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

\[ \tilde{k}(x, x') = B \otimes k(x, x') \]
Multi task Bayesian Optimization
[Wersky et al., 2013]

- 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]
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.
The beta distributions allows for a rich family of transformations.

\[
w_d(x_d) = \text{BetaCDF}(x_d; \alpha_d, \beta_d),
\]

\[
= \int_0^{x_d} \frac{u^{\alpha_d-1}(1-u)^{\beta_d-1}}{B(\alpha_d, \beta_d)} \, du,
\]
Non-stationary Bayesian Optimization
[Snoek et al., 2014]

Idea: transform the function to make it stationary.

A non-stationary periodic function
Original Objective Function

Exponential decay
Warping Function

Post-Warping
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.
In many optimization problems the domain of the function is not an hypercube.
An option is to penalize the EI with an indicator function that vanishes the acquisition out the domain of interest.

$$I_C(\hat{x}) = \Delta(\hat{x}) \max \{0, \ell(x^*) - \ell(\hat{x})\} = \Delta(\hat{x})I(\hat{x})$$
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.
Bayesian Optimization in a Billion Dimensions via Random Embeddings

<table>
<thead>
<tr>
<th>Name</th>
<th>Affiliation</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ziyu Wang</td>
<td>University of British Columbia</td>
<td><a href="mailto:ZIYUW@CS.UBC.CA">ZIYUW@CS.UBC.CA</a></td>
</tr>
<tr>
<td>Masrou Zoghi</td>
<td>University of Amsterdam</td>
<td><a href="mailto:M.ZOGHI@UVA.NL">M.ZOGHI@UVA.NL</a></td>
</tr>
<tr>
<td>David Matheson</td>
<td>University of British Columbia</td>
<td><a href="mailto:DAVIDM@CS.UBC.CA">DAVIDM@CS.UBC.CA</a></td>
</tr>
<tr>
<td>Frank Hutter</td>
<td>University of British Columbia</td>
<td><a href="mailto:HUTTER@CS.UBC.CA">HUTTER@CS.UBC.CA</a></td>
</tr>
<tr>
<td>Nando de Freitas</td>
<td>University of British Columbia</td>
<td><a href="mailto:NANDO@CS.UBC.CA">NANDO@CS.UBC.CA</a></td>
</tr>
</tbody>
</table>
A function $f : \mathcal{X} \to \mathbb{R}$ 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 $x_\top \subset \mathcal{T}^\top \subset \mathcal{T}$ 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 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 decomposition

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

where

- \( f_0 = \int_{X} f(x)dx \)
- \( f_i(x_i) = \int_{X_{-i}} f(x)dx_{-i} - f_0 \)
- 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

Kirthivasan Kandasamy
Jeff Schneider
Barnabás Póczos
Carnegie Mellon University, Pittsburgh, PA, USA

KANDASAMY@CS.CMU.EDU
SCHNEIDER@CS.CMU.EDU
BAPOCZOS@CS.CMU.EDU

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 modelling $f$ 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 $x_t$ where the function is likely to be high. For this, we first we use the posterior GP to construct an acquisition function $\varphi_t$ which captures the value of the experiment. Then we maximise $\varphi_t$ to determine $x_t$.

Gaussian process bandits and Bayesian optimisation (GPB/
Challenges and extensions in Bayesian Optimization

- Multi-task Bayesian optimization.
- Non-stationary Bayesian optimization.
- Inequality constraints.
- 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 $f(x_n) = \text{cost of } \{f(x_{n,1}), \ldots, f(x_{n,nb})\}$.
- Many cores available, simultaneous lab experiments, etc.
Considerations when designing a batch

- Available pairs \( \{(x_j, y_i)\}_{i=1}^n \) are augmented with the evaluations of \( f \) on \( B_t^{nb} = \{x_{t,1}, \ldots, x_{t,nb}\} \).

- Goal: design \( B_1^{nb}, \ldots, B_m^{nb} \).

Notation:

- \( I_n \): represents the available data set \( D_n \) and the \( GP \) structure when \( n \) data points are available.

