Proceedings of the ASME 2016 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2016
August 21-24, 2016, Charlotte, United States

DETC2016-59775

LEARNING HUMAN SEARCH STRATEGIES FROM A CROWDSOURCING GAME

Thurston Sexton
Mechanical Engineering
Arizona State University
Tempe, Arizona, 85287
Email: tbsexton@asu.edu

Max Yi Ren
Mechanical Engineering
Arizona State University
Tempe, Arizona, 85287
Email: yiren@asu.edu

ABSTRACT

There is evidence that humans can be more efficient than existing algorithms at searching for good solutions in high-dimensional and non-convex design or control spaces, potentially due to our prior knowledge and learning capability. This work attempts to quantify the search strategy of human beings to enhance a Bayesian optimization (BO) algorithm for an optimal design and control problem. We consider the sequence of human solutions as generated from BO, and propose to recover the algorithmic parameters of BO by maximizing the likelihood of the observed solution path. The method is different from inverse reinforcement learning (where an optimal control solution is learned based on human demonstrations) in that the latter requires near-optimal solutions from humans, while we only require the existence of a good search strategy. The method is first verified through simulation studies and then applied to the human solutions crowdsourced through a gamification of the problem under study [1]. We learn BO parameters from a player with a demonstrated good search strategy and show that applying the BO algorithm with these parameters to the game noticeably improves the convergence of the search from using a default BO setting.

1 Introduction

Optimal control and/or design problems often have large variable spaces and highly non-convex objectives and constraints, preventing effective or even tractable searches via existing algorithms. Despite this, human beings have demonstrated ability at finding good solutions for high-dimensional optimization problems. Examples can be found from crowdsourcing scientific solutions for protein folding [2, 3], RNA synthesis [4, 5], genome sequence alignment [6], robot arm movements [7], to name a few, as well as from our daily achievements in packing luggage, scheduling conference sessions [8], and playing games [1, 9, 10]. As a particular example, our previous study investigated the value of crowdsourcing NP optimal design and control problems by comparing the search performance of an anonymous crowd with an algorithm on an electric vehicle time-trial game [1]. In the game, the player needs to both determine the control policy and the final gear ratio to minimize energy consumption for completing a given track in 36 seconds. We found that the majority of the crowd had worse convergence performance, yet a small group of players identified good solutions earlier than the algorithm (see Fig. 1). While often attributed to the "creativity" or "intuition" of human beings, we consider such ability as prior knowledge, in the form of algorithmic parameters, that helps to enhance an inherent search algorithm used by human beings to find an optimal solution. The variability in the prior knowledge for a particular problem leads to different search capabilities of players. The above evidence and hypothesis motivate the question: Can we recover a player's knowledge and use it to improve a computer solver?

To this end, this paper investigates (1) how algorithmic parameters, when they exist, can be estimated based on demonstrated search trajectories, and (2) whether recovered parameters from a player can be used to improve the convergence of the solver, using the ecoRacer game as a case study. Specifically, we consider a player's search trajectory to be produced from a Bayesian Optimization (BO, also called "Efficient Global Opti-

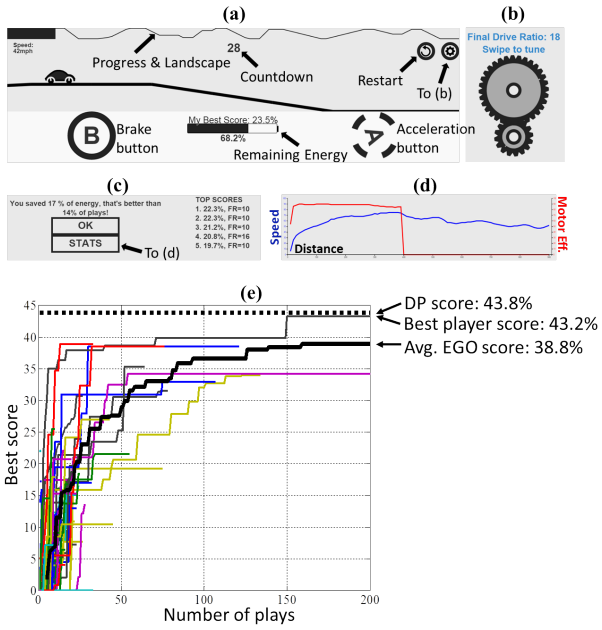

FIGURE 1: (a-d) ecoRacer interface: The player is asked to minimize energy consumption while completing the track in 36 seconds. Besides the control strategy, the player can also tune the final drive ratio, which affects both the maximum speed and maximum torque. (e) Results from the game showed that some players can identify good solutions earlier than a computer algorithm. Image reproduced from [1].

mization" [11]) algorithm where each new trial optimizes an expected improvement function learned and adjusted by the player during the search. This assumption is weakly supported by a recent study that showed that human searches in 1D optimization problems resemble BO [12] more than other gradient and non-gradient methods. We then introduce a likelihood model parameterized by the observed search trajectory, and through maximizing these the BO parameters can be estimated.

The paper provides two contributions: (1) Our empirical studies show that the proposed method can successfully recover algorithmic parameters when the search trajectories are created by BO. Further, by playing the ecoRacer game with parameters learned from a human player with demonstrated good search capability, the BO algorithm achieves a noticeable improvement in its convergence from using a default parameter setting. This indicates the potential usage of the proposed method when near-optimal solutions do not exist. See Sections 3 to 5. (2) We provide a discussion on the commonalities and differences between learning an optimization algorithm (this study) and learning an optimal control solution (inverse reinforcement learning), see Section 6.

1.1 Terminologies, Notations and Problem Statement

Consider an optimization problem $\mathrm{P} := \min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$ where $\mathcal{X} \subseteq \mathbb{R}^p$. For each iteration k, the state of the search can be represented by the tuple $s_k := (\mathbf{X}_k, \mathbf{f})_k$, where $\mathbf{X}$ and $\mathbf{f}$ represent samples in $\mathcal{X}$ and their corresponding objective values. Consider a search algorithm parameterized by Λ: each new sample is determined by the state and the parameter: $\mathbf{x}_{k+1} = \mathrm{argmax}_{\mathbf{x}\in\mathcal{X}} Q(s_k, \mathbf{x}|\Lambda)$ through the search criterion $Q(\cdot,\cdot|\Lambda)$ (called a "Q-function" to be consistent with optimal control convention). The N iterations of the search forms a trajectory $\zeta = \{(s_k, \mathbf{x}_{k+1})\}_{k=0}^{N-1}$. Without loss of generality, we assume all searches have the same number of iterations. This study proposes a method to estimate Λ given ζ, $f(\cdot)$, and $Q(\cdot,\cdot|\Lambda)$.

