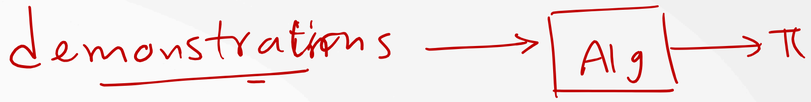


Inverse Reinforcement Learning

Deep Reinforcement Learning

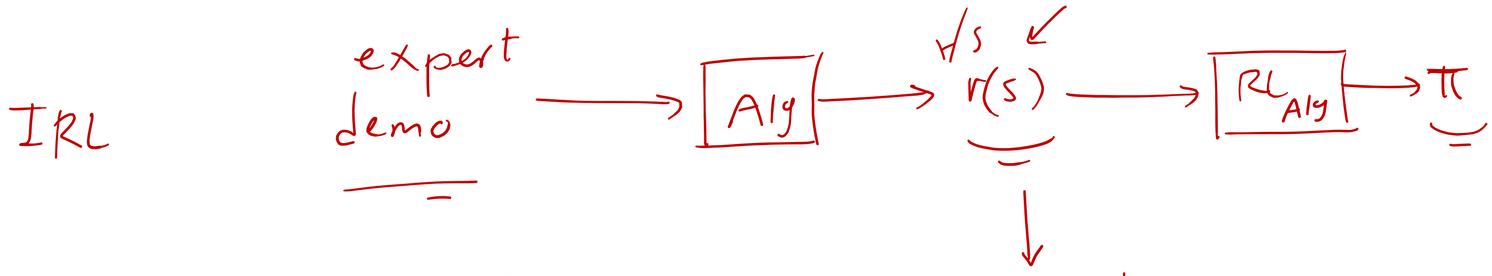
Mohammad Hossein Rohban, Ph.D.

Courtesy: Some slides are adopted from CS 285 Berkeley, and CS 234 Stanford, and Pieter Abbeel's compact series on RL.

RIML Lab, CE Department, Sharif University of Technology

Spring 2026

Lecture Outline

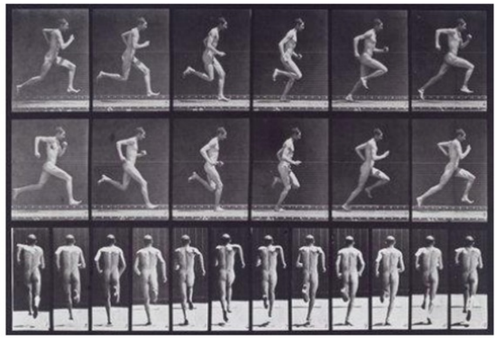


- So far: manually design reward function to define a task
- What if we want to learn the reward function from observing an expert, and then use reinforcement learning?
- Apply approximate optimality model from last time, but now learn the reward!

Goals:

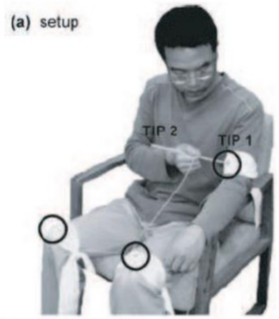
- ☑ Understand the inverse reinforcement learning problem definition
- ☑ Understand how probabilistic models of behavior can be used to derive inverse reinforcement learning algorithms
- ☑ Understand a few practical inverse reinforcement learning algorithms we can use

Modeling Human Behavior with Optimal Control



Muybridge (c. 1870)

Mombaur et al. '09



Li & Todorov '06

Ziebart '08

$$\mathbf{a}_1, \dots, \mathbf{a}_T = \arg\max_{\mathbf{a}_1,\dots,\mathbf{a}_T} \sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t)$$

$$\mathbf{s}_{t+1} = f(\mathbf{s}_t, \mathbf{a}_t)$$

optimize this to explain the data

$$\pi = \arg\max_{\pi} E_{\mathbf{s}_{t+1}\sim p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t),\mathbf{a}_t\sim\pi(\mathbf{a}_t|\mathbf{s}_t)}[r(\mathbf{s}_t, \mathbf{a}_t)]$$

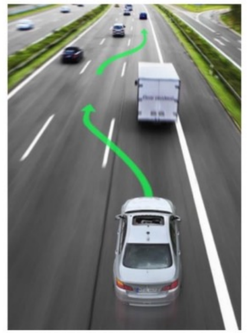
$$\mathbf{a}_t \sim \pi(\mathbf{a}_t|\mathbf{s}_t)$$

Imitation learning vs RL perspective

The imitation learning perspective

- Standard imitation learning:
  - copy the actions performed by the expert
  - no reasoning about outcomes of actions
- Human imitation learning:
  - copy the intent of the expert
  - might take very different actions!

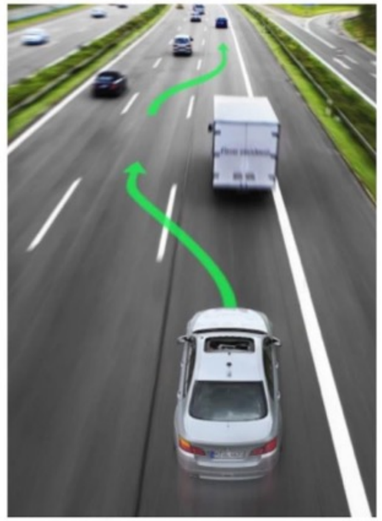
The reinforcement learning perspective



what is the reward?

Inverse Reinforcement Learning

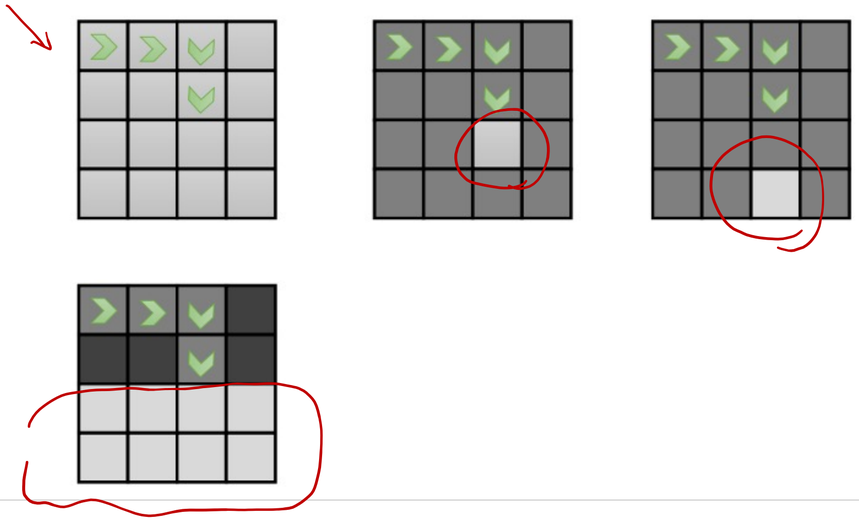
Infer **reward functions** from **demonstrations**



$r(\mathbf{s}, \mathbf{a})$

by itself, this is an **underspecified** problem

many reward functions can explain the **same** behavior



Inverse Reinforcement Learning Formulation

"forward" reinforcement learning

inverse reinforcement learning

Inverse Reinforcement Learning Formulation

"forward" reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$

inverse reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$

Inverse Reinforcement Learning Formulation

"forward" reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. reward function $r(\mathbf{s}, \mathbf{a})$

inverse reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$

Inverse Reinforcement Learning Formulation

"forward" reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. reward function $r(\mathbf{s}, \mathbf{a})$

inverse reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$

Inverse Reinforcement Learning Formulation

"forward" reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. reward function $r(\mathbf{s}, \mathbf{a})$
4. learn $\pi^\star(\mathbf{a}|\mathbf{s})$

inverse reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$

Inverse Reinforcement Learning Formulation

"forward" reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. reward function $r(\mathbf{s}, \mathbf{a})$
4. learn $\pi^\star(\mathbf{a}|\mathbf{s})$

inverse reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$
4. learn $r_\psi(\mathbf{s}, \mathbf{a})$ (ψ is reward parameters)

