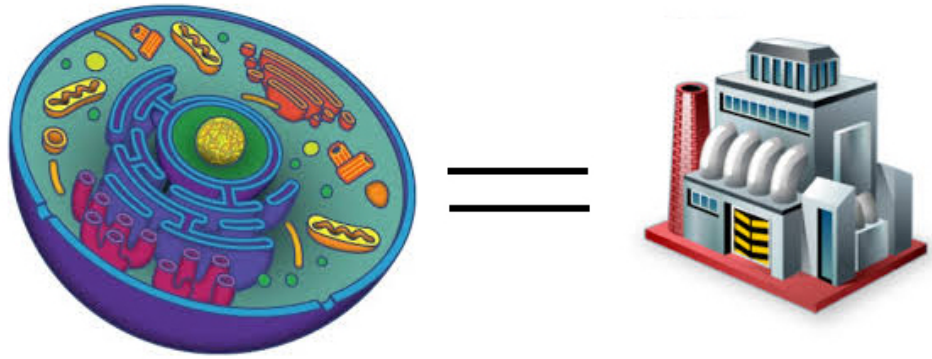
Model based optimization and experimental design

Javier González

Microsoft Research Cambridge

Research question

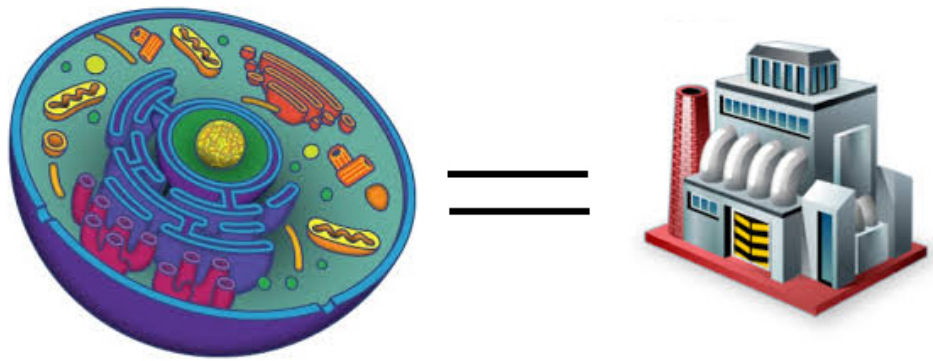


- Use mammalian cells to make protein products.
- Control the ability of the cell-factory to use synthetic DNA.

How can we optimize genes (ATTGGTUGA...) to best enable the cell-factory to operate most efficiently?

Problem definition

Ojective function f: mapping from genes to amount of protein produced.



- f is explicitly unknown and multimodal.
- Evaluations of f may be perturbed by noise.
- Evaluations of f are expensive (require a lab experiment)

How to maximize f if we can run a limited number of labs?

Big picture

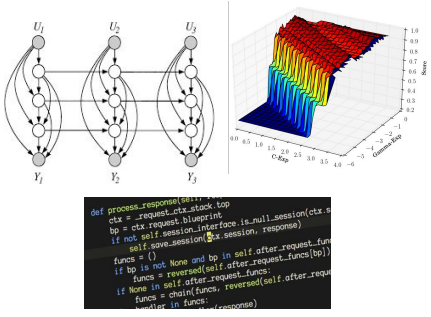
"Civilization advances by extending the number of important operations which we can perform without thinking of them."
(Alfred North Whitehead)

We are interested on automation:

- Automatic model configuration.
- Automate the design of physical experiments.

Data science pipeline/Autonomous system

Challenges and needs for automation



Data Collection

Optimal design

Features extraction

Filters, dimensionality reduction

Modelling

Model tuning and configuration, code optimisation

Results Interpretation/ Decision

Data visualisation, Sequential experimentation

Pipeline improvement / Interaction with environment

Experimental Design - Uncertainty Quantification

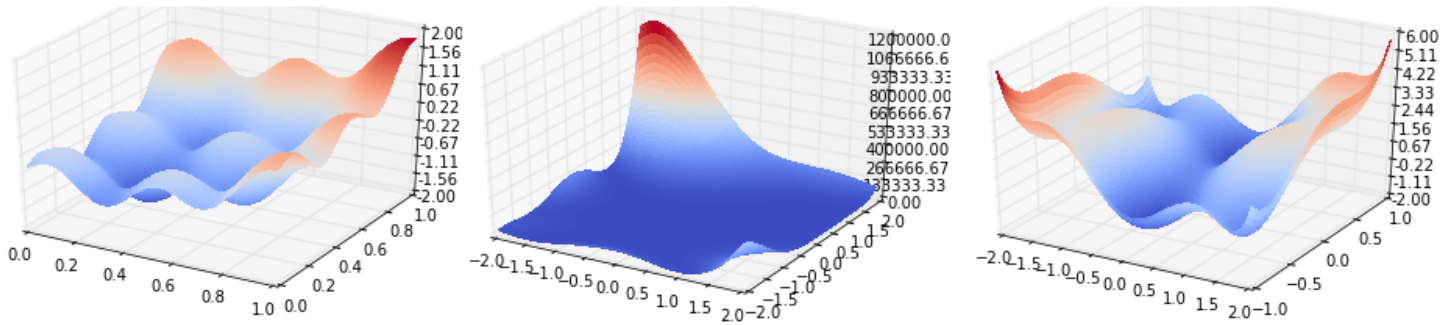
Can we automate/simplify the process of designing complex experiments?

Emulator - Simulator - Physical system

Problem definition

$f : \mathcal{X} \to \mathbb{R}$ where $\mathcal{X} \subseteq \mathbb{R}^D$ is 'well behaved' function is a bounded domain. Find

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

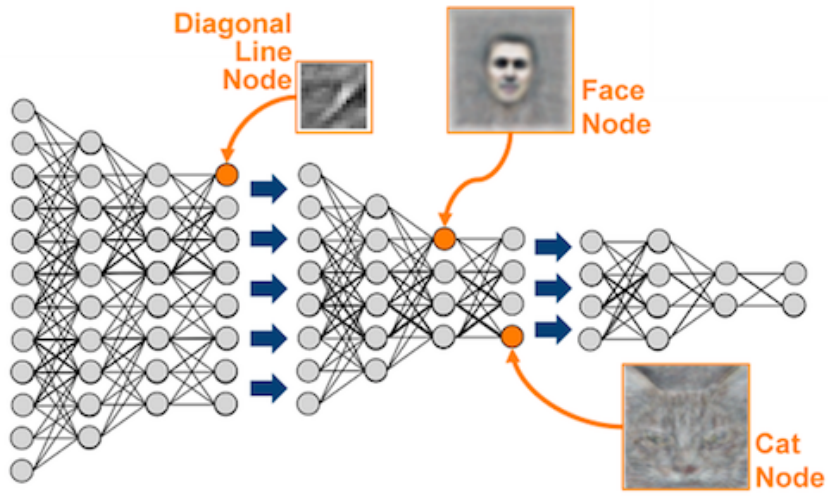


- f is explicitly unknown and multimodal.
- Evaluations of f may be perturbed by noise.
- Evaluations of f are expensive.

Applications to hyper-parameter optimization, robotics, intractable likelihoods, molecules design, etc.

Expensive functions, who doesn't have one?

Parameter tuning in ML algorithms.



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

Figure source: http://theanalyticsstore.com/deep-learning

Expensive functions, who doesn't have one?

Many other problems:

- Robotics, control, reinforcement learning.
- Scheduling, planning.
- Compilers, hardware, software.
- Industrial design.
- Intractable likelihoods.

What to do?

Option 1: Use previous knowledge

To select the parameters at hand. Perhaps not very scientific but still in use...

What to do?

Option 2: Grid search?

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

$$f(x_M) - f(x_{M,n}) \leq \epsilon$$

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

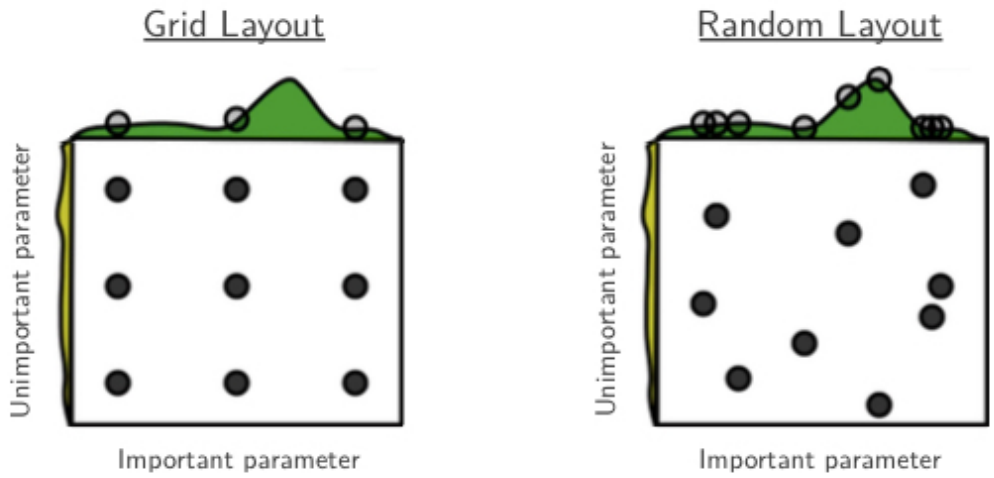
$$(L/\epsilon)^D evaluations!$$

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

What to do?

Option 3: Random search?

We can sample the space uniformly [Bergstra and Bengio 2012]



Better than grid search in various senses but still expensive to guarantee good coverage.

What to do?

Key question:

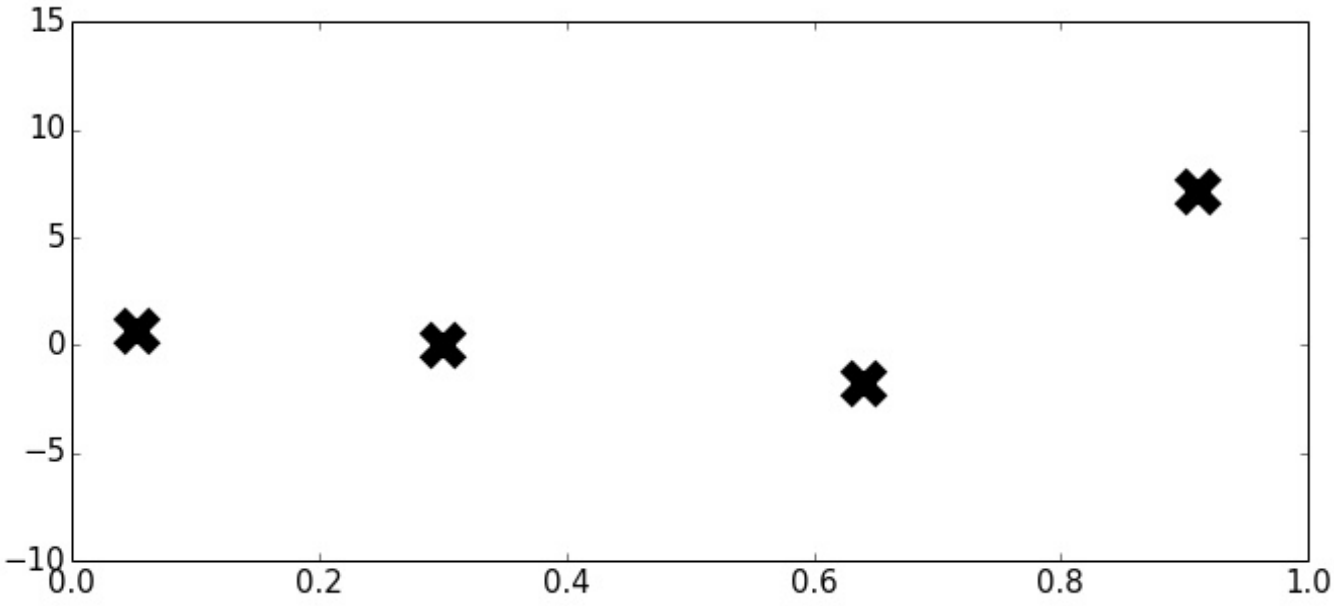
Can we do better?

Problem (the audience is encouraged to participate!)

- Find the optimum of some function f in the interval [0,1].
- f is (L-Lipchitz) continuous and differentiable.
- Evaluations of f are exact and we have 4 of them!

Situation

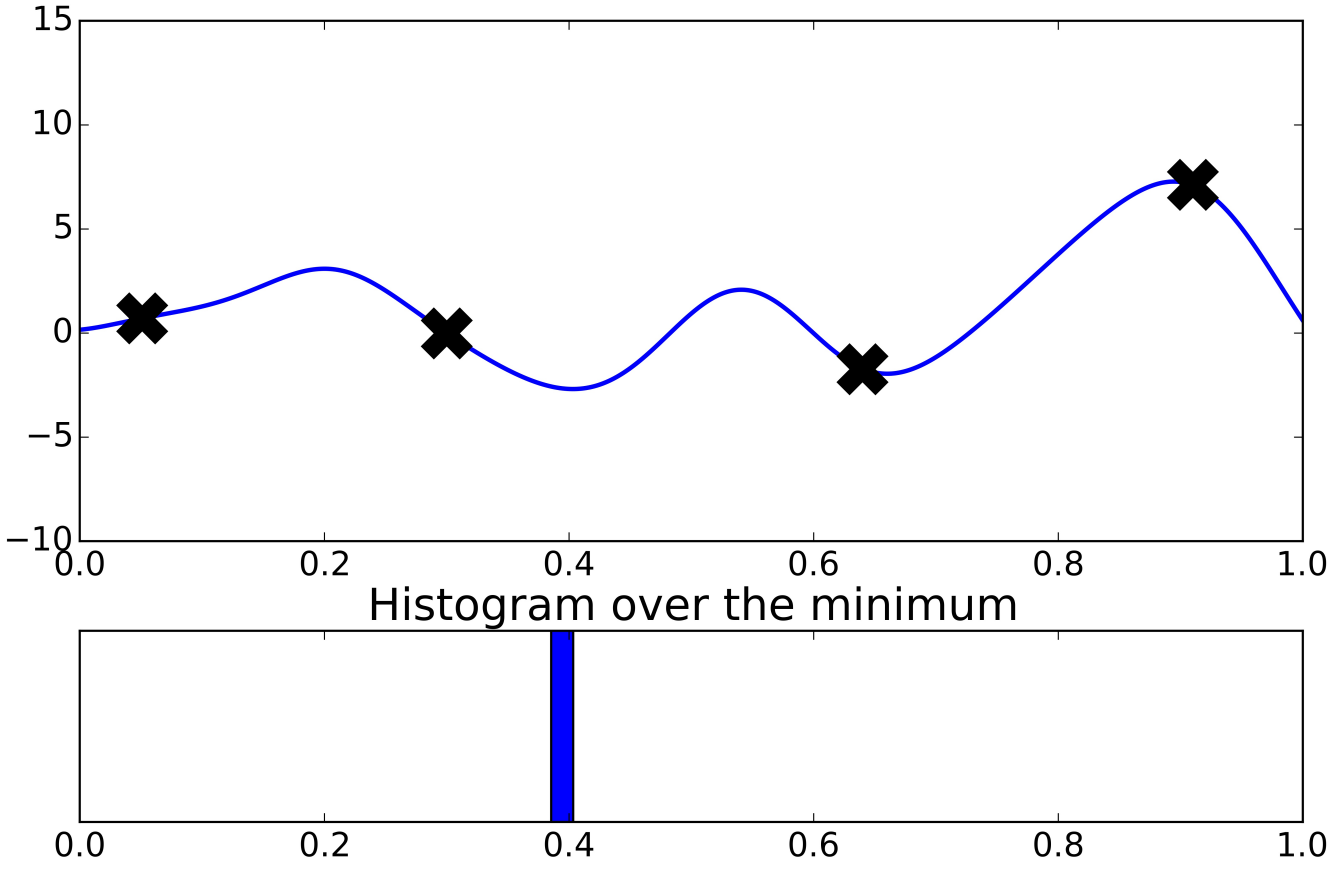
We have a few function evaluations



Where is the minimum of f?
Where should we take the next evaluation?

