Course in Bayesian Optimization

Javier González

University of Sheffield, Sheffield, UK

29th October 2015

Recap

Yesterday we discussed:

- Bayesian Optimization is an efficient strategy to make ML completely automatic.
- We can use Bayesian optimization principles to design experiments sequentially.
- Probability theory and uncertainty are the keys.

Bayesian Optimization is AI for AI.

Today's agenda

- Which are the current challenges in Bayesian Optimisation?
- Can we extend BO ideas to other domains?
- Some fresh research results.

BO is very active field with still many open questions.

Challenges and extensions in Bayesian Optimization

- **Multi-task Bayesian optimization.**
- Non-stationary Bayesian optimization.
- Inequality constrains
- Scalable BO: high dimensional problems.
- Scalable BO: parallel approaches.
- Non-myopic methods.
- Applications: molecule design.

Multi task Bayesian Optimization

[Wersky et al., 2013]

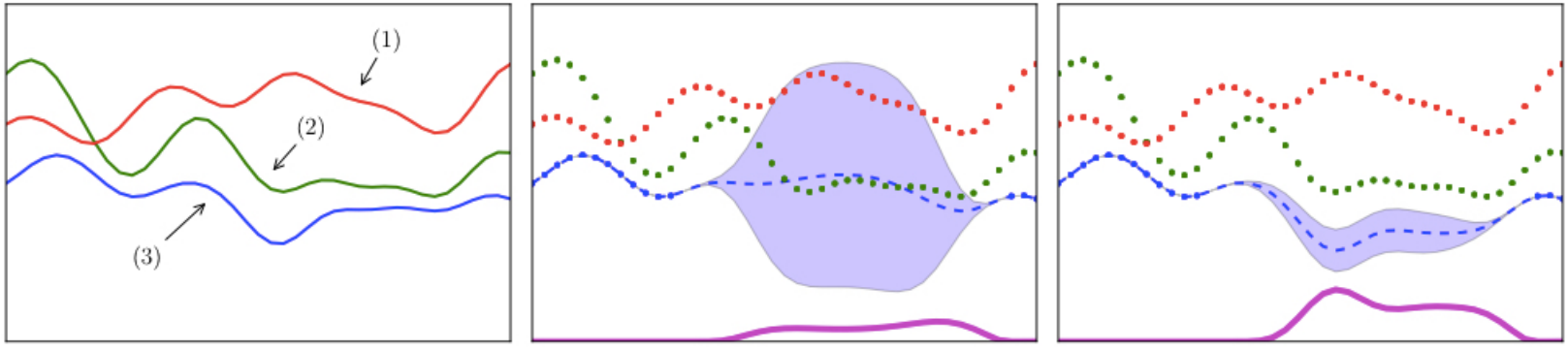
- We want to optimise an objective that it is very expensive to evaluate but we have access to another function, correlated with objective, that is cheaper to evaluate.
- The idea is to use the correlation among the function to improve the optimization.

Multi-output Gaussian process

$$\tilde{k}(x, x') = \mathbf{B} \otimes k(x, x')$$

Multi task Bayesian Optimization

[Wersky et all., 2013]



(a) Multi-task GP sample functions (b) Independent GP predictions (c) Multi-task GP predictions

- Correlation among tasks reduces global uncertainty.
- The choice (acquisition) changes.

Multi task Bayesian Optimization

[Wersky et al,, 2013]

- In other cases we want to optimize several tasks at the same time.
- We need to use a combination of them (the mean, for instance) or have a look to the Pareto frontiers of the problem.

Averaged expected improvement.

Multi task Bayesian Optimization

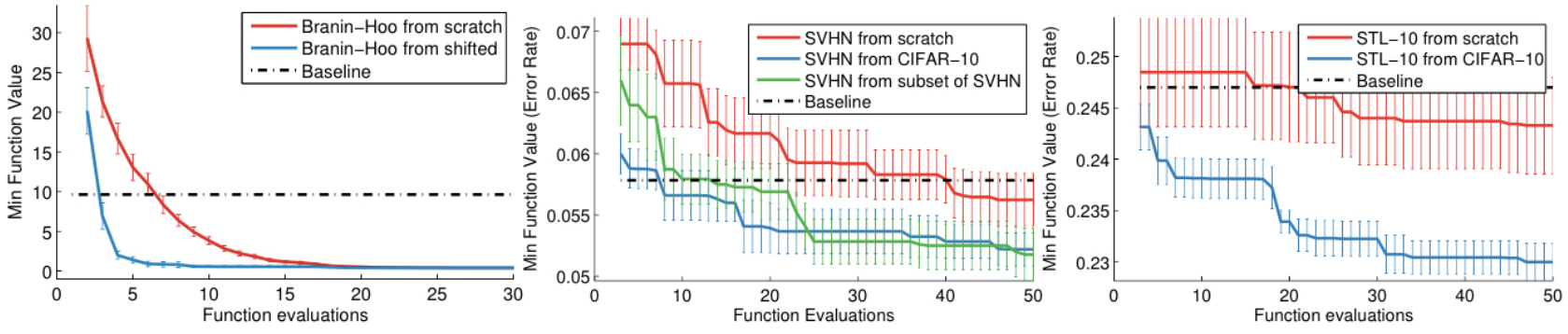
[Wersky et al., 2013]

- In other cases we want to optimize several tasks at the same time.
- We need to use a combination of them (the mean, for instance) or have a look to the Pareto frontiers of the problem.

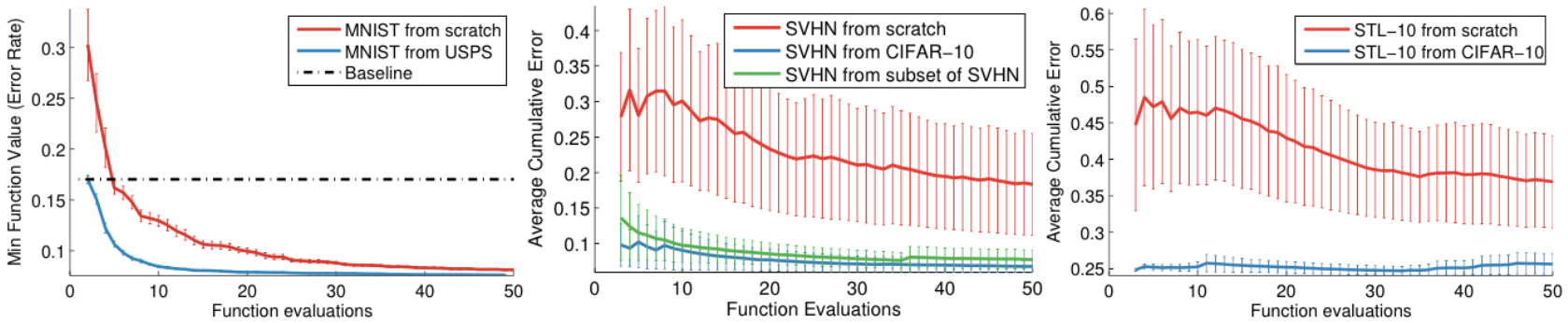
Averaged expected improvement.

Multi task Bayesian Optimization

[Wersky et al., 2013]



(a) Shifted Branin-Hoo (b) CNN on SVHN (c) CNN on STL-10



(d) LR on MNIST (e) SVHN ACE (f) STL-10 ACE

Challenges and extensions in Bayesian Optimization

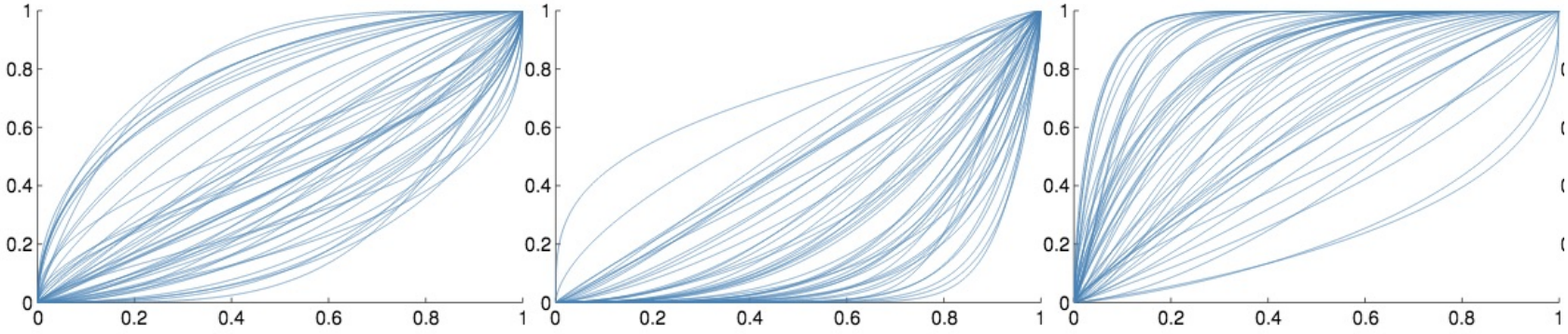
- Multi-task Bayesian optimization.
- **Non-stationary Bayesian optimization.**
- Inequality constrains
- Scalable BO: high dimensional problems.
- Scalable BO: parallel approaches.
- Non-myopic methods.
- Applications: molecule design.

Non-stationary Bayesian Optimization

[Snoek et al., 2014]

The beta distributions allows for a rich family of transformations.

$$
\begin{aligned}
w_d(\mathbf{x}_d) &= \text{BetaCDF}(\mathbf{x}_d; \alpha_d, \beta_d)\,, \\
&= \int_0^{\mathbf{x}_d} \frac{u^{\alpha_d - 1}(1-u)^{\beta_d - 1}}{B(\alpha_d, \beta_d)} \mathrm{d}u\,,
\end{aligned}
$$

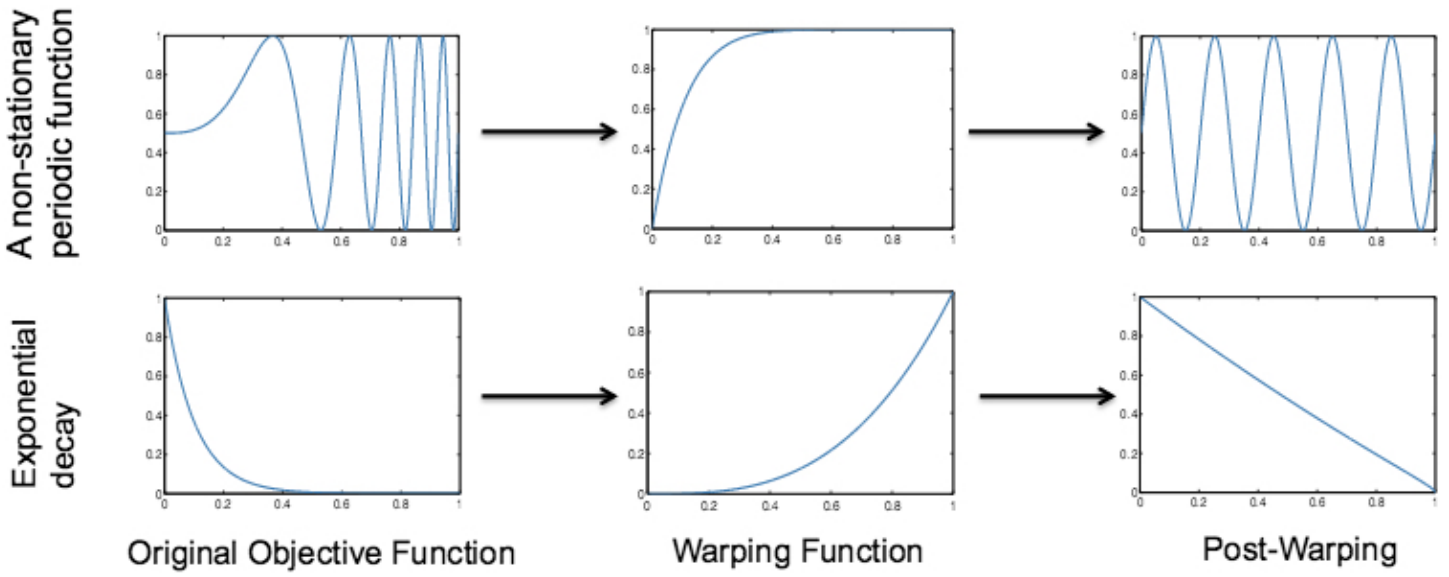


(a) Linear (b) Exponential (c) Logarithmic

Non-stationary Bayesian Optimization

[Snoek et al., 2014]

Idea: transform the function to make it stationary.

