In-Trajectory Inverse Reinforcement Learning: Learn Incrementally Before An Ongoing Trajectory Terminates

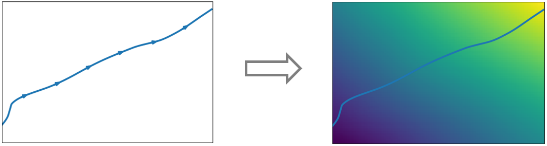
Shicheng Liu & Minghui Zhu

The Pennsylvania State University

Neural Information Processing Systems 2024

In-Trajectory Inverse Reinforcement Learning

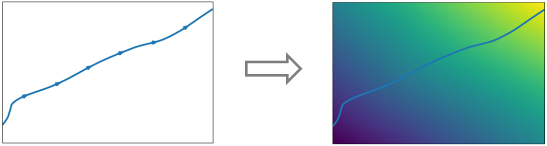
- Standard IRL: Learn a reward function from complete trajectories.



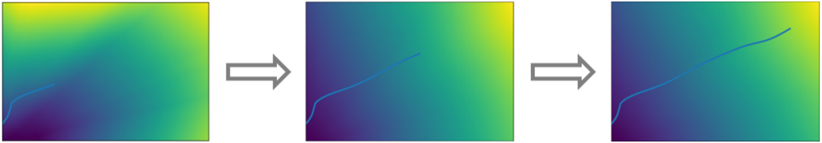
- Limitation: Has to wait until a complete trajectory is collected.

In-Trajectory Inverse Reinforcement Learning

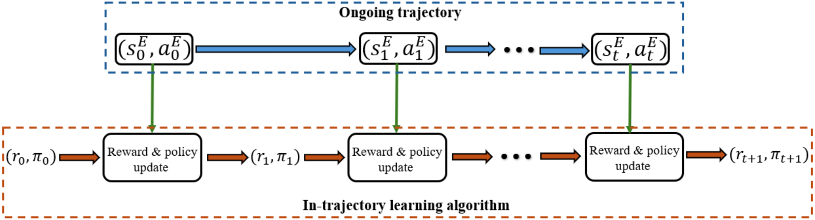
- Standard IRL: Learn a reward function from complete trajectories.



- Limitation: Has to wait until a complete trajectory is collected.
- In-trajectory IRL: Learn incrementally before the trajectory terminates.



Ongoing trajectory & In-trajectory learning



- At each time t, a new state-action pair (s_t, a_t) is observed.
- In-trajectory learning uses this new state-action pair (s_t, a_t) to update the last models (r_t, π_t) to (r_{t+1}, π_{t+1}).
    - A reward model and a policy model are available at any time within the ongoing trajectory.
    - The reward and policy models improve incrementally when more state-action pairs are revealed.

Online bi-level optimization formulation

Definition of local regret

Given a sequence of loss functions $\{f_t(x)\}_{t=0}^{T}$ with decision variable x and time index t, the local regret is $\sum_{t=0}^{T} ||\frac{1}{t+1}\sum_{i=0}^{t}\nabla f_i(x_t)||^2$.

- Loss function at time t:

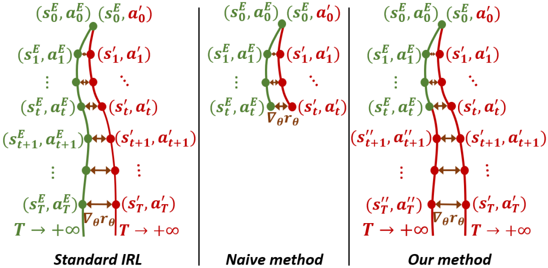
$$L_t(\theta;(S_t^E, A_t^E)) = -\log \pi_{r_\theta}(A_t^E|S_t^E) + \frac{\lambda}{2}||\theta - \bar{\theta}||^2,$$
$$\text{s.t.} \quad \pi_{r_\theta} = \arg\max_{\pi} J_{r_\theta}(\pi) + H(\pi).$$

