Alberto Marinelli (638283)

Midterm 4 (9.)

Ng and Russell, Algorithms for Inverse Reinforcement Learning, ICML 2000

Inverse reinforcement learning (IRL) problem

Given:

1. Measurements of an agent's behaviour over time (and sensory inputs, if needed)
2. A model of the environment (if available)

Determine:

1. The reward function being optimized

IRL in Finite State Space

The IRL problem can be modelled using the **Markov Decision Processes** in the simplest case where the state space is finite, the model is known and the complete policy is observed.

Given:

1. Finite state space S
2. Set of k actions A = $\{a_1, \dots, a_k\}$
3. Transition probabilities $\{P_{sa}\}$
4. Discount factor Υ
5. Policy π

Determine:

1. Set of possible reward functions R such that π is an optimal policy in the MDP $(S, A, \{P_{sa}\}, \Upsilon, R)$

↓ More in depth, with a theorem is defined a **necessary and sufficient condition** for $\pi \equiv a_1$ to be the **unique optimal policy**

Theorem *Let a finite state space S, a set of actions $A = \{a_1, \dots, a_k\}$, transition probability matrices $\{\boldsymbol{P}_a\}$, and a discount factor $\gamma \in (0, 1)$ be given. Then the policy π given by $\pi(s) \equiv a_1$ is optimal if and only if, for all $a = a_2, \dots, a_k$, the reward $\boldsymbol{R}$ satisfies*

$$(\boldsymbol{P}_{a_1} - \boldsymbol{P}_a)(\boldsymbol{I} - \gamma \boldsymbol{P}_{a_1})^{-1} \boldsymbol{R} \succeq 0 \quad \text{(a)}$$

Algorithm

Problems in IRL in Finite State Space

1. **R = 0 is always solution**, if the reward is the same no matter what action is taken, then any policy is optimal
2. For most MDPs there are **many choices of R** that satisfy the necessary and sufficient condition *(a)*

Solutions

Choose R that makes π optimal and favor the solution that make any single step deviation from π as costly as possible.
In addition, to select the solutions with small rewards is necessary add to the objective funtion a weight decay defined as: $-\lambda \| R \|_1$, where λ is a penality coefficient.

Resulting optimization problem

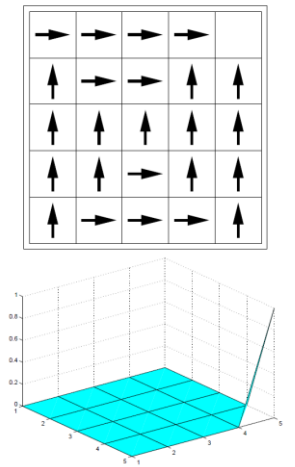
$$
\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{N} \min_{a \in \{a_2, \ldots, a_k\}} \{(\boldsymbol{P}_{a_1}(i) - \boldsymbol{P}_a(i)) \\
& \qquad (\boldsymbol{I} - \gamma \boldsymbol{P}_{a_1})^{-1} \boldsymbol{R}\} - \lambda ||\boldsymbol{R}||_1 \\
\text{s.t.} \quad & (\boldsymbol{P}_{a_1} - \boldsymbol{P}_a)(\boldsymbol{I} - \gamma \boldsymbol{P}_{a_1})^{-1} \boldsymbol{R} \succeq 0 \\
& \qquad \forall a \in A \setminus a_1 \\
& |\boldsymbol{R}_i| \leq R_{\max}, \quad i = 1, \ldots, N
\end{aligned}
$$

where $\boldsymbol{P}_a(i)$ denotes the ith row of $\boldsymbol{P}_a$.

Condition (a)

Penalty term

Experiment



Decription of the experiment

In a 5x5 grid, the agent starts from the lower-left grid square, and has to make its way to the upper-right grid square, whereupon it receives a reward of 1. The actions corresponds to the movement in four directions (that are noisy and have a 30% chance of resulting moving in a random direction instead).

True reward function

Figure 1. Top: 5x5 grid world with optimal policy. *Bottom:* True reward function.

