2011 American Control Conference
on O'Farrell Street, San Francisco, CA, USA
June 29 - July 01, 2011

# Inverse Reinforcement Learning with Gaussian Process

Qifeng Qiao and Peter A. Beling
Department of Systems and Information Engineering
University of Virginia
Charlottesville, Virginia 22904
Email: qq2r, pb3a@virginia.edu

*Abstract*— **We present new algorithms for inverse reinforcement learning (IRL, or inverse optimal control) in convex optimization settings. We argue that finite-space IRL can be posed as a convex quadratic program under a Bayesian inference framework with the objective of maximum a posteriori estimation. To deal with problems in large or even infinite state space, we propose a Gaussian process model and use preference graphs to represent observations of decision trajectories. Our method is distinguished from other approaches to IRL in that it makes no assumptions about the form of the reward function and yet it retains the promise of computationally manageable implementations for potential real-world applications. In comparison with an establish algorithm on small-scale numerical problems, our method demonstrated better accuracy in apprenticeship learning and a more robust dependence on the number of observations.**

## I. Introduction

Imitation learning is a subfield of machine learning in which the objective is to learn to mimic human behavior solely through observation of the actions taken by the subject. Technical approaches to imitation learning generally fall into two broad categories [1]. One category contains behavioral cloning approaches that attempt to use supervised learning to predict actions directly from observations of features of the environment. The other category consists of IRL approaches, first introduced in [2], use training examples in the form of decision trajectories defined in terms of a Markov decision process (MDP) model of the underlying sequential decision task. IRL algorithms attempt to discover the reward function for the MDP solely on the basis of observations of a decision-maker's solution to that problem. This approach is appealing because knowledge of the reward function offers the promise that behavior can be predicted in domains unseen during the period of observation.

A variety of approaches have been proposed for IRL. In early work, Ng and Russel [2] advance the key idea of choosing the reward function to maximize the difference between the optimal and suboptimal policies, under the assumption that the reward function can be approximated by a linear combination of basis functions. A principal motivation for considering IRL problems is the idea of apprenticeship learning, in which observations of state-action pairs are used to learn the policies followed by experts for the purpose of mimicking or cloning behavior. By its nature, apprenticeship learning problems arise in situations where it is not possible or desirable to observe all state-action pairs for the decision maker's policy. In recent approaches to apprenticeship learning, partial policy observation is dealt with by searching mixed solutions in a space of learned policies with the goal that the accumulative feature expectation is near that of the expert [3], [4]. In such approaches, the reward function is approximated by a linear combination of features, which in turn allows for linear approximation of value functions with consequent simplification of the learning problem. In such methods, algorithm performance is strongly influenced by the modeler's choice of features. Another algorithm for IRL is policy matching in which the loss function penalizing deviations from expert's policy is minimized by tuning the parameters of reward functions [5].

The assumption that the reward function can be linearly approximated, which underlies a number of IRL approaches, may not be reasonable for many problems of practical interest. The ill-posed nature of the inverse learning problem also presents difficulties. Multiple reward functions may yield the same optimal policy, and there may be multiple observations at a state given the true reward function. To deal with these problems, we design algorithms that do not assume linear structure for reward function, but yet remain computationally efficient. In particular, we propose new IRL models and algorithms that assign a Gaussian prior on the reward function or treat the reward function as a Gaussian process. This approach is similar in perspective to that Ramachandran and Eyal [6], who view the state-action samples from the expert as the evidence that will be used to update a prior on the reward function, under a Bayesian framework. Other approaches to IRL include game-theoretic methods [7] and algorithms derived from linearly-solvable stochastic optimal control [8].

The main contributions of our work are as follows. First, we model the reward function in a finite state space using a Bayesian framework with known Gaussian priors. We show that this problem is a convex quadratic program, and hence that it can be efficiently solved. Second, for the general case that allows noisy observation of incomplete policies, representation of the reward function is challenging and requires more computation. We show that Gaussian process is appropriate in that case. Our model constructs a preference graph in action space to represent the multiple observations at a state. Even in cases where the state space is much larger than the number of observations, IRL via Gaussian processes has the promise of offering robust predictions and results that

978-1-4577-0079-8/11/$26.00 ©2011 AACC

are relatively insensitive to number of observations.

It is worth mentioning here that the preference graph we use in IRL is based on an understanding of the agent's preferences over action space. In the machine learning literature, there has been study of a learning scenario called learning label preference that focuses on finding the latent function that predicts preference relations among a finite set of labels. This scenario is a generalization of some standard problems, such as classification and label ranking [9]. Considering the latent function values as a Gaussian process, Chu and Ghahramani [10] observed that Bayesian framework is an efficient and competitive method for learning label preferences, and they proposed a novel likelihood function to capture preference relations and the use of a Gaussian process model for learning label preferences. We also use Bayesian inference and build off several of the ideas in [10] and related work, but our method differs from label preference learning for classification and label ranking. Our input data depends on states and actions in the context of an MDP. Moreover, we are learning the reward that indirectly determines how actions are chosen during the sequential evolution of an MDP, while preference learning studies the latent functions preserving preferences.

