Sequential Preference-Based Optimization

Ian Dewancker **Jakob Bauer**
Uber Advanced Technologies Group
Pittsburgh, PA
{idewancker, jbauer1}@uber.com

Michael McCourt
SigOpt
San Francisco, CA
mccourt@sigopt.com

Abstract

Many real-world engineering problems rely on human preferences to guide their design and optimization. We present PrefOpt, an open source package to simplify sequential optimization tasks that incorporate human preference feedback. Our approach extends an existing latent variable model for binary preferences to allow for observations of equivalent preference from users.

1 Introduction

Defining metrics amenable to optimization can be quite challenging in complex engineering systems. Systems often have components that rely on human perception to judge performance and it can be difficult to develop quantitative measures to capture these perceptual metrics. We are motivated by the problem of tuning the behavior of a motion planning system for a driverless car. In this setting, it may be more practical to employ a tuning strategy that only requires a user to conduct pairwise comparisons of planning system configurations as either more, less or equivalently comfortable.

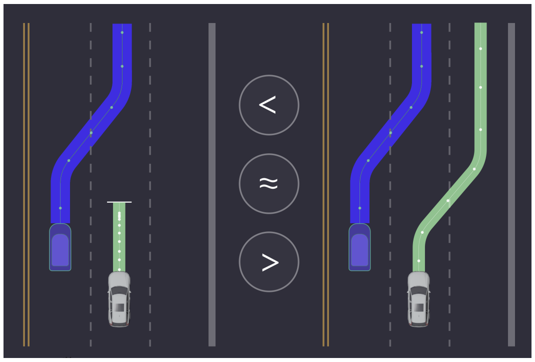


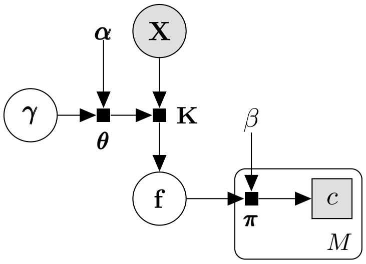
Figure 1: Hypothetical preference-based optimization of a motion planning system. A user is asked to compare a sequence of configurations of a planning system as either worse, better or equivalent. The pairwise comparisons are used to refine the search for the optimal configuration.

By only requiring comparative judgements as feedback, the user is freed from developing a rigorous, single scalar measurement of comfort. Human-in-the-loop preference-based optimization methods [3, 4, 10, 7, 8, 15] have been developed to alleviate some of the requirements of previous model-based optimization methods [2, 11, 14]. These interactive optimization methods allow users to more easily optimize systems that are measured using perceptual metrics or multiple metrics. In this article, we introduce an extension to an existing preference model, allowing for the user to report configurations as equivalently preferable. We also introduce PrefOpt, an open source library that builds on Edward [16] to conduct sequential preference-based optimization using the extended preference model.

Second workshop on Bayesian Deep Learning (NIPS 2017), Long Beach, CA, USA.

2 Preference Model Supporting Ties

Models that relate discrete preference observations to latent function values drawn from Gaussian process priors have been well studied [5, 4, 9, 3]. Previous preference models have required that the user state a binary preference when presented with two options ($\mathbf{x}_i, \mathbf{x}_j \in \Omega \subset \mathbb{R}^D$). In many real-world applications, it may be that even experts occasionally have difficulty discerning two alternatives in terms of absolute preference. To address this concern, we extend the discrete preference observations with a third option: specifying equivalent preference between the two alternatives. Specifically, we adopt a modified Bradley-Terry model that supports ties, or configurations with equivalent preference [13].



