Maximum Entropy Inverse Reinforcement Learning in Continuous State Spaces with Path Integrals

Navid Aghasadeghi AGHASAD1@ILLINOIS.EDU
University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA

Timothy W. Bretl TBRETL@ILLINOIS.EDU
University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA

1. Introduction

Inverse reinforcement learning (IRL) is the problem of recovering a cost function that is consistent with observations of optimal or "expert" trajectories and with a given dynamic model (Ng & Russell, 2000). In some cases, for example in the study of human motor control, it is precisely this cost function that we want to know. In other cases, imitating the behavior of an expert might be the goal. IRL problems are of interest in a wide range of applications, from basic science (Todorov, 2004; Kording & Wolpert, 2004) to optimal control of aircraft (Krstic & Tsiotras, 1999) and more recently aerobatic helicopter flight (Abbeel et al., 2010) within the robotics community.

In this paper, we consider the problem of inverse reinforcement learning (IRL) for a class of stochastic continuous state-space systems, under the assumption that the cost function is parametric with known basis functions. Our goal is to produce a cost function for which a set of trajectories, observed in experiment, is most likely. We proceed by enforcing a constraint on the relationship between input noise and input cost that produces a maximum entropy distribution over the space of all sample paths. We apply maximum likelihood (ML) estimation to approximate the parameters of this distribution (hence, of the cost function) given a finite set of sample paths. We iteratively improve our approximation by adding to this set the sample path that would be optimal given our current estimate of the cost function. Preliminary results in simulation provide empirical evidence that our algorithm converges.

2. System Model

We consider the following system:

$$\dot{x}_t = f(x_t) + G(x_t)\Big(u_t + \epsilon_t\Big), \tag{1}$$

with state $x_t \in \mathbb{R}^{n\times 1}$, passive dynamics $f(x_t) \in \mathbb{R}^{n\times 1}$, controls $u_t \in \mathbb{R}^{p\times 1}$ and control matrix $G(x_t) \in \mathbb{R}^{n\times p}$. Also, ϵ_t is a mean zero Gaussian with variance Σ_ϵ.

We define finite horizon trajectories starting at t_i and ending at t_f by $\tau_{t_i} = (x_t, t_i \le t \le t_f)$ and the cost by:

$$J(x, t_i, u_t) = \phi_{t_f} + \int_{t_i}^{t_f} \big(q_t + \frac{1}{2} u_t^T R u_t\big) dt. \tag{2}$$

where ϕ_{t_f} and q_t represent the terminal cost and the state dependent cost, and $\frac{1}{2}u_t^T R u_t$ for a positive semi-definite matrix R represent quadratic input cost.

3. Inverse Reinforcement Learning

The IRL problem addressed is recovering weights of a parameterized cost function, given dynamics of the system and a set of expert-demonstrated trajectories.

We consider a parameterized version of the cost function (2), parameterized with weights $\boldsymbol{\beta}^*$ and known basis functions $\tilde{\Phi}$, i.e. $J(x, t_i, u_t) = J(\tau_{t_i}) = {\boldsymbol{\beta}^*}^T \tilde{\Phi}$. Furthermore, we consider having a set of M expert-demonstrated trajectories, $\Omega^* = \{\tau_1, ..., \tau_M\}$, which are optimal with respect to the expert's cost function.

The inverse reinforcement learning problem is now written as a maximum likelihood problem in the following way:

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \prod_{i=1}^{M} P(\tau_i|\boldsymbol{\beta}), \tag{3}$$

4. Method of Approach

Our approach to this problem is based on the use of Path Integrals, as described in (Theodorou et al., 2010). Using this approach, we obtain a closed form probability distribution over the set of all trajectories which could result from an optimal policy, and we will use this distribution to address the IRL problem (3). The obtained distribution will be of the maximum entropy form, therefore, our proposed approach will share similarities with the maximum entropy approach introduced by (Ziebart et al., 2008), and used in (Boularias et al., 2011).