2 Preliminaries on Bayesian optimization

We start by introducing a Bayesian optimization algorithm with a specific search criterion $Q(\cdot,\cdot|\Lambda)$. BO is a non-gradient optimization algorithm suitable for problems with a black-box (and costly) objective and a continuous and bounded variable space [11, 13]. Briefly, the algorithm starts with a small set of N_k samples s_k at iteration $k = 0$. A Gaussian Process (GP) model (or more specifically a Kriging model) is then created based on s_0 to predict objective values over $\mathcal{X}$, with the following prediction $\hat{f}$ for input $\mathbf{x}$:

$$\hat{f}(\mathbf{x}; s_k, \Lambda) = \mathbf{b} + \mathbf{r}^T\mathbf{R}^{-1}(\mathbf{f}_k - \mathbf{b}), \tag{1}$$

where $\mathbf{b} = \frac{(\mathbf{1}^T\mathbf{R}^{-1}\mathbf{f}_k)}{(\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1})}$, $r_i = \exp\left(-(\mathbf{x}-\mathbf{x}_i)^T\Lambda(\mathbf{x}-\mathbf{x}_i)\right)$ for $i = 1,\cdots,N_k$, $R_{ij} = \exp\left(-(\mathbf{x}_i-\mathbf{x}_j)^T\Lambda(\mathbf{x}_i-\mathbf{x}_j)\right)$ for $i,j = 1,\cdots,N_k$, $\mathbf{1}$ is a column vector with all ones, and $\Lambda = \mathrm{diag}(\lambda_1,\cdots,\lambda_p)$ is a diagonal covariance matrix. The Kriging variance σ^2 can be derived as

$$\sigma^2 = \left(1 - \mathbf{r}^T\mathbf{R}^{-1}\mathbf{r} + \frac{(1-\mathbf{1}\mathbf{R}^{-1}\mathbf{r})^2}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}}\right) \cdot \frac{(\mathbf{f}_k-\mathbf{b})^T\mathbf{R}^{-1}(\mathbf{f}_k-\mathbf{b})}{N_k}. \tag{2}$$

For a new sample $\mathbf{x} \in \mathcal{X}$, its *expected improvement* from the current best objective value $f_{\min}$ (assuming a minimization problem) follows

$$Q_{EI}(s_k, \mathbf{x}|\Lambda) = (f_{\min} - \hat{f})\Phi\left(\frac{f_{\min}-\hat{f}}{\sigma}\right) + \sigma\phi\left(\frac{f_{\min}-\hat{f}}{\sigma}\right), \tag{3}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the cumulative distribution function and probability density function for a standard normal distribution, respectively. A common sampling strategy for BO is to choose $\mathbf{x}_{k+1} = \mathrm{argmax}_{\mathbf{x}\in\mathcal{X}} Q_{EI}(s_k, \mathbf{x}|\Lambda)$. This nested problem can be

highly non-convex, and a solution (often sub-optimal) can be found using a global optimization algorithm. Eq. (1) and Eq. (3) are then updated based on s_{k+1}.

3 Estimation of BO parameters

We now consider the problem of estimating Λ given ζ, $f(\cdot)$, and $Q_{EI}(\cdot,\cdot|\Lambda)$. To characterize the nested optimization $\text{argmax}_{\mathbf{x}} Q_{EI}(s_k, \mathbf{x}|\Lambda)$ involved in BO, we assume that the probability density of choosing any $\mathbf{x} \in \mathcal{X}$ as the next sample conditioned on the current state s_k follows a Boltzmann distribution:

$$p(\mathbf{x}|s_k) = \frac{\exp\left(\alpha Q_{EI}(s_k, \mathbf{x}|\Lambda)\right)}{Z(s_k, \Lambda, \alpha)}, \tag{4}$$

where $Z(s_k, \Lambda, \alpha)$ is a partition function defined as:

$$Z(s_k, \Lambda, \alpha) = \int_{\mathbf{x}\in\mathcal{X}} \exp\left(\alpha Q_{EI}(s_k, \mathbf{x}|\Lambda)\right) d\mathbf{x}. \tag{5}$$

The value of α indicates the quality of the nested global optimization: A large α indicates that the global optimum of $Q_{EI}(\mathbf{x})$ can be found with high probability, while an α close to zero indicates random search. It should be noted that the calculation of the partition function is intractable in practice when $\mathcal{X}$ is high-dimensional. In this study, we resort to a primitive discrete approximation where a set $\mathbf{X}_s$ is pre-determined through Latin Hypercube sampling (LHS):

$$\hat{Z}(s_k, \Lambda, \alpha) = \frac{1}{|\mathbf{X}_s|} \sum_{\mathbf{x}'\in\mathbf{X}_s\subset\mathcal{X}} \exp\left(\alpha Q_{EI}(s_k, \mathbf{x}'|\Lambda)\right). \tag{6}$$

The likelihood of parameters α and Λ for a given search trajectory ζ is then

$$\mathcal{L}(\alpha, \Lambda; \zeta, s_0) = P(\zeta|s_0; \alpha, \Lambda) = \prod_{k=0}^{N-1} P(\mathbf{x}_{k+1}|s_k; \alpha, \Lambda). \tag{7}$$

The maximum likelihood estimation of α and Λ can be derived by solving

$$\max_{\alpha,\Lambda} \sum_{k=0}^{N-1} \log\left(\mathcal{L}(\alpha, \Lambda; \zeta, s_0)\right). \tag{8}$$

Solving Eq. (8) is costly due to its non-convexity and repeated updates of $\hat{Z}(s_k, \Lambda)$. Discussions on potentially efficient optimization techniques will be presented in Section 6.

4 Simulation studies

We use simulation studies to examine the behavior of BO under various Λ settings, and to verify the performance of the proposed estimation method.

4.1 Estimation performance

The simulation studies are detailed as follows: Three standard test functions are used, namely, the 2D Branin function, the 2D Six-hump Camelback function, and the 6D Rosenbrock function. For each function, BO is run for up to 100 iterations, and can terminate when the expected improvement for the next iteration is less than 10^{-3}. At each iteration, the expected improvement is maximized using a multi-start L-BFGS-B[1] (algorithm [14] with 100 initial points sampled in the corresponding feasible variable space through LHS. To initialize a BO search, we use 10 initial samples for s_0, also drawn through LHS. A set of $\Lambda^* = 0.01I, 0.1I, 1.0I, 10.0I$ are used for the studies, where I is the identity matrix. A total of 30 BO trajectories are recorded for each of the #function $\times$ Λ^* cases. The convergence of BO under all three test functions, and settings of Λ^*, are summarized in Fig. 2. To verify the proposed estimation method, we perform a grid search for the maximization problem in Eq. (8) using $\alpha = 0.01, 0.1, 1.0, 10.0$ and $\Lambda = 0.01I, 0.1I, 1.0I, 10.0I$. Fig. 3 lists the negative log-likelihood values ($-\log\mathcal{L}(\alpha, \Lambda; \zeta, s_0)$) on the grid for all three cases and four Λ settings. Due to the randomness in the calculation of $\hat{Z}(s_k, \Lambda)$, the reported values are sample means from the 30 trials. Sample standard deviations are reported in brackets. The estimates with the best mean likelihood values are denoted by $\hat{\Lambda}$ and $\hat{\alpha}$. From the result, we see that for all cases, $\hat{\Lambda}$s match their ground truth, and the corresponding $\hat{\alpha}$s in the 2D functions are estimated as the largest value of the grid, i.e., 10.0. This result is reasonable since the gradient-based algorithm applied to solve the nested improvement maximization problems can successfully find good local solutions in the constrained 2D spaces (although global solutions are not guaranteed). For the 6D Rosenbrock function, the estimation results $\hat{\Lambda}$ are in general consistent with Λ^*, expect for $\Lambda^* = 0.1$ where $\hat{\Lambda}$ could take either 0.1 or 1.0. In addition, the lower values of $\hat{\alpha}$ in this case automatically reflect the fact that solving the improvement maximization problem in a 6-dimensional space using the same settings as in the 2D cases leads to less probability of choosing the global optima.