$$
\begin{aligned}
\gamma_d &\sim \textbf{Normal}(0, 1)\\
\theta_d &= S(\gamma_d)(\alpha_{d_U} - \alpha_{d_L}) + \alpha_{d_L}\\
K_{i,j} &= rbf(\mathbf{x}_i, \mathbf{x}_j, \boldsymbol{\theta})\\
\mathbf{f} &\sim \textbf{MVNormal}(\mathbf{0}, \mathbf{K})\\
d_m &= \frac{f_m^1 - f_m^2}{\sqrt{2\sigma^2}}\\
z_m^1 &= S(d_m), \quad z_m^2 = 1 - S(d_m)\\
\pi_m^{\prec} &= \frac{z_m^2}{z_m^2 + \beta z_m^1}, \quad \pi_m^{\approx} = \frac{(\beta^2 - 1) z_m^1 z_m^2}{(z_m^1 + \beta z_m^2)(z_m^2 + \beta z_m^1)}\\
c_m &\sim \textbf{Categorical}_3(\pi_m^{\prec},\ \pi_m^{\approx},\ (1 - \pi_m^{\prec} - \pi_m^{\approx}))
\end{aligned}
$$

Figure 2: Graphical model *(left)* and generative process *(right)* of the preference model.

The model draws latent function vectors $\mathbf{f} \in \mathbb{R}^N$ from a Gaussian process prior where each entry corresponds to one of the N unique query points $(\mathbf{X})$ the user has compared so far. The $\boldsymbol{\gamma}$ variables are drawn from normal priors and then transformed to form the length-scales $\boldsymbol{\theta}$ of the covariance function; they, in turn, are used to define $\mathbf{K}$, the covariance matrix. Length scales are bounded by $\boldsymbol{\alpha}$ which, along with $S(x) = \frac{1}{1+e^{-x}}$, is used to define the transformation from $\boldsymbol{\gamma}$ to $\boldsymbol{\theta}$. Here, $rbf(\mathbf{x}_i, \mathbf{x}_j, \boldsymbol{\theta}) = \sigma^2 \exp\Big(-\frac{1}{2}\sum_{d=1}^{D} \frac{1}{\theta_d^2}(x_{id} - x_{jd})^2\Big)$.

The generalized Bradley-Terry model relates the observed discrete preference data c_m to the latent function values (f_m^1, f_m^2) associated with the two points $(\mathbf{x}_m^1, \mathbf{x}_m^2)$ compared by the user during an interactive query. The tie parameter $\beta \geq 1$, is inversely related to the precision with which a user can state a preference [13]. A higher value for β leads to more mass being place in the equivalence bin $(\pi_m^{\approx})$ of the categorical distribution over the three possible preference outcomes for two query points.

2.1 Variational Inference

In place of approximating the posterior with a multivariate Gaussian using the Laplace approximation around a MAP estimate of the latent variables [5, 9], we opt for an approximation that employs variational inference. We set out to approximate $p(\mathbf{z} \mid \mathbf{X}, \mathbf{c})$, the posterior of the latent random variables, where $\mathbf{z} = \{\mathbf{f}, \boldsymbol{\gamma}\}$ is the combined set of latent random variables in our model. We use a mean field approximation strategy to construct our approximating distribution q : a factored set of Gaussians each parametrized by a mean and variance as shown below.

$$
p(\mathbf{z} \mid \mathbf{X}, \mathbf{c}) \approx q(\mathbf{z}\ ;\ \boldsymbol{\lambda}) = \prod_{i=1}^{N} \mathcal{N}(z_i \mid \lambda_{\mu_i},\ \lambda_{\sigma i}) \prod_{k=1}^{D} \mathcal{N}(z_k \mid \lambda_{\mu_k},\ \lambda_{\sigma k})
$$

We rely on the techniques built into Edward [16] to perform the optimization required to recover the variational parameters $\boldsymbol{\lambda}$ that minimize the reverse KL divergence between the true posterior distribution p and the approximating distribution q. In total there will be $2N + 2D$ variational parameters; two for each of the N entries in $\mathbf{f}$ and two for each of the D elements of $\boldsymbol{\gamma}$.