Inverse Reinforcement Learning Formulation

"forward" reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. reward function $r(\mathbf{s}, \mathbf{a})$
4. learn $\pi^\star(\mathbf{a}|\mathbf{s})$

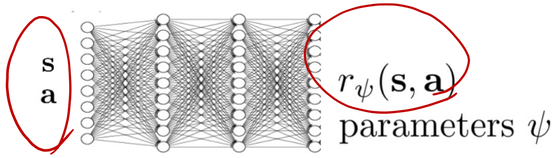
inverse reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$
4. learn $r_\psi(\mathbf{s}, \mathbf{a})$ (ψ is reward parameters)

linear reward function:

$$r_\psi(\mathbf{s}, \mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s}, \mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s}, \mathbf{a})$$

neural net reward function:



Inverse Reinforcement Learning Formulation

"forward" reinforcement learning

1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. reward function $r(\mathbf{s}, \mathbf{a})$
4. learn $\pi^\star(\mathbf{a}|\mathbf{s})$

inverse reinforcement learning

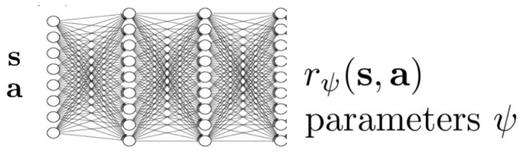
1. given:
2. states $\mathbf{s} \in \mathcal{S}$, actions $\mathbf{a} \in \mathcal{A}$ (sometimes)
   transitions $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$
3. samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$
4. learn $r_\psi(\mathbf{s}, \mathbf{a})$ (ψ is reward parameters)

...and then use it to learn $\pi^\star(\mathbf{a}|\mathbf{s})$

linear reward function:

$$r_\psi(\mathbf{s}, \mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s}, \mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s}, \mathbf{a})$$

neural net reward function:



Feature Matching Inverse RL

- linear reward function:

$$r_\psi(\mathbf{s}, \mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s}, \mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s}, \mathbf{a})$$

Feature Matching Inverse RL

$\forall i \quad \mathbb{E}_{s \sim \pi_{expert}}\{f_i(s,a)\} = \mathbb{E}_{\pi_{r\psi}}\{f_i(s,a)\}$

$\hookrightarrow$ RL on r_ψ

- linear reward function:

$$r_\psi(\mathbf{s},\mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s},\mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s},\mathbf{a})$$

$\dim(\psi) \gg \dim\{f_1 \cdots f_k\}$

- if features $\mathbf{f}$ are important, what if we match their expectations?

Feature Matching Inverse RL

- linear reward function:

$$r_\psi(\mathbf{s}, \mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s}, \mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s}, \mathbf{a})$$

- if features $\mathbf{f}$ are important, what if we match their expectations?
- let π^{r_ψ} be the optimal policy for r_ψ

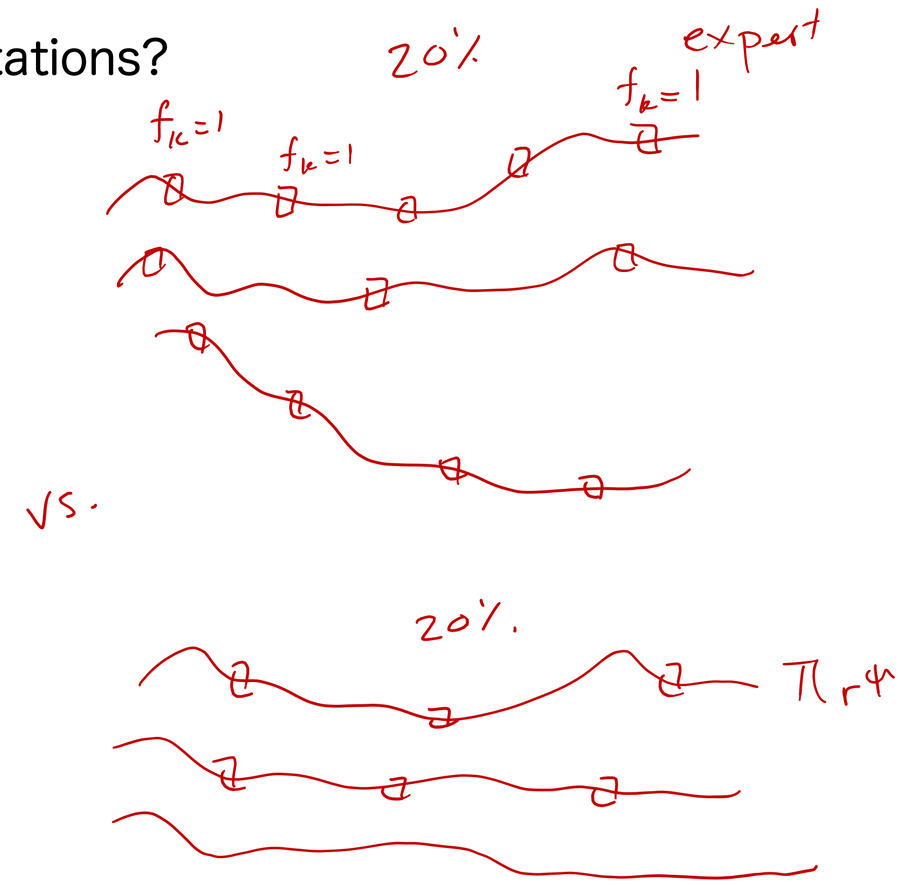
Feature Matching Inverse RL

- linear reward function:

$$r_\psi(\mathbf{s},\mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s},\mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s},\mathbf{a})$$

- if features $\mathbf{f}$ are important, what if we match their expectations?
- let π^{r_ψ} be the optimal policy for r_ψ
- pick ψ such that:

$$E_{\pi^{r_\psi}}[\mathbf{f}(\mathbf{s},\mathbf{a})] = E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})]$$



Feature Matching Inverse RL

- linear reward function:

$$r_\psi(\mathbf{s},\mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s},\mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s},\mathbf{a})$$

- if features $\mathbf{f}$ are important, what if we match their expectations?
- let π^{r_ψ} be the optimal policy for r_ψ
- pick ψ such that:

$$E_{\pi^{r_\psi}}[\mathbf{f}(\mathbf{s},\mathbf{a})] = E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})]$$

state-action marginal under π^{r_ψ}

unknown optimal policy. approximate using expert samples

Feature Matching Inverse RL

- linear reward function:

$$r_\psi(\mathbf{s},\mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s},\mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s},\mathbf{a})$$

- if features $\mathbf{f}$ are important, what if we match their expectations?
- let π^{r_ψ} be the optimal policy for r_ψ
- pick ψ such that:

$$E_{\pi^{r_\psi}}[\mathbf{f}(\mathbf{s},\mathbf{a})] = E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})]$$

still ambiguous!

state-action marginal under π^{r_ψ}

unknown optimal policy. approximate using expert samples

Feature Matching Inverse RL

- linear reward function:

$$r_\psi(\mathbf{s},\mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s},\mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s},\mathbf{a})$$

