Still optimizing in the dark? Bayesian optimization for model configuration and experimental design

Javier González

University of Sheffield, Sheffield, UK

February 18, 2016. Groningen, The Netherlands.

The University Of Sheffield.

Sheffield Institute for Translational Neuroscience

Goal of the talk

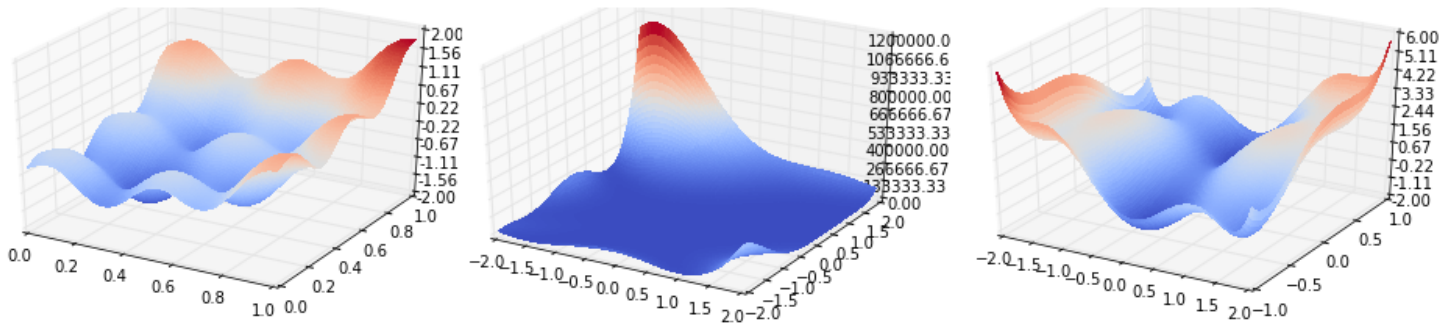
"Civilization advances by extending the number of important operations which we can perform without thinking of them."
(Alfred North Whitehead)

- To configure statistical/ML models automatically.
- To automatically design sequential experiments to optimize physical processes.

General framework: global optimization

Consider a *well behaved* function $f : \mathcal{X} \to \mathbb{R}$ where $\mathcal{X} \subseteq \mathbb{R}^D$ is (in principle) a bounded domain.

$$x_M = \arg\min_{x \in \mathcal{X}} f(x).$$

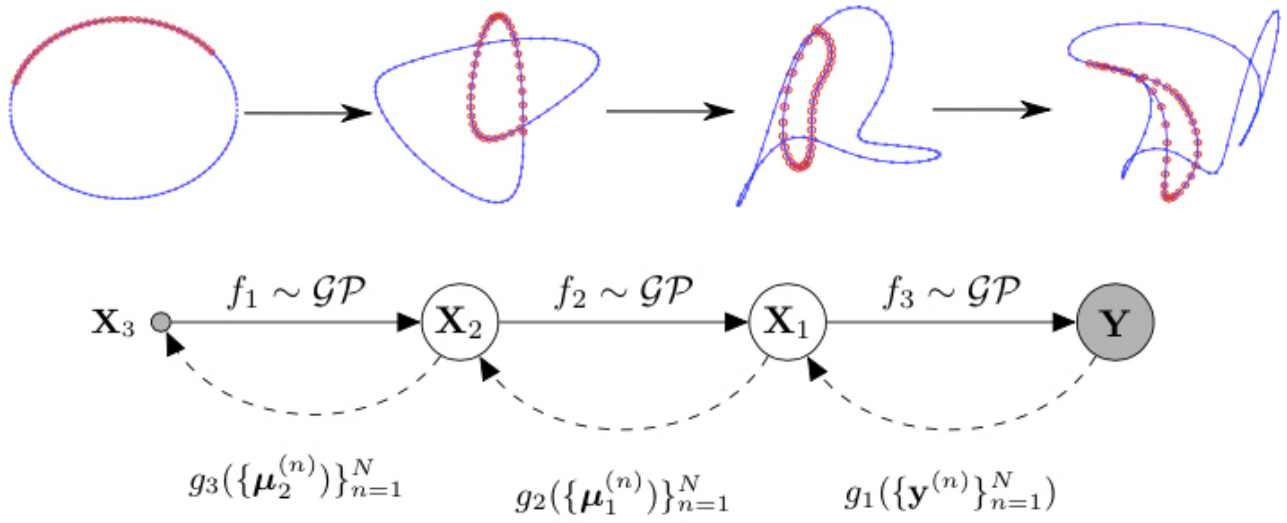


- f is explicitly unknown (computer model, process embodied in a physical process) and multimodal.
- Evaluations of f may be perturbed.
- Evaluations of f are (very) expensive.

Expensive functions, who doesn't have one?

[Dai, Damianou, González, Lawrence, 2016]

Model configuration: deep models parameter tuning

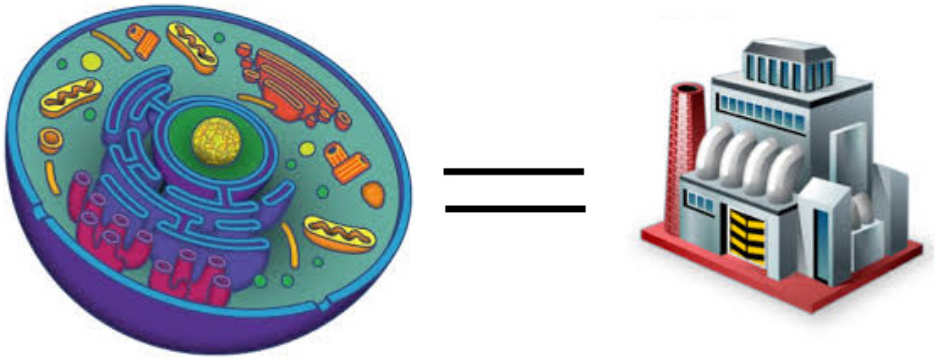


- Number of layers/units per layer.
- Weight penalties, variational parameters, learning rates, etc.

Expensive functions, who doesn't have one?

[González, Lonworth, James and Lawrence, 2014, 2015]

Design of experiments: gene optimization



- 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.

What to do?

If f is L-Lipschitz continuous and we are in a noise-free domain to guarantee that we propose some $\mathbf{x}_{M,n}$ such that

$$f(\mathbf{x}_M) - f(\mathbf{x}_{M,n}) \leq \epsilon$$

we need to evaluate f on a D-dimensional unit hypercube:

$$(L/\epsilon)^D evaluations!$$

Example: $(10/0.01)^5 = 10e14...$
... but function evaluations are very expensive!

Alternatives: random search, genetic algorithms, CMA-ES still require many function evaluations.

Regret minimization

The goal is to make a series of $x_1, \ldots, x_N$ evaluations of f such that the *cumulative regret*

$$r_N = \sum_{n=1}^{N} f(x_n) - N f(x_M)$$

is minimized.

r_N is minimized if we start evaluating f at x_M as soon as possible.

Probabilistic numerics approach

http://www.probabilistic-numerics.org/

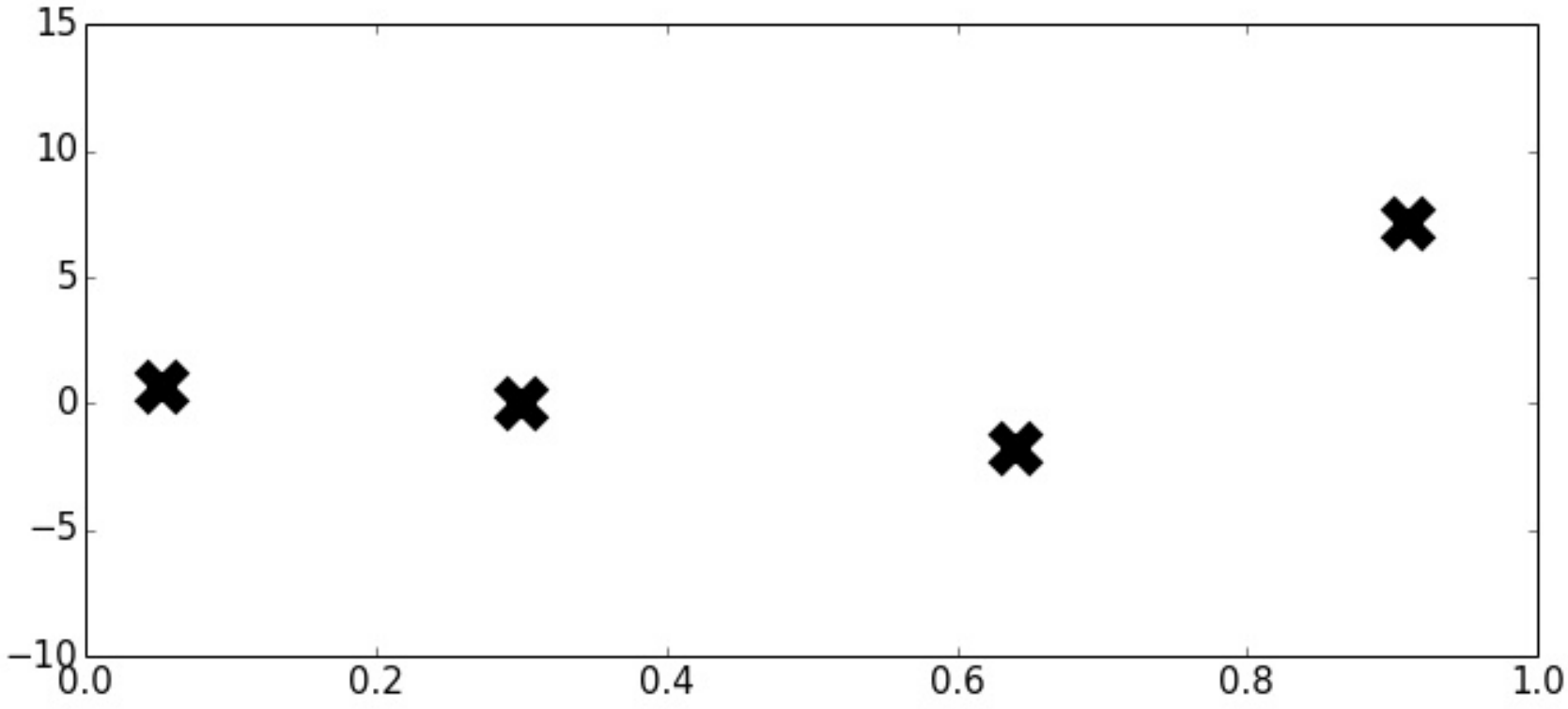
1. Minimizing the regret implies to see an *optimization* problem as a *decision* problem.

2. *Decision* problems can be seen as *inference* if we take into account the *epistemic* uncertainty we have about the system we are studying.

Probability theory is the right way to model uncertainty.

Typical situation

We have a few function evaluations

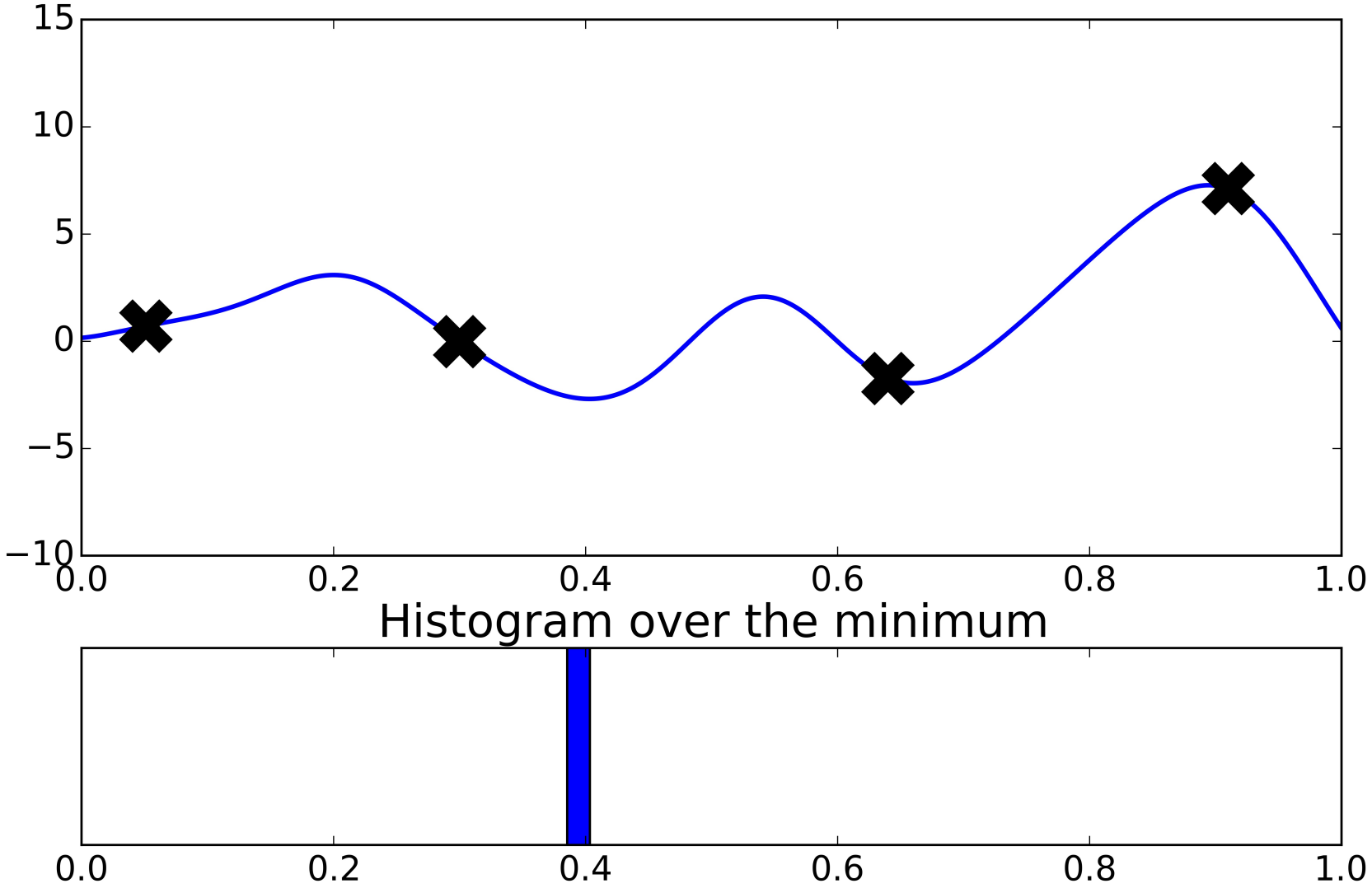


Where is the minimum of f?

Where should we take the next evaluation?

