BayesOpt: hot topics and current challenges

Javier González

Masterclass, 7-February, 2107 @Lancaster University

Agenda of the day

- **9:00-11:00, Introduction to Bayesian Optimization**:
  - What is BayesOpt and why it works?
  - Relevant things to know.
- **11:30-13:00, Connections, extensions and applications**:
  - Extensions to multi-task problems, constrained domains, early-stopping, high dimensions.
  - Connections to Armed bandits and ABC.
  - An applications in genetics.
- **14:00-16:00, GPyOpt LAB!**: Bring your own problem!
- **16:30-15:30, Hot topics current challenges**:
  - Parallelization.
  - Non-myopic methods
  - Interactive Bayesian Optimization.

Section III: Hot topics and challenges

- Parallel Bayesian Optimization
- Non-myopic methods.
- Interactive Bayesian Optimization.

Scalable BO: Parallel/batch BO

Avoiding the bottleneck of evaluating f

- Cost of $f(\mathbf{x}_n)$ = cost of $\{f(\mathbf{x}_{n,1}), \ldots, f(\mathbf{x}_{n,nb})\}$.
- Many cores available, simultaneous lab experiments, etc.

Considerations when designing a batch

- Available pairs $\{(\mathbf{x}_j, y_i)\}_{i=1}^{n}$ are augmented with the evaluations of f on $\mathcal{B}_t^{n_b} = \{\mathbf{x}_{t,1}, \ldots, \mathbf{x}_{t,nb}\}$.
- Goal: design $\mathcal{B}_1^{n_b}, \ldots, \mathcal{B}_m^{n_b}$.

Notation:

- $\mathcal{I}_n$: represents the available data set $\mathcal{D}_n$ and the $\mathcal{GP}$ structure when n data points are available ($\mathcal{I}_{t,k}$ in the batch context).
- $\alpha(\mathbf{x}; \mathcal{I}_n)$: generic acquisition function given $\mathcal{I}_n$.

Optimal greedy batch design

Sequential policy: Maximize:

$$\alpha(\mathbf{x}; \mathcal{I}_{t,0})$$

Greedy batch policy, 1st element t-th batch: Maximize:

$$\alpha(\mathbf{x}; \mathcal{I}_{t,0})$$

Optimal greedy batch design

Sequential policy: Maximize:

$$\alpha(\mathbf{x}; \mathcal{I}_{t,0})$$

Greedy batch policy, 2nd element t-th batch: Maximize:

$$\int \alpha(\mathbf{x}; \mathcal{I}_{t,1}) p(y_{t,1} | \mathbf{x}_{t,1}, \mathcal{I}_{t,0}) p(\mathbf{x}_{t,1} | \mathcal{I}_{t,0}) d\mathbf{x}_{t,1} dy_{t,1}$$

- $p(y_{t,1} | \mathbf{x}_1, \mathcal{I}_{t,0})$: predictive distribution of the $\mathcal{GP}$.
- $p(\mathbf{x}_1 | \mathcal{I}_{t,0}) = \delta(\mathbf{x}_{t,1} - \arg\max_{\mathbf{x} \in \mathcal{X}} \alpha(\mathbf{x}; \mathcal{I}_{t,0}))$.

Optimal greedy batch design

Sequential policy: Maximize:

$$\alpha(\mathbf{x}; \mathcal{I}_{t,k-1})$$

Greedy batch policy, k-th element t-th batch: Maximize:

$$\int \alpha(\mathbf{x}; \mathcal{I}_{t,k-1}) \prod_{j=1}^{k-1} p(y_{t,j}|\mathbf{x}_{t,j}, \mathcal{I}_{t,j-1}) p(\mathbf{x}_{t,j}|\mathcal{I}_{t,j-1}) d\mathbf{x}_{t,j} dy_{t,j}$$

- $p(y_{t,j}|\mathbf{x}_{t,j}, \mathcal{I}_{t,j-1})$: predictive distribution of the $\mathcal{GP}$.
- $p(\mathbf{x}_j|\mathcal{I}_{t,j-1}) = \delta(\mathbf{x}_{t,j} - \arg\max_{\mathbf{x}\in\mathcal{X}} \alpha(\mathbf{x}; \mathcal{I}_{t,j-1}))$.