The rest of this paper is organized as follows: In Section II, we introduce IRL preliminaries. In Sections III and IV, we propose our principal models and algorithms. In Section V, we describe the results of two small-scale numerical experiments. Finally, in Section VI, we offer some concluding remarks.

## II. Preliminaries

A finite-state, infinite horizon *Markov decision process (MDP)* is defined as a tuple $\boldsymbol{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$, where $\mathcal{S} = \{s_1, s_2, \cdots, s_n\}$ is a set of $n$ states; $\mathcal{A} = \{a_1, a_2, \cdots, a_m\}$ is a set of $m$ actions; $\mathcal{P} = \left\{P_{a_j}\right\}_{j=1}^{m}$ is a set of state transition probabilities; $\gamma$ is a discount factor; and $r$ is the reward function which can be written as $r(s, a)$, if we define it as depending on state $s$ and action $a$. For any $a \in A$ and $P_a$ is a $n \times n$ matrix, each row of which, denoted as $P_{as}$, is the transition probabilities upon taking action $a$ in state $s$.

Consider a decision maker who selects actions according to a *policy* $\pi : S \to A$ that maps states to actions. Define the *value function* at state $s$ with respect to policy $\pi$ to be $V^{\pi}(s) = E[\sum_{t=0}^{\infty} \gamma^t r(s^t, \pi(s^t))|\pi]$, where the expectation is over the distribution of the state sequence $\left\{s^0, s^1, \dots\right\}$ given policy $\pi$, where superscripts index time. A decision maker who aims to maximize expected reward will, at every state $s$, choose the action that maximizes $V^{\pi}(s)$. Similarly, define the $Q$-factor for state $s$ and action $a$ under policy $\pi$, $Q^{\pi}(s, a)$, to be the expected return from state $s$, taking action $a$ and thereafter following policy $\pi$. Given a policy $\pi$, $\forall s \in S, a \in \mathcal{A}$, $V^{\pi}(s)$ and $Q^{\pi}(s, a)$ satisfy

$$\begin{aligned} V^{\pi}(s) &= r(s, \pi(s)) + \gamma \sum_{s'} P_{\pi(s)s}(s') V^{\pi}(s') \\ Q^{\pi}(s, a) &= r(s, a) + \gamma \sum_{s'} P_{as}(s') V^{\pi}(s') \end{aligned}$$

The well-known Bellman optimality conditions state that $\pi$ is optimal if and only if, $\forall s \in S$, we have $\pi(s) \in \arg\max_{a \in \mathcal{A}} Q^{\pi}(s, a)$ [11].

Given an MDP $M = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$, let us define the *inverse Markov decision process (IMDP)* $M_I = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{O})$. The process $M_I$ includes the states, actions, and dynamics of $M$, but lacks a specification of the reward vector, $r$. By way of compensation, $M_I$ includes a set of observations $\mathcal{O}$ that consists of state-action pairs generated through the observation of a decision maker. We can define the *inverse reinforcement learning* (IRL) problem associated with $M_I = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{O})$ to be that of finding the reward function $r$ such that the observations $\mathcal{O}$ could have come from an optimal policy for $M = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$. The IRL problem is, in general, highly underspecified, which has led researchers to consider various models for restricting the set of reward vectors under consideration. In a seminal consideration of IMDPs and associated IRL problems, Ng and Russel [2] observe that, by the optimality equations, the only reward vectors consistent with an optimal policy $\pi$ are those that satisfy the set of inequalities

$$(P_{\pi} - P_a)(I_n - \gamma P_{\pi})^{-1}\mathbf{r} \quad \geq \quad \mathbf{0}, \quad \forall a \in \mathcal{A}, \qquad (1)$$

where $P_{\pi}$ is the transition probability matrix relating to observed policy $\pi$, $P_a$ denotes the transition probability matrix for other actions, $I_n$ is a $n \times n$ identity matrix, and $\mathbf{r}$ is a reward vector that depends only on state. Note that the trivial solution $\mathbf{r} = 0$ satisfies these constraints, which highlights the underspecified nature of the problem and the need for reward selection mechanisms. Ng and Russel [2] choose the reward function to maximize the difference between the optimal and suboptimal policies, which can be done using a linear programming formulation. In the sections that follow, we propose the idea of selecting reward on the basis of Maximum a posterior (MAP) estimation in a Bayesian framework.

## III. Bayesian IRL with Gaussian Distribution

Suppose that we have a prior distribution $p(r)$ for the rewards in an IMDP $M_I$, along with a likelihood function $p(\mathcal{O}|r)$. Then we can define the associated Bayesian IRL problem to be that of finding the MAP estimate of $r$. In this section we consider this problem for priors with a Gaussian distribution, showing that the MAP estimation problem can be formulated as a convex optimization problem. We assume all the states, value functions, and transition probabilities can be stored in the memory of a computer.

Specifically, let $\mathbf{r} \in \Re^n$ be a random vector only depending on state. The entry $\mathbf{r}(s_i)$ denotes the reward at i-th state. We assign a Gaussian prior on the $\mathbf{r}$: $\mathbf{r} \sim \mathcal{N}(\mu_r, \Sigma_r)$. This is a subjective distribution; before anything is known about optimal policies for the MDP, the learner has characterized a prior belief by $\mu_r$ with confidence by $\Sigma_r$.

