Sequential Preference-Based Optimization

Ian Dewancker[1], Jakob Bauer[1], Michael McCourt[2]

[1]Uber Advanced Technologies Group, [2]SigOpt

Introduction

Many real-world engineering problems rely on human preferences to guide their design and optimization. We present PrefOpt, an open source package built on Edward, 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.

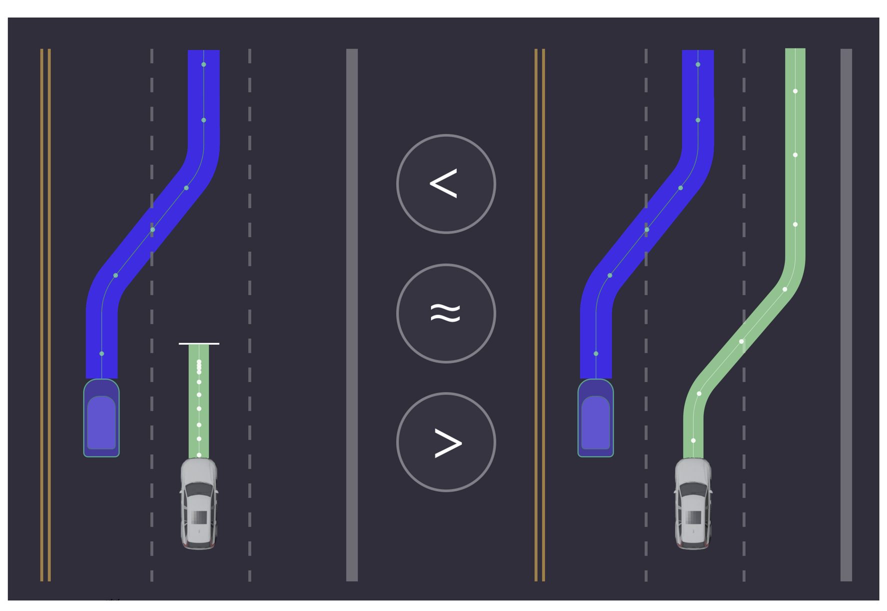
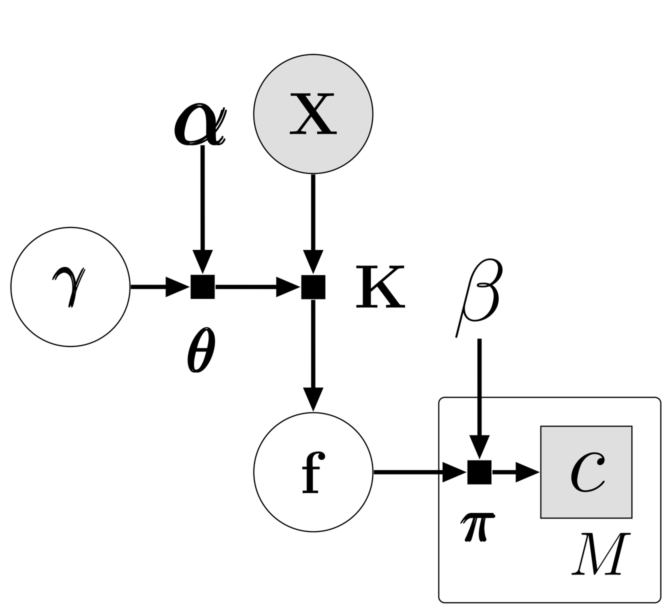


Figure 1: Hypothetical preference-based optimization of a motion planning system. A user is asked to compare a sequence of configurations of a planning system as either more, less or equivalently comfortable. The pairwise comparisons are used to refine the search for the optimal configuration.

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.

Preference Model Supporting Ties

Models that relate discrete preference observations to latent function values drawn from Gaussian process priors have been well studied. 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.



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

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

Here, $rbf(\mathbf{x}_i, \mathbf{x}_j, \boldsymbol{\theta}) = \sigma^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}\frac{1}{\theta_d^2}(x_{id} - x_{jd})^2\right)$ and $S(x) = \frac{1}{1+e^{-x}}$. 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 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.

PrefOpt Software

To determine the next point ($\mathbf{x}^n$) to be presented to the user as a comparison point, we adopt a strategy that searches for where the expected improvement of the latent function is highest relative to the current, most preferred point ($\mathbf{x}^b$).