In this formulation, a critical assumption imposed on the structure of the input cost matrix is that $\lambda R^{-1} = \Sigma_\epsilon$, for a constant λ. This assumption imposes high costs on controls that are less noisy, and low costs on controls with high noise. This is reasonable because we need a significant control authority over more noisy controls, and vice versa. A similar framework has been considered in (Dvijotham & Todorov, 2010).

The following probability distribution is obtained:

$$P(\tau_m|\Omega) = \frac{e^{-\frac{1}{\lambda}S(\tau_m)}}{\sum_{k=1}^{K} e^{-\frac{1}{\lambda}S(\tau_k)}}, \tag{4}$$

where,

$$\begin{aligned} S(\tau) =& \Phi(x_N) + \sum_{j=1}^{N-1} q(x_j)\Delta t \\ &+ \frac{1}{2}\sum_{j=i}^{N-1} \left|\left| \frac{x_{j+1}^{(c)} - x_j^{(c)}}{\Delta t} - f_j^{(c)} \right|\right|^2_{H_j^{-1}} \Delta t, \\ H_j =& {G_j^{(c)}}^T R^{-1} G_j^{(c)}, \end{aligned}$$

where we define the notation $||v||^2_M = v^T M v$, and $f_i = f(x_i), G_i = G(x_i)$. Here $S(\tau)$ represents the cost of a path obtained from the path integral formulation. Also the superscripts (c) denotes the actuated component of the state (for more details see (Theodorou et al., 2010)).

In the above equations, we are considering a discretized version of the costs and trajectories, as discussed in further detail in (Theodorou et al., 2010). Furthermore, assume all trajectories start at time zero, and use τ_m to denote the m-th trajectory from the set of all trajectories $\Omega = \{\tau_1, ..., \tau_K\}$, which includes the expert-demonstrated trajectories as well, i.e. $\Omega^* \subset \Omega$. The parameterization of the cost function $J(\tau)$ leads to a parameterization of $S(\tau) = \boldsymbol{\beta}^T \Phi$ where:

$$\Phi = \begin{pmatrix} \sum_{i=0}^{N-1} \boldsymbol{\psi}_i \\ \frac{1}{2}\sum_{j=i}^{N-1} \left|\left| \frac{x_{j+1}^{(c)} - x_j^{(c)}}{\Delta t} - f_j^{(c)} \right|\right|^2_{\hat{H}_j^{-1}} \Delta t \\ \psi_N \end{pmatrix},$$

for known features $\boldsymbol{\psi}_i$ and ψ_N and known matrix $\hat{R}$ parameterizing the state cost, terminal cost and the input cost matrix, and for $\hat{H}_j = {G_j^{(c)}}^T \hat{R}^{-1} G_j^{(c)}$.

The IRL problem now reduces to solving for β in:

$$\begin{aligned} &\arg\max_\beta \prod_{i=1}^{M} P(\tau_i|\beta, \Omega) = \\ &\arg\max_\beta \prod_{i=1}^{M} \frac{e^{-\frac{1}{\lambda}(\beta^T \Phi(\tau_i))}}{\sum_{k=1}^{K} e^{-\frac{1}{\lambda}(\beta^T \Phi(\tau_k))}}. \end{aligned} \tag{5}$$

The solution to (5) involves two steps. First step is to solve the ML given Ω using the following iterative process, known as the *Iterative Scaling Algorithm* (Darroch & Ratcliff, 1972; Chen & Rosenfeld, 2002), for every coordinate n of the β vector:

$$\beta_{t+1,n} = \beta_{t,n} - ln \frac{\bar{\phi}_n}{E_{\boldsymbol{\beta}_t} \phi_n},$$

where,

$$\begin{aligned} \bar{\phi}_n &= \frac{1}{M}\sum_{i=1}^{M} \phi_n(\tau_i), \\ \phi_n(\tau) &= \sum_{i=1}^{N} \phi_n(x_i), \\ E_{\boldsymbol{\beta}_t} \phi_n &= \sum_{k=1}^{K} P(\tau_k|\beta_t)\phi_n(\tau_k). \end{aligned}$$