In addition, we demonstrate the effect of anisotropy of Λ^* on optimizing a simple 2D parabolic function, using $\lambda_y = 1.0$ and $\lambda_x \in [0.01, 1.0, 100.0]$. From Fig. 4, one can see that a high weight in one direction often leads to overly exploitation of the existing solution, while a low weight leads to excessive exploration. This conclusion will be revisited during the analysis of the real player data. Also notice that the results for $\lambda_x = 0.01, \lambda_y = 1.0$ and $\lambda_y = 100.0, \lambda_y = 1.0$ are not symmetric.

5 BO on ecoRacer with $\hat{\Lambda}$ Learned from Human Plays

With support from the simulation studies, we now apply the estimation method to the human data previously collected from

[1] Bounded, Limited memory, Broyden-Fletcher-Goldfarb-Shanno algorithm

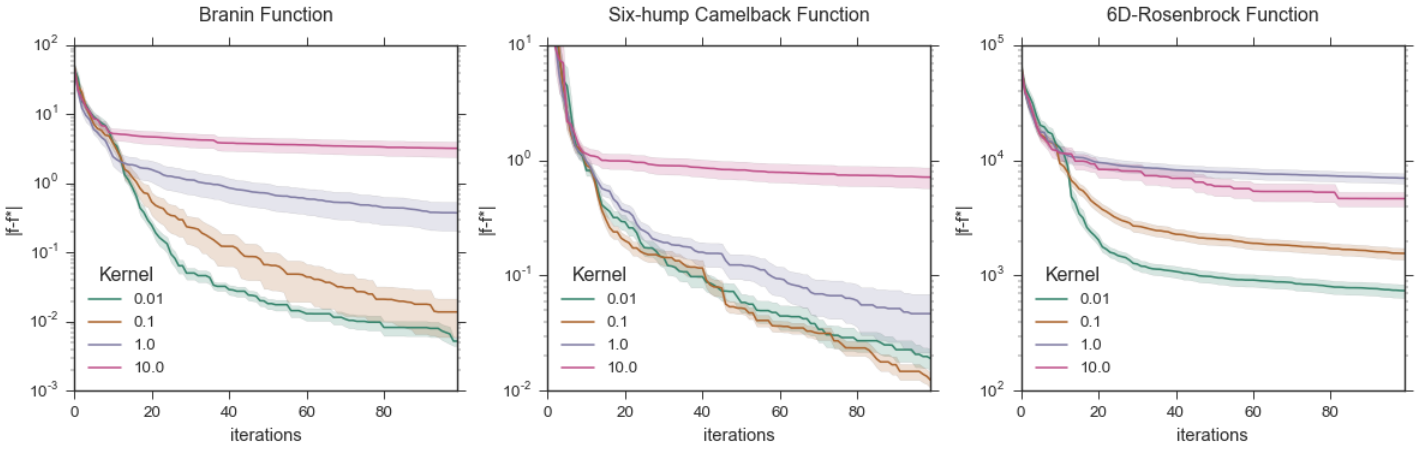

FIGURE 2: Convergence behavior for the three test functions under isotropic Λ. The parameters Λ for BO have a marked effect on the actual convergence rate. These plots contain 1-σ confidence intervals over 30 trials, via 5k bootstrap sampling [15].

the ecoRacer game[2], and examine if the learned $\hat{\Lambda}$ can improve the performance of BO for the game.

5.1 Dimension reduction for player control signals

Each game play data-set consists of (1) a final gear ratio used for the vehicle, (2) a control signal with acceleration and braking records, and (3) a corresponding game score. The length of a raw control signal matches that of the track, i.e., actions are stored for each of the 18160 distance steps. Due to the fact that control signals across plays share common patterns, and that BO is ineffective for high-dimensional problems, it is necessary to encode the control signals from all players to a lower dimensional space. In [1], this was done by introducing state-dependent bases to parameterize the action. The bases include the states (the velocity of the car, slope of the track, distance to the terminal, remaining battery energy, time spent) and polynomial terms based on these states. The underlying assumption that human players are aware of all the state-dependent bases is untested. Therefore this work explores a different approach to dimension reduction by using Independent Component Analysis (ICA). In the following, we justify the use of ICA.

In experimental psychological literature, research has shown that humans are prone to focusing on only salient features of their environment or problem, filtering out other information. In fact, concurrent performance of multiple tasks consistently leads to impairment in one or more of those tasks [16]. This causes us to switch between tasks quickly if they must be performed at the same time, and the time we spend on each task before switching back is related to our observed productivity at it, called discretionary task interleaving. Such a view is derived from optimal foraging theory, which understands animal foraging as an optimization of the rate of energy gain, and then viewing human behavioral solutions as ones that optimize the rate of information gain in the problem [17]. An example of this is a studying strategy while preparing for an exam, and the tendency we have to switch between difficult and easy topics depending on our perceived productivity at learning them [18].

Similarly, we propose that a human will split the problem presented to them in ecoRacer into separate sub-tasks, in such a way as to maximize their rate of information gain as they switch between exclusive focus on each one. Each sub-task will be an update decision for a salient feature of the control space. One natural separation of the tasks is to spatially segment the large track into smaller stretches of track. To maximize the information gained from exclusive focus on each, we hypothesize that these segments will contain a minimum amount of shared information with each other. The control decisions in each segment will therefore be minimally influenced by the shape of other segments.