One can envision two principal types of experiments for collecting a set of observations $\mathcal{O}$:

1) *Decision Mapping*: the observations are obtained by finding a mapping between state and action; e.g., we

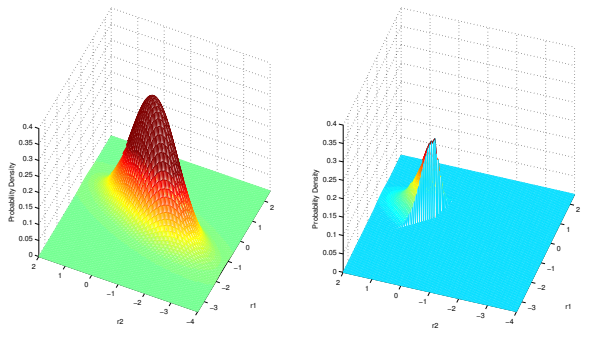


Fig. 1. An example showing the Bayesian IRL given full observation of the decision maker's policy.

ask the expert which action he, she, or it would choose at state $s$, and then repeat the process. Ultimately, we will have a set of independent state-action pairs, $\mathcal{O}_1 = \left\{(s^h, a^h)\right\}_{h=1}^{t}$.

2) *Decision Trajectory*: Given an initial state, we simulate the decision problem and record the history of the expert's behavior, $\mathcal{O}_2 = \left\{s^1, a^1, s^2, a^2, \cdots, s^t, a^t\right\}$.

Formally, we define an experiment $E$ to be a triple $(O, \mathbf{r}, \{p(\mathcal{O}|\mathbf{r})\})$, where $O$ is a random vector with probability mass function $p(\mathcal{O}|\mathbf{r})$ for some $\mathbf{r}$ in the function space. Given what experiment E was performed and a particular observation of $\mathcal{O}$, the experimenter is able to make inference and draw some evidence about $\mathbf{r}$ arising from $E$ and $\mathcal{O}$. This evidence we denote by $Ev(E, \mathcal{O})$. Consider observations made using decision mapping $\mathcal{O}_1$ and decision trajectory $\mathcal{O}_2$, with corresponding experiments $E_1 = (O_1, \mathbf{r}, \{p(\mathcal{O}_1|\mathbf{r})\})$ and $E_2 = (O_2, \mathbf{r}, \{p(\mathcal{O}_2|\mathbf{r})\})$. We would like to show that $Ev(E_1, \mathcal{O}_1) = Ev(E_2, \mathcal{O}_2)$, if the states in $\mathcal{O}_1$ and $\mathcal{O}_2$ are the same. This fact implies that inference conclusions drawn from $\mathcal{O}_1$ and $\mathcal{O}_2$ should be identical.

Making use of independence of state-action pairs in decision mapping, we calculate the joint probability density as

$$p(\mathcal{O}_1|\mathbf{r}) = \prod_{h=1}^{t} p(s^h, a^h|\mathbf{r}) = \prod_{h=1}^{t} p(s^h)p(a^h|s^h, \mathbf{r}).$$

Considering Markov transition in decision trajectory, we write the joint probability density as

$$p(\mathcal{O}_2|\mathbf{r}) = p(s^1)p(a^1|s^1, \mathbf{r}) \prod_{h=2}^{t} p(s^h|s^{h-1}, a^{h-1})p(a^h|s^h, \mathbf{r}).$$

Finally, we get $p(\mathcal{O}_1|\mathbf{r}) = c(\mathcal{O}_1, \mathcal{O}_2)p(\mathcal{O}_2|\mathbf{r})$, where $c(\mathcal{O}_1, \mathcal{O}_2)$ is a constant. The above equation implies an equivalence of evidence for inference of $\mathbf{r}$ between the use of a decision map or a decision trajectory.

To simplify computation, we eliminate the elements in likelihood function $p(\mathcal{O}|\mathbf{r})$ that do not contain $\mathbf{r}$, which yields $p(\mathcal{O}|\mathbf{r}) = \prod_{h=1}^{t} p(a^h|s^h, \mathbf{r})$. Further, we model $p(a^h|s^h, \mathbf{r})$ by

$$p(a^h|s^h, \mathbf{r}) = \begin{cases} 1, \text{if } Q(s^h, a^h) \geq Q(s^h, a), \ \forall a \in \mathcal{A} \\ 0, \text{otherwise.} \end{cases} \quad (2)$$

This form for the likelihood function is based on the assumption each observed action is an optimal choice on the part of the expert. Note that the set of reward values that make $p(a^h|s^h, \mathbf{r})$ equal to one is given by Eq. 1.

*Proposition 1:* Assume a countable state and control space and a stationary policy. Then IRL using Bayesian MAP inference is a quadratic convex programming problem.