- if features $\mathbf{f}$ are important, what if we match their expectations?
- let π^{r_ψ} be the optimal policy for r_ψ
- pick ψ such that:

$$E_{\pi^{r_\psi}}[\mathbf{f}(\mathbf{s},\mathbf{a})] = E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})]$$

still ambiguous!

state-action marginal under π^{r_ψ}

unknown optimal policy. approximate using expert samples

- maximum margin principle:

$$\max_{\psi,m} m \quad \textit{such that} \quad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi\in\Pi} \psi^T E_\pi[\mathbf{f}(\mathbf{s},\mathbf{a})] + m$$

Avg. reward on expert demos

Avg. reward on other (best) policy

Feature Matching Inverse RL

- linear reward function:

$$r_\psi(\mathbf{s},\mathbf{a}) = \sum_i \psi_i f_i(\mathbf{s},\mathbf{a}) = \psi^T \mathbf{f}(\mathbf{s},\mathbf{a})$$

- if features $\mathbf{f}$ are important, what if we match their expectations?
- let π^{r_ψ} be the optimal policy for r_ψ
- pick ψ such that:

$$E_{\pi^{r_\psi}}[\mathbf{f}(\mathbf{s},\mathbf{a})] = E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})]$$

still ambiguous!

state-action marginal under π^{r_ψ}

unknown optimal policy. approximate using expert samples

- maximum margin principle:

Linear prog.

$$\max_{\psi,m} m \quad \text{such that} \quad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi\in\Pi} \psi^T E_\pi[\mathbf{f}(\mathbf{s},\mathbf{a})] + m$$

need to somehow "weight" by similarity between $\pi^\star$ and π

$\forall \pi$

$$\mathbb{E}_{\pi^\star}\{\psi^T f\} = \psi^T \mathbb{E}_{\pi^\star}\{f\} \geq \psi^T \mathbb{E}_{\pi}\{f\} + m$$

Feature Matching IRL & Maximum Margin

- remember the "SVM trick":

$$\max_{\psi,m} m \qquad \textit{such that} \qquad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi \in \Pi} \psi^T E_{\pi}[\mathbf{f}(\mathbf{s},\mathbf{a})] + m$$

Feature Matching IRL & Maximum Margin

- remember the "SVM trick":

$$\max_{\psi,m} m \quad \text{such that} \quad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi\in\Pi} \psi^T E_{\pi}[\mathbf{f}(\mathbf{s},\mathbf{a})] + m$$

$$\min_{\psi} \frac{1}{2} \| \psi \|^2 \quad \text{such that} \quad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi\in\Pi} \psi^T E_{\pi}[\mathbf{f}(\mathbf{s},\mathbf{a})] + 1$$

$\pi^* \in \Pi$

Feature Matching IRL & Maximum Margin

- remember the "SVM trick":

$$\max_{\psi,m} m \quad \textit{such that} \quad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi\in\Pi} \psi^T E_{\pi}[\mathbf{f}(\mathbf{s},\mathbf{a})] + m$$

$$\Updownarrow$$

$$\min_{\psi} \frac{1}{2} \| \psi \|^2 \quad \textit{such that} \quad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi\in\Pi} \psi^T E_{\pi}[\mathbf{f}(\mathbf{s},\mathbf{a})] + D(\pi, \pi^\star)$$

Feature Matching IRL & Maximum Margin

- remember the "SVM trick":

$$\max_{\psi,m} m \qquad \textit{such that} \qquad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi \in \Pi} \psi^T E_{\pi}[\mathbf{f}(\mathbf{s},\mathbf{a})] + m$$

$$\min_{\psi} \frac{1}{2} \| \psi \|^2 \ \textit{such that} \quad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi \in \Pi} \psi^T E_{\pi}[\mathbf{f}(\mathbf{s},\mathbf{a})] + D(\pi,\pi^\star)$$

($D(\pi,\pi^\star)$: e.g., difference in feature expectations!)

Feature Matching IRL & Maximum Margin

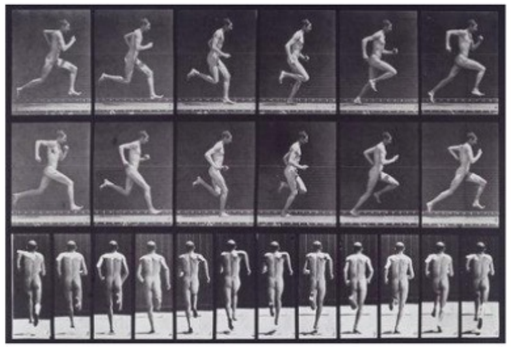
- remember the "SVM trick":

$$\max_{\psi,m} m \qquad \textit{such that} \qquad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi \in \Pi} \psi^T E_{\pi}[\mathbf{f}(\mathbf{s},\mathbf{a})] + m$$

$$\min_{\psi} \frac{1}{2} \| \psi \|^2 \ \textit{such that} \quad \psi^T E_{\pi^\star}[\mathbf{f}(\mathbf{s},\mathbf{a})] \geq \max_{\pi \in \Pi} \psi^T E_{\pi}[\mathbf{f}(\mathbf{s},\mathbf{a})] + D(\pi,\pi^\star)$$ ($D(\pi,\pi^\star)$: e.g., difference in feature expectations!)

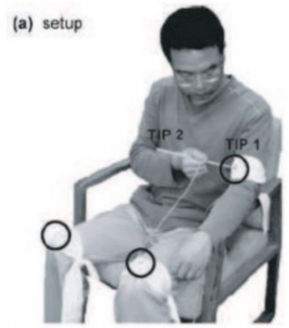
- Issues:
  - Maximizing the margin is a bit arbitrary
  - No clear model of expert suboptimality (can add slack variables...)
  - Messy constrained optimization problem – not great for deep learning!
- Further reading:
  - Abbeel & Ng: Apprenticeship learning via inverse reinforcement learning
  - Ratliff et al: Maximum margin planning

Modeling Human Behavior with Optimal Control



Muybridge (c. 1870)

Mombaur et al. '09



Li & Todorov '06

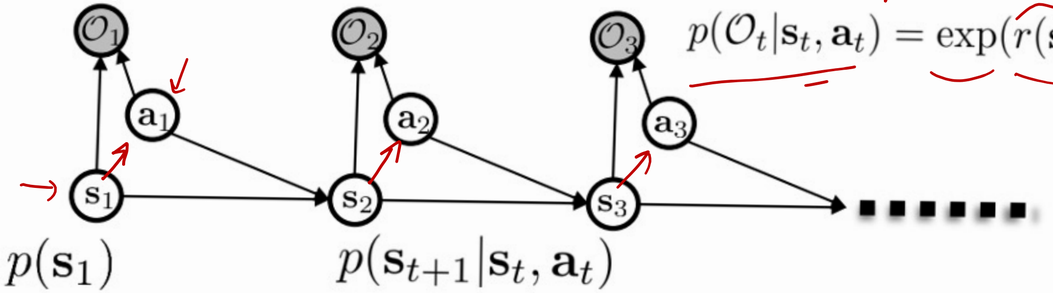
Ziebart '08

$O_i \in \{0, 1\}$

$r \leq 0$

$r \Rightarrow \psi$

$p(\mathbf{s}_{1:T}, \mathbf{a}_{1:T} | \mathcal{O}_{1:T})$



$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t) = \exp(r(\mathbf{s}_t, \mathbf{a}_t)) \leq 1$

$\propto p(s_{1:T}, a_{1:T}, O_{1:T})$

$= p(s_1) \cdot p(a_1) \cdot p(O_1|s_1, a_1) \cdot p(s_2|s_1, a_1) \cdot p(a_2) \cdot p(O_2|s_2, a_2) \cdots$