- \( \alpha(x; I_n) \): generic acquisition function given \( I_n \).
Selecting $x_{t,k}$, the k-th element of the t-th batch

**Sequential policy**: Maximize:

$$\alpha(x; I_{t,k-1})$$
Selecting $x_{t,k}$, the k-th element of the t-th batch

**Sequential policy:** Maximize:

$$\alpha(x; I_{t,k-1})$$

**Greedy batch policy:** it is not tractable: Maximize:

$$\int \alpha(x; I_{t,k-1}) \prod_{j=1}^{k-1} p(y_{t,j}|x_{t,j}, I_{t,j-1}) p(x_{t,j}|I_{t,j-1}) dx_{t,j} dy_{t,j}$$

- $p(y_{t,j}|x_{j}, I_{t,j-1})$: predictive distribution of the GP.
- $p(x_{j}|I_{t,j-1}) = \delta(x_{t,j} - \text{arg max}_{x \in \mathcal{X}} \alpha(x; I_{t,j-1}))$. 
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 $p(y_{t,j}|x_j, I_{t,j-1})$.
- Simultaneously optimize elements on the batch using the joint distribution of $y_{t_1}, \ldots y_{t,nb}$.

**Bottleneck**

All these methods require to iteratively update $p(y_{t,j}|x_j, I_{t,j-1})$ to model the iteration between the elements in the batch: $O(n^3)$

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 GP model into the batch design”

Lipschitz continuity:

$$|f(x_1) - f(x_2)| \leq L\|x_1 - x_2\|_p.$$
Interpretation of the Lipschitz continuity of $f$

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

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

$x_M \notin B_{r_{x_j}}(x_j)$ otherwise, the Lipschitz condition is violated.
Probabilistic version of $B_{r_x}(x)$

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

- $r_{x_j}$ is Gaussian with $\mu(r_{x_j}) = \frac{M - \mu(x_j)}{L}$ and $\sigma^2(r_{x_j}) = \frac{\sigma^2(x_j)}{L^2}$. 
Probabilistic version of $B_{r_x}(x)$

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

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

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

\[
\varphi(x; x_j) = p(r_{x_j} < ||x - x_j||) = 0.5 \text{erfc}(-z)
\]

where $z = \frac{1}{\sqrt{2\sigma^2_n(x_j)}}(L||x_j - x|| - M + \mu_n(x_j))$.

- Reflects the size of the 'Lipschitz' exclusion areas.
- Approaches to 1 when $x$ is far from $x_j$ and decreases otherwise.
Idea to collect the batches
Without using explicitly the model.

Optimal batch: maximization-marginalization

\[
\int \alpha(x; I_{t,k-1}) \prod_{j=1}^{k-1} p(y_{t,j}|x_{t,j}, I_{t,j-1})p(x_{t,j}|I_{t,j-1})dx_{t,j}dy_{t,j}
\]

Proposal: maximization-penalization.

Use the \(\varphi(x; x_j)\) to penalize the acquisition and predict the expected change in \(\alpha(x; I_{t,k-1})\).
Local penalization strategy
[González, Dai, Hennig, Lawrence, 2015]

The maximization-penalization strategy selects $x_t, k$ as:

$$x_t, k = \arg \max_{x \in X} \left\{ g(\alpha(x; I_t, 0)) \right\},$$

where $g$ is a transformation of $\alpha(x; I_t, 0)$ to make it always positive.
Local penalization strategy
[González, Dai, Hennig, Lawrence, 2015]

The maximization-penalization strategy selects $x_{t,k}$ as

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

$g$ is a transformation of $\alpha(x; \mathcal{I}_{t,0})$ 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 \( f : X \to \mathbb{R} \) be a \( L \)-Lipschitz continuous function defined on a compact subset \( X \subseteq \mathbb{R}^D \). Then

\[
L_p = \max_{x \in X} \| \nabla f(x) \|_p,
\]

is a valid Lipschitz constant.

The gradient of \( f \) at \( x^* \) is distributed as a multivariate Gaussian

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

We choose:

\[
\hat{L}_{GP-LCA} = \max_{x \in X} \| \mu_{\nabla}(x^*) \|
\]
Sobol function

Best (average) result for some given time budget.

