - Compute $\nabla_\theta \log \pi_\theta(a_t | m_t)$ with BPTT all the way to initial observation o_0
- Problems: computation graph > RAM; vanishing / exploding grads
  - Solutions: stop gradients every k steps; use attention
- Problem: cannot learn longer memory — but that's hard anyway



RNNs in off-policy methods

- Problem: RNN states in replay buffer disagree with current RNN params

- Solution 1: use n-step rollouts to reduce mismatch effect

$$Q_\theta(o_t, m_t, a_t) \to r_t + \gamma r_{t+1} + \cdots + \gamma^{n-1} r_{t+n-1} + \gamma^n \max_{a'} Q_\theta(o_{t+n}, m_{t+n}, a')$$

- Solution 2: “burn in” m_t from even earlier stored steps
  - Same target, but m_t is initialized from $(o_{t-k}, \ldots, o_{t-1})$

- In practice: RNNs rarely used
  - Stacking k frames every step $(o_{t-k+1}, \ldots, o_t)$ may help with short-term memory

Deep RL as partial observability

- Memory-based policies fail us in Deep RL, where we need them most:
  - Deep RL is inherently partially observable
- Consider what deeper layers get as input:
  - High-level / action-relevant state features are not Markov!
- Memory management is a huge open problem in Deep RL
  - Actually, in other areas of ML too: NLP, time-series analysis, video processing, ...

Recap and further considerations

- Let policies depend on observable history through memory
- Memory update: Bayesian, approximate, or learned
  - Learning to update memory is one of the biggest open problems in all of ML
- Let policy be stochastic
  - Should memory be stochastic? interesting research question...
- Let policies be non-stationary if possible, otherwise learning may be unstable
  - Time-dependent policies for finite-horizon tasks
  - Periodic policies for periodic tasks

Today's lecture

Belief-state MDPs

RNNs

IRL, Feature Matching

MaxEnt IRL

Learning rewards from demonstrations

- RL: rewards → policy; IL: demonstrations → policy
- Inverse Reinforcement Learning (IRL): demonstrations → reward function
  - Better understand agents (humans, animals, users, markets)
    - Preference elicitation, teleology (the "what for" of actions), theory of mind, language
  - First step toward Apprenticeship Learning: demos → rewards → policy
    - Infer the teacher's goals and learn to achieve them; overcome suboptimal demos
    - Partly model-based (learn r but not p); may be easier to learn, generalize, transfer
    - Teacher and learner can have different action spaces (e.g., human → robot)

Inverse Reinforcement Learning (IRL)

- Given a dataset of demonstration trajectories $\mathscr{D} = \{\xi_i\}$

  $r(s)$ **expressive enough**

- Find teacher's reward function $r : \mathcal{S} \times \cancel{\mathcal{A}} \to \mathbb{R}$
  - Principle: demonstrated actions should achieve high expected return
- IRL is ill-defined
  - How low is the reward for states and actions not in $\mathscr{D}$?
  - How is the reward distributed along the trajectory?
    - Sparse rewards = identify "subgoal" states; dense = score each step, as hard as IL
  - Demonstrator can be fallible = take suboptimal actions; how much?

Feature matching

- Assume linear reward $r_\theta(s) = \theta^\intercal f_s$ in given state features $f_s \in \mathbb{R}^d$
  - Value = $J_\theta^\pi = \sum_t \gamma^t \mathbb{E}_{s_t \sim p_\pi}[\theta^\intercal f_{s_t}] = \mathbb{E}_{s \sim p_\pi}[\theta^\intercal f_s]$, with $p_\pi(s) \propto \sum_t \gamma^t p_\pi(s_t)$

$t \sim \mathrm{Geom}(1-\gamma)$

missing const: $(1-\gamma)$

- Teacher optimality: expert value $J_\theta^{\pi^*}$ higher than any other policy's value J_θ^π
  - Find θ that maximizes the gap $J_\theta^{\pi^*} - J_\theta^\pi$; but for which π?
  - Apprenticeship Learning: find π that maximizes J_θ^π ; but for which θ?
- Solve: $\max_\theta \min_\pi \{J_\theta^{\pi^*} - J_\theta^\pi\} = \max_\theta \min_\pi \{\mathbb{E}_{s \sim p^*}[\theta^\intercal f_s] - \mathbb{E}_{s \sim p_\pi}[\theta^\intercal f_s]\}$
  - Approximate $s \sim p^*$ with $s \sim \mathcal{D}$