Specifically, we hypothesize that human players segment the solution space $\mathcal{X}$ into m discrete sections: $\{\mathcal{X}_i | \forall_{i \leq m} \mathcal{X}_i \subseteq \mathcal{X}\}$, such that some measure of independence $F(\mathcal{X}_1, \cdots, \mathcal{X}_m)$ is maximized. Separation of observed signals into a number of independent components is the task of blind source separation [19], which can be elegantly addressed using Independent Component Analysis (ICA) [20]. Formally, ICA is a linear static transformation of an observed set of N signals $\mathbf{u} = [u_1, \cdots, u_N]$, each with length M, into a desired number (N^*) of independent components $\mathbf{s} = [s_1, \cdots, s_{N^*}]$ with the same length: $\mathbf{u} = \mathbf{A}\mathbf{s}$. $\mathbf{A}$ is the mixing matrix, so to get a reduced dimensionality representation of another signal we use the un-mixing matrix $W = \mathbf{A}^{-1} = [\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_n]^T$, where each of the N^* w's has

[2]The raw data from 2391 plays of 124 players is available at: `http://www.public.asu.edu/~yren32/resource/RenPapers/idetc2015game_data.zip`.

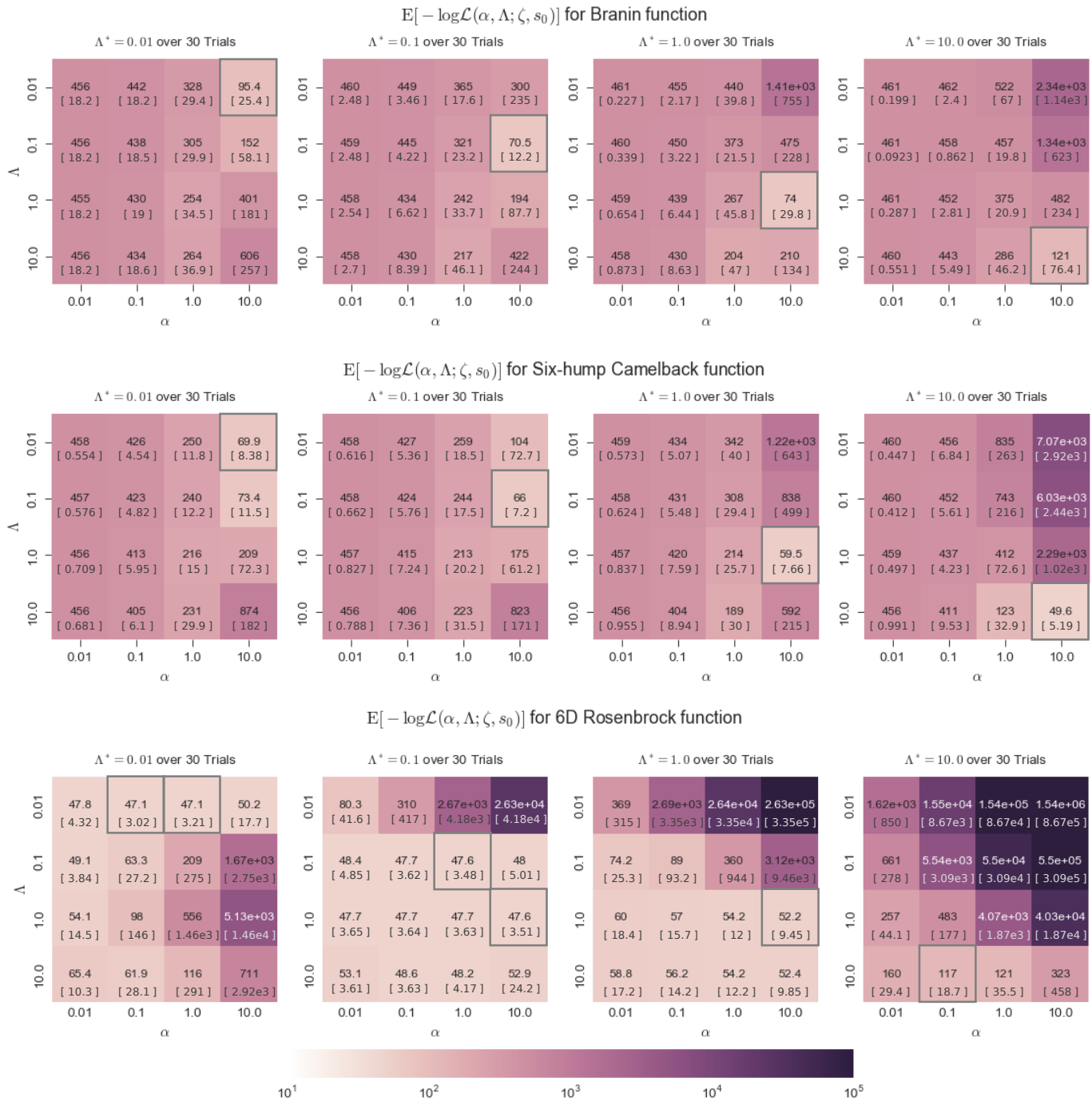

FIGURE 3: Maximum likelihood grid search for all three test functions under different settings for Λ^*. All cases were able to maximize the likelihood of the true setting in comparison to the other available options. Most likely settings are highlighted (lower is better). Results are averaged over all 30 trials, with sample standard deviations displayed in brackets.

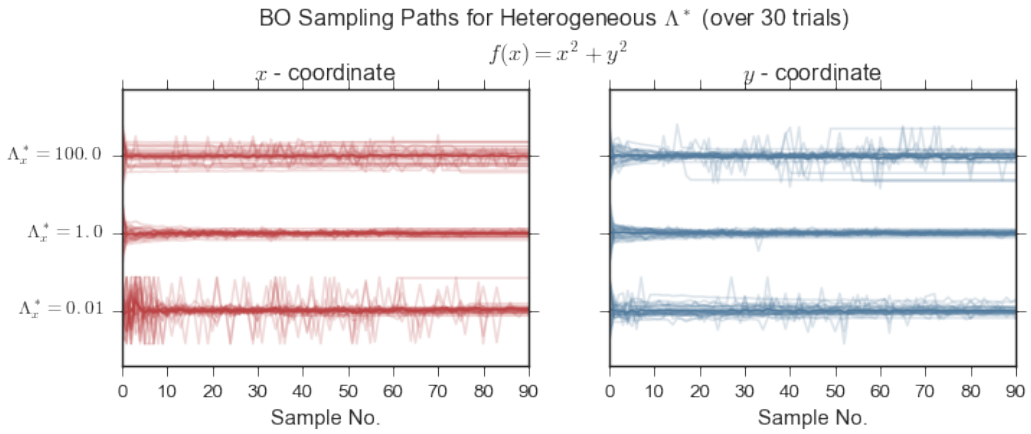

FIGURE 4: Sampling paths for BO using anisotropic Λ. While maintaining a constant $\lambda_y = 1.0$, changing the kernel distance weighting in the other direction has an effect on the sample paths in both directions. While all eventually converge to the true minimum, too high of a parameter value will cause premature certainty, while too low will cause a constant state of uncertainty.

length M. Unlike separation into Principal Components, where bases are orthogonal, ICA assumes that each of the s_i share a minimum amount of information, maximizing their mutual independence. [3]

By applying ICA to the set of all 2391 recorded human plays, so that $N = 2391$ and $M = 18160$, a reduced-dimensionality representation of the control-space (i.e. the track) can be obtained, where inputs are scalar in $\mathbb{R}^{30}$, with the weight of each input determining the control decision, and each dimension determining the location of that control on the track. This is opposed to the original ternary in $\{-1, 0, 1\}^{18160}$. Because ICA tends to separate complicated non-periodic domains into a set of edge-filter-like bases [21], the components arrived at using ICA are naturally correlated with spatial locations, much like wavelets, with each having a peak at some unique position (see Fig. 5).