*Proof:* By Bayes rule, the posterior distribution of reward

$$p(\mathbf{r}|\mathcal{O}) = \frac{1}{(2\pi)^{n/2}|\Sigma_r|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{r} - \mu_r)^T \Sigma_r^{-1}(\mathbf{r} - \mu_r)\right).$$

This posterior probability $p(\mathbf{r}|\mathcal{O})$ quantifies the evidence that $\mathbf{r}$ is the reward for the observations in $\mathcal{O}$. Using Eq. 1, we formulate the IRL problem as

$$\begin{aligned} \min_r \ & \frac{1}{2}(\mathbf{r} - \mu_r)^T \Sigma_r^{-1}(\mathbf{r} - \mu_r) \\ \text{s.t.} \ & (P_{a^*} - P_a)(I_n - \gamma P_{a^*})^{-1}\mathbf{r} > 0, \ \forall a \in \mathcal{A} \\ & \mathbf{r}_{\min} < \mathbf{r} < \mathbf{r}_{\max} \end{aligned} \quad (3)$$

Since the objective is convex quadratic and constraints are affine, Problem 3 is a convex quadratic program. ■

Fig. 1 shows a Gaussian prior on reward and its posterior after truncation by accounting for the linear constraints on reward implied by observation $\mathcal{O}$. Note the shift in mode.

The development above assumes the availability of a complete set of observations, giving the optimal action at every state. If necessary, it may be possible to expand observations of partial policies to fit the framework. A naive approach would be to state transition probabilities averaged over all possible actions at unobserved states.

## IV. Gaussian Processes for Generalized IRL

In this section, we introduce a Gaussian process IRL model. Our model involves the construction of a preference graph, defined below, that is used to record the actions of the expert under observation. The choice of one action over the others at any given state will be governed by Q-function values, if the expert acts optimally. Hence, these values may be used to define preference relations among actions.

*Definition 1* At state $s_i \in \mathcal{S}$, $\forall \hat{a}, \check{a} \in \mathcal{A}$, we define the *preference relation* as: if $Q(s_i, \hat{a}) \geq Q(s_i, \check{a})$, the action $\hat{a}$ is weakly preferred to $\check{a}$, denoted as $\hat{a} \succeq_{s_i} \check{a}$; strictly preferred, denoted as $\hat{a} \succ_{s_i} \check{a}$,if and only if $Q(s_i, \hat{a}) > Q(s_i, \check{a})$; $\hat{a}$ is equivalent to $\check{a}$, denoted as $\hat{a} \sim_{s_i} \check{a}$, if and only if $\hat{a} \succeq_{s_i} \check{a}$ and $\check{a} \succeq_{s_i} \hat{a}$.

*Definition 2* A *preference graph* over action space is a directed graph showing preference relations among the countable actions at a given state. At state $s_i$, a preference graph $\epsilon_i$ consists of the node set $\mathcal{V}_i$ and edge set $\mathcal{E}_i$. Each node represents an action in $\mathcal{A}$. Define a one-to-one mapping $\varphi : \mathcal{V}_i \to \mathcal{A}$. Each edge indicates the preference relation between two nodes.

Suppose we are given a dataset of observations, denoted as $\mathcal{O} = \{\mathcal{S}, \mathcal{G}\} = \{s_i, \epsilon_i\}_{i=1}^{\hat{n}}$. Each pair $(s_i, \epsilon_i)$ consists of two components: one is the input $s_i$ that is a feature vector constructed by a mapping $\phi : \mathcal{S} \to [0, 1]^d$; the other,

denoted as $\epsilon_i = (\mathcal{V}_i, \mathcal{E}_i)$, is a two layer preference graph over actions observed at $s_i$. As shown in Figure 2, the node set $\mathcal{V}_i$ can be divided into two subsets: a set of nodes in the top layer to represent optimal actions, denoted as $\mathcal{V}_i^+$; a set of nodes in the bottom layer to represent other actions, denoted as $\mathcal{V}_i^-$. The graph $\epsilon_i = \left\{(u \to v)_{l=1}^{n_i}, u \in \mathcal{V}_i^+,\ v \in \mathcal{V}_i^-\right\} \cup \left\{(u \leftrightarrow v)_{k=1}^{m_i},\ u, v \in \mathcal{V}_i^+\right\}$, where $n_i$ is the number of edges denoting strict preference relations and $m_i$ is the number of edges denoting equivalent relations. Consider action's


(a) (b)

Fig. 2. Examples of preference graph

influence on the reward function. Here we define $\mathbf{r}$ as follows.

$$\begin{aligned}\mathbf{r} &= (\underbrace{\mathbf{r}_{a_1}(s_1), ..., \mathbf{r}_{a_1}(s_{\hat{n}})}, \ldots, \underbrace{\mathbf{r}_{a_m}(s_1), \ldots, \mathbf{r}_{a_m}(s_{\hat{n}})}) \\ &= (\quad \mathbf{r}_{a_1}, \quad \cdots, \quad \mathbf{r}_{a_m}) \end{aligned} \tag{4}$$

where $\mathbf{r}_{a_j}, \forall j \in \{1, 2, \cdots, m\}$, denotes the reward only associated with j-th action. Given $\mathbf{r}$, a ranking function can be naturally formulated as arrangement of the nodes in sorting of the values of Q-functions. We write the ranking function with respect to a node $u$ at state $s$ as $Q(s, \varphi(u))$.