Feature matching

- Solving $\max_{\theta} \min_{\pi} \{ \mathbb{E}_{s\sim p^*}[\theta^\intercal f_s] - \mathbb{E}_{s\sim p_\pi}[\theta^\intercal f_s] \}$

```
Algorithm Feature Matching
  Initialize policy set $\Pi = \{\pi_0\}$
  repeat
    Solve Quadratic Program: $\max_{\eta, \|\theta\|_2 \le 1} \eta$
      s.t. $\mathbb{E}_{s\sim\mathcal{D}}[\theta^\intercal f_s] \ge \mathbb{E}_{s\sim p_\pi}[\theta^\intercal f_s] + \eta \quad \forall \pi \in \Pi$
    $\pi \leftarrow$ optimal policy for $r_\theta(s) = \theta^\intercal f_s$
    Add $\pi$ to $\Pi$
```

θ must be bounded, or solution at ∞

- On convergence: π optimal for θ (no gap), can't find θ with gap
  - $\Rightarrow \mathbb{E}_{s\sim\mathscr{D}}[\theta^\intercal f_s] \approx \mathbb{E}_{s\sim p_\pi}[\theta^\intercal f_s]$ for all $\theta \Rightarrow \mathbb{E}_{s\sim\mathscr{D}}[f_s] \approx \mathbb{E}_{s\sim p_\pi}[f_s]$

feature matching

Today's lecture

Belief-state MDPs

RNNs

IRL, Feature Matching

MaxEnt IRL

Modeling bounded teachers

- An expert teacher maximizes the value $J_\theta^{\pi^*} = \sum_t \gamma^t \mathbb{E}_{s_t \sim p^*}[\theta^\intercal f_{s_t}] = \mathbb{E}_{\xi \sim p^*}[\theta^\intercal f_\xi]$
  - With trajectory-summed features $f_\xi = \sum_t \gamma^t f_{s_t}$
- Assume teacher has unintentional / uninformed prior policy π_0
  - Bounded rationality: cost to intentionally diverge $\mathbb{D}[\pi^* \| \pi_0]$ (with π_0 uniform: $\mathbb{H}[\pi^*]$)
  - Total cost: $\sum_t \mathbb{E}_{(s_t,a_t) \sim p^*} \left[\log \frac{\pi^*(a_t \mid s_t)}{\pi_0(a_t \mid s_t)} \right] = \mathbb{E}_{\xi \sim p^*} \left[\log \frac{p^*(\xi)}{p_0(\xi)} \right] = \mathbb{D}[p^*(\xi) \| p_0(\xi)]$
- Bounded optimality: $\max_{\pi^*} \mathbb{E}_{\xi \sim p^*}[\theta^\intercal f_\xi] - \tau \mathbb{D}[p^* \| p_0]$

Bounded optimality: naïve solution

- Bounded optimality: $\max_{\cancel{\pi^*}\; p^*} \mathbb{E}_{\xi \sim p^*}[\theta^\intercal f_\xi] - \mathbb{D}[p^* \| p_0]$
  - Naïve solution: allow any distribution p^* over trajectories
  - No need to be consistent with dynamics $p(s' \,|\, s, a) \Rightarrow p^*$ may be unachievable
- Add the constraint $\sum_\xi p^*(\xi) = 1$ with Lagrange multiplier λ
- Differentiate by $p^*(\xi)$ and $= 0$ to optimize

$$\theta^\intercal f_\xi - \log p^*(\xi) + \log p_0(\xi) - 1 + \lambda = 0 \Longrightarrow p^*(\xi) = \frac{p_0(\xi)\exp(\theta^\intercal f_\xi)}{\sum_{\bar{\xi}} p_0(\bar{\xi})\exp(\theta^\intercal f_{\bar{\xi}})}$$

IRL with bounded teacher

- Assume that demonstrations are distributed $p_\theta(\xi) = \frac{1}{Z_\theta} p_0(\xi) \exp(\theta^\intercal f_\xi)$
  - With partition function $Z_\theta = \mathbb{E}_{\bar{\xi} \sim p_0}[\exp(\theta^\intercal f_{\bar{\xi}})]$
- Find θ that minimizes NLL of demonstrations

$$\nabla_\theta \log p_\theta(\xi) = \nabla_\theta(\theta^\intercal f_\xi - \log Z_\theta) = f_\xi - \frac{1}{Z_\theta} \nabla_\theta Z_\theta$$

$$= f_\xi - \frac{1}{Z_\theta} \mathbb{E}_{\bar{\xi} \sim p_0}[\exp(\theta^\intercal f_{\bar{\xi}}) f_{\bar{\xi}}] = f_\xi - \mathbb{E}_{\bar{\xi} \sim p_\theta}[f_{\bar{\xi}}]$$

  - To compute gradient, we need p_θ, but how to compute Z_θ?

