Sequential Preference-Based Optimization

Ian Dewancker

ian.dewancker@gmail.com

Motivation

Sequential model-based (Bayesian) optimization techniques have demonstrated success as black box optimizers in low-dim spaces (hyperparams for ML models)

Motivation

Sequential model-based (Bayesian) optimization techniques have demonstrated success as black box optimizers in low-dim spaces (hyperparams for ML models)

$$\mathbf{x}_{opt} = \arg\max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$$

$$f(\boldsymbol{\lambda}) = \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}(\boldsymbol{\lambda}, \mathcal{D}_{\text{train}}^{(i)}, \mathcal{D}_{\text{valid}}^{(i)})$$

Motivation

Sequential model-based (Bayesian) optimization techniques have demonstrated success as black box optimizers in low-dim spaces (hyperparams for ML models)

$$\mathbf{x}_{opt} = \arg\max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$$

$$f(\boldsymbol{\lambda}) = \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}(\boldsymbol{\lambda}, \mathcal{D}_{\text{train}}^{(i)}, \mathcal{D}_{\text{valid}}^{(i)})$$

Open Source : SMAC, HyperOpt, Spearmint, MOE

Companies : Whetlab, SigOpt

Motivation

Problem : *Real-world scalar valued objectives can be hard to obtain!*

Motivation

Problem : *Real-world scalar valued objectives can be hard to obtain!*

- Multi-objective problems
- Objective fundamentally relates to human perception
- Physical system is difficult or costly to instrument

Motivation

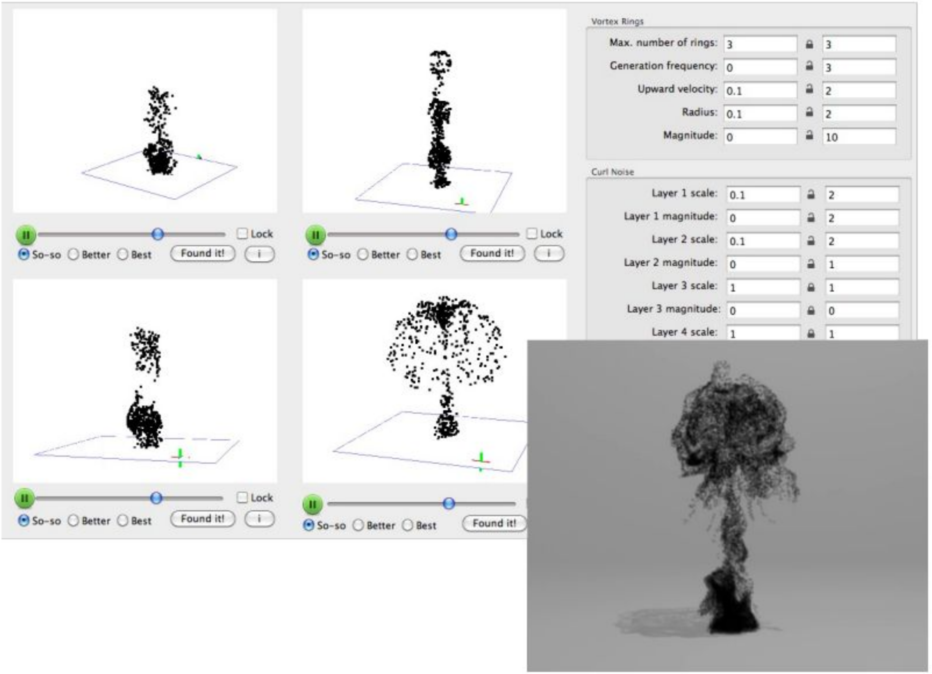
Problem : *Real-world scalar valued objectives can be hard to obtain!*

- Multi-objective problems
- Objective fundamentally relates to human perception
- Physical system is difficult or costly to instrument

Approach : Interactive procedure to query user and guide optimization based on preference observations (bring human back into the loop!)

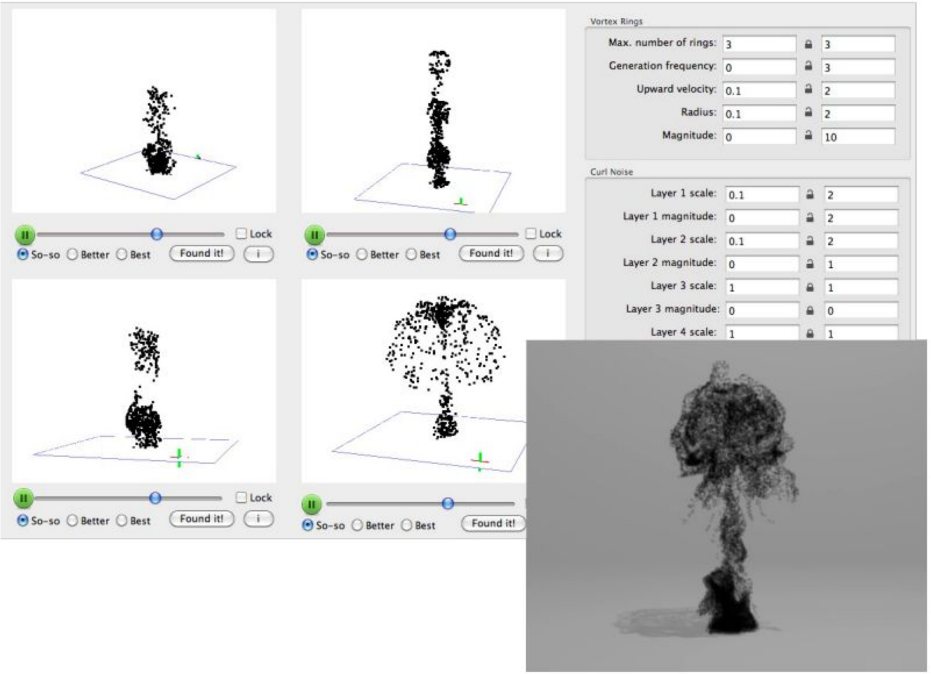
Prior Work

Tuning procedural animation parameters for particle simulations [1]



Prior Work

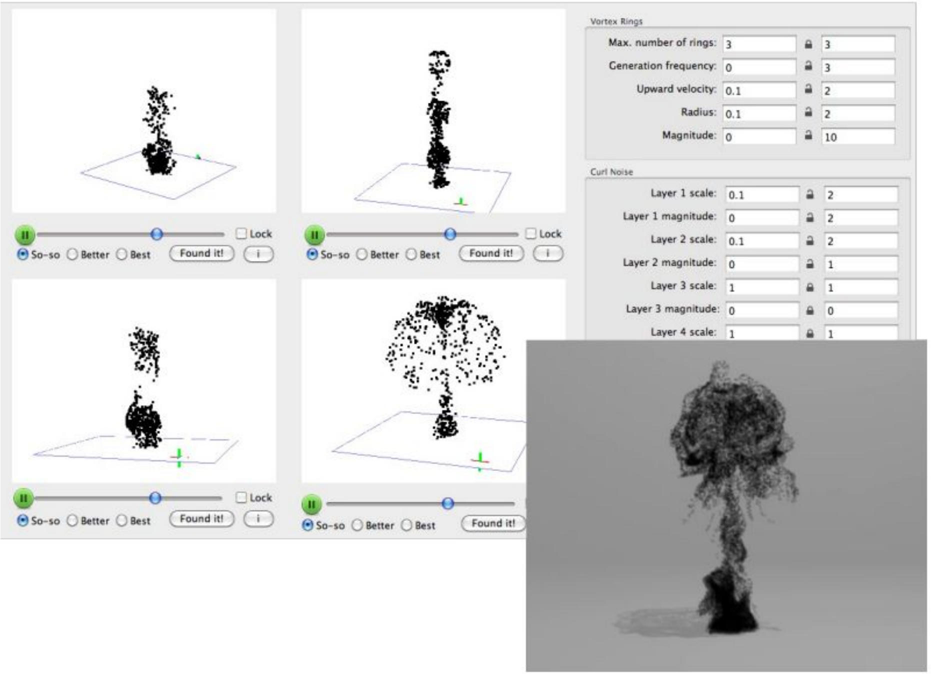
Tuning procedural animation parameters for particle simulations [1]



1. Ask user to rank several parametrizations of a particle animation

Prior Work

Tuning procedural animation parameters for particle simulations [1]

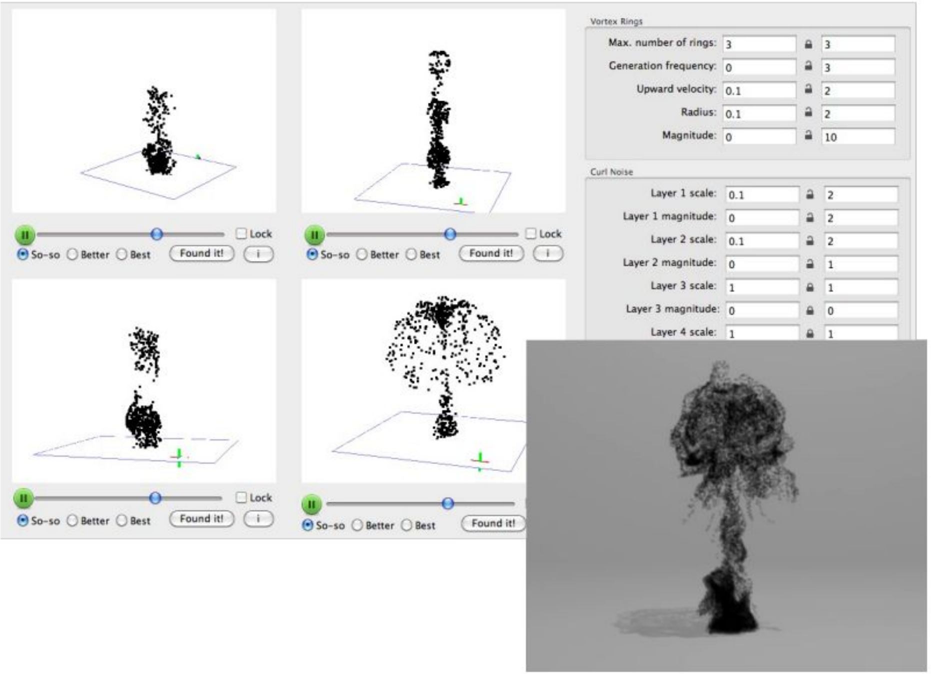


1. Ask user to rank several parametrizations of a particle animation

2. Update latent utility model that best adheres to observed preference data

Prior Work

Tuning procedural animation parameters for particle simulations [1]



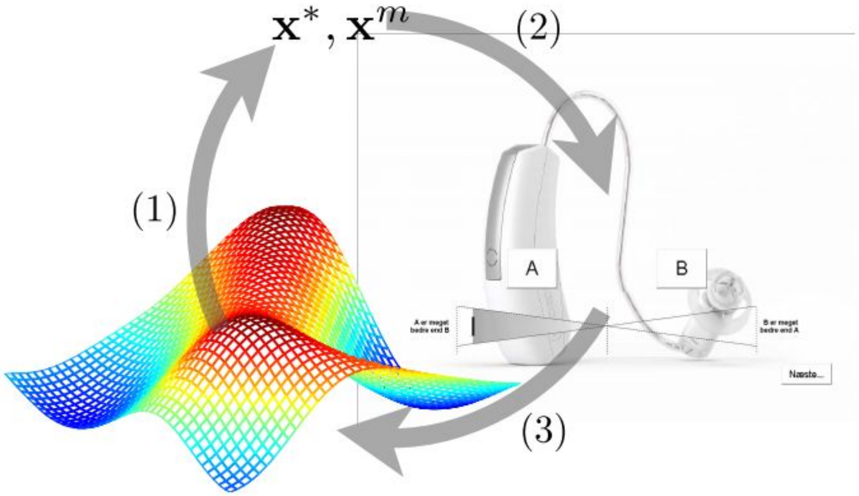
1. Ask user to rank several parametrizations of a particle animation

