Theory and Methods for Reinforcement Learning

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 12: Inverse Reinforcement Learning

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-618 (Spring 2020)

License Information for Reinforcement Learning Slides

- This lecture
  1. Inverse Reinforcement Learning

Recommended reading

- Andrew Y. Ng, and Stuart Russel. *Algorithms for Inverse Reinforcement Learning.* Proceedings of the twenty-first international conference on Machine learning. ACM, 2000
- Abbeel, Pieter, and Andrew Y. Ng. *Apprenticeship learning via inverse reinforcement learning.* Proceedings of the twenty-first international conference on Machine learning. ACM, 2004
- Ziebart, Brian D., et al. *Maximum Entropy Inverse Reinforcement Learning.* AAAI. Vol. 8. 2008.
- Osa, Takayuki, et al. *An algorithmic perspective on imitation learning.* Foundations and Trends®in Robotics 7.1-2 (2018): 1-179

Motivation

Motivation

So far we have manually designed reward function to define a task. Given an expert's behaviour, can we learn the reward function?

Learning from Demonstrations (LfD)

- **Setting**: An oracle teaches an agent how to perform a given task.

- **Given**: Samples of an MDP agent's behavior over time and in different circumstances, from a supposedly optimal policy π^*, i.e.,
  - A set of trajectories $\{\xi_i\}_{i=1}^n$, $\xi_i = \{(s_t, a_t)\}_{t=0}^{H_i-1}$, $a_t \sim \pi^*(s_t)$.
  - Reward signal $r_t = R(s_t, a_t, s_{t+1})$ unobserved
  - Transition model $T(s, a, s') = P(s' \mid s, a)$ known/unknown.

- **Goals**:
  - Recover teacher's policy π^* directly: **behavioral cloning**, or **imitation learning**.
  - Recover teacher's latent reward function $R^*(s, a, s')$: **IRL**.
  - Recover teacher's policy π^* indirectly by first recovering $R^*(s, a, s')$: **apprenticeship learning via IRL**.

IRL Formulation 1: Small, Discrete MDPs

- **Given**: An incomplete MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, R, \gamma)$
  - known transition model $T(s, a, s') = P(s' \mid s, a), \forall s, a, s'$
  - unobserved but bounded reward signal, $|R(s, a, s')| \leq r_{\max}, \forall s, a, s'$ (for simplicity, consider state-dependent reward functions, $R(s)$)
  - known, supposedly optimal policy $\pi^*(s), \forall s \in \mathcal{S}$, instead of $\{\xi_i\}_{i=1}^n$.

- **Find** $R : \mathcal{S} \to [-r_{\max}, r_{\max}]$ such that teacher's policy π^* is optimal,
  - furthermore: simple, and robust reward function
  - Notes: in the following we fix an enumeration on the state space: $\mathcal{S} = \left\{s_1, \ldots, s_{|\mathcal{S}|}\right\}$. Then R is a column vector in $\mathbb{R}^{|\mathcal{S}|}$, with $R_i = R(s_i)$.

IRL Formulation 1: Small, Discrete MDPs

- **Find** $R \in \mathbb{R}^{|\mathcal{S}|}$ such that teacher's policy π^* is optimal.
- Recall Bellman optimality theorem (for a known MDP):

$$
\begin{aligned}
& \pi^* \text{ is optimal} \\
\iff \quad & \pi^*(s) \in \arg\max_a Q^{\pi^*}(s,a), \quad \forall s \in \mathcal{S} \\
\iff \quad & Q^{\pi^*}(s, \pi^*(s)) \ \geq \ Q^{\pi^*}(s,a), \quad \forall s \in \mathcal{S}, a \in \mathcal{A} \qquad (1)
\end{aligned}
$$

- Define policy-conditioned transition matrices P^* and $P^a \in [0,1]^{|\mathcal{S}| \times |\mathcal{S}|}$:

$$[P^*]_{ij} := P(s_j \mid s_i, \pi^*(s_i)), \text{ and } [P^a]_{ij} := P(s_j \mid s_i, a), \ \forall s_i, s_j \in \mathcal{S}$$

IRL Formulation 1: Small, Discrete MDPs

- We can represent the constraints on R as [3]:

$$(P^* - P^a)\,(I - \gamma P^*)^{-1}\,R \;\succeq\; \mathbf{0},\ \forall a \in \mathcal{A} \tag{2}$$