While it is theoretically possible to find some "most likely" number of bases using information-theoretic criteria for model selection [22], this work assumes $N^* = 30$ important components to the track. These were able to capture $> 95\%$ of the original control input behavior, while maintaining a reasonable dimensionality for use in the BO solver. [4]

Note that ICA bases have no inherent sorting (unlike PCA), and are not identically scaled to the supposed true "source" signals. However, as figure 5 shows, the dimension reduction from ICA on the human play data can be easily matched to discrete locations along the track due to their wavelet-like shape.

5.2 BO performance using learned $\hat{\Lambda}$

Notice that the computational cost for evaluating the likelihood function in Eq. 4 scales quadratically with the number of games played ($|\zeta|$). Therefore we estimate Λ using the search trajectory from the player who achieved the second highest score with only 31 plays in total[5]. This player is referred to as P2. The plays are first encoded using the ICA bases in a 30-dimensional space, alont with a scaled representation of the final gear ratio, for a total of 31 dimensions to search. The maximum likelihood estimator, $\hat{\Lambda}$, is then derived by solving Eq.(8) with a grid search on $\hat{\alpha}$. See the summary in Table 1. The bounds on the diagonal values of $\hat{\Lambda}$ are set as $[0.01, 10.0]^{30}$. The optimal $\hat{\alpha}$ is found to be 10.0.

We use Fig.5 as an attempt to qualitatively verify $\hat{\Lambda}$: We first mark the peak locations of the ICA bases on the track (as shaded lines), and then visualize $\hat{\Lambda}$ on the track, where darker shade means higher lambda (more certainty).

1. The high $\hat{\lambda}$ value associated with basis closest to the starting point of the track indicates that the exploration in that component is neglected by the player. This is qualitatively consistent with the observation that P2 ceases to change his control strategy at the beginning of the track after relatively few plays (see Figure 6), due to the obvious fact that the car has to be accelerated to get over the first hill.
2. Intermediate λ indicates a region with heightened focus in general, where previous plays clearly impact the update de-

[3] The basis vectors are usually found by minimizing their mutual information, via maximizing F as the Kullback-Leibler Divergence, or some similar measure of relative entropy. If $|x| > |s|$, as is the case when reducing dimensionality, A^{-1} will in fact be the pseudo-inverse

[4] Using 1000 PCA components as preprocessing, the most likely number of ICA components using MDL, AIC, and KIC information criteria were 187, 464, and 373 components, respectively [22]. While this dimensionality could make sense from a neurological perspective ($36\text{s}/187 = 192\text{ms}$ is close to the range for the time-frame of attentional blink, 200-500 ms [16]), this is intractable for BO as used in this work.

[5] The player with the highest score played more than 150 times.

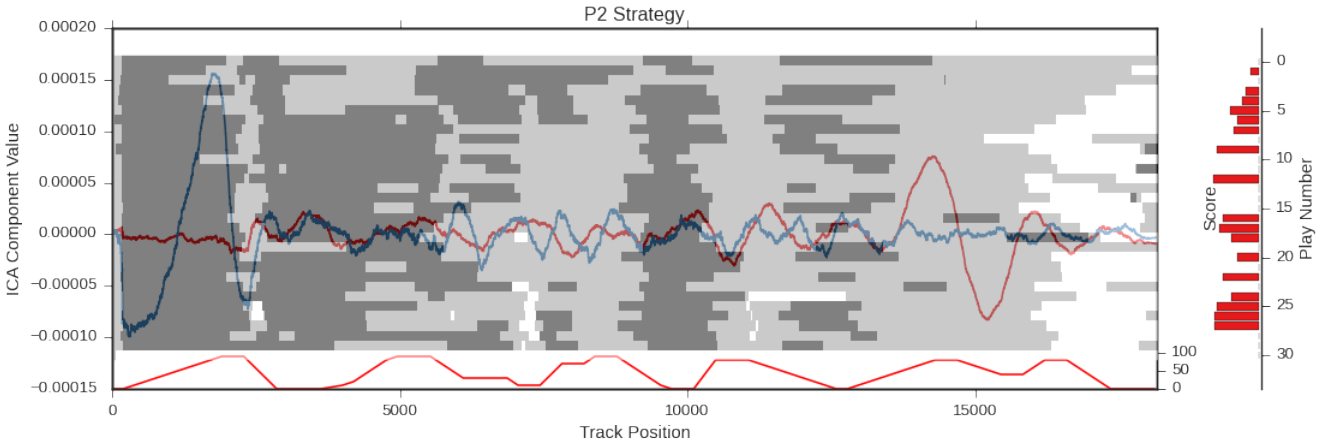

FIGURE 6: P2's strategy, visualized as a ternary signal matrix (black:accelerate, white:brake, gray:no action). Track is shown below signals to demonstrate locality of decisions. Also included is a plot of the two highest-weighted ICA components from $\hat{\Lambda}$. Note that these locations show a high amount of exploitation over exploration, as $\hat{\Lambda}$ suggests for these components.

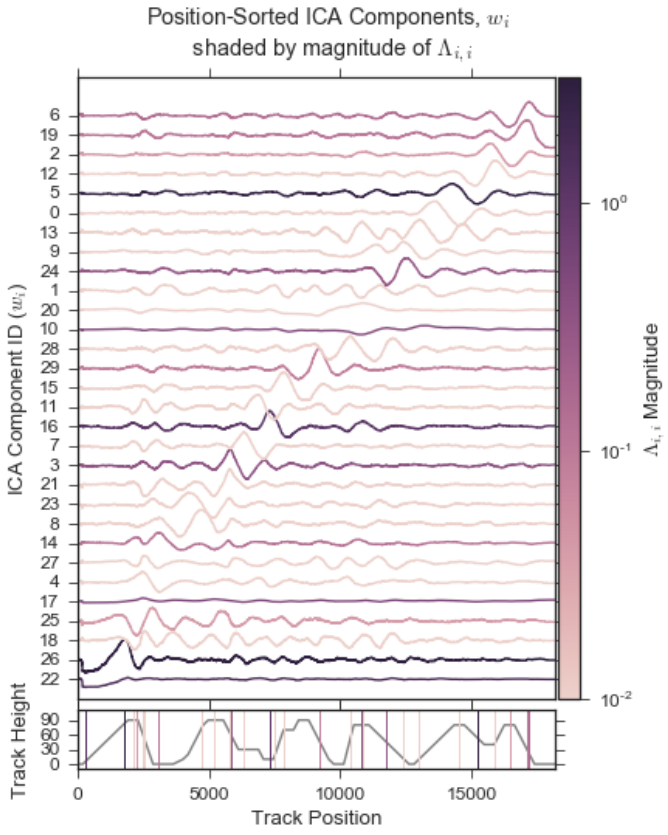

FIGURE 5: (Top) Learned ICA bases from all human plays, colored by $\hat{\Lambda}$. (Bottom) Locations of peak basis magnitudes marked on the track. Note the uneven distribution of feature locations. Components are sorted by $\arg\max_t |s_i(t)|$

TABLE 1: The $-\log \mathcal{L}(\alpha, \Lambda; \zeta, s_0)$ solution to the objective function, Eq.(8) for various α.