2. Update latent utility model that best adheres to observed preference data

3. Suggest new parameterizations that are expected to improve over best seen using latent utility

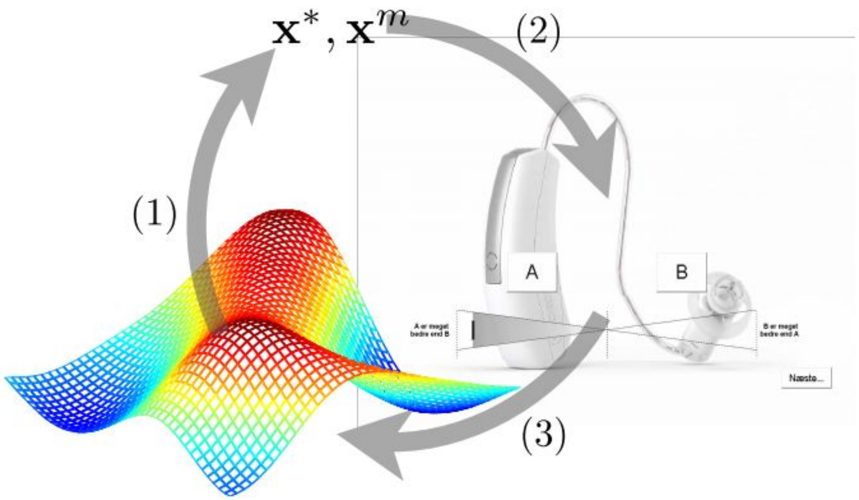
Prior Work

Tuning frequency-dependent hearing thresholds for hearing aid personalization [2]



Prior Work

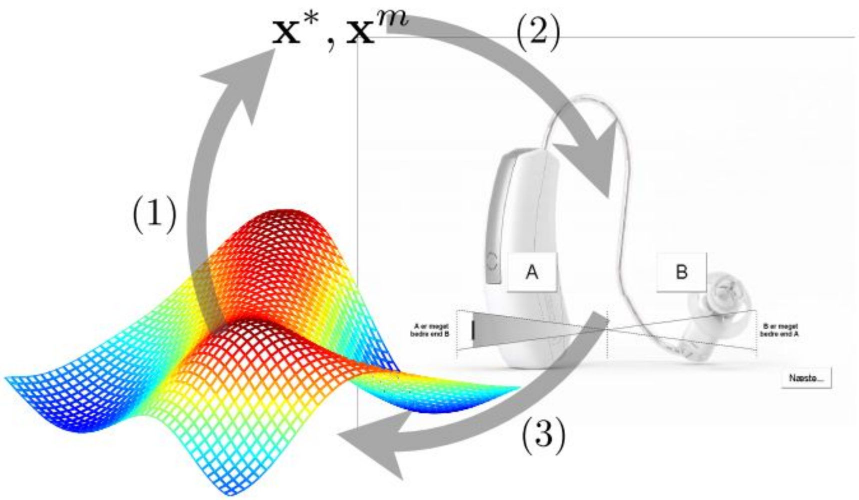
Tuning frequency-dependent hearing thresholds for hearing aid personalization [2]



1. Ask user to compare two configurations of hearing aid parameters in terms of subjective preference

Prior Work

Tuning frequency-dependent hearing thresholds for hearing aid personalization [2]

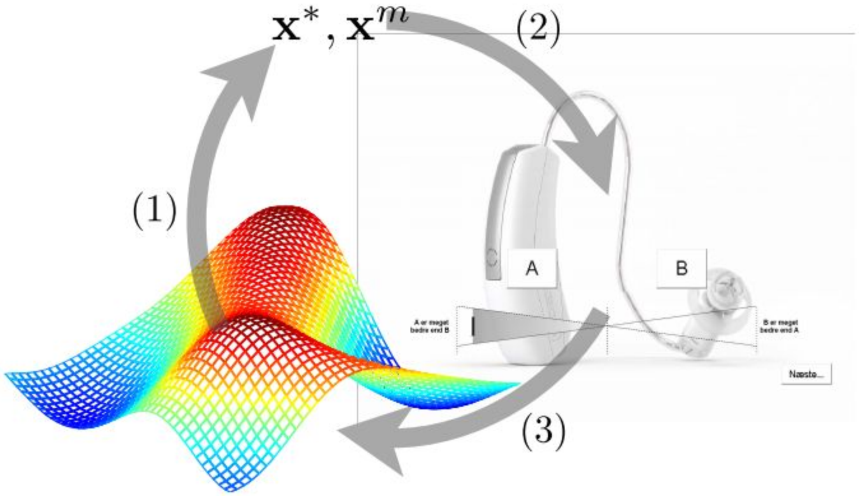


1. Ask user to compare two configurations of hearing aid parameters in terms of subjective preference

2. Update latent utility model that best adheres to observed preference data

Prior Work

Tuning frequency-dependent hearing thresholds for hearing aid personalization [2]



1. Ask user to compare two configurations of hearing aid parameters in terms of subjective preference

2. Update latent utility model that best adheres to observed preference data

3. Suggest next configuration that is expected to improve over best seen using latent utility

Prior Work

Tuning feedback gains of a neuromuscular walking model [3]

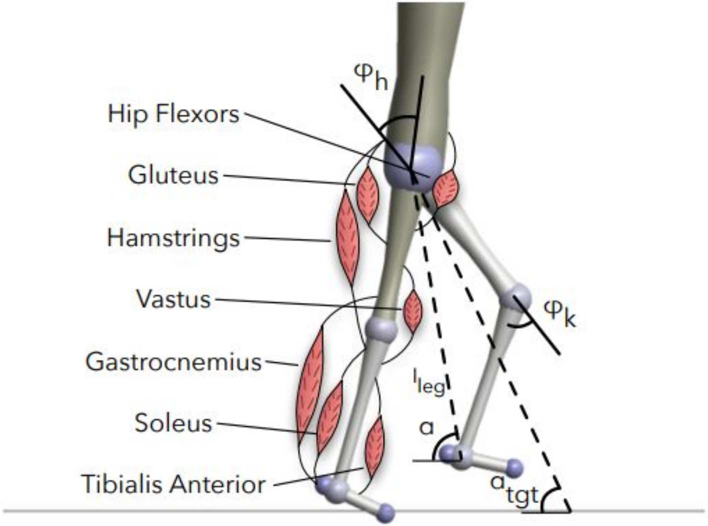


Fig. 2: Unimpaired human walking model with labeled muscles and definitions of hip and knee angles, leg length, and current and target swing leg angles.

Prior Work

Tuning feedback gains of a neuromuscular walking model [3]

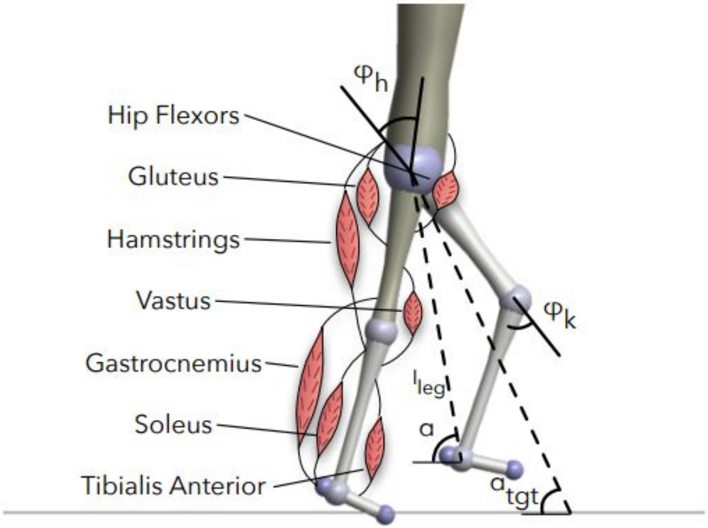


Fig. 2: Unimpaired human walking model with labeled muscles and definitions of hip and knee angles, leg length, and current and target swing leg angles.

1. Compare two configurations of feedback gains for subset of muscle actuators and report preference based on cost of transport and distance walked in 20 sec

Prior Work

Tuning feedback gains of a neuromuscular walking model [3]

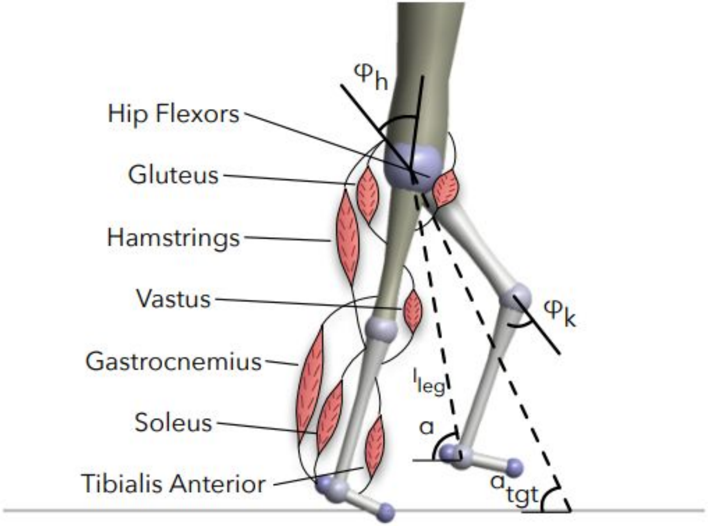


Fig. 2: Unimpaired human walking model with labeled muscles and definitions of hip and knee angles, leg length, and current and target swing leg angles.