### A. Bayesian inference

Below we describe our models for prior information, likelihood functions, and inference.

*1) Gaussian prior:* Consider $\mathbf{r}_{a_j}$ as a stochastic process. Then $\mathbf{r}_{a_j}$ is a Gaussian process if, for any $\{s_1, \cdots, s_{\hat{n}}\} \in \mathcal{S}$, the random variables $\left\{\mathbf{r}_{a_j}(s_1), \cdots, \mathbf{r}_{a_j}(s_{\hat{n}})\right\}$ are normally distributed. We denote by $k_{a_j}(s_c, s_d)$ the function generating the value of entry $(c, d)$ for covariance matrix $K_{a_j}$, which leads to $\mathbf{r}_{a_j} \sim N(0, K_{a_j})$. Then the joint prior probability of the reward is a product of multivariate Gaussian, namely $p(\mathbf{r}|\mathcal{S}) = \prod_{j=1}^{m} p(\mathbf{r}_{a_j}|\mathcal{S})$ and $\mathbf{r} \sim N(0, K)$. Thus $\mathbf{r}$ is completely specified by the positive definite covariance matrix $K$. As we assume the $m$ latent processes are uncorrelated, the covariance matrix $K$ is block diagonal in the covariance matrices $\{K_1, ..., K_m\}$. In practice, we use a squared exponential kernel function, written as $k_{a_j}(s_c, s_d) = e^{\frac{1}{2}(s_c - s_d)^T M_a (s_c - s_d)} + \sigma_{a_j}^2 \delta(s_c, s_d)$ where $M_{a_j} = \kappa_{a_j} I_{\hat{n}}$ and $I_{\hat{n}}$ is an identity matrix of size $\hat{n}$. The function $\delta(.)$ is the Kronecker delta.

*2) Likelihood:* Given an edge $u \to v$, we adopt a variant of the likelihood function proposed by Chu and Ghahramani in [10] to capture the preference relation in that edge. Specifically,

$$\begin{aligned} &p_{\text{ideal}}(u \to v|\mathbf{r}_{\varphi(u)}(s), \mathbf{r}_{\varphi(v)}(s)) \\ &= \begin{cases} 1 \quad if\ Q(s, \varphi(u)) > Q(s, \varphi(v)) \\ 0 \quad \text{otherwise,} \end{cases} \end{aligned} \tag{5}$$

where $u$ and $v$ are two nodes in the preference graph. By Definition 2, these nodes can be mapped to two actions $\varphi(u)$ and $\varphi(v)$ in space $\mathcal{A}$. We write the Q-function as,

$$Q(s, a) = \mathbf{r}_a(s) + \gamma \hat{P}_{as} (I_{\hat{n}} - \gamma \hat{P}_{a^*})^{-1} \hat{I} \mathbf{r} \tag{6}$$

where $\hat{P}_{as}$ and $\hat{P}_{a^*}$ are transition probabilities for the observed $\hat{n}$ states, and $\hat{I}$ is a matrix with $\hat{n}$ rows and $\hat{n} \times m$ columns. The production of $\hat{I}$ and $\mathbf{r}$ is a $\hat{n} \times 1$ vector containing the reward for taking the optimal action at each state. After assuming that the latent functions are contaminated with Gaussian noise that has zero mean and unknown variance $\sigma^2$ [10], the likelihood function for $l$-th strict preference edge in graph $\epsilon_i$ becomes

$$\begin{aligned} &p(u_l \to v_l)|\mathbf{r}_{\varphi(u_l)}(s_i) + \delta_{u_l}, \mathbf{r}_{\varphi(v_l)}(s_i) + \delta_{v_l}) \\ &= \int\int p_{\text{ideal}}(\varphi(u_l) \succ \varphi(v_l)|\mathbf{r}_{\varphi(u_l)}(s_i), \mathbf{r}_{\varphi(v_l)}(s_i)) \\ &\mathbf{N}(\delta_u, 0, \sigma^2)\mathbf{N}(\delta_v, 0, \sigma^2) d\delta_u d\delta_v = \Phi(z_i^l) \end{aligned} \tag{7}$$

where $z_i^l = \frac{Q(s_i, \varphi(u_l)) - Q(s_i, \varphi(v_l))}{\sqrt{2}\sigma}$, $\mathbf{N}(\delta_u, 0, \sigma^2)$ denotes a Gaussian distribution for $\delta_u$, and $\Phi(z) = \int_{-\infty}^{z} \mathbf{N}(\gamma, 0, 1) d\gamma$. The l-th edge $(u_l \to v_l)$ in preference graph $\epsilon_i$ denotes the strict preference relation $\varphi(u_l) \succ \varphi(v_l)$. Consequently, we have $p(\varphi(u_l) \succ_{s_i} \varphi(v_l)|\mathbf{r}) = \Phi(z_i^l)$. With a two-layer preference graph, we are only interested in the directed edges between two layers as well as the equivalent relation in the top layer. We propose a new likelihood function for the $k$-th equivalent preference edge as follows,

$$p(u_k \leftrightarrow v_k|\mathbf{r}) \propto e^{-\frac{1}{2}(Q(s_i, \varphi(u_k)) - Q(s_i, \varphi(v_k)))^2} \tag{8}$$