- Proof:
  - Bellman equations $\implies Q^{\pi^*}(s,a) = R(s) + \gamma \sum_{s'} P(s' \mid s,a) V^{\pi^*}(s')$, and $V^{\pi^*} = (I - \gamma P^*)^{-1} R$.
  - Denote by $Q^{\pi^*}_{\pi}$ a length-$|\mathcal{S}|$ column vector with elements $Q^{\pi^*}_{\pi}(s) = Q^{\pi^*}(s, \pi(s))$, i.e., $Q^{\pi^*}_{\pi} = R + \gamma P^{\pi} V^{\pi^*}$.
  - The set of $|\mathcal{S}| \times |\mathcal{A}|$ constraints in Eq. (1) can be written in matrix form (by fixing an action a for all starting states $s \in \mathcal{S}$) as:

$$Q^{\pi^*}_{\pi^*} - Q^{\pi^*}_{a} \succeq \mathbf{0}, \forall a \in \mathcal{A} \iff \text{Eq. (2)}$$

IRL Formulation 1: Small, Discrete MDPs

- Challenges:
  - What if noisy teacher? (i.e., $a_t \neq \pi^*(s_t)$ at some t)
  - Instead of full $\pi^*(s), \forall s \in \mathcal{S}$, only given sampled trajectories $\{\xi_i\}_{i=1}^n$?
  - Computationally expensive/infeasible: $|\mathcal{S}| \times |\mathcal{A}|$ constraints for each R
  - Reward function ambiguity: IRL is ill-posed! ($R = 0$ is a solution.)
  - From reward-shaping theory: If the MDP $\mathcal{M}$ with reward function R admits π^* as an optimal policy, then $\mathcal{M}'$ with affine-transformed reward function below also admits π^* as an optimal policy: $R'(s, a, s') = \alpha R(s, a, s') + \gamma\psi(s') - \psi(s)$, with $\psi : \mathcal{S} \to \mathbb{R}$, $\alpha \neq 0$.

IRL Formulation 1: Small, Discrete MDPs

- One solution (to the reward ambiguity issue): find simple, and robust R,
  - e.g., use ℓ_1-norm penalty $\|R\|_1$, and
  - maximize sum of value-margins $\Delta V^{\pi^*}(s)$ of π^* & second-best action,

$$\Delta V^{\pi^*}(s) = Q^{\pi^*}(s,\pi^*(s)) - \max_{a\neq\pi^*(s)} Q^{\pi^*}(s,a) = \min_{a\neq\pi^*(s)} \left[Q^{\pi^*}(s,\pi^*(s)) - Q^{\pi^*}(s,a)\right]$$

- Combining altogether:

$$\begin{aligned}\max_{R\in\mathbb{R}^{|\mathcal{S}|}} \quad & \left\{\sum_{s\in\mathcal{S}} \min_{a\in\mathcal{A}\setminus\pi^*(s)} \left\{(P_s^* - P_s^a)(I-\gamma P^*)^{-1} R\right\} - \lambda \|R\|_1\right\} \\ \text{s.t.} \quad & (P^* - P^a)(I-\gamma P^*)^{-1} R \succeq \mathbf{0}, \ \forall a \in \mathcal{A} \\ & |R(s)| \leq r_{\max}, \ \forall s \in \mathcal{S}\end{aligned}$$

with P_s^a the row vector of transition probabilities $P(s' \mid s,a), \forall s' \in \mathcal{S}$, i.e., P_s^*, P_s^a are the s-th rows of P^*, P^a, respectively.

IRL Formulation 2: With LFA

- For large/continuous domains, with sampled trajectories.
- Assume $s_0 \sim P_0(\mathcal{S})$; for teacher's policy π^* to be optimal:

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \middle| \pi^*\right] \geq \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \middle| \pi\right], \ \forall \pi$$

- Using LFA: $R(s) = w^\top \phi(s)$, where $w \in \mathbb{R}^n$, $\|w\|_1 \leq 1$, and $\phi : \mathcal{S} \to \mathbb{R}^n$.