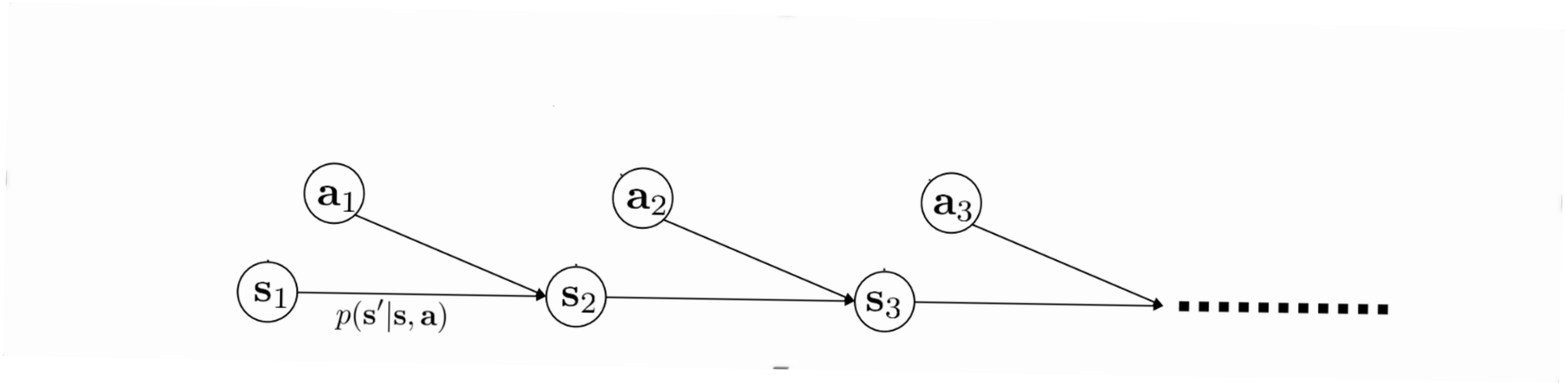
$\pi(a_1|s_1)$

$\pi(a_2|s_2)$

A probabilistic graphical model of decision making

- $p(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}) = ??$

τ



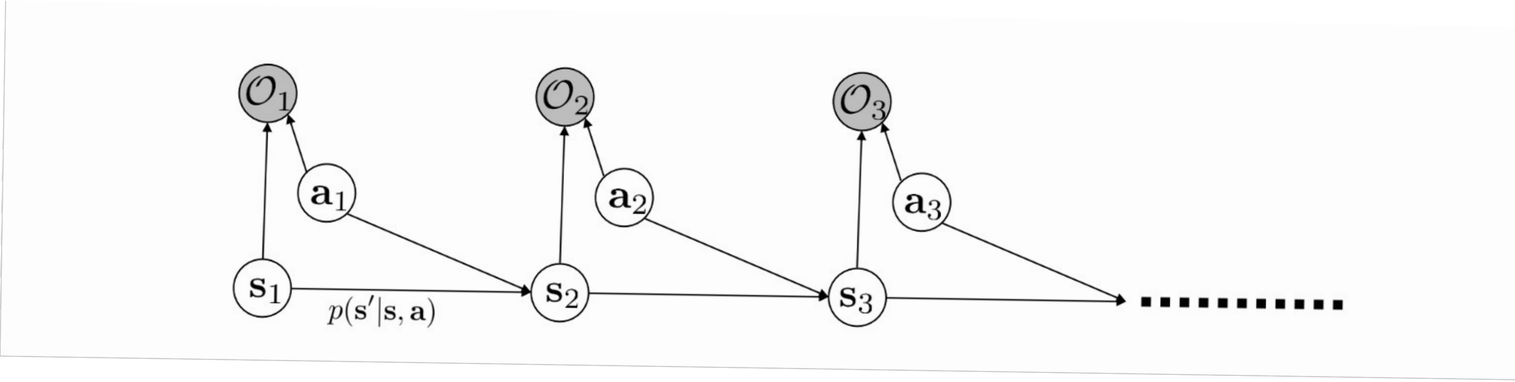
A probabilistic graphical model of decision making

- $p(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})$ =?? no assumption of optimal behavior!

τ

$$p(\tau|\mathcal{O}_{1:T})$$

$$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t) = \exp(r(\mathbf{s}_t, \mathbf{a}_t))$$



A probabilistic graphical model of decision making

- $p(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})$ =?? no assumption of optimal behavior!

τ

$$p(\tau|\mathcal{O}_{1:T})$$

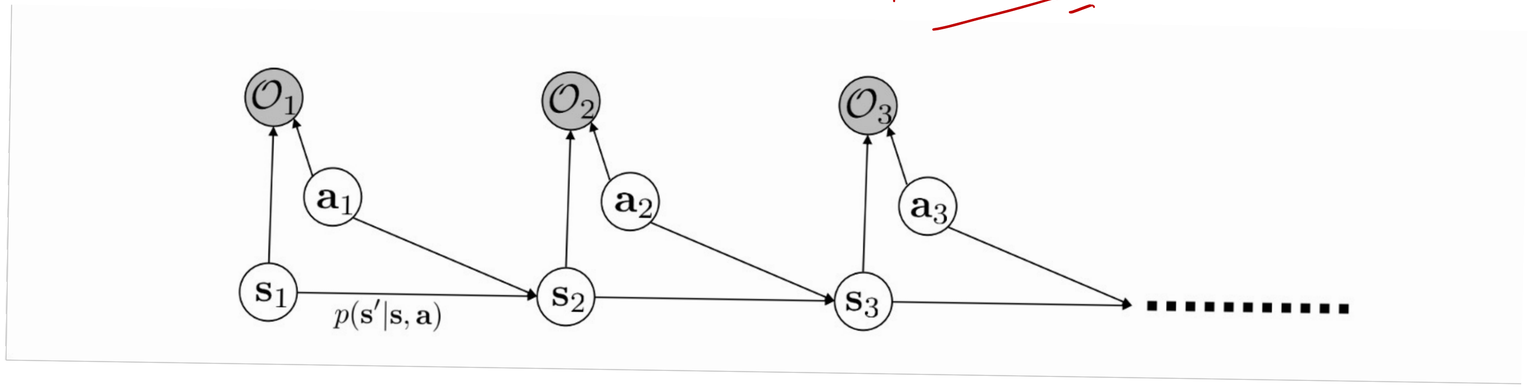
$$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t) = \exp(r(\mathbf{s}_t, \mathbf{a}_t))$$

$$p(\tau|\mathcal{O}_{1:T}) = \frac{p(\tau, \mathcal{O}_{1:T})}{p(\mathcal{O}_{1:T})}$$

$$\propto p(\tau)\prod_t \exp(r(\mathbf{s}_t, \mathbf{a}_t)) = p(\tau)\exp\left(\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right)$$