where $u_k, v_k \in V^+$ and the k-th edge $(u_k \leftrightarrow v_k)$ denotes the equivalent relation $\varphi(u_k) \sim_{s_i} \varphi(v_k)$. We have $p(\varphi(u_k) \sim_{s_i} \varphi(v_k)|\mathbf{r}) = p(u_k \leftrightarrow v_k|\mathbf{r})$ that is shown in Eq.8. Then we compute the likelihood function for all observed preference graphs using the following equation,

$$\begin{aligned} p(\mathcal{G}|\mathcal{S}, \mathbf{r}, \theta) &= \prod_{i=1}^{\hat{n}} p(\epsilon_i|s_i, \mathbf{r}) = \prod_{i=1}^{\hat{n}} \prod_{l=1}^{n_i} \Phi(z_i^l) \\ \exp(\sum_{i=1}^{\hat{n}} \sum_{k=1}^{m_i} &-\frac{1}{2}(Q(s_i, \varphi(u_k)) - Q(s_i, \varphi(v_k)))^2). \end{aligned} \tag{9}$$

We put all the unknown parameters into a hyper-parameter vector $\theta = \left\{\kappa_{a_j}, \sigma_{a_j}, \sigma\right\}$, and then adjust the hyper-parameters on the basis of maximum a posterior estimation.

*3) Posterior inference:* Here we adopt a hierarchical model. At the lowest level are function values encoded as a parameter vector $\mathbf{r}$. At the top level are hyper-parameters in $\theta$ controlling the distribution of the parameters at the bottom level. Inference takes place one level at a time. At the bottom level, the posterior over function values are given by Bayes' rule as $p(\mathbf{r}|\mathcal{S}, \mathcal{G}, \theta) = p(\mathcal{G}|\mathcal{S}, \theta, \mathbf{r}) p(\mathbf{r}|\mathcal{S}, \theta) / p(\mathcal{G}|\mathcal{S}, \theta)$.

The posterior combines the information from the prior and the data, which reflects the updated belief about $\mathbf{r}$ after observing the decisions. By Eq. 4, our task is to minimize

the negative log posterior equation $U(\mathbf{r})$, which is

$$U(\mathbf{r}) = \frac{1}{2}\sum_{j=1}^{m}\mathbf{r}_{a_j}^T K_{a_j}^{-1}\mathbf{r}_{a_j} + \sum_{i=1}^{\hat{n}}\sum_{k=1}^{m_i}\frac{1}{2}(\sum_{j=1}^{m}\rho_{a_j}^{ik}\mathbf{r}_{a_j})^2 - \sum_{i=1}^{\hat{n}}\sum_{l=1}^{n_i}\ln\Phi(z_i^l). \quad (10)$$

Given the k-th equivalent relation $\varphi(u_k) \sim \varphi(v_k)$, let $\mathbf{\Delta}_k \triangleq \gamma(\hat{P}_{\phi(u_k)s_i} - \hat{P}_{\phi(v_k)s_i})(I_{\hat{n}} - \gamma\hat{P}_{a^*})^{-1}$, then we have

$$\rho_{a_j}^{ik} = \alpha_i[\mathbf{1}(a_j = \phi(u_k)) - \mathbf{1}(a_j = \phi(v_k))] + \mathbf{\Delta}_k\hat{I}_{a_j}$$

where $\hat{I}_{a_j}$ is a block matrix of $\hat{I} = [\hat{I}_{a_1}, \hat{I}_{a_2}, \cdots, \hat{I}_{a_m}]$ and $\alpha_i$ is a $1 \times \hat{n}$ vector whose entry $\alpha_i(i) = 1$, and $\alpha_i(-i) = 0$. The notation $\mathbf{1}(.)$ is an indicator function.

**Remark** Minimizing Eq.10 is a convex optimization problem. The proof can be found in our supplemental report [12].

At the minimum of $U(\mathbf{r})$ we have

$$\frac{\partial U}{\partial \mathbf{r}_{a_j}} = 0 \Rightarrow \hat{\mathbf{r}}_{a_j} = K_{a_j}(\nabla \log P(\mathcal{G}|\mathcal{S}, \hat{\mathbf{r}}, \theta)), \quad (11)$$

where $\hat{\mathbf{r}} = (\hat{\mathbf{r}}_1, \cdots, \hat{\mathbf{r}}_{a_j}, \cdots, \hat{\mathbf{r}}_m)$. In Eq.11, we can use Newton's method to find the maximum of $U$ with the iteration

$$\mathbf{r}_{a_j}^{\text{new}} = \mathbf{r}_{a_j} - (\frac{\partial^2 U}{\partial \mathbf{r}_{a_j}\partial \mathbf{r}_{a_j}})^{-1}\frac{\partial U}{\partial \mathbf{r}_{a_j}}.$$