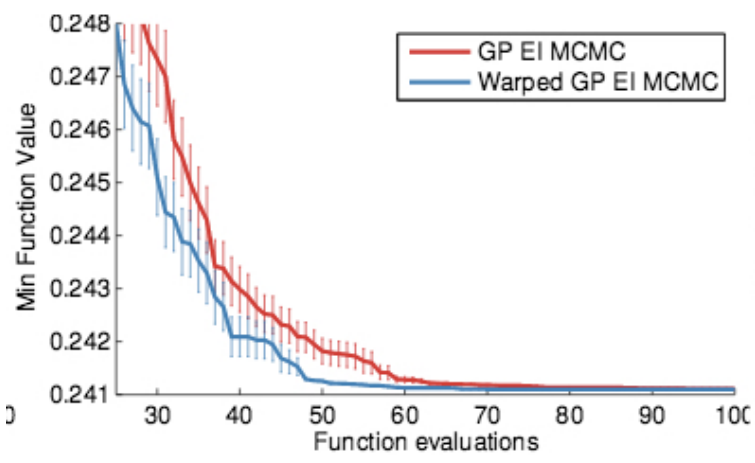


Non-stationary Bayesian Optimization

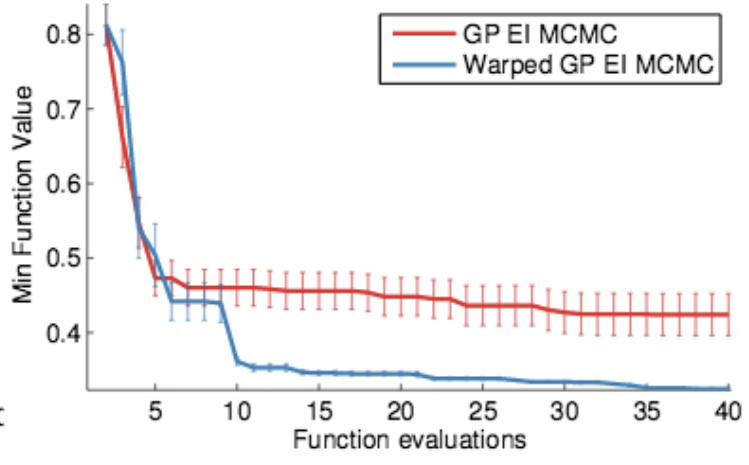
[Snoek et al., 2014]

Results improve in many experiments by warping the inputs.

Extensions to multi-task warping.



(c) Structured SVM



(d) Cifar 10 Subset

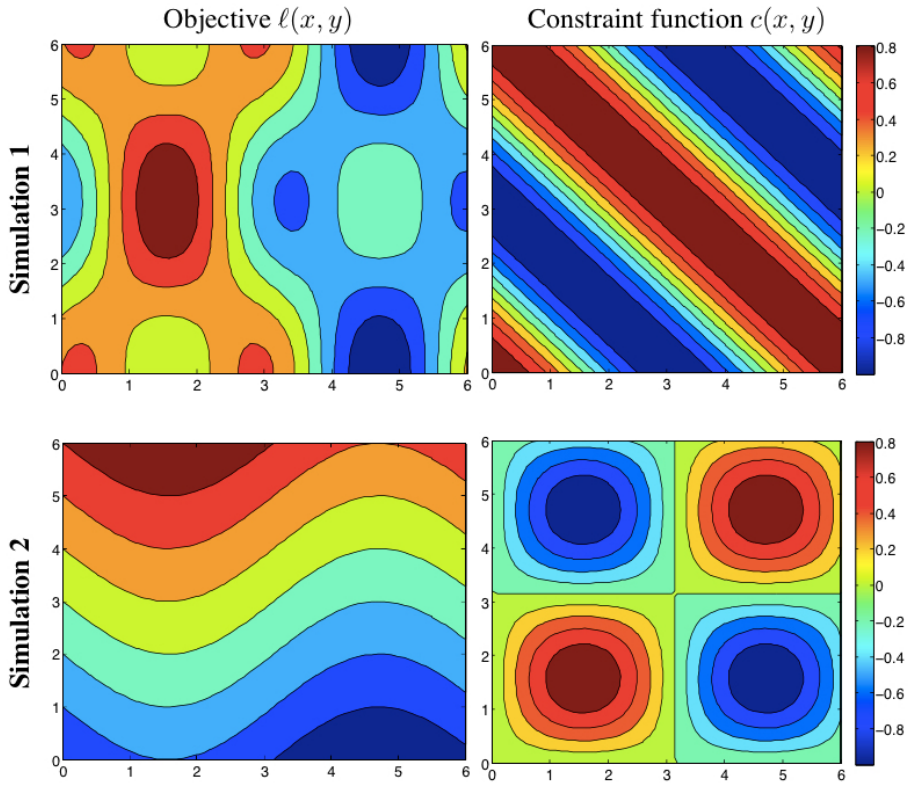
Challenges and extensions in Bayesian Optimization

- Multi-task Bayesian optimization.
- Non-stationary Bayesian optimization.
- **Inequality constrains**
- Scalable BO: high dimensional problems.
- Scalable BO: parallel approaches.
- Non-myopic methods.
- Applications: molecule design.

Inequality Constraints

[Gardner et al., 2014]

In many optimization problems the domain of the function is not an hypercube.



Inequality Constraints

[Gardner et al., 2014]

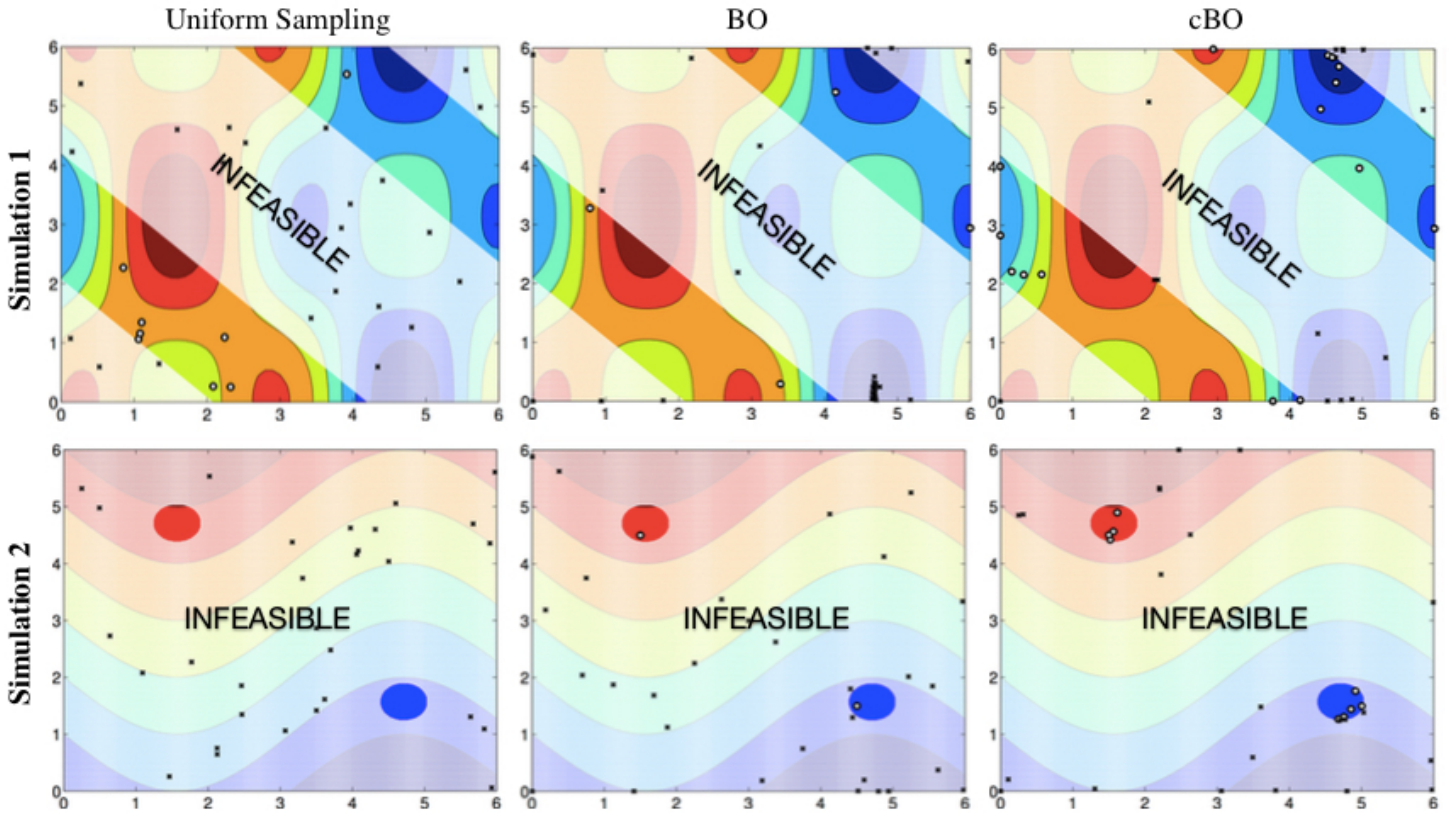
An option is to penalize the EI with an indicator function that vanishes the acquisition out the domain of interest.

$$I_C(\hat{\mathbf{x}}) = \Delta(\hat{\mathbf{x}}) \max\left\{0, \ell(\mathbf{x}^+) - \ell(\hat{\mathbf{x}})\right\} = \Delta(\hat{\mathbf{x}}) I(\hat{\mathbf{x}})$$

Inequality Constraints

[Gardner et al., 2014]

Much more efficient than standard approaches.



Challenges and extensions in Bayesian Optimization

- Multi-task Bayesian optimization.
- Non-stationary Bayesian optimization.
- Inequality constrains
- **Scalable BO: high dimensional problems.**
- Scalable BO: parallel approaches.
- Non-myopic methods.
- Applications: molecule design.