This step can be compared to IRL approach in (Kalakrishnan et al., 2010), as it produces an estimate of the cost function with a fixed set of sampled trajectories. The second step involves updating the set of sampled trajectories. Since the summation over a set of sampled trajectories is merely an approximation for the distribution, we iteratively update the set of sampled trajectories, by adding the trajectory which is optimal with respect to the current cost estimate to the set of all sampled trajectories. We obtain the following algorithm which resembles the algorithm in (Abbeel & Ng, 2004):

1. Solve the ML $\hat{\beta}_t = \arg\max_\beta \prod_{i=1}^{M} P(T = \tau_i|\beta, \Omega_t)$.
2. Solve optimal control $\hat{\tau}_{t+1} = \arg\min_\tau \hat{\beta}_t \phi(\tau)$ (Using (Theodorou et al., 2010)).

3. Add trajectory to the set of all trajectories: $\Omega_{t+1} = \Omega_t \cup \{\hat{\tau}_{t+1}\}$.

4. Let $\Delta_t = |\hat{\boldsymbol{\beta}}_t^T \phi(\tau^*) - \min_{\tau \in \Omega_t \setminus \Omega^*} \hat{\boldsymbol{\beta}}_t^T \phi(\tau)|$. If the change in improvement is smaller than a threshold, i.e. if $\left|\frac{\Delta_{t+1}-\Delta_t}{\Delta_{t+1}}\right| < \delta_{thr}$, then **terminate**, otherwise $t \leftarrow t+1$ and go back to step 1.

5. Evaluation

In order to evaluate our method, we performed a simulation of a 2-D point mass system. Furthermore, we considered a class of parameterized policies called Dynamic Movement Primitives (DMP), where the time-varying policy was parameterized by the parameter vector θ which scaled 10 time-varying basis functions (as discussed further in (Ijspeert et al., 2003; Theodorou et al., 2010)). Therefore, the optimal control solver found the optimal parameters θ with respect to the defined cost function, instead of solving for the optimal control at every time step. The dynamics of this system are shown below.

$$\ddot{x} = \frac{1}{m}(-b\dot{x} + u),$$
$$u = m\ddot{x}_d + b\dot{x} + k_P(x_d - x) + k_D(\dot{x}_d - \dot{x}),$$

where the parameters $x_d, \dot{x}_d$, and $\ddot{x}_d$ represent the desired output trajectory of the DMP. The DMP is a parameterized policy, which performs as a point attractor, and moves the system from some initial point, in this case the origin at $[0, 0]$, to some goal state, in our simulations the point $[1, 1]$. , The parameters θ of the DMP which determine the shape of the trajectory are optimized with respect to the following cost function, using the PI^2 approach described in (Theodorou et al., 2010).

The cost function defined was a mixture of Gaussian state costs, terminal state costs, and input costs. We generated 6 different features, each feature being a sum of many Gaussians with random means and covariance matrices. We denote the features by $\Phi_i, i = 1, ..., 6$. We then used a weighted sum of these features, with weights $\beta_i, i = 1, ..., 6$, as part of the cost function in our simulation. The boldface symbols $\boldsymbol{\Phi}$ and $\boldsymbol{\beta}$ denote the vector of these parameters. The resulting cost function can be seen in Fig. 1. Subsequently, added the terminal state cost and input costs to obtain the

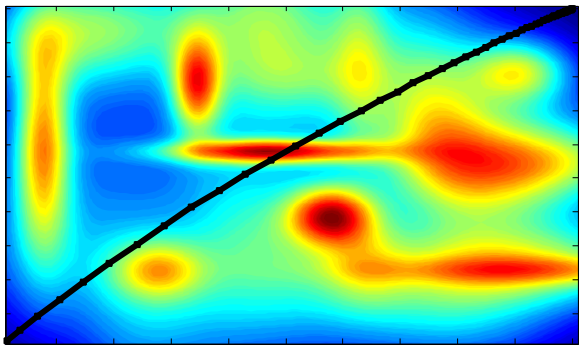

Figure 1. The true cost function, and the observed nominal trajectory.