<table>
<thead>
<tr>
<th>$d$</th>
<th>$n_b$</th>
<th>EI</th>
<th>UCB</th>
<th>Rand-EI</th>
<th>Rand-UCB</th>
<th>SM-UCB</th>
<th>B-UCB</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>0.31±0.03</td>
<td>0.32±0.06</td>
<td>0.32±0.05</td>
<td>0.31±0.05</td>
<td>1.86±1.06</td>
<td>0.56±0.03</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>0.67±0.31</td>
<td>0.75±0.32</td>
<td>4.40±2.97</td>
<td></td>
<td>-</td>
<td>0.57±0.01</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>8.84±3.69</td>
<td>11.89±9.44</td>
<td>9.19±5.32</td>
<td>10.59±5.04</td>
<td>137.2±113.0</td>
<td>6.01±0.00</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>1.74±1.47</td>
<td>2.20±1.85</td>
<td>2.18±2.30</td>
<td>2.76±3.06</td>
<td>-</td>
<td>2.53±0.00</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>559.1±1014</td>
<td>1463±1803</td>
<td>690.5±947.5</td>
<td>1825±2149</td>
<td>9e+04±7e+04</td>
<td>2098±0.00</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>200.9±455.9</td>
<td>1149±1830</td>
<td>639.4±1204</td>
<td>385.9±642.9</td>
<td>-</td>
<td>1656±0.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$d$</th>
<th>$n_b$</th>
<th>PE-UCB</th>
<th>Pred-EI</th>
<th>Pred-UCB</th>
<th>qEI</th>
<th>LP-EI</th>
<th>LP-UCB</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>0.99±0.74</td>
<td>0.41±0.15</td>
<td>0.45±0.16</td>
<td>1.53±0.86</td>
<td>0.35±0.11</td>
<td>0.31±0.06</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>0.66±0.29</td>
<td>1.16±0.70</td>
<td>1.26±0.81</td>
<td>3.82±2.09</td>
<td>0.66±0.48</td>
<td>0.69±0.51</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>123.5±81.43</td>
<td>10.43±4.88</td>
<td>11.77±9.44</td>
<td>15.70±8.90</td>
<td>11.85±5.68</td>
<td>10.85±8.08</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>120.8±78.56</td>
<td>9.58±7.85</td>
<td>11.66±11.48</td>
<td>17.69±9.04</td>
<td>3.88±4.15</td>
<td>1.88±2.46</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>98.60±82.60</td>
<td>8.58±8.13</td>
<td>10.86±10.89</td>
<td>-</td>
<td>6.53±4.12</td>
<td>1.44±1.93</td>
</tr>
<tr>
<td>20</td>
<td>5e+04±2e+05</td>
<td>793.0±1226</td>
<td>1412±3032</td>
<td>-</td>
<td>1881±1176</td>
<td>1194±1428</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>6e+04±8e+04</td>
<td>442.6±717.9</td>
<td>1725±3205</td>
<td>-</td>
<td>1042±1562</td>
<td>100.4±338.7</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>5e+04±4e+04</td>
<td>1091±1724</td>
<td>2231±3110</td>
<td>-</td>
<td>1249±1570</td>
<td>20.75±50.12</td>
<td></td>
</tr>
</tbody>
</table>
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 constraints
- 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 a function by considering the significance of the next function evaluation on function evaluations (steps) further into the future.
Myopic loss

Denote by $\eta = \min\{y_0\}$, the current best found value. We can define the loss of evaluating $f$ this last time at $x_*$ assuming it is returning $y_*$ as

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

Its expectation is

$$\Lambda_1(x_*|I_0) \triangleq \mathbb{E}[\min(y_*, \eta)] = \int \lambda(y_*)p(y_*|x_*, I_0)dy_*$$
The myopic loss has closed form under Gaussian likelihoods

\[ \Lambda_1(x_*|\mathcal{I}_0) \triangleq \eta \int_\eta^\infty N(y_*; \mu, \sigma^2)dy_* \]
\[ + \int_{-\infty}^\eta y_*N(y_*; \mu, \sigma^2)dy_* \]
\[ = \eta + (\mu - \eta)\Phi(\eta; \mu, \sigma^2) - \sigma^2 N(\eta, \mu, \sigma^2), \]

where we have abbreviated \( \sigma^2(y_*|\mathcal{I}_0) \) as \( \sigma^2 \) and \( \mu(y_*|\mathcal{I}_0) \) as \( \mu \).
Looking many steps ahead

\[ \Lambda_n(x_\ast|I_0) = \int \lambda(y_n) \prod_{j=1}^n p(y_j|x_j,I_{j-1})p(x_j|I_{j-1}) \]
\[ dy_\ast \ldots dy_n dx_2 \ldots dx_n \]

where

\[ p(y_j|x_j,I_{j-1}) = N \left( y_j; \mu(x_j;I_{j-1}), \sigma^2(x_j|I_{j-1}) \right) \]

is the predictive distribution of the GP at \( x_j \) and

\[ p(x_j|I_{j-1}) = \delta(x_j - \arg \min_{x_\ast \in \mathcal{X}} \Lambda_{n-j+1}(x_\ast|I_{j-1})) \]

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!