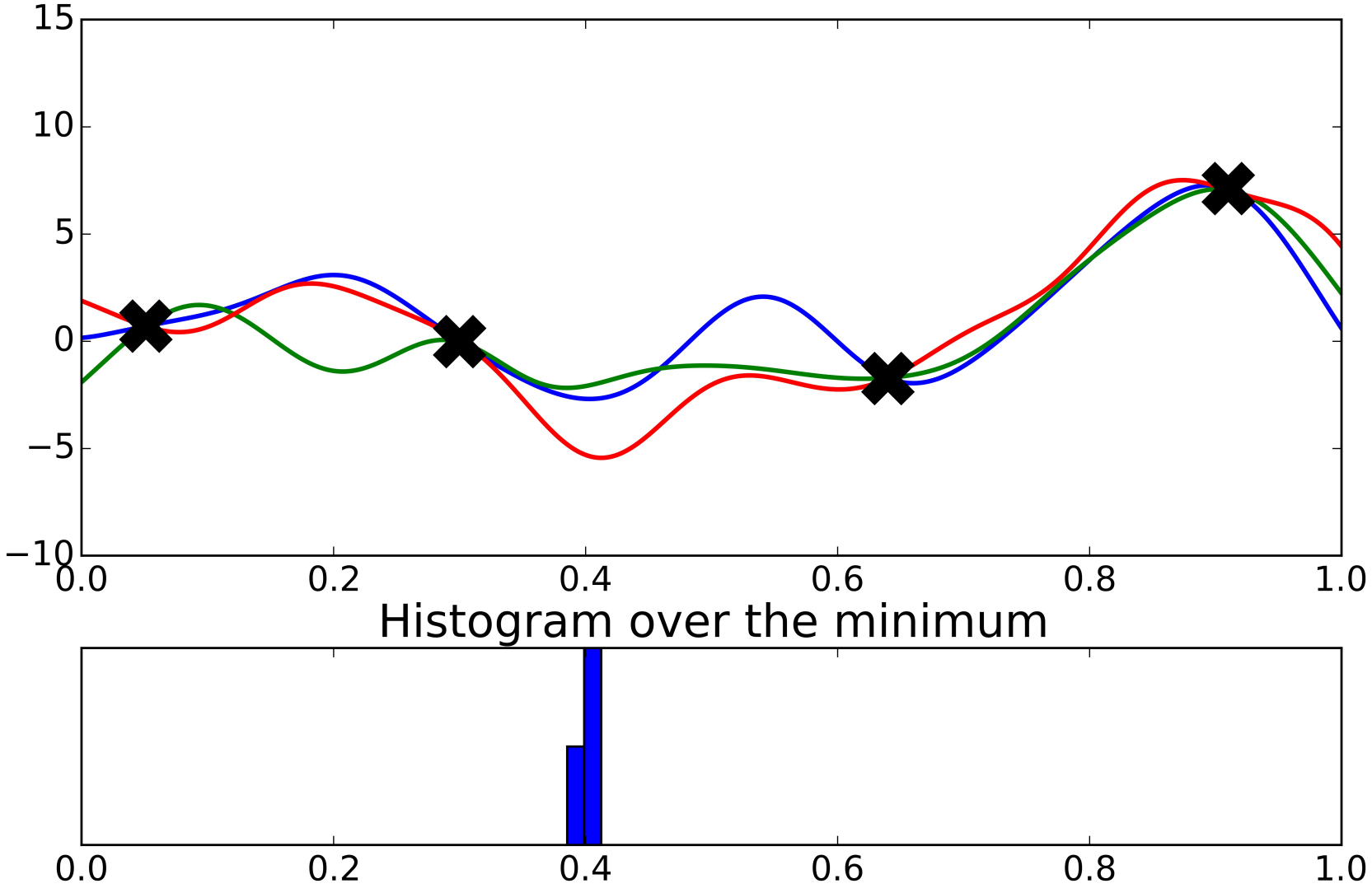
Intuitive solution

One curve



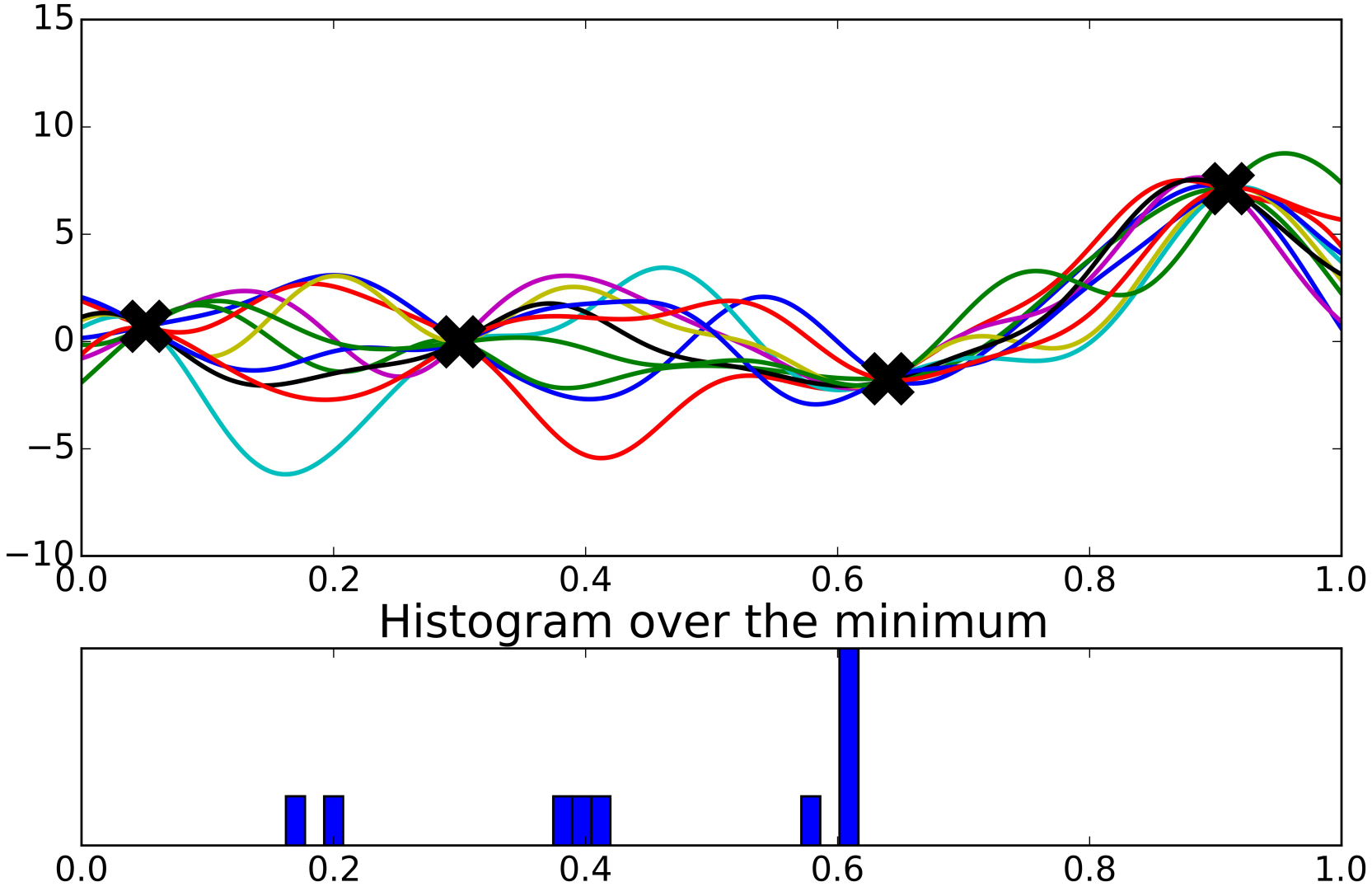
Intuitive solution

Three curves



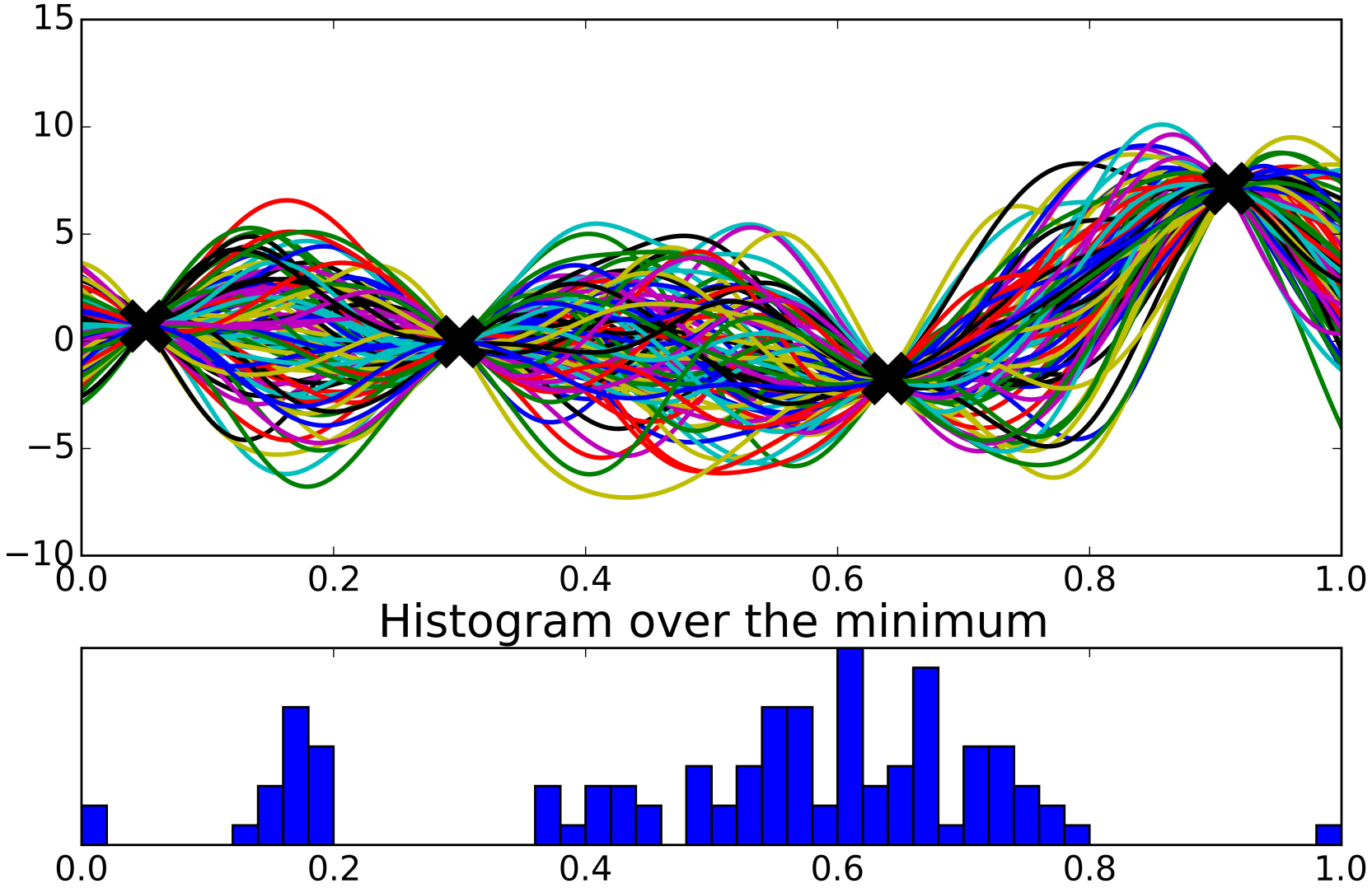
Intuitive solution

Ten curves



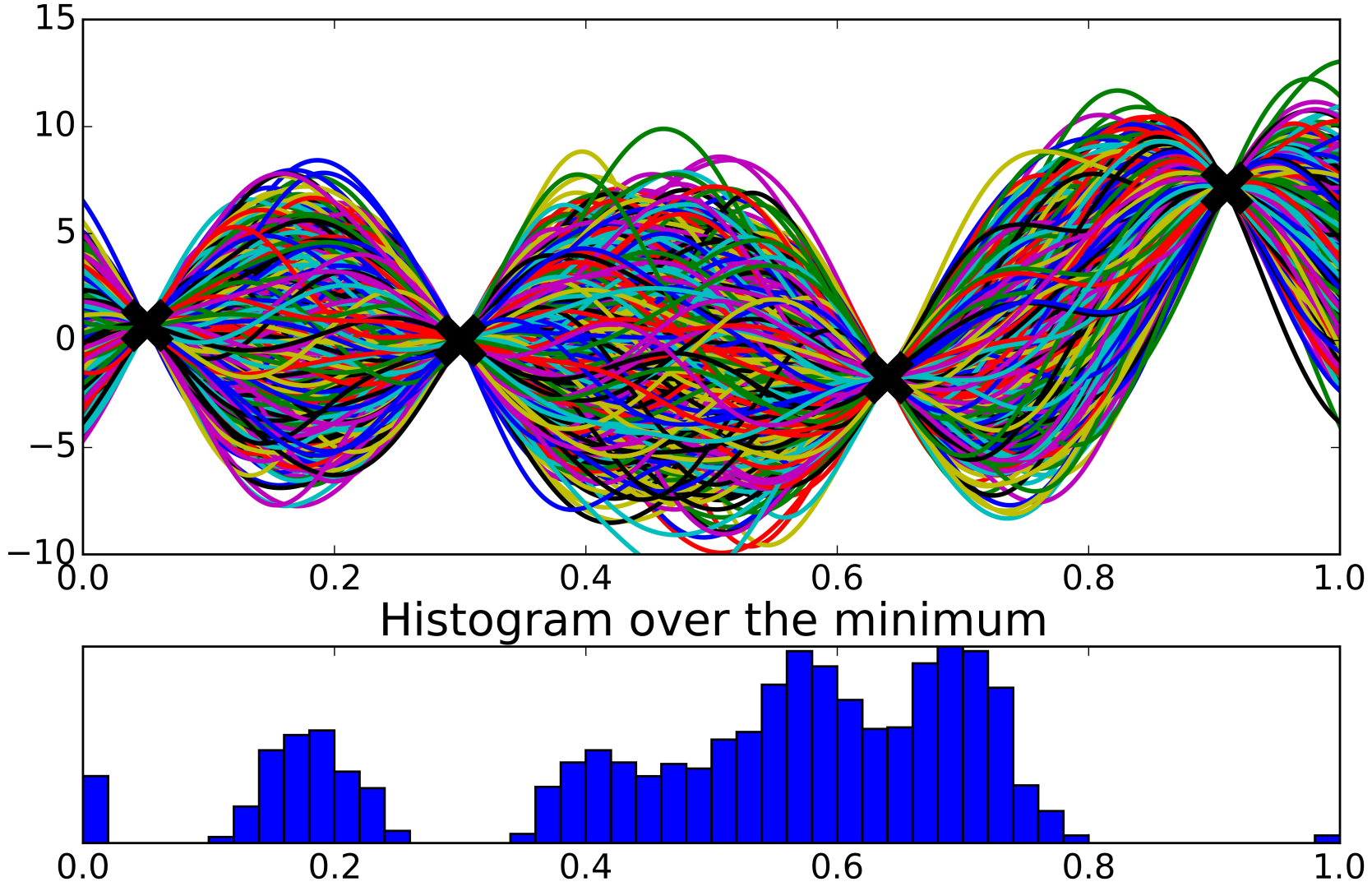
Intuitive solution

Hundred curves



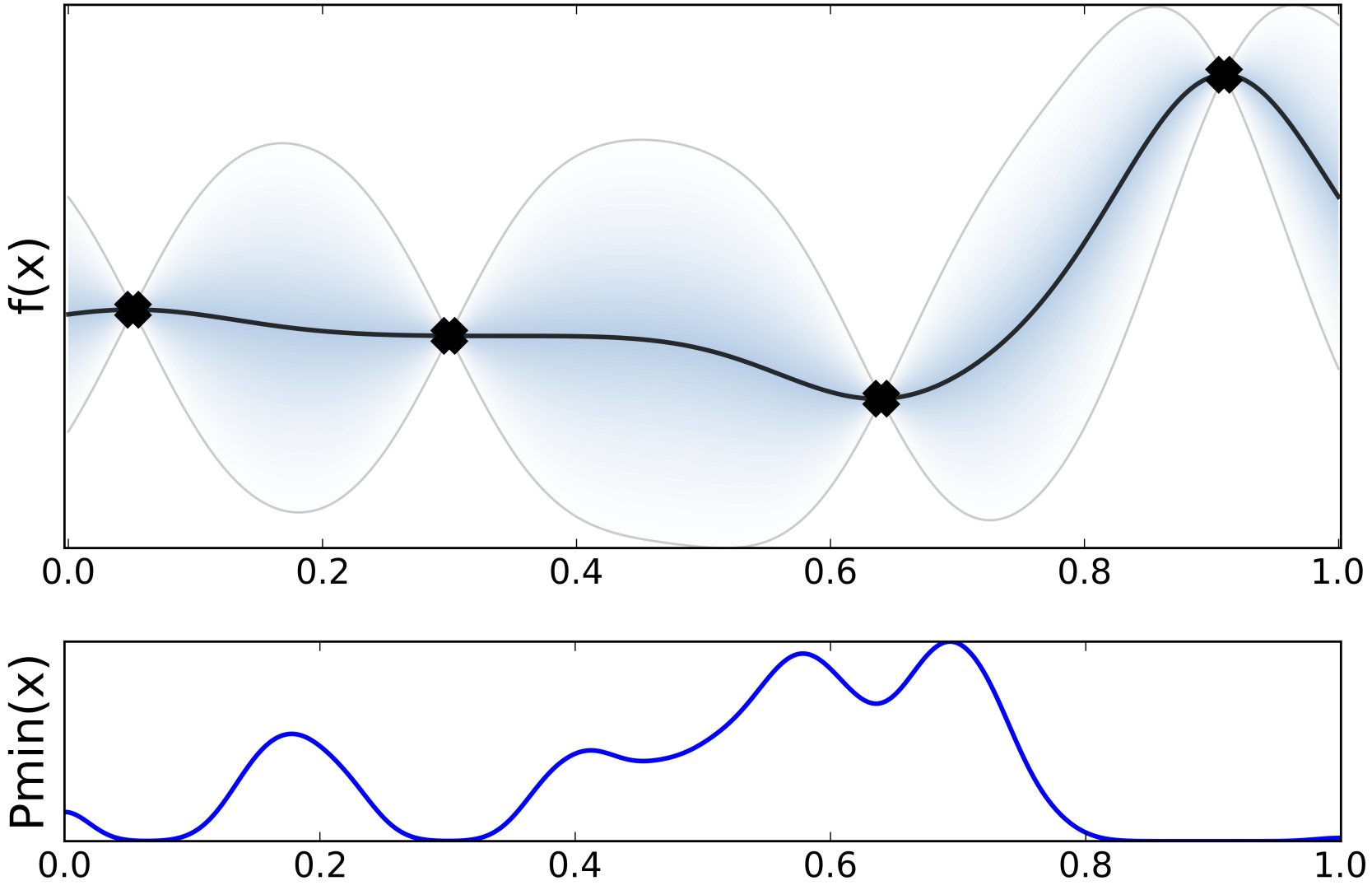
Intuitive solution

Many curves



Intuitive solution

Infinite curves



What just happened?