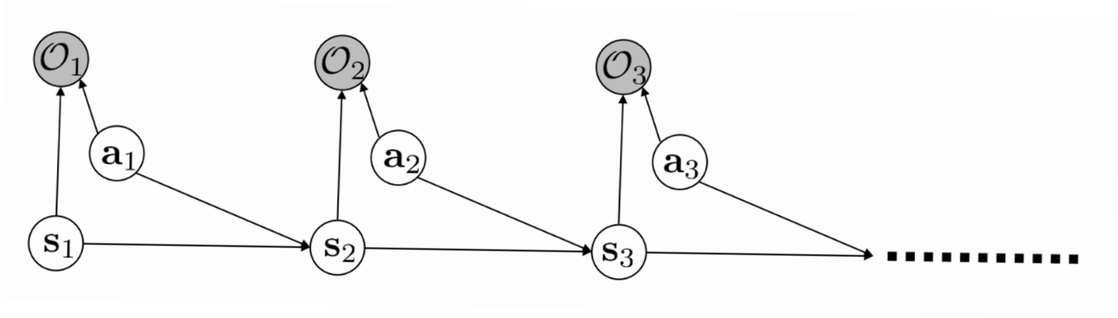
R

Soft optimality



Learning the optimality variable

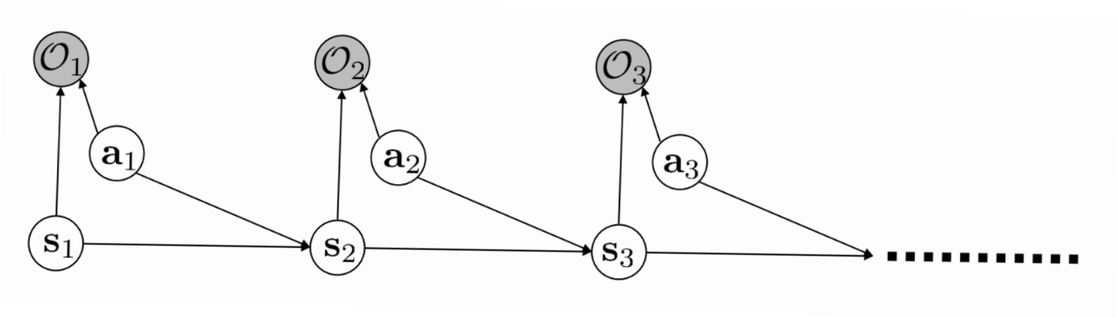
$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t) = \exp(r_\psi(\mathbf{s}_t, \mathbf{a}_t))$ ψ is reward parameters



Learning the optimality variable

$$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t, \psi) = \exp(r_\psi(\mathbf{s}_t, \mathbf{a}_t))$$

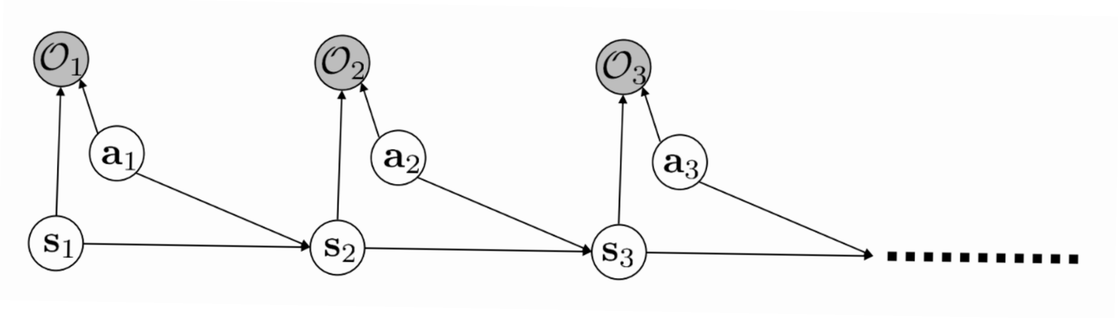
$$p(\tau|\mathcal{O}_{1:T}, \psi)$$



Learning the optimality variable

$$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t, \psi) = \exp(r_\psi(\mathbf{s}_t, \mathbf{a}_t))$$

$$p(\tau|\mathcal{O}_{1:T}, \psi) \propto p(\tau)\exp\left(\sum_t r_\psi(\mathbf{s}_t, \mathbf{a}_t)\right)$$



Learning the optimality variable

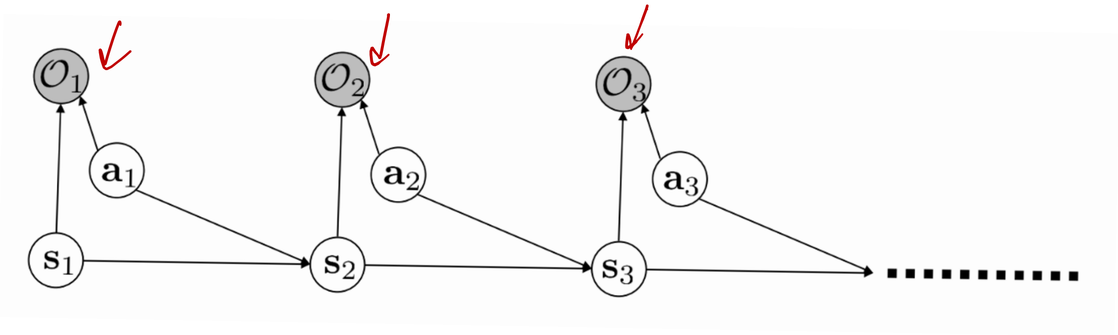
$$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t, \psi) = \exp(r_\psi(\mathbf{s}_t, \mathbf{a}_t))$$

$$p(\tau|\mathcal{O}_{1:T}, \psi) \propto p(\tau)\exp\left(\sum_t r_\psi(\mathbf{s}_t, \mathbf{a}_t)\right)$$

given:

samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$

not necessarily Opt.



Learning the optimality variable

$$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t, \psi) = \exp(r_\psi(\mathbf{s}_t, \mathbf{a}_t))$$

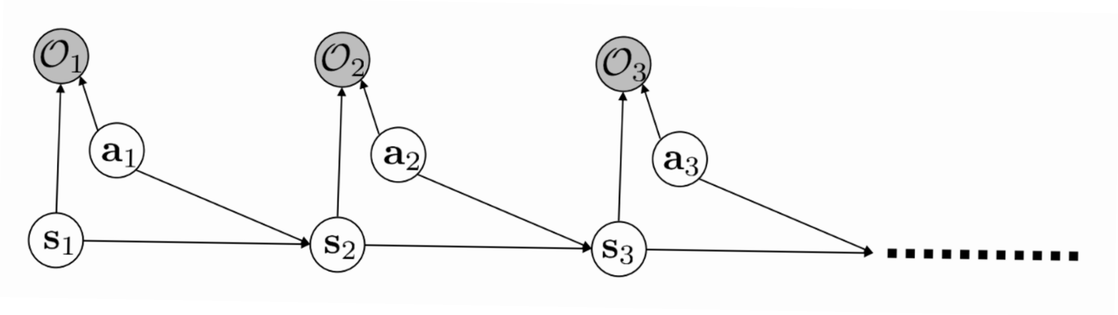
$$p(\tau|\mathcal{O}_{1:T}, \psi) \propto p(\tau)\exp\left(\sum_t r_\psi(\mathbf{s}_t, \mathbf{a}_t)\right)$$

- maximum likelihood learning:

$$\max_\psi \frac{1}{N}\sum_{i=1}^{N} \log p(\tau_i|\mathcal{O}_{1:T}, \psi)$$

given:

samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$



Learning the optimality variable

$$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t, \psi) = \exp(r_\psi(\mathbf{s}_t, \mathbf{a}_t))$$

$$p(\tau|\mathcal{O}_{1:T}, \psi) \propto p(\tau)\exp\left(\sum_t r_\psi(\mathbf{s}_t, \mathbf{a}_t)\right)$$