Available approaches

[Azimi et al., 2010; Desautels et al., 2012; Chevalier et al., 2013; Contal et al. 2013]

- Exploratory approaches, reduction in system uncertainty.
- Generate 'fake' observations of f using $p(y_{t,j}|\mathbf{x}_j, \mathcal{I}_{t,j-1})$.
- Simultaneously optimize elements on the batch using the joint distribution of $y_{t_1}, \dots y_{t,nb}$.

Bottleneck: All these methods require to iteratively update $p(y_{t,j}|\mathbf{x}_j, \mathcal{I}_{t,j-1})$ to model the iteration between the elements in the batch: $\mathcal{O}(n^3)$

How to design batches reducing this cost? Local penalization

Goal: eliminate the marginalization step

"To develop an heuristic approximating the 'optimal batch design strategy' at lower computational cost, while incorporating information about global properties of f from the $\mathcal{GP}$ model into the batch design"

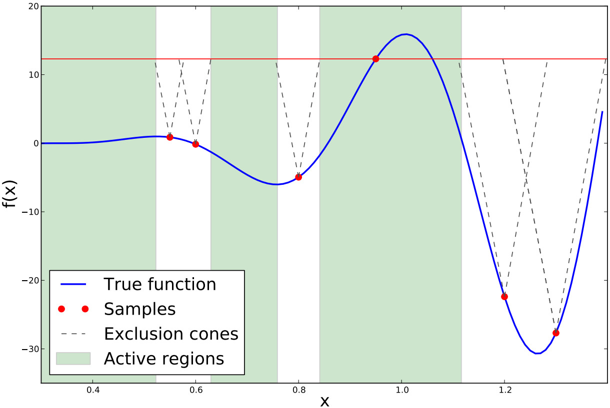
Lipschitz continuity:

$$|f(\mathbf{x}_1) - f(\mathbf{x}_2)| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|_p.$$

Interpretation of the Lipschitz continuity of f

$M = \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$ and $B_{r_{x_j}}(\mathbf{x}_j) = \{\mathbf{x} \in \mathcal{X} : \|\mathbf{x} - \mathbf{x}_j\| \leq r_{x_j}\}$ where

$$r_{x_j} = \frac{M - f(\mathbf{x}_j)}{L}$$



$x_M \notin B_{r_{x_j}}(\mathbf{x}_j)$ otherwise, the Lipschitz condition is violated.

Probabilistic version of $B_{r_x}(\mathbf{x})$

We can do this because $f(\mathbf{x}) \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$

- r_{x_j} is Gaussian with $\mu(r_{x_j}) = \frac{M-\mu(\mathbf{x}_j)}{L}$ and $\sigma^2(r_{x_j}) = \frac{\sigma^2(\mathbf{x}_j)}{L^2}$.

Local penalizers: $\varphi(\mathbf{x}; \mathbf{x}_j) = p(\mathbf{x} \notin B_{r_{\mathbf{x}_j}}(\mathbf{x}_j))$

$$\begin{aligned} \varphi(\mathbf{x}; \mathbf{x}_j) &= p(r_{\mathbf{x}_j} < \|\mathbf{x} - \mathbf{x}_j\|) \\ &= 0.5\mathrm{erfc}(-z) \end{aligned}$$

where $z = \frac{1}{\sqrt{2\sigma_n^2(\mathbf{x}_j)}}(L\|\mathbf{x}_j - \mathbf{x}\| - M + \mu_n(\mathbf{x}_j))$.

- Reflects the size of the 'Lipschitz' exclusion areas.
- Approaches to 1 when $\mathbf{x}$ is far form $\mathbf{x}_j$ and decreases otherwise.

Idea to collect the batches

Without using explicitly the model.