Results

By running the algorithm it is possible to see how most of the original structure of the problem is recovered. A more precise result is obtained by using the penalty term (bottom of the figure)

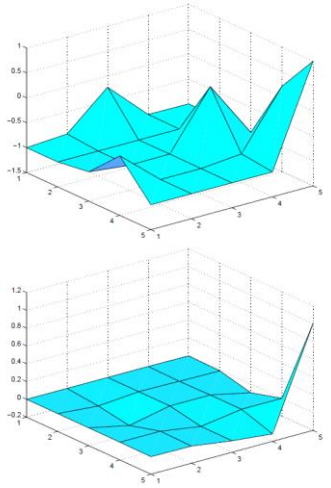


Figure 2. Inverse RL on the 5×5 grid. *Top:* $\lambda = 0$. *Bottom:* $\lambda = 1.05$.

IRL in Infinite State Spaces

The **IRL problem in infinite state space** can be defined as in the finite state space, with regard to the reward function R that is a function from $S = \mathbb{R}^n$ into the reals, and a general solution to IRL that would require working with this space of all functions $\mathbb{R}^n \rightarrow \mathbb{R}$

To work in this space is used a linear approximation of the reward function R :

$$R(s) = \alpha_1\phi_1(s) + \alpha_2\phi_2(s) + \cdots + \alpha_d\phi_d(s)$$

Parameters that we want to find

Basis functions that maps from S into $\mathbb{R}$

The condition *(a)* must be adapted to the above reward function. Considering that in this case, by the linearity expectations, the value function V_i^{π} of policy π is defined as: $V^{\pi} = a_1V_1^{\pi} + \cdots + a_dV_d^{\pi}$, and using the *Bellman Optimality*, the condition *(a)* become:

$$\mathrm{E}_{s' \sim P_{sa_1}}[V^{\pi}(s')] \geq \mathrm{E}_{s' \sim P_{sa}}[V^{\pi}(s')] \quad \text{(b)}$$

For all state s and all actions $a \in A \setminus a_1$

Algorithm

Problems in IRL in Infinite State Space

1. Is impossible to check all the **inifinitely constraints in *(b)***
2. Using the linear function approximator to express R, the algorithm may **not be able to express any reward function** for which π is optimal

Solutions

1. Sample a (large) finite subset of S_0 of the states, and apply this constraint only at those states $s \in S_0$
2. Relax the constraint of *(b)* paying a penality when they are violated

Resulting optimization problem

$$\begin{aligned}&\text{maximize } \textstyle\sum_{s\in S_0} \min_{a\in\{a_2,\ldots,a_k\}}\{\\ &\qquad p(\mathrm{E}_{s'\sim P_{sa_1}}[V^\pi(s')] - \mathrm{E}_{s'\sim P_{sa}}[V^\pi(s')])\}\\ &\text{s.t. } |\alpha_i| \le 1,\ \ i = 1,\ldots,d\end{aligned}$$

p is given by $p(x) = x$ if $x \geq 0$, $p(x) = 2x$ otherwise,
and penalizes violation of the constrains of (b)

Experiment

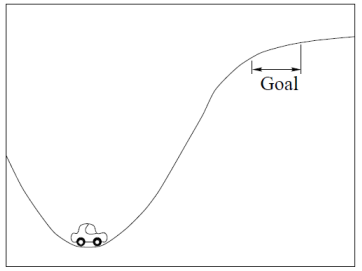


Figure 3. Cartoon of the mountain–car problem (not shown to scale).

Decription of the experiment
The experiment is the «mountain-car» task. The reward in this experiment is -1 per step until the goal of the top of the hill is reached, and the state is the car's x-position and velocity.

Results

1) The first run of this experiment is made by choosing the function approximator class for the reward to be functions of the car's x-position only, with the class consisting of all linear combinations of 26 evenly spaced Gaussian-shaped basis functions. The result is that the structure of the problem was well captured by the algorithm.

2) The second run is slightly different, the optimal policy is to go quickly to the bottom of the hill and park there. The result is that once again the algorithm has reconstructed the structure of the original problem adding an artifact on the right side due to the effect of unavoidability "shooting off" the right end.

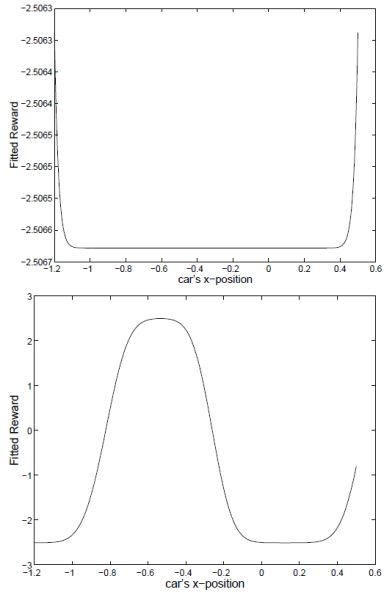


Figure 4. Typical solutions found by IRL for the mountain–car. *Top*: Original problem (note scale on y axis). *Bottom*: Problem of parking at bottom of hill.