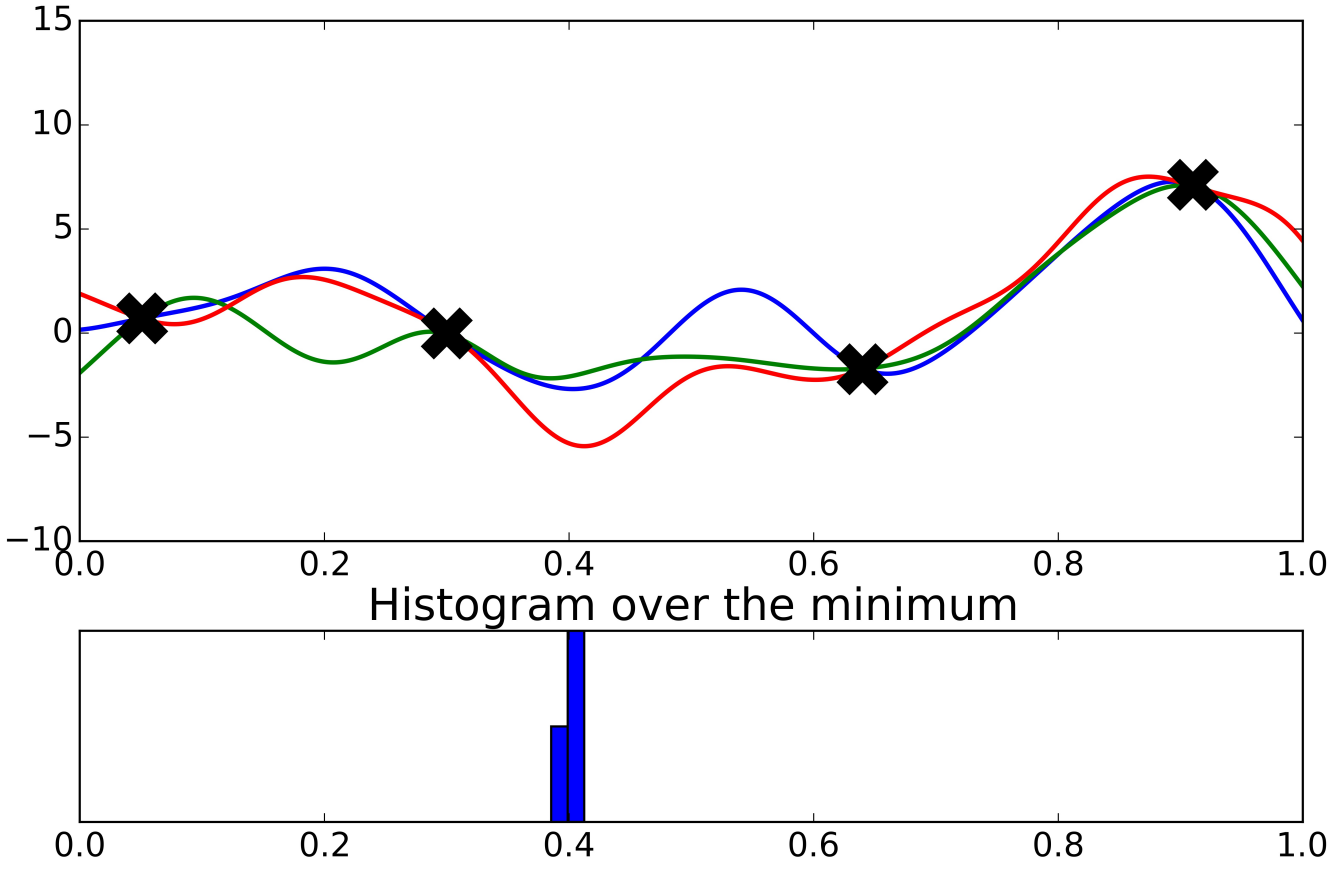
Intuitive solution

One curve



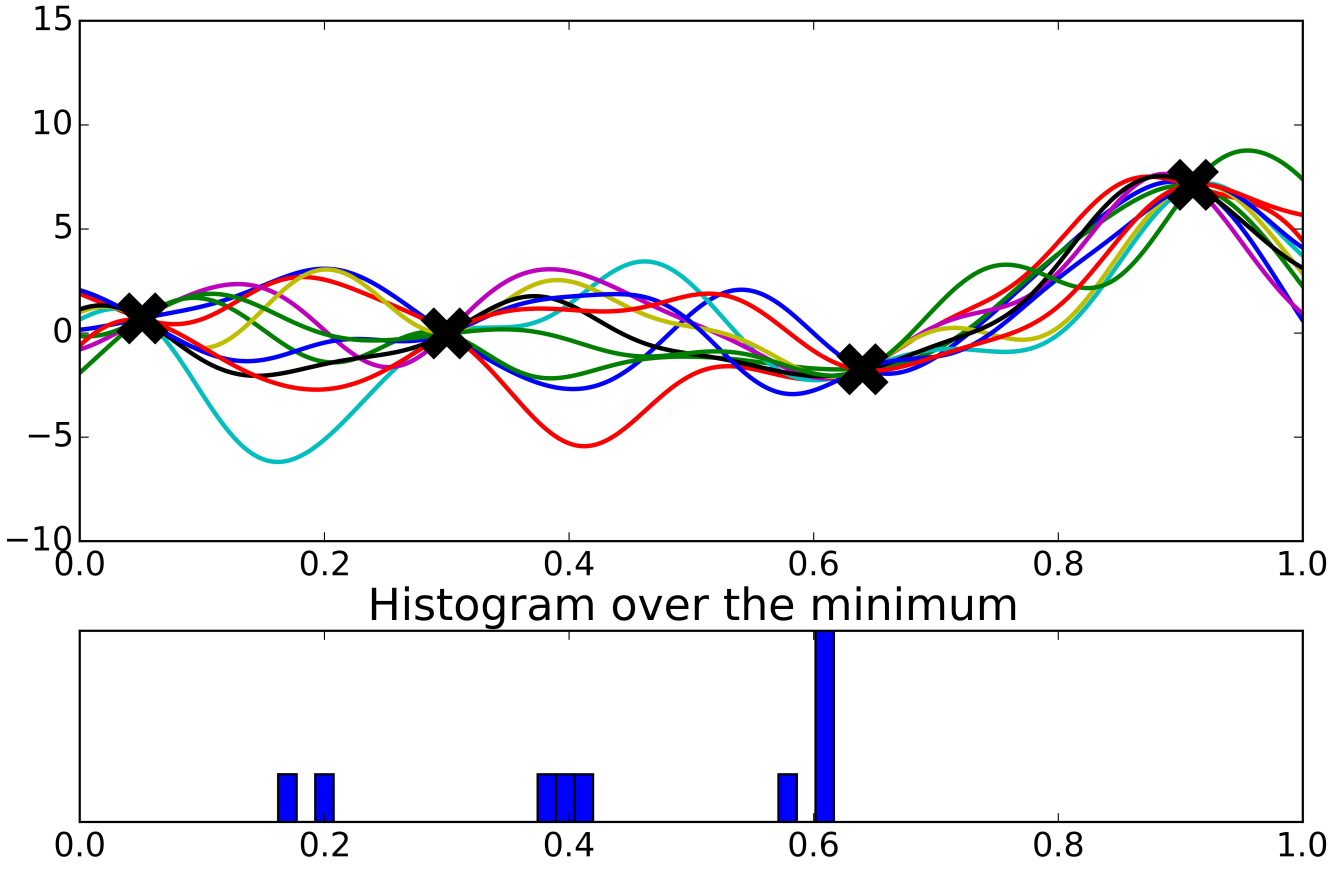
Intuitive solution

Three curves



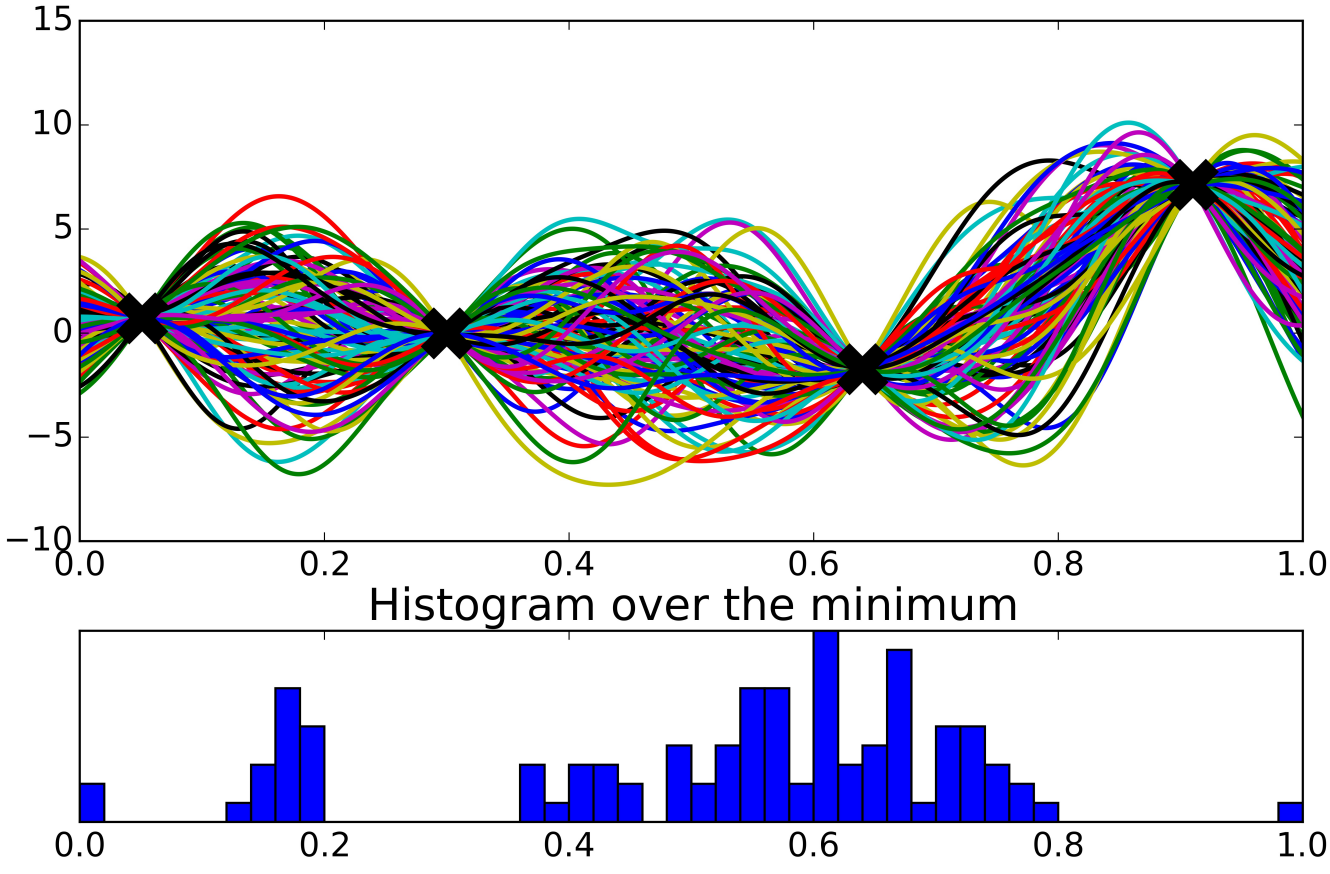
Intuitive solution

Ten curves



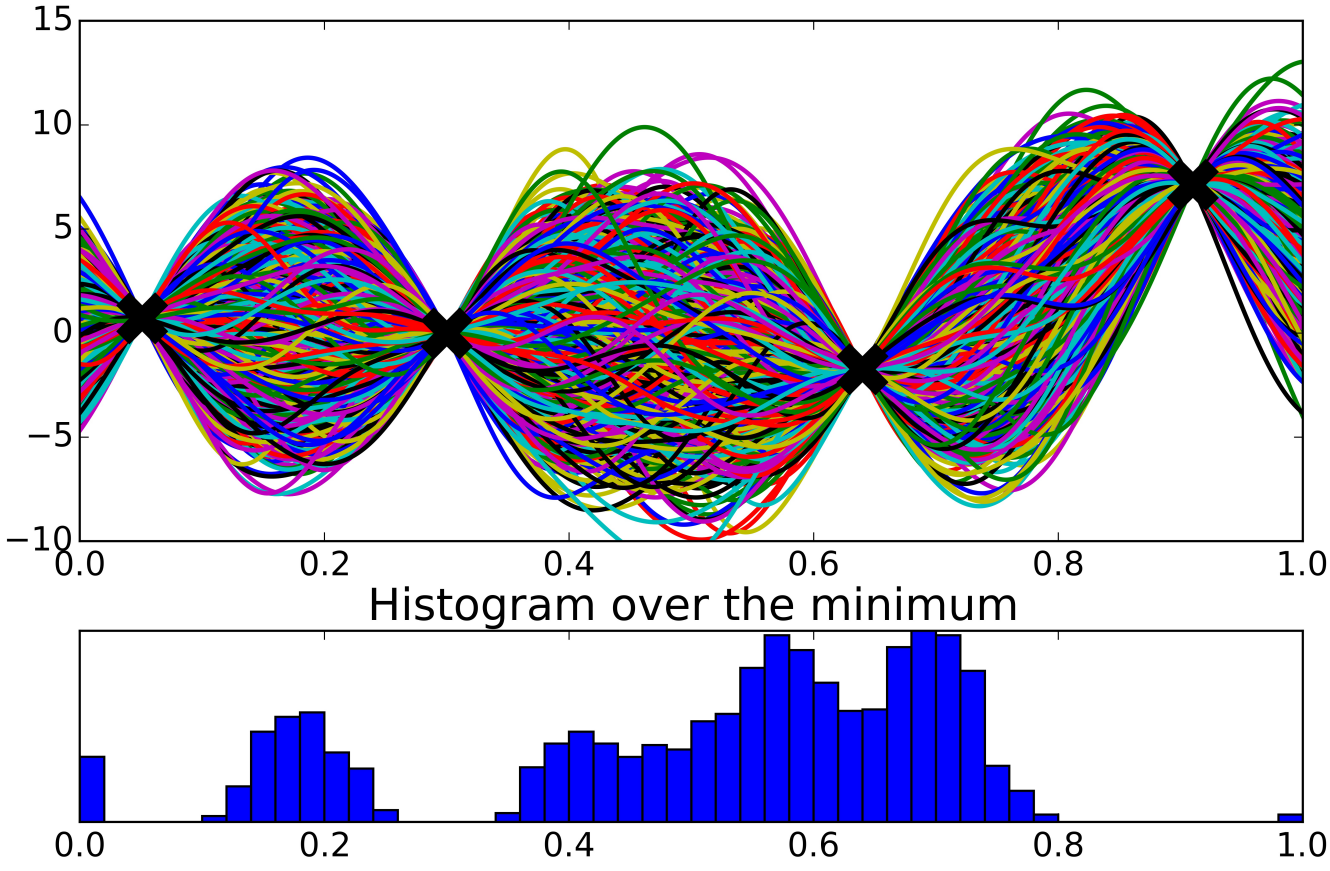
Intuitive solution

Hundred curves



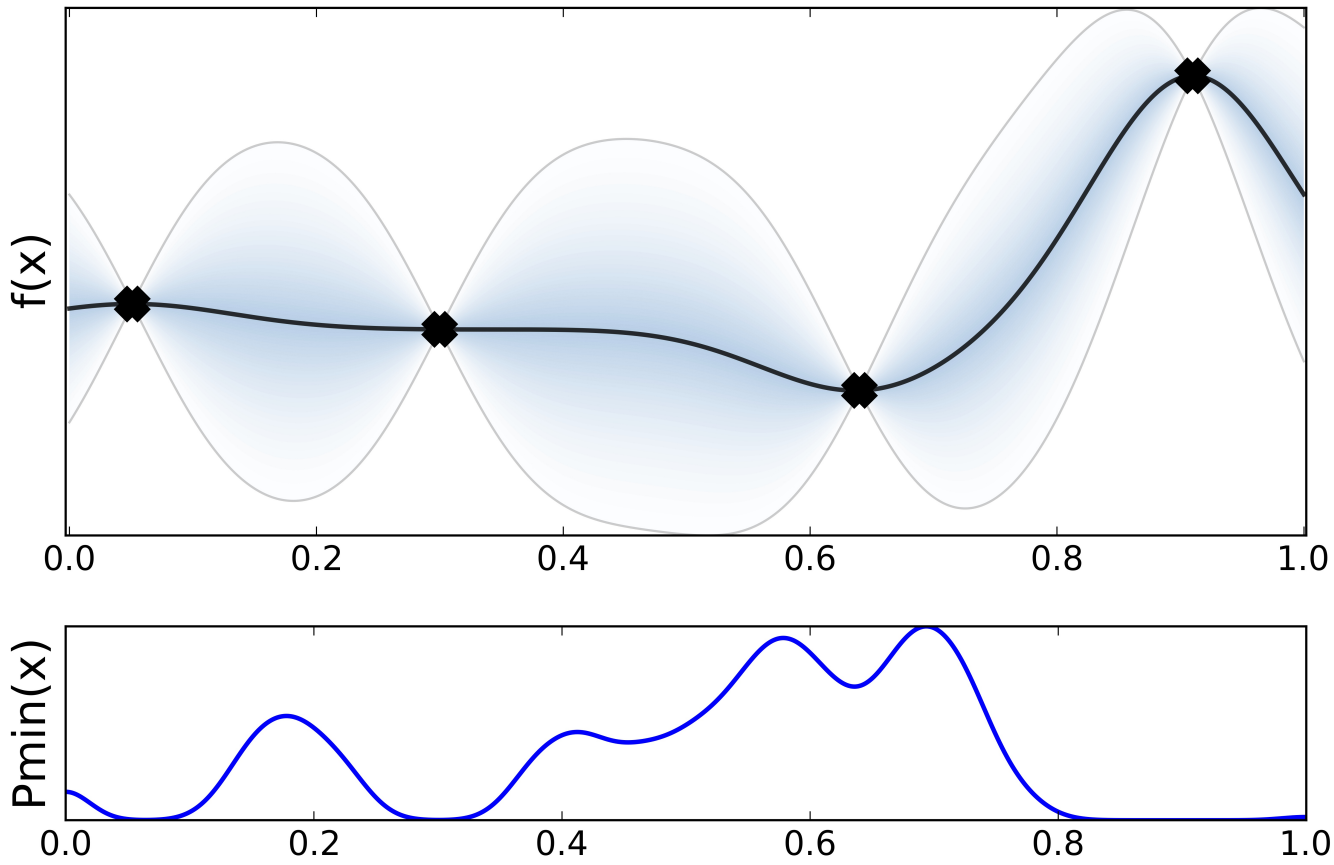
Intuitive solution

Many curves



Intuitive solution

Infinite curves



Surrogate modelling

1. Use a surrogate model of f to carry out the optimization.

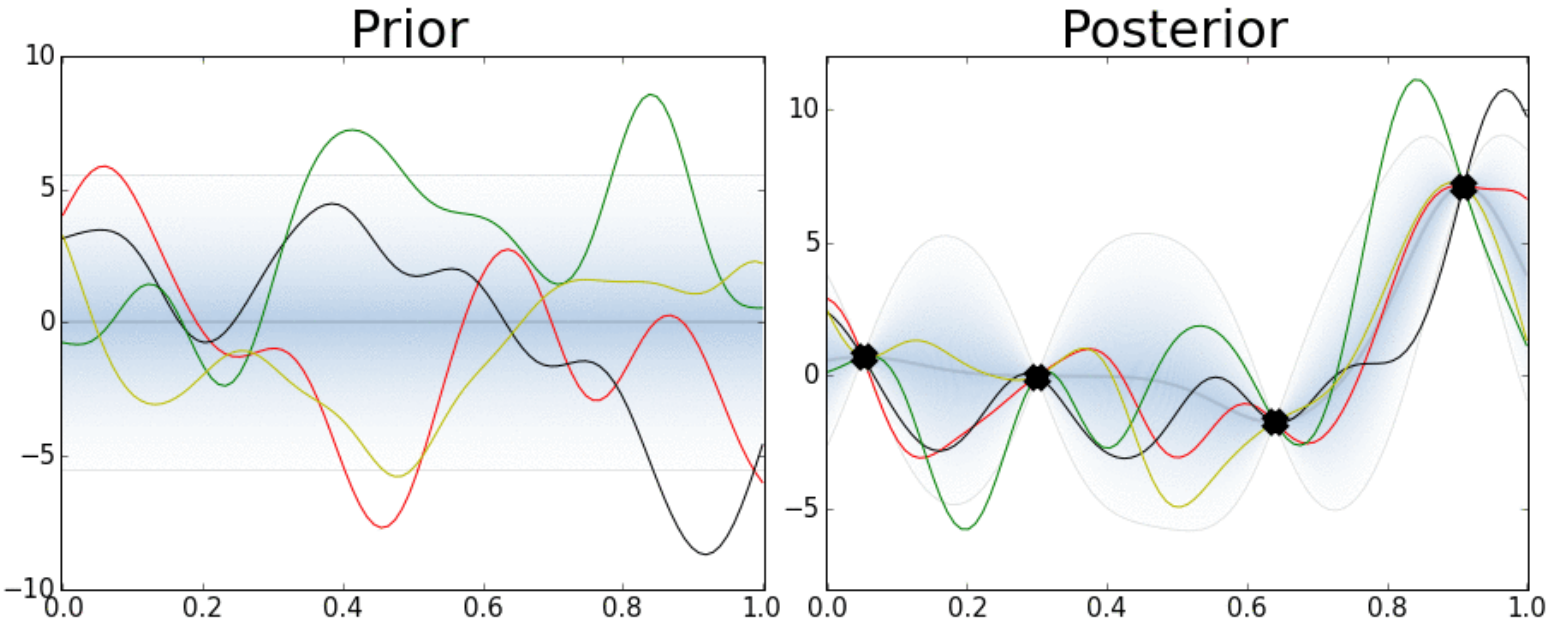
2. Define an utility function to collect new data points satisfying some optimality criterion: *optimization* as *decision*.

3. Study *decision* problems as *inference* using the surrogate model: use a probabilistic model able to calibrate both, epistemic and aleatoric uncertainty.

Surrogate model: Gaussian process

Default Choice: Gaussian processes [Rasmunsen and Williams, 2006]

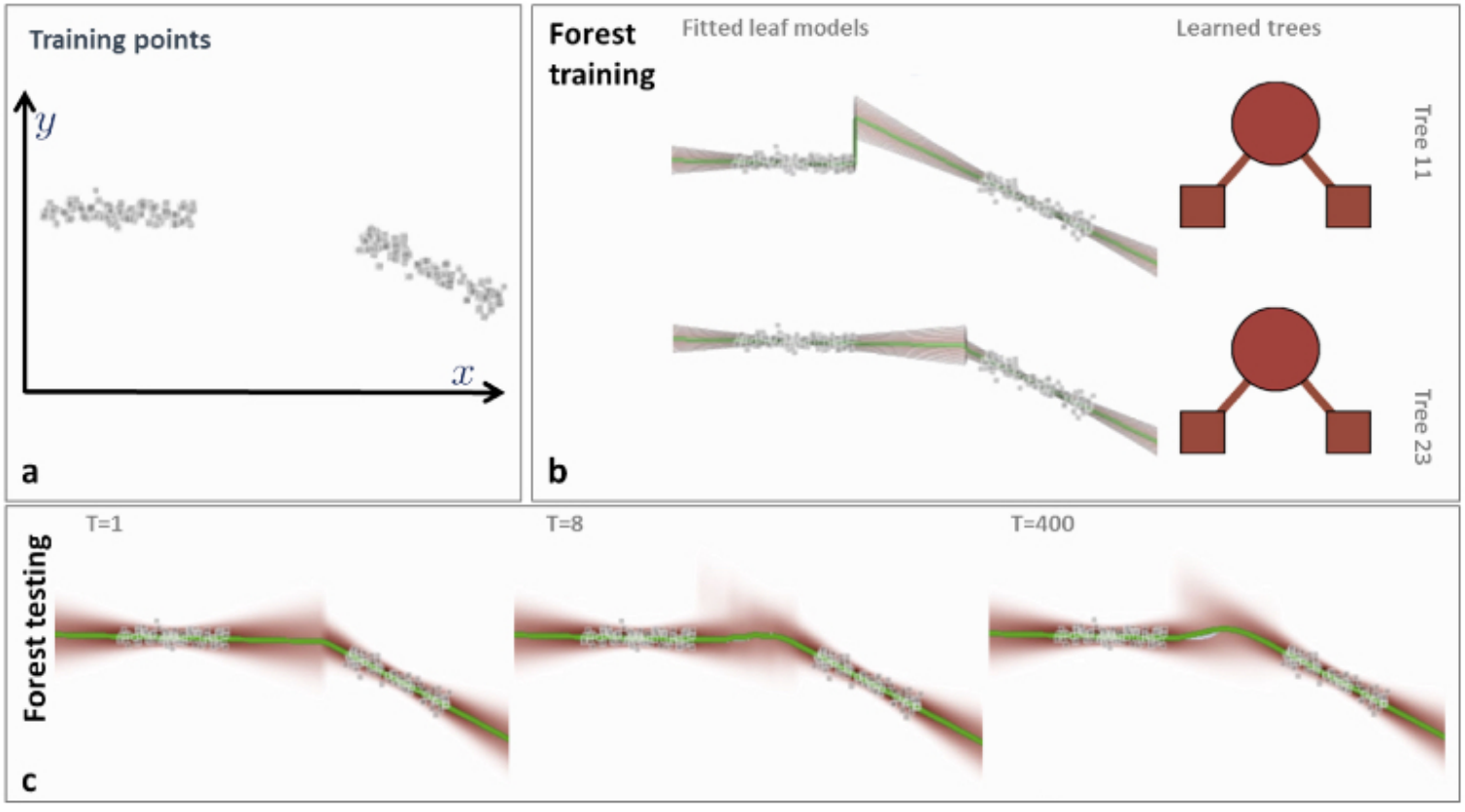
Infinite-dimensional probability density, such that each linear finite-dimensional restriction is multivariate Gaussian.



- Model $f(x) \sim \mathcal{GP}(\mu(x), k(x, x'))$ is determined by the *mean function* $m(x)$ and *covariance function* $k(x, x'; \theta)$.

Other models are also possible: Random Forrest

[Criminisi et al, 2011]



Other models are also possible: t-Student processes

Student-t Processes as Alternatives to Gaussian Processes

Amar Shah
University of Cambridge

Andrew Gordon Wilson
University of Cambridge

Zoubin Ghahramani
University of Cambridge

Abstract

We investigate the Student-t process as an alternative to the Gaussian process as a non-parametric prior over functions. We derive closed form expressions for the marginal likelihood and predictive distribution of a Student-t process, by integrating away an

simple exact learning and inference procedures, and impressive empirical performances [Rasmussen, 1996], Gaussian processes as kernel machines have steadily grown in popularity over the last decade.

At the heart of every Gaussian process (GP) is a parametrized covariance kernel, which determines the properties of likely functions under a GP. Typically simple parametric kernels, such as the Gaus-

Exploration vs. exploitation

The exploration exploitation dilemma is present in most of our day-by-day decisions.

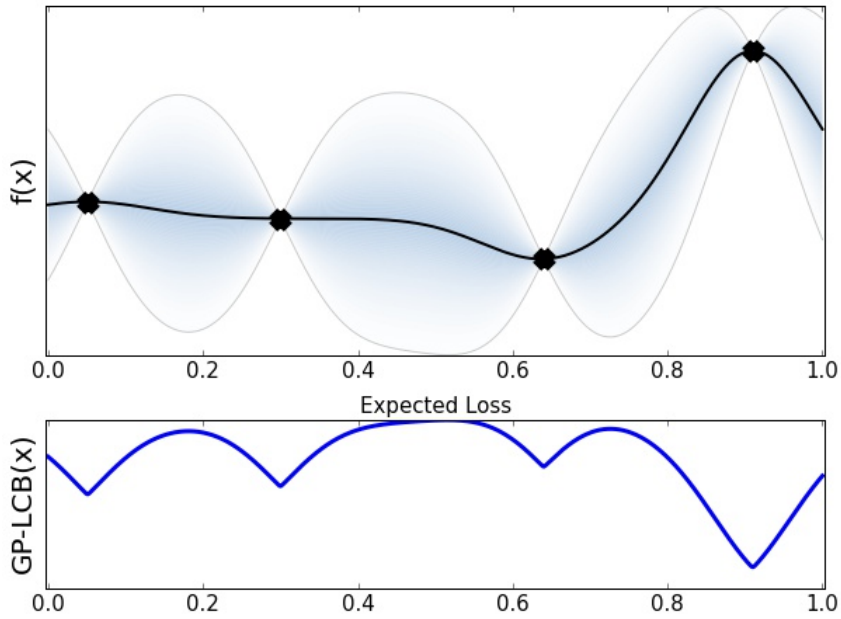
Bayesian reasoning.