Optimal batch: maximization-marginalization

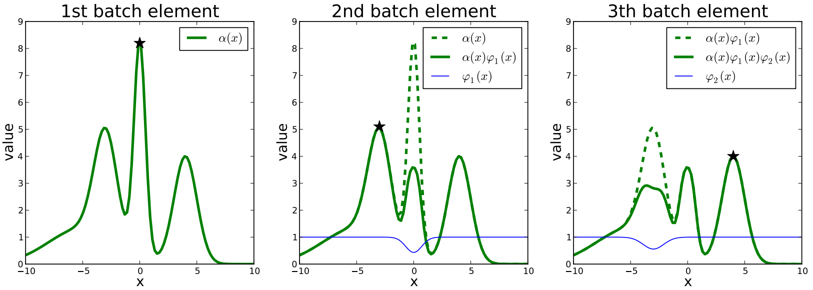
$$\int \alpha(\mathbf{x}; \mathcal{I}_{t,k-1}) \prod_{j=1}^{k-1} p(y_{t,j}|\mathbf{x}_{t,j}, \mathcal{I}_{t,j-1}) p(\mathbf{x}_{t,j}|\mathcal{I}_{t,j-1}) d\mathbf{x}_{t,j} dy_{t,j}$$

Proposal: maximization-penalization.

Use the $\varphi(\boldsymbol{x}; \boldsymbol{x}_j)$ to penalize the acquisition and predict the expected change in $\alpha(\boldsymbol{x}; \mathcal{I}_{t,k-1})$.

Local penalization strategy

[González, Dai, Hennig, Lawrence, 2016]



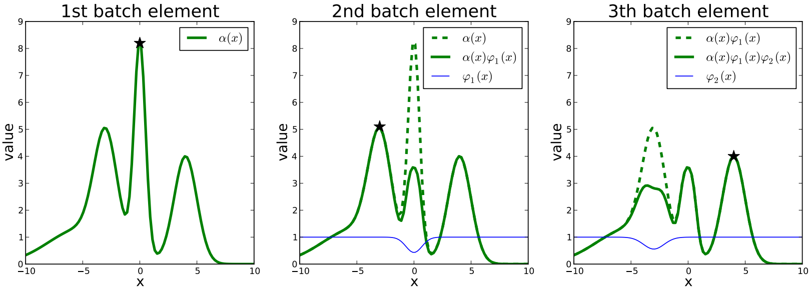
The maximization-penalization strategy selects $\mathbf{x}_{t,k}$ as

$$\mathbf{x}_{t,k} = \arg\max_{x \in \mathcal{X}} \left\{ g(\alpha(\mathbf{x}; \mathcal{I}_{t,0})) \prod_{j=1}^{k-1} \varphi(\mathbf{x}; \mathbf{x}_{t,j}) \right\},$$

g is a transformation of $\alpha(\mathbf{x}; \mathcal{I}_{t,0})$ to make it always positive.

Local penalization strategy

[González, Dai, Hennig, Lawrence, 2016]

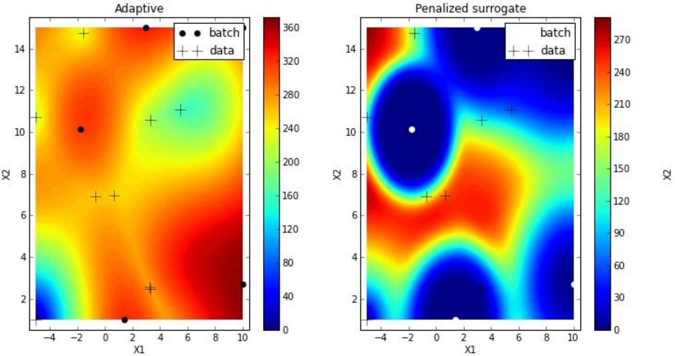


The maximization-penalization strategy selects $\mathbf{x}_{t,k}$ as

$$\mathbf{x}_{t,k} = \arg\max_{x \in \mathcal{X}} \left\{ g(\alpha(\mathbf{x}; \mathcal{I}_{t,0})) \prod_{j=1}^{k-1} \varphi(\mathbf{x}; \mathbf{x}_{t,j}) \right\},$$

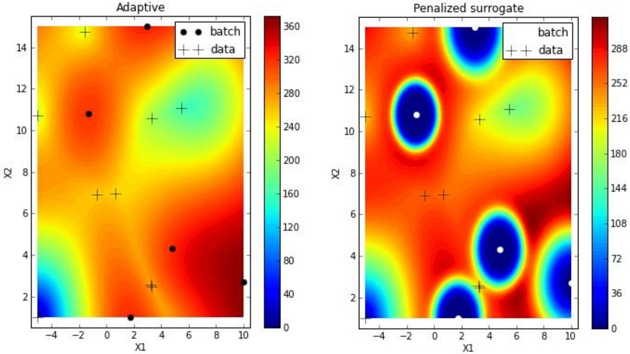
g is a transformation of $\alpha(\mathbf{x}; \mathcal{I}_{t,0})$ to make it always positive.