Scalable BO: REMBO

[Wang et al., 2013]

Bayesian Optimization in a Billion Dimensions via Random Embeddings

Ziyu Wang ZIYUW@CS.UBC.CA
University of British Columbia

Masrour Zoghi M.ZOGHI@UVA.NL
University of Amsterdam

David Matheson DAVIDM@CS.UBC.CA
University of British Columbia

Frank Hutter HUTTER@CS.UBC.CA
University of British Columbia

Nando de Freitas NANDO@CS.UBC.CA
University of British Columbia

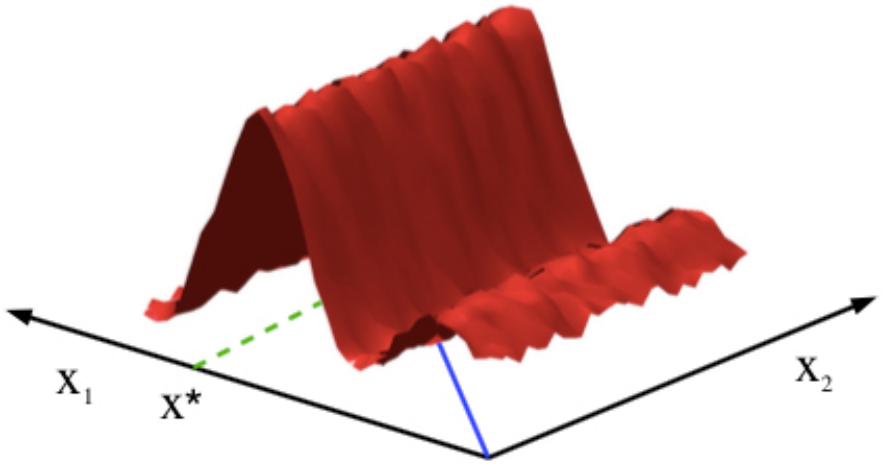
Scalable BO: REMBO

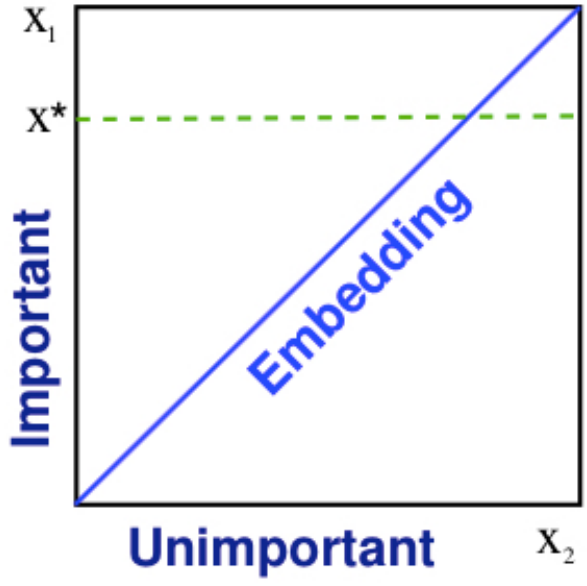
[Wang et al., 2013]

A function $f : \mathcal{X} \to \mathfrak{R}$ is called to have effective dimensionality d with $d \leq D$ if there exist a linear subspace $\mathcal{T}$ of dimension d such that for all $x_\perp \subset \mathcal{T}$ and $x_\top \subset \mathrm{T}^\top \subset \mathcal{T}$ we have $f(x_\perp) = f(x_\perp + x_\top)$ where $\mathcal{T}^\top$ is the orthogonal complement of $\mathcal{T}$.

Scalable BO: REMBO

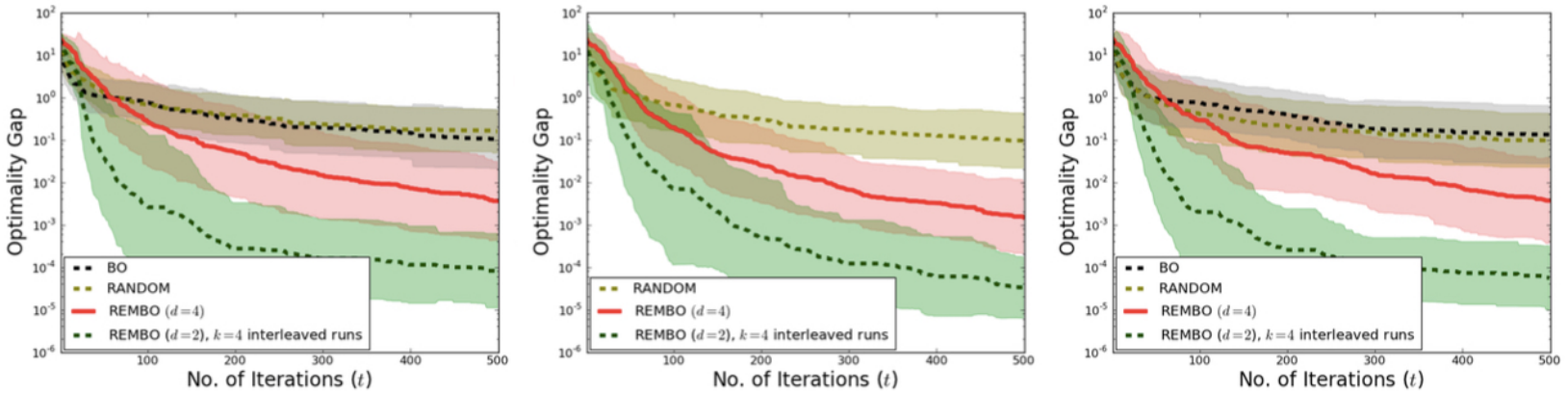
[Wang et al., 2013]





Scalable BO: REMBO

[Wang et al., 2013]



- Better in cases in the which the intrinsic dimensionality of the function is low.
- Hard to implement (need to define the bounds of the optimization after the embedding).

Scalable BO: Additive models

Use the Sobol-Hoeffding decompostion

$$f(x) = f_0 + \sum_{i=1}^{D} f_i(x_i) + \sum_{i<j} f_{ij}(x_i, x_j) + \cdots + f_{1,\ldots,D}(x)$$

where

- $f_0 = \int_{\mathcal{X}} f(x)dx$
- $f_i(x_i) = \int_{\mathcal{X}_{-i}} f(x)dx_{-i}$ - f_0
- etc...

and assume that the effects of high order than q are null.

Scalable BO: Additive models

High Dimensional Bayesian Optimisation and Bandits via Additive Models

Kirthevasan Kandasamy KANDASAMY@CS.CMU.EDU
Jeff Schneider SCHNEIDE@CS.CMU.EDU
Barnabás Póczos BAPOCZOS@CS.CMU.EDU
Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

Bayesian Optimisation (BO) is a technique used in optimising a D-dimensional function which is typically expensive to evaluate. While there have been many successes for BO in low dimensions, scaling it to high dimensions has been notoriously difficult. Existing literature on the topic are under very restrictive settings. In this paper, we identify two key challenges in this endeavour. We tackle these challenges by assuming an addi-

Bayesian Optimisation (Mockus & Mockus, 1991) refers to a suite of methods that tackle this problem by modeling f as a Gaussian Process (GP). In such methods the challenge is two fold. At time step t, first estimate the unknown f from the query value-pairs. Then use it to intelligently query at $\mathbf{x}_t$ where the function is likely to be high. For this, we first we use the posterior GP to construct an acquisition function φ_t which captures the value of the experiment. Then we maximise φ_t to determine $\mathbf{x}_t$.

Gaussian process bandits and Bayesian optimisation (GPB/

Challenges and extensions in Bayesian Optimization

- Multi-task Bayesian optimization.
- Non-stationary Bayesian optimization.
- Inequality constrains
- Scalable BO: high dimensional problems.
- **Scalable BO: parallel approaches.**
- Non-myopic methods.
- Applications: molecule design.

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.
- $\alpha(\mathbf{x}; \mathcal{I}_n)$: generic acquisition function given $\mathcal{I}_n$.

Selecting $\mathbf{x}_{t,k}$, the k-th element of the t-th batch

Sequential policy: Maximize:

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

Selecting $\mathbf{x}_{t,k}$, the k-th element of the t-th batch

Sequential policy: Maximize:

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

Greedy batch policy: it is not tractable: Maximize:

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

- $p(y_{t,j}|\mathbf{x}_j, \mathcal{I}_{t,j-1})$: 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; Azimi et al., 2011; Azimi et al., 2012; 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? BBO-LP

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"

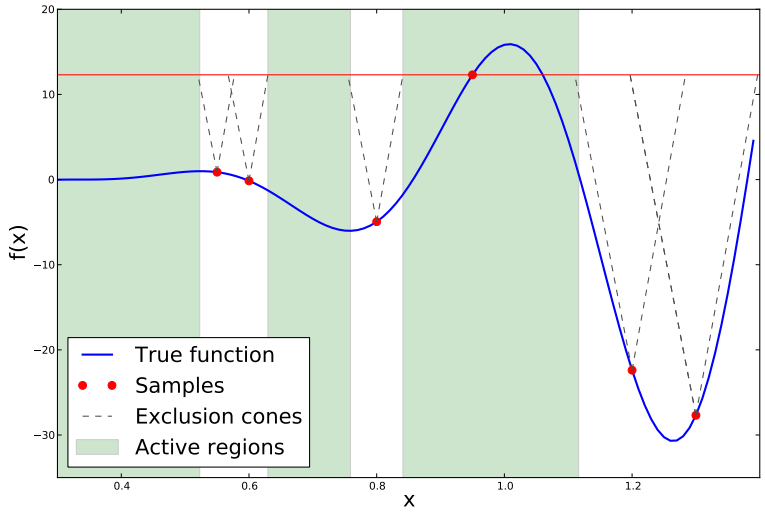
Lipschitz continuity:

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

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 $\sigma^2(r_{x_j}) = \frac{\sigma^2(\mathbf{x}_j)}{L^2}$.

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

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

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

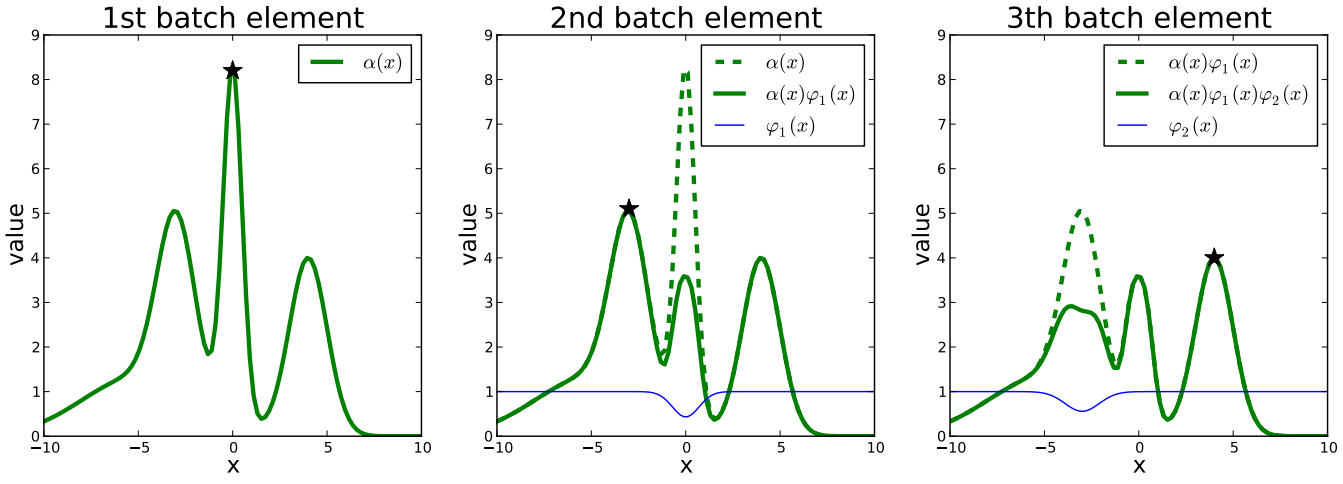
$$\int \alpha(\mathbf{x}; \mathcal{I}_{t,k-1}) \prod_{j=1}^{k-1} p(y_{t,j}|\mathbf{x}_{t,j}, \mathcal{I}_{t,j-1}) p(\mathbf{x}_{t,j}|\mathcal{I}_{t,j-1}) d\mathbf{x}_{t,j} dy_{t,j}$$

Proposal: maximization-penalization.