GP upper (lower) confidence band

[Srinivas et al., 2010]

Direct balance between exploration and exploitation:

$$\alpha_{LCB}(\mathbf{x}; \theta, \mathcal{D}) = -\mu(\mathbf{x}; \theta, \mathcal{D}) + \beta_t \sigma(\mathbf{x}; \theta, \mathcal{D})$$



GP upper (lower) confidence band

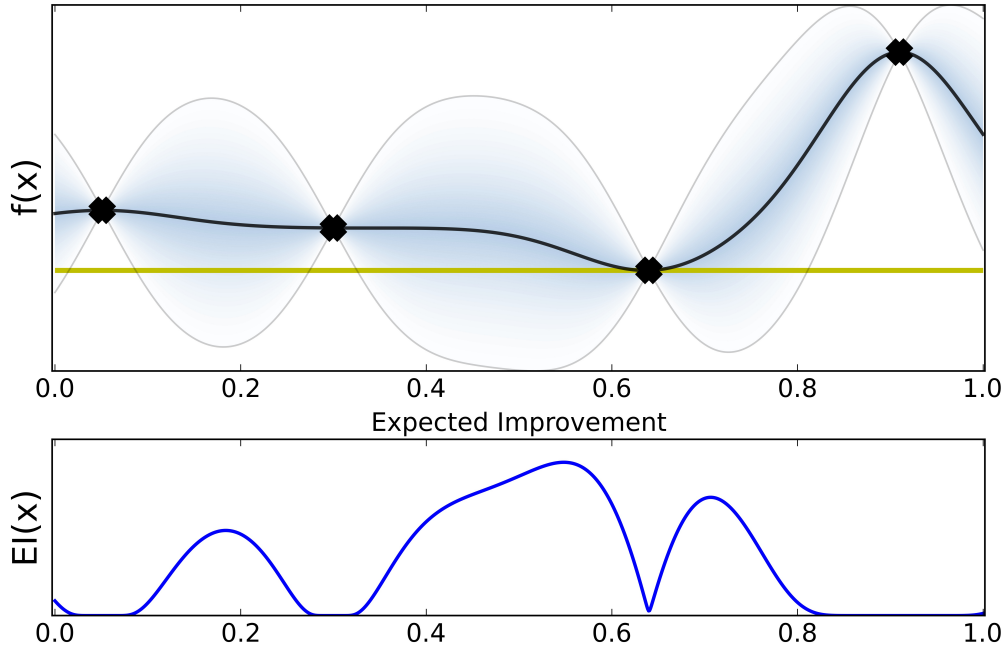
[Srinivas et al., 2010]

- In noiseless cases, it is a lower bound of the function to minimize.
- This allows to compute a bound on how close we are to the minimum.
- Optimal choices available for the 'regularization parameter'.

Expected improvement

[Jones et al., 1998]

$$\alpha_{EI}(\mathbf{x};\theta,\mathcal{D}) = \int_y \max(0, y_{best} - y) p(y|\mathbf{x};\theta,\mathcal{D}) dy$$



Expected improvement

[Jones et al., 1998]

- Perhaps the most used acquisition.
- Explicit form available for Gaussian posteriors.
- It is too greedy in some problems. It is possible to make more explorative adding an 'explorative' parameter

$$\alpha_{EI}(\mathbf{x}; \theta, \mathcal{D}) = \sigma(\mathbf{x}; \theta, \mathcal{D})(\gamma(x)\Phi(\gamma(x))) + \mathcal{N}(\gamma(x); 0, 1).$$

where

$$\gamma(x) = \frac{f(x_{best}) - \mu(\mathbf{x}; \theta, \mathcal{D}) + \psi}{\sigma(\mathbf{x}; \theta, \mathcal{D})}.$$