- We made some *prior assumptions* about f.
- Information about the minimum is now encoded in a new function: *the probability distribution* $p_{\min}$.
- We can use $p_{\min}$ (or a functional of it) to *decide where to sample* next.
- Other functions to encode relevant information about the minimum are possible, e. g. the expected loss at each location.

Bayesian Optimization

[Mockus, 1978]

Methodology to perform global optimization of multimodal black-box functions.

1. Choose some *prior measure* over the space of possible objectives f.

2. Combine prior and the likelihood to get a *posterior* over the objective given some observations.

3. Use the posterior to decide where to take the next evaluation according to some *acquisition/loss function*.

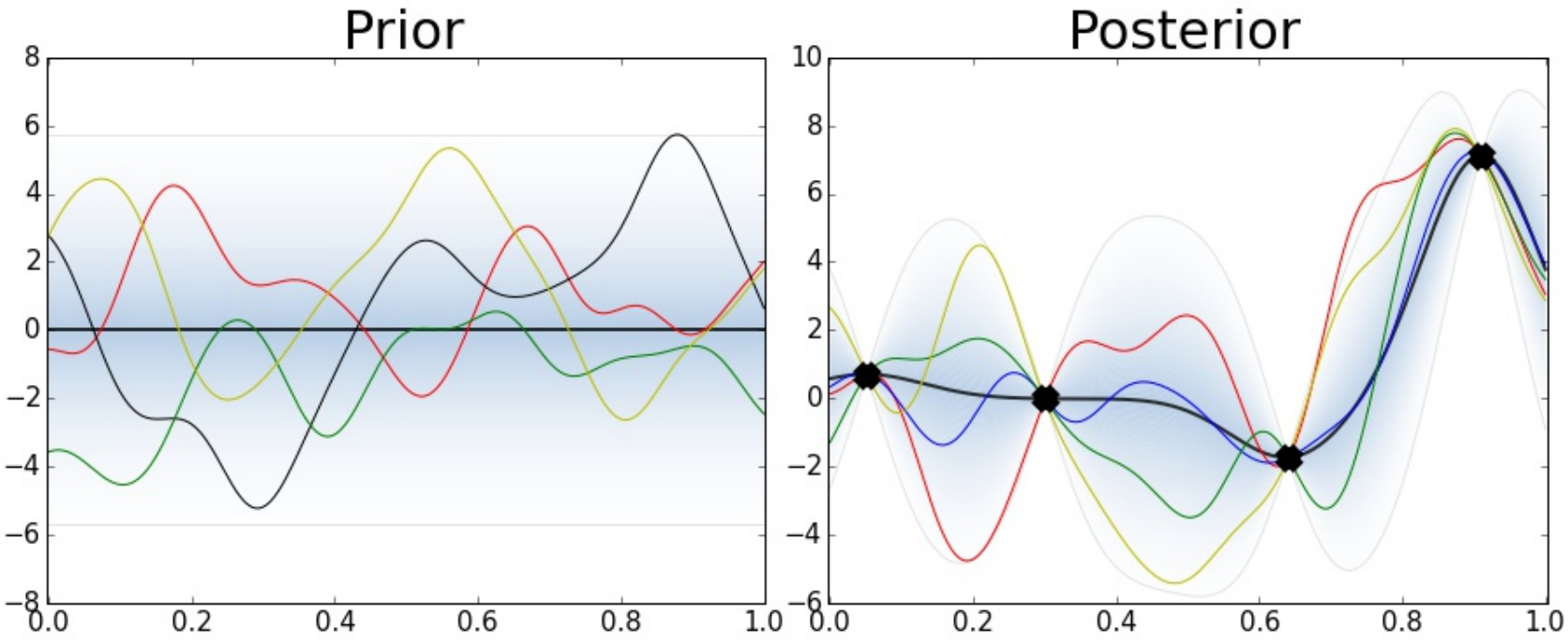
4. Augment the data.

Iterate between 2 and 4 until the evaluation budget is over.

Probability measure over functions

Gaussian processes [Rasmunsen and Williams, 2006]

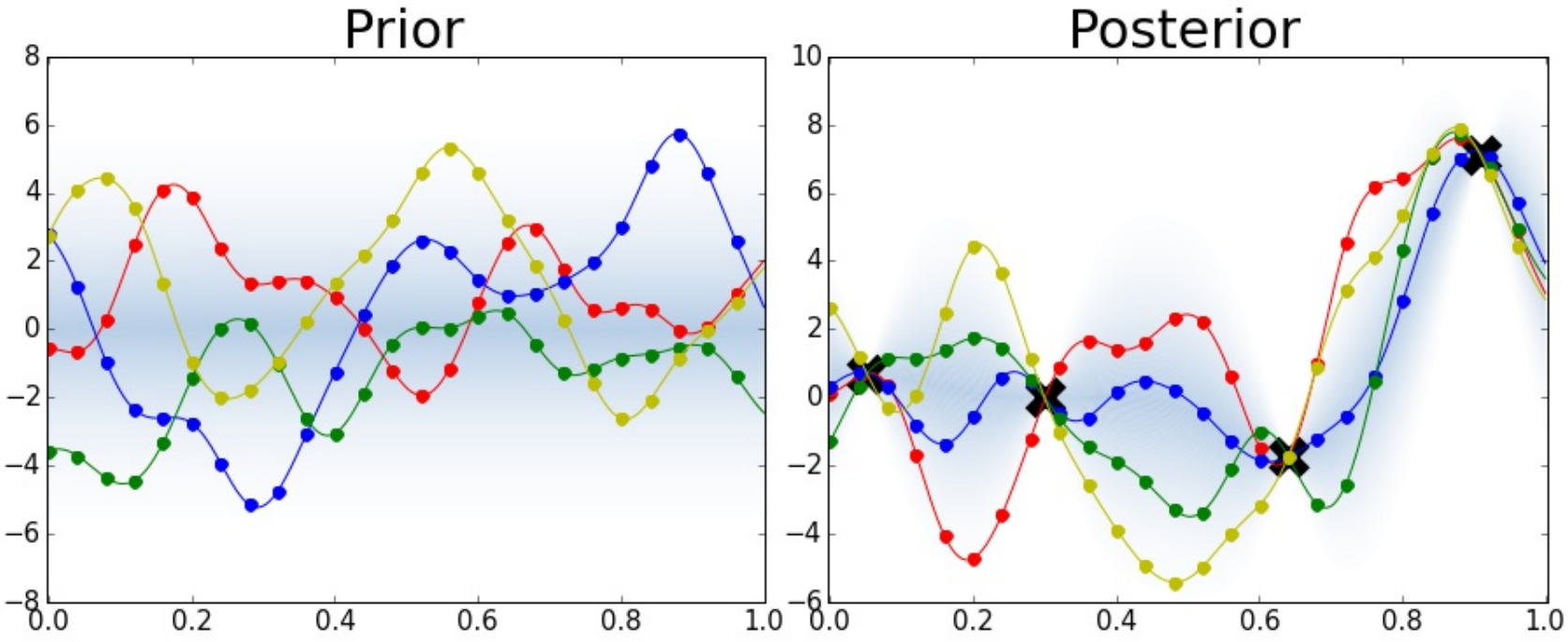
Infinite-dimensional probability density, such that each linear finite-dimensional restriction is multivariate Gaussian.



Probability measure over functions

Gaussian processes [Rasmunsen and Williams, 2006]

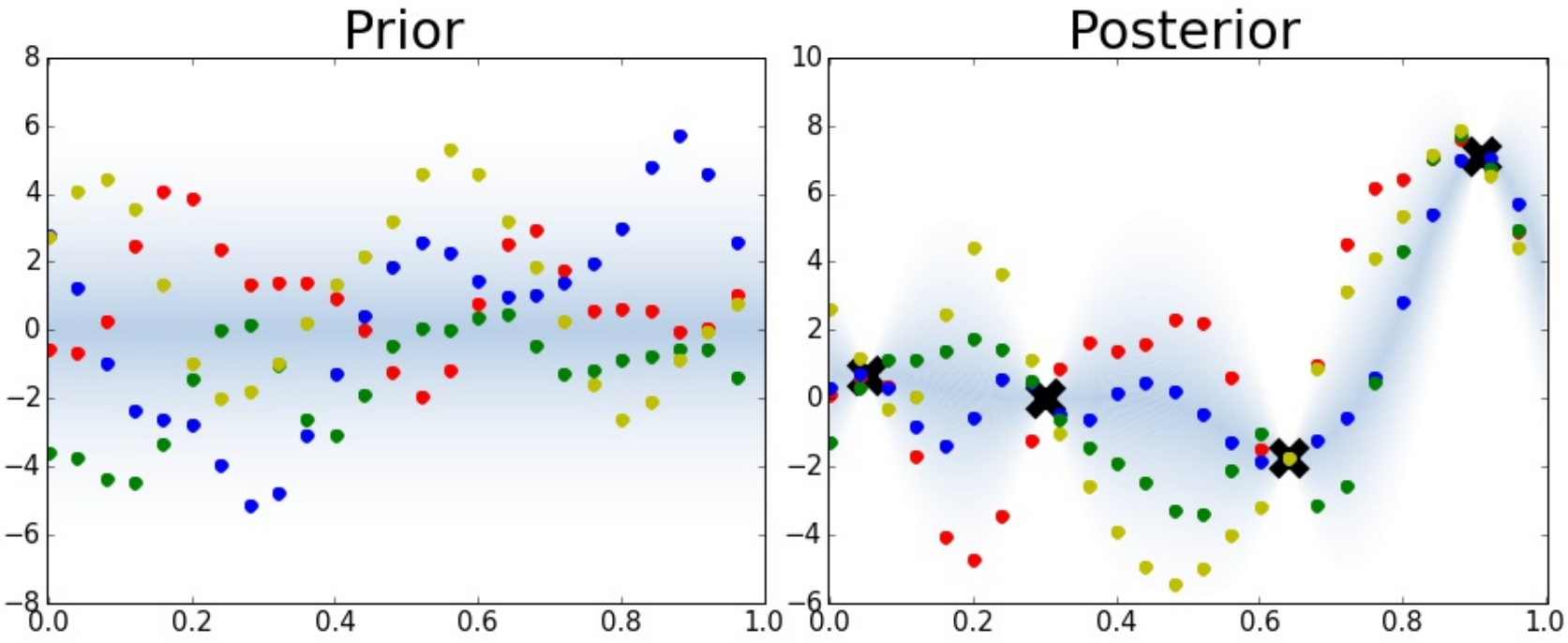
Infinite-dimensional probability density, such that each linear finite-dimensional restriction is multivariate Gaussian.



Probability measure over functions

Gaussian processes [Rasmunsen and Williams, 2006]

Infinite-dimensional probability density, such that each linear finite-dimensional restriction is multivariate Gaussian.



Probability measure over functions

Gaussian processes [Rasmunsen and Williams, 2006]

- GP is fully determined by a covariance function $k(\mathbf{x}, \mathbf{x}'; \theta)$ operator.
- Regression problems: $y_i = f(\mathbf{x}_i) + \epsilon_i$.
- Marginals at any $\mathbf{x}_*$ are Gaussians with mean and variance

$$\mu(\mathbf{x}_*|\theta, \mathcal{D}) = \mathbf{k}_{\boldsymbol{\theta}}(\mathbf{X}_*)^\top [\mathbf{k}_{\boldsymbol{\theta}} + \sigma^2 \mathbf{I}]^{-1} \mathbf{y}$$

$$\sigma^2(\mathbf{x}_*|\theta, \mathcal{D}) = k_{\boldsymbol{\theta}}(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_{\boldsymbol{\theta}}(\mathbf{x}_*)^\top [\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2 \mathbf{I}]^{-1} \mathbf{k}_{\boldsymbol{\theta}}(\mathbf{x}_*)$$

where $\mathcal{D}$ is available dataset.

Acquisition functions

Making use of the model uncertainty

GPs has marginal closed-form for the posterior mean $\mu(\mathbf{x}_*)$ and variance $\sigma^2(\mathbf{x}_*)$.

- **Exploration**: Evaluate in places where the variance is large.
- **Exploitation**: Evaluate in places where the mean is low.

Acquisition functions balance these two factors to determine where to evaluate next.

Exploration vs. exploitation

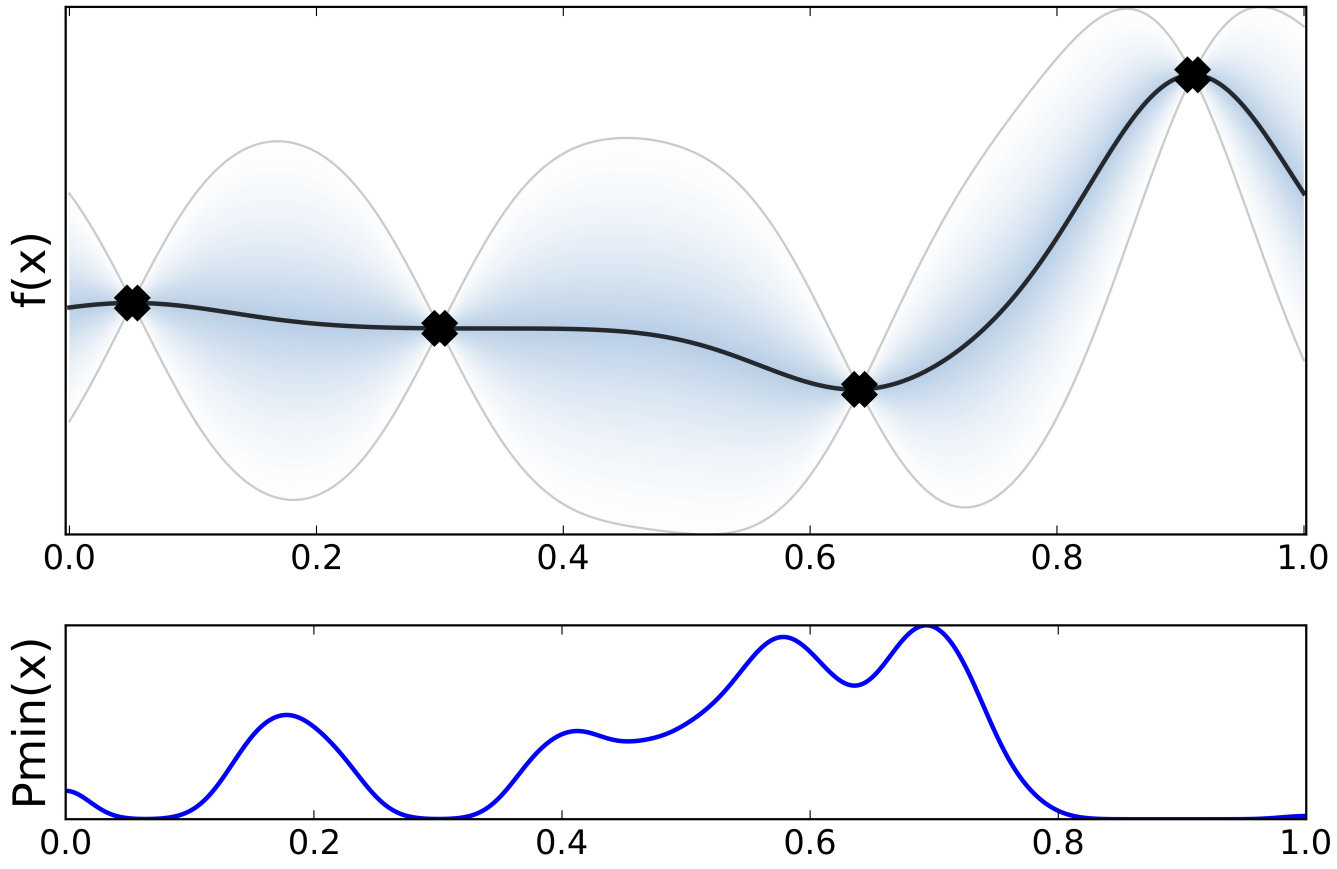
[Borji and Itti, 2013]

Bayesian optimization explains human active search.