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

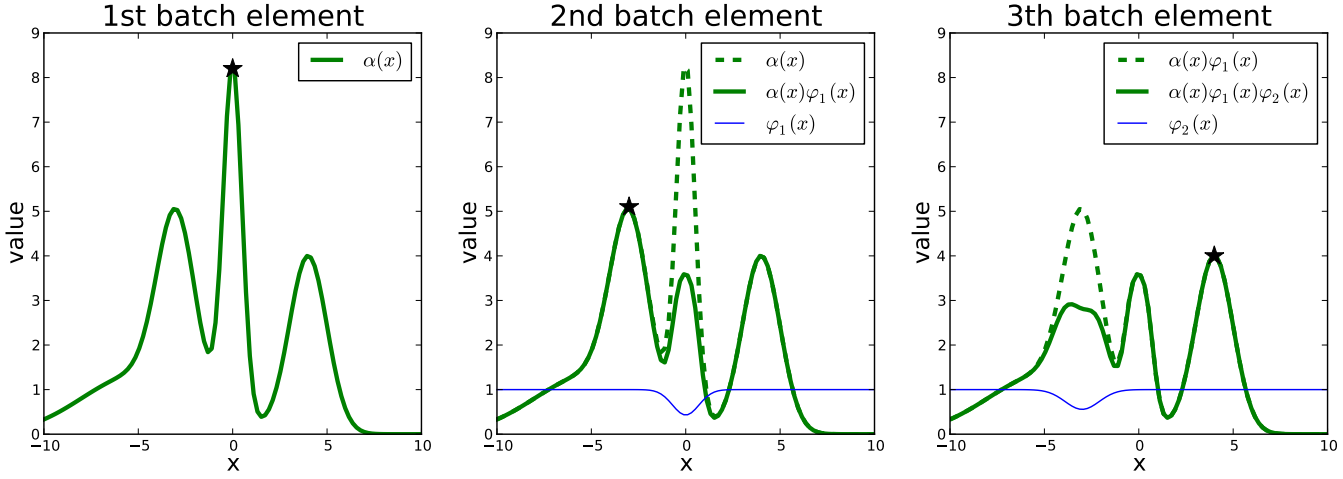
Local penalization strategy

[González, Dai, Hennig, Lawrence, 2015]



Local penalization strategy

[González, Dai, Hennig, Lawrence, 2015]

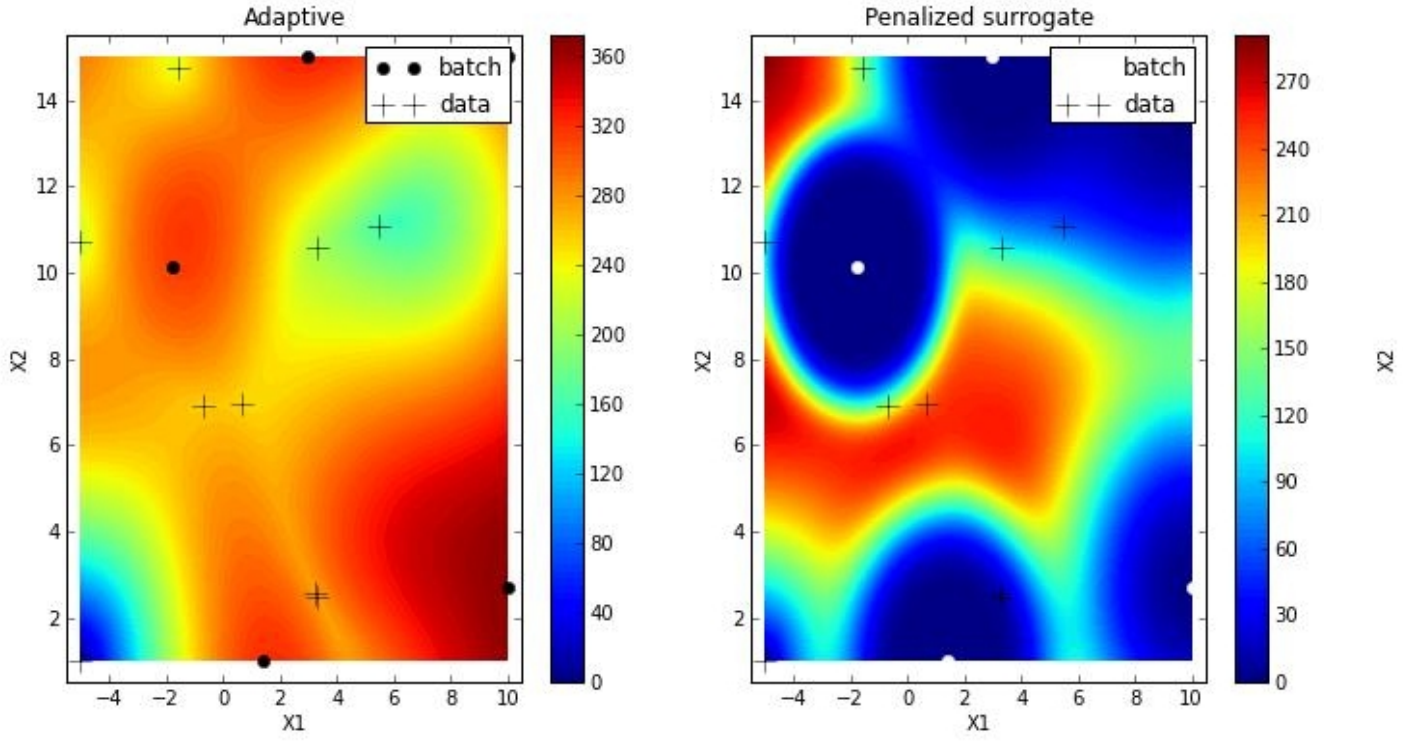


The maximization-penalization strategy selects $\mathbf{x}_{t,k}$ as

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

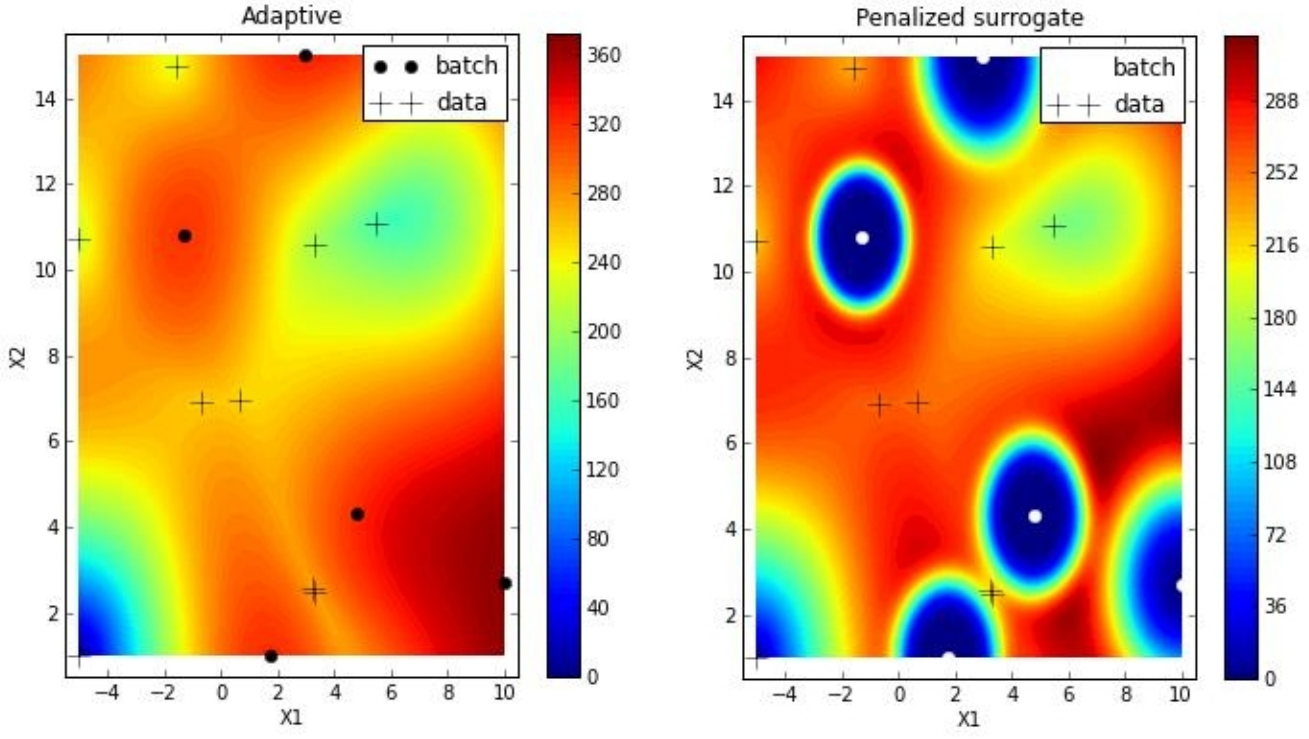
g is a transformation of $\alpha(\mathbf{x}; \mathcal{I}_{t,0})$ to make it always positive.

Example for $L = 50$



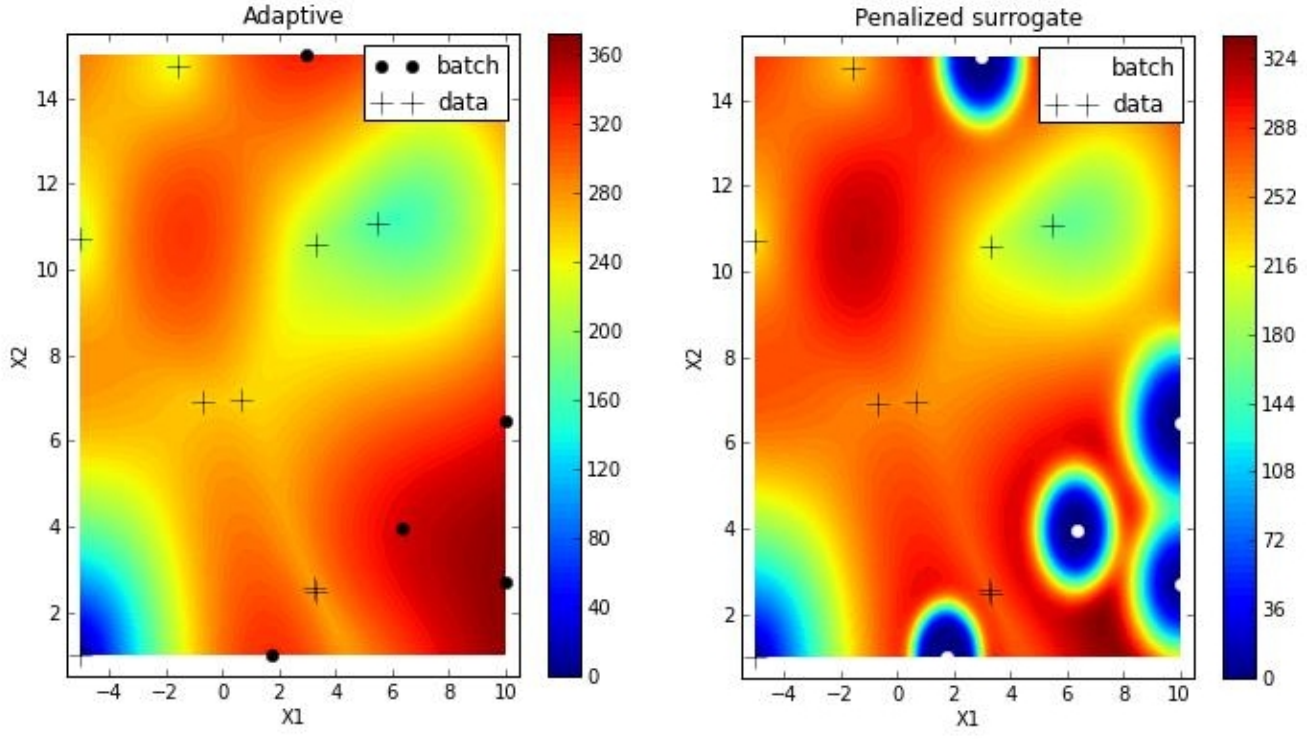
L controls the exploration-exploitation balance within the batch.