Illustration of BO

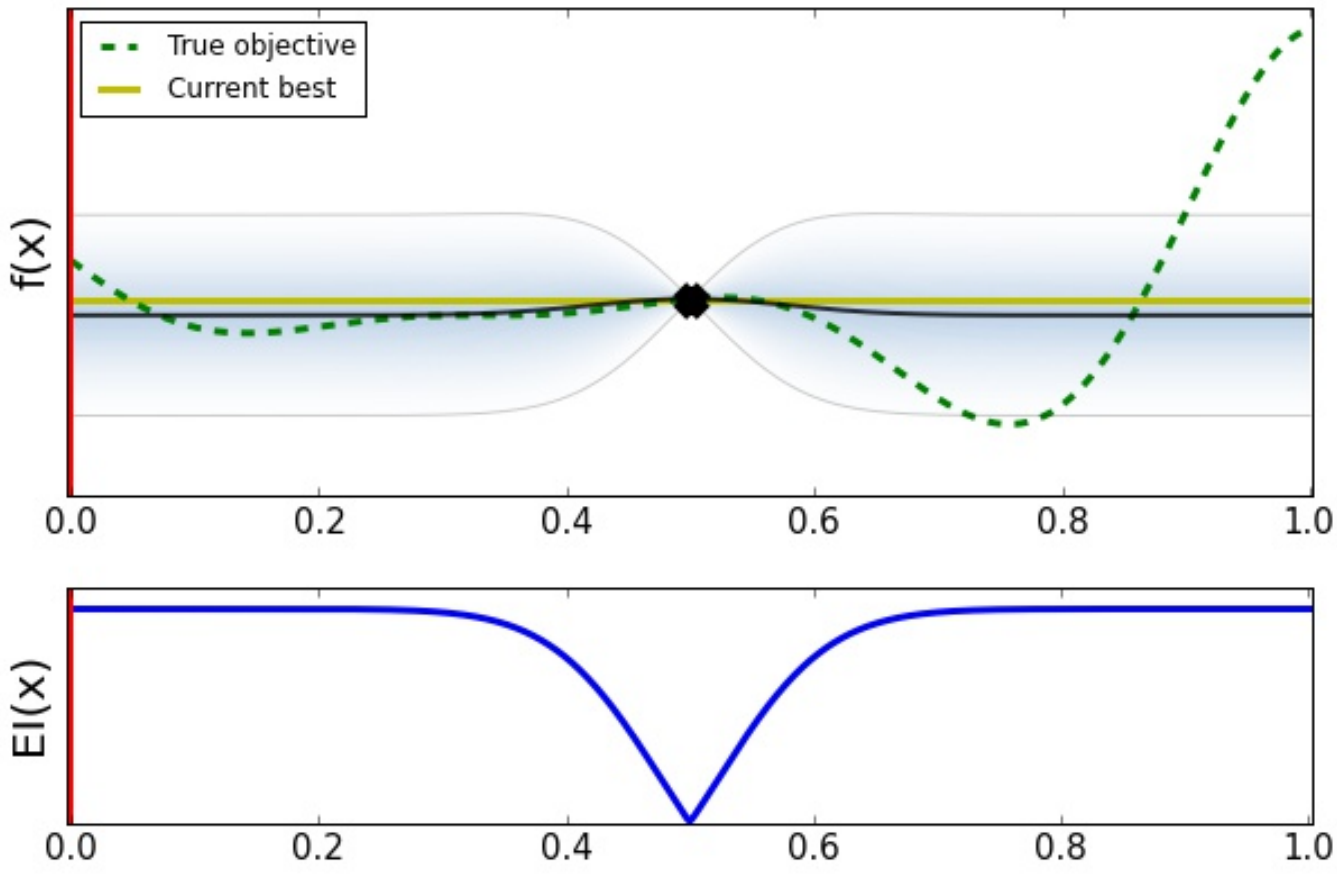


Illustration of BO

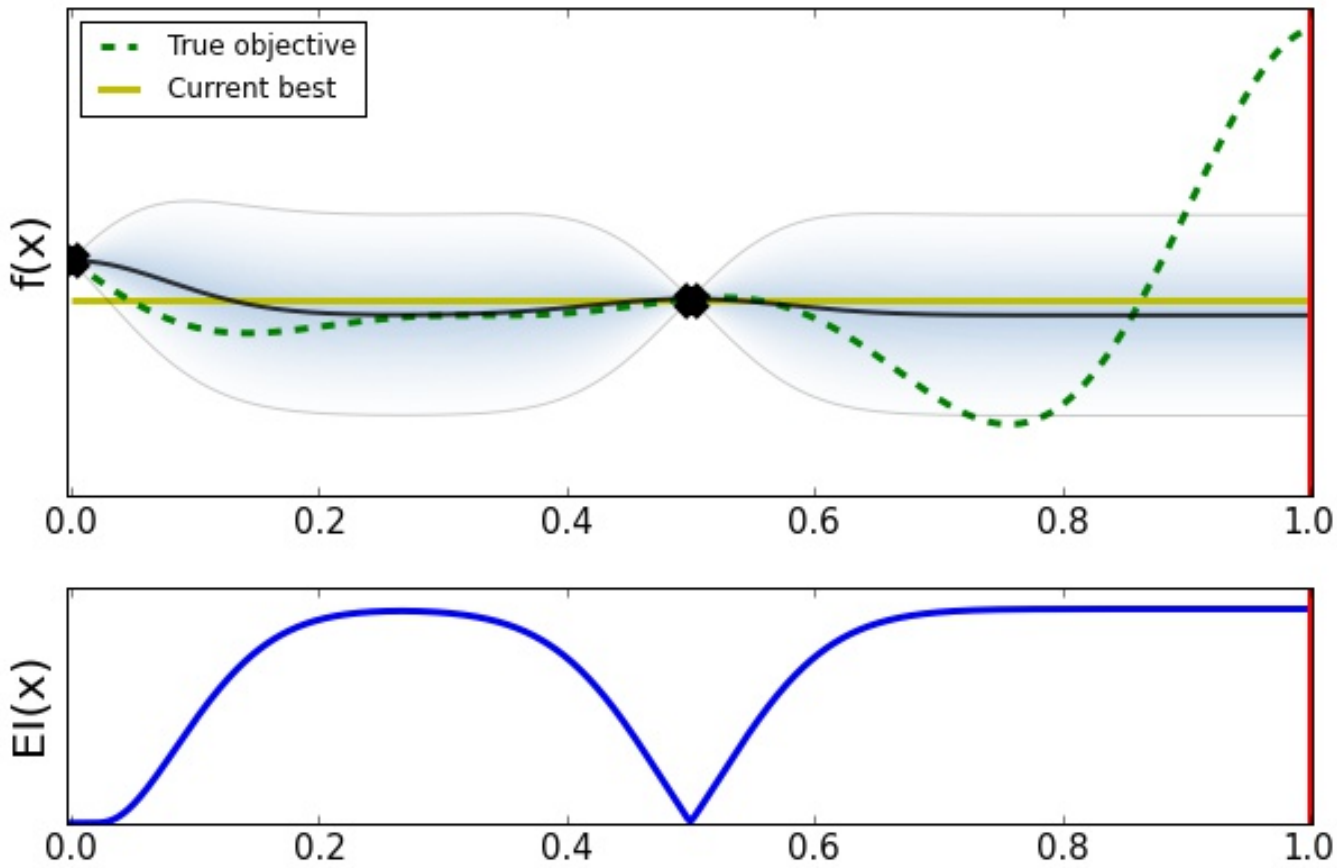


Illustration of BO

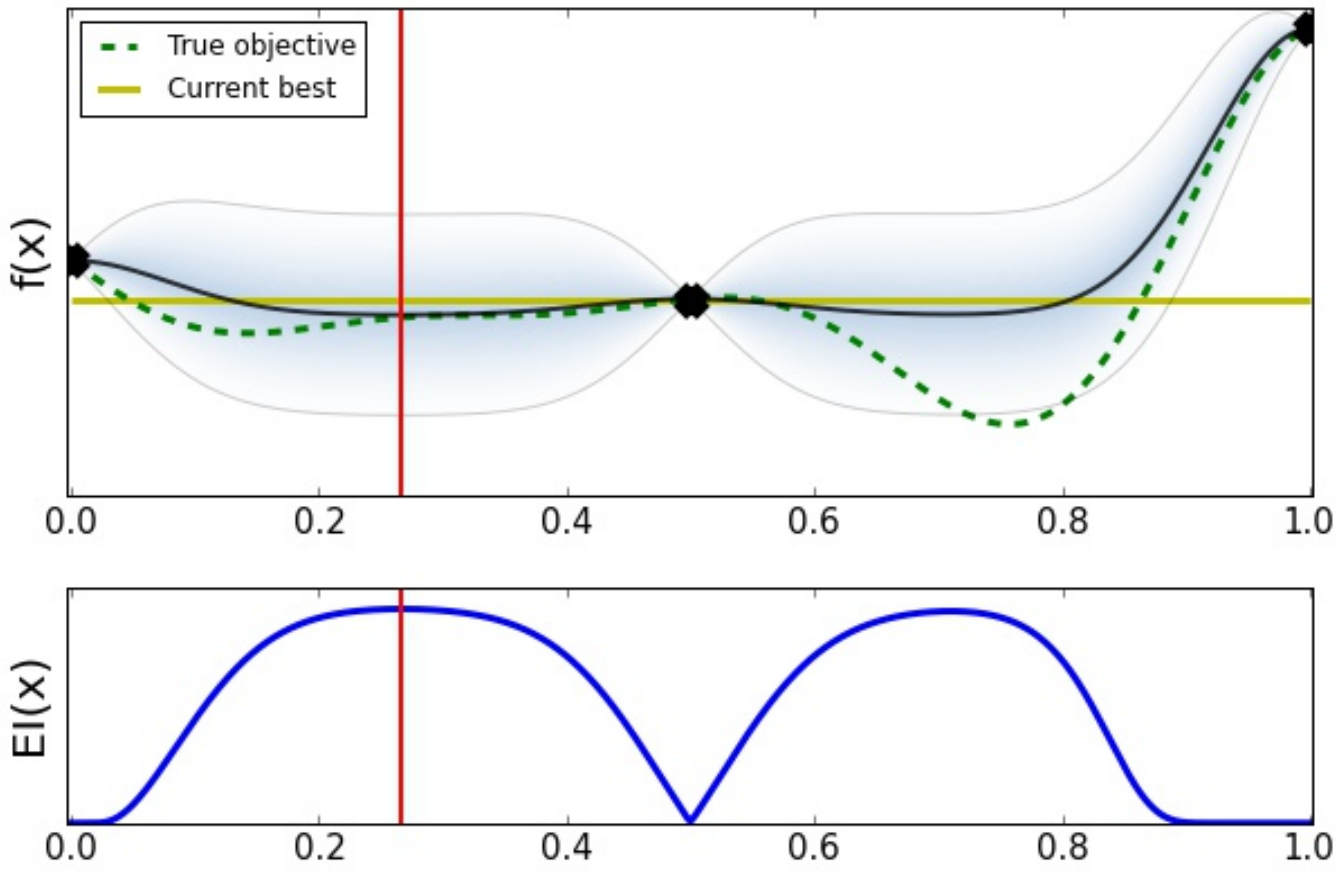


Illustration of BO

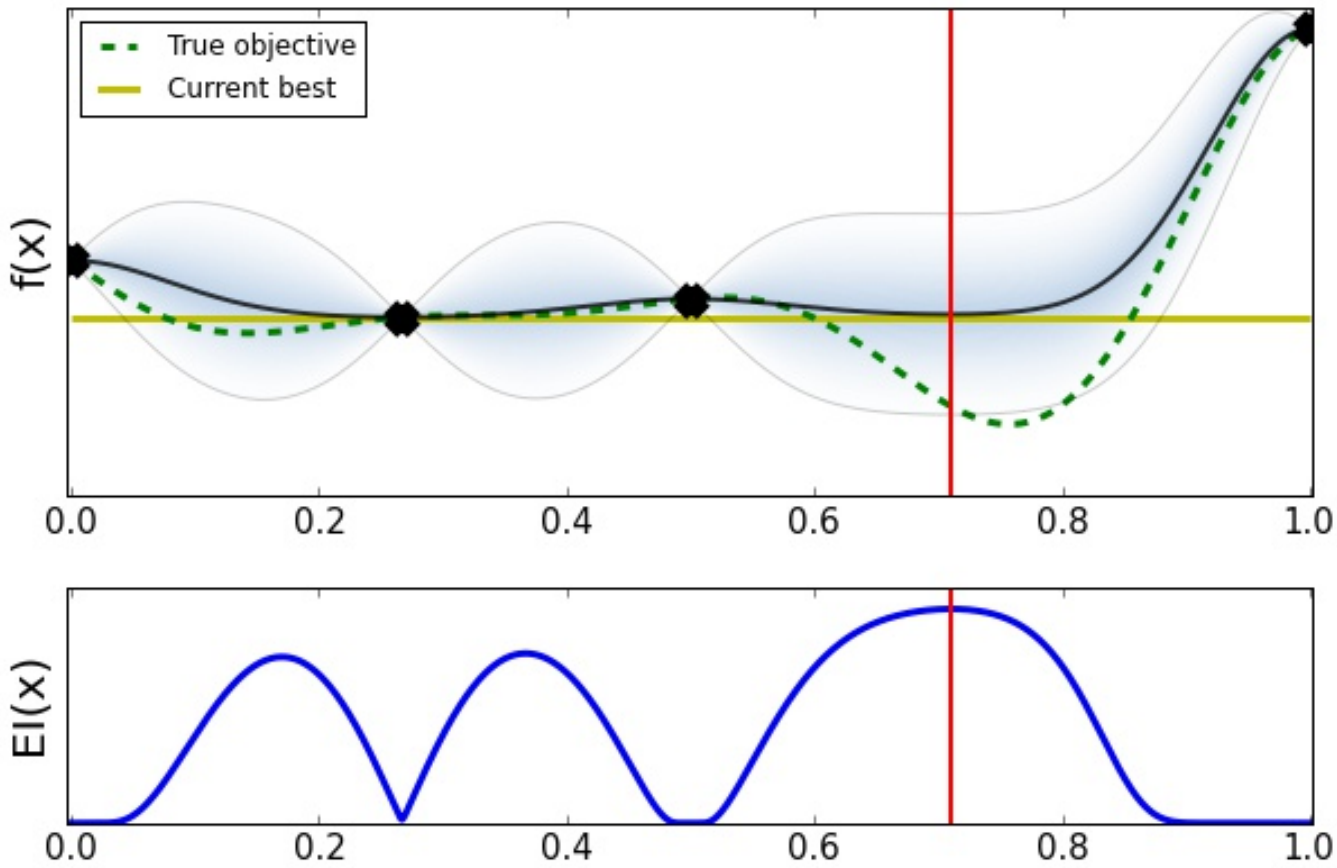