1. Compare two configurations of feedback gains for subset of muscle actuators and report preference based on cost of transport and distance walked in 20 sec

2. Update latent utility model that best adheres to observed preference data

Prior Work

Tuning feedback gains of a neuromuscular walking model [3]

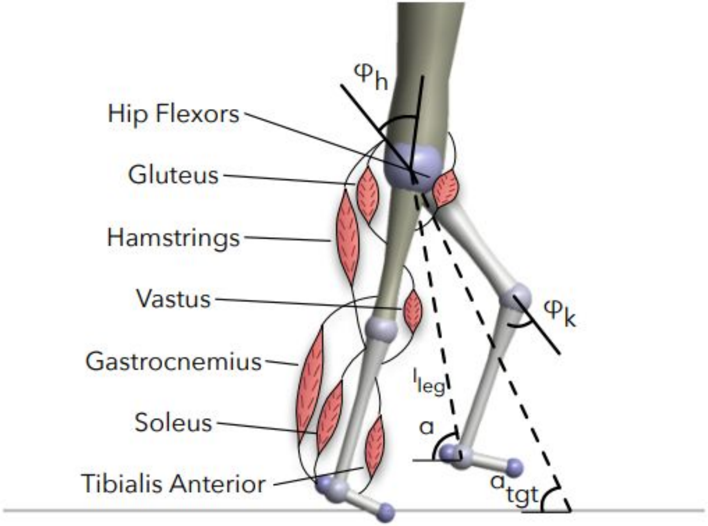


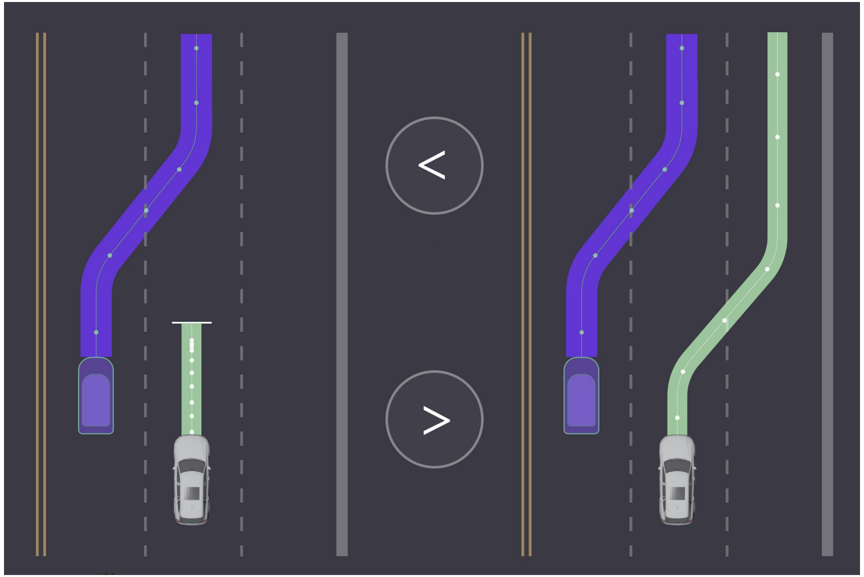
Fig. 2: Unimpaired human walking model with labeled muscles and definitions of hip and knee angles, leg length, and current and target swing leg angles.

1. Compare two configurations of feedback gains for subset of muscle actuators and report preference based on cost of transport and distance walked in 20 sec

2. Update latent utility model that best adheres to observed preference data

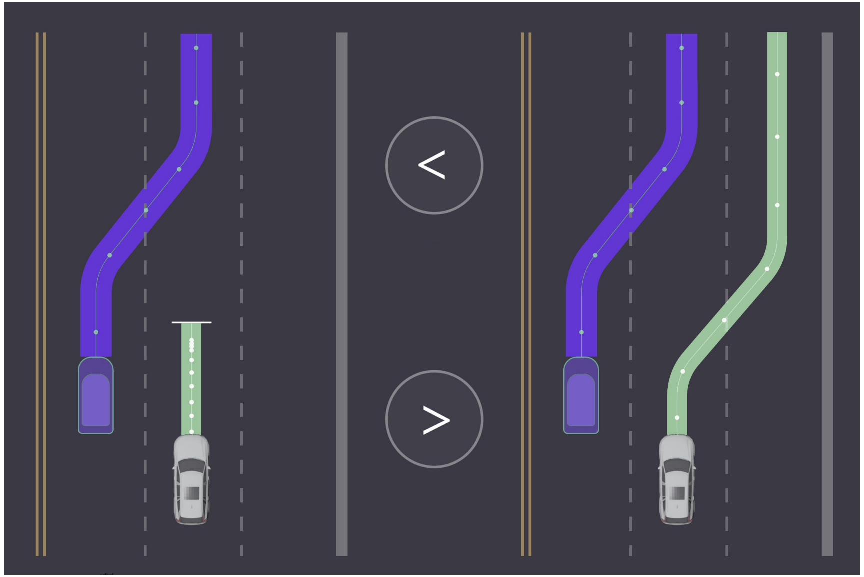
3. Suggest next configuration that is expected to improve over best seen using latent utility function

Motivation



Lots of complex engineering systems with many magic numbers (gains, decision thresholds, noise model parameters, etc)

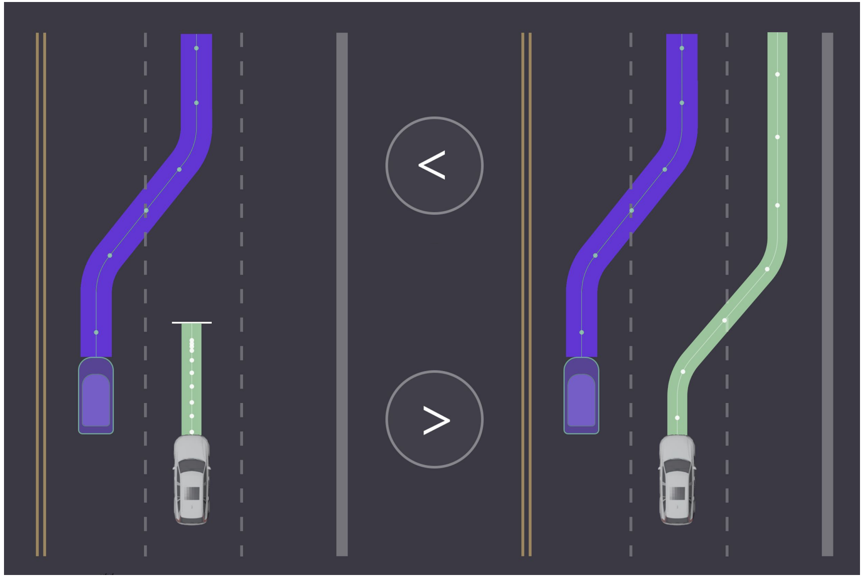
Motivation



Lots of complex engineering systems with many magic numbers (gains, decision thresholds, noise model parameters, etc)

Often even experts have trouble discerning absolute quality preference between two alternative configurations

Motivation

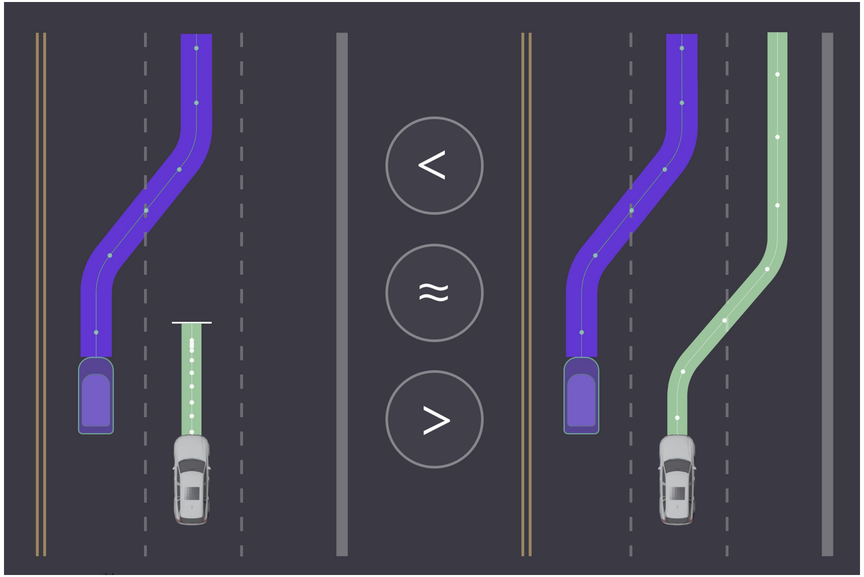


Lots of complex engineering systems with many magic numbers (gains, decision thresholds, noise model parameters, etc)

Often even experts have trouble discerning absolute quality preference between two alternative configurations

Previous work has supported only binary preference observations

Motivation



Lots of complex engineering systems with many magic numbers (gains, decision thresholds, noise model parameters, etc)

Often even experts have trouble discerning absolute quality preference between two alternative configurations

Previous work has supported only binary preference observations

Can we build on previous ideas to support observations having **equivalent** preference?

Goals

System for design of experiments where only feedback is preference observations

Goals

System for design of experiments where only feedback is preference observations

1. Query user for preference information based on set of configuration proposals

Goals

System for design of experiments where only feedback is preference observations

1. Query user for preference information based on set of configuration proposals
2. Infer a latent utility function (utility values for every observation) that best adheres to observed preferences (allowing for equivalent preferences)

Goals

System for design of experiments where only feedback is preference observations

1. Query user for preference information based on set of configuration proposals
2. Infer a latent utility function (utility values for every observation) that best adheres to observed preferences (allowing for equivalent preferences)
3. Search latent utility function for new configurations that are expected to have high utility to compare to current best

Goals

System for design of experiments where only feedback is preference observations

1. Query user for preference information based on set of configuration proposals
2. Infer a latent utility function (utility values for every observation) that best adheres to observed preferences (allowing for equivalent preferences)
3. Search latent utility function for new configurations that are expected to have high utility to compare to current best
4. Repeat 1 with new configurations

Goals

System for design of experiments where only feedback is preference observations

1. Query user for preference information based on set of configuration proposals
2. Infer a latent utility function (utility values for every observation) that best adheres to observed preferences (allowing for equivalent preferences)
3. Search latent utility function for new configurations that are expected to have high utility to compare to current best
4. Repeat 1 with new configurations

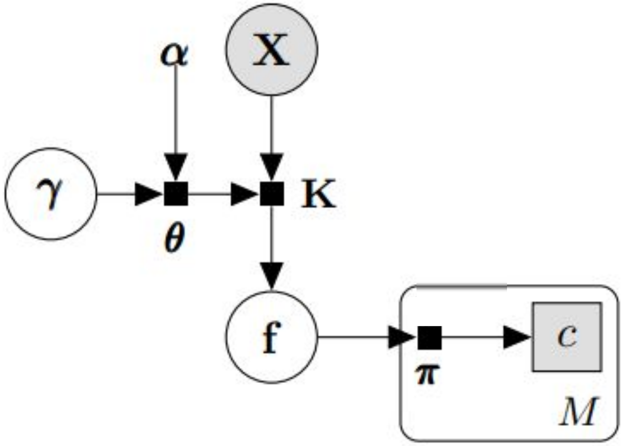
Simple software interface to allow for extensions and rapid experiments

Gaussian Process Model Binary Preferences

M preference observations (**c**) are related to N unique query points (**X**) via a GP prior over latent function vectors (**f**) and a Bradley-Terry preference model [1,2,3]

Gaussian Process Model Binary Preferences

M preference observations (**c**) are related to N unique query points (**X**) via a GP prior over latent function vectors (**f**) and a Bradley-Terry preference model [1,2,3]



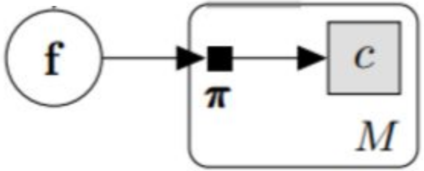
$$
\begin{aligned}
\gamma_d &\sim \mathbf{Normal}(0,1) \\
\theta_d &= S(\gamma_d)(\alpha_{d_U} - \alpha_{d_L}) + \alpha_{d_L} \\
K_{i,j} &= rbf(\mathbf{x}_i, \mathbf{x}_j, \boldsymbol{\theta}) \\
\mathbf{f} &\sim \mathbf{MVNormal}(\mathbf{0}, \mathbf{K}) \\
d_m &= \frac{f_m^1 - f_m^2}{\sqrt{2\sigma^2}}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

$$
rbf(\mathbf{x}_i, \mathbf{x}_j, \boldsymbol{\theta}) = \sigma^2 \exp\left(-\tfrac{1}{2}\sum_{d=1}^{D} \tfrac{1}{\theta_d^2}(x_{i\,d} - x_{j\,d})^2\right) \qquad S(x) = \tfrac{1}{1+e^{-x}}
$$

Bradley-Terry Model Extension for Ties

The Bradley-Terry model relates the comparison points' latent function values to discrete binary preferences.