Global optimisation with Look-Ahead through Stochastic Simulation and Expected-loss Search

[Gonzalez, Osborne, Lawrence, 2015]
Idea: jointly model the epistemic uncertainty in all steps ahead.
\( p(x_2, \ldots, x_n | I_0, x_\ast) \): joint probability distribution over the steps ahead:

\[
\Gamma_n(x_\ast | I_0) = \int \lambda(y_n)p(y|X, I_0, x_\ast)p(X|I_0, x_\ast)dydX
\]

- \( y = \{y_\ast, \ldots, \ldots, y_n\} \) the vector of future evaluations of \( f \).
- \( X \) the \((n - 1) \times q\) dimensional matrix whose rows are the future evaluations \( x_2, \ldots, x_n \).
- \( p(y|X, I_0, x_\ast) \) is multivariate Gaussian.
Select a good $p(X|\mathcal{I}_0, x_*)$ is complicated.

We fix some $x$: the result of some oracle $\mathcal{F}_n(x_*)$.

Denote by $y = (y_*, \ldots, y_n)^T$ the vector of future locations evaluations of $f$ at $\mathcal{F}_n(x_*)$.

It is possible to rewrite the expected loss $\Lambda_n(x_* | \mathcal{I}_0, \mathcal{F}_n(x_*)$ as

$$\Lambda_n(x_* | \mathcal{I}_0, \mathcal{F}_n(x_*) = \mathbb{E}[\min(y, \eta)]$$
Use Expectation Propagation observing that

\[
\mathbb{E}[\min(y, \eta)] = \eta \int_{\mathbb{R}^n} \prod_{i=1}^{n} h_i(y) N(y; \mu, \Sigma) dy + \sum_{j=1}^{n} \int_{\mathbb{R}^n} y_j \prod_{i=1}^{n} t_{j,i}(y) N(y; \mu, \Sigma) dy
\]

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

\[
t_{j,i}(y) = \begin{cases} 
\mathbb{I}\{y_j \leq \eta\} & \text{if } i=j \\
\mathbb{I}\{0 \leq y_i - y_j\} & \text{otherwise.}
\end{cases}
\]
To predict the steps ahead we use the batch method in [Gonzalez, Dai, Hennig and Lawrence, 2015]
The more steps remain, the more explorative is the non-myopic loss.
GLASSES: results

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

GLASSES is overall 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

Gene $\xrightarrow{\text{Transcription rate}}$ mRNA $\xrightarrow{\text{Translation rate}}$ Protein

ATGCTGCAGATGTGGGGGTTTGTCT
GCTGCAGGACAGGGTGTGGAGCAGC
CTGCCAAAATTGTCTGTGGAGGGA
Remark: ‘Natural’ gene sequences are not necessarily optimized to maximize protein production.

ATGCTGCAGATGTGGGGGTTTGTTCTCTATCTCTTTCCCTGAC
TTGTCTCTATCTCTCTCTCCTGACCTTTTGGTTCTCTATCTCTTCC...

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 amino acid.
- ACA = ACU encodes for Threonine.

\[ \text{ATUUUGACA} = \text{ATUUUGACU} \]

synonyms sequences \(\rightarrow\) same protein but different efficiency
Redundancy of the genetic code

Gene → mRNA → Protein

ATGCTGCAGATGTGGGGGTTTGGTTCT
GCTGCAGGACAGGGTGTTGGAGCAGC
CTGCCAAATTGATGTCTGTGGAGGGA
ACCTTTTGGCTCGG
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 \((x_i)\) extracted gene sequences \((s_i)\): codon frequency, cai, gene length, folding energy, etc.

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

Model type
Multi-output Gaussian process \(f \approx \mathcal{GP}(m, K)\) where \(K\) 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]

\[ \alpha(x) = \bar{\sigma}(x)(-u \Phi(-u) + \phi(u)) \]

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

\[ \bar{m}(x) = \frac{1}{2} \sum_{l=\alpha,\beta} f_*(x), \quad \bar{\sigma}^2(x) = \frac{1}{2^2} \sum_{l,l'=\alpha,\beta} (K_*(x, x))_{l,l'} . \]

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: \( eval(s|x^*) = \sum_{j=1}^{p} w_j|x_j - x^*_j| \)

- \( x^* \): optimal gene design rules.
- \( s, x_j \): 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

Predicted performance of recombinant gene profiles

Average of the log ratios

Gene 1 | Gene 2 | Gene 3 | Gene 4 | Gene 5 | Gene 6 | Gene 7 | Gene 8 | Gene 9 | Gene 10

- Original gene
- Recombinant gene
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!
Use Bayesian optimization!