Example for $L = 50$



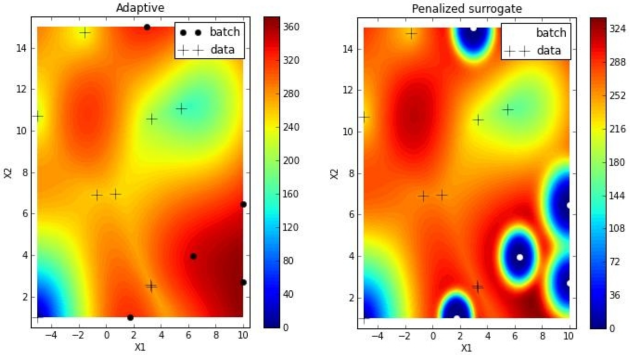
L controls the exploration-exploitation balance within the batch.

Example for $L = 100$



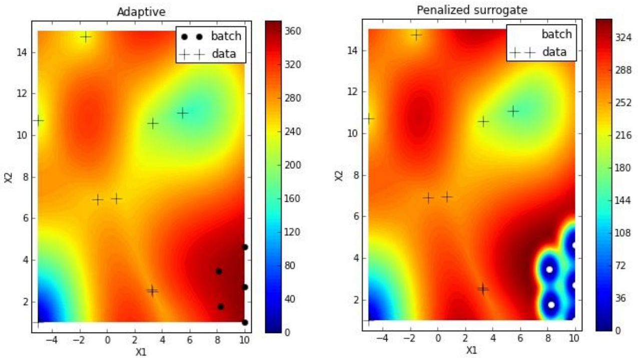
L controls the exploration-exploitation balance within the batch.

Example for $L = 150$



L controls the exploration-exploitation balance within the batch.

Example for $L = 250$



L controls the exploration-exploitation balance within the batch.

Finding an unique Lipschitz constant

Let $f : \mathcal{X} \to \mathbb{R}$ be a L-Lipschitz continuous function defined on a compact subset $\mathcal{X} \subseteq \mathbb{R}^D$. Then

$$L_p = \max_{\mathbf{x} \in \mathcal{X}} \|\nabla f(\mathbf{x})\|_p,$$

is a valid Lipschitz constant.

The gradient of f at $\mathbf{x}^*$ is distributed as a multivariate Gaussian

$$\nabla f(\mathbf{x}^*) | \mathbf{X}, \mathbf{y}, \mathbf{x}^* \sim \mathcal{N}(\mu_\nabla(\mathbf{x}^*), \Sigma^2_\nabla(\mathbf{x}^*))$$

We choose:

$$\hat{L} = \max_{\mathcal{X}} \|\mu_\nabla(\mathbf{x}^*)\|$$

Experiments: Sobol function

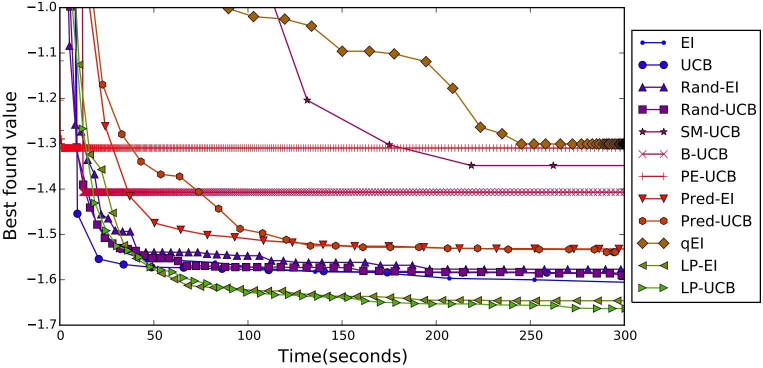
Best (average) result for some given time budget.