Trial	α	Min. Obj.
0	0.1	192.2
1	1.0	187.4
2	5.0	161.7
4	**10.0**	**144.7**
5	100.0	190.1

cision, but exploration continues.

3. Minimal λ implies a lack of structure for P2's decisions here; this can be interpreted as viewing this location as relatively unimportant to the optimization in general.

We now examine the effect of $\hat{\Lambda}$ on BO by running the search algorithm on the ecoRacer game using both an isotropic $\Lambda = I$ and $\hat{\Lambda}$, each with 200 iterations and 20 independent runs. The results are summarized in Fig.7. The convergence using $\hat{\Lambda}$ noticeably outperforms that of the standard BO.

6 Discussion

The above study provides a starting point for investigations on quantitative incorporation of human solution-search data into computational algorithms. Yet, many urging questions are not answered in this preliminary work. This section will address a few notable ones: We first discuss the potential cases where the pro-

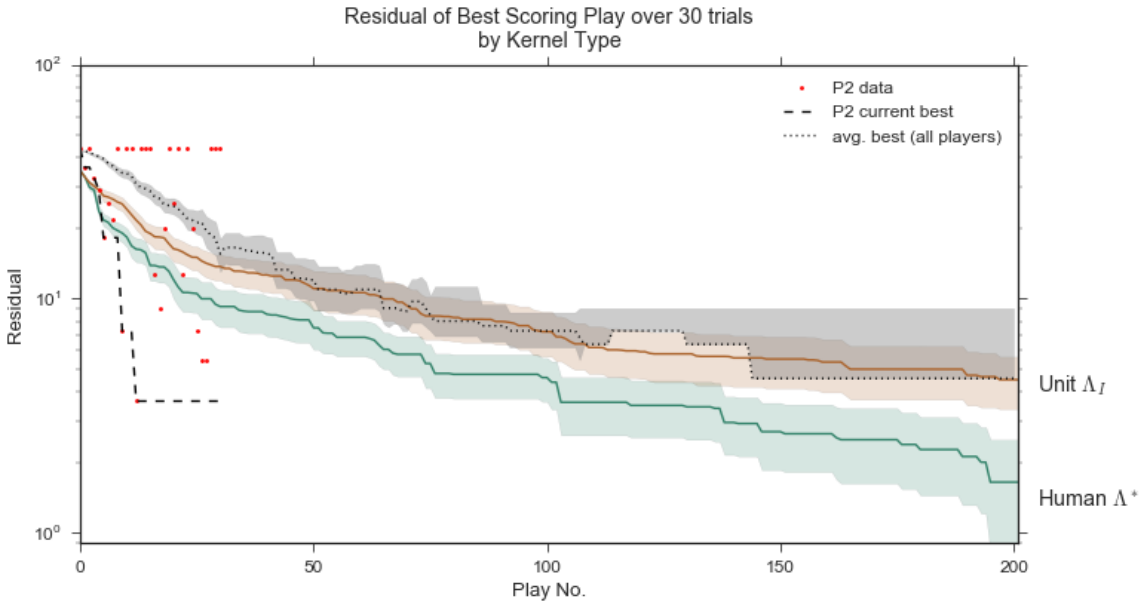

FIGURE 7: The residual of current best score vs. the known solution, for both the human-learned $\hat{\Lambda}$ and the standard unit Λ_I. Since a non-deterministic, genetic algorithm-based method was used to maximize the expected improvement for the validation BO process, the results are shown as averages over 30 trials. Average current best for all players is also shown, matching closely with Unit Λ_I. 68% confidence intervals were calculated via 5k bootstrap samples.

posed algorithm is useful in practice, and then provide a technical discussion on the connection between this work and (inverse) reinforcement learning, and lastly we comment on computational issues.

6.1 Scenarios where this work could be useful

The presented case using human data has limited appeal in practice since we learn $\hat{\Lambda}$ all plays from P2 to surpass a default algorithm. In fact, one could achieve faster convergence by using all P2's plays to initialize BO with default settings ($\Lambda = I$). Yet, the simulation studies indicates that with if a good estimation of Λ can be derived from a good search strategy with some limited amount of solutions, one could achieve significantly improved search efficiency. This leads to the question: In what scenarios would learning from limited and non-optimal human solutions be useful? Before we speculate on the potential answers, we shall note that there are limited cases in real life where human beings learn from searches: During the search of a design or control solution, the knowledge we borrow from existing solutions are often merely the solutions themselves, from where we perform local exploration (This is equivalent to initializing a search algorithm with better guesses). On the other hand, understanding the rationale behind changes in a sequence of solutions is a higher-level mental activity that is not necessarily intuitive to human beings. With this, we speculate on two scenarios where this work could be useful.

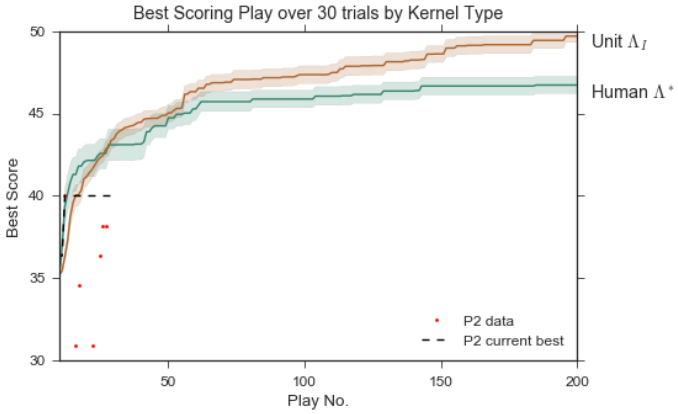

FIGURE 8: The current best score for both the human-learned Λ^* and the standard unit Λ_I, when both are initialized with P2's first 11 plays. Λ^* was only found using those 11 plays. Initial improvement is seen for about 20 plays, where Λ^* discovers P2's best strategy (their 12th play), but the advantage diminishes after this, falling behind Λ_I.