following cost function:

$$\begin{aligned} S(\tau) =& \boldsymbol{\beta}^{\mathbf{T}} \boldsymbol{\Phi}(\boldsymbol{\tau}) \\ &+ C_1(x_{t_N} - [1,1]^T)^T(x_{t_N} - [1,1]^T) \\ &+ C_2(\dot{x}_{t_N}^T \dot{x}_{t_N}) \\ &+ \frac{C_3}{2} \sum_{j=i}^{N-1} \left|\left| \frac{x_{j+1}^{(c)} - x_j^{(c)}}{\Delta t} - f_j^{(c)} \right|\right|_{\hat{H}_j^{-1}}^2 \end{aligned}$$

where the first term $\boldsymbol{\beta}^{\mathbf{T}} \boldsymbol{\Phi}(\boldsymbol{\tau})$ reflects the cost of the trajectory due to the sum of Gaussian features. The second and third term enforce some cost for not being at the goals state at time t_N and having a non-zero velocity at that time. Lastly, the fourth term enforces a cost on the inputs. Note that in this simulation the constants C_1, C_2 and C_3 are known, and we will only be estimating the weights $\boldsymbol{\beta}$.

Using the discussed dynamic equations and cost function, we then found a single optimal trajectory, which is shown in Fig. 1. To do so, we also utilized the code provided in *"http://www-clmc.usc.edu/Resources/Software"*. Moreover, we sampled 50 trajectories, in order to construct a probability distribution over all trajectories. These trajectories were sampled by running 50 trajectory rollouts around the expert-demonstrated trajectory. (One can use techniques in (Ijspeert et al., 2003) to obtain a policy parameterized by θ describing the expert-demonstrated trajectory, and subsequently use the parameter $\theta^* + \epsilon_t$ to generate sampled trajectories.)

We applied our proposed algorithm to recover the cost function using the nominal demonstrated trajectory and the 50 sampled trajectories. Fig. 2 demonstrates the recovered cost function, and the best candidate trajectory with respect to the final estimated cost function. The iterative improvement in estimation is demonstrated in Fig. 3, which plots the excess cost

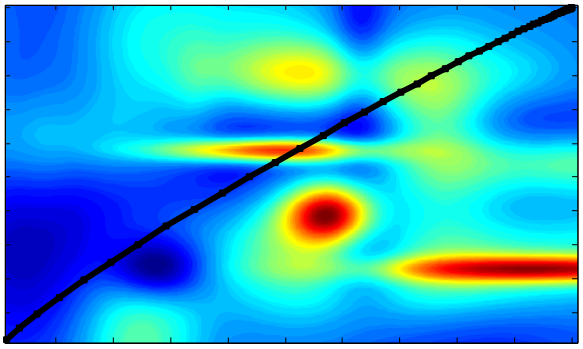

Figure 2. The recovered cost function and the best candidate sample trajectory $\hat{\tau}^*_{t_{end}}$.

of the best candidate trajectory versus the number of iterations. The set of all 50 sampled trajectories are included in Fig 4.

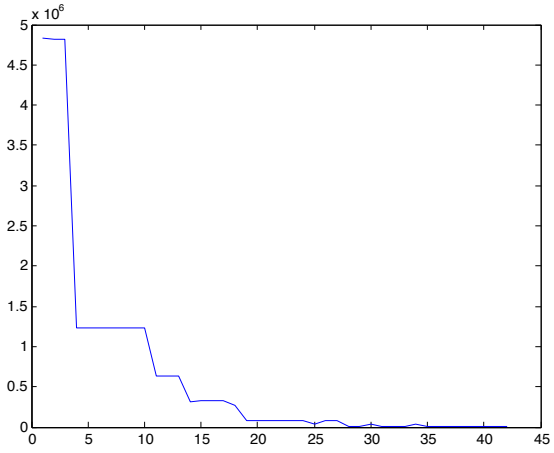

Figure 3. Plot of $\Delta_t^G = \boldsymbol{\beta}^{*T}\phi(\hat{\tau}_t^*) - \boldsymbol{\beta}^{*T}\phi(\tau^*)$ versus the number of iterations t, where $\hat{\tau}_t^* = \arg\min_{\tau\in\Omega_t\setminus\{\tau^*\}} \hat{\boldsymbol{\beta}}_t^T\phi(\tau)$.