d	n_b	EI	UCB	Rand-EI	Rand-UCB	SM-UCB	B-UCB
2	5	0.31±0.03	0.32±0.06	0.32±0.05	**0.31±0.05**	1.86±1.06	0.56±0.03
	10			0.65±0.32	0.79±0.42	4.40±2.97	**0.59±0.00**
	20			0.67±0.31	0.75±0.32	-	0.57±0.01
5	5	8.84±3.69	11.89±9.44	9.19±5.32	10.59±5.04	137.2±113.0	**6.01±0.00**
	10			1.74±1.47	2.20±1.85	108.7±74.38	3.77±0.00
	20			2.18±2.30	2.76±3.06	-	2.53±0.00
10	5	559.1±1014	1463±1803	**690.5±947.5**	1825±2149	9e+04±7e+04	2098±0.00
	10			200.9±455.9	1149±1830	9e+04±1e+05	857.8±0.00
	20			639.4±1204	385.9±642.9	-	1656±0.00

d	n_b	PE-UCB	Pred-EI	Pred-UCB	qEI	LP-EI	LP-UCB
2	5	0.99±0.74	0.41±0.15	0.45±0.16	1.53±0.86	0.35±0.11	**0.31±0.06**
	10	0.66±0.29	1.16±0.70	1.26±0.81	3.82±2.09	0.66±0.48	0.69±0.51
	20	0.75±0.44	1.28±0.93	1.34±0.77	-	**0.50±0.21**	0.58±0.21
5	5	123.5±81.43	10.43±4.88	11.77±9.44	15.70±8.90	11.85±5.68	10.85±8.08
	10	120.8±78.56	9.58±7.85	11.66±11.48	17.69±9.04	3.88±4.15	**1.88±2.46**
	20	98.60±82.60	8.58±8.13	10.86±10.89	-	6.53±4.12	**1.44±1.93**
10	5	2e+05±2e+05	793.0±1226	1412±3032	-	1881±1176	1194±1428
	10	6e+04±8e+04	442.6±717.9	1725±3205	-	1042±1562	**100.4±338.7**
	20	5e+04±4e+04	1091±1724	2231±3110	-	1249±1570	**20.75±50.12**

2D experiment with ‘large domain’

Comparison in terms of the wall clock time



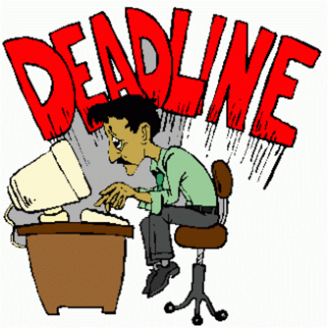
Myopia of optimisation techniques

- Most global optimisation techniques are myopic, in considering no more than a single step into the future.
- Relieving this myopia requires solving the *multi-step lookahead* problem.

Figure: Two evaluations, if the first evaluation is made myopically, the second must be sub-optimal.

Non-myopic thinking

To think non-myopically is important: it is a way of integrating in our decisions the information about our available (limited) resources to solve a given problem.



Acquisition function: expected loss

[Osborne, 2010]

Loss of evaluating f at $\mathbf{x}_*$ assuming it is returning y_*:

$$\lambda(y_*) \triangleq \begin{cases} y_*; & \text{if} \quad y_* \leq \eta \\ \eta; & \text{if} \quad y_* > \eta. \end{cases}$$

where $\eta = \min\{\mathbf{y}_0\}$, the current best found value.

The loss expectation is :

$$\Lambda_1(\mathbf{x}_*|\mathcal{I}_0) \triangleq \mathbb{E}[\min(y_*, \eta)] = \int \lambda(y_*) p(y_*|\mathbf{x}_*, \mathcal{I}_0) dy_*$$

$\mathcal{I}_0$ is the current information $\mathcal{D}$, θ and likelihood type.