Information-theoretic approaches

[Hennig and Schuler, 2013; Hernández-Lobato et al., 2014]

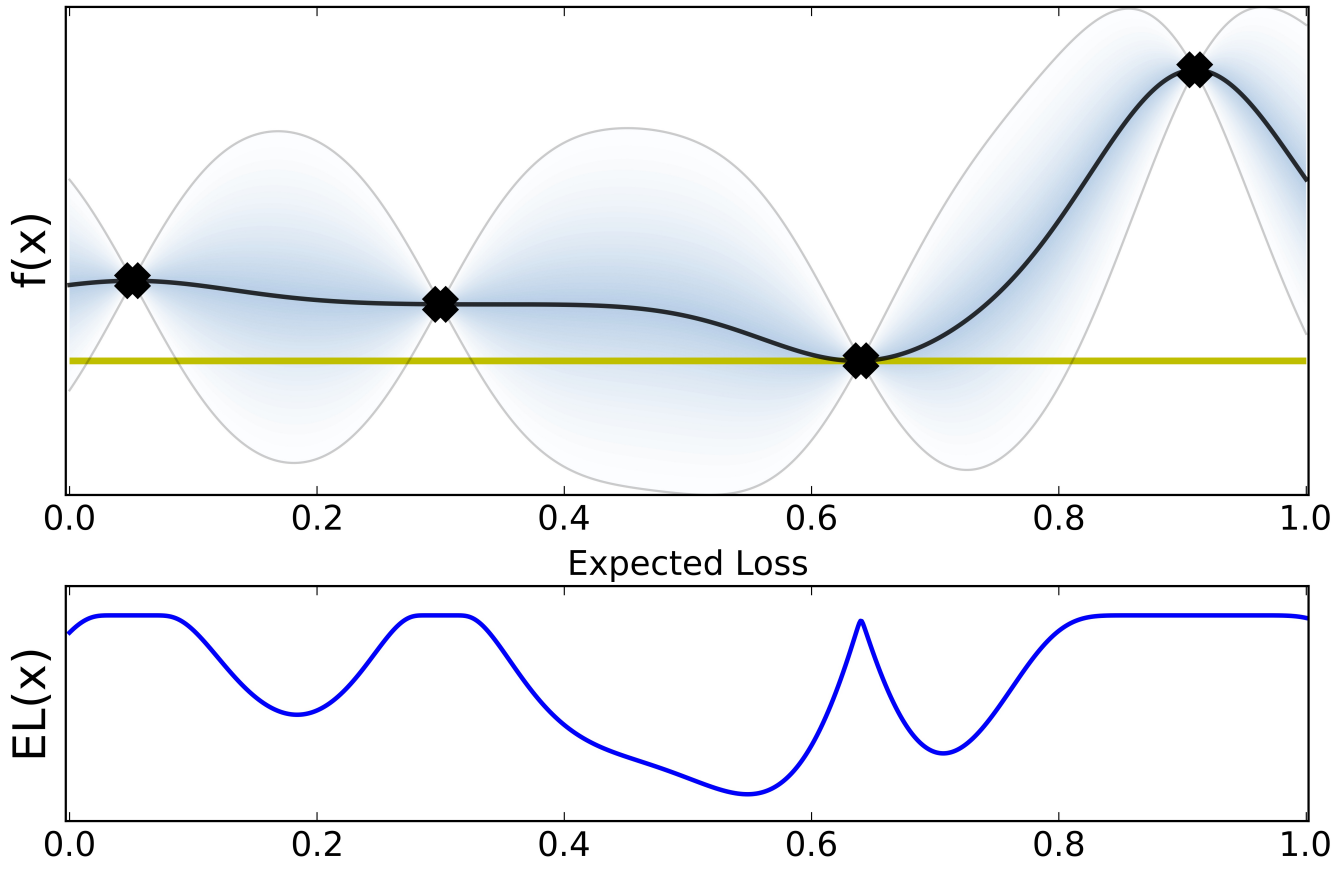
$$\alpha_{ES}(\mathbf{x}; \theta, \mathcal{D}) \triangleq H[p(x_{min}|\mathcal{D})] - \mathbb{E}_{p(y|\mathcal{D},\mathbf{x})}[H[p(x_{min}|\mathcal{D} \cup \{\mathbf{x}, y\})]]$$



Expected Loss

[Osborne, 2010]

$$\alpha_{EL}(\mathbf{x};\theta,\mathcal{D}) \triangleq \mathbb{E}[\min(y_*, y_{min})]$$



Bayesian Optimization

As a 'mapping' between two problems

BO is an strategy to transform the problem

$$x_M = \arg\min_{x \in \mathcal{X}} f(x)$$

unsolvable!

into a series of problems:

$$x_{n+1} = \arg\max_{x \in \mathcal{X}} \alpha(x; \mathcal{D}_n, \theta_n)$$

solvable!

where now:

- $\alpha(x)$ is inexpensive to evaluate.
- The gradients of $\alpha(x)$ are typically available.
- Still need to find x_{n+1}: gradient descent, DIRECT or other heuristics.