Binary



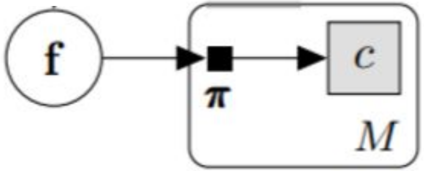
$$d_m = \frac{f_m^1 - f_m^2}{\sqrt{2\sigma^2}}$$

$$\pi_m^{\succ} = S(d_m), \quad \pi_m^{\prec} = 1 - S(d_m)$$

Bradley-Terry Model Extension for Ties

The Bradley-Terry model relates the comparison points' latent function values to discrete binary preferences.

Binary

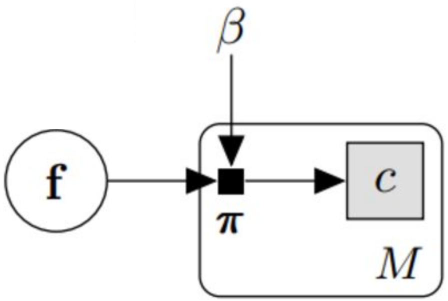


$$d_m = \frac{f_m^1 - f_m^2}{\sqrt{2\sigma^2}}$$

$$\pi_m^{\succ} = S(d_m), \quad \pi_m^{\prec} = 1 - S(d_m)$$

An extension to the BT model produces preference probabilities over 3 classes [4]

Ternary



$$d_m = \frac{f_m^1 - f_m^2}{\sqrt{2\sigma^2}}$$

$$z_m^1 = S(d_m), \quad z_m^2 = 1 - S(d_m)$$

$$\pi_m^{\prec} = \frac{z_m^2}{z_m^2 + \beta z_m^1}, \quad \pi_m^{\approx} = \frac{(\beta^2 - 1) z_m^1 z_m^2}{(z_m^1 + \beta z_m^2)(z_m^2 + \beta z_m^1)}$$

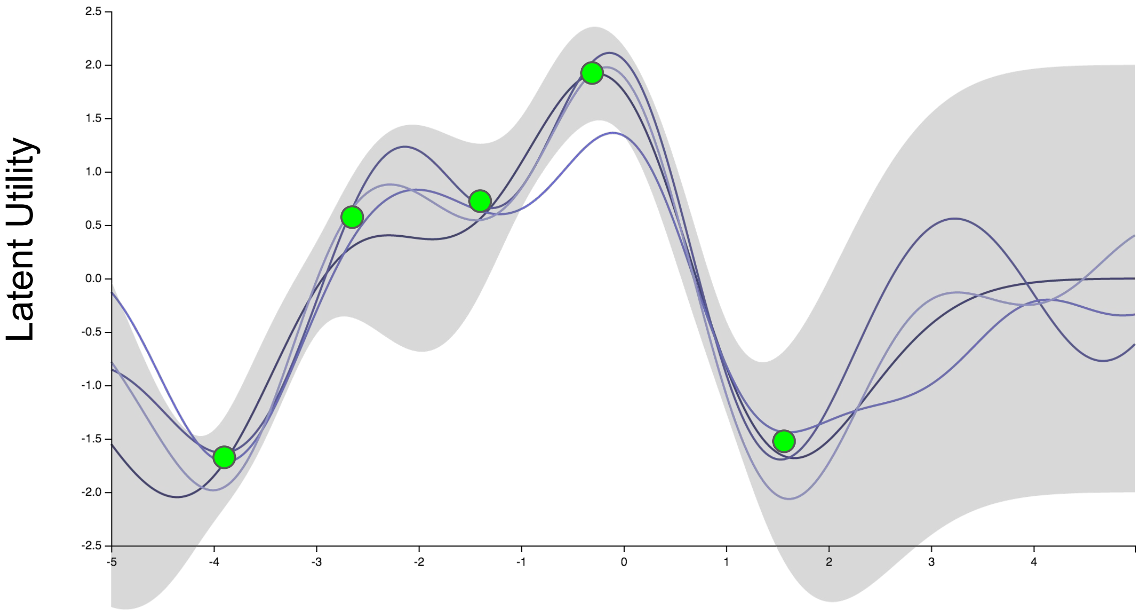
$$\pi_m^{\succ} = (1 - \pi_m^{\prec} - \pi_m^{\approx})$$

Bradley-Terry Model Extension for Ties

The Bradley-Terry model relates the comparison points’ latent function values to discrete binary preferences.

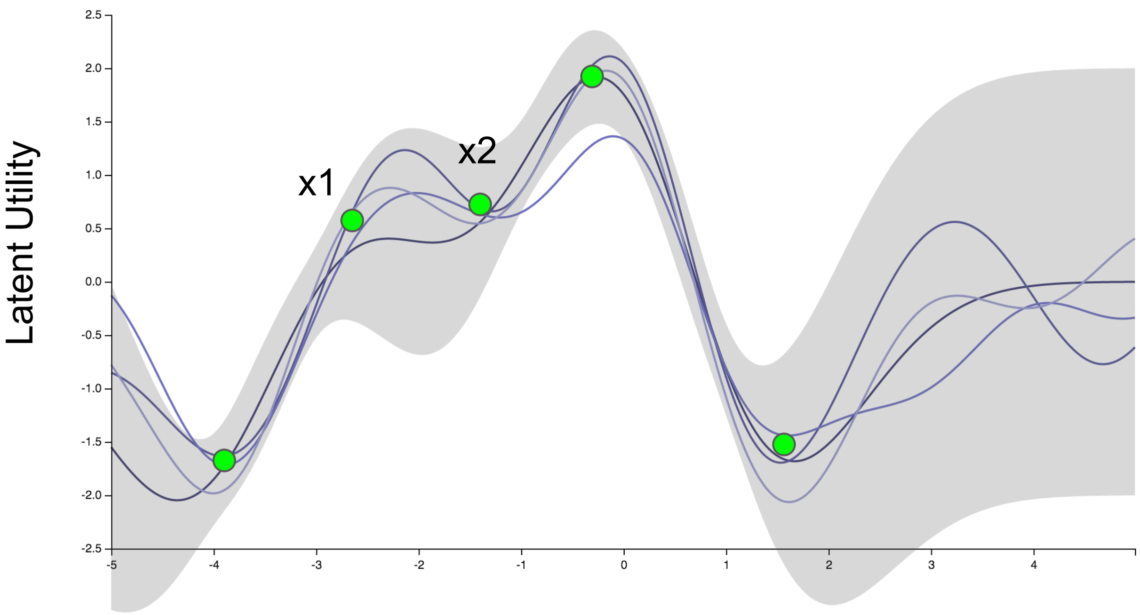
Bradley-Terry Model Extension for Ties

The Bradley-Terry model relates the comparison points' latent function values to discrete binary preferences.



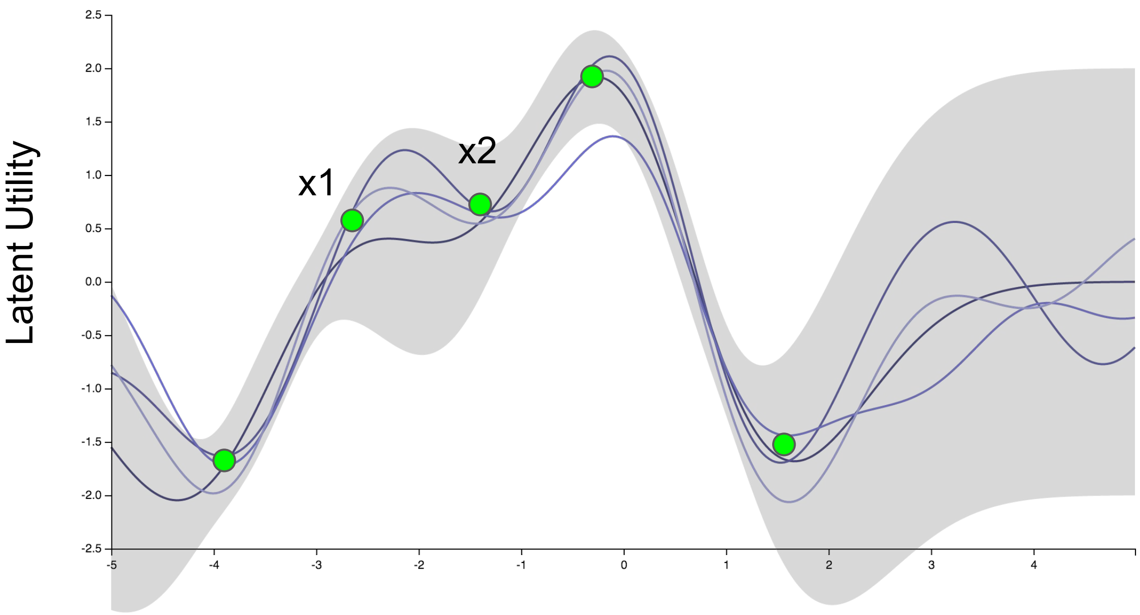
Bradley-Terry Model Extension for Ties

The Bradley-Terry model relates the comparison points' latent function values to discrete binary preferences.



Bradley-Terry Model Extension for Ties

The Bradley-Terry model relates the comparison points' latent function values to discrete binary preferences.

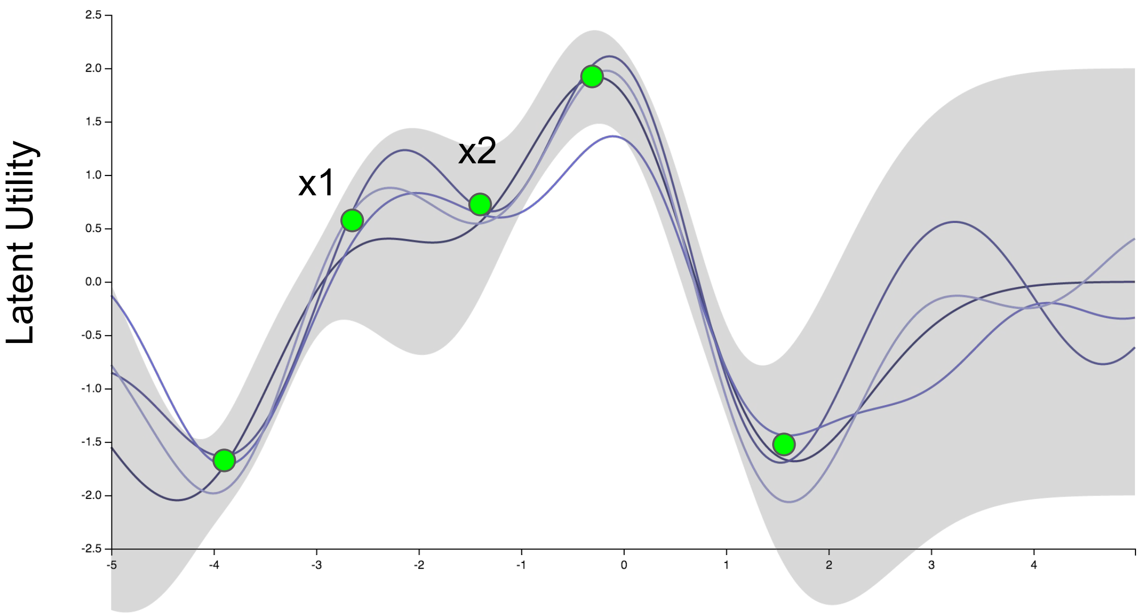


x1 = -2.9 x2 = -1.2

f1 = 0.5 f2 = 0.6

Bradley-Terry Model Extension for Ties

The Bradley-Terry model relates the comparison points' latent function values to discrete binary preferences.