Illustration of BO

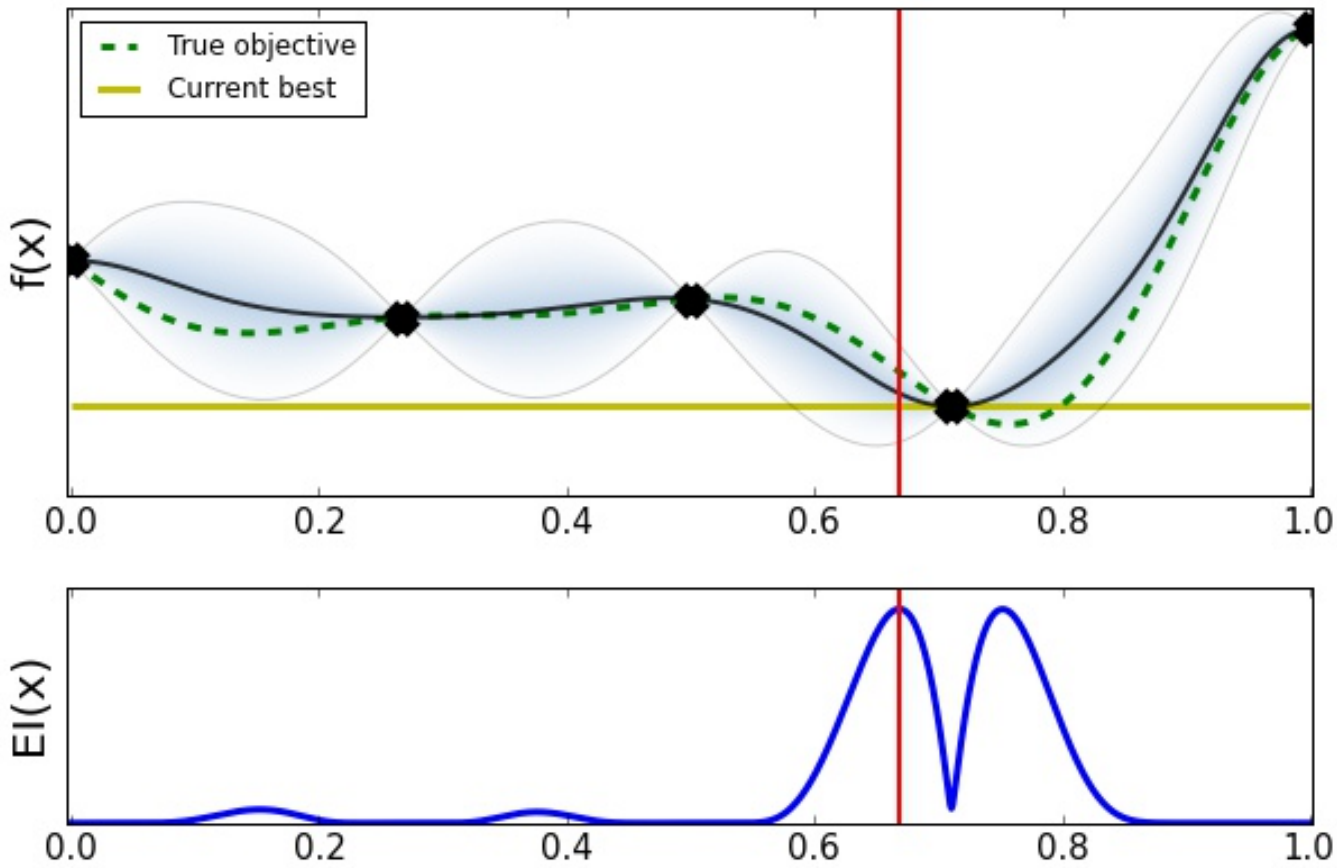


Illustration of BO

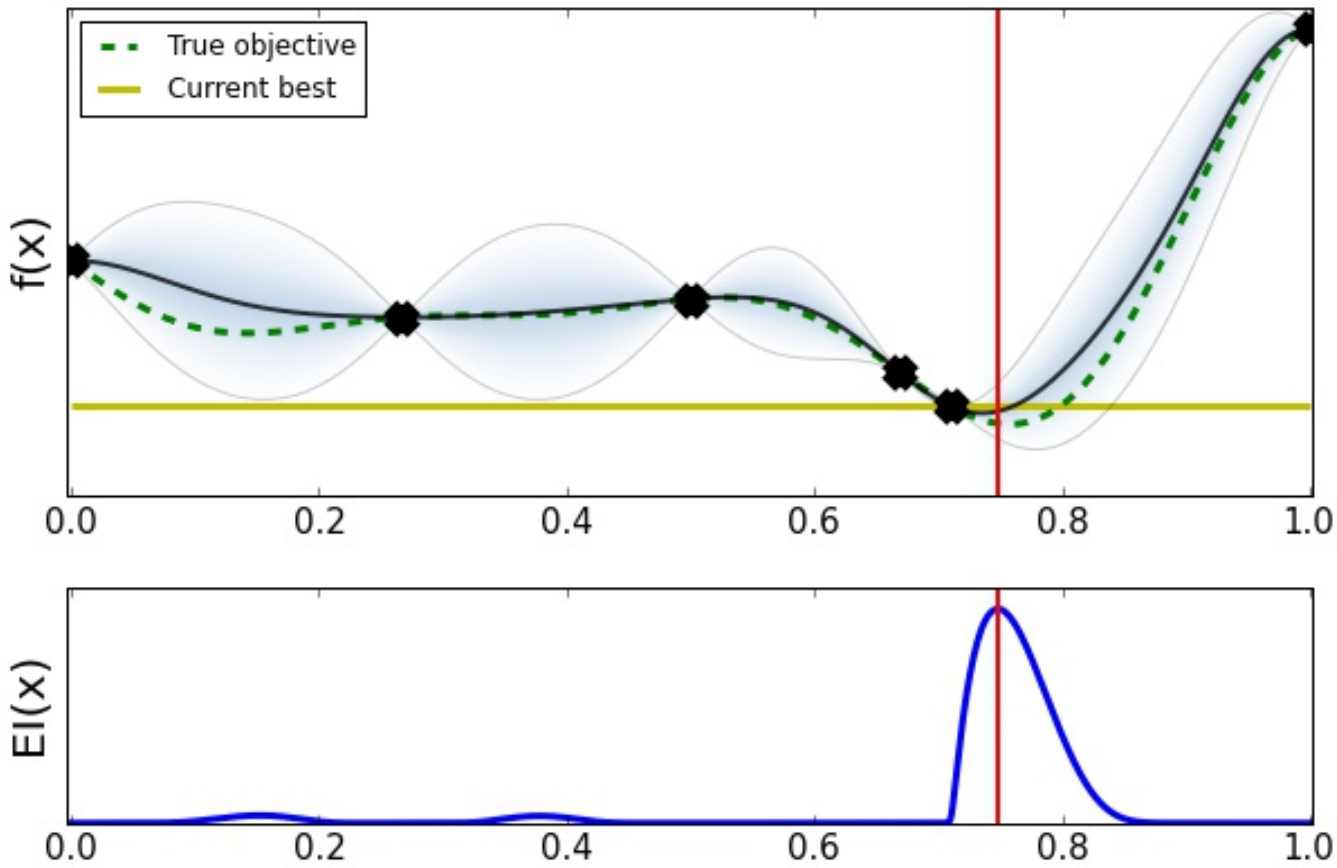


Illustration of BO

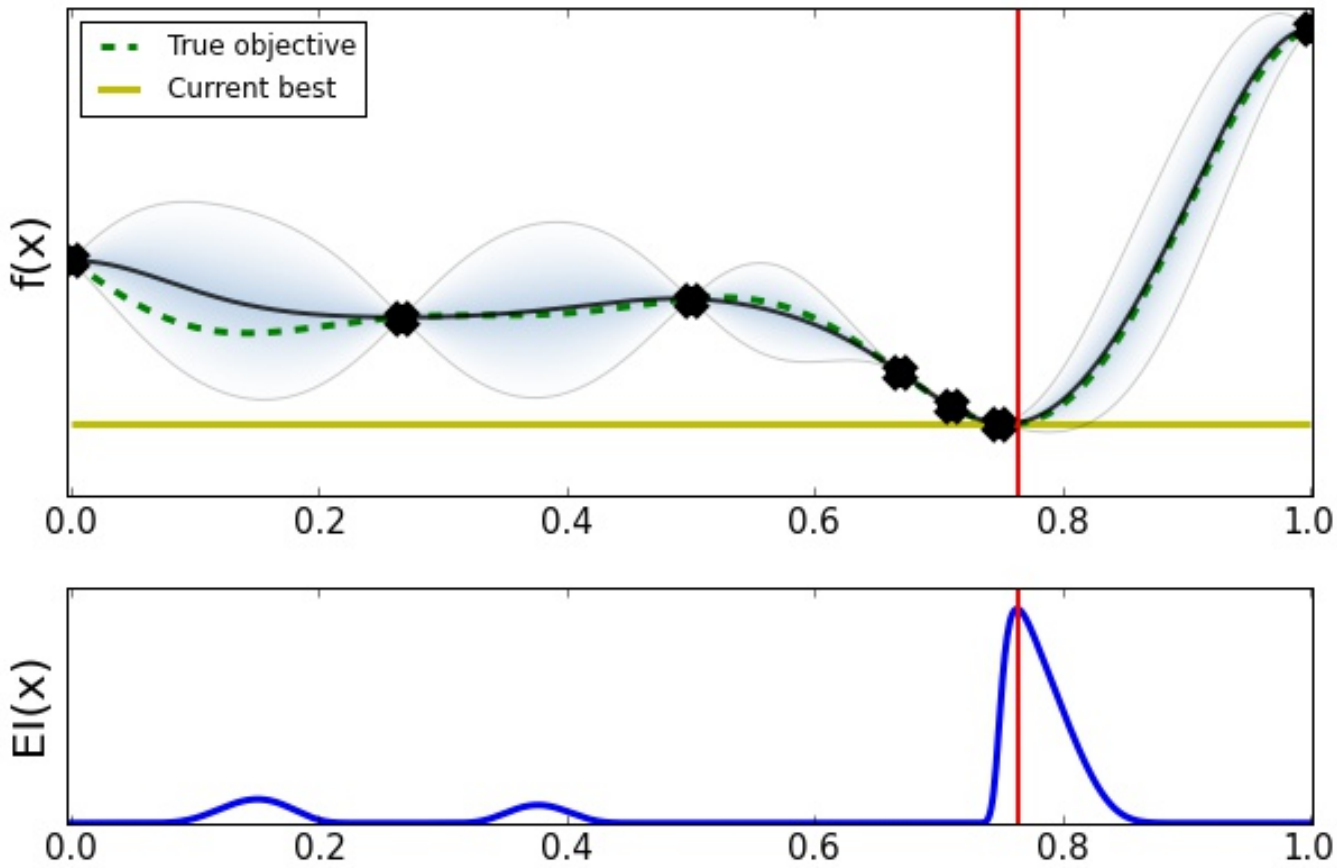
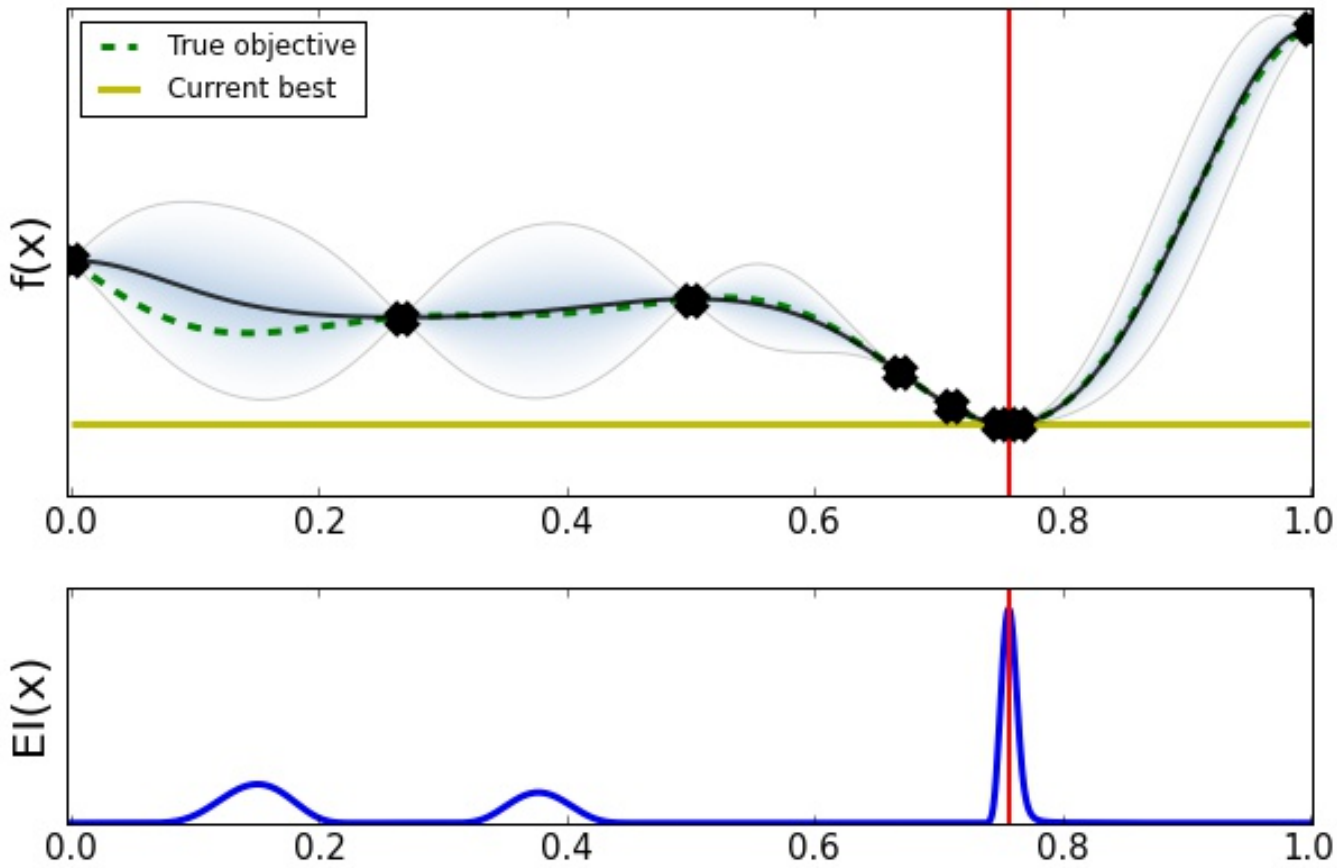