Illustration of BO

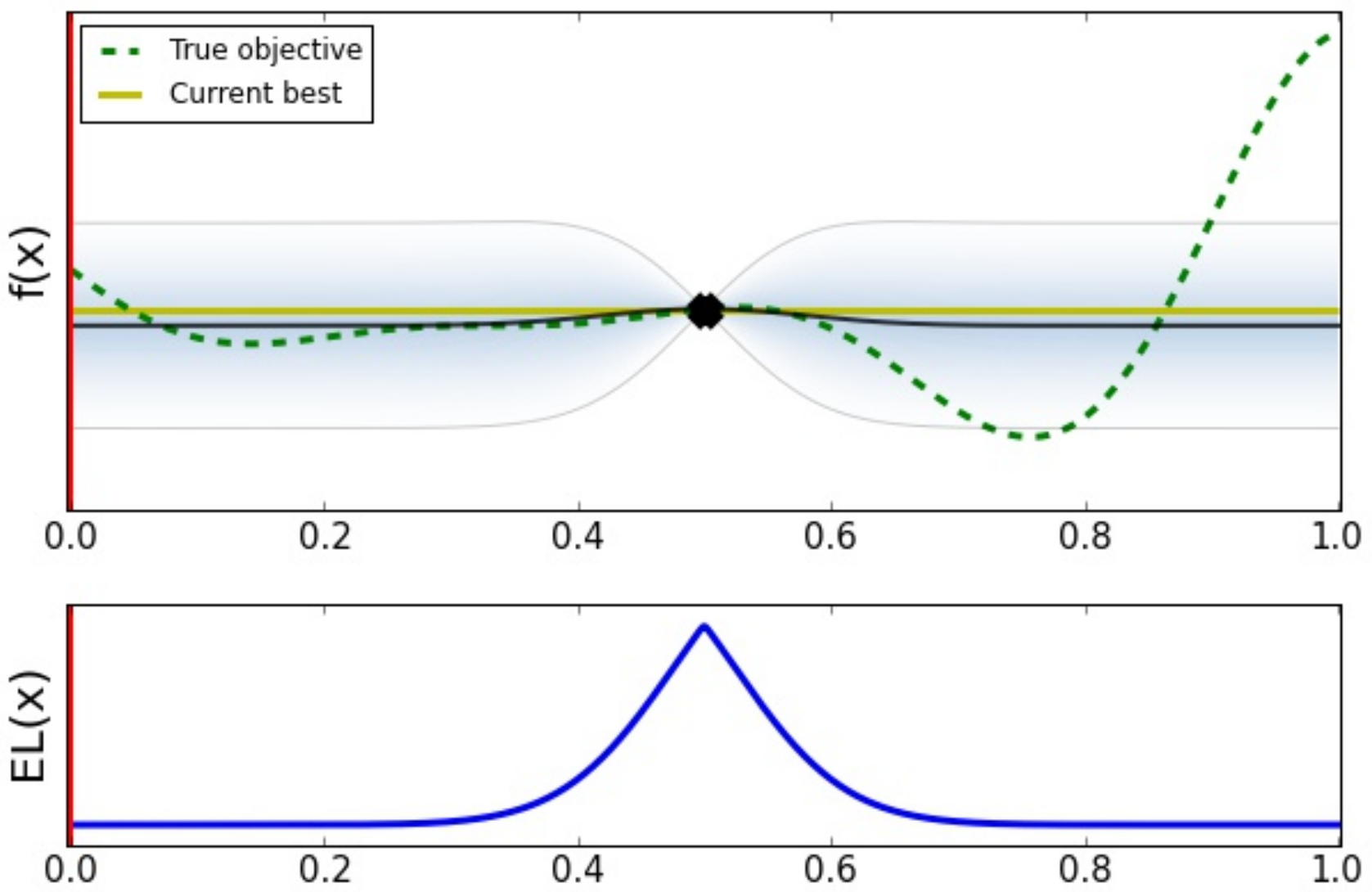


Illustration of BO

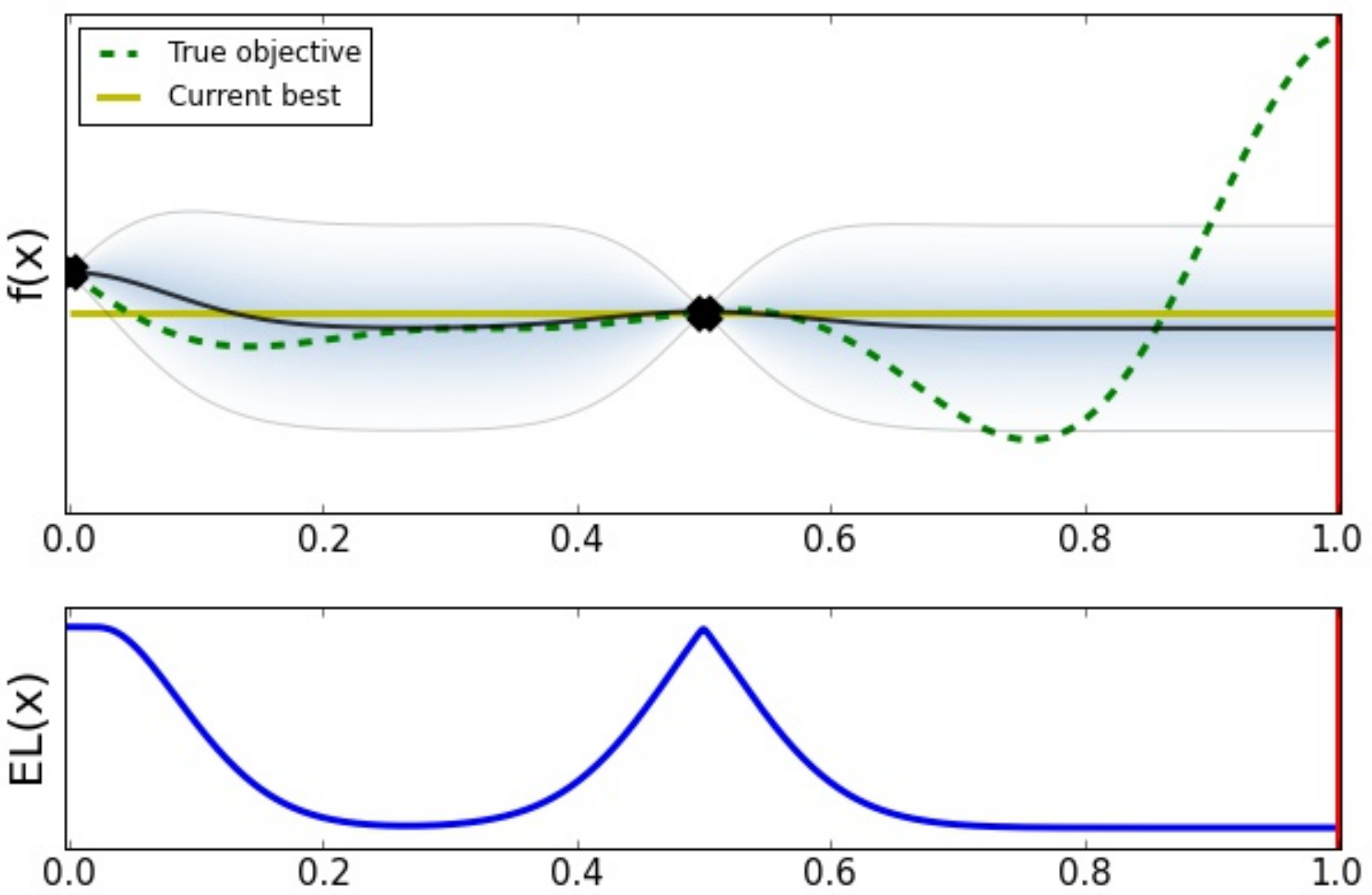


Illustration of BO

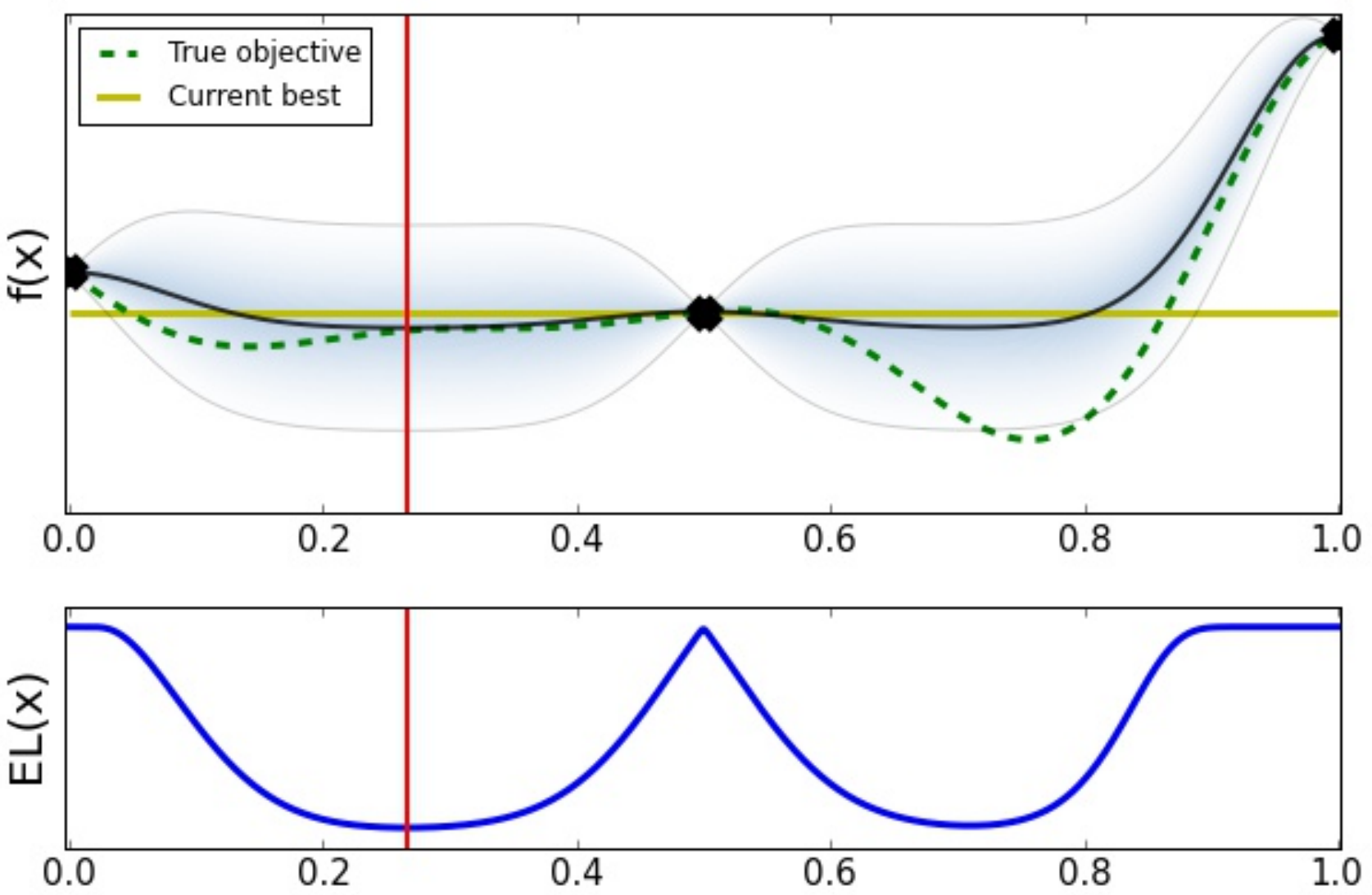


Illustration of BO

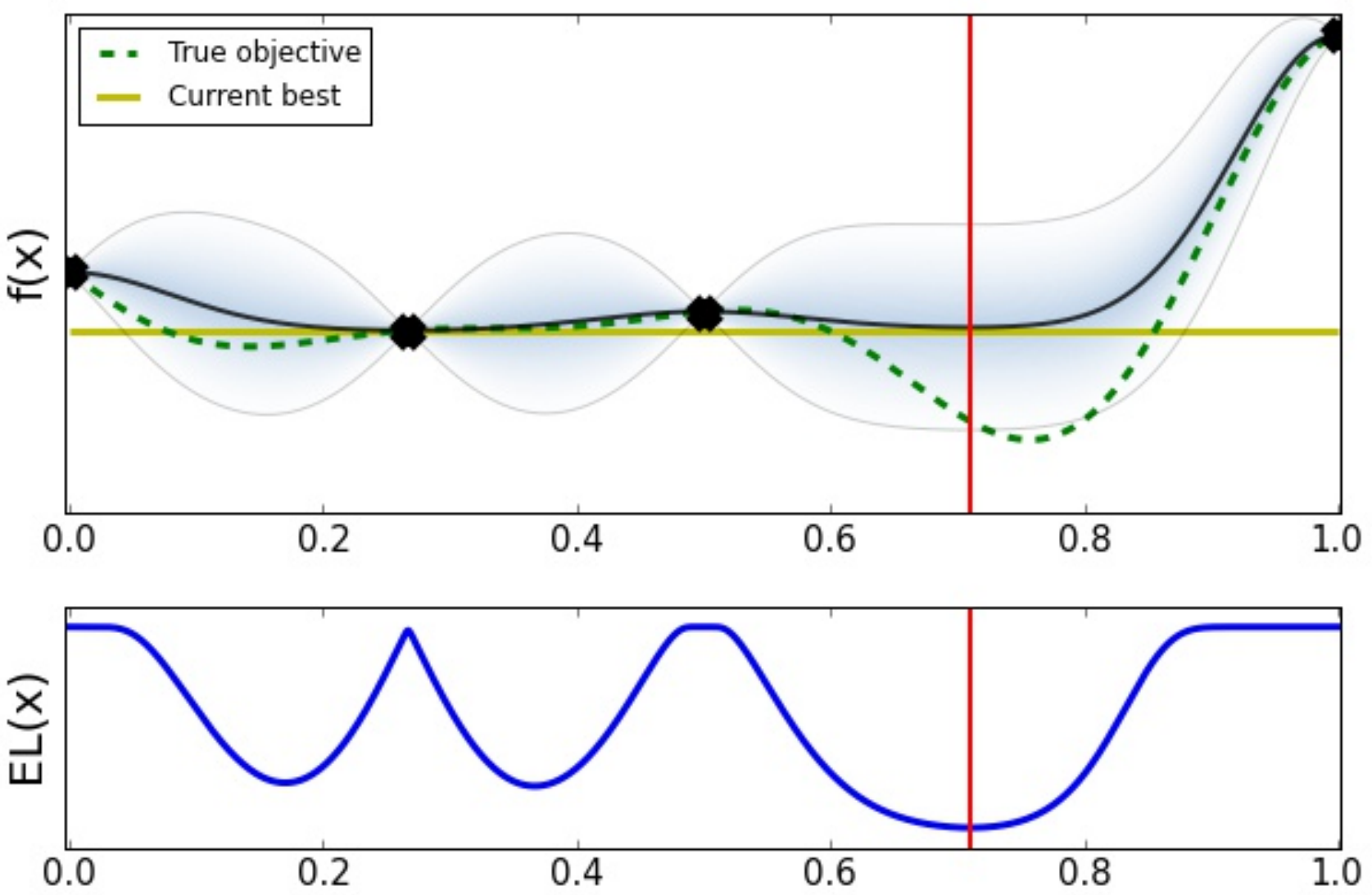


Illustration of BO

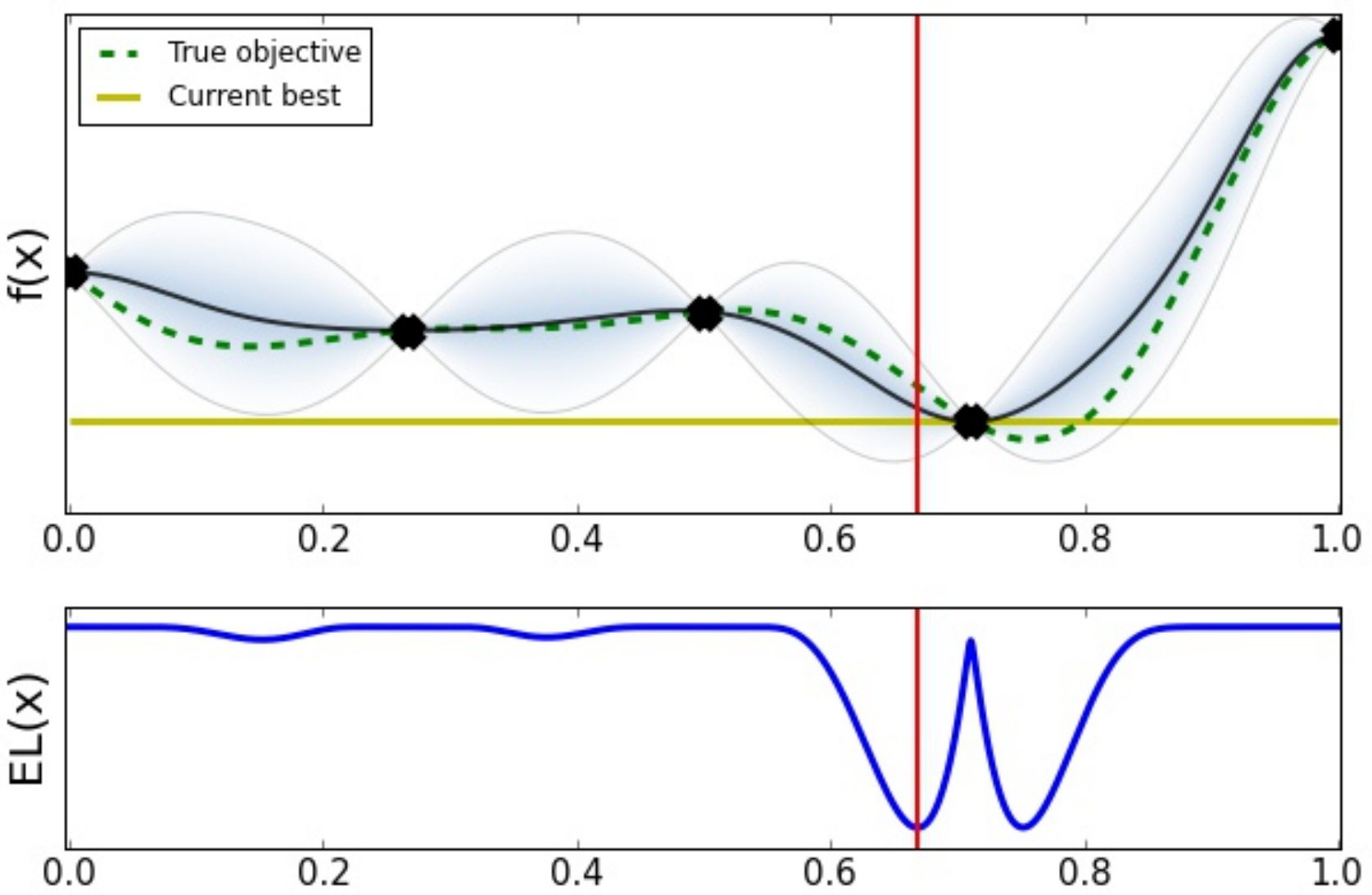


Illustration of BO

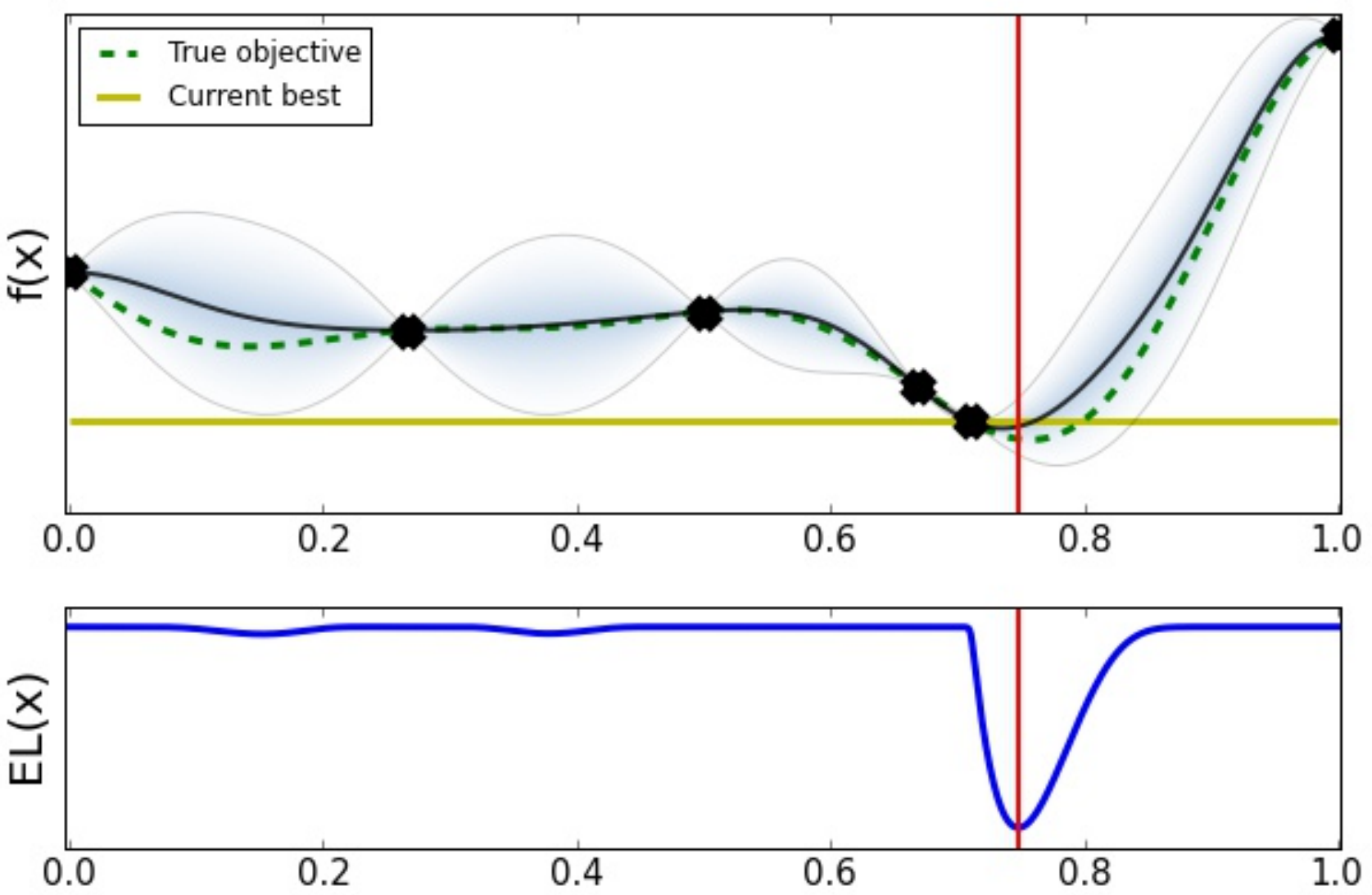


Illustration of BO

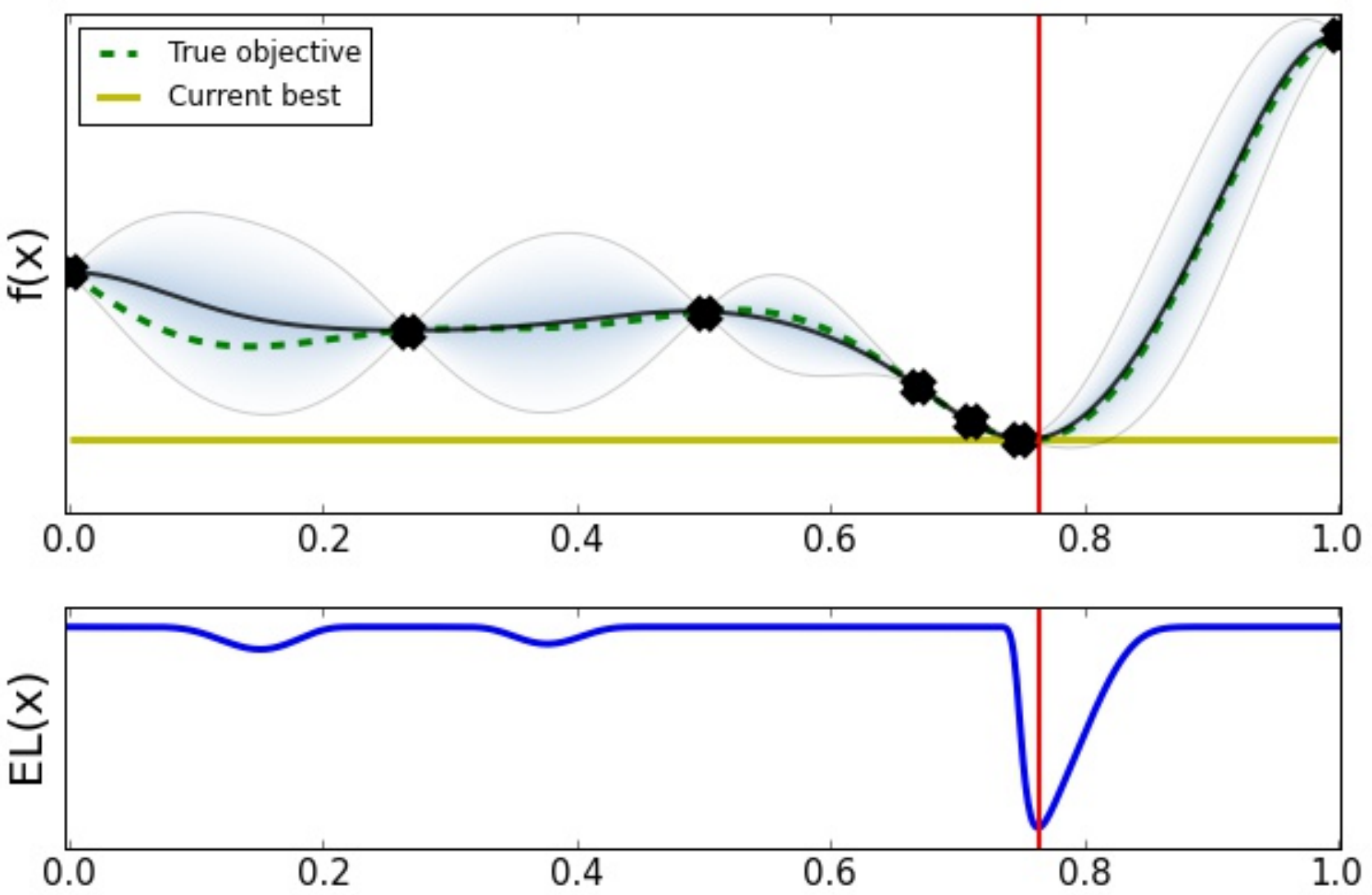
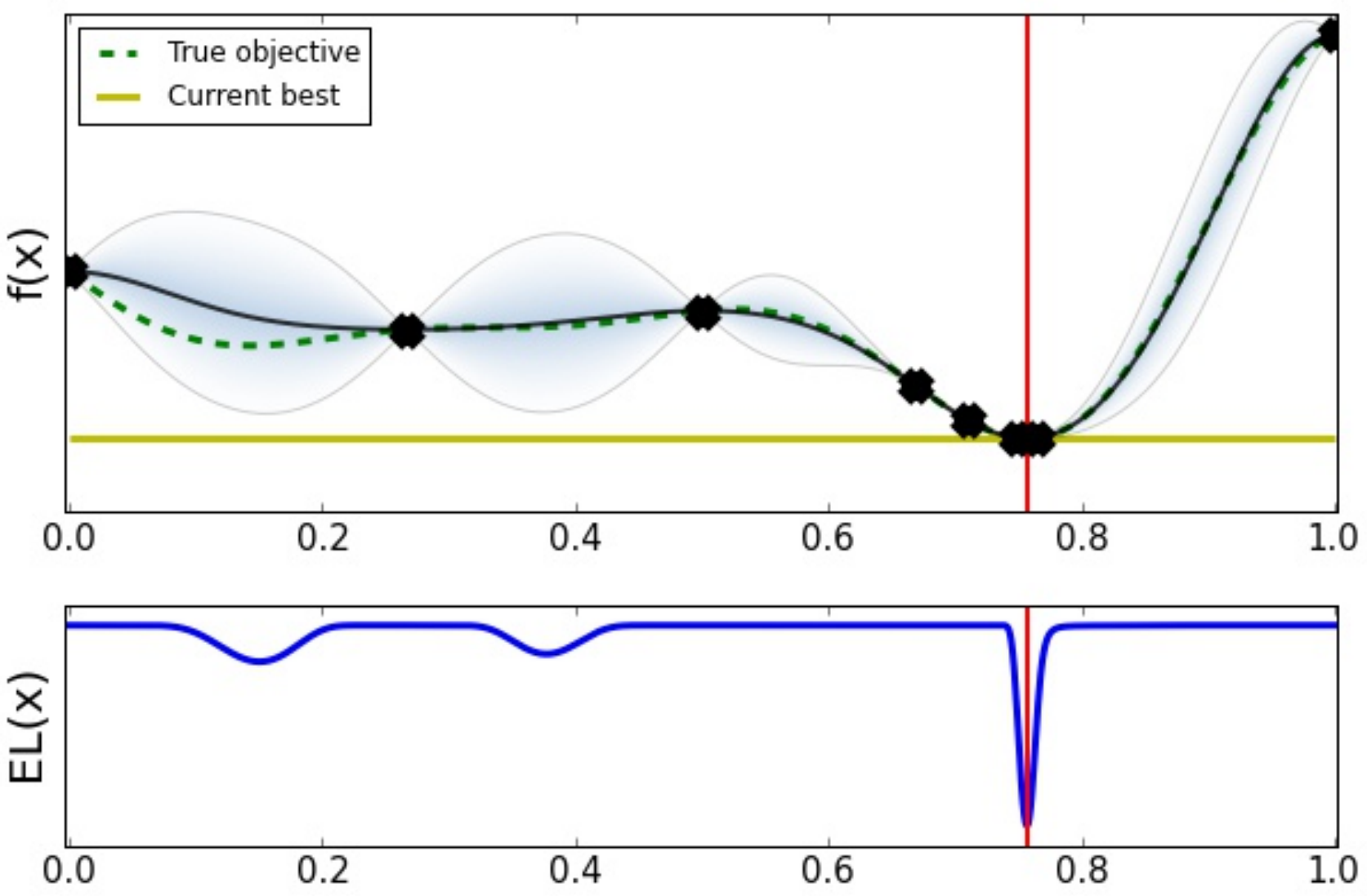


Illustration of BO



Why these ideas have been ignored for years?