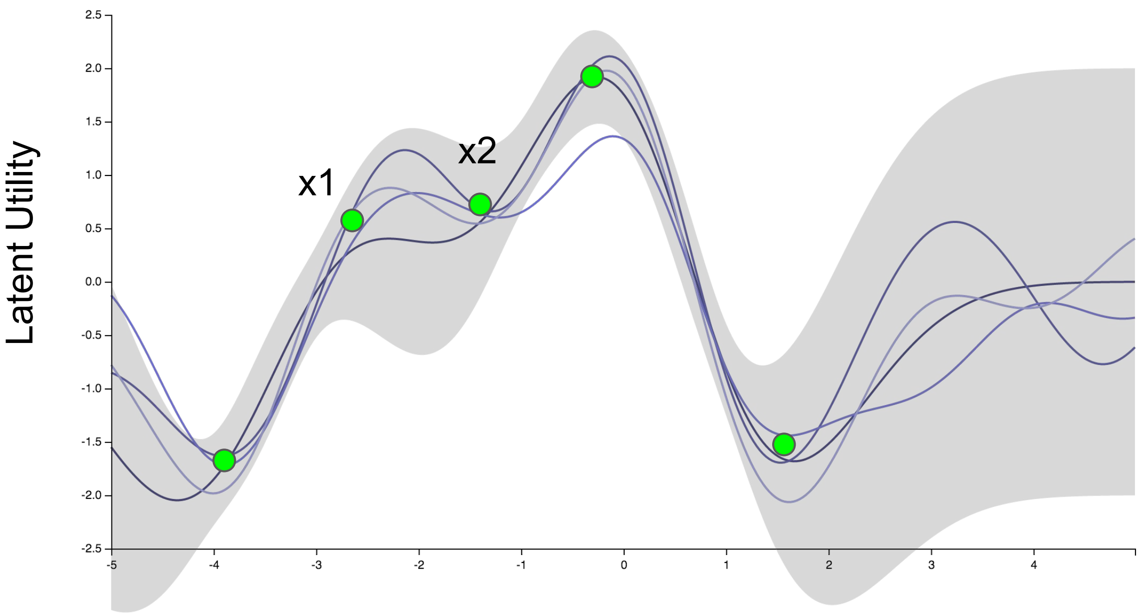


x1 = -2.9   x2 = -1.2
f1 = 0.5   f2 = 0.6

d = (0.5 - 0.6) / (2.0)
d = -0.05

Bradley-Terry Model Extension for Ties

The Bradley-Terry model relates the comparison points' latent function values to discrete binary preferences.



x1 = -2.9 x2 = -1.2
f1 = 0.5 f2 = 0.6

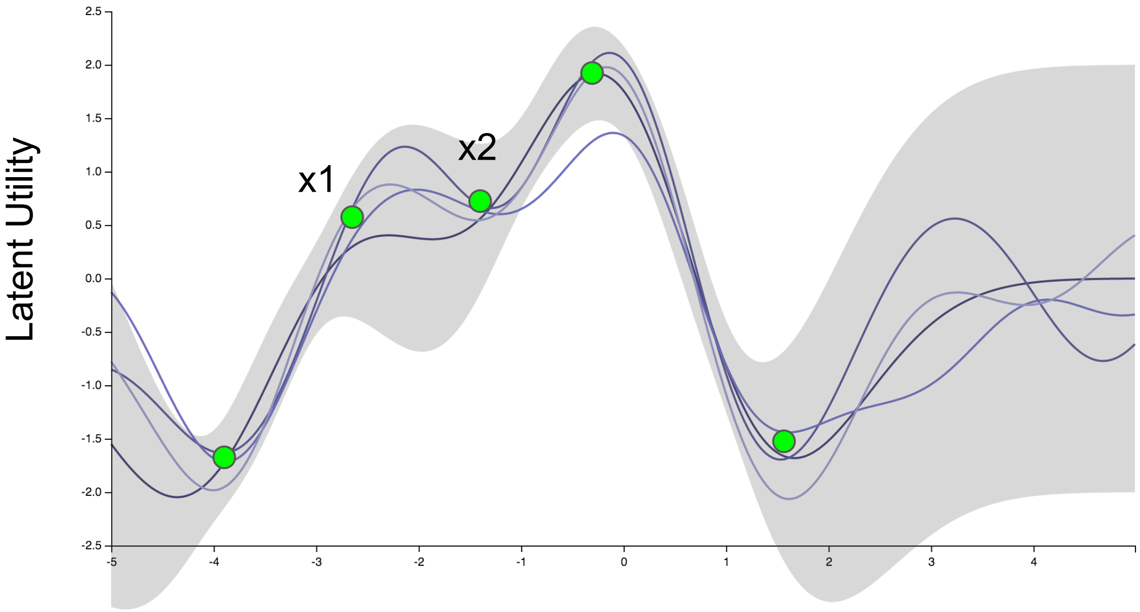
d = (0.5 - 0.6) / (2.0)
d = -0.05

Bradley-Terry

p = [0.48, 0.52]

Bradley-Terry Model Extension for Ties

The Bradley-Terry model relates the comparison points' latent function values to discrete binary preferences.



x1 = -2.9 x2 = -1.2

f1 = 0.5 f2 = 0.6

d = (0.5 - 0.6) / (2.0)

d = -0.05

Bradley-Terry

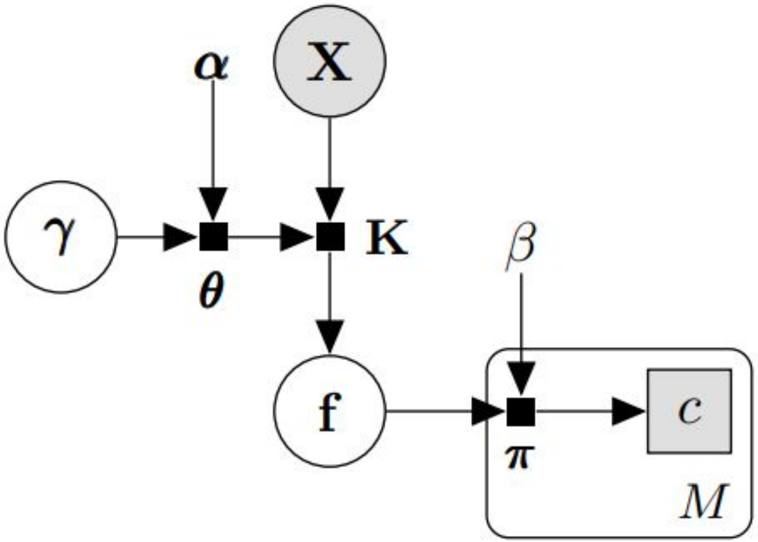
p = [0.48, 0.52]

Bradley-Terry with Ties (β = 2.5)

p = [0.28, 0.43, 0.29]

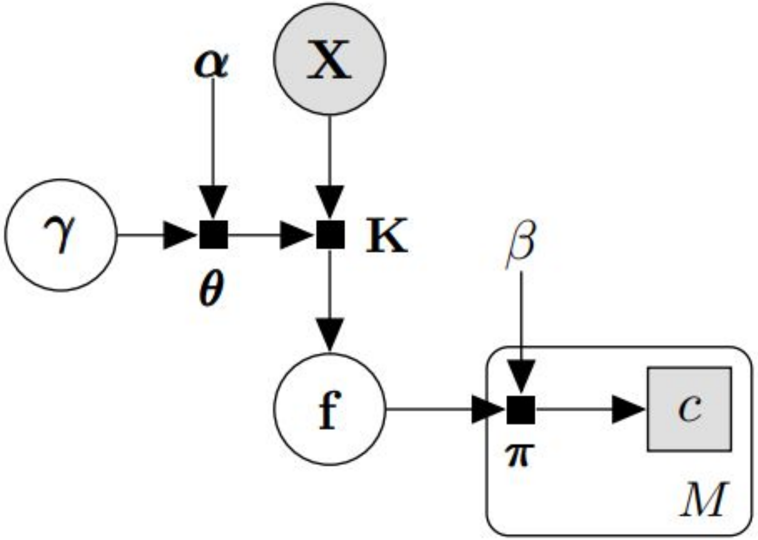
Inference

We have defined a probabilistic model capable of generating preference observations, with the possibility of ties.



Inference

We have defined a probabilistic model capable of generating preference observations, with the possibility of ties.



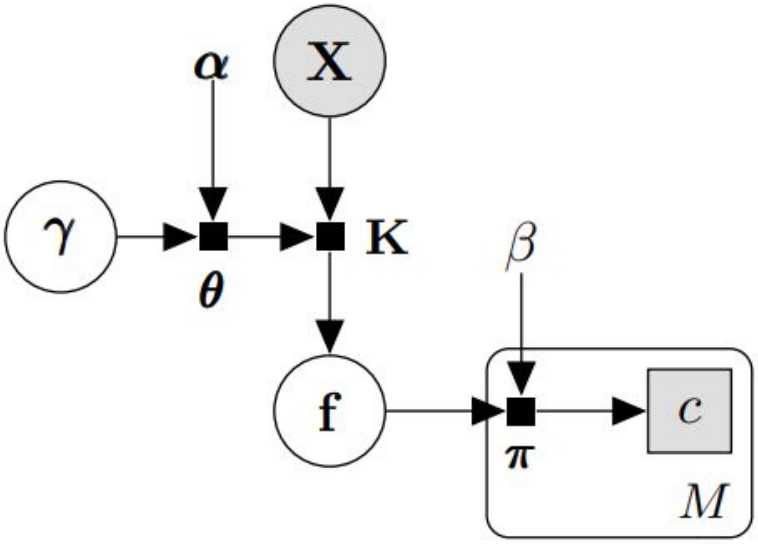
z = {**f**, **γ**}
Latent vars

x = {**X**, **c**}
Observed vars

Inference

We have defined a probabilistic model capable of generating preference observations, with the possibility of ties.

p(**x**, **z**) is fully defined (joint distribution) when we know **z**. However, we are interested in inferring the latent variables given only the observed preferences and the unique comparison points.



z = {**f**, **γ**}
Latent vars

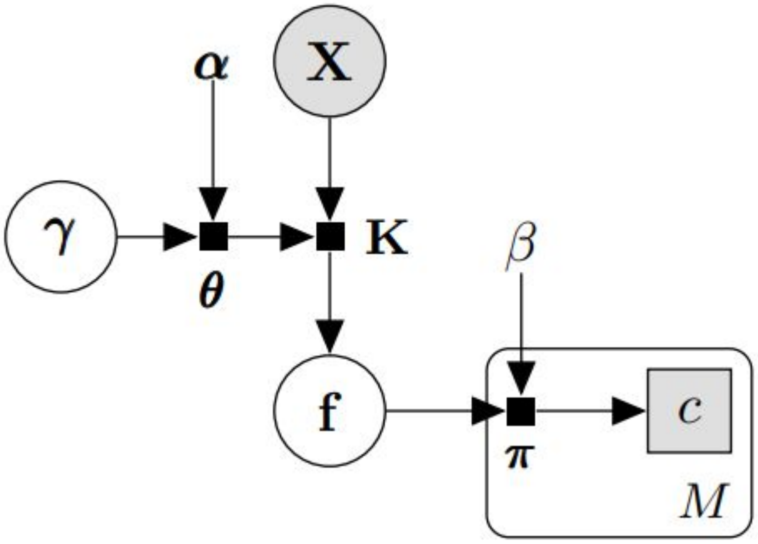
x = {**X**, **c**}
Observed vars

Inference

We have defined a probabilistic model capable of generating preference observations, with the possibility of ties.

p(**x**, **z**) is fully defined (joint distribution) when we know **z**. However, we are interested in inferring the latent variables given only the observed preferences and the unique comparison points.