The first scenario concerns improving the search performance with limited human demonstrations. As we show in Fig.8, in this situation, the proposed algorithm learns BO parameters that allow marginal improvements from the default parameters

in a short term. It is worth investigating if the BO algorithm with $\hat{\Lambda}$ updated *along human plays* can always stay ahead of the algorithm with default settings (and the human player).

The second scenario considers transferability of the learned knowledge from one problem to another. Taking the ecoRacer case study as an example, the parameters in $\hat{\Lambda}$ are associated with discretized regions of the track. Thus it is possible to map $\hat{\Lambda}$ from descriptors of the problem (track) when plays on multiple tracks are collected. This would allow a prediction on $\hat{\Lambda}$ for a new problem (track). One major challenge specific to BO is the dependence of the learned knowledge ($\hat{\Lambda}$) on the initial plays s_0. A solution to this could be to treat the initial plays, jointly with the problem descriptors, as inputs to the predictive model.

6.2 The difference between learning from searches and learning from solutions

The problem definition and solutions introduced in this paper are closely related to reinforcement learning (RL) and Inverse RL (or called imitation learning, apprenticeship learning, or inverse optimal control, with subtle differences among some). To elucidate, we first introduce Markov Decision Processes (MDP) that are common settings of RL and IRL problems and make an analogy between RL and problem of designing an optimization algorithm (denoted as DO). We then introduce IRL techniques most related to this paper and distinguish our problem from existing IRL problem formulations by highlighting its unique challenges.

6.2.1 Markov Decision Processes Formally, an MDP consists of a tuple $< S, A, T, R, \gamma, b_0 >$ where: S is a finite set of states; A is a finite set of actions; $T(s, a, s')$ determines the probability of changing from state s to s' when action a is taken; $R(s, a)$ is the immediate reward of taking action a at state s; $\gamma \in [0, 1)$ is the discount factor; $b_0(s)$ specifies the probability of starting the process at state s. In RL, a control policy is a mapping from a state to an action $\pi : S \to A$. The long-term *value* of π for a given state s is: $V^\pi(s) = R(s, \pi(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s') V^\pi(s')$, and the value of π is the expectation: $V^\pi = \sum_{s \in \mathcal{S}} b_0(s) V^\pi(s)$. Similarly, the Q-function is defined as $Q^\pi(s, a) = R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') V^\pi(s')$. An RL algorithm finds the optimal policy π^* that satisfies the Bellman optimality condition: $V^*(s) = \max_{a \in A} \left(R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') V^*(s') \right)$. Conceptually, for a given reward function R, the RL algorithm identifies a Q-function $Q^*(s, a)$ such that $\pi^*(s) = \text{argmax}_{a \in A} Q(s, a)$.

6.2.2 A Comparison Between RL, IRL and DO Here we make an analogy between Reinforcement Learning (RL) and Designing an Optimizer (DO). In DO, an optimization algorithm needs to be identified as a "control policy", where the value of the policy is the best objective value after a finite number of iterations, the states are the existing samples ($\mathbf{x}$ and $f(\mathbf{x})$ pairs), the actions are the next samples, the transition function corresponds to the evaluation of the objective function, and the reward is constantly 0 for all states and actions until the last iteration, and $f(\mathbf{x}_N)$ for the last iteration. Consider the particular case of designing BO algorithms. The diagonals of Λ are analogous to unknown policy parameters, and the expected improvement criterion is a Q-function heuristically designed, from where the next action ($\mathbf{x}$) is determined. It is worth noting that DO differs from RL in two significant ways: (1) The state s is vectorial for MDPs, while the equivalent state in DO is the cumulative sample set ($(\mathbf{x}, f(\mathbf{x})$ tuples)[6]. One could, however, consider the joint set $\mathcal{X}^N \times \mathbb{R}^N$ as the set of states for BO where N is a given maximum iteration number, although this could lead to an enormous state set when N and $|\mathcal{X}|$ are large. More importantly, (2) RL relies on the Bellman optimality condition, while for DO, it is not clear if there exists a criterion to tell when the algorithm is optimally configured to solve the problem at hand.

While RL identifies an optimal control policy for an MDP with a given reward function, real-world applications hardly define a reward function, e.g., the reward for "drives well" cannot be explicitly defined. Alternatively, it is easier for a human expert to demonstrate state-action sequences that are near-optimal for a certain task. IRL techniques have thus been developed to identify the inherent reward (and thus the Q-function) that explains human demonstrations, either by estimating the reward parameters Λ so that the demonstrated policy has a higher value than any other policies by a margin [23–26], or by maximizing the likelihood of Λ by assuming near-optimal control of the demonstration [27,28]. Our method in Section 3 is mathematically similar to the maximum entropy IRL approach from the latter group. In its simplest setting, this method proposes the following maximum likelihood estimation for the reward based on a set of demonstrations $\{\zeta\}$:

$$\hat{\Lambda} = \text{argmax}_\Lambda \sum_{\{\zeta\}} \log P(\zeta|\Lambda), \tag{9}$$

where the probability

$$P(\zeta|\Lambda) = \frac{\exp\left(\sum_{(s_i,a_i)\in\zeta} R(s_i, a_i, \Lambda)\right)}{\prod_{(s_i,a_i)\in\zeta} Z_i(\Lambda)}, \tag{10}$$

and $Z_i(\Lambda)$ is a partition function for the visited state s_i. One can notice the similarities between Eqs. (8) and (9): (1) Both are maximum likelihood estimations of parameters related to the Q-function. In Eq. (9), this is through the estimation of the reward R. (2) Both involves partition functions that are hard-to-compute, state-related, and variable-dependent. Due to the dependency of partition function on Λ, a direct Markov-Chain Monte Carlo (MCMC) approach on the space of Λ's (e.g., as in [28]) cannot be applied since the partition values for two different samples of Λ do not cancel. In [27], this computational challenge is addressed

[6] Gradient-based algorithms could be an exception when only the current sample is used to determine the next sample.

by an "Expected Edge Frequency Calculation" algorithm that has a complexity of $O(N|S||A|)$ for each gradient calculation of the objective in Eq. (9), where N is a large number. This approach can be infeasible for Eq. (8) since (1) the space $\mathcal{X}$ is usually continuous, and (2) even with a discretization of $\mathcal{X}$, the enormous size of S and A can easily make the calculation intractable, based on the discussion in Section 6.2. While the mathematical formulations are similar, we shall note that IRL and DO are conceptually different: In IRL, human demonstrations are used to explain the underlying reward, and in turn to derive the optimal control parameters. Thus, human demonstrations are required to be near-optimal, and no causality is modeled among these demonstrations. In DO, however, human demonstrations are considered as a sequence of related actions, from where the "optimal controller" (e.g., Λ in BO) is directly derived. The demonstrations do not need to be close to the optimal solution.