Example for $L = 100$



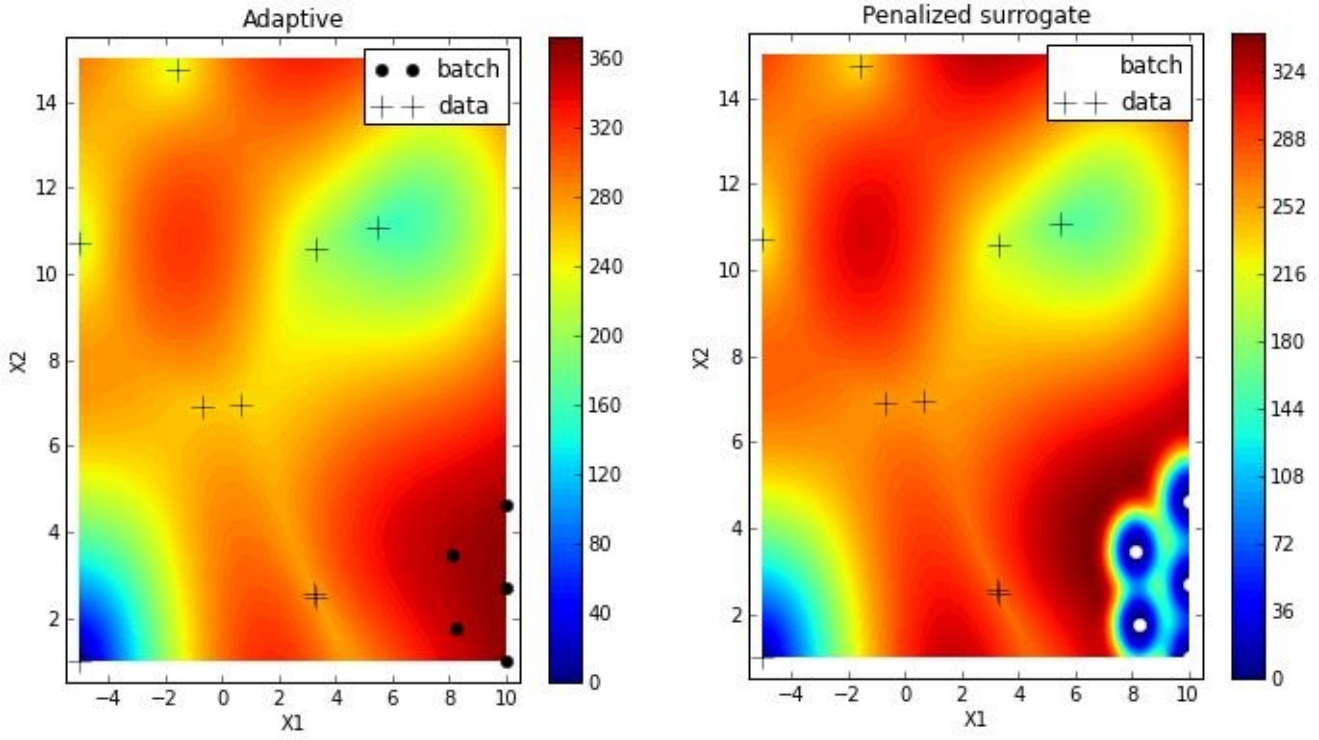
L controls the exploration-exploitation balance within the batch.

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:

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

Sobol function

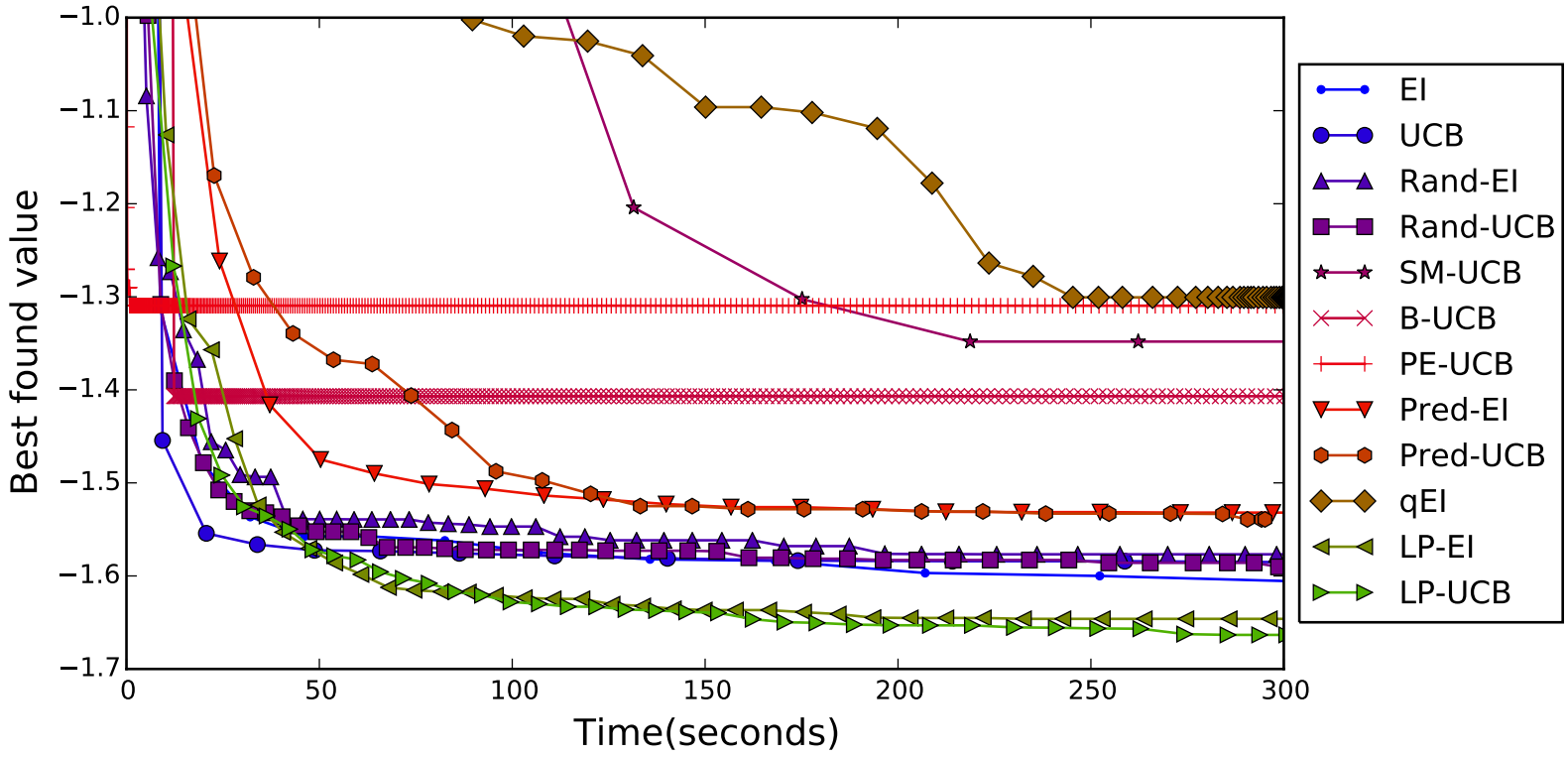
Best (average) result for some given time budget.

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**

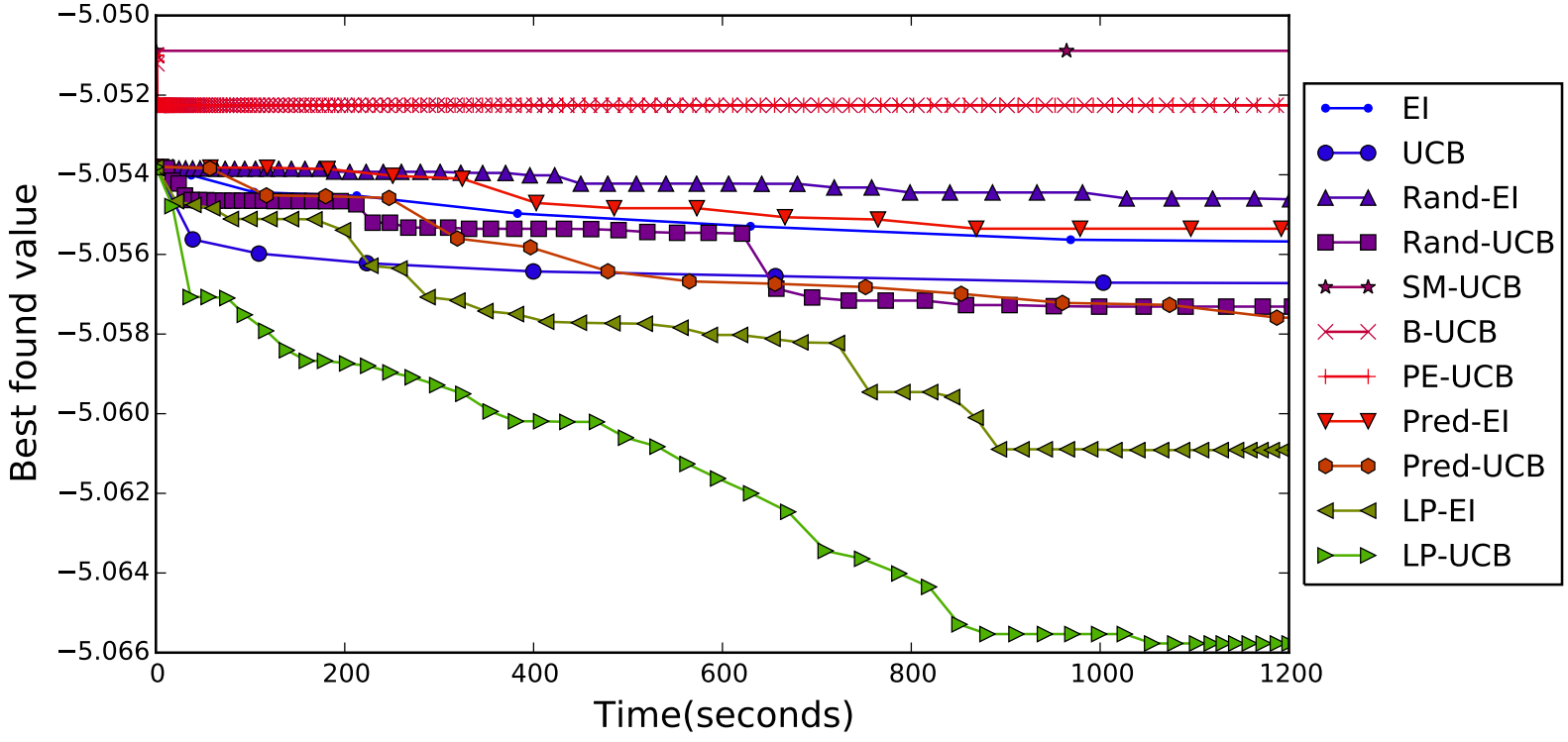
2D experiment with ‘large domain’

Comparison in terms of the wall clock time



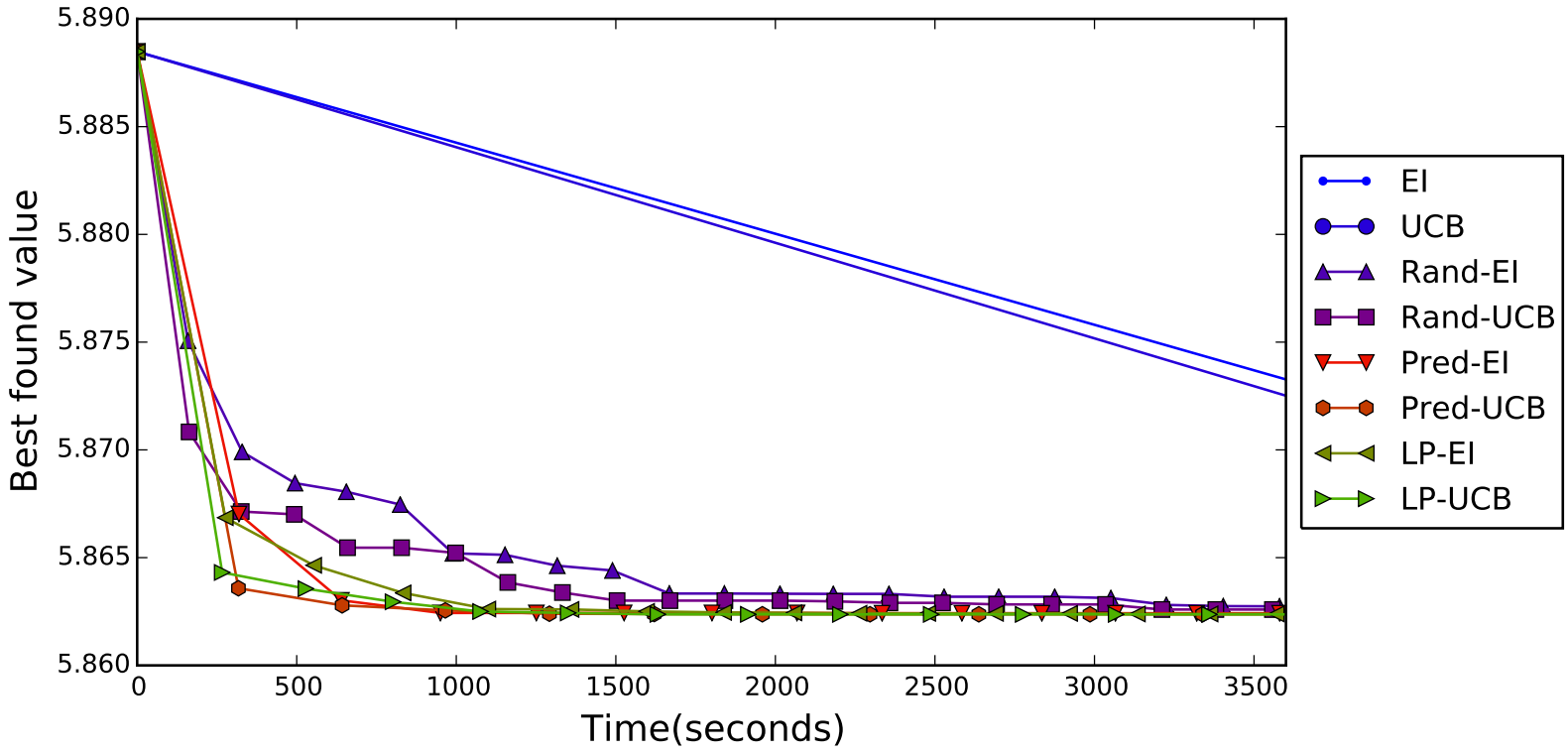
Maximizing gene translation

- Maximization of a 70 dimensional surface representing the efficiency of hamster cells producing proteins.