- BO depends on its own (model) parameters.
- Lack of software to apply these methods as a black optimization boxes.
- Reduced scalability in dimensions and number of evaluations.

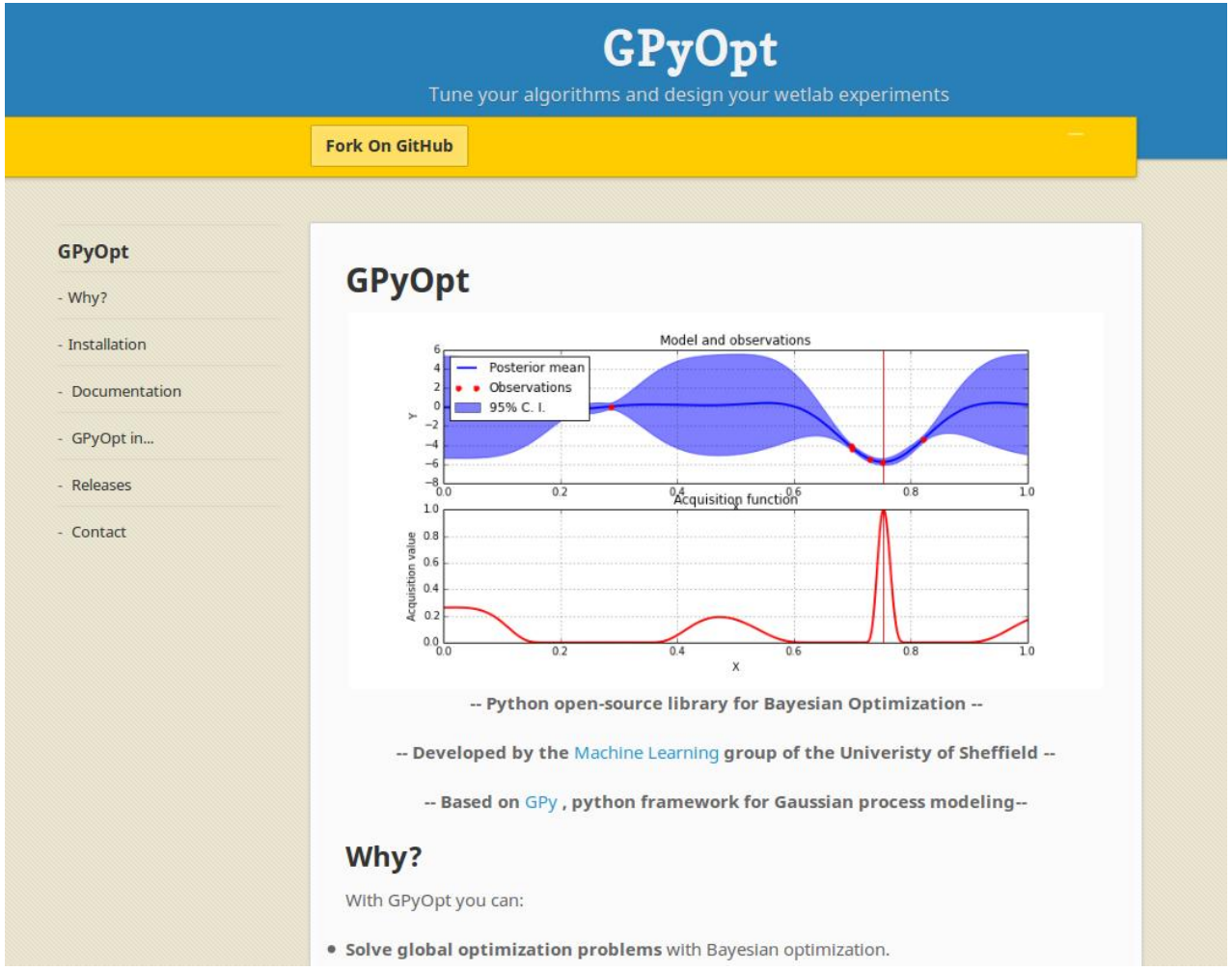
Practical Bayesian Optimisation of Machine Learning Algorithms.
Snoek, Larochelle and Adams. NIPS 2012

+

Other works of M. Osborne, P. Hennig, N. de Freitas, etc.

Open Software: GPyOpt

http://sheffieldml.github.io/GPyOpt/



Open Software: GPyOpt

http://sheffieldml.github.io/GPyOpt/

- Easy python interface.
- Surrogate models available: GPs, sparse GPs, deep GPs, etc.
- MCMC integration of the acquisition functions.
- Parallel (synchronous batch) optimization.
- Constrained optimization.
- Handles continous and discrete inputs.
- More to come!

Open source code. You can contribute!

Non myopic Bayesian optimization

- 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.

Myopic loss (one step ahead)

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 :

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

($\mathcal{I}_0$: current information $\mathcal{D}$, θ and likelihood type).

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 n steps ahead

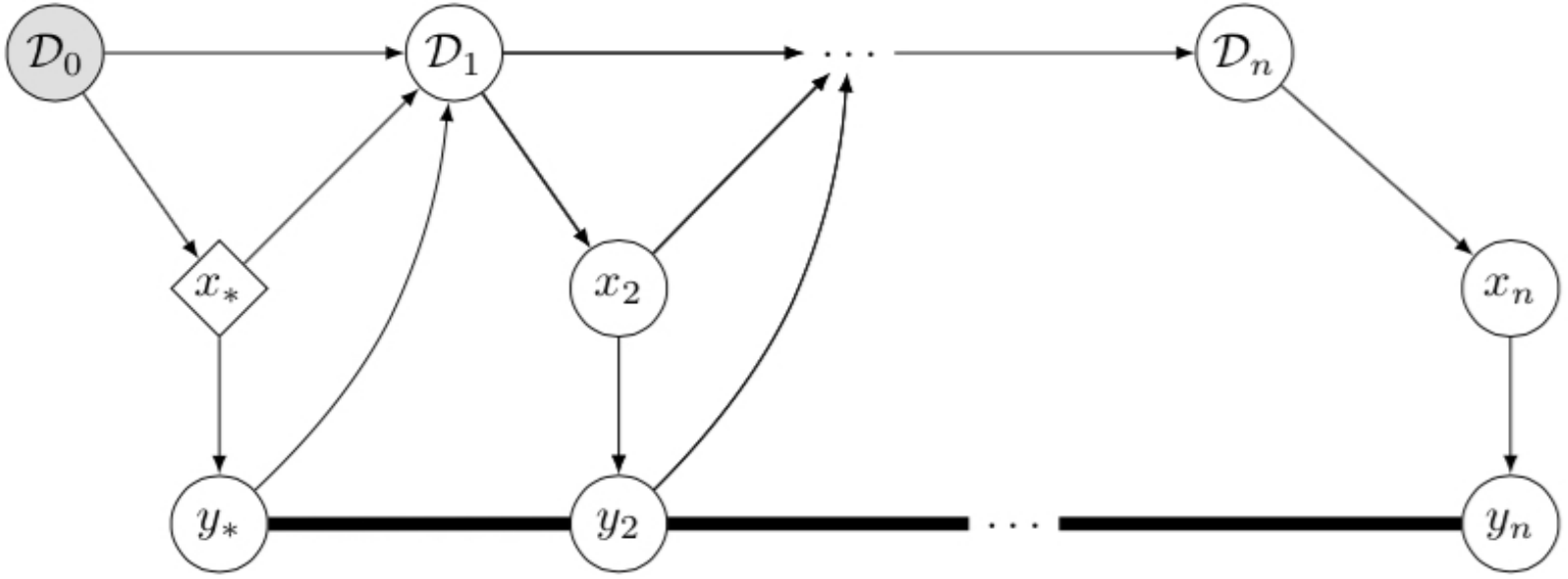
$$\Lambda_n(\mathbf{x}_*|\mathcal{I}_0) = \int \lambda(y_n) \prod_{j=1}^{n} p(y_j|\mathbf{x}_j, \mathcal{I}_{j-1}) p(\mathbf{x}_j|\mathcal{I}_{j-1}) dy_* \dots dy_n \mathrm{d}\mathbf{x}_2 \dots \mathrm{d}\mathbf{x}_n$$

- $p(y_j|\mathbf{x}_j, \mathcal{I}_{j-1}) = \mathcal{N}\left(y_j; \mu(\mathbf{x}_j; \mathcal{I}_{j-1}), \sigma^2(\mathbf{x}_j|\mathcal{I}_{j-1})\right)$: predictive distribution of the GP at $\mathbf{x}_j$ and
- $p(\mathbf{x}_j|\mathcal{I}_{j-1}) = \delta\Big(\mathbf{x}_j - \arg\min_{\mathbf{x}_* \in \mathcal{X}} \Lambda_{n-j+1}(\mathbf{x}_*|\mathcal{I}_{j-1})\Big)$: optimization step.

Intractable even for a handful number of steps ahead!

Looking n steps ahead

Graphical model representing the decision process of a myopic loss.



Relieving the myopia of Bayesian optimization

We present...

Relieving the myopia of Bayesian optimization

We present... GLASSES!

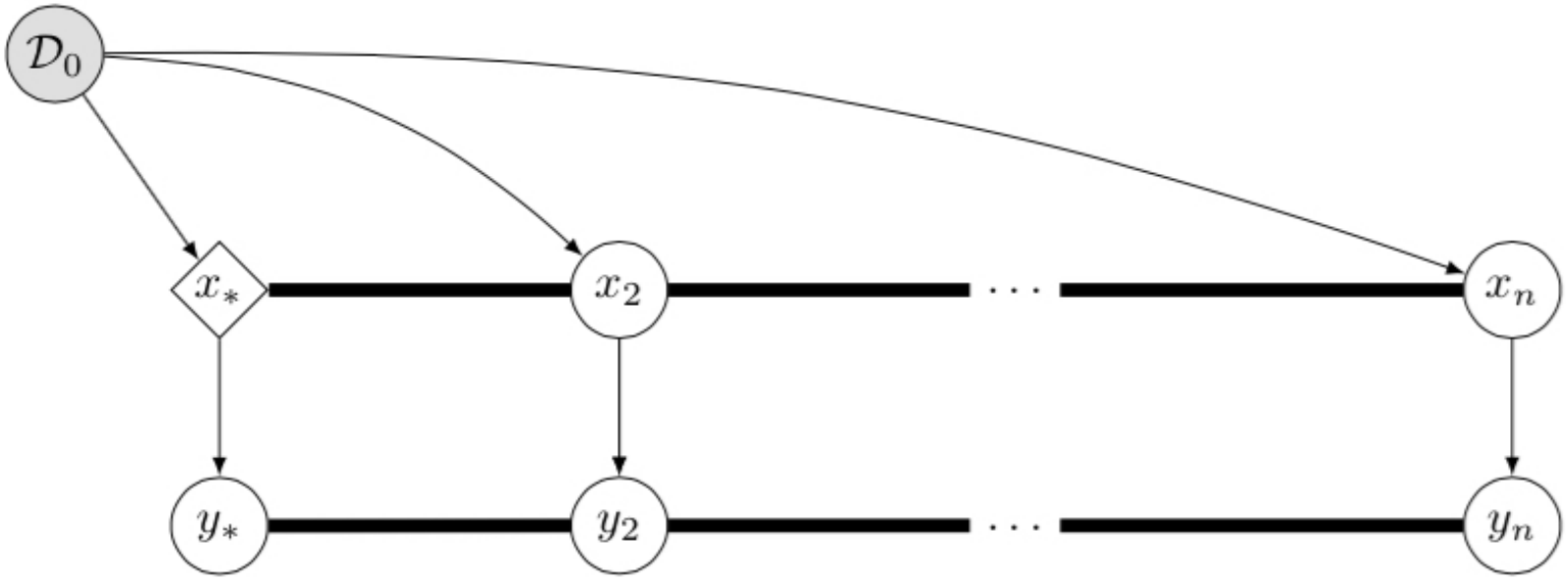
***G**lobal optimisation with **L**ook-**A**head through **S**tochastic **S**imulation and **E**xpected-loss **S**earch*

[González, Osborne, Lawrence, 2016]

GLASSES

Making the approximation sparse

Idea: jointly model the epistemic uncertainty about the steps ahead.



Render the approximation sparse by using some $p(\mathbf{x}_2, \ldots, \mathbf{x}_n | \mathcal{I}_0, \mathbf{x}_*)$.

GLASSES

Making the approximation sparse

Replace:

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

GLASSES

Making the approximation sparse

With

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

GLASSES

Making the approximation sparse

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

- $\mathbf{y} = \{y_*, \ldots, \ldots, y_n\}$ future evaluations of f.
- $\mathbf{X}$ the $(n-1) \times q$ locations of the future evaluations $\mathbf{x}_2, \ldots, \mathbf{x}_n$.
- $p(\mathbf{y}|\mathbf{X}, \mathcal{I}_0, \mathbf{x}_*)$ is multivariate Gaussian.
- $p(\mathbf{X}|\mathcal{I}_0, \mathbf{x}_*)$: distribution over the steps ahead.

GLASSES

Not easy to integrate the uncertainty in future values

Poblem: Select a good $p(\mathbf{X}|\mathcal{I}_0, \mathbf{x}_*)$ is complicated (could use a determinantal point process but still expensive).