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

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 ($\mathbf{x}^b$). The goal of the PrefOpt software package is to provide a simple interface for conducting human-in-the-loop, preference-based optimization tasks. The open source PrefOpt package will be hosted at `https://github.com/prefopt/prefopt`.

```python
import prefopt
bounding_box = [[-5.0, 5.0], [0.0, 10.0]]
exp = prefopt.PreferenceExperiment(bounding_box)
for i in xrange(1,N):
    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])
    exp.prefer(X[0], X[1], order)
```

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

Experimental Results

We considered our method's 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 preference observations were simulated as follows :

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

We 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 (ϵ) and compared the expected improvement acquisition function, a "pure exploration" acquisition function and random search.

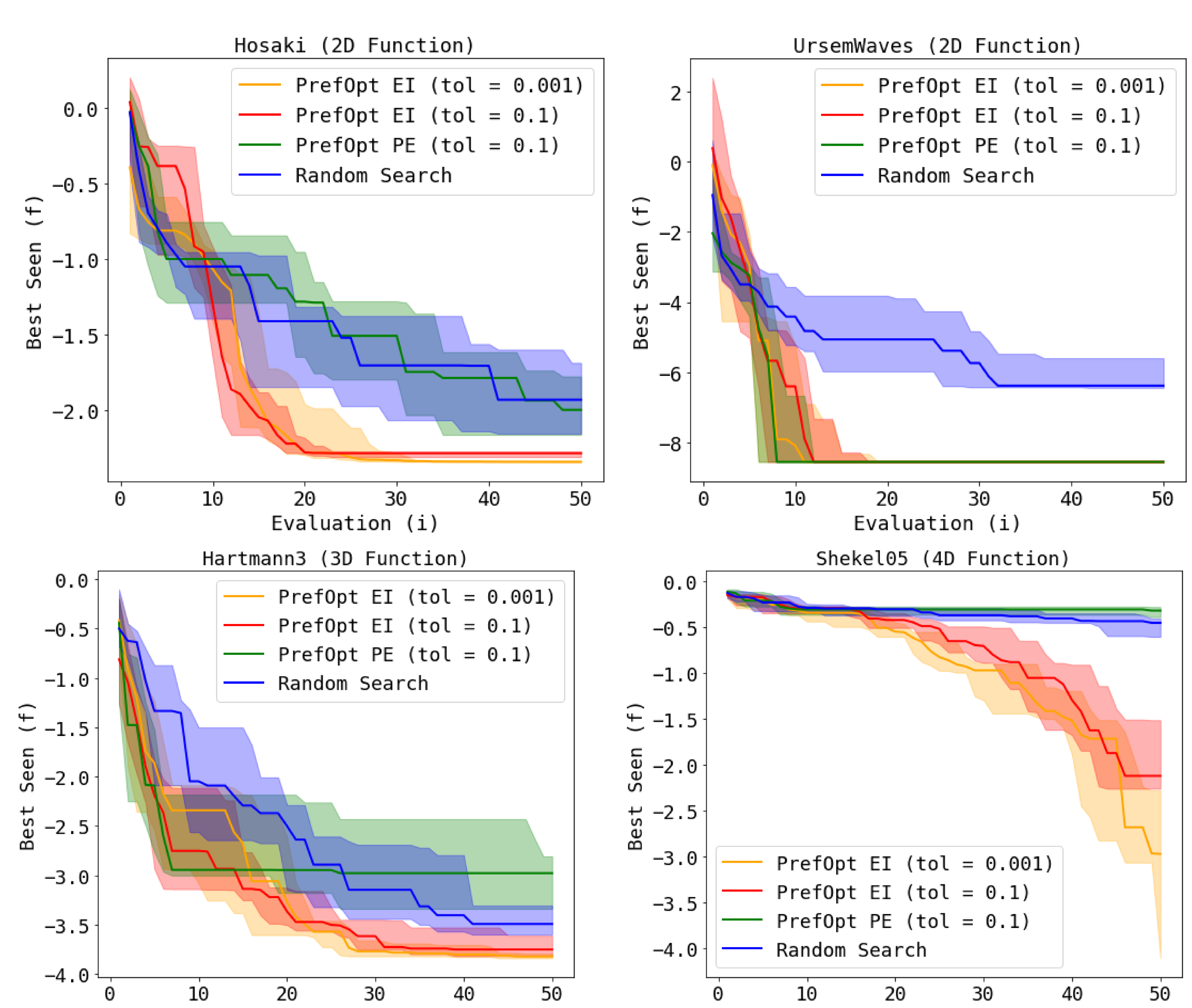


Figure 4: Summary of PrefOpt optimization traces on a collection of synthetic test problems.