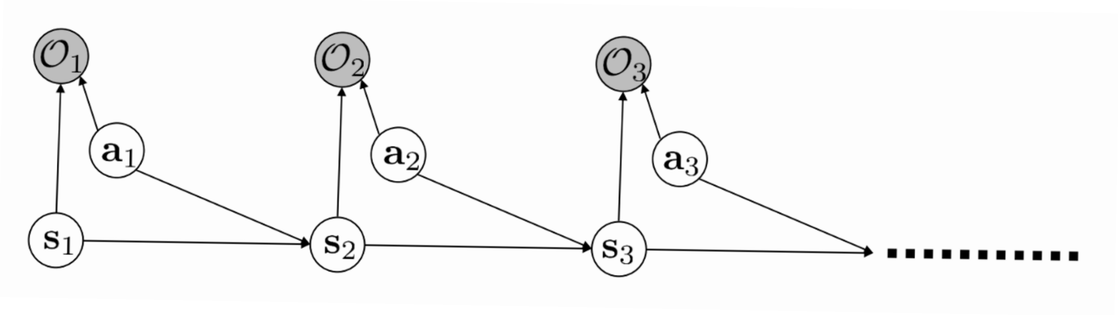
can ignore (independent of ψ)

- maximum likelihood learning:

$$\max_\psi \frac{1}{N}\sum_{i=1}^{N} \log p(\tau_i|\mathcal{O}_{1:T}, \psi)$$

given:

samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$



Learning the optimality variable

$$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t, \psi) = \exp(r_\psi(\mathbf{s}_t, \mathbf{a}_t))$$

$$p(\tau|\mathcal{O}_{1:T}, \psi) \propto p(\tau)\exp\left(\sum_t r_\psi(\mathbf{s}_t, \mathbf{a}_t)\right)$$

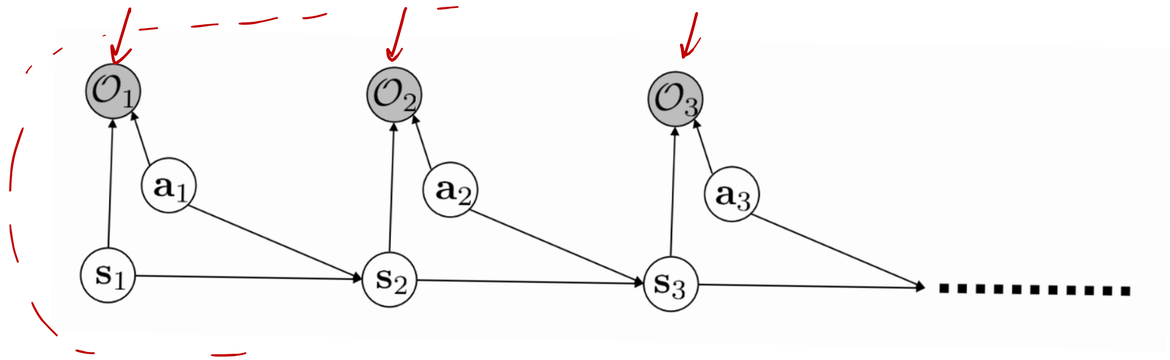
can ignore (independent of ψ)

- maximum likelihood learning:

$$\max_\psi \frac{1}{N}\sum_{i=1}^{N} \log p(\tau_i|\mathcal{O}_{1:T}, \psi) = \max_\psi \frac{1}{N}\sum_{i=1}^{N} r_\psi(\tau_i) - \log Z$$

given:

samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$



Learning the optimality variable

$p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t, \psi) = \exp(r_\psi(\mathbf{s}_t, \mathbf{a}_t))$

$p(\tau|\mathcal{O}_{1:T}, \psi) \propto p(\tau)\exp\left(\sum_t r_\psi(\mathbf{s}_t, \mathbf{a}_t)\right)$

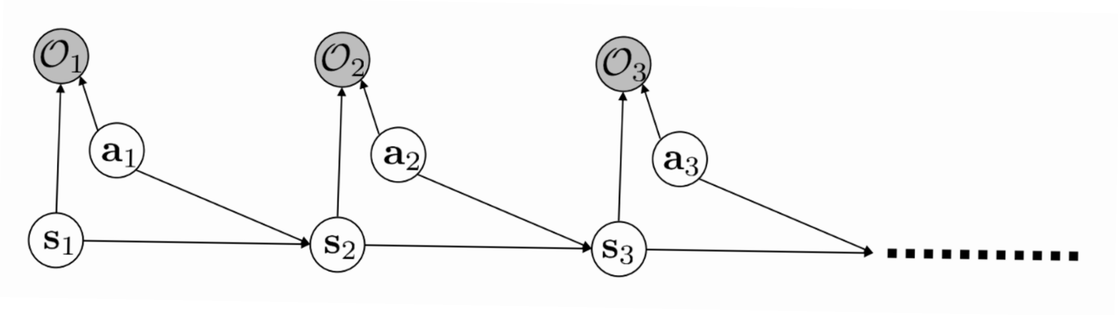
can ignore (independent of ψ)

given:

samples $\{\tau_i\}$ sampled from $\pi^\star(\tau)$

- maximum likelihood learning:

$$\max_\psi \frac{1}{N}\sum_{i=1}^{N} \log p(\tau_i|\mathcal{O}_{1:T}, \psi) = \max_\psi \frac{1}{N}\sum_{i=1}^{N} r_\psi(\tau_i) - \log Z$$ ($\log Z$: partition function (the hard part))



The IRL Partition Function

$$\max_{\psi} \frac{1}{N} \sum_{i=1}^{N} r_{\psi}(\tau_i) - \log Z$$

The IRL Partition Function

Backprop.

$$\max_{\psi} \frac{1}{N} \sum_{i=1}^{N} r_\psi(\tau_i) - \log Z$$

$$p(\tau|O,\psi) = \frac{1}{Z} p(\tau) \cdot \exp(R_\psi(\tau))$$

$$\int p(\tau|O,\psi)\, d\tau = \frac{1}{Z} \int p(\tau) \exp(R_\psi(\tau))\, d\tau$$

$$\int p(\tau|O,\psi)\, d\tau = 1$$

$$Z = \int p(\tau) \exp(r_\psi(\tau))\, d\tau$$

$r \to R$

The IRL Partition Function

$$\max_{\psi} \frac{1}{N} \sum_{i=1}^{N} r_\psi(\tau_i) - \log Z$$

$$\nabla Z = \int p(\tau) \exp(r_\psi(\tau)) d\tau$$

$$\nabla_\psi \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{Z} \int p(\tau) \exp(r_\psi(\tau)) \nabla_\psi r_\psi(\tau) d\tau$$

$$\int \frac{1}{Z} p(\tau) \exp(R_\psi(\tau)) \cdot \nabla_\psi R_\psi(\tau) d\tau$$

$$p(\tau | O, \psi)$$

The IRL Partition Function

$$\max_{\psi} \frac{1}{N} \sum_{i=1}^{N} r_\psi(\tau_i) - \log Z$$

$$Z = \int p(\tau) \exp(r_\psi(\tau)) d\tau$$

$$\nabla_\psi \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{Z} \int p(\tau) \exp(r_\psi(\tau)) \nabla_\psi r_\psi(\tau) d\tau$$

$$p(\tau | \mathcal{O}_{1:T}, \psi)$$

The IRL Partition Function

$$\max_{\psi} \frac{1}{N} \sum_{i=1}^{N} r_\psi(\tau_i) - \log Z$$

$$Z = \int p(\tau) \exp(r_\psi(\tau)) d\tau$$

$$\nabla_\psi \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{Z} \int p(\tau) \exp(r_\psi(\tau)) \nabla_\psi r_\psi(\tau) d\tau$$

$$p(\tau | \mathcal{O}_{1:T}, \psi)$$

$$\nabla_\psi \mathcal{L} = E_{\tau \sim \pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - E_{\tau \sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]$$

traj. from the graphical model

$\equiv$ RL on r_ψ

The IRL Partition Function

$$\max_{\psi} \frac{1}{N} \sum_{i=1}^{N} r_\psi(\tau_i) - \log Z$$

$$Z = \int p(\tau) \exp(r_\psi(\tau)) d\tau$$

$$\nabla_\psi \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{Z} \int p(\tau) \exp(r_\psi(\tau)) \nabla_\psi r_\psi(\tau) d\tau$$

$$p(\tau|\mathcal{O}_{1:T}, \psi)$$

$$\nabla_\psi \mathcal{L} = E_{\tau \sim \pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - E_{\tau \sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]$$