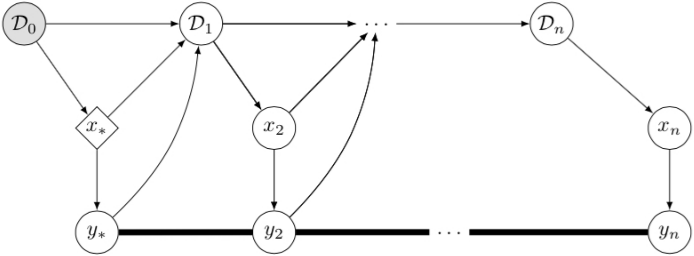
The expected loss (improvement) is myopic

- Selects the next evaluation as if it was the last one.
- The remaining available budget is not taken into account when deciding where to evaluate.

How to take into account the effect of future evaluations in the decision?

Expected loss with n steps ahead

Intractable even for a handful number of steps ahead



$$\Lambda_n(\mathbf{x}_*|\mathcal{I}_0) = \int \lambda(y_n) \prod_{j=1}^{n} p(y_j|\mathbf{x}_j, \mathcal{I}_{j-1}) p(\mathbf{x}_j|\mathcal{I}_{j-1}) dy_* \dots dy_n d\mathbf{x}_2 \dots d\mathbf{x}_n$$

- $p(y_j|\mathbf{x}_j, \mathcal{I}_{j-1})$: predictive distribution of the GP at $\mathbf{x}_j$ and
- $p(\mathbf{x}_j|\mathcal{I}_{j-1})$: optimisation step.

Relieving the myopia of Bayesian optimisation

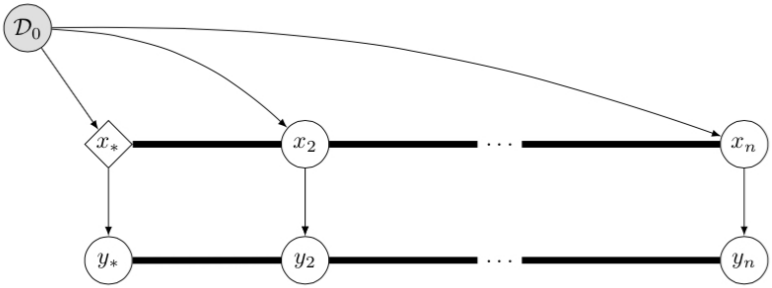
We present... GLASSES!

***G**lobal optimisation with **L**ook-**A**head through **S**tochastic **S**imulation and **E**xpected-loss **S**earch*

GLASSES

Rendering the approximation sparse

Idea: jointly model the epistemic uncertainty about the steps ahead using some defining *some* point process.



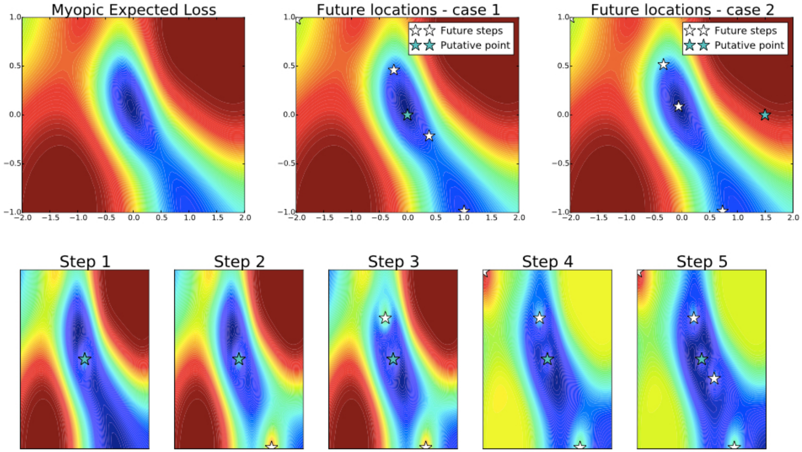
$$\Gamma_n(\mathbf{x}_*|\mathcal{I}_0) = \int \lambda(y_n) p(\mathbf{y}|\mathbf{X}, \mathcal{I}_0, \mathbf{x}_*) p(\mathbf{X}|\mathcal{I}_0, \mathbf{x}_*) d\mathbf{y} d\mathbf{X}$$

Selecting a good $p(\mathbf{X}|\mathcal{I}_0, \mathbf{x}_*)$ is complicated.