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \middle| \pi\right] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t w^\top \phi(s_t) \middle| \pi\right] = w^\top \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t) \middle| \pi\right] = w^\top \mu(\pi)$$

- The problem becomes find w such that $w^\top \mu(\pi^*) \geq w^\top \mu(\pi), \forall \pi$
- $\mu(\pi)$: feature expectation of policy π – evaluated with sampled trajectories from π

$$\mu(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t) \middle| \pi\right] \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T_i} \gamma^t \phi(s_t)$$

Feature Expectation Matching: Max Margin

- Let expert's feature expectation be $\mu_E = \mu(\pi^*)$

- To find a policy who's performance is close to that of the expert's, we need to find a policy $\bar{\pi}$ such that $\|\mu_E - \mu(\bar{\pi})\| \leq \epsilon$:

$$\begin{aligned}\left|\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t)\middle|\bar{\pi}\right] - \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t)\middle|\pi^*\right]\right| &= \left|w^\top \mu(\bar{\pi}) - w^\top \mu_E\right| \\ &\leq \|w\|_2 \|\mu(\bar{\pi}) - \mu_E\|_2 \\ &\leq 1 \cdot \epsilon\end{aligned}$$

- Let Π denote the set of stationary policies for an MDP. Given two policies $\pi_1, \pi_2 \in \Pi$, we can construct a new policy π_3 by mixing them together.

- π_3 operates by flipping a coin with bias λ, and with probability λ picks and always acts according to π_1, and with probability $1 - \lambda$ always acts according to π_2.

- From linearity of expectation, clearly we have that $\mu(\pi_3) = \lambda\mu(\pi_1) + (1-\lambda)\mu(\pi_2)$.

Feature Expectation Matching: Max Margin

Algorithm 1 Max-Margin

Initialize $\pi^{(0)}$, compute $\mu(\pi^{(0)})$, $i = 1$ and $t^0 = \infty$
while $t^{(i)} > \epsilon$ **do**
  Compute $t^{(i)} = \max_w \min_{j \in \{0 \dots (i-1)\}} w^T(\mu_E - \mu^{(j)})$
  Solve for $\pi^{(i)}$ with $R = w^T \phi$
  Compute $\mu^{(i)}$
  Set $i \leftarrow i + 1$
end while

Feature Expectation Matching: Max Margin

• Similar to SVMs, we aim to find the separating hyperplane given a set of points, where μ_E is given a label of 1 and $\{\mu(\pi^{(j)}) : j = 0, \ldots, (i-1)\}$ a label of -1:

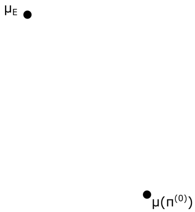
$$\begin{aligned} \max_{t,w} \quad & t \\ \text{s.t.} \quad & w^\top \mu_E \ \geq \ w^\top \mu^{(j)} + t, \quad j = 0, \ldots, i-1 \\ & \|w\|_2 \ \leq \ 1 \end{aligned}$$

• When the algorithm terminates with $t^{(n+1)} \leq \epsilon$, we have:

$$\forall w \text{ with } \|w\|_2 \leq 1 \ \exists i \text{ s.t. } w^\top \mu^{(i)} \geq w^\top \mu_E - \epsilon$$

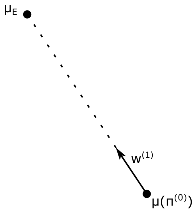
• Since $\|w^*\|_2 \leq \|w^*\|_1 \leq 1$, this means that there is at least one policy from the set returned by the algorithm, whose performance under R^* is at least as good as the expert's performance minus ϵ.