Illustration of BO



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:

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

solvable!

where now:

- $\alpha(x)$ is inexpensive to evaluate.
- The gradients of $\alpha(x)$ are typically available.
- Still need to find x_{n+1}.

BO vs other methods

[Osborne et al, 2009]

Bayesian optimization works better in practice!

				GPGO 1-Step		GPGO 2-Step
	EGO	RBF	DIRECT	Non-Periodic	Periodic	Non-Periodic
Br	0.943	0.960	0.958	**0.980**	—	—
C6	0.962	0.962	0.940	0.890	—	**0.967**
G–P	0.783	0.815	**0.989**	0.804	—	**0.989**
H3	0.970	0.867	0.868	**0.980**	—	—
H6	0.837	0.701	0.689	**0.999**	—	—
Sh5	0.218	0.092	0.090	**0.485**	—	—
Sh7	0.159	0.102	0.099	**0.650**	—	—
Sh10	0.135	0.100	0.100	**0.591**	—	—
GK2	0.571	0.567	0.538	**0.643**	—	—
GK3	0.519	0.207	0.368	**0.532**	—	—
Shu	**0.492**	0.383	0.396	0.437	0.348	0.348
G2	0.979	**1.000**	0.981	**1.000**	**1.000**	—
G5	**1.000**	0.998	0.908	0.925	0.957	—
A2	0.347	0.703	0.675	0.606	0.612	**0.781**
A5	0.192	**0.381**	0.295	0.089	0.161	—
R	0.652	0.647	0.776	0.675	**0.933**	—
mean	0.610	0.593	0.604	**0.705**	—	—

Recap

- Bayesian optimization is a way of encoding our beliefs about a property of a function (the minimum)
- Two key elements: the model and the acquisition function.
- Many choices in both cases, especially in terms of the acquisition function used.
- The key is to find a good balance between exploration and exploitation.

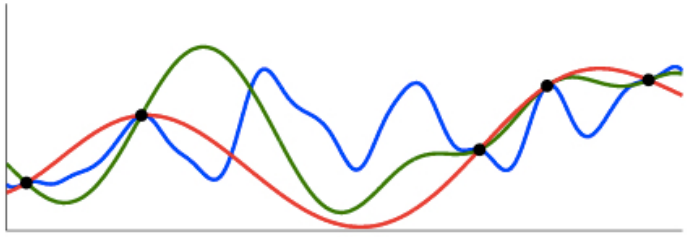
Main issues

- What to do with the hyper-parameters of the model?
- How to optimize the acquisition function?

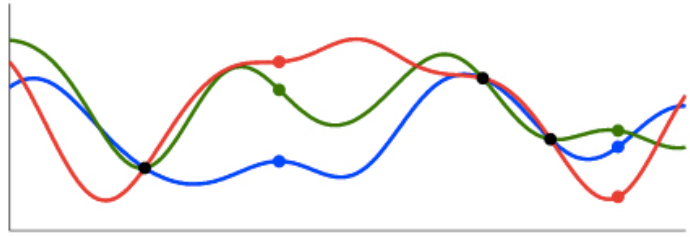
BO independent of the parameters of the GP.

[Snoek et al. 2012]

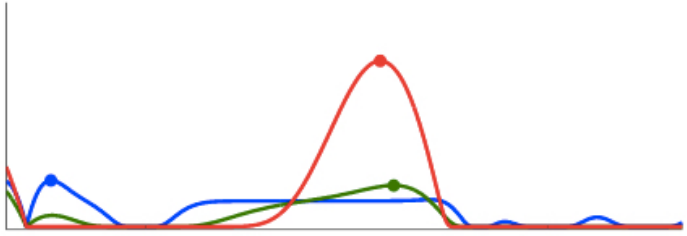
Integrate out across parameter values or location outputs.



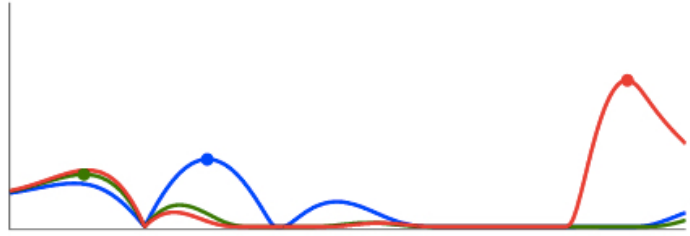
(a) Posterior samples under varying hyperparameters



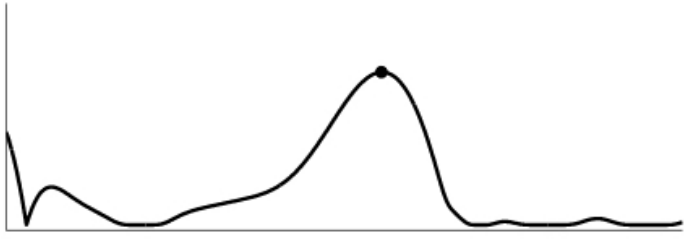
(a) Posterior samples after three data



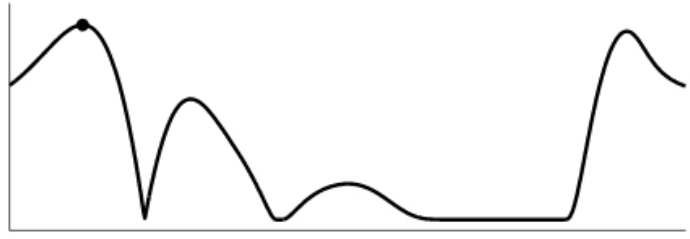
(b) Expected improvement under varying hyperparameters



(b) Expected improvement under three fantasies



(c) Integrated expected improvement



(c) Expected improvement across fantasies

Methods to optimise the acquisition function

This may not be easy.

- Gradient descent methods: Conjugate gradient, BFGS, etc.
- Lipschitz based heuristics: DIRECT.
- Evolutionary algorithms: CMA.

Some of these methods can also be used to directly optimize f.

Multi-task Bayesian Optimization

[Wersky et al., 2013]

Two types of problems:

1. Multiple, and conflicting objectives: design an engine more powerful but more efficient.

2. The objective is very expensive, but we have access to another cheaper and correlated one.

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.

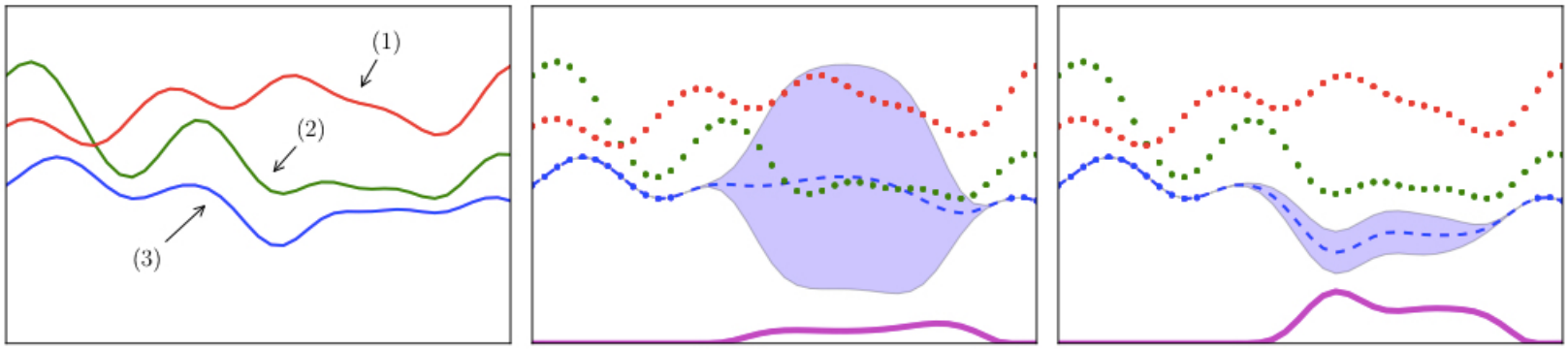
- The idea is to use the correlation among the function to improve the optimization.

Multi-output Gaussian process

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

Multi-task Bayesian Optimization

[Wersky et al., 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

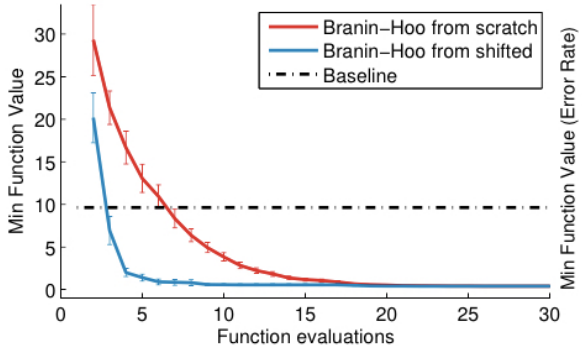
[Wersky et al., 2013]

- 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.

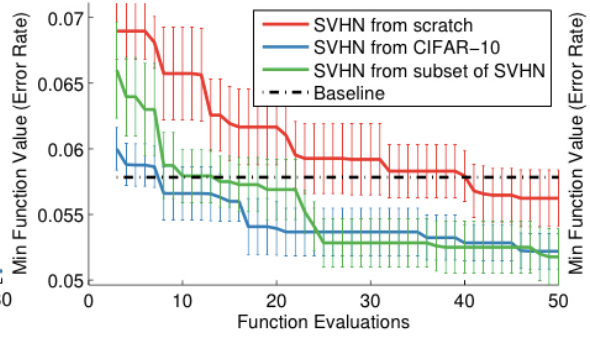
Averaged expected improvement.

Multi-task Bayesian Optimization

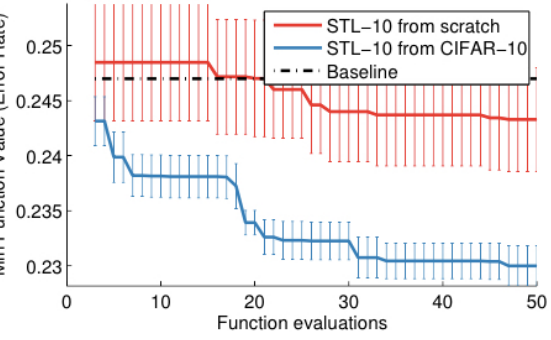
[Wersky et al., 2013]