How to find (or approximate) the posterior p(**z** | **x**) ?



z = {**f**, **γ**}
Latent vars

x = {**X**, **c**}
Observed vars

Variational Inference

Many strategies to approximate $p(z \mid x)$. One idea is to pick an approximating distribution $q(z)$ and minimize divergence between $p(z \mid x)$ and $q(z)$

Variational Inference

Many strategies to approximate p(z | x). One idea is to pick an approximating distribution q(z) and minimize divergence between p(z | x) and q(z)

This is the core idea behind variational inference : use optimization to minimize divergence between approximating distribution q(z) and true posterior p(z | x)

Variational Inference

Many strategies to approximate p(z | x). One idea is to pick an approximating distribution q(z) and minimize divergence between p(z | x) and q(z)

This is the core idea behind variational inference : use optimization to minimize divergence between approximating distribution q(z) and true posterior p(z | x)

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

One approximation strategy is to use the **mean-field approximation** that simply uses factorized gaussians for each of the latent random variables of interest.

Variational Inference

How do we minimize divergence between posterior p and approximation q?

Variational Inference

How do we minimize divergence between posterior p and approximation q?

$$\boldsymbol{\lambda}^* = \arg\min_{\boldsymbol{\lambda}} \; KL(q \,||\, p)$$

Variational Inference

How do we minimize divergence between posterior p and approximation q?

$$\begin{aligned}
\boldsymbol{\lambda}^* &= \arg\min_{\boldsymbol{\lambda}} \; KL(q \,||\, p) \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log \frac{q(\mathbf{z}\,;\,\boldsymbol{\lambda})}{p(\mathbf{z} \mid \mathbf{x})} \, d\mathbf{z}
\end{aligned}$$

Variational Inference

How do we minimize divergence between posterior p and approximation q?

$$
\begin{aligned}
\boldsymbol{\lambda}^* &= \arg\min_{\boldsymbol{\lambda}} \; KL(q \,||\, p) \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log \frac{q(\mathbf{z}\,;\,\boldsymbol{\lambda})}{p(\mathbf{z}\mid\mathbf{x})}\, d\mathbf{z} \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log \frac{q(\mathbf{z}\,;\,\boldsymbol{\lambda})}{p(\mathbf{z},\mathbf{x})}\, d\mathbf{z} + \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log p(\mathbf{x})\, d\mathbf{z}
\end{aligned}
$$

Variational Inference

How do we minimize divergence between posterior p and approximation q?

$$\begin{aligned}
\boldsymbol{\lambda}^* &= \arg\min_{\boldsymbol{\lambda}} \; KL(q \,||\, p) \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log \frac{q(\mathbf{z}\,;\,\boldsymbol{\lambda})}{p(\mathbf{z}\mid\mathbf{x})}\, d\mathbf{z} \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log \frac{q(\mathbf{z}\,;\,\boldsymbol{\lambda})}{p(\mathbf{z},\mathbf{x})}\, d\mathbf{z} + \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log p(\mathbf{x})\, d\mathbf{z} \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log \frac{q(\mathbf{z}\,;\,\boldsymbol{\lambda})}{p(\mathbf{z},\mathbf{x})}\, d\mathbf{z}
\end{aligned}$$

Variational Inference

How do we minimize divergence between posterior p and approximation q?

$$
\begin{aligned}
\boldsymbol{\lambda}^* &= \arg\min_{\boldsymbol{\lambda}} \; KL(q \,||\, p) \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log \frac{q(\mathbf{z}\,;\,\boldsymbol{\lambda})}{p(\mathbf{z}\mid\mathbf{x})}\, d\mathbf{z} \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log \frac{q(\mathbf{z}\,;\,\boldsymbol{\lambda})}{p(\mathbf{z},\mathbf{x})}\, d\mathbf{z} + \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log p(\mathbf{x})\, d\mathbf{z} \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z}\,;\,\boldsymbol{\lambda}) \log \frac{q(\mathbf{z}\,;\,\boldsymbol{\lambda})}{p(\mathbf{z},\mathbf{x})}\, d\mathbf{z} \\
&= \arg\min_{\boldsymbol{\lambda}} \; \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})} \left[\log q(\mathbf{z}\,;\,\boldsymbol{\lambda}) - \log p(\mathbf{z},\mathbf{x})\right]
\end{aligned}
$$

Variational Inference

How do we minimize divergence between posterior p and approximation q?

$$
\begin{aligned}
\boldsymbol{\lambda}^* &= \arg\min_{\boldsymbol{\lambda}} \; KL(q \,||\, p) \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z} \,;\, \boldsymbol{\lambda}) \log \frac{q(\mathbf{z} \,;\, \boldsymbol{\lambda})}{p(\mathbf{z} \mid \mathbf{x})} \, d\mathbf{z} \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z} \,;\, \boldsymbol{\lambda}) \log \frac{q(\mathbf{z} \,;\, \boldsymbol{\lambda})}{p(\mathbf{z}, \mathbf{x})} \, d\mathbf{z} + \int q(\mathbf{z} \,;\, \boldsymbol{\lambda}) \log p(\mathbf{x}) \, d\mathbf{z} \\
&= \arg\min_{\boldsymbol{\lambda}} \int q(\mathbf{z} \,;\, \boldsymbol{\lambda}) \log \frac{q(\mathbf{z} \,;\, \boldsymbol{\lambda})}{p(\mathbf{z}, \mathbf{x})} \, d\mathbf{z} \\
&= \arg\min_{\boldsymbol{\lambda}} \; \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z};\boldsymbol{\lambda})} \left[\log q(\mathbf{z} \,;\, \boldsymbol{\lambda}) - \log p(\mathbf{z}, \mathbf{x})\right] \\
&= \arg\max_{\boldsymbol{\lambda}} \; \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z};\boldsymbol{\lambda})} \left[\log p(\mathbf{z}, \mathbf{x}) - \log q(\mathbf{z} \,;\, \boldsymbol{\lambda})\right]
\end{aligned}
$$

Evidence Lower Bound (ELBO)

Variational Inference

Maximizing the ELBO is equivalent to minimizing KL(q || p). If we can estimate the gradient of this function we can optimize using gradient descent [6]

Variational Inference

Maximizing the ELBO is equivalent to minimizing KL(q || p). If we can estimate the gradient of this function we can optimize using gradient descent [6]

$$\begin{aligned}\boldsymbol{\lambda}^* &= \arg\max_{\boldsymbol{\lambda}} \ \mathrm{ELBO}(\boldsymbol{\lambda}) \\ &= \arg\max_{\boldsymbol{\lambda}} \ \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})}\left[\log p(\mathbf{z},\mathbf{x}) - \log q(\mathbf{z}\ ;\ \boldsymbol{\lambda})\right]\end{aligned}$$

Variational Inference

Maximizing the ELBO is equivalent to minimizing KL(q || p). If we can estimate the gradient of this function we can optimize using gradient descent [6]

$$\begin{aligned}\boldsymbol{\lambda}^* &= \arg\max_{\boldsymbol{\lambda}} \ \mathrm{ELBO}(\boldsymbol{\lambda}) \\ &= \arg\max_{\boldsymbol{\lambda}} \ \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})}\left[\log p(\mathbf{z},\mathbf{x}) - \log q(\mathbf{z}\ ;\ \boldsymbol{\lambda})\right]\end{aligned}$$

$$\nabla_{\boldsymbol{\lambda}}\, \mathrm{ELBO}(\boldsymbol{\lambda}) = \nabla_{\boldsymbol{\lambda}}\, \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})}\left[\log p(\mathbf{z},\mathbf{x}) - \log q(\mathbf{z}\ ;\ \boldsymbol{\lambda})\right]$$

Variational Inference

Maximizing the ELBO is equivalent to minimizing KL(q || p). If we can estimate the gradient of this function we can optimize using gradient descent [6]

$$
\begin{aligned}
\boldsymbol{\lambda}^* &= \arg\max_{\boldsymbol{\lambda}} \ \mathrm{ELBO}(\boldsymbol{\lambda}) \\
&= \arg\max_{\boldsymbol{\lambda}} \ \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})}\left[\log p(\mathbf{z},\mathbf{x}) - \log q(\mathbf{z}\,;\,\boldsymbol{\lambda})\right]
\end{aligned}
$$

$$
\begin{aligned}
\nabla_{\boldsymbol{\lambda}}\,\mathrm{ELBO}(\boldsymbol{\lambda}) &= \nabla_{\boldsymbol{\lambda}}\,\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})}\left[\log p(\mathbf{z},\mathbf{x}) - \log q(\mathbf{z}\,;\,\boldsymbol{\lambda})\right] \\
&= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})}\left[\left(\log p(\mathbf{z},\mathbf{x}) - \log q(\mathbf{z}\,;\,\boldsymbol{\lambda})\right)\nabla_{\boldsymbol{\lambda}}\log q(\mathbf{z}\,;\,\boldsymbol{\lambda})\right]
\end{aligned}
$$

*[5] for deriv.

Variational Inference

Maximizing the ELBO is equivalent to minimizing KL(q || p). If we can estimate the gradient of this function we can optimize using gradient descent [6]

$$\begin{aligned}\boldsymbol{\lambda}^* &= \arg\max_{\boldsymbol{\lambda}} \ \mathrm{ELBO}(\boldsymbol{\lambda}) \\ &= \arg\max_{\boldsymbol{\lambda}} \ \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})}\left[\log p(\mathbf{z},\mathbf{x}) - \log q(\mathbf{z}\,;\,\boldsymbol{\lambda})\right]\end{aligned}$$

$$\begin{aligned}\nabla_{\boldsymbol{\lambda}}\,\mathrm{ELBO}(\boldsymbol{\lambda}) &= \nabla_{\boldsymbol{\lambda}}\,\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})}\left[\log p(\mathbf{z},\mathbf{x}) - \log q(\mathbf{z}\,;\,\boldsymbol{\lambda})\right] \\ &= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z};\boldsymbol{\lambda})}\left[(\log p(\mathbf{z},\mathbf{x}) - \log q(\mathbf{z}\,;\,\boldsymbol{\lambda}))\,\nabla_{\boldsymbol{\lambda}}\log q(\mathbf{z}\,;\,\boldsymbol{\lambda})\right] \\ &\approx \frac{1}{S}\sum_{s=1}^{S}\left[(\log p(\mathbf{z}_s,\mathbf{x}) - \log q(\mathbf{z}_s\,;\,\boldsymbol{\lambda}))\,\nabla_{\boldsymbol{\lambda}}\log q(\mathbf{z}_s\,;\,\boldsymbol{\lambda})\right] \\ &\quad \text{where } \mathbf{z}_s \sim q(\mathbf{z}\,;\,\boldsymbol{\lambda})\end{aligned}$$

*[5] for deriv.