3 Acquisition Function for Preference-Based Optimization

To determine the next point ($\mathbf{x}^n$) to be presented to the user as a comparison point, we adopt a strategy that searches the domain for where the expected improvement of the latent function is highest relative to the current, most preferred point ($\mathbf{x}^b$) [4]. With our approximation $q(\mathbf{z}\ ;\ \boldsymbol{\lambda})$ of the posterior, it is possible to explore the use of an integrated acquisition function, as proposed in [14].

$$\begin{aligned}
\mathbf{k}_* &= [rbf(\mathbf{x}^*, \mathbf{x}_1, \boldsymbol{\theta})\ \cdots\ rbf(\mathbf{x}^*, \mathbf{x}_N, \boldsymbol{\theta})] \\
\mu(\mathbf{x}^*) &= \mathbf{k}_*^\mathsf{T} \mathbf{K}^{-1} \mathbf{f} \\
s^2(\mathbf{x}^*) &= rbf(\mathbf{x}^*, \mathbf{x}^*, \boldsymbol{\theta}) - \mathbf{k}_*^\mathsf{T} \mathbf{K}^{-1} \mathbf{k}_* \\
d &= \mu(\mathbf{x}^*) - f_{best} \\
a_{\mathsf{EI}}(\mathbf{x}^*; \mathbf{z}) &= \begin{cases} d\Phi(\frac{d}{s(\mathbf{x}^*)}) + s(\mathbf{x}^*)\phi(\frac{d}{s(\mathbf{x}^*)}), & \text{if } s(\mathbf{x}^*) > 0 \\ 0, & \text{if } s(\mathbf{x}^*) = 0 \end{cases} \\
\mathbf{x}^n &= \arg\max_{\mathbf{x}^*} \int a_{\mathsf{EI}}(\mathbf{x}^*; \mathbf{z}) q(\mathbf{z}\ ;\ \boldsymbol{\lambda}) d\mathbf{z}
\end{aligned}$$

Here $\Phi(\cdot)$ and $\phi(\cdot)$ denote the CDF and PDF of the standard normal distribution, respectively. The value f_{best} is the latent function value associated with the currently most preferred configuration $\mathbf{x}^b$. If the user is always asked to compare $\mathbf{x}^n$ against the current most preferred point $\mathbf{x}^b$, the most preferred point can be updated as a result of this comparison.

4 PrefOpt Software

The goal of the PrefOpt software package is to provide a simple interface for conducting human-in-the-loop, preference-based optimization tasks. The user is required to initially define a bounding box to represent the search domain of the parameters of interest. The optimization proceeds by iteratively proposing two query points for the user to compare 3. The preference order of the two query points is recorded and the underlying latent preference model is updated. Currently only two query points will be returned each iteration for the user to compare, however, a strategy of returning larger batches of comparison points could be pursued [3, 6].

```python
import prefopt
# define the domain of the search space
bounding_box = [[-5.0, 5.0], [0.0, 10.0]]
exp = prefopt.PreferenceExperiment(bounding_box)

for i in xrange(1,N):
    # search for the next points to compare
    X = exp.find_next()
    # get user preference : -1 denotes x1 < x2,  0 denotes x1 = x2, 1 denotes x1 > x2
    order = get_user_pref(X[0], X[1])
    # update model with new preference observation
    exp.prefer(X[0], X[1], order)
```

Figure 3: Example usage of PrefOpt to conduct preference-based optimization.

The PrefOpt library will hopefully facilitate future investigations into preference-based optimization and similar interactive optimization techniques. In particular, it is exciting to consider applications in the self-driving space where metrics are often difficult to specify or intrinsically linked to human perception. Other interesting avenues of future work could include investigating strategies to propose multiple comparison query points at each iteration, evaluating new acquisition functions, or comparing the effectiveness of non-GP based methods for capturing user preference data.

The open source PrefOpt package will be hosted at `https://github.com/prefopt/prefopt`.