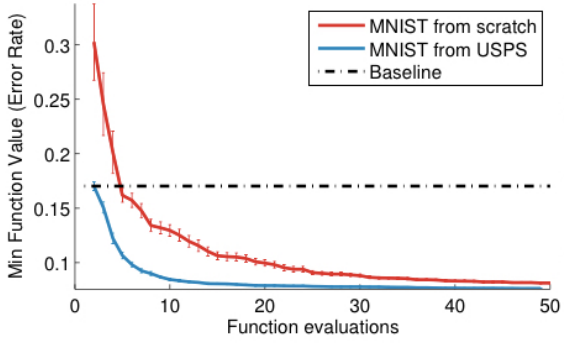
(a) Shifted Branin-Hoo



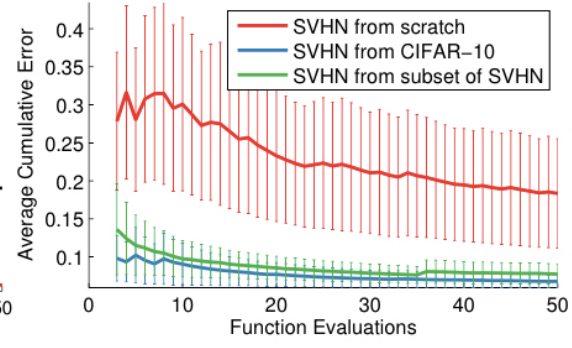
(b) CNN on SVHN



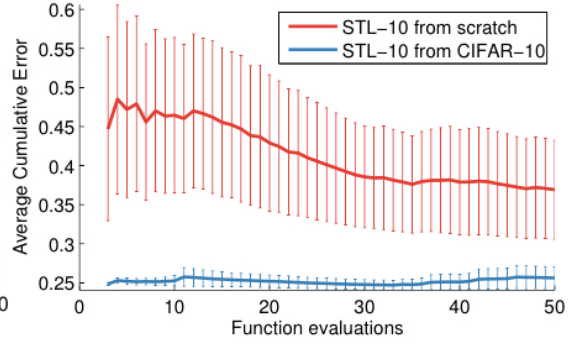
(c) CNN on STL-10



(d) LR on MNIST

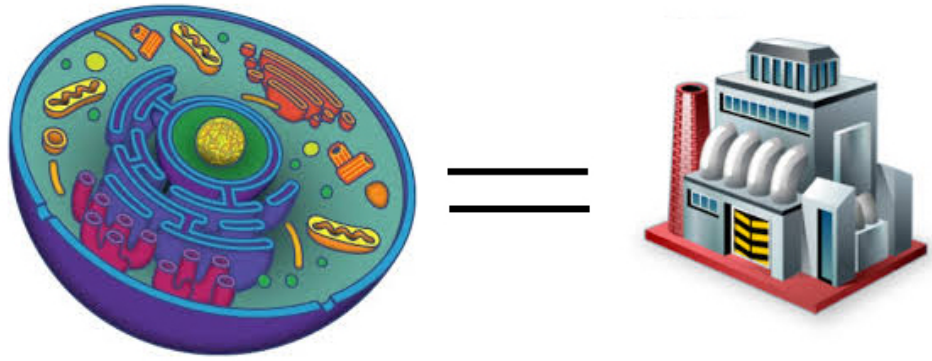


(e) SVHN ACE



(f) STL-10 ACE

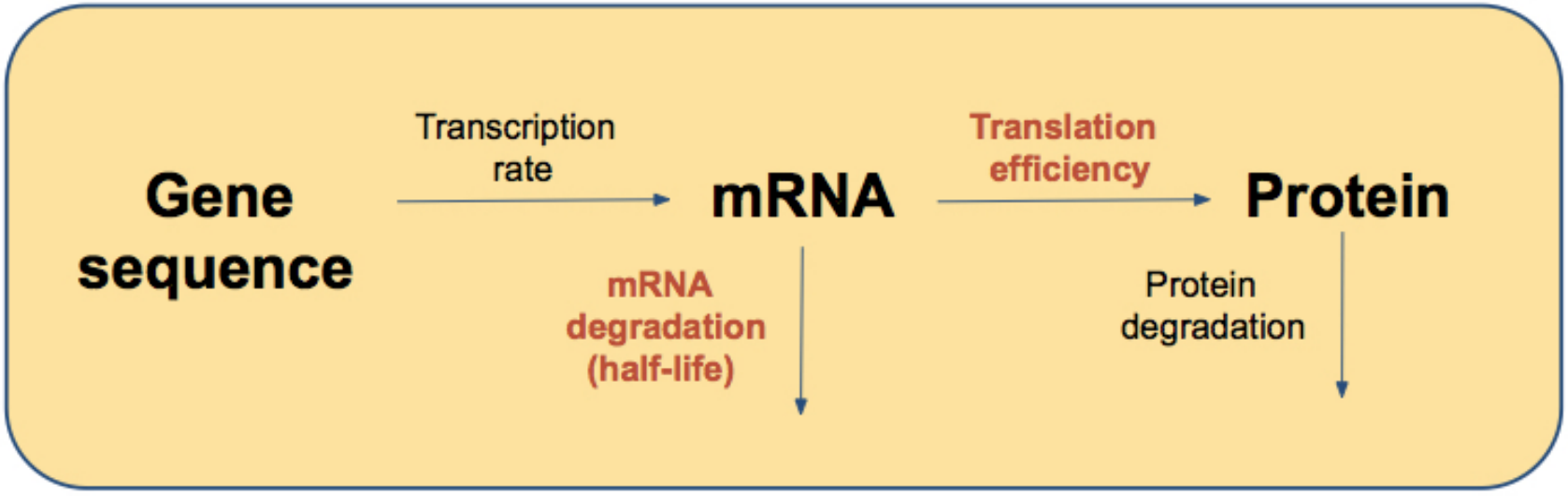
Synthetic gene design with Bayesian optimization



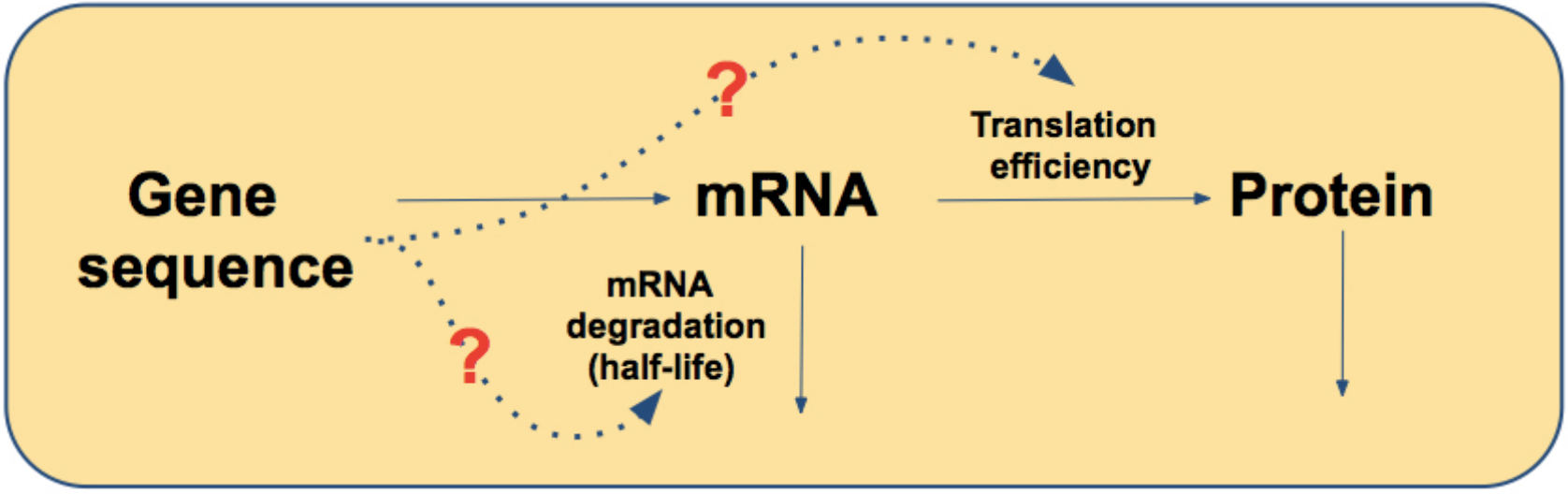
- Use mammalian cells to make protein products.
- Control the ability of the cell-factory to use synthetic DNA.

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

Central dogma of molecular biology



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 $\rightarrow$ same protein.
- The sequence affects the synthesis efficiency.

Which is the most efficient sequence to produce a protein?

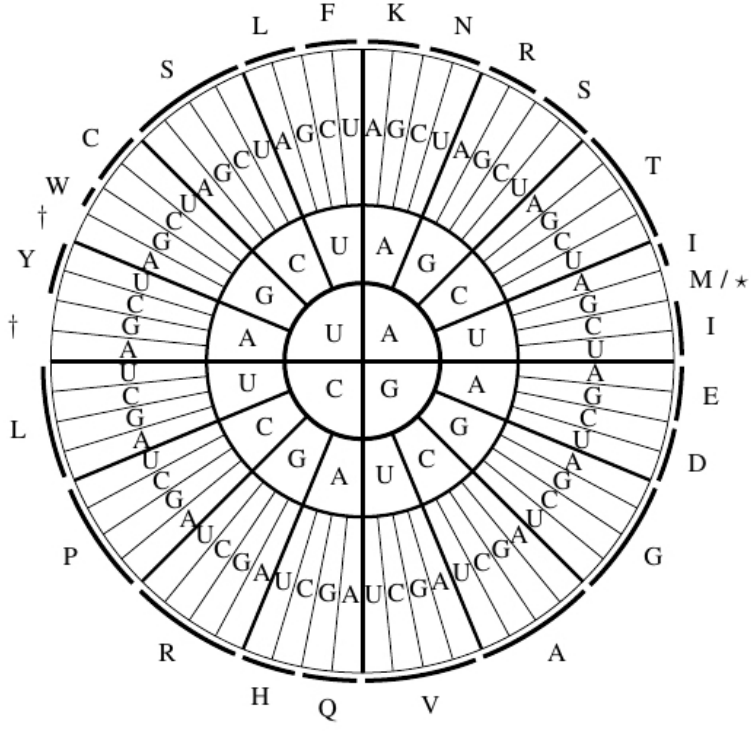
Redundancy of the genetic code

- 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

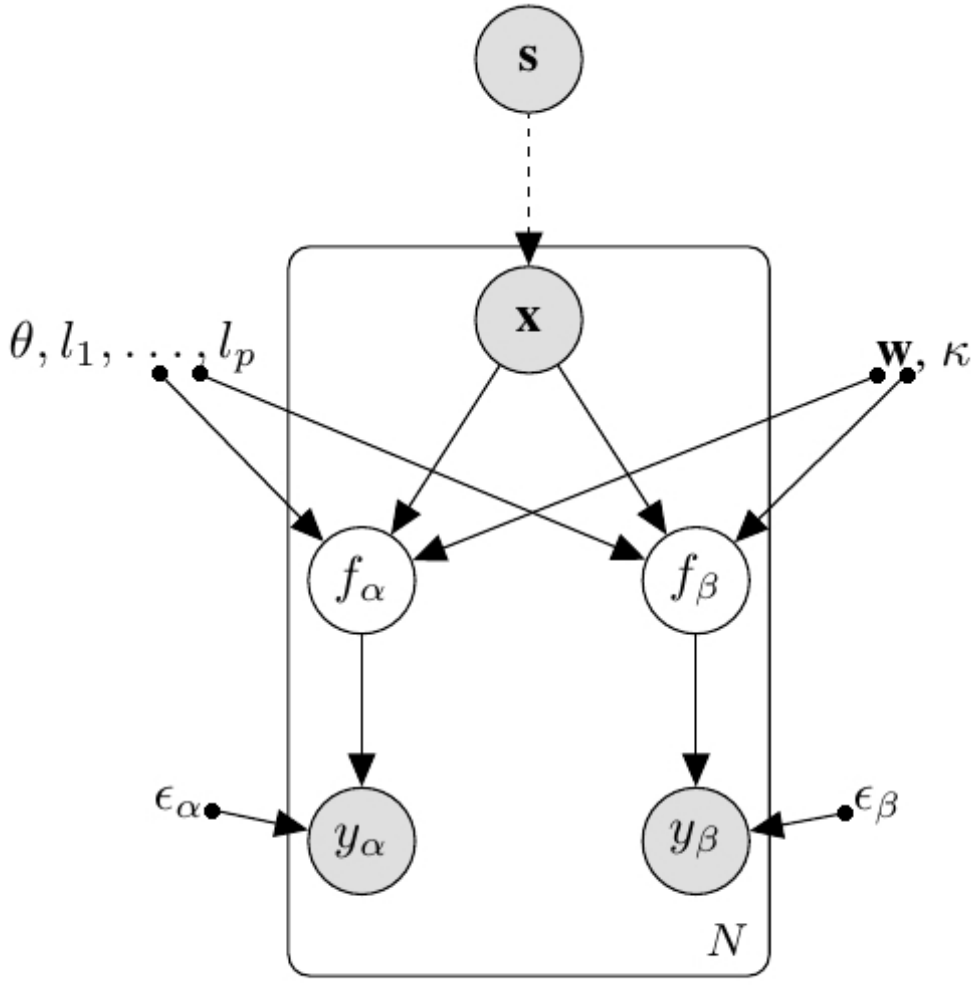
Model inputs
Features ($\mathbf{x}_i$) extracted gene sequences ($\mathbf{s}_i$): codon frequency, cai, gene length, folding energy, etc.

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

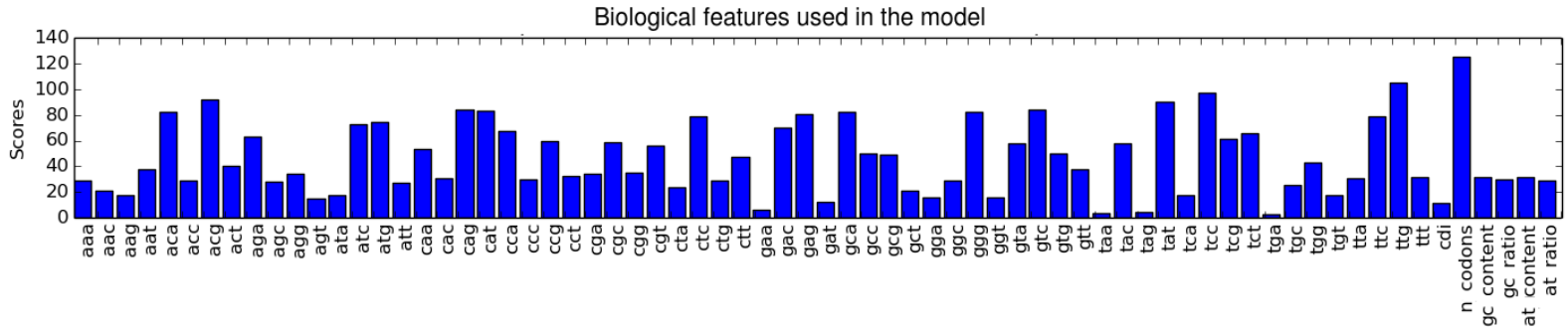
Simulating-matching approach:

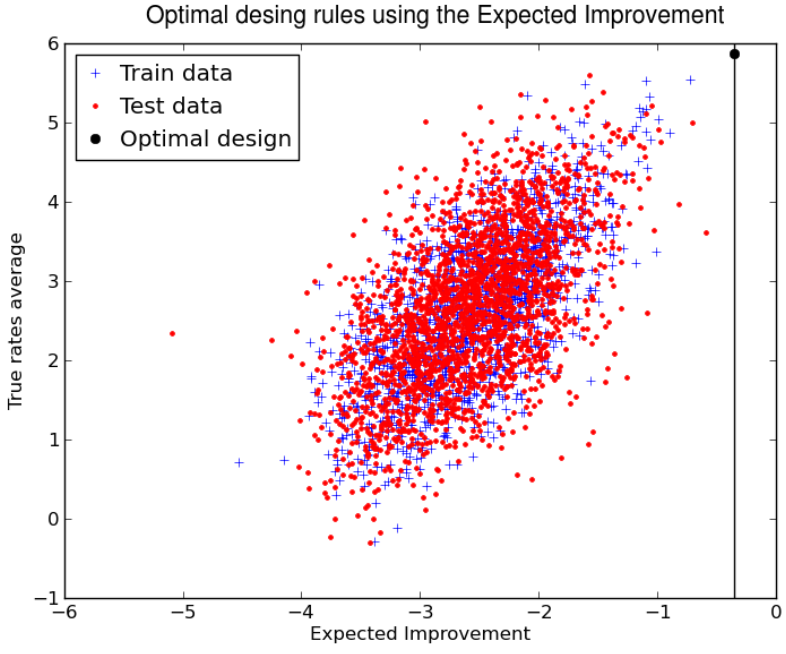
1. Simulate genes 'coherent' with the target (same amino-acids).
2. Extract features.
3. Rank synthetic genes according to their similarity with the 'optimal' design rules.