"estimate with expert samples"

"soft optimal policy under current reward"

SAC

Unknown Dynamics & Large State/Action Spaces

Assume we don't know the dynamics, but we can sample, like in standard RL

Unknown Dynamics & Large State/Action Spaces

Assume we don't know the dynamics, but we can sample, like in standard RL

recall:

$$\nabla_\psi \mathcal{L} = E_{\tau \sim \pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - E_{\tau \sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]$$

Unknown Dynamics & Large State/Action Spaces

Assume we don't know the dynamics, but we can sample, like in standard RL

recall:

$$\nabla_\psi \mathcal{L} = E_{\tau \sim \pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - E_{\tau \sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]$$

"estimate with expert samples"

"soft optimal policy under current reward"

Unknown Dynamics & Large State/Action Spaces

Bayesian RL

Assume we don't know the dynamics, but we can sample, like in standard RL

recall:

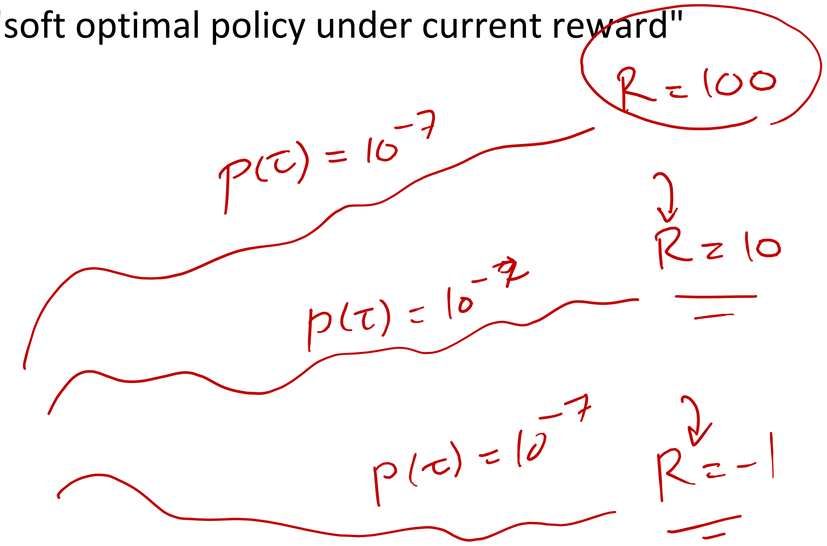
$$p(\tau \mid O_{1:T}=\{1,\dots 1\}, \psi) \propto p(\tau) \cdot \exp\{\sum r_i\}$$

$$\nabla_\psi \mathcal{L} = E_{\tau\sim\pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - E_{\tau\sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]$$

"estimate with expert samples"

"soft optimal policy under current reward"

- idea: learn $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm



Unknown Dynamics & Large State/Action Spaces

Assume we don't know the dynamics, but we can sample, like in standard RL

recall:

$$\nabla_\psi \mathcal{L} = E_{\tau\sim\pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - E_{\tau\sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]$$

"estimate with expert samples"

"soft optimal policy under current reward"

- idea: learn $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

$$J(\theta) = \sum_t E_{\pi(\mathbf{s}_t,\mathbf{a}_t)}[r_\psi(\mathbf{s}_t,\mathbf{a}_t)] + E_{\pi(\mathbf{s}_t)}[\mathcal{H}(\pi(\mathbf{a}|\mathbf{s}_t))]$$

Unknown Dynamics & Large State/Action Spaces

Assume we don't know the dynamics, but we can sample, like in standard RL

recall:

$$\nabla_\psi \mathcal{L} = E_{\tau \sim \pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - E_{\tau \sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]$$

"estimate with expert samples"

"soft optimal policy under current reward"

- idea: learn $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

$$J(\theta) = \sum_t E_{\pi(\mathbf{s}_t,\mathbf{a}_t)}[r_\psi(\mathbf{s}_t, \mathbf{a}_t)] + E_{\pi(\mathbf{s}_t)}[\mathcal{H}(\pi(\mathbf{a}|\mathbf{s}_t))]$$

then run this policy to sample $\{\tau_j\}$

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M}\sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Unknown Dynamics & Large State/Action Spaces

Assume we don't know the dynamics, but we can sample, like in standard RL

recall:

$$\nabla_\psi \mathcal{L} = E_{\tau\sim\pi^\star(\tau)}[\nabla_\psi r_\psi(\tau_i)] - E_{\tau\sim p(\tau|\mathcal{O}_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]$$

"estimate with expert samples"

"soft optimal policy under current reward"

- idea: learn $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

$$J(\theta) = \sum_t E_{\pi(\mathbf{s}_t,\mathbf{a}_t)}[r_\psi(\mathbf{s}_t,\mathbf{a}_t)] + E_{\pi(\mathbf{s}_t)}[\mathcal{H}(\pi(\mathbf{a}|\mathbf{s}_t))]$$

then run this policy to sample $\{\tau_j\}$

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M}\sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Sum over expert samples

Sum over policy samples

More Efficient Sample-Based Updates

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M} \sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Sum over expert samples

Sum over policy samples

learn $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

More Efficient Sample-Based Updates

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M} \sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Sum over expert samples

Sum over policy samples

learn $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

then run this policy to sample $\{\tau_j\}$

More Efficient Sample-Based Updates

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M} \sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Sum over expert samples

Sum over policy samples

learn $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

then run this policy to sample $\{\tau_j\}$

looks expensive! what if we use "lazy" policy optimization?

More Efficient Sample-Based Updates

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M} \sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Sum over expert samples

Sum over policy samples

Improve ~~learn~~ $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

then run this policy to sample $\{\tau_j\}$

looks expensive! what if we use "lazy" policy optimization?

More Efficient Sample-Based Updates

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M} \sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Sum over expert samples

Sum over policy samples

Improve (a little) ~~learn~~ $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

then run this policy to sample $\{\tau_j\}$

looks expensive! what if we use "lazy" policy optimization?

More Efficient Sample-Based Updates

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M}\sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Sum over expert samples

Sum over policy samples

Improve (a little) ~~learn~~ $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

then run this policy to sample $\{\tau_j\}$

looks expensive! what if we use "lazy" policy optimization?

- problem: estimator is now biased! wrong distribution!

$\frac{1}{M}\sum_{i=1}^{M} \frac{p(x_i)}{q(x_i)} g(x_i) = \mathbb{E}_{x\sim p}[g(x)]$

$x_1 \cdots x_N \sim q \neq p$

$\mathbb{E}_{x_i\sim q}\left\{ g(x_i) \cdot \frac{p(x_i)}{q(x_i)} \right\}$ imp. weight

Issue: By "lazy policy optimization", τ_j's wouldn't be soft optimal anymore, making $\frac{1}{M}\sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$ biased.