Computing Z_θ: backward recursion

- Partition function: $Z_\theta = \mathbb{E}_{\xi \sim p_0}[\exp(\theta^\intercal f_\xi)]$

- Compute Z_θ recursively backward: like a value function, but + becomes ·

$$\mathrm{Z}_\theta(s_t, a_t) = \mathbb{E}_{p_0}[\exp(\theta^\intercal f_{\xi \geq t}) \,|\, s_t, a_t] = \exp(\theta^\intercal f_{s_t}) \mathbb{E}_{(s_{t+1}|s_t,a_t)\sim p}[\mathrm{Z}_\theta(s_{t+1})]$$

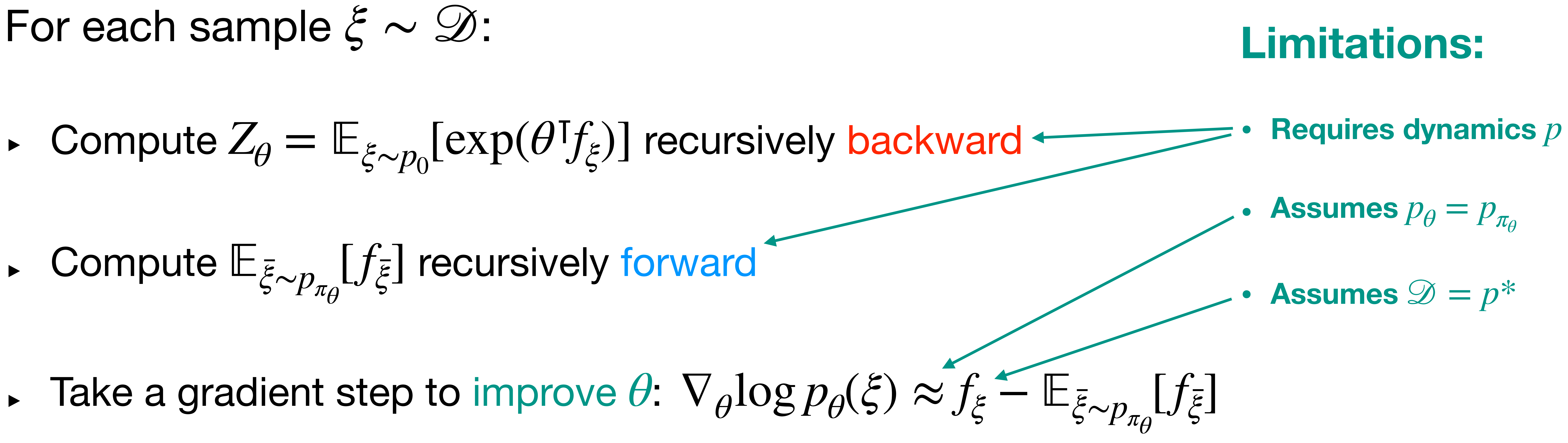
$$\mathrm{Z}_\theta(s_t) = \mathbb{E}_{p_0}[\exp(\theta^\intercal f_{\xi \geq t}) \,|\, s_t] = \mathbb{E}_{(a_t|s_t)\sim \pi_0}[\mathrm{Z}_\theta(s_t, a_t)]$$

- How to get a policy from Z_θ?
  - Marginalize: $\pi_\theta(a_t \,|\, s_t) = \dfrac{p_\theta(\xi \,|\, s_t, a_t)}{p_\theta(\xi \,|\, s_t)} = \pi_0(a_t \,|\, s_t) \dfrac{\mathrm{Z}_\theta(s_t, a_t)}{\mathrm{Z}_\theta(s_t)}$

consistent π may not even exist

  - This π_θ is not globally consistent $p_\theta(\xi) \neq p_{\pi_\theta}(\xi)$, $p_\theta(\xi)$ ignores the dynamics

MaxEnt IRL



- For each sample $\xi \sim \mathscr{D}$:
    - Compute $Z_\theta = \mathbb{E}_{\xi \sim p_0}[\exp(\theta^\intercal f_\xi)]$ recursively backward
    - Compute $\mathbb{E}_{\bar{\xi} \sim p_{\pi_\theta}}[f_{\bar{\xi}}]$ recursively forward
    - Take a gradient step to improve θ: $\nabla_\theta \log p_\theta(\xi) \approx f_\xi - \mathbb{E}_{\bar{\xi} \sim p_{\pi_\theta}}[f_{\bar{\xi}}]$
- At the optimum: feature matching $\mathbb{E}_{\xi \sim \mathscr{D}}[f_\xi] = \mathbb{E}_{\xi \sim p_{\pi_\theta}}[f_\xi]$
    - MaxEnt IRL approximates $\max_\theta \mathbb{H}[\pi_\theta] \quad$ s.t. $\mathbb{E}_{\xi \sim \mathscr{D}}[f_\xi] = \mathbb{E}_{\xi \sim p_{\pi_\theta}}[f_\xi]$

Limitations:

- **Requires dynamics p**
- **Assumes $p_\theta = p_{\pi_\theta}$**
- **Assumes $\mathscr{D} = p^*$**