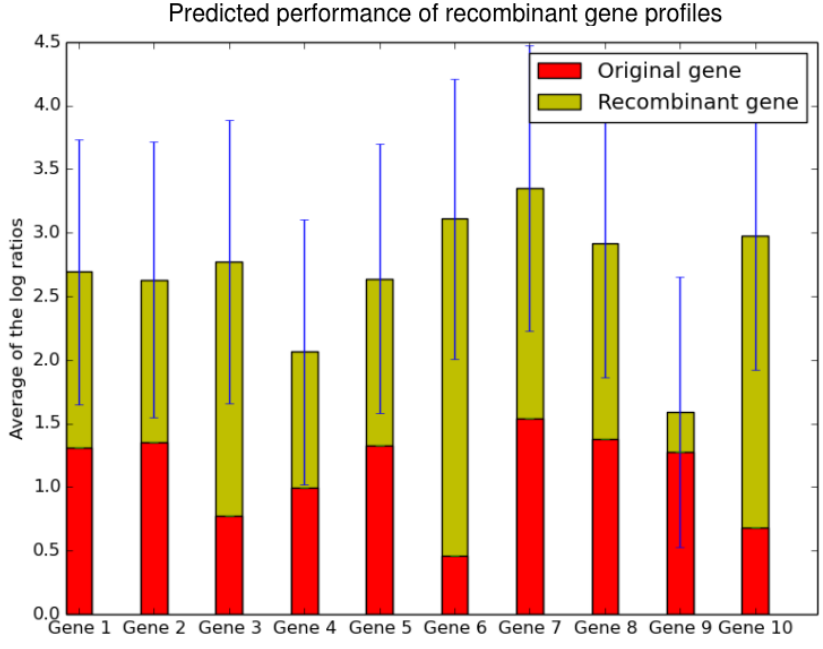
Ranking criterion: $eval(\mathbf{s}|\mathbf{x}^\star) = \sum_{j=1}^{p} w_j |\mathbf{x}_j - \mathbf{x}_j^\star|$

- $\mathbf{x}^\star$: 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


Biological features used in the model
Scores
0
20
40
60
80
100
120
140
aaa aac aag aat aca acc acg act aga agc agg agt ata atc atg att caa cac cag cat cca ccc ccg cct cga cgc cgg cgt cta ctc ctg ctt gaa gac gag gat gca gcc gcg gct gga ggc ggg ggt gta gtc gtg gtt taa tac tag tat tca tcc tcg tct tga tgc tgg tgt tta ttc ttg ttt cdi n codons gc content gc ratio at content at ratio


Optimal desing rules using the Expected Improvement
Train data
Test data
Optimal design
True rates average
Expected Improvement
−6
−5
−4
−3
−2
−1
0
−1
0
1
2
3
4
5
6


Predicted performance of recombinant gene profiles
Original gene
Recombinant gene
Average of the log ratios
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
Gene 1 Gene 2 Gene 3 Gene 4 Gene 5 Gene 6 Gene 7 Gene 8 Gene 9 Gene 10

Questions?

What happen if I am interested on the entire process and not just a property of it?

Experimental design

- **Model free**: Latin hypercubes, Sobol sequences, grids, etc.
- **Model based**: Collected points that maximize the information gain with respect to the model.

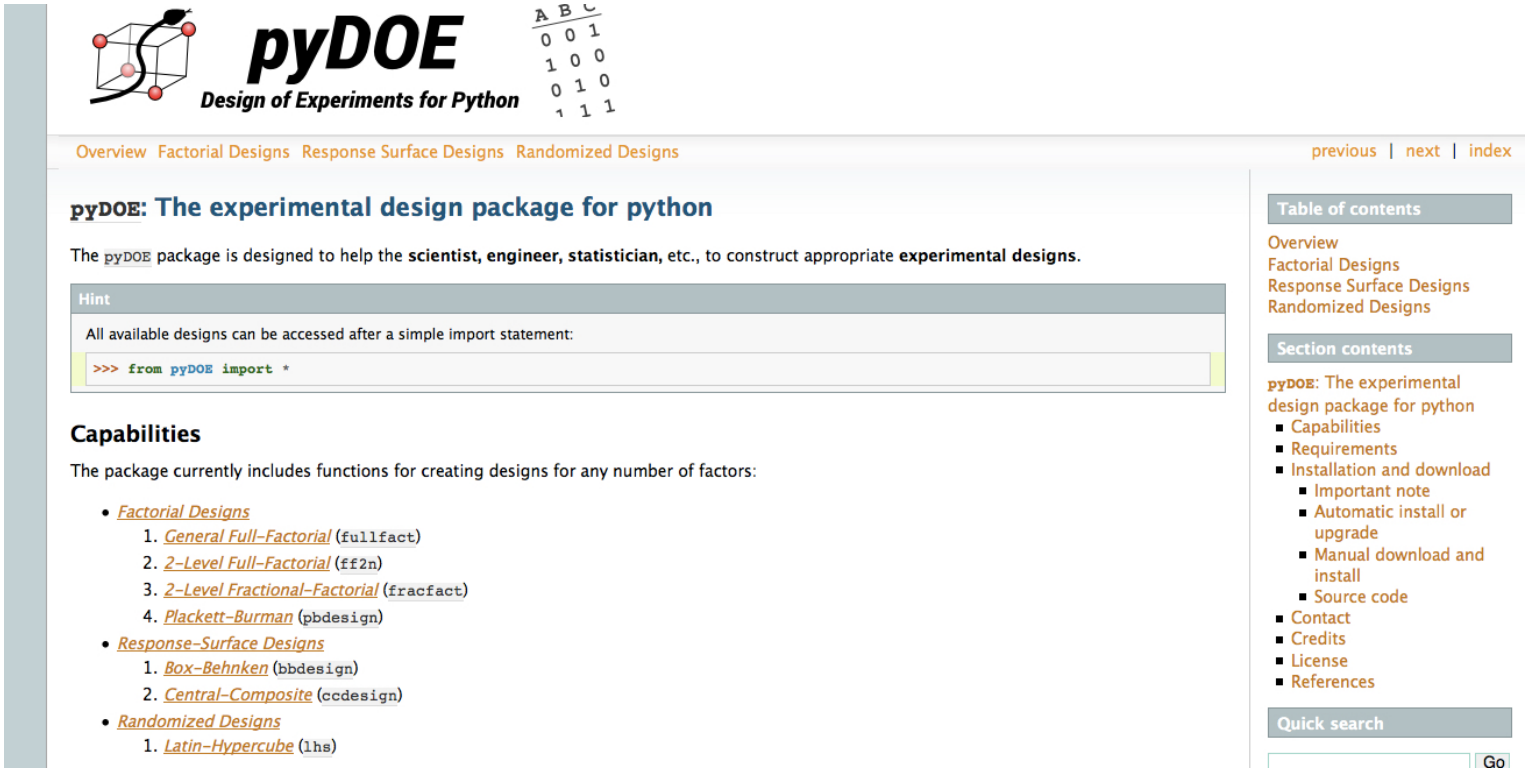
Latin design

$n \times n$ array filled with n different symbols, each occurring exactly once in each row and exactly once in each column.

A	B	F	C	E	D
B	C	A	D	F	E
C	D	B	E	A	F
D	E	C	F	B	A
E	F	D	A	C	B
F	A	E	B	D	C

pyDOE

Python framework for standard experimental design



pyDOE
Design of Experiments for Python

Overview Factorial Designs Response Surface Designs Randomized Designs previous | next | index

pyDOE: The experimental design package for python

The pyDOE package is designed to help the **scientist, engineer, statistician,** etc., to construct appropriate **experimental designs**.

Hint

All available designs can be accessed after a simple import statement:

```
>>> from pyDOE import *
```

Capabilities

The package currently includes functions for creating designs for any number of factors:

- *Factorial Designs*
  1. *General Full-Factorial* (`fullfact`)
  2. *2-Level Full-Factorial* (`ff2n`)
  3. *2-Level Fractional-Factorial* (`fracfact`)
  4. *Plackett-Burman* (`pbdesign`)
- *Response-Surface Designs*
  1. *Box-Behnken* (`bbdesign`)
  2. *Central-Composite* (`ccdesign`)
- *Randomized Designs*
  1. *Latin-Hypercube* (`lhs`)

Table of contents

Overview
Factorial Designs
Response Surface Designs
Randomized Designs

Section contents

pyDOE: The experimental design package for python
- Capabilities
- Requirements
- Installation and download
  - Important note
  - Automatic install or upgrade
  - Manual download and install
  - Source code
- Contact
- Credits
- License
- References

Quick search

Go

Halton sequences

[Halton, 1964]

- Used to generate points in $(0, 1) \times (0, 1)$
- Sequence that is constructed according to a deterministic method that uses a prime number as its base.

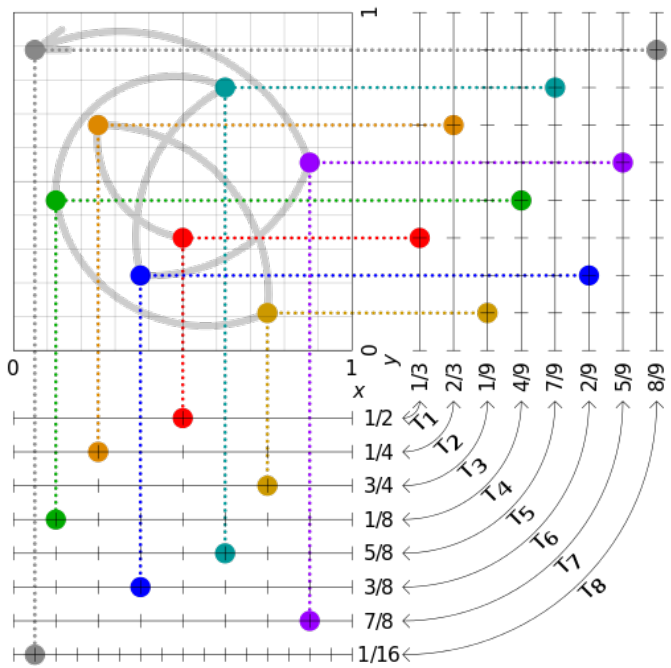
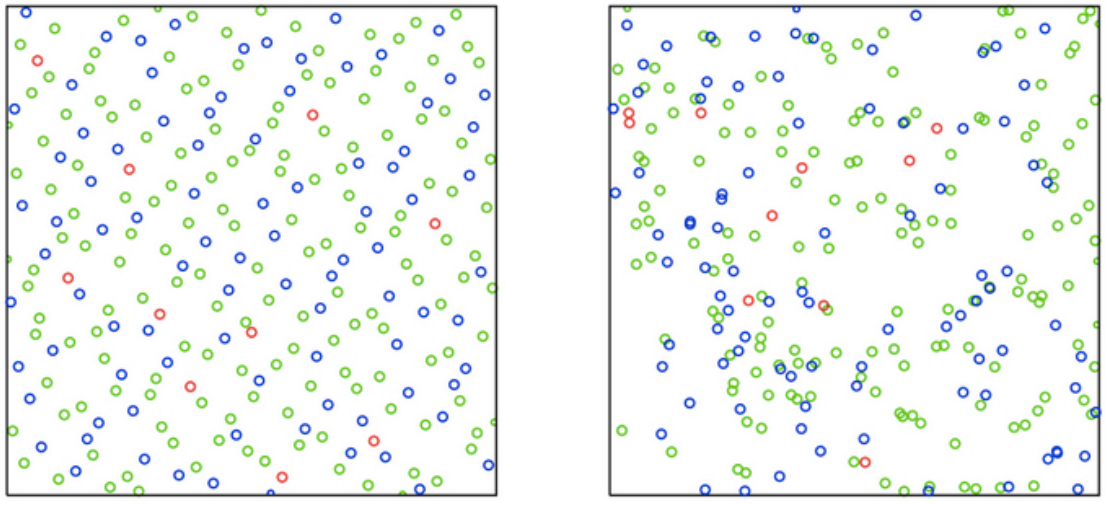


Figure source: Wikipedia

Halton sequences

[Halton, 1964]

Better coverage than random.



Halton *Random*

Figure source: Wikipedia

Model based experimental design

Given:

- A mapping function $y = f(x)$, expensive simulator for instance.
- A class of models to obtain $\hat{f}$ or $p(f)$.
- An algorithm to fit those models to inputs and outputs of f.

How to select $\{\mathbf{x}_i\}_{i=1}^n \in \mathcal{X}$ so we can guarantee that the model/approximation is 'good'?

Using a GP to design an experiment

Model to use:

$$y_i = f(\mathbf{x}_i) + \epsilon_i$$

- ϵ_i is zero-mean Gaussian with variance σ^2.
- $p(f)$ a GP with some covariance k.
- We are interested on modelling the behaviour of f in $\mathcal{X}$.

How to get a sample of points $\mathcal{S}$ so we can estimate the function globally well?

How can we learn about f as rapidly as possible?

[Srinivas et al., 2010]

Bayesian experimental design:

- Informativeness of a set of points $\mathcal{S} \in \mathcal{X}$ is measured by the information collected.
- Mutual information between f and $\mathbf{y}_\mathcal{S} = \mathbf{f}_\mathcal{S} + \epsilon_\mathcal{S}$.
- Can be computed as: $I(\mathbf{y}_\mathcal{S}; f) = \frac{1}{2} \log |\mathbf{I} + \sigma^{-2} \mathbf{K}_\mathcal{S}|$.

Issue when using the mutual information

Finding $\mathcal{S}$ using the MI is NP-hard

- Approximates greedy search. Collect points in $\mathcal{S}$ iteratively:

$$\mathbf{x}_t = \arg\max_{\mathcal{X}} I(\mathbf{y}_{\mathcal{S}_{t-1} \bigcup \{\mathbf{x}\}}; f).$$

- This is equivalent to collect

$$\mathbf{x}_t = \arg\max_{\mathcal{X}} \sigma_{t-1}(\mathbf{x}).$$

Issue when using the mutual information

Finding $\mathcal{S}$ using the MI is NP-hard

- Approximates greedy search. Collect points in $\mathcal{S}$ iteratively:

$$\mathbf{x}_t = \arg\max_{\mathcal{X}} I(\mathbf{y}_{\mathcal{S}_{t-1} \bigcup \{\mathbf{x}\}}; f).$$

- This is equivalent to collect

$$\mathbf{x}_t = \arg\max_{\mathcal{X}} \sigma_{t-1}(\mathbf{x}).$$

Other alternatives to experimental design

Integrated Variance, [Gorodetsky and Marzouk, 2016]

- Select the point that reduce the most the 'accumulated' variance in the entire domain $\mathcal{X}$.
- Equivalent to an expected integrated squared error of the posterior mean.

Select $\mathcal{S} = \{\mathbf{x}_1^*, \ldots, \mathbf{x}_n^*\}$ such that

$$\mathcal{S} = \arg\min_{\mathcal{X}} \int_{\mathcal{X}} \sigma(\mathbf{x}|\mathcal{S}) d\mathbf{x} \approx \frac{1}{N_{mc}} \sum_{i=1}^{N} \sigma(\mathbf{x}_i|\mathcal{S})$$

for N_{mc} is the number of Monte Carlo samples.

Other alternatives to experimental design

Integrated Variance, [Gorodetsky and Marzouk, 2016]

- Select the point that reduce the most the 'accumulated' variance in the entire domain $\mathcal{X}$.
- Equivalent to an expected integrated squared error of the posterior mean.

Select $\mathcal{S} = \{\mathbf{x}_1^*, \ldots, \mathbf{x}_n^*\}$ such that

$$\mathcal{S} = \arg\min_{\mathcal{X}} \int_{\mathcal{X}} \sigma(\mathbf{x}|\mathcal{S}) d\mathbf{x} \approx \frac{1}{N_{mc}} \sum_{i=1}^{N} \sigma(\mathbf{x}_i|\mathcal{S})$$

for N_{mc} is the number of Monte Carlo samples.

Review articles to go further

Bayesian Experimental Design: A Review

Kathryn Chaloner and Isabella Verdinelli

Statistical Science, Volume 10, Number 3 (1995), 273-304.

On a measure of information provided by an experiment

Lindley, D. V.

Annals of Mathematical Statistics, 27 (4): 9861005,1956