5 Experimental Results

To measure the effectiveness of our preference-based optimization method, we considered its efficiency in minimizing synthetic test functions using only pairwise comparative observations. At each iteration a query point ($\mathbf{x}^n$) is selected and compared against the current best point ($\mathbf{x}^b$) using a test function (f_{test}). The discrete preference observations were simulated in the following way :

$$\text{pref}(\mathbf{x}_1, \mathbf{x}_2) = \begin{cases} \mathbf{x}_1 \approx \mathbf{x}_2, & \text{if } |f_{\text{test}}(\mathbf{x}_1) - f_{\text{test}}(\mathbf{x}_2)| \leq \epsilon \\ \mathbf{x}_1 \succ \mathbf{x}_2, & \text{else if } f_{\text{test}}(\mathbf{x}_1) > f_{\text{test}}(\mathbf{x}_2) \\ \mathbf{x}_1 \prec \mathbf{x}_2, & \text{otherwise} \end{cases}$$

We initialized the search with $2D + 1$ samples from a latin hypercube sequence, where D is the number of input parameters of the test function. We repeated each experiment 10 times and reported the median and interquartile range of the best seen objective value after each iteration. We included two settings of the tolerance parameter $\epsilon = \{0.001, 0.1\}$ and evaluated the expected improvement acquisition function, a "pure exploration" (a_{PE}) acquisition function [6, 4] and random search [1].

$$a_{\mathsf{PE}}(\mathbf{x}^*; \mathbf{z}) = rbf(\mathbf{x}^*, \mathbf{x}^*, \boldsymbol{\theta}) - \mathbf{k}_*^{\mathsf{T}} \mathbf{K}^{-1} \mathbf{k}_*$$

$$\mathbf{x}^n = \underset{\mathbf{x}^*}{\arg\max} \int a_{\mathsf{PE}}(\mathbf{x}^*; \mathbf{z}) q(\mathbf{z}\ ;\ \boldsymbol{\lambda}) d\mathbf{z}$$

We evaluated PrefOpt using several optimization test functions [12]. We observed the expected improvement acquisition function outperforming the baselines across the selected test functions 4. Our approach seems work well under both settings of the tolerance parameter which is encouraging as it suggests our method might be useful even to users with less than expert ability to discern quality.