Feature Expectation Matching: Max Margin



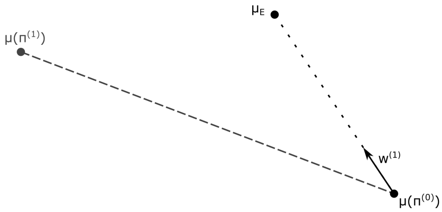
Initialization

Feature Expectation Matching: Max Margin



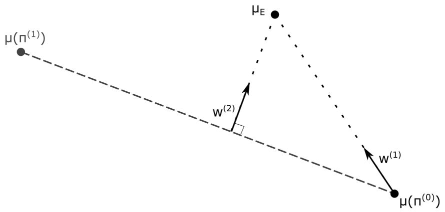
First Iteration

Feature Expectation Matching: Max Margin



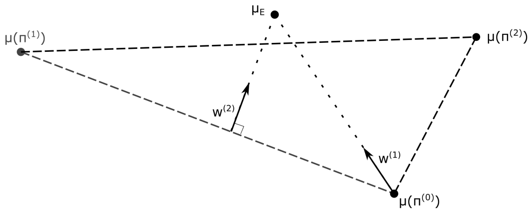
First Iteration

Feature Expectation Matching: Max Margin



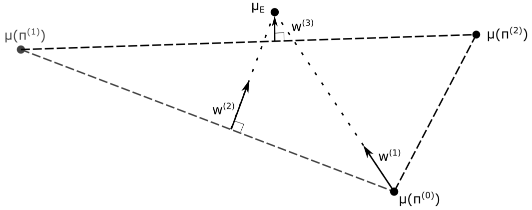
Second Iteration

Feature Expectation Matching: Max Margin



Second Iteration

Feature Expectation Matching: Max Margin



Third Iteration

Feature Expectation Matching: Max Margin

• We can find the point closest to μ_E in the convex closure of $\mu^{(0)}, \ldots, \mu^{(n)}$ by solving the following QP:

$$\min \|\mu_E - \mu\|_2 \text{ s.t. } \mu = \sum_i \lambda_i \mu^{(i)}, \lambda_i \geq 0, \sum_i \lambda_i = 1.$$

• Because μ_E is "separated" from the points $\mu^{(i)}$ by a margin of at most ϵ, we know that for the solution μ we have $\|\mu_E - \mu\|_2 \leq \epsilon$.

• Further, by "mixing" together the policies $\pi^{(i)}$ according to the mixture weights λ_i, we obtain a policy whose feature expectations are given by μ.

Convergence of Max-Margin Algorithm [1]

Theorem

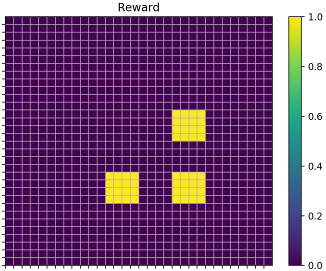
For $\phi : S \to [0,1]^k$, the algorithm terminates after at most

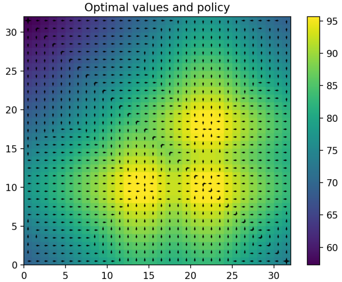
$$\mathcal{O}\left(\frac{k}{(1-\gamma)^2\epsilon^2}\log\frac{k}{(1-\gamma)\epsilon}\right)$$

iterations, where k is the number of features.

Example: Gridworld

- Setup:
  - 32×32 grid
  - Non-overlapping 4×4 regions called macrocells
  - For each of the 64 macrocells, there one feature $\phi_i(s)$ indicating if state s is in macrocell i
  - $R^* = w^T\phi$ where $p(w_i = 0) = 0.95$ and $p(w_i = 1) = 0.05$
  - $\gamma = 0.99$





Example: Gridworld

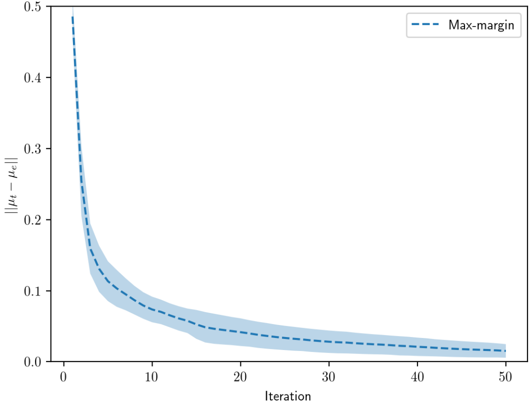


Figure: Convergence of Max-Margin algorithm

Maximum Causal Entropy IRL [5, 4]

- The underlying reward function is given by $R^E : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$.

- We consider the learner model with parametric reward function $R^L_\lambda : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ where $\lambda \in \mathbb{R}^d$ is a parameter.

- The reward function also depends on the learner's feature representation $\phi^L : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^{d'}$.