Problem

More Efficient Sample-Based Updates

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M}\sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Sum over expert samples

Sum over policy samples

Improve (a little) ~~learn~~ $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

then run this policy to sample $\{\tau_j\}$

looks expensive! what if we use "lazy" policy optimization?

- problem: estimator is now biased! wrong distribution!
  - solution 1: use importance sampling

importance weight

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j}\sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$$

More Efficient Sample-Based Updates

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M}\sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$$

Sum over expert samples

Sum over policy samples

Improve (a little) ~~learn~~ $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}, \psi)$ using any max-ent RL algorithm

then run this policy to sample $\{\tau_j\}$

looks expensive! what if we use "lazy" policy optimization?

- problem: estimator is now biased! wrong distribution!
  - solution 1: use importance sampling

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j}\sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$$

$$w_j = \frac{p(\tau)\exp(r_\psi(\tau_j))}{\pi(\tau_j)}$$

SAC policy under improvement

Importance Sampling

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j} \sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$$

$$w_j = \frac{p(\tau) \exp(r_\psi(\tau_j))}{\pi(\tau_j)}$$

Importance Sampling

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j}\sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$$

$$w_j = \frac{p(\tau)\exp(r_\psi(\tau_j))}{\pi(\tau_j)}$$

$$= \frac{p(\mathbf{s}_1)\prod_t p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)\exp(r_\psi(\mathbf{s}_t,\mathbf{a}_t))}{p(\mathbf{s}_1)\prod_t p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)\pi(\mathbf{a}_t|\mathbf{s}_t)}$$

Importance Sampling

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j}\sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$$

$$w_j = \frac{p(\tau)\exp(r_\psi(\tau_j))}{\pi(\tau_j)}$$

$$= \frac{p(\mathbf{s}_1)\prod_t p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)\exp(r_\psi(\mathbf{s}_t,\mathbf{a}_t))}{p(\mathbf{s}_1)\prod_t p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)\pi(\mathbf{a}_t|\mathbf{s}_t)}$$

Importance Sampling

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j}\sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$$

$$w_j = \frac{p(\tau)\exp(r_\psi(\tau_j))}{\pi(\tau_j)}$$

$$= \frac{p(\mathbf{s}_1)\prod_t p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)\exp(r_\psi(\mathbf{s}_t,\mathbf{a}_t))}{p(\mathbf{s}_1)\prod_t p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)\pi(\mathbf{a}_t|\mathbf{s}_t)}$$

$$= \frac{\exp(\sum_t r_\psi(\mathbf{s}_t,\mathbf{a}_t))}{\prod_t \pi(\mathbf{a}_t|\mathbf{s}_t)}$$

Importance Sampling

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j}\sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$$

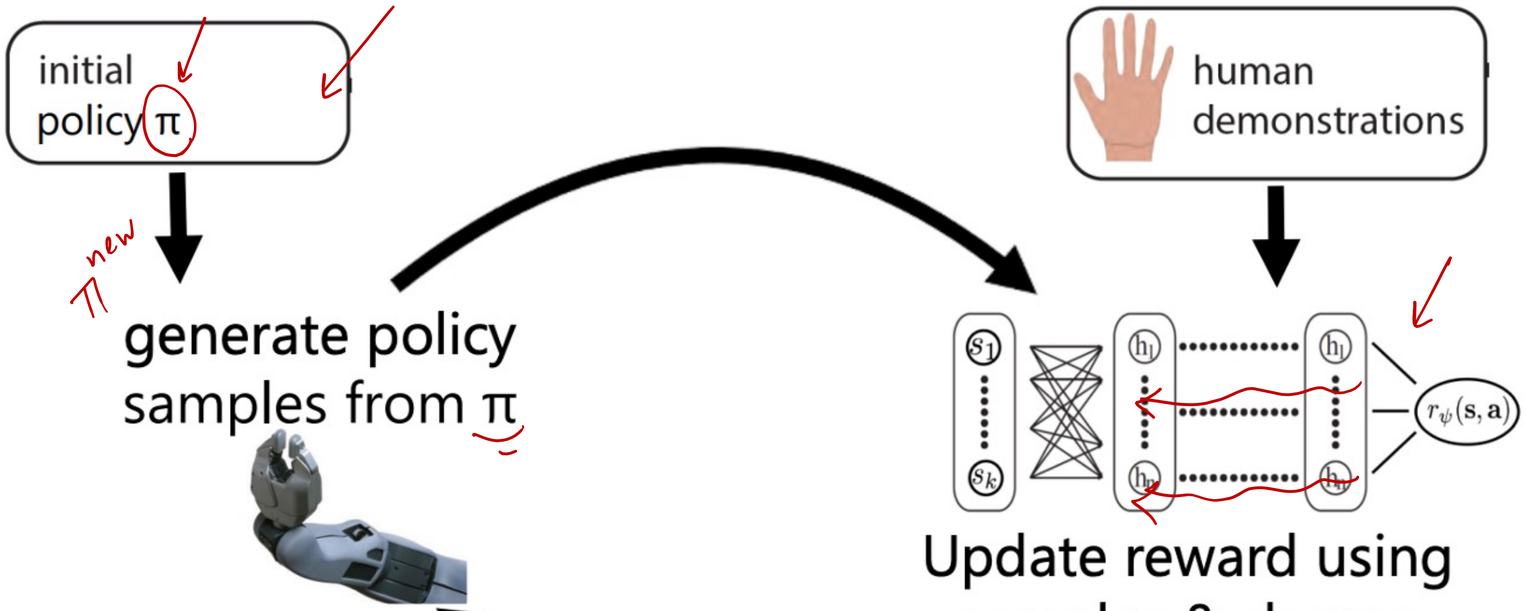
$$w_j = \frac{p(\tau)\exp(r_\psi(\tau_j))}{\pi(\tau_j)}$$

$$= \frac{p(\mathbf{s}_1)\prod_t p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)\exp(r_\psi(\mathbf{s}_t,\mathbf{a}_t))}{p(\mathbf{s}_1)\prod_t p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)\pi(\mathbf{a}_t|\mathbf{s}_t)}$$

$$= \frac{\exp(\sum_t r_\psi(\mathbf{s}_t,\mathbf{a}_t))}{\prod_t \pi(\mathbf{a}_t|\mathbf{s}_t)}$$

each policy update w.r.t. r_ψ brings us closer to the target distribution!

Guided Cost Learning Algorithm (Finn et al. ICML '16)



$$w_j = \frac{\exp(r_\psi(\tau_j))}{\pi(\tau_j)}$$

$$\nabla_\psi \mathcal{L} \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j}\sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$$

Suggested Reading on Inverse RL

- Classic Papers:
  - Abbeel & Ng ICML '04. Apprenticeship Learning via Inverse Reinforcement Learning. Good introduction to inverse reinforcement learning
  - Ziebart et al. AAAI '08. Maximum Entropy Inverse Reinforcement Learning. Introduction to probabilistic method for inverse reinforcement learning
- Modern Papers:
  - Finn et al. ICML '16. Guided Cost Learning. Sampling based method for MaxEnt IRL that handles unknown dynamics and deep reward functions
  - Wulfmeier et al. arXiv '16. Deep Maximum Entropy Inverse Reinforcement Learning. MaxEnt inverse RL using deep reward functions
  - Ho & Ermon NIPS '16. Generative Adversarial Imitation Learning. Inverse RL method using generative adversarial networks
  - Fu, Luo, Levine ICLR '18. Learning Robust Rewards with Adversarial Inverse Reinforcement Learning