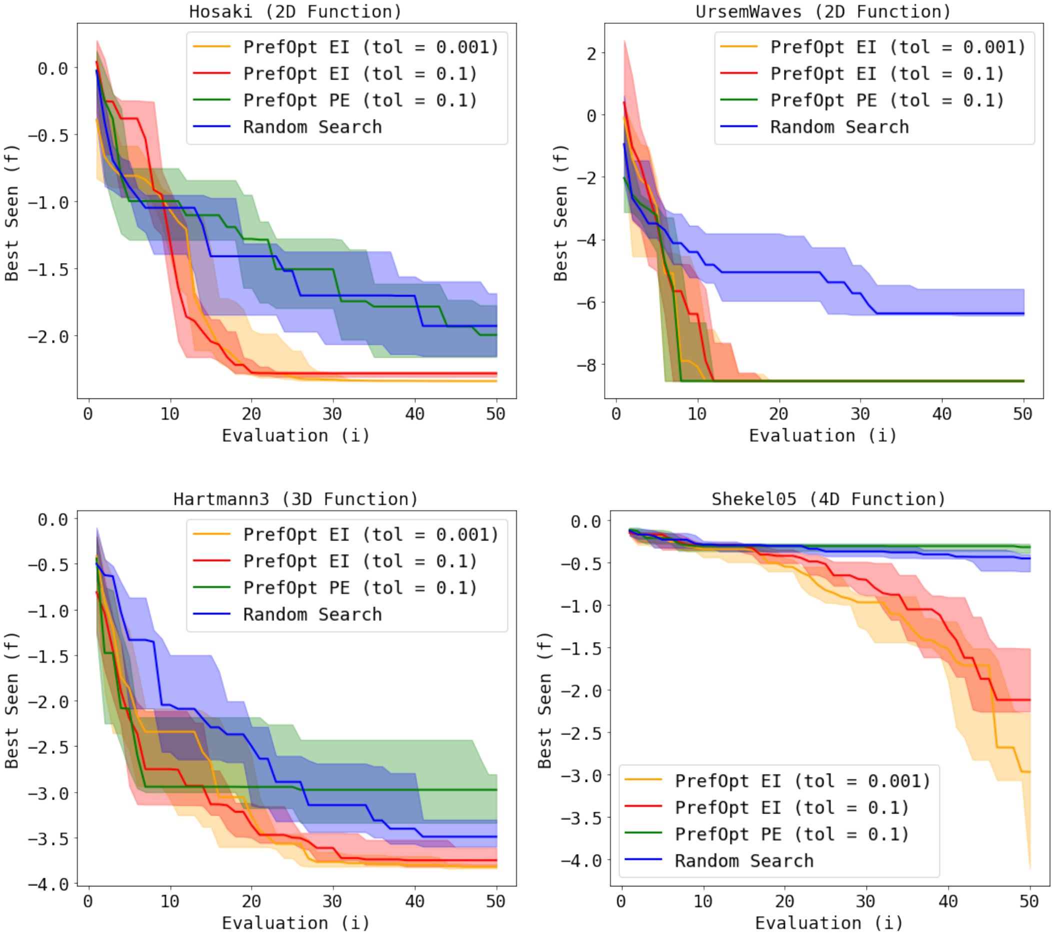


Figure 4: Summary of PrefOpt optimization traces on a collection of synthetic test problems.

References

[1] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[2] James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, pages 2546–2554, 2011.

[3] Eric Brochu, Tyson Brochu, and Nando de Freitas. A bayesian interactive optimization approach to procedural animation design. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 103–112. Eurographics Association, 2010.

[4] Eric Brochu, Nando de Freitas, and Abhijeet Ghosh. Active preference learning with discrete choice data. In *Advances in Neural Information Processing Systems*, pages 409–416, 2008.

[5] Wei Chu and Zoubin Ghahramani. Preference learning with gaussian processes. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 137–144. ACM, 2005.

[6] Emile Contal, David Buffoni, Alexandre Robicquet, and Nicolas Vayatis. Parallel gaussian process optimization with upper confidence bound and pure exploration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 225–240. Springer, 2013.

[7] Ian Dewancker, Michael McCourt, and Samuel Ainsworth. Interactive preference learning of utility functions for multi-objective optimization. In *NIPS Future of Interactive Learning Machines Workshop*, 2016.

[8] Javier Gonzalez, Zhenwen Dai, Andreas Damianou, and Neil D Lawrence. Preferential bayesian optimization. *arXiv preprint arXiv:1704.03651*, 2017.

[9] Shengbo Guo, Scott Sanner, and Edwin V Bonilla. Gaussian process preference elicitation. In *Advances in Neural Information Processing Systems*, pages 262–270, 2010.

[10] Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Jose M Hernández-Lobato. Collaborative gaussian processes for preference learning. In *Advances in Neural Information Processing Systems*, pages 2096–2104, 2012.

[11] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization*, pages 507–523. Springer, 2011.

[12] Michael McCourt. Optimization Test Functions. `https://github.com/sigopt/evalset`, 2016.

[13] PV Rao and Lawrence L Kupper. Ties in paired-comparison experiments: A generalization of the bradley-terry model. *Journal of the American Statistical Association*, 62(317):194–204, 1967.

[14] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, pages 2951–2959, 2012.

[15] Nitish Thatte, Helei Duan, and Hartmut Geyer. A sample-efficient black-box optimizer to train policies for human-in-the-loop systems with user preferences. *IEEE Robotics and Automation Letters*, 2(2):993–1000, 2017.

[16] Dustin Tran, Alp Kucukelbir, Adji B. Dieng, Maja Rudolph, Dawen Liang, and David M. Blei. Edward: A library for probabilistic modeling, inference, and criticism. *arXiv preprint arXiv:1610.09787*, 2016.