### B. Model selection

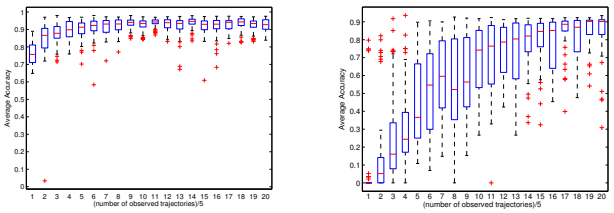
Model selection is the process of choosing a covariance function for a Gaussian process. The process can be considered to be training of a Gaussian process [13]. At the top level, we can optimize the hyper-parameters by maximizing the posterior over these hyper-parameters. The posterior $p(\theta|\mathcal{G}, \mathcal{S})$ is given by $p(\theta|\mathcal{G}, \mathcal{S}) = p(\mathcal{G}|\mathcal{S}, \theta)p(\theta)/p(\mathcal{G}|\mathcal{S})$, where the normalizing constant can be omitted for simplifying the optimization problem. If the prior distribution of hyper-parameters has no population basis, we assign the non-informative prior density to $\theta$. Optimization over $\theta$ becomes the problem of maximizing the marginal likelihood $p(\mathcal{G}|\mathcal{S}, \theta)$. We approximate the integral of the marginal likelihood $p(\mathcal{G}|\mathcal{S}, \theta)$ using a Laplace approximation local expansion around the maximum, which is written as

$$p(\mathcal{G}|\mathcal{S}, \theta) \approx p(\mathcal{G}|\mathcal{S}, \hat{\mathbf{r}}, \theta) \times p(\hat{\mathbf{r}}|\theta)\delta_{\mathbf{r}|\mathcal{S}}. \quad (12)$$

where $\delta_{\mathbf{r}|\mathcal{S}} = |-\nabla\nabla \ln P(\mathbf{r}|\mathcal{G}, \mathcal{S}, \theta)|^{-\frac{1}{2}}$ is the posterior uncertainty in $\mathbf{r}$, which is known as the Occam factor, automatically incorporating a trade-off between model fit and model complexity. As the number of data increases, the approximation is expected to become increasingly accurate. The marginal likelihood can be further written as

$$\log p(\mathcal{G}|\mathcal{S}, \theta) = -U(\hat{\mathbf{r}}) - \frac{1}{2}\log|I_{\hat{n}} + K\Pi|, \quad (13)$$

where $\hat{\mathbf{r}}$ is the MAP estimation in Eq.11 and $\Pi$ is the second derivative matrix of the sum of the second and third part in Eq. 10. Now we can find the optimal hyper-parameters by maximizing Eq.13.



(a) GPIRL accuracy (b) LIRL accuracy

Fig. 3. Average accuracy as a function of the number of observed decision trajectories, for GridWorld experiments.

### C. Posterior predictive reward

When the observed state-action pairs are limited, e.g. in the large state space or infinite state space, how to predict the reward at new state is desirable. Our IRL with Gaussian process provides a probabilistic model to predict reward on new coming state $s^*$, which is a Gaussian model $p(\mathbf{r}^*|\mathcal{G}, \mathcal{S}, s^*, \theta)$ with the following mean function

$$E(\mathbf{r}_{a_j}^*|\mathcal{G}, \mathcal{S}, s^*, \theta) = k_{a_j}(S, s^*)^T(K_{a_j} + \sigma^2 I_{\hat{n}})^{-1}\hat{\mathbf{r}}_{a_j}$$

and covariance function

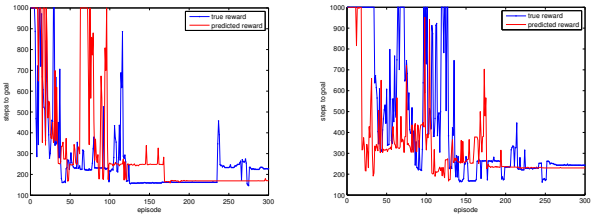
$$\text{cov}(\mathbf{r}_{a_j}^*|\mathcal{G}, \mathcal{S}, s^*, \theta) = k_{a_j}(s^*, s^*) - k_{a_j}(S, s^*)^T(K_{a_j} + \sigma^2 I_{\hat{n}})^{-1}k_{a_j}(S, s^*),$$

where $k_{a_j}(S, s^*)$ is the vector of covariance between the test point and training points for the covariance function relating to the action $a_j \in \mathcal{A}$.

## V. Experiments

In this section, we report on a simple GridWorld experiment in which an agent starts from the a square of the grid and attempts to navigate to the goal square, with the possibility of encountering obstacles that block movement to certain squares. The agent is able to take five actions: remaining in the current square or moving in one of the four cardinal directions. Each movement action results in movement in the intended direction with probability 0.65, movement in an unintended direction with probability 0.2, and failure to move with probability 0.15.

We compared three algorithms: our convex programming method from Section III (*CPIRL*), our Gaussian process method from Section IV (*GPIRL*), and the linear approximation method in [2] (*LIRL*). Given observation of a complete policy, each of the algorithms was successful in finding a reward vector that yields an optimal policy identical to that observed. For each of the reward vectors returned by the algorithms, we recorded the amount of computation time needed to find a best policy using reinforcement learning. Table I shows the average of these time over 50 simulations. Notably, reinforcement learning converges more quickly with reward vectors returned by CPIRL and GPIRL than with those returned by LIRL. We hypothesize our methods tend to shape reward, providing additional feedback to the agent and leading to an improvement in learning rate.