Support Vector Regression

- Minimization of the RMSE on a test set over 3 parameters.
- 'Physiochemical' properties of protein tertiary structure?.
- 45730 instances and 9 continuous attributes.



Challenges and extensions in Bayesian Optimization

- Multi-task Bayesian optimization.
- Non-stationary Bayesian optimization.
- Inequality constrains
- Scalable BO: high dimensional problems.
- Scalable BO: parallel approaches.
- **Non-myopic methods.**
- Applications: molecule design.

Non myopic methods

- 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: the global optimisation of an function by considering the significance of the next function evaluation on function evaluations (steps) further into the future.

Myopic loss

Denote by $\eta = \min\{\mathbf{y}_0\}$, the current best found value. We can define the loss of evaluating f this last time at $\mathbf{x}_*$ assuming it is returning y_* as

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

Its expectation is

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

The myopic loss has closed form under Gaussian likelihoods

$$\begin{aligned}\Lambda_1(\mathbf{x}_*|\mathcal{I}_0) &\triangleq \eta \int_{\eta}^{\infty} \mathcal{N}(y_*; \mu, \sigma^2) \mathrm{d}y_* \\ &+ \int_{-\infty}^{\eta} y_* \mathcal{N}(y_*; \mu, \sigma^2) dy_* \\ &= \eta + (\mu - \eta)\Phi(\eta; \mu, \sigma^2) - \sigma^2 \mathcal{N}(\eta, \mu, \sigma^2),\end{aligned}$$

where we have abbreviated $\sigma^2(y_*|\mathcal{I}_0)$ as σ^2 and $\mu(y_*|\mathcal{I}_0)$ as μ.

Looking many steps ahead

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

where

$$p(y_j|\mathbf{x}_j, \mathcal{I}_{j-1}) = \mathcal{N}\left(y_j; \mu(\mathbf{x}_j; \mathcal{I}_{j-1}), \sigma^2(\mathbf{x}_j|\mathcal{I}_{j-1})\right)$$

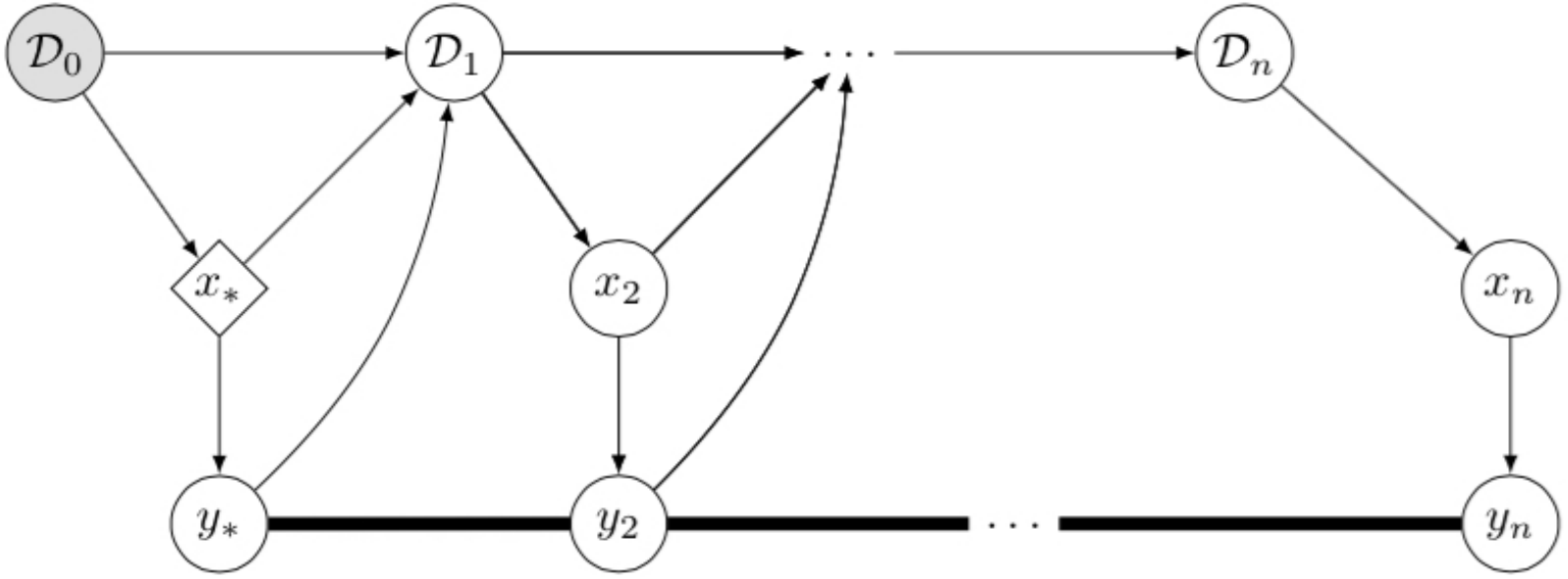
is the predictive distribution of the GP at $\mathbf{x}_j$ and

$$p(\mathbf{x}_j|\mathcal{I}_{j-1}) = \delta\Big(\mathbf{x}_j - \arg\min_{\mathbf{x}_* \in \mathcal{X}} \Lambda_{n-j+1}(\mathbf{x}_*|\mathcal{I}_{j-1})\Big)$$

reflects the optimization step.

Looking many steps ahead

Graphical model representing the decision process of a myopic loss.



Problems

- The myopic loss is very expensive to compute.
- As in the batch Bayesian optimization cases, it requires to iterative solve an expectation-optimization problem.

Relieving the myopia of Bayesian optimization

Relieving the myopia of Bayesian optimization

We present...

Relieving the myopia of Bayesian optimization

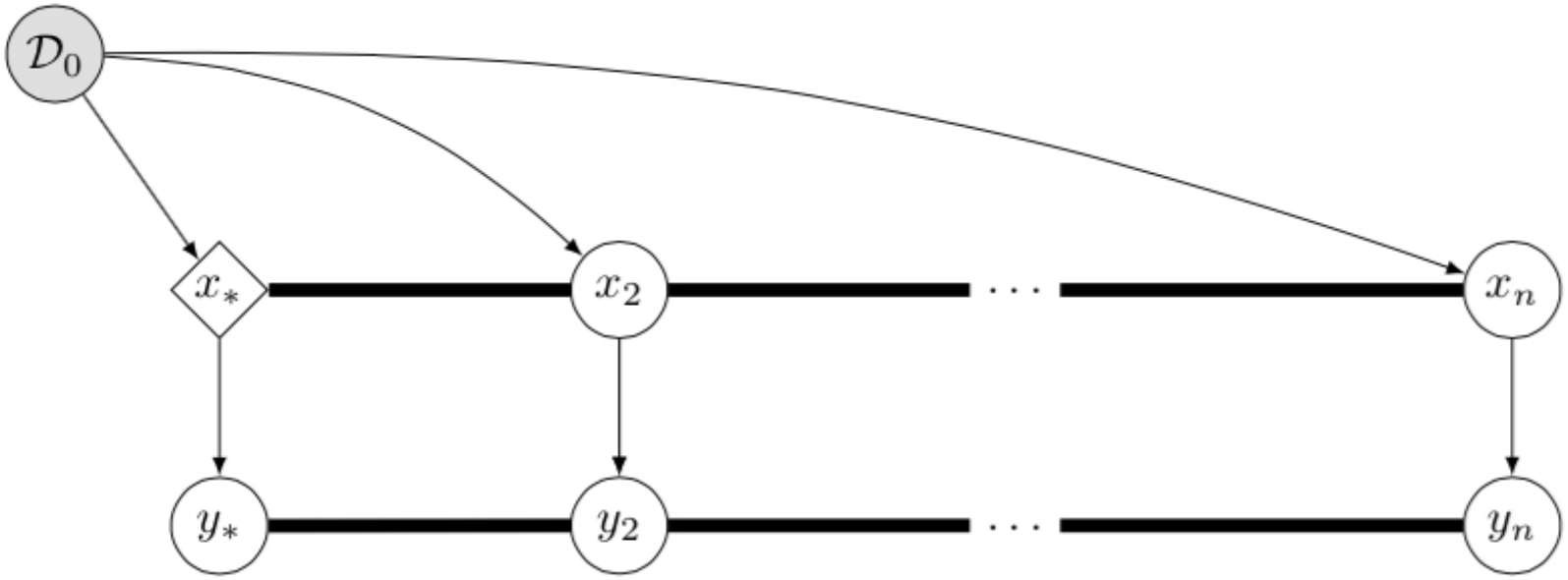
We present... GLASSES!

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

[Gonzalez, Osborne, Lawrence, 2015]

GLASSES

Idea: jointly model the epistemic uncertainty in all steps ahead.



GLASSES

$p(\mathbf{x}_2, \ldots, \mathbf{x}_n | \mathcal{I}_0, \mathbf{x}_*)$: joint probability distribution over the steps ahead:

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

- $\mathbf{y} = \{y_*, \ldots, \ldots, y_n\}$ the vector of future evaluations of f.
- $\mathbf{X}$ the $(n-1) \times q$ dimensional matrix whose rows are the future evaluations $\mathbf{x}_2, \ldots, \mathbf{x}_n$.
- $p(\mathbf{y} | \mathbf{X}, \mathcal{I}_0, \mathbf{x}_*)$ is multivariate Gaussian.