- Instead of integrate $p(\mathbf{X}|\mathcal{I}_0, \mathbf{x}_*)$ replace it by an oracle $\mathcal{F}_n(\mathbf{x}_*)$ able to predict n future locations.
- $\mathbf{y} = (y_*, \ldots, y_n)^T$: (Gaussian) outputs of f at $\mathcal{F}_n(\mathbf{x}_*)$.
- Now:

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

Computing the value of the expected loss

Use Expectation Propagation for Gaussians densities [Cunningham and Hennig, 2011]

Use that:

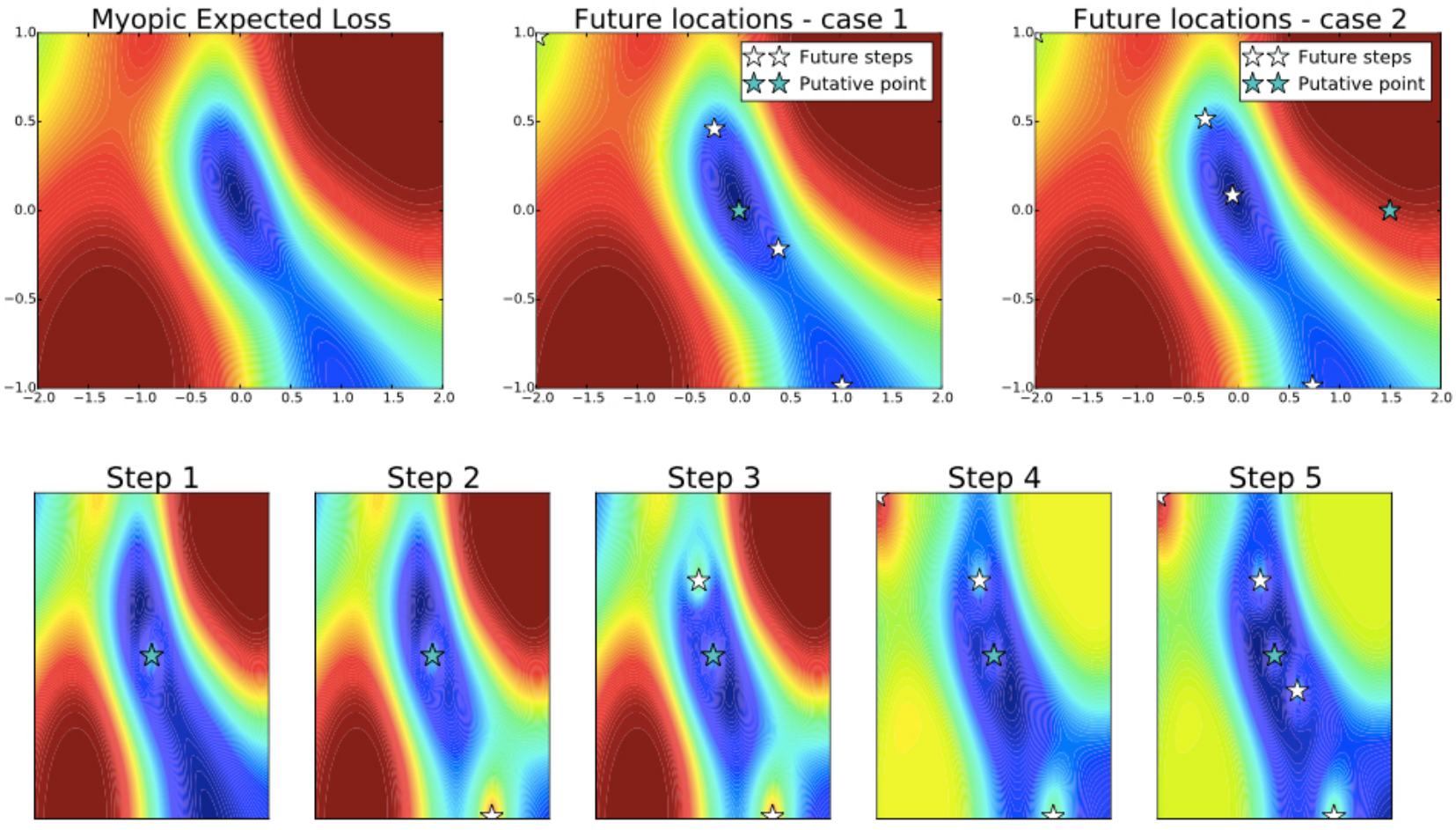
$$\begin{aligned}\mathbb{E}[\min(\mathbf{y}, \eta)] &= \eta \int_{\mathbb{R}^n} \prod_{i=1}^{n} h_i(\mathbf{y}) N(\mathbf{y}; \mu, \Sigma) \mathrm{d}\mathbf{y} \\ &+ \sum_{j=1}^{n} \int_{\mathbb{R}^n} y_j \prod_{i=1}^{n} t_{j,i}(\mathbf{y}) N(\mathbf{y}; \mu, \Sigma) \mathrm{d}\mathbf{y}\end{aligned}$$

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

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

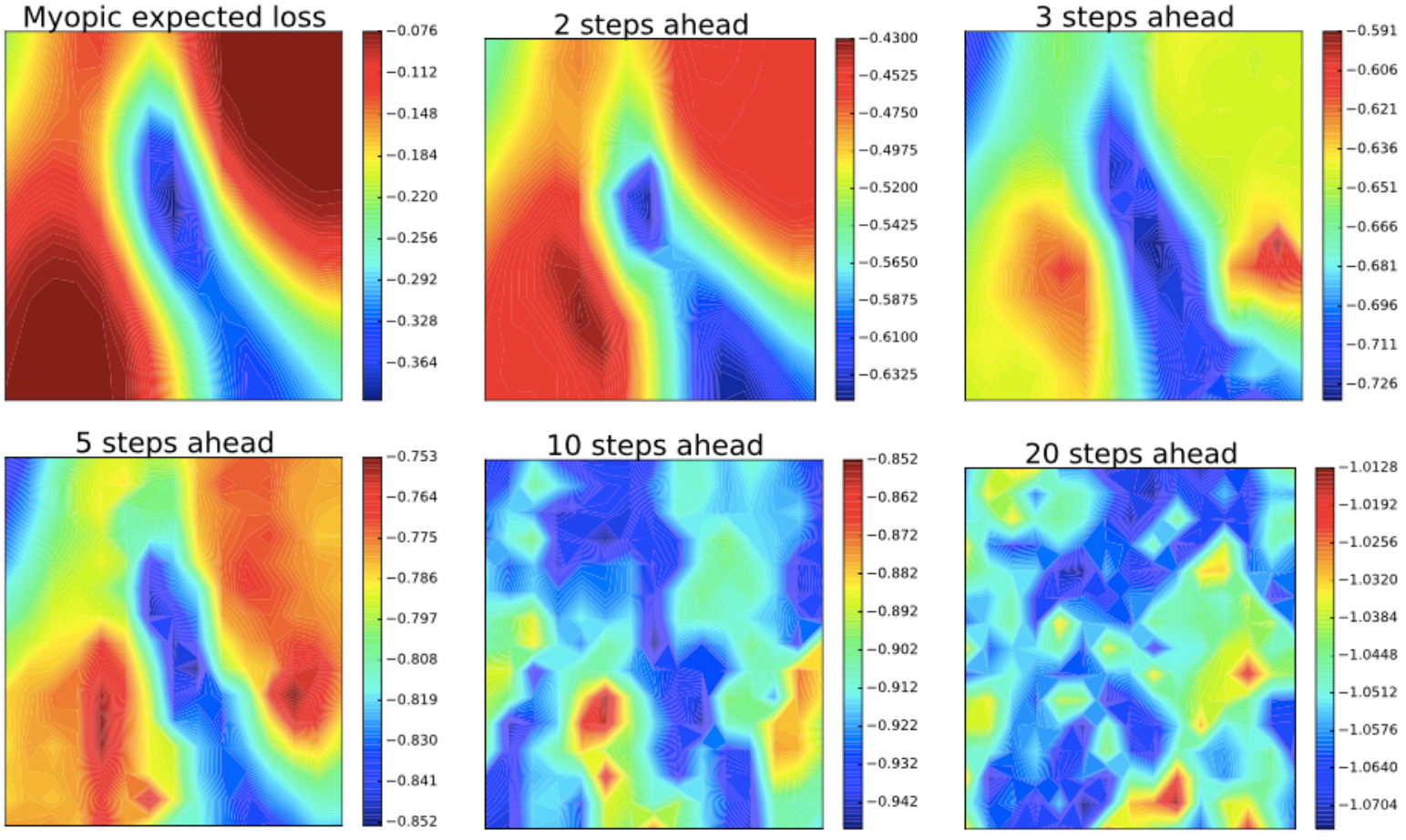
GLASSES: predicting the steps ahead

Oracle based on a batch BO method [Gonzalez, Dai, Hennig and Lawrence, 2016]



Can be interpreted as the MAP of a DPP.

GLASSES: loss function



The more steps remain: more explorative loss.

Automatic balance between exploration and exploitation.

Results in a benchmark of objectives

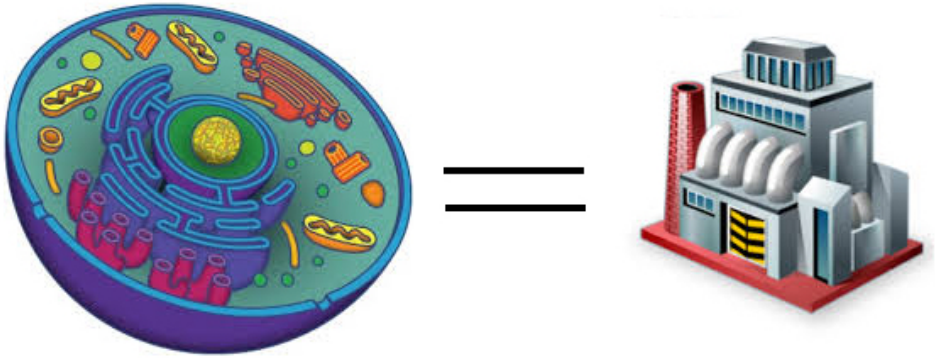
	MPI	GP-LCB	EL	EL-2	EL-3	EL-5	EL-10	GLASSES
SinCos	0.7147	0.6058	0.7645	*0.8656*	0.6027	0.4881	*0.8274*	***0.9000***
Cosines	0.8637	0.8704	0.8161	*0.8423*	*0.8118*	0.7946	0.7477	***0.8722***
Branin	0.9854	0.9616	**0.9900**	0.9856	0.9673	0.9824	0.9887	0.9811
Sixhumpcamel	0.8983	**0.9346**	0.9299	0.9115	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 overal the best method.

Make sense to use GLASSES!

Application: Synthetic gene design

[González, Lonworth, James and Lawrence, 2014, 2015]



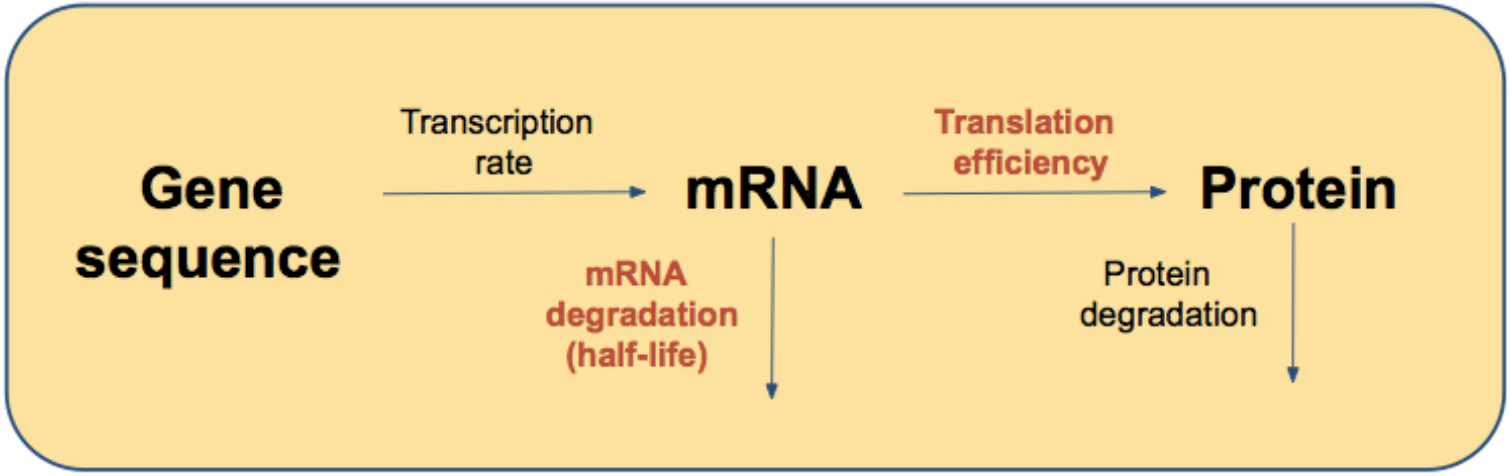
- Use mammalian cells to make protein products.
- Control the ability of the cell-factory to use synthetic DNA.

Cornerstone of modern biotechnology: Design DNA code that will best enable the cell-factory to operate most efficiently.

Synthetic gene design

Natural cells vs. cell factories

Central dogma of systems biology

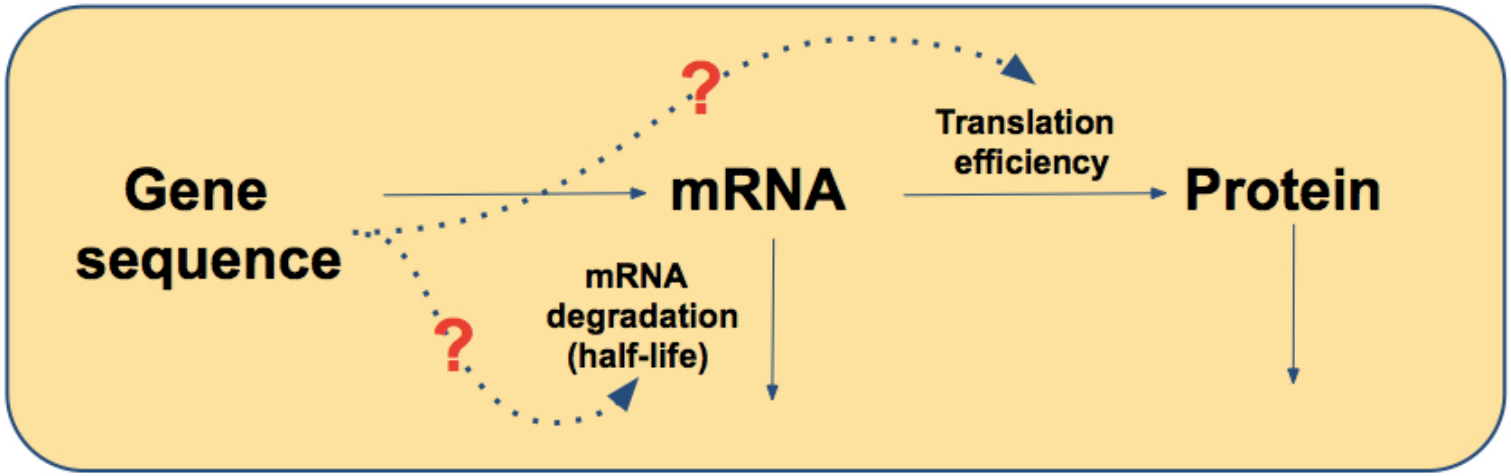


In a natural mammalian cell:

- Not all genes encode proteins of therapeutical interest.
- 'Natural' genes are not optimized to maximize protein production.

Natural cells vs. cell factories

Central dogma of systems biology



Current tools in synthetic biology allow to:

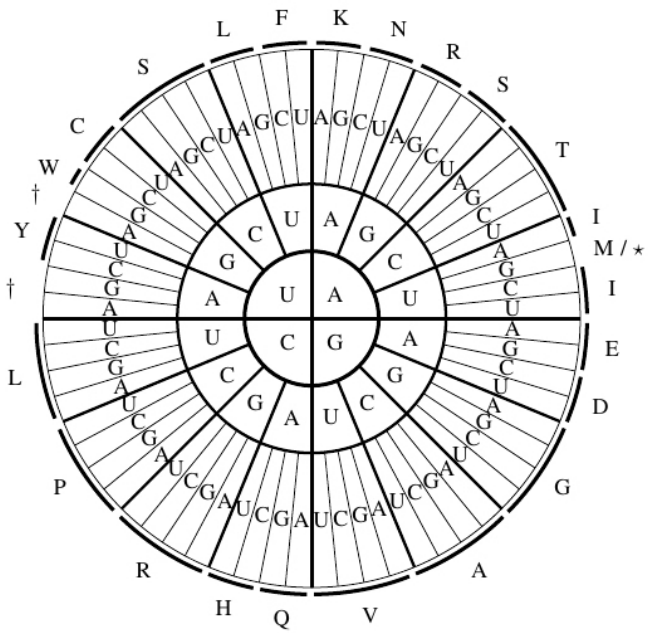
- control cell transcription...
- ...but it is unknown how to control cell translation and mRNA stability.

Key question

Develop a synthetic gene design tool to control/optimise translation

Why can we rewrite the genetic code?

- Different gene sequences may encode the same protein...
- ...but the sequence affects the synthesis efficiency.
- The codon usage is the key (codon = triplet of bases).



The genetic code is redundant:

ATGUUGACA... = *ATGUUGACU*...

Both genes encode the same protein.

Challenges

- Huge and structured design space: gene features extraction.
- Unknown mechanistic model of the cell behaviour: multioutput Gaussian processes.
- Expensive and time consuming experiments: Bayesian Optimization.

Gene features extraction

Sequence lengths

- Coding sequence length.
- 5'UTR length, 3'UTR length.

Nucleotide frequencies and properties

- 3'UTR, 5'UTR, coding region free folding energy.
- Free folding energy in 3'UTR, 5'UTR, the end of 5'UTR, the first 40 nucleotides of the coding region.
- Best local, normalized and randomize secondary structure for 3'UTR and 5'UTR with window size 40, 60.