- For linear reward model, λ represents the weights as $R^L_\lambda(s,a) = \lambda^\top \phi^L(s,a)$.

- As an example of a non-linear reward model, λ could be the weights of a neural network with $\phi^L(s,a)$ as input layer and $R^L_\lambda(s,a)$ as output.

Maximum Causal Entropy IRL

• For any policy π, the occupancy measure ρ and the total expected reward ν of π in the MDP $\mathcal{M}$ are defined as follows respectively:

$$\begin{aligned}\rho^{\pi}(s,a) &:= (1-\gamma)\,\pi(a \mid s)\sum_{\tau=0}^{\infty}\gamma^{\tau}\mathbb{P}\left\{S_{\tau}=s \mid \pi,\mathcal{M}\right\}\\ \nu^{\pi} &:= \frac{1}{1-\gamma}\sum_{s,a}\rho^{\pi}(s,a)\,R^{E}(s,a)\end{aligned}$$

Here, $\mathbb{P}\left\{S_{\tau}=s \mid \pi,\mathcal{M}\right\}$ denotes the probability of visiting the state s after τ steps by following the policy π.

• Similarly, for any demonstration $\xi = \{(s_{\tau}, a_{\tau})\}_{\tau=0,1,\ldots}$, we define

$$\rho^{\xi}(s,a) := (1-\gamma)\sum_{\tau=0}^{\infty}\gamma^{\tau}\mathbb{I}\left\{s_{\tau}=s, a_{\tau}=a\right\}$$

Then for a collection of demonstrations $\Xi = \{\xi_t\}_{t=1,2,\ldots}$, we have $\rho^{\Xi}(s,a) := \frac{1}{|\Xi|}\sum_t \rho^{\xi_t}(s,a)$.

Maximum Causal Entropy IRL

- Let π^L denote the learner's final policy at the end of teaching.

- The performance of the policy π^L (w.r.t. π^E) in $\mathcal{M}$ can be evaluated via the following measures (for some fixed $\epsilon > 0$):

1. $\left| \nu^{\pi^E} - \nu^{\pi^L} \right| \leq \epsilon$, ensuring high reward [1, 4].
2. $D_{\mathrm{TV}}\left(\rho^{\pi^E}, \rho^{\pi^L}\right) \leq \epsilon$, ensuring that learner's behavior induced by the policy π^L matches that of the teacher [2]. Here $D_{\mathrm{TV}}(p, q)$ is the total variation distance between two probability measures p and q.

Maximum Causal Entropy IRL

- Given a collection of demonstrations $\Xi = \{\xi_t\}_{t=1,2,\ldots}$ (where $\xi_t = \{(s_{t,\tau}, a_{t,\tau})\}_{\tau=0,1,\ldots}$), the MCE-IRL algorithm returns a parametric policy:

$$
\begin{aligned}
\pi_\lambda^L(a \mid s) &= \exp\left(Q_\lambda(s,a) - V_\lambda(s)\right) \\
V_\lambda(s) &= \log \sum_a \exp\left(Q_\lambda(s,a)\right) \\
Q_\lambda(s,a) &= R_\lambda^L(s,a) + \gamma \sum_{s'} T(s' \mid s,a) V_\lambda\left(s'\right).
\end{aligned}
\tag{3}
$$

- The optimal parameter is obtained via solving

$$
\underset{\lambda}{\text{maximize}} \quad c(\lambda; \Xi) := \sum_t \sum_\tau \log \pi_\lambda^L(a_{t,\tau} \mid s_{t,\tau}). \tag{4}
$$

Maximum Causal Entropy IRL

• The above optimization problem can be solved by the gradient descent update rule given by

$$\lambda^{+} \leftarrow \lambda - \eta g, \tag{5}$$

where η denotes the learning rate, and the gradient is given by

$$g = \sum_{s,a} \left\{ \rho^{\pi_{\lambda}^{L}}(s,a) - \rho^{\Xi}(s,a) \right\} \frac{\partial R_{\lambda}^{L}(s,a)}{\partial \lambda}.$$

• For any given λ, the corresponding policy π_{λ}^{L} can be efficiently computed via Soft-Value-Iteration procedure (see [4, Algorithm. 9.1]).