GLASSES

- Select a good $p(\mathbf{X}|\mathcal{I}_0, \mathbf{x}_*)$ is complicated.
- We fix some $\mathbf{x}$: the result of some oracle $\mathcal{F}_n(\mathbf{x}_*)$.
- Denote by $\mathbf{y} = (y_*, \ldots, \ldots, y_n)^T$ the vector of future locations evaluations of f at $\mathcal{F}_n(\mathbf{x}_*)$.
- It is possible to rewrite the expected loss $\Lambda_n\Big(\mathbf{x}_* \mid \mathcal{I}_0, \mathcal{F}_n(\mathbf{x}_*)\Big)$ as

$$\Lambda_n\Big(\mathbf{x}_* \mid \mathcal{I}_0, \mathcal{F}_n(\mathbf{x}_*)\Big) = \mathbb{E}\Big[\min(\mathbf{y}, \eta)\Big]$$

GLASSES: Computing the value of the expected loss

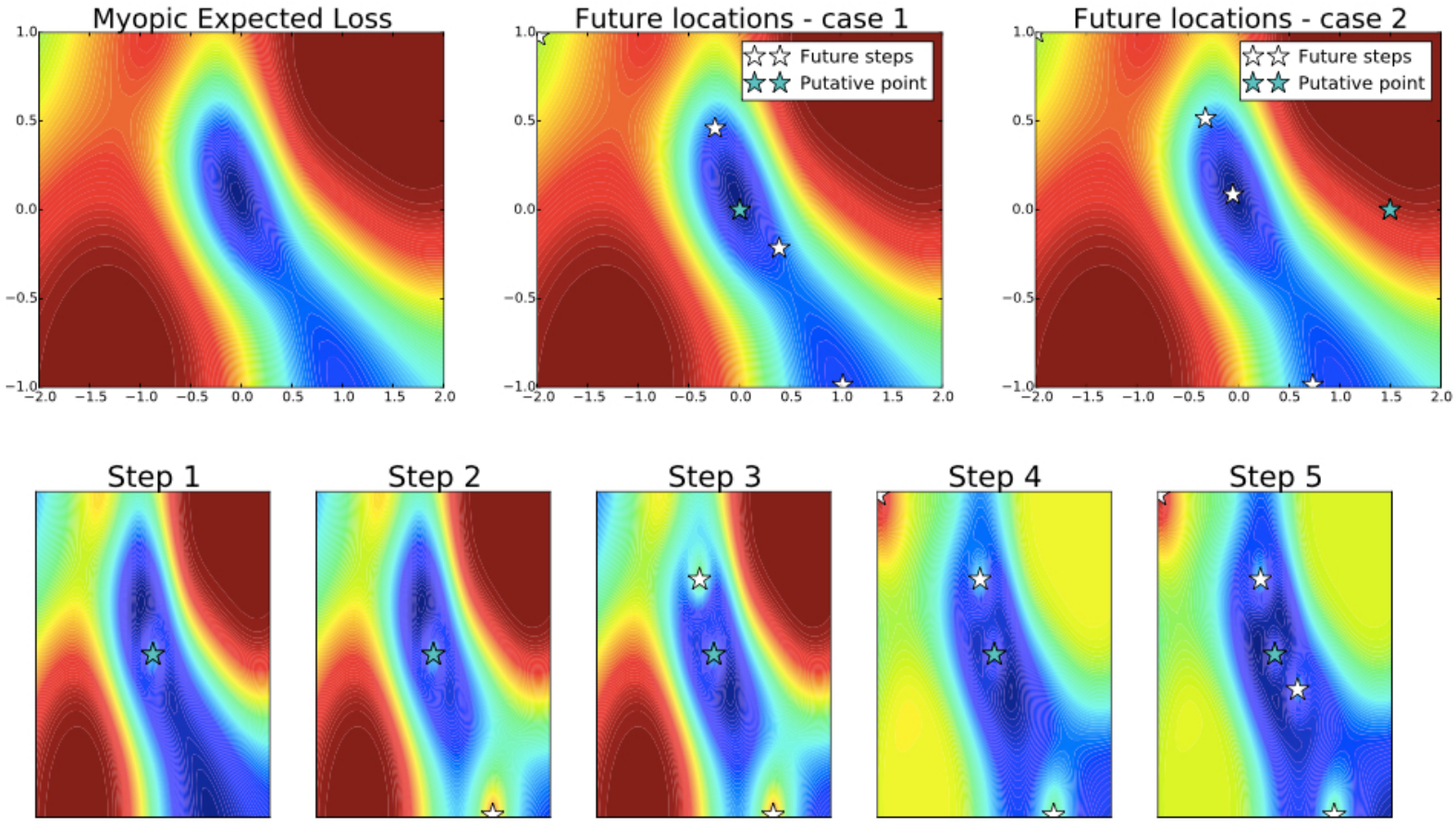
Use Expectation Propagation observing that

$$\begin{aligned}\mathbb{E}[\min(\mathbf{y},\eta)] &= \eta \int_{\mathbb{R}^n} \prod_{i=1}^{n} h_i(\mathbf{y}) N(\mathbf{y};\mu,\Sigma)\mathrm{d}\mathbf{y} \\ &+ \sum_{j=1}^{n} \int_{\mathbb{R}^n} y_j \prod_{i=1}^{n} t_{j,i}(\mathbf{y}) N(\mathbf{y};\mu,\Sigma)\mathrm{d}\mathbf{y}\end{aligned} \tag{0}$$

where $h_i(\mathbf{y}) = \mathbb{I}\{y_i > \eta\}$ and

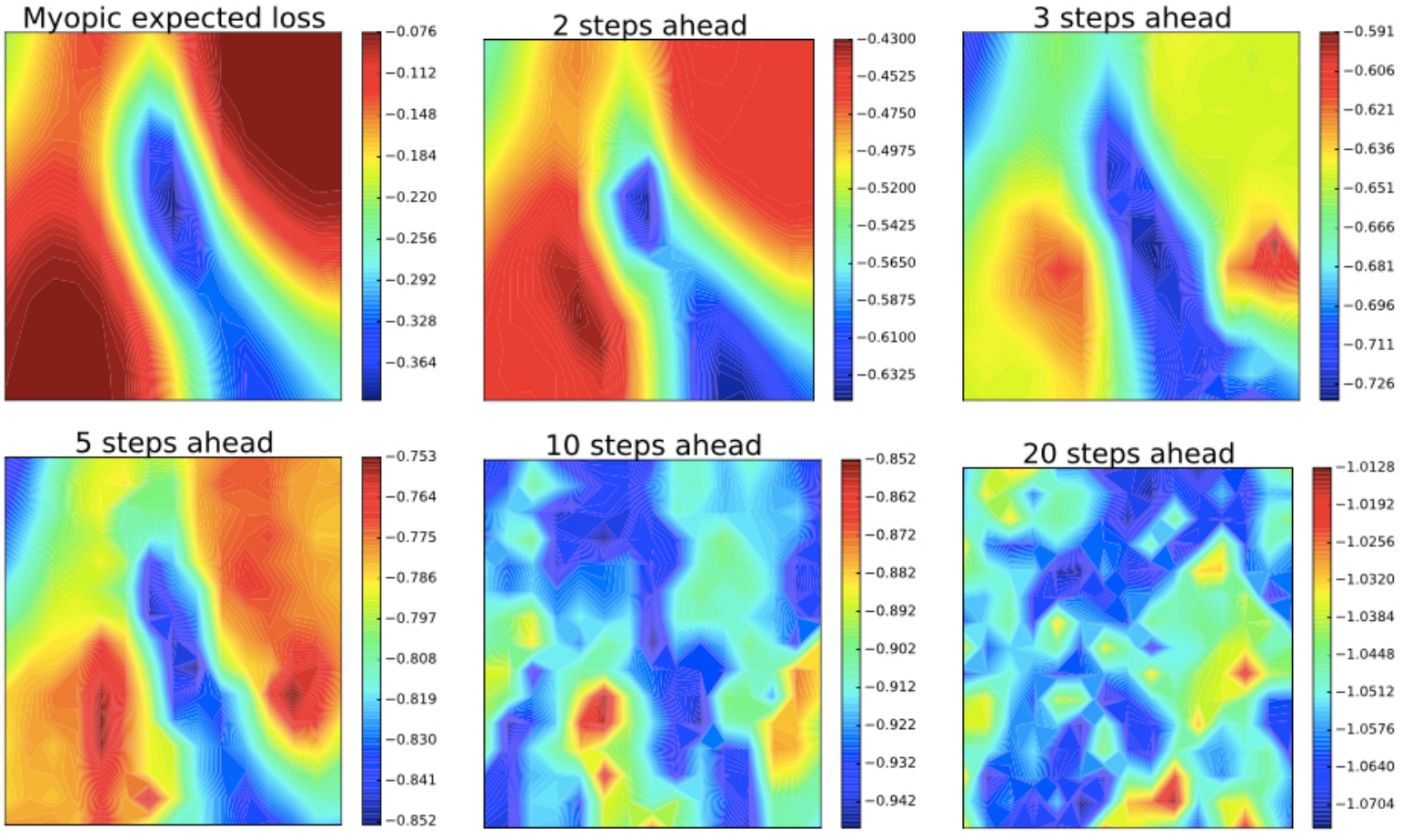
$$t_{j,i}(\mathbf{y}) = \begin{cases} \mathbb{I}\{y_j \leq \eta\} & \text{if i=j} \\ \mathbb{I}\{0 \leq y_i - y_j\} & \text{otherwise.} \end{cases}$$

GLASSES: predicting the steps ahead



To predict the steps ahead we use the batch method in [Gonzalez, Dai, Hennig and Lawrence, 2015]

GLASSES: loss function



The more steps remain, the more explorative is the non-myopic loss.

GLASSES: results

	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

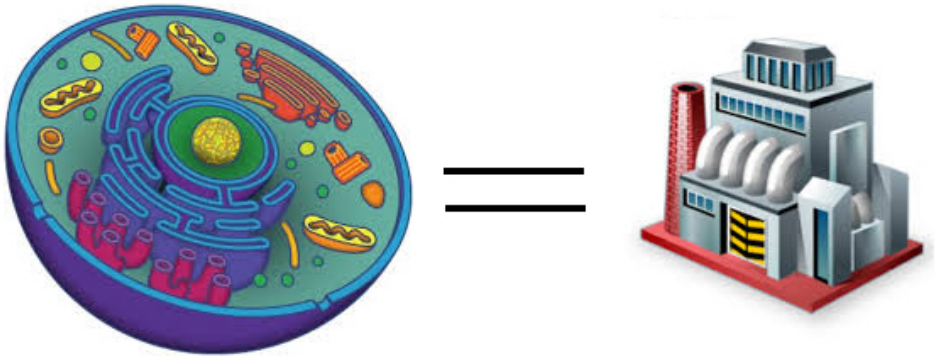
GLASSES is overal the best method.

Make sense to use GLASSES!

Challenges and extensions in Bayesian Optimization

- Multi-task Bayesian optimization.
- Non-stationary Bayesian optimization.
- Inequality constrains
- Scalable BO: high dimensional problems.
- Scalable BO: parallel approaches.
- Non-myopic methods.
- **Applications: molecule design.**

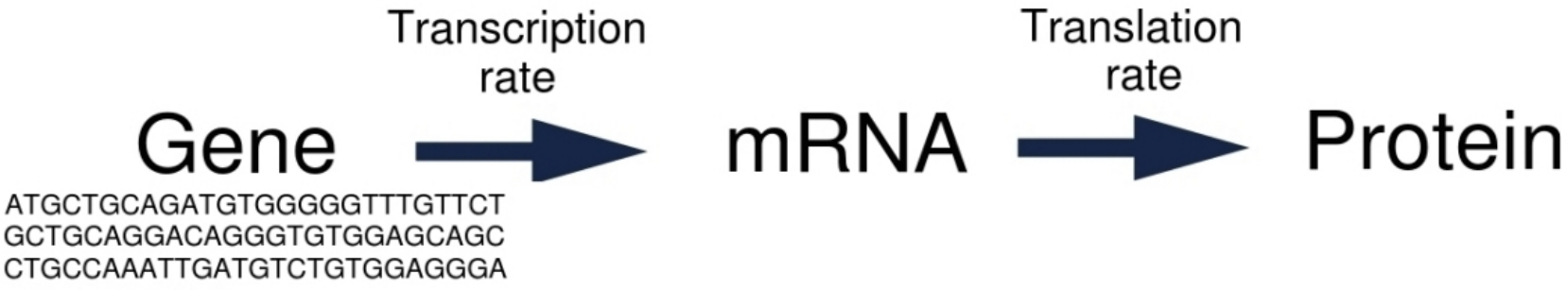
Application: Synthetic gene design



- Use mammalian cells to make protein products.
- Control the ability of the cell-factory to use synthetic DNA.

Optimize genes (ATTGGTUGA...) to best enable the cell-factory to operate most efficiently [González et al. 2014].

Central dogma of molecular biology



Big question

Remark: 'Natural' gene sequences are not necessarily optimized to maximize protein production.

ATGCTGCAGATGTGGGGGTTTGTTCTCTATCTCTTCCTGAC
TTTGTTCTCTATCTCTTCCTGACTTTGTTCTCTATCTCTTC...