To summarize, the presented problem can be considered as designing an optimization algorithm by human demonstrations. We show that this problem is similar to an IRL in that both aim to speed up the forward search by inversely learning a useful "control policy" through human demonstrations. However, the calculation of the partition function is intractable for maximum likelihood estimation of the algorithmic parameters, and cannot be addressed by existing approaches used in IRL.

6.3 Computational difficulties

In practice, the partition function $Z(s_k, \Lambda)$ is infeasible to calculate directly, since it regularizes the probability space via a sum over all state and action pairs. In this paper, LHC sampling was employed to randomly sample the action space ($\mathcal{X}$). However, this in general requires a large number of samples to converge to the true value of Z. Further work aims to use Markov-chain Monte Carlo methods such as NUTS [29] to speed up the convergence.

7 Conclusions

We showed in this paper the possibility of retrieving quantitative domain knowledge about an NP design optimization problem from human solvers that have demonstrated good ability at finding solutions, and using that knowledge to improve a computer solver. Specifically, the presented method found algorithmic parameters of a Bayesian Optimization algorithm by maximizing the likelihood of the observed sequence of solutions. Future work is necessary to investigate real-world scenarios where the proposed method could lead to more cost-effective searches than a purely computational approach. In particular, it is yet to be shown that the gathered domain knowledge can be transferred from existing problems to solve new problems.

Acknowledgement

This work has been supported by the National Science Foundation under Grant No. CMMI-1266184 and the start-up funding from Arizona State University. These supports are gratefully acknowledged.

REFERENCES

[1] Ren, Y., Bayrak, A. E., and Papalambros, P. Y., 2015. "ecoracer: Game-based optimal electric vehicle design and driver control using human players". In ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers, pp. V02AT03A009–V02AT03A009.

[2] Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., Leaver-Fay, A., Baker, D., Popović, Z., et al., 2010. Foldit. `http://fold.it`.

[3] Khatib, F., Cooper, S., Tyka, M. D., Xu, K., Makedon, I., Popović, Z., Baker, D., and Players, F., 2011. "Algorithm discovery by protein folding game players". *Proceedings of the National Academy of Sciences,* ***108***(47), pp. 18949–18953.

[4] Lee, J., Kladwang, W., Lee, M., Cantu, D., Azizyan, M., Kim, H., Limpaecher, A., Yoon, S., Treuille, A., and Das, R., 2014. eterna. `http://eterna.cmu.edu`.

[5] Lee, J., Kladwang, W., Lee, M., Cantu, D., Azizyan, M., Kim, H., Limpaecher, A., Yoon, S., Treuille, A., and Das, R., 2014. "Rna design rules from a massive open laboratory". *Proceedings of the National Academy of Sciences,* ***111***(6), pp. 2122–2127.

[6] Kawrykow, A., Roumanis, G., Kam, A., Kwak, D., Leung, C., Wu, C., Zarour, E., Sarmenta, L., Blanchette, M., Waldispühl, J., et al., 2012. "Phylo: a citizen science approach for improving multiple sequence alignment". *PloS one,* ***7***(3), p. e31362.

[7] Sung, J., Jin, S. H., and Saxena, A., 2015. "Robobarista: Object part based transfer of manipulation trajectories from crowd-sourcing in 3d pointclouds". *arXiv preprint arXiv:1504.03071*.

[8] Chilton, L. B., Kim, J., André, P., Cordeiro, F., Landay, J. A., Weld, D. S., Dow, S. P., Miller, R. C., and Zhang, H., 2014. "Frenzy: collaborative data organization for creating conference sessions". In Proceedings of the 32nd annual ACM conference on Human factors in computing systems, ACM, pp. 1255–1264.

[9] Von Ahn, L., 2006. "Games with a purpose". *Computer,* ***39***(6), pp. 92–94.

[10] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M., 2013. "Playing atari with deep reinforcement learning". *arXiv preprint arXiv:1312.5602*.

[11] Jones, D., Schonlau, M., and Welch, W., 1998. "Efficient global optimization of expensive black-box functions". *Journal of Global Optimization,* ***13***(4), pp. 455–492.

[12] Borji, A., and Itti, L., 2013. "Bayesian optimization explains human active search". In Advances in neural information processing systems, pp. 55–63.

[13] Brochu, E., Cora, V. M., and De Freitas, N., 2010. "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning". *arXiv preprint arXiv:1012.2599*.

[14] Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J., 1994. "L-bfgs-b: Fortran subroutines for large scale bound constrained optimization". *Report NAM-11, EECS Department, Northwestern University*.

[15] Davison, A. C., and Hinkley, D. V., 1997. *Bootstrap methods and their application*, Vol. 1. Cambridge university press.

[16] Tombu, M. N., Asplund, C. L., Dux, P. E., Godwin, D., Martin, J. W., and Marois, R., 2011. "A unified attentional bottleneck in the human brain". *Proceedings of the National Academy of Sciences,* ***108***(33), pp. 13426–13431.

[17] Payne, S. J., Duggan, G. B., and Neth, H., 2007. "Discretionary task interleaving: heuristics for time allocation in cognitive foraging.". *Journal of Experimental Psychology: General,* ***136***(3), p. 370.

[18] Metcalfe, J., 2002. "Is study time allocated selectively to a region of proximal learning?". *Journal of Experimental Psychology: General,* ***131***(3), p. 349.

[19] Lee, S., 2005. "Blind source separation and independent component analysis: A review".

[20] Stone, J. V., 2004. *Independent component analysis*. Wiley Online Library.

[21] Bell, A. J., and Sejnowski, T. J., 1997. "The "independent components" of natural scenes are edge filters". *Vision research,* ***37***(23), pp. 3327–3338.

[22] Hui, M., Li, J., Wen, X., Yao, L., and Long, Z., 2011. "An empirical comparison of information-theoretic criteria in estimating the number of independent components of fmri data". *PloS one,* ***6***(12), p. e29274.

[23] Abbeel, P., and Ng, A. Y., 2004. "Apprenticeship learning via inverse reinforcement learning". In Proceedings of the twenty-first international conference on Machine learning, ACM, p. 1.

[24] Ng, A. Y., Russell, S. J., et al., 2000. "Algorithms for inverse reinforcement learning.". In Icml, pp. 663–670.

[25] Ratliff, N. D., Bagnell, J. A., and Zinkevich, M. A., 2006. "Maximum margin planning". In Proceedings of the 23rd international conference on Machine learning, ACM, pp. 729–736.

[26] Syed, U., and Schapire, R. E., 2007. "A game-theoretic approach to apprenticeship learning". In Advances in neural information processing systems, pp. 1449–1456.

[27] Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K., 2008. "Maximum entropy inverse reinforcement learning.". In AAAI, pp. 1433–1438.

[28] Ramachandran, D., and Amir, E., 2007. "Bayesian inverse reinforcement learning". *Urbana,* ***51***, p. 61801.

[29] Homan, M. D., and Gelman, A., 2014. "The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo". *The Journal of Machine Learning Research,* ***15***(1), pp. 1593–1623.