- Replace integrating over $p(\mathbf{X}|\mathcal{I}_0, \mathbf{x}_*)$ by conditioning over an oracle predictor $\mathcal{F}_n(\mathbf{x}_*)$ of the n future locations.
- $\mathbf{y} = (y_*, \dots, y_n)^T$: Gaussian outputs of f at $\mathcal{F}_n(\mathbf{x}_*)$.
- $\Lambda_n\big(\mathbf{x}_* \mid \mathcal{I}_0, \mathcal{F}_n(\mathbf{x}_*)\big) = \Gamma_n(\mathbf{x}_*|\mathcal{I}_0, \mathcal{F}_n(\mathbf{x}_*)) = \mathbb{E}\big[\min(\mathbf{y}, \eta)\big]$.
- $\mathbb{E}\big[\min(\mathbf{y}, \eta)\big]$ is computed using Expectation Propagation.

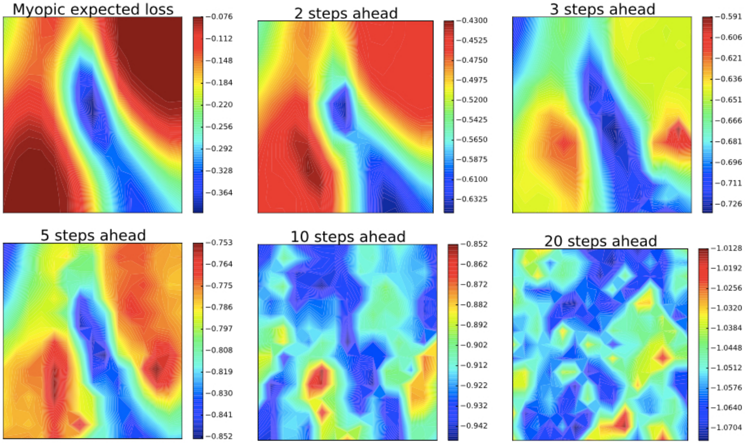
GLASSES: predicting the steps ahead

Oracle based on a batch BO method [Gonzalez et al., AISTATS'2016]



Can be interpreted as the MAP of a determinantal point process.

GLASSES: interpretation of the loss



Automatic balance between *exploration* and *exploitation*.

Results in a benchmark of objectives

	MPI	GP-LCB	EL	EL-2	EL-3	EL-5	EL-10	GLASSES
SinCos	0.7147	0.6058	0.7645	*0.8656*	0.6027	0.4881	*0.8274*	***0.9000***
Cosines	0.8637	0.8704	0.8161	*0.8423*	*0.8118*	0.7946	0.7477	***0.8722***
Branin	0.9854	0.9616	**0.9900**	0.9856	0.9673	0.9824	0.9887	0.9811
Sixhumpcamel	0.8983	**0.9346**	0.9299	0.9115	0.9067	0.8970	0.9123	0.8880
Mccormick	**0.9514**	0.9326	0.9055	*0.9139*	*0.9189*	*0.9283*	*0.9389*	*0.9424*
Dropwave	0.7308	0.7413	0.7667	0.7237	0.7555	0.7293	0.6860	***0.7740***
Powers	0.2177	0.2167	0.2216	*0.2428*	*0.2372*	*0.2390*	*0.2339*	***0.3670***
Ackley-2	0.8230	**0.8975**	0.7333	0.6382	0.5864	0.6864	0.6293	0.7001
Ackley-5	0.1832	0.2082	0.5473	*0.6694*	0.3582	0.3744	***0.6700***	0.4348
Ackley-10	0.9893	0.9864	0.8178	*0.9900*	*0.9912*	***0.9916***	*0.8340*	*0.8567*
Alpine2-2	**0.8628**	0.8482	0.7902	0.7467	0.5988	0.6699	0.6393	0.7807
Alpine2-5	0.5221	0.6151	**0.7797**	0.6740	0.6431	0.6592	0.6747	0.7123

GLASSES is overall the best method.

Interactive Bayesian optimization

Gonzalez et al, [2016]

Key question: what if it is easier to compare two points in the domain than obtaining a single output value for each one?