Considerations

- Different gene sequences → same protein.
- The sequence affects the synthesis efficiency.

Which is the most efficient sequence to produce a protein?

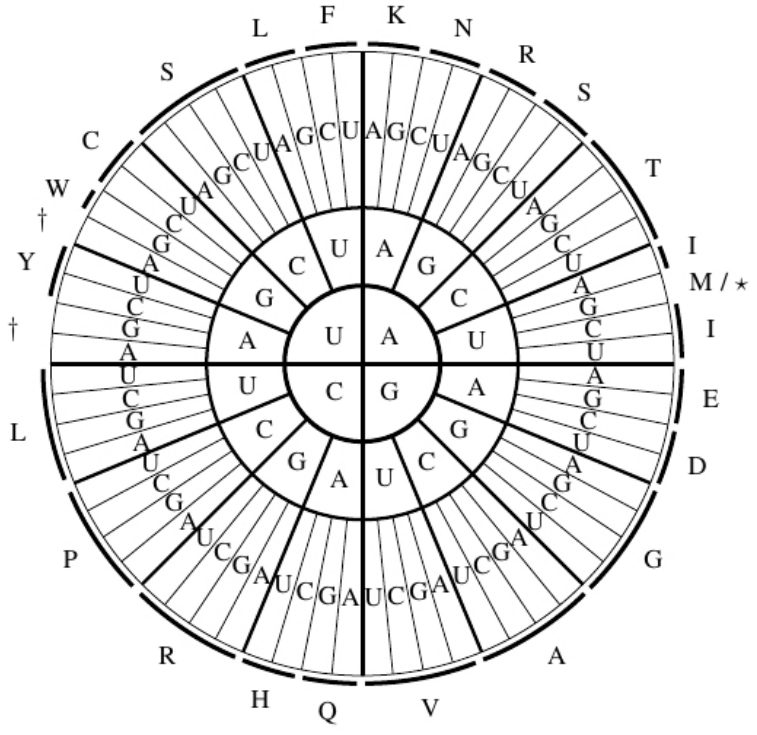
Redundancy of the genetic code

- Codon: Three consecutive bases: AAT, ACG, etc.
- Protein: sequence of amino acids.
- Different codons may encode the same aminoacid.
- ACA=ACU encodes for Threonine.

$$ATUUUGACA = ATUUUGACU$$

synonyms sequences $\rightarrow$ same protein but different efficiency

Redundancy of the genetic code



How to design a synthetic gene?

A good model is crucial—: Gene sequence features → protein production efficiency.

Bayesian Optimization principles for gene design

do:

1. Build a GP model as an emulator of the cell behavior.
2. Obtain a set of gene design rules (features optimization).
3. Design one/many new gene/s coherent with the design rules.
4. Test genes in the lab (get new data).

until the gene is optimized (or the budget is over...).

Model as an emulator of the cell behavior

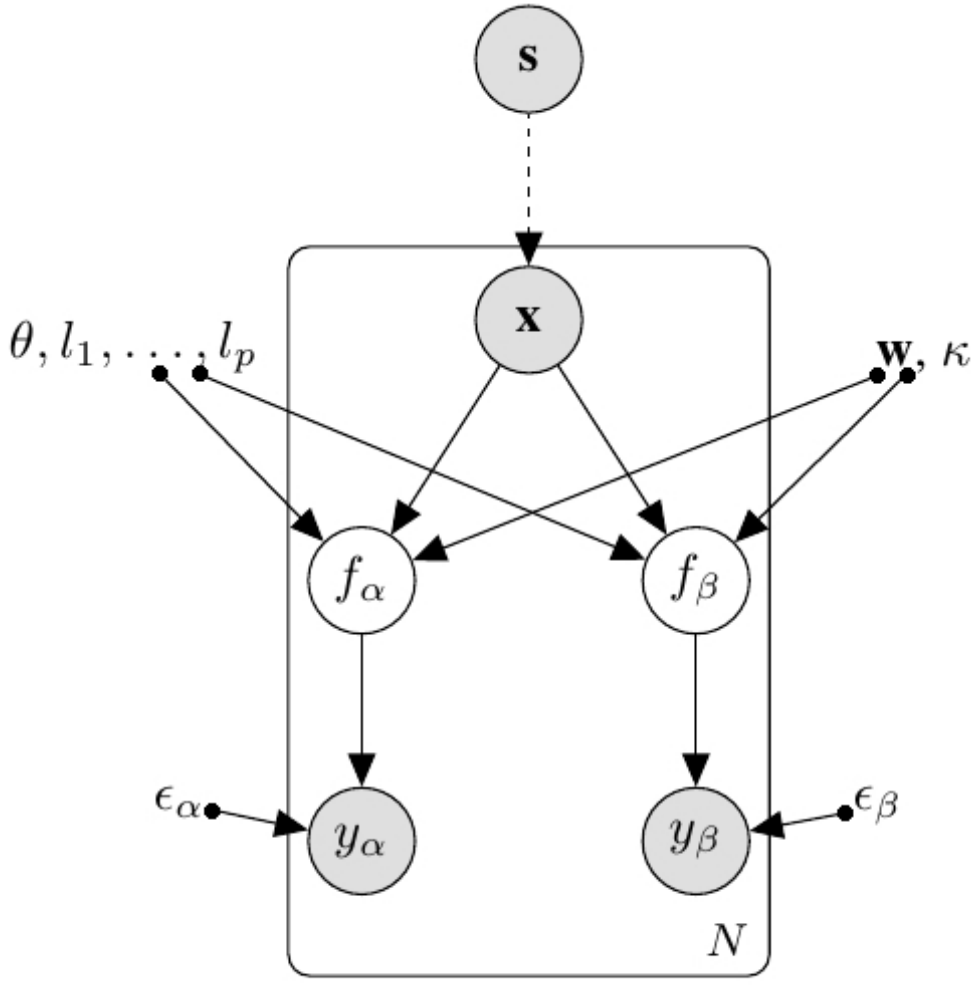
Model inputs
Features ($\mathbf{x}_i$) extracted gene sequences ($\mathbf{s}_i$): codon frequency, cai, gene length, folding energy, etc.

Model outputs
Transcription and translation rates $\mathbf{f} := (f_\alpha, f_\beta)$.

Model type
Multi-output Gaussian process $\mathbf{f} \approx \mathcal{GP}(\mathbf{m}, \mathbf{K})$ where $\mathbf{K}$ is a corregionalization covariance for the two-output model (+ SE with ARD).

The correlation in the outputs help!

Model as an emulator of the cell behavior



Obtaining optimal gene design rules

Maximize the averaged EI [Swersky et al. 2013]

$$\alpha(\mathbf{x}) = \bar{\sigma}(\mathbf{x})(-u\Phi(-u) + \phi(u))$$

where $u = (y_{max} - \bar{m}(\mathbf{x}))/\bar{\sigma}(x)$ and

$$\bar{m}(\mathbf{x}) = \frac{1}{2}\sum_{l=\alpha,\beta} \mathbf{f}_*(\mathbf{x}),\ \bar{\sigma}^2(\mathbf{x}) = \frac{1}{2^2}\sum_{l,l'=\alpha,\beta} (\mathbf{K}_*(\mathbf{x},\mathbf{x}))_{l,l'}.$$

A batch method is used when several experiments can be run in parallel

Designing new genes coherent with the optimal design rules

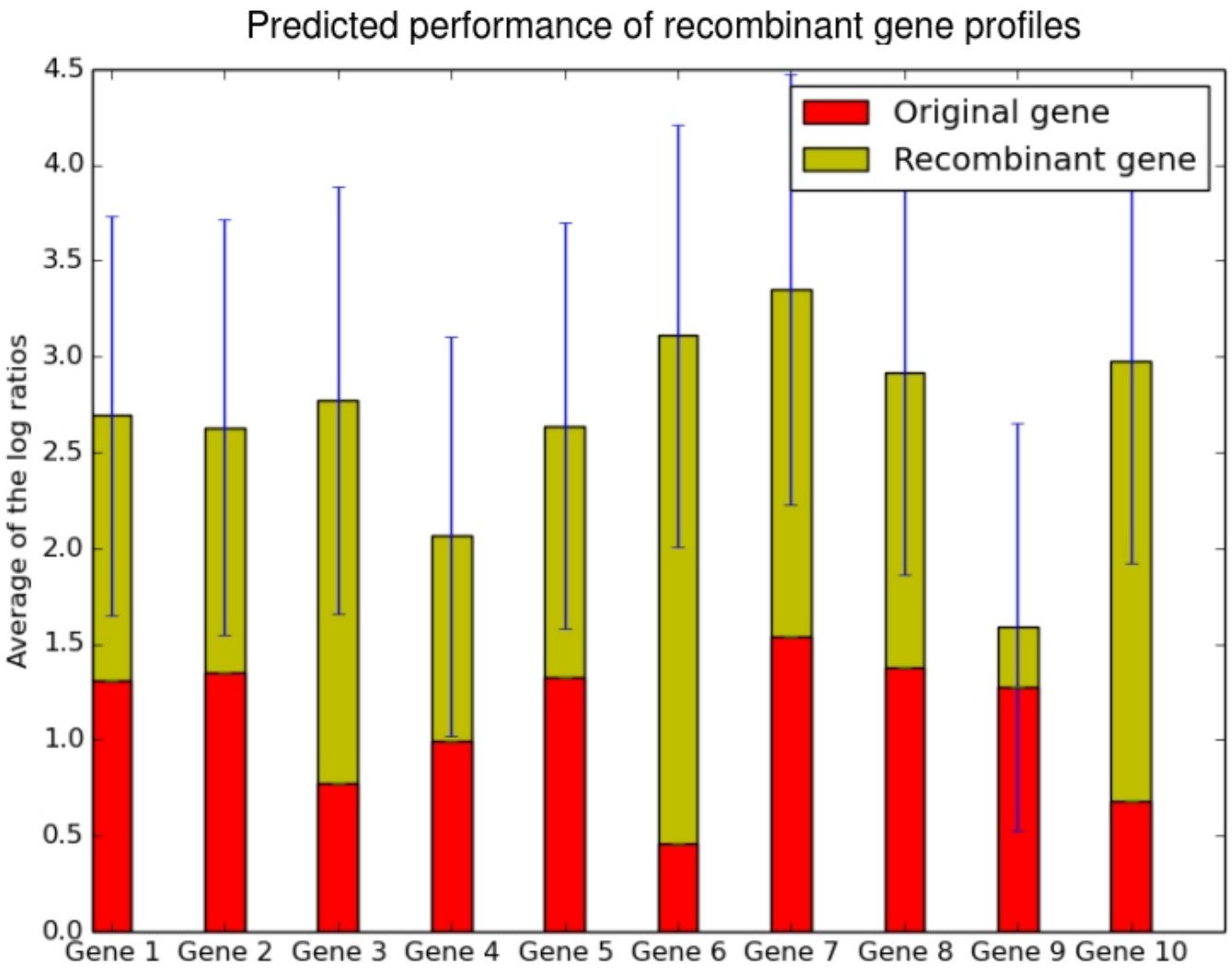
Simulating-matching approach:

1. Simulate genes 'coherent' with the target (same amino-acids).
2. Extract features.
3. Rank synthetic genes according to their similarity with the 'optimal' design rules.

Ranking criterion: $eval(\mathbf{s}|\mathbf{x}^\star) = \sum_{j=1}^{p} w_j |\mathbf{x}_j - \mathbf{x}_j^\star|$

- $\mathbf{x}^\star$: optimal gene design rules.
- $\mathbf{s}$, $\mathbf{x}_j$ generated 'synonyms sequence' and its features.
- w_j: weights of the p features (inverse length-scales of the model covariance).

Results for 10 low-expressed genes



Wrapping up

- BO is fantastic tool for parameter optimization in ML and experimental design.
- The model and acquisition function are the two most important bits.
- Many useful extensions for BO.
- To scale BO is a current challenge.
- Software available!

Picture source: http://peakdistrictcycleways.co.uk

Use Bayesian optimization!