Maximum Causal Entropy IRL: Linear Reward

- Learner model with linear reward function $R^L_\lambda(s,a) = \lambda^\top \phi^L(s,a)$.
- Teacher with linear reward function $R^E(s,a) = \left(w^E\right)^\top \phi^L(s,a)$.
- Let $\mu^{\pi^L} = \sum_{s,a} \rho^{\pi^L}(s,a)\phi^L(s,a)$, and $\mu^\Xi = \sum_{s,a} \rho^\Xi(s,a)\phi^L(s,a)$.
- The corresponding primal problem (with feature expectation matching):

$$\begin{aligned} \underset{\pi^L(a|s)}{\text{maximize}} \quad & \sum_{\tau=0}^{\infty} \gamma^\tau H(a_\tau \mid a_{0:\tau-1}, s_{0:\tau}) \\ \text{subject to} \quad & \mu^{\pi^L} = \mu^\Xi \\ & \sum_a \pi^L(a \mid s) = 1,\ \pi^L(a \mid s) \geq 0, \end{aligned} \tag{6}$$

where H is the conditional entropy.

Maximum Causal Entropy IRL: Linear Reward

Algorithm 7 Batch MCE-IRL (Linear Reward)

Input: collection of demonstrations Ξ
Initialization: λ_1 and π_1^L
for $j = 1, 2, \ldots$ **do**
  $\lambda_{j+1} \leftarrow \lambda_j - \eta_j \sum_{s,a} \left(\mu^{\pi_j^L} - \mu^{\Xi} \right)$
  $\pi_{j+1}^L \leftarrow \text{Soft-Value-Iteration} \left(\mathcal{M} \backslash R^E, R_{\lambda_{j+1}}^L \right)$

Maximum Causal Entropy IRL: DNN Reward

- Consider a feature map Φ which takes $\phi^L(\cdot,\cdot) \in \mathbb{R}^{d'}$ as input, and is parameterized by the weights $W \in \mathbb{R}^{d_1}$ of a deep neural network, i.e., $\Phi\left(\phi^L(\cdot,\cdot);W\right) \in \mathbb{R}^{d_2}$.

- Given $\alpha \in \mathbb{R}^{d_2}$, denote $\lambda = (\alpha, W) \in \mathbb{R}^d$ with $d = d_1 + d_2$.

- Then for the learner model with reward function $R^L_\lambda(s,a) = \alpha^\top \Phi\left(\phi^L(s,a);W\right)$, we attempt to learn α, and W jointly.

Maximum Causal Entropy IRL: DNN Reward

Algorithm 8 Batch MCE-IRL (DNN Reward)

Input: collection of demonstrations Ξ
Initialization: λ_1 and π_1^L
for $j = 1, 2, \ldots$ **do**

$$\alpha_{j+1} \leftarrow \alpha_j - \eta_j \sum_{s,a} \left\{ \rho^{\pi_j^L}(s,a) - \rho^{\Xi}(s,a) \right\} \Phi\left(\phi^L(s,a); W_j\right)$$

$$W_{j+1} \leftarrow W_j - \eta_j \sum_{s,a} \left\{ \rho^{\pi_j^L}(s,a) - \rho^{\Xi}(s,a) \right\} \frac{\partial R_\lambda^L(s,a)}{\partial W}\Big|_{\alpha=\alpha_j, W=W_j}$$

$$\pi_{j+1}^L \leftarrow \text{Soft-Value-Iteration}\left(\mathcal{M} \backslash R^E, R^L_{\lambda_{j+1}}\right)$$

References

[1] Pieter Abbeel and Andrew Y Ng.
Apprenticeship learning via inverse reinforcement learning.
In *Proceedings of the twenty-first international conference on Machine learning*, page 1. ACM, 2004.

[2] Jonathan Ho and Stefano Ermon.
Generative adversarial imitation learning.
In *Advances in Neural Information Processing Systems*, pages 4565–4573, 2016.

[3] Andrew Y Ng, Stuart J Russell, et al.
Algorithms for inverse reinforcement learning.
In *Icml*, pages 663–670, 2000.

[4] Brian D Ziebart.
Modeling purposeful adaptive behavior with the principle of maximum causal entropy.
Carnegie Mellon University, 2010.

[5] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey.
Maximum entropy inverse reinforcement learning.
In *AAAI*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.