Monte Carlo Estimate

Acquisition Function for Preference-Based Opt.

With a model and inference algorithm selected, we still need to decide on mechanism to select next point to present to user to compare. Previous work has proposed use of expected improvement [1]

Acquisition Function for Preference-Based Opt.

With a model and inference algorithm selected, we still need to decide on mechanism to select next point to present to user to compare. Previous work has proposed use of expected improvement [1]

$$\mathbf{k}_* = [rbf(\mathbf{x}^*, \mathbf{x}_1, \boldsymbol{\theta}) \ \cdots \ rbf(\mathbf{x}^*, \mathbf{x}_N, \boldsymbol{\theta})]$$

$$\mu(\mathbf{x}^*) = \mathbf{k}_*^{\mathsf{T}} \mathbf{K}^{-1} \mathbf{f}$$

$$s^2(\mathbf{x}^*) = rbf(\mathbf{x}^*, \mathbf{x}^*, \boldsymbol{\theta}) - \mathbf{k}_*^{\mathsf{T}} \mathbf{K}^{-1} \mathbf{k}_*$$

$$d = \mu(\mathbf{x}^*) - f_{best}$$

$$a_{\mathsf{EI}}(\mathbf{x}^*; \mathbf{z}) = \begin{cases} d\Phi(\frac{d}{s(\mathbf{x}^*)}) + s(\mathbf{x}^*)\phi(\frac{d}{s(\mathbf{x}^*)}), & \text{if } s(\mathbf{x}^*) > 0 \\ 0, & \text{if } s(\mathbf{x}^*) = 0 \end{cases}$$

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

Experiments

Considered efficiency of method in minimizing synthetic test functions using only pairwise comparative observations

Experiments

Considered efficiency of method in minimizing synthetic test functions using only pairwise comparative observations

At each iteration a query point (x_n) is selected by the algorithm and compared against the current best point (x_b) using a test function (f_test)

Experiments

Considered efficiency of method in minimizing synthetic test functions using only pairwise comparative observations

At each iteration a query point (x_n) is selected by the algorithm and compared against the current best point (x_b) using a test function (f_test)

The discrete preference observations are simulated in the following way :

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

Experiments

Vary equivalent tolerance thresh under two settings (0.1, 0.001)

Experiments

Vary tolerance thresh under two settings (0.1, 0.001)

Baselines search methods

Random Search	Pick random point in domain

Experiments

Vary tolerance thresh under two settings (0.1, 0.001)

Baselines search methods

Random Search — Pick random point in domain

Pure Exploration Search

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

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

Experiments

Vary tolerance thresh under two settings (0.1, 0.001)

Baselines search methods

Random Search — Pick random point in domain

Pure Exploration Search

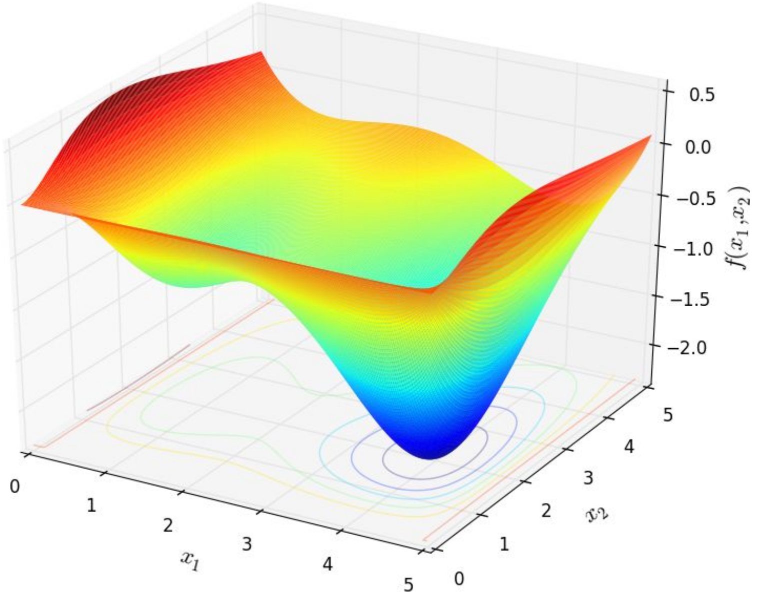
$$a_{\mathsf{PE}}(\mathbf{x}^*; \mathbf{z}) = rbf(\mathbf{x}^*, \mathbf{x}^*, \boldsymbol{\theta}) - \mathbf{k}_*^{\mathsf{T}} \mathbf{K}^{-1} \mathbf{k}_*$$

$$\mathbf{x}^n = \arg\max_{\mathbf{x}^*} \int a_{\mathsf{PE}}(\mathbf{x}^*; \mathbf{z}) q(\mathbf{z} \, ; \, \boldsymbol{\lambda}) dz$$

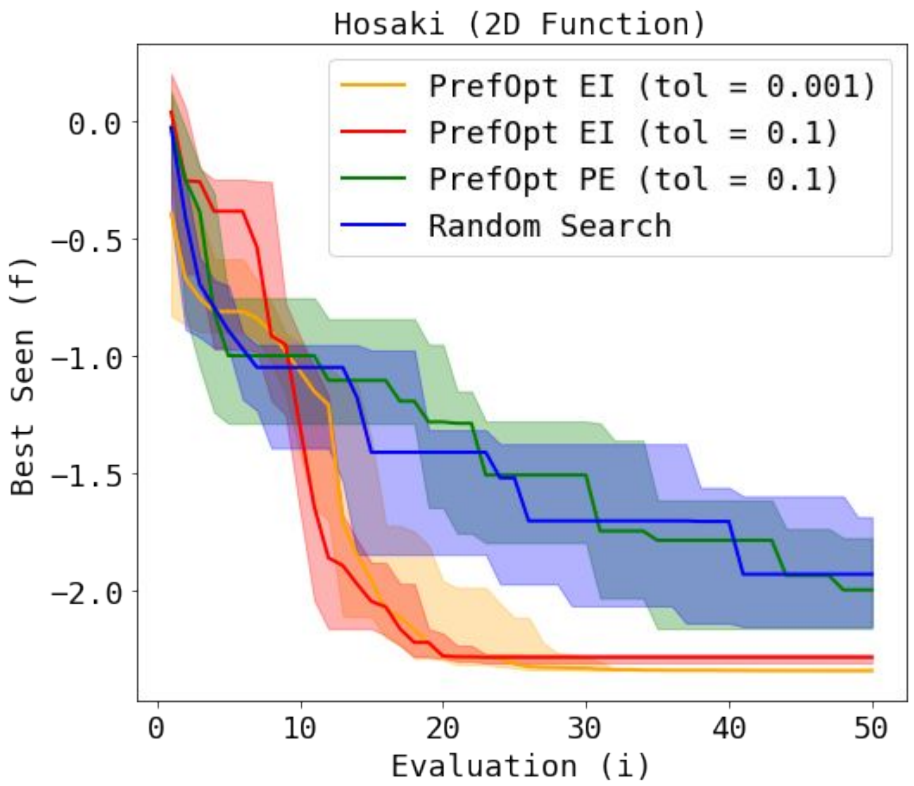
Run each strategy 10 times and record interquartile range