Codons and aminoacid frequencies and properties

- Codons usage in coding region and in the first 30-50 codons in the coding region.
- Amino-acid frequencies.
- Relative synonymous codon usage.

Specific motifs frequency

- AT content.
- GC ratio.
- GC content in coding region, whole gene, starting of coding region, codon positions 1, 2 and 3.
- AUG (atg) frequency in 5'UTR (uORF frequency in 5'UTR).
- Number of A's at the beginning of the 5'UTR.
- Number of Polyadenylation sites/3'UTR (AUAAA motif).
- Motif RYMRMVAUGGC.

Codon bias indices

- Codon adaptation index in coding region.
- Codon adaptation index in first 30-50 codons.
- Codon bias index (relative codon bias sequence RCBS)

Model as an emulator of the cell behavior

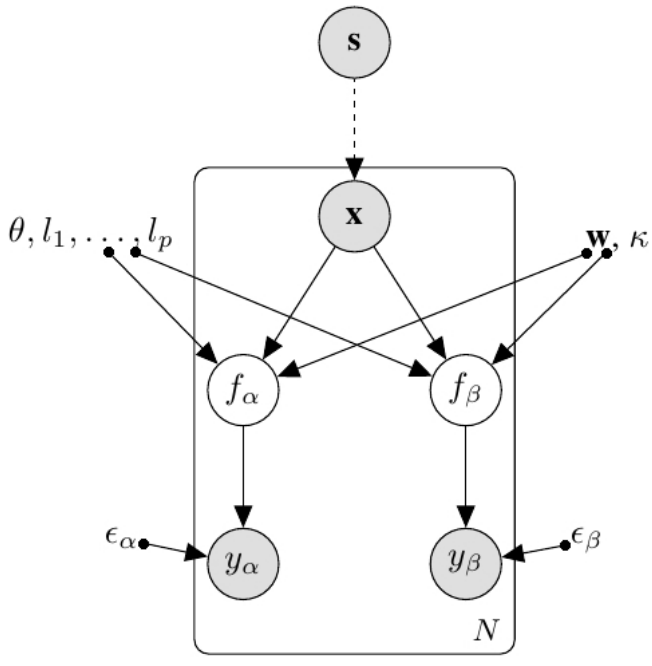
-Model inputs
Gene features ($\mathbf{x}_i$).

-Model outputs
Translation rates and mRNA half-life $\mathbf{f} := (f_\alpha, f_\beta)$.

-Model: Multi-output GP

$$\mathbf{f} \approx \mathcal{GP}(\mathbf{m}, \mathbf{K})$$

where $\mathbf{K} = \mathbf{B} \otimes \mathbf{K}_{in}$ with ARD.



Bayesian Optimization principles for gene design

[González, Lonworth, James and Lawrence, 2014]

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...).

Designing new genes coherent with the optimal design rules

Simulating-matching approach:

1. Simulate genes 'coherent' with the target.
2. Extract features.
3. Rank synthetic genes according to their similarity with the 'optimal' design rules.

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

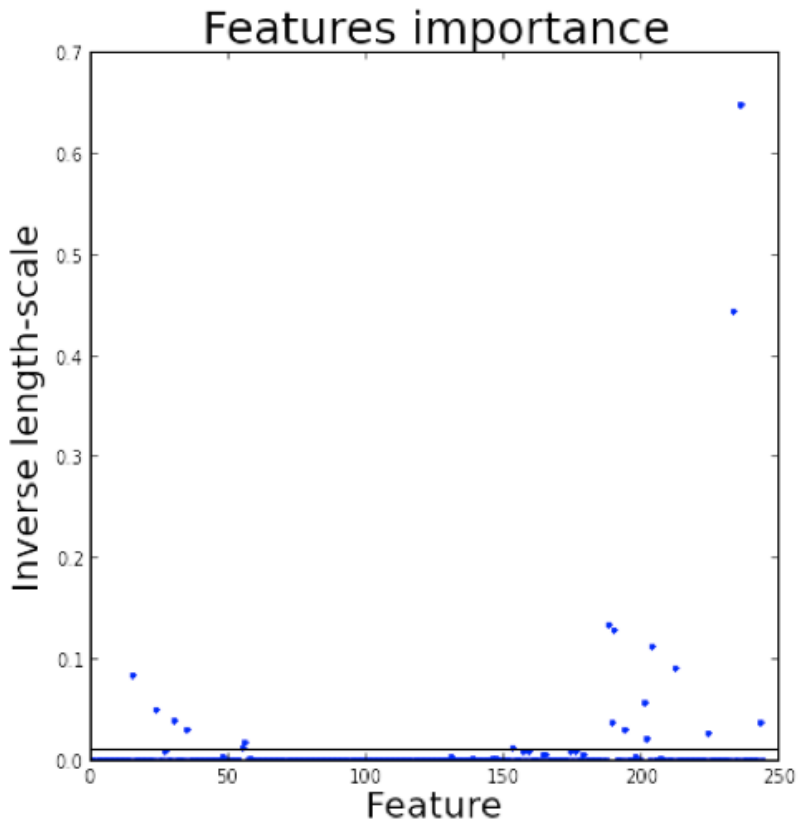
- $\mathbf{x}^\star$: optimal gene design rules.
- $\mathbf{s}$, $\mathbf{x}_j$ generated 'synonyms sequence' and its features.
- w_j: weights of the p features.

Experiments

- Dataset in Schwanhausser et al. (2011) for 3810 genes rates. Sequences extracted from http:wet-labpic/www.ensembl.org.
- 250 features involving 5'UTR, 3'UTR and coding region.
- Gaussian process with ARD and coregionalized outputs.
- Synthetic genes to produce siaP.
- 10,000 random 'synonyms sequences' generated from each gene.
- GPy and GPyOpt (https://github.com/SheffieldML/).

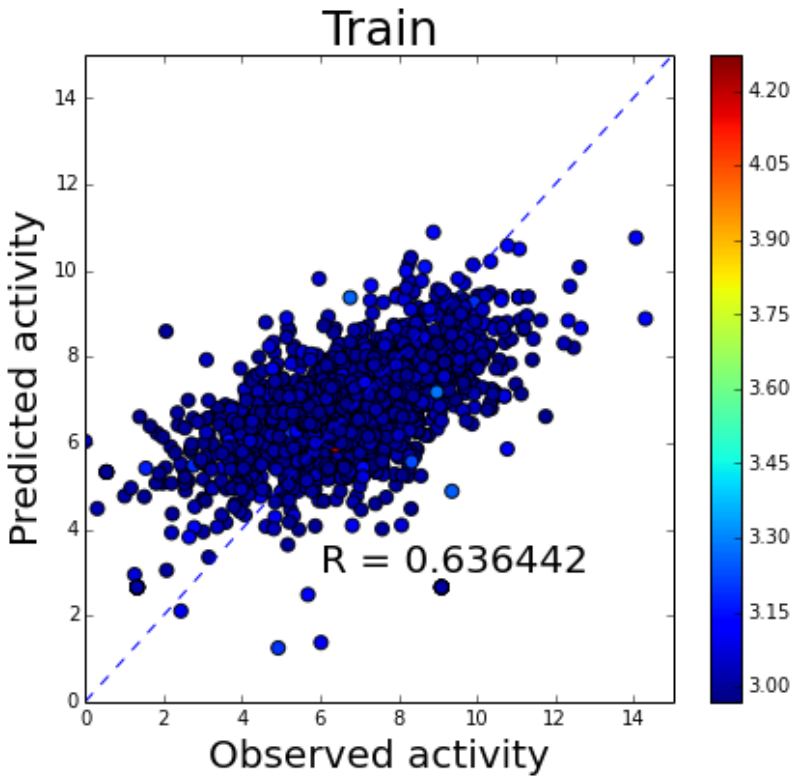
We can evaluate the gene features relevance

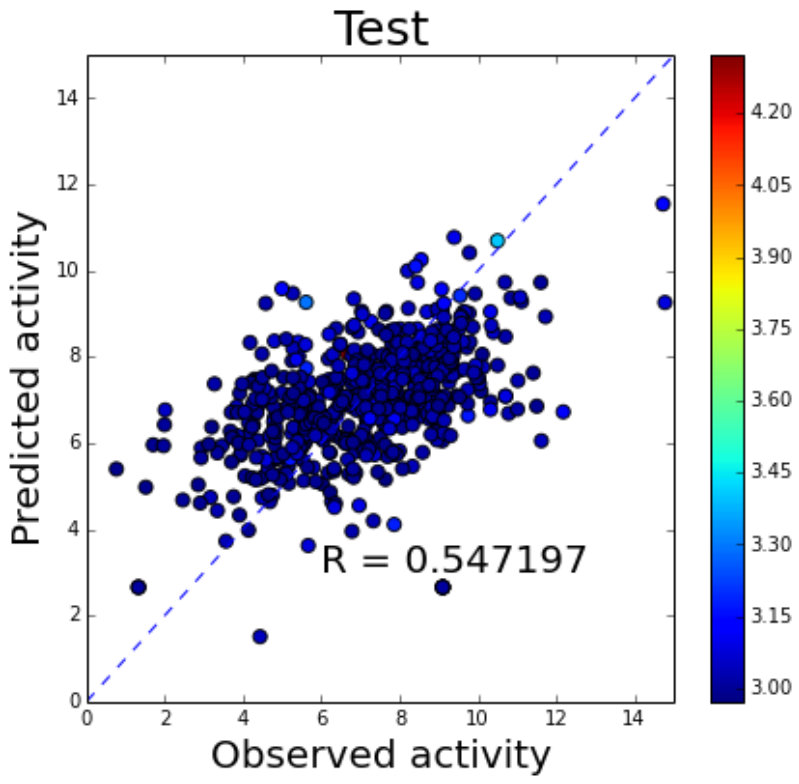
Feature	Score
5' UTR free fold energy	0.644
5' UTR length	0.443
number of stop codons	0.134
Cysteine	0.128
Serine	0.112
Length	0.090
Codon ATT	0.084
Proline	0.057
Codon CGA	0.050
Codon CTG	0.038
Alanine	0.037
Free folding energy (size window 60) in 5'UTR	0.036
Glycine	0.029
Codon GAT	0.029
3' UTR length	0.027



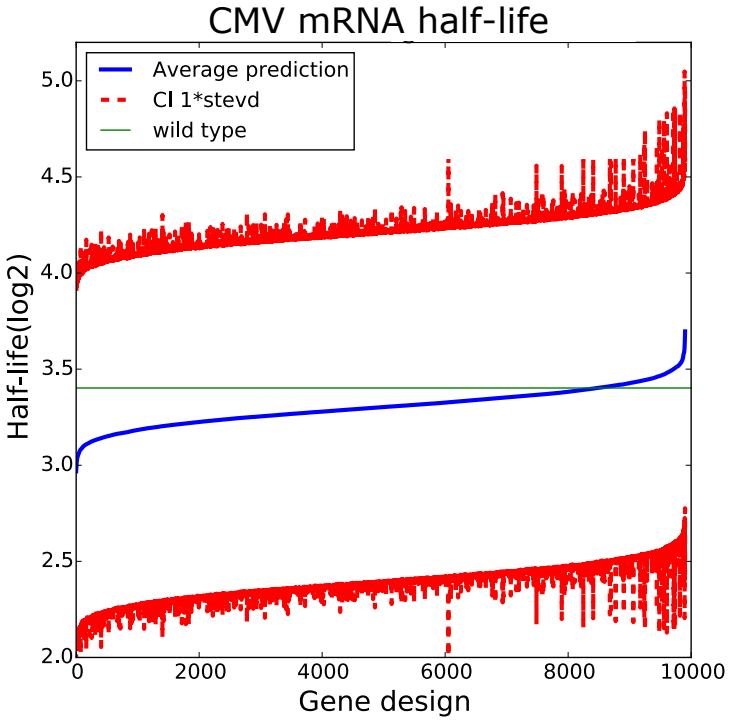
A few number of features are relevant

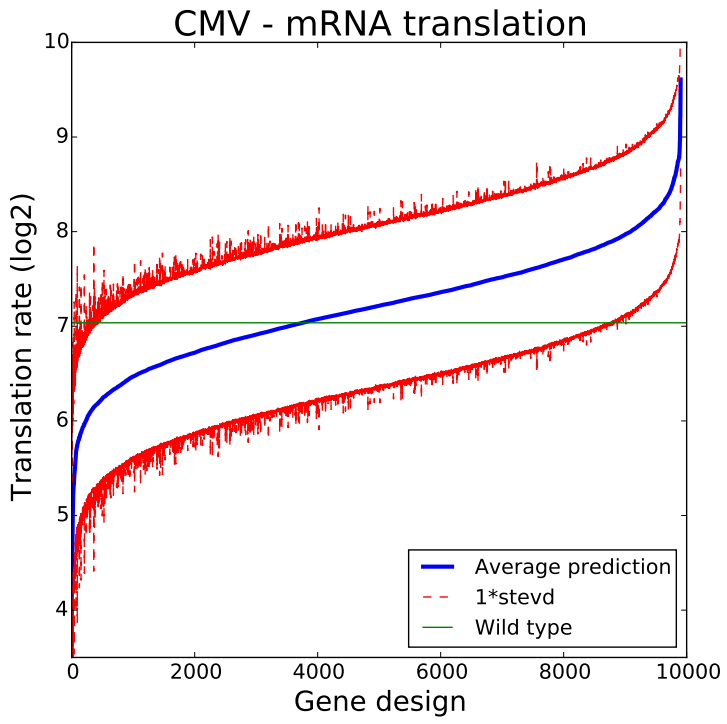
The model is able to predict translation rates





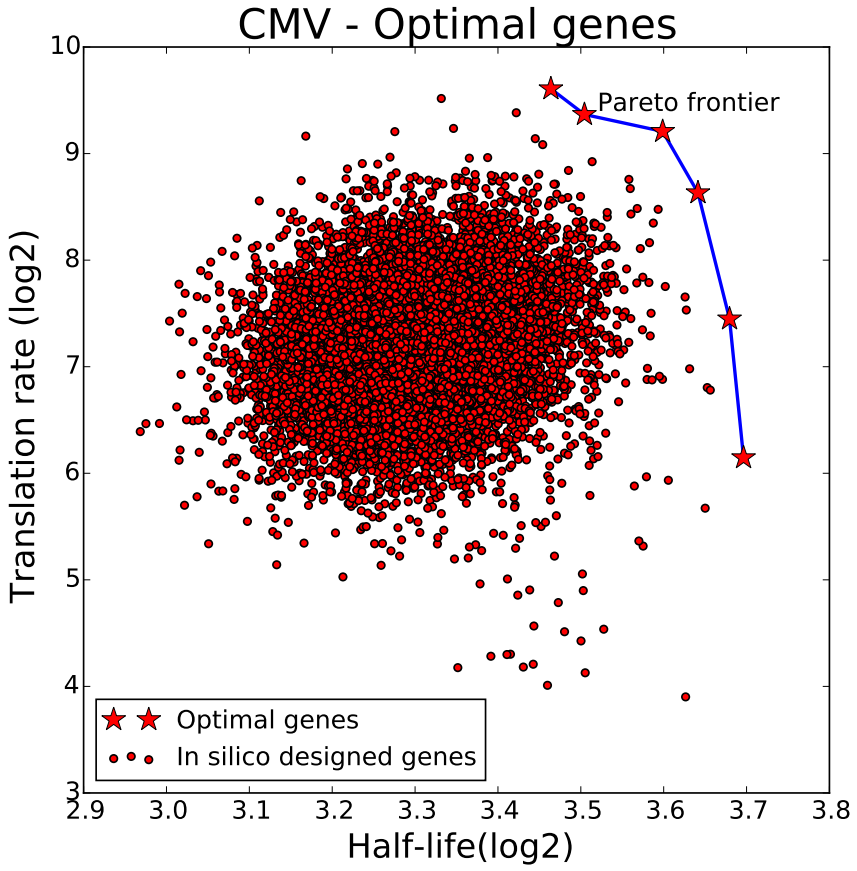
We can use the model to control translation





Ranking of 10,000 recombinant simulate sequences for the average translation rates and mRNA half-life.

Optimal multi-objective designs



Multi-objective optimization problem.

Wrapping up

- BO is fantastic tool for global parameter optimization in ML and experimental design.
- The model and the acquisition function are the two most important bits.
- Non myopic approach are needed to find good balance between exploration and exploitation.
- Software available! Use GPyOpt!

Many thanks to

- Neil Lawrence, University of Sheffield.
- Zhenwen Dai, University of Sheffield.
- Philipp Hennig, Max Planck institute.
- Michael Osborne, University of Oxford.
- David James, Joseph Longworth and others at CBE, University of Sheffield.