Meta-regularized in-trajectory inverse reinforcement learning (MERIT-IRL)

$$E_{\{(S_t^E, A_t^E)\sim \mathbb{P}_t^{\pi^E}(\cdot,\cdot)\}_{t\geq 0}}\left[\sum_{t=0}^{T-1} ||\frac{1}{t+1}\sum_{i=0}^{t}\nabla L_i(\theta_t;(S_i^E, A_i^E))||^2\right], \quad \text{(upper level)}$$
$$\text{s.t.} \quad \pi_{r_{\theta_t}} = \arg\max_{\pi} J_{r_{\theta_t}}(\pi) + H(\pi). \quad \text{(lower level)}$$

Policy update & reward update

- Policy update: Solve the lower-level problem via one-step soft policy iteration.
  - Compute the soft Q-function $Q^{\text{soft}}_{\theta_t,\pi_t}$ under the current reward r_{θ_t} and policy π_t.
  - Update $\pi_{t+1}(a|s) \propto \exp(Q^{\text{soft}}_{\theta_t,\pi_t}(s,a))$.
- Reward update: Estimate hyper-gradient $g_t = \sum_{i=0}^{\infty} \gamma^i \nabla_\theta r_{\theta_t}(s'_i, a'_i) - \sum_{i=0}^{\infty} \gamma^i \nabla_\theta r_{\theta_t}(s''_i, a''_i) + \frac{\lambda(1-\gamma^{t+1})}{1-\gamma}(\theta_t - \bar{\theta})$ and update $\theta_{t+1} = \theta_t - \alpha_t g_t$.



Theoretical guarantee

Sub-linear local regret

Suppose $\alpha_t = \frac{(1-\gamma)(t+1)^{-1/2}}{\lambda}$, it holds that:

$$E_{\{(S_t^E, A_t^E) \sim P_t^{\pi_E}(\cdot,\cdot)\}_{t\geq 0}} \left[\sum_{t=0}^{T-1} || \frac{1}{t+1} \sum_{i=0}^{t} \nabla L_i(\theta_t; (S_i^E, A_i^E)) ||^2 \right]$$

$$\leq O(\log T + \sqrt{T} + \sqrt{T} \log T).$$

Theoretical guarantee

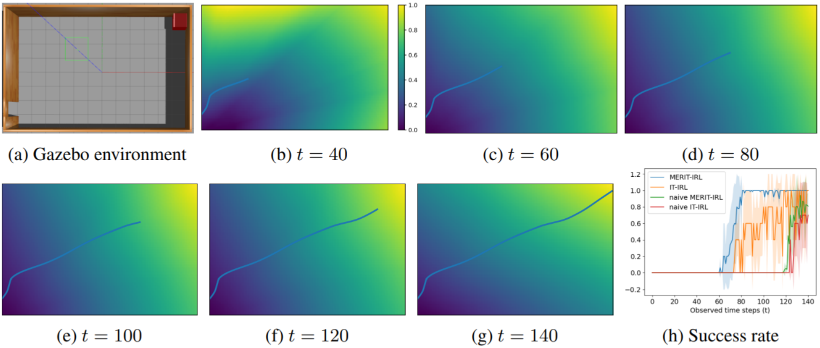
Sub-linear regret

Suppose the expert reward function r_E and the parameterized reward r_θ are linear and choose $\alpha_t = \frac{1-\gamma}{\lambda(t+1)(1-\gamma^{t+1})}$, we have that:

$$E_{\{(S_t^E,A_t^E)\sim P_t^{\pi_E}(\cdot,\cdot)\}_{t\geq 0}}\left[\sum_{t=0}^{T-1} L_t(\theta_t;(S_t^E,A_t^E))\right]$$

$$-\min_{\theta} E_{\{(S_t^E,A_t^E)\sim \mathbb{P}_t^{\pi_E}(\cdot,\cdot)\}_{t\geq 0}}\left[\sum_{t=0}^{T-1} L_t(\theta;(S_t^E,A_t^E))\right] \leq O(\log T).$$

Simulations



(a) Gazebo environment (b) $t = 40$ (c) $t = 60$ (d) $t = 80$

(e) $t = 100$ (f) $t = 120$ (g) $t = 140$ (h) Success rate

- MERIT-IRL can get (relatively) accurate reward and policy before the ongoing trajectory terminates.

Conclusion

- Learn incrementally before an ongoing trajectory terminates.
- Formulate as an online bi-level optimization problem.
- MERIT-IRL: Theoretical framework effective to ongoing trajectories.