6. Discussion and Future Work

We proposed an algorithm for inverse reinforcement learning in a framework where the cost function was a weighted linear combination of some known basis functions, and where the input cost was inversely proportional to the noise variance. We have shown using simulations that this approach improves the estimates of the cost function iteratively. These results should be considered preliminary. A formal comparison between our approach and existing IRL approaches in literature, and a rigorous evaluation of the performance of the algorithm are topics of future work.

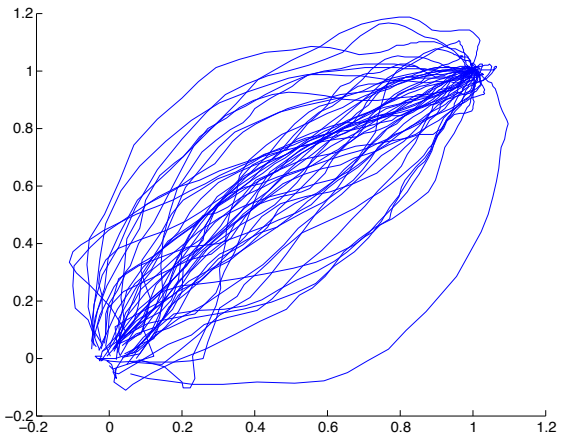

Figure 4. All sampled trajectories.

References

Abbeel, P. and Ng, A.Y. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 1. ACM, 2004.

Abbeel, P., Coates, A., and Ng, A.Y. Autonomous helicopter aerobatics through apprenticeship learning. *International Journal of Robotics Research*, 29(13): 1608–1639, 2010. ISSN 0278-3649.

Boularias, A., Kober, J., and Peters, J. Relative entropy inverse reinforcement learning. *ICAPS*, 15: 20–27, 2011.

Chen, S.F. and Rosenfeld, R. A survey of smoothing techniques for ME models. *Speech and Audio Processing, IEEE Transactions on*, 8(1):37–50, 2002. ISSN 1063-6676.

Darroch, J.N. and Ratcliff, D. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):1470–1480, 1972. ISSN 0003-4851.

Dvijotham, K. and Todorov, E. Inverse Optimal Control with Linearly-Solvable MDPs. In *Proceedings of the Interntional Conference on Machine Learning*. Citeseer, 2010.

Ijspeert, A.J., Nakanishi, J., and Schaal, S. Learning attractor landscapes for learning motor primitives. *Advances in neural information processing systems*, pp. 1547–1554, 2003. ISSN 1049-5258.

Kalakrishnan, M., Theodorou, E., and Schaal, S. Inverse Reinforcement Learning with PI 2. In *The Snowbird Workshop, submitted to*, 2010.

Kording, K.P. and Wolpert, D.M. The loss function of sensorimotor learning. *Proceedings of the National Academy of Sciences of the United States of America*, 101(26):9839, 2004.

Krstic, M. and Tsiotras, P. Inverse optimal stabilization of a rigid spacecraft. *Automatic Control, IEEE Transactions on*, 44(5):1042–1049, 1999. ISSN 0018-9286.

Ng, A.Y. and Russell, S. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 663–670, 2000.

Theodorou, E.A., Buchli, J., and Schaal, S. A Generalized Path Integral Control Approach to Reinforcement Learning. *Journal of Machine Learning Research*, 11:3137–3181, 2010.

Todorov, E. Optimality principles in sensorimotor control. *Nature neuroscience*, 7(9):907–915, 2004.

Ziebart, B.D., Maas, A., Bagnell, J.A., and Dey, A.K. Maximum entropy inverse reinforcement learning. In *Proc. AAAI*, pp. 1433–1438, 2008.