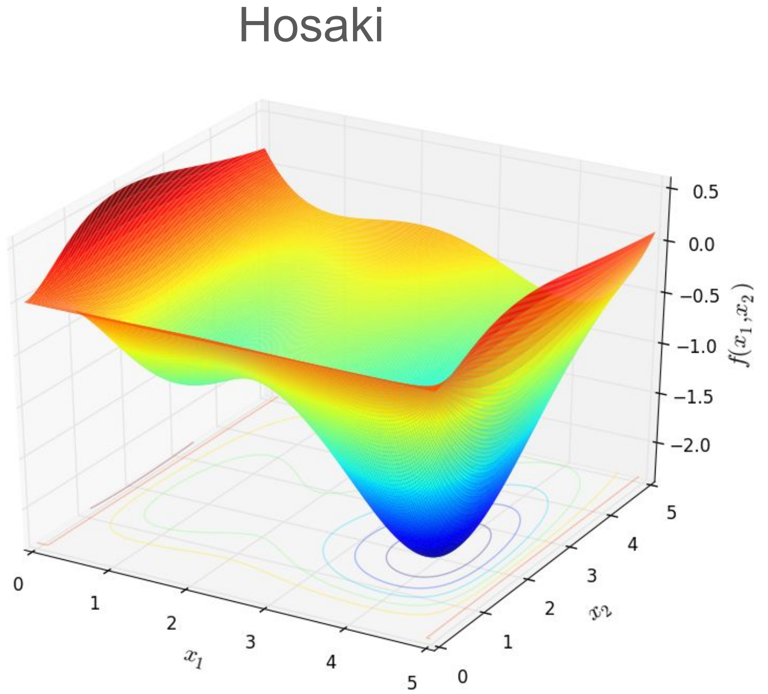
Experiments

Hosaki



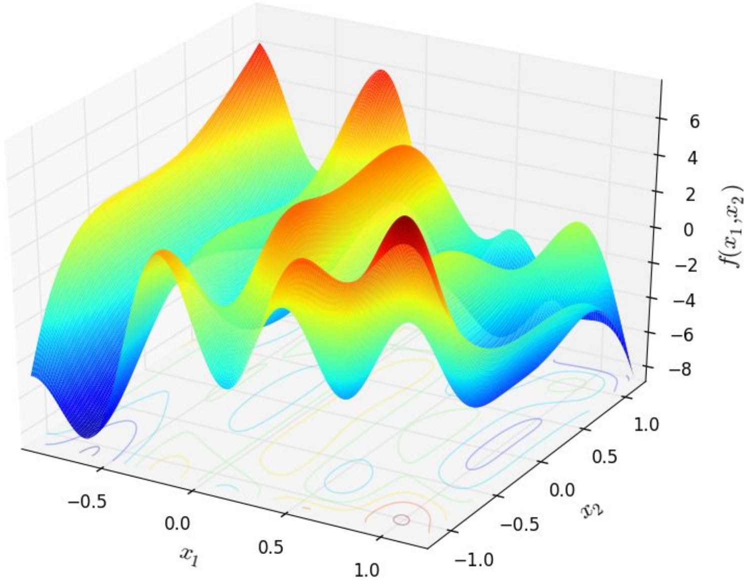
Experiments



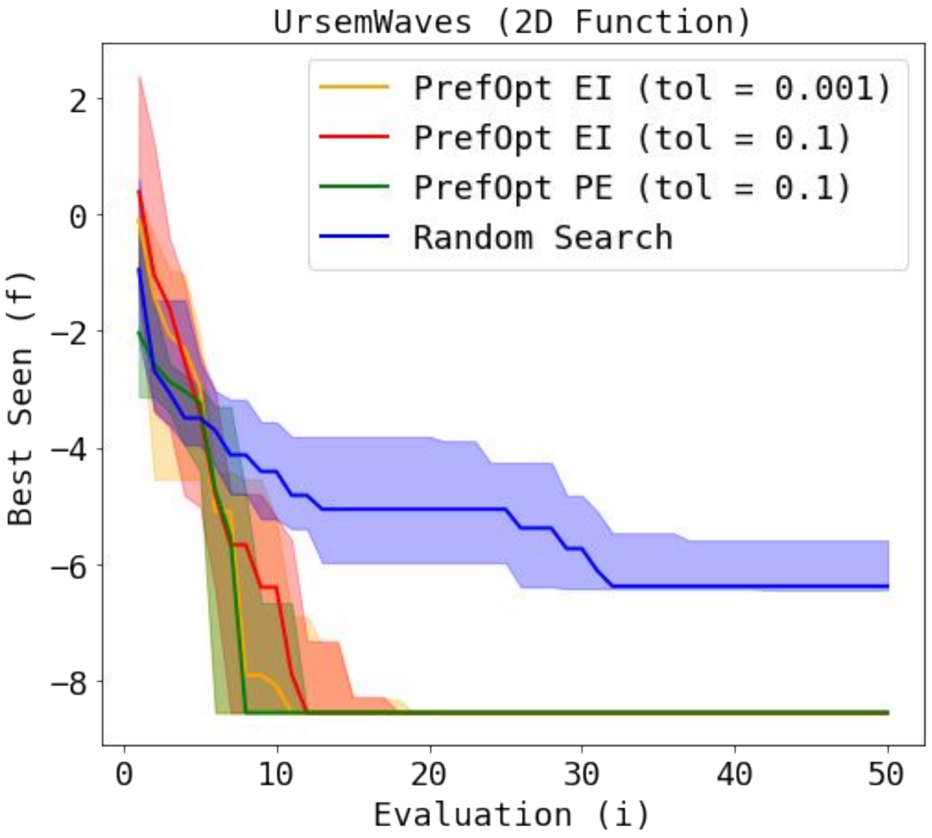


Experiments

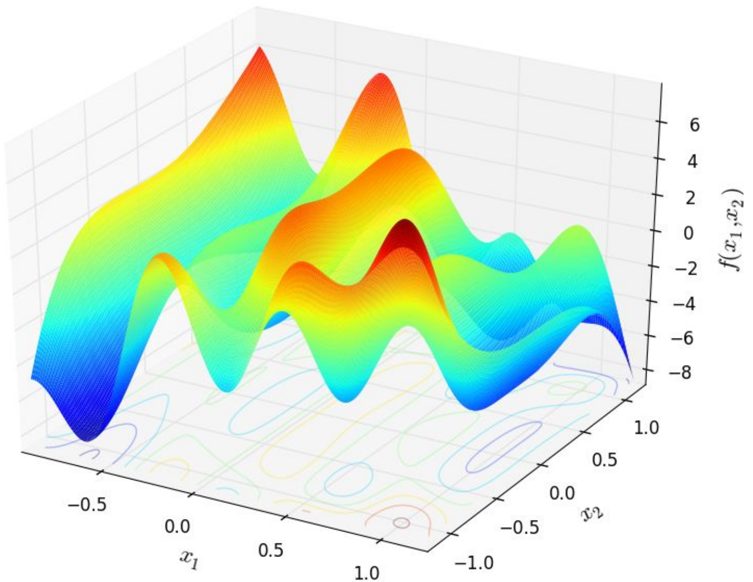
UrsemWaves



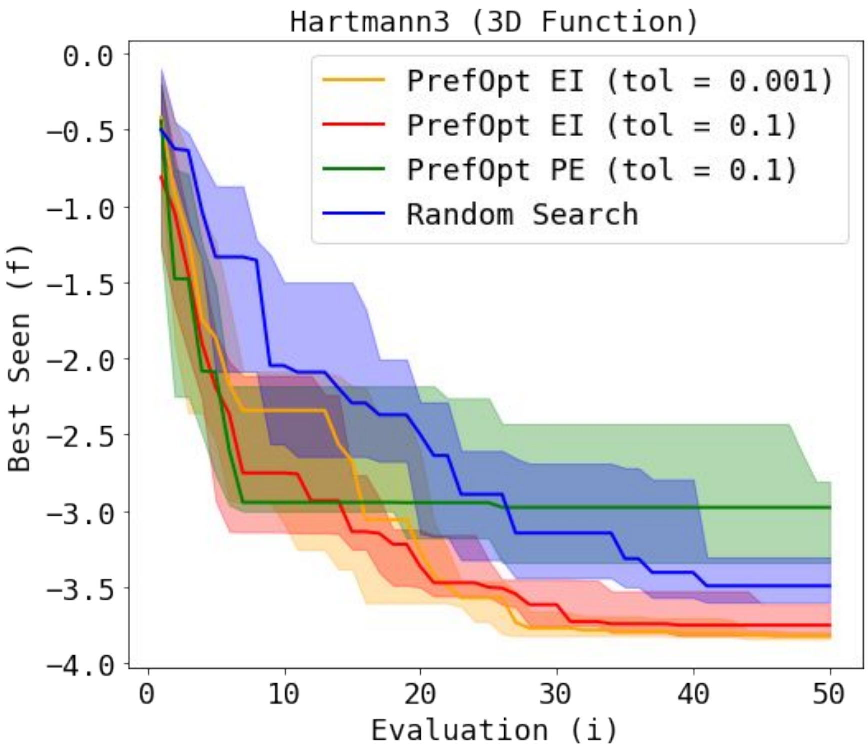
Experiments



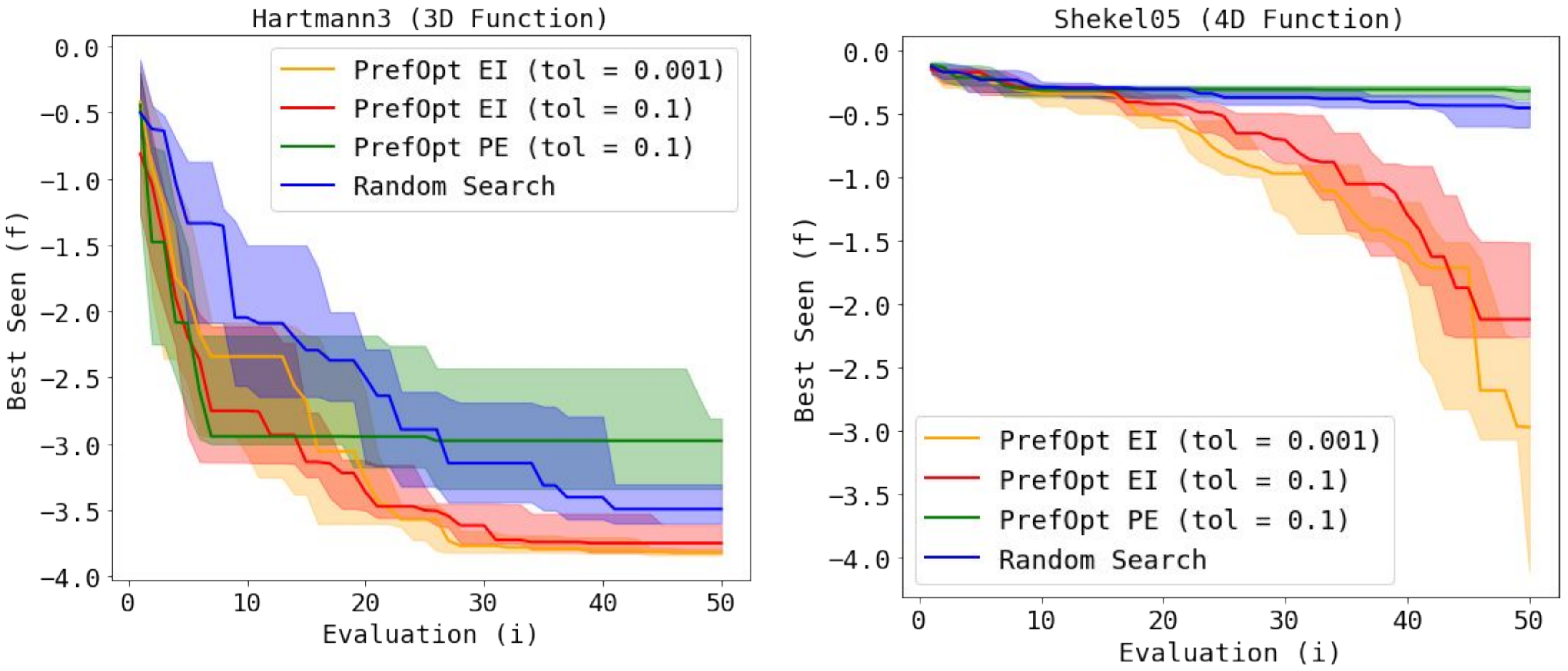
UrsemWaves



Experiments



Experiments



PrefOpt Software

Open source python library built on top of Edward, TensorFlow

PrefOpt Software

Open source python library built on top of Edward, TensorFlow

Succinct interface to conduct preference-based optimizations

```
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)
```

Future Work

Could it all be much simpler?

Future Work

Could it all be much simpler?

Mean-field approx. for variational inference might be giving poorer quality results

Future Work

Could it all be much simpler?

Mean-field approx. for variational inference might be giving poorer quality results

Alternative sampling mechanisms to estimate posterior

Future Work

Could it all be much simpler?

Mean-field approx. for variational inference might be giving poorer quality results

Alternative sampling mechanisms to estimate posterior

Still have nuisance parameters in model (β, σ, α), how to best eliminate?

Future Work

Could it all be much simpler?

Mean-field approx. for variational inference might be giving poorer quality results

Alternative sampling mechanisms to estimate posterior

Still have nuisance parameters in model (β, σ, α), how to best eliminate?

Batch query points (lots of work in Bayes Opt on this problem already)

Future Work

Could it all be much simpler?

Mean-field approx. for variational inference might be giving poorer quality results

Alternative sampling mechanisms to estimate posterior

Still have nuisance parameters in model (β, σ, α), how to best eliminate?

Batch query points (lots of work in Bayes Opt on this problem already)

Beyond low dimensional spaces : How can we simplify building ML / RL systems via preferences? [7]

Questions / Comments

Thanks for Listening!

References

[1] Eric Brochu. *Interactive Bayesian Optimization: Learning Parameters for Graphics and Animation*.

[2] JB Nielsen, JN Widex, BS Jensen, J Larsen *Hearing Aid Personalization*

[3] Nitish Thatte, Helei Duan, Hartmut Geyer *A Sample-Efficient Black-Box Optimizer to Train Policies for Human-in-the-Loop Systems with User Preferences*

[4] PV Rao and L Kupper. *Ties in paired-comparison experiments: A generalization of the bradley-terry model*.

[5] Rajesh Ranganath, Sean Gerrish, David M. Blei. *Black Box Variational Inference*

[6] D Tran, MD. Hoffman, Rif A. Saurous, E Brevdo, K Murphy, D Blei. *Deep Probabilistic Programming*

[7] P Christiano, J Leike, TB Brown, M Martic, S Legg, D Amodei. *Deep reinforcement learning from human preferences*