(a) 60-state discretization (b) 120-state discretization

Fig. 4. Solutions to the hill climbing problem based off true reward (blue) and reward recovered from GPIRL (red), for two levels of discretization.

Fig. 3 provides the basis for an accuracy comparison of *GPIRL* and *LIRL* for experiments in which only partial observations were available for reward learning. Accuracy is calculated to be the fraction of runs in which the apprentice is able to achieve the teacher's goal state. The process of computing accuracy includes: 1) generating some GridWorld problems and sampling the decision trajectories from the teacher's demonstration; 2) inferring the reward function using GPIRL and LIRL; 3) generating 1000 new GridWorld problems with random initial state and solving these problems by applying reinforcement learning using the reward output by IRL; 4) comparing the results of the GPIRL and LIRL apprentices with the teacher. If the apprentice reaches the teacher's goal state, we consider that trial a success for the apprentice. As can be seen in Fig. 3, the accuracy of GPIRL is higher than that of LIRL, especially when the number of observations is small. Additionally, GPIRL has clearly lower variance in accuracy.

TABLE I

TIME(SEC) TO FIND THE APPRENTICE POLICY

| GridWorld Size | LIRL | CPIRL | GPIRL |
|---|---|---|---|
| 10x10 | 2.61 | 2.06 | 1.20 |
| 20x20 | 20.05 | 15.75 | 9.32 |
| 30x30 | 75.12 | 64.30 | 35.11 |

We also performed an experiment based on a simulation of an under-powered car attempting to drive out of a U-shaped valley. In this simulation, the car lacks enough power to climb the valley slopes from a standstill. Instead, it must first reverse up a slope in order to accumulate energy that will help it rush up the opposite slope. We choose the car's position and velocity as state features, discretizing those naturally continuous quantities. To test GPIRL's ability to predict the reward on unseen states, we sampled only half the discretized states as the observation data for GPIRL. Given a state space with 120 states, for example, we would observe behavior in only 60 states. Figure 4 shows the number of steps needed to escape the valley for a range of starting conditions, or episodes, for policies learned from the true reward (blue) and from the reward returned by GPIRL (red). The results in the figure suggest that GPIRL is able to effectively recover the reward with incomplete observations, since the solver, using the reward predicted by GPIRL, has a performance on par with that of the teacher, using true reward.

## VI. CONCLUSIONS

We propose new IRL algorithms in the domain of convex programming. To deal with the IRL problems with ill-posed nature in large (or even infinite) state space, we model the reward using Gaussian process and interpret the observation of state-action space using preference graphs. Our posterior prediction method can estimate the reward at unobserved new coming states, which is promising for problems with large state space. Numerical experiments suggest that our method is able to find the reward approaching the true underlying reward with fewer observations than are needed with standard approaches. We will continue our research on IRL with Gaussian process in continuous space.

## ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation under Grant No. EEC-0827153.

## REFERENCES

[1] Nathan Ratliff, Brian Ziebart, Kevin Peterson, J. Andrew Bagnell, Martial Hebert, Anind K. Dey, and Siddhartha Srinivasa. Inverse optimal heuristic control for imitation learning. In *Proc. AISTATS*, pages 424–431, 2009.

[2] Andrew Y. Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *Proc. 17th International Conf. on Machine Learning*, pages 663–670. Morgan Kaufmann, 2000.

[3] Peter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proc. 21st International Conf. on Machine learning*, page 1. ACM, 2004.

[4] Umar Syed, Michael Bowling, and Robert E. Schapire. Apprenticeship learning using linear programming. In *Proc. 25th international Conf. on Machine learning*, pages 1032–1039. ACM, 2008.

[5] Gergely Neu and Csaba Szepesvari. Apprenticeship learning using inverse reinforcement learning and gradient methods. In *Proc. Uncertainty in Artificial Intelligence*, 2007.

[6] Ramachandran Deepak and Amir Eyal. Bayesian inverse reinforcement learning. In *Proc. 20th International Joint Conf. on Artificial Intelligence*, 2007.

[7] Umar Syed and Robert E. Schapire. A game-theoretic approach to apprenticeship learning. In *Advances in Neural Information Processing Systems*, pages 1449–1456. MIT Press, 2008.

[8] Krishnamurthy Dvijotham and Emanuel Todorov. Inverse optimal control with linearly-solvable mdps. In *Proc. 27th International Conf. on Machine learning*. ACM, 2010.

[9] J. Fürnkranz and E. Hüllermeier. Preference learning. In *Künstliche Intelligenz*, 2005.

[10] Chu Wei and Ghahramani Zoubin. Preference learning with gaussian processes. In *Proc. 22th Iinternational Conf. on Machine learning*, pages 137–144. ACM, 2005.

[11] Bellman R. *Dynamic programming*. Princeton University Press, 1957.

[12] Qifeng Qiao and Peter A. Beling. Inverse reinforcement learning with gaussian process (supplemental materials), 2010. http://people.virginia.edu/∼qq2r/IRLACCsupplements.pdf.

[13] Carl Edward Rasmussen and Christopher K.I.Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.