Preferential returns

Interactive Bayesian optimization

Gonzalez et al, [2016]

To find

$$\mathbf{x}_{min} = \arg\min_{\mathbf{x}\in\mathcal{X}} g(\mathbf{x}).$$

where g is not directly accessible. Queries to g can only be done in pairs of points or *duels* $[\mathbf{x}, \mathbf{x}'] \in \mathcal{X} \times \mathcal{X}$ from which binary feedback $\{0, 1\}$ is obtained

Useful when modeling human preferences

Modelling preferences

The model of choice is a Bernoulli probability function:

$$p(y = 1|[\mathbf{x}, \mathbf{x}']) = \pi_f([\mathbf{x}, \mathbf{x}'])$$

and

$$p(y = 0|[\mathbf{x}, \mathbf{x}']) = \pi_f([\mathbf{x}', \mathbf{x}])$$

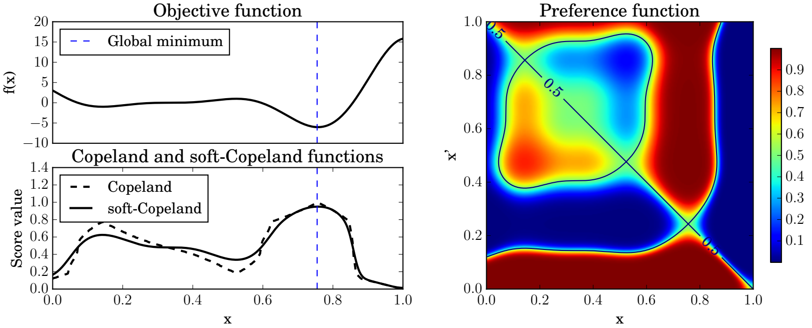
where $\pi : \Re \times \Re \to [0, 1]$ is a link function.

A natural choice for π_f is the logistic function

$$\pi_f([\mathbf{x}', \mathbf{x}]) = \sigma(f([\mathbf{x}', \mathbf{x}])) = \frac{1}{1 + e^{-f([\mathbf{x}', \mathbf{x}])}}$$

for $f([\mathbf{x}, \mathbf{x}']) = g(\mathbf{x}') - g(\mathbf{x})$.

Elements of the problem



Key concepts:

- Preference function: $\pi_f([\mathbf{x}', \mathbf{x}])$.
- Soft-Copeland score: $C(\mathbf{x}) = \mathrm{Vol}(\mathcal{X})^{-1} \int_{\mathcal{X}} \pi_f([\mathbf{x}, \mathbf{x}'])d\mathbf{x}'$.
- Condorcet's winner: point with maximal soft-Copeland score.

Idea

- Modeling the preference with a Gaussian process for classification.
- Select the new duel than maximizes the Copeland's score in expectation.

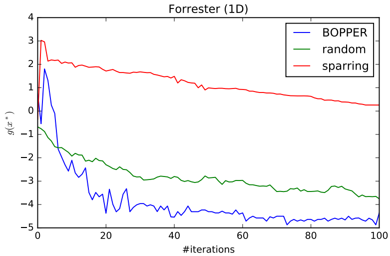
Compeland's expected improvement (CEI)

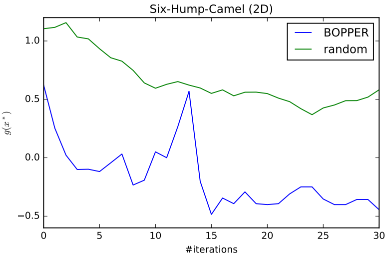
Acquisition for duels:

$$\begin{aligned}\alpha_{CEI}([\mathbf{x},\mathbf{x}'];\mathcal{D},\theta) &= \mathbb{E}\left[\max(0,c-c^\star)\right]\\ &= \pi_{f,j}([\mathbf{x},\mathbf{x}'])(c^\star_{j,\mathbf{x}}-c^\star_j)_+ + \pi_{f,j}([\mathbf{x}',\mathbf{x}])(c^\star_{j,\mathbf{x}'}-c^\star)\end{aligned}$$

- $c_j^\star$ is the value of the Condorcet's winner at iteration j.
- $c^\star_{\mathbf{x}}$ the value of the estimated Condorcer winner resulting of augmenting $\mathcal{D}_j$ with $\{[\mathbf{x},\mathbf{x}'],1\}$

Results





Model correlations with the Gaussian process helps!

Questions?