IRL from Sampled Trajectories

This approach don't require an explicit Markov Decision Process. In this case is possible to use the policy π only through a set of actual trajectories in the state space.

The initial state distribution D is fixed, the **goal** is to find R such that the unknown policy π maximizes $E_{s0\sim D}[V^{\pi}(s_0)]$.

R is defined using a linear-approximator class.

For each policy π, the value of $V^{\pi}(s_0)$ for any setting of the $a_i s$ is estimated executing *m* Monte Carlo trajectories under π. Then, $\hat{V}_i^{\pi}(s_0)$ is defined as the average over the empirical returns of m such trajectories.

$$\hat{V}^{\pi}(s_0) = \alpha_1 \hat{V}_1^{\pi}(s_0) + \cdots + \alpha_d \hat{V}_d^{\pi}(s_0) \qquad \text{(c)}$$

Algorithm

Given a set of policies $\{\pi_1, \dots, \pi_k\}$, find a setting of the $a_i s$ that the resulting reward function satisfies:

$$V^{\pi^*}(s_0) \geq V^{\pi_i}(s_0), \quad i = 1, \dots, k$$

Resulting optimization problem

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{k} p\left(\hat{V}^{\pi^*}(s_0) - \hat{V}^{\pi_i}(s_0)\right) \\ \text{s.t.} \quad & |\alpha_i| \leq 1, \quad i = 1, \dots, d \end{aligned}$$

p is given by $p(x) = x$ if $x \geq 0$, $p(x) = 2x$ otherwise, and penalizes violation of the constrains of *(c)*

Finally, find a policy π_{k+1} that maximizes $V^{\pi}(s_0)$ under R, add π_{k+1} to the current set of policies, and continue.

Experiment

Decription of the experiment

Continuos version of the 5x5 grid world experiment in the IRL for the finite state space. The function approximator class consisted of all linear combinations of a 15x15 array of two dimensional Gaussian basis functions. The initial state distribution D was uniform over the state space.
The run of the algorithm is made on m = 5000 trajectories, each of 30 steps, to evaluate each policy.

Results

The results of the experimet are analyzed with a comparison between the **fitted reward's optimal policy** and the **true optimal policy**.
The figure on the top shows that there are few discrepancies with many distinct optimal policies.
Another aspect is that there is not statistically significant difference from a comparison of the **quality** between the fitted reward's optimal policy and the true optimal policy (figure on the bottom).

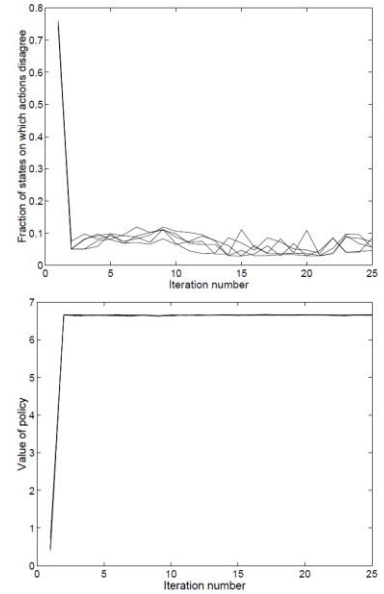


Figure 5. Results on the continuous grid world, for 5 runs. *Top*: Fraction of states on which the fitted reward's optimal policy disagrees with the true optimal policy, plotted against iteration number. *Bottom*: The value of the fitted reward's optimal policy. (Estimates are from 50000 Monte Carlo trials of length 50 each; negligible errorbars).

Conclusions

Inverse Reinforcement Learning is an experimental approach but the results of the experiments show that IRL could solve complex learning problems.

In this work, using three algorithms, the functioning of the IRL was shown in simple discrete / finite and continuous / infinite state problems and in particular how the problem of degeneracy is addressed (the existence of a large set of reward functions for which the observed policy is optimal).

Despite the good results obtained, there are many open questions on how IRL could perform in the case of real problems and address the noise of the data or the presence of many optimal policies of which only few are observable.

Thanks for attention!

Alberto Marinelli