AI and the Uncertainty Challenge in Fundamental Physics
CNRS AISSAI and CNRS IN2P3

An introduction to Bayesian optimization

Emmanuel Vazquez
Laboratoire des Signaux et Sytèmes – Paris-Saclay

Nov. 27, 2023

Contents

1 Computer experiments and optimization

2 Optimization of expensive-to-evaluate functions

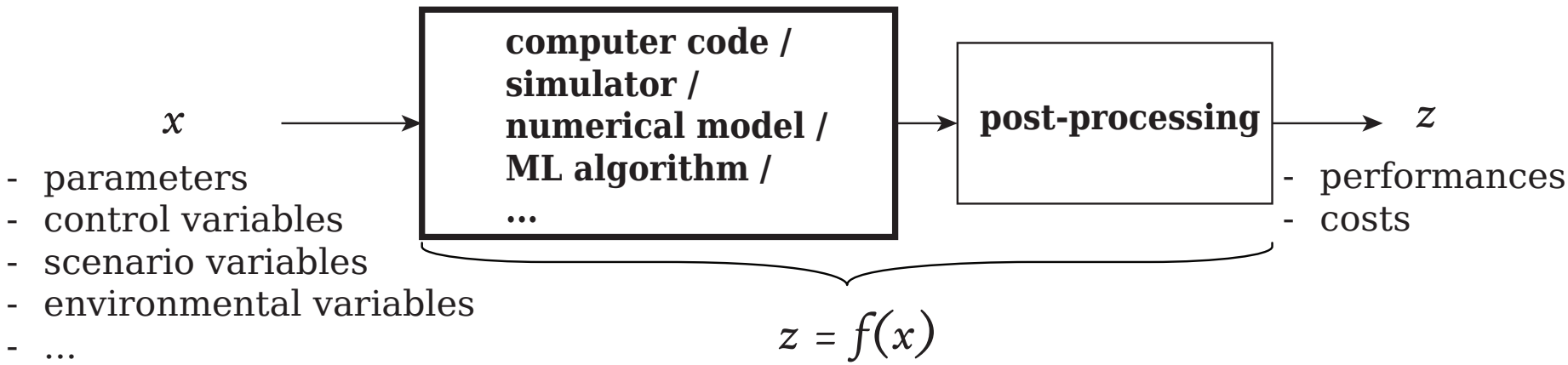
3 Bayesian black-box modeling

4 Bayesian optimization

5 Concluding remarks

1 Computer experiments and optimization

- Consider a computer procedure with inputs and outputs



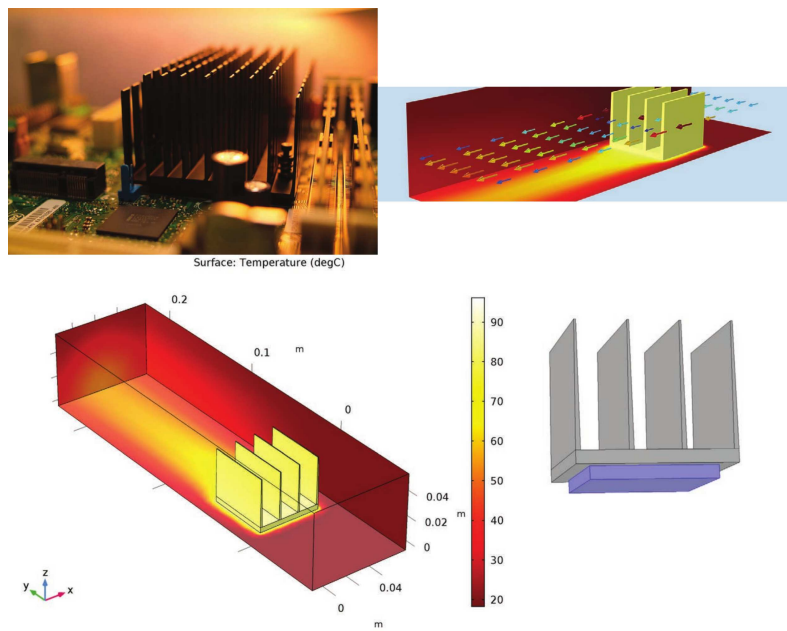
- Selecting a value for $x \in \mathbb{X}$ and observing the resulting output $z \in \mathbb{Z}$ is a computer experiment.

- In many cases, we aim to minimize or maximize the value of z.

Example (1/2): optimization of a system

Find the best values for the design parameters of a system

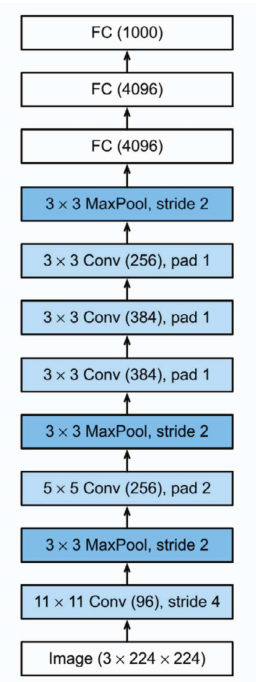
- Heat sink $\rightarrow$ shape controls airflow characteristics and radiation, which have direct impact on cooling performances
- $f : \mathbb{X} \subset \mathbb{R}^d \rightarrow \mathbb{R}$ performance as a function of design parameters.
- Computing $f(x)$: time-consuming! (solve PDEs using a finite element method)
- Optimization must be done within a limited budget of simulations



Simulation of a heat sink in COMSOL Multiphysics

Example (2/2): optimization of a DNN

- $f : \mathbb{X} \to \mathbb{R} \rightsquigarrow$ validation loss of a DNN as a function of its parameters (size of layers, dropout...).
- Computing $f(x)$ on large datasets is resource- and time-consuming




```
self.net = nn.Sequential(
    nn.LazyConv2d(96, kernel_size=11, stride=4, padding=1),
    nn.ReLU(),
    nn.MaxPool2d(kernel_size=3, stride=2),
    nn.LazyConv2d(256, kernel_size=5, padding=2),
    nn.ReLU(), nn.MaxPool2d(kernel_size=3, stride=2),
    nn.LazyConv2d(384, kernel_size=3, padding=1),
    nn.ReLU(),
    nn.LazyConv2d(384, kernel_size=3, padding=1),
    nn.ReLU(),
    nn.LazyConv2d(256, kernel_size=3, padding=1),
    nn.ReLU(),
    nn.MaxPool2d(kernel_size=3, stride=2), nn.Flatten(),
    nn.LazyLinear(4096), nn.ReLU(),
    nn.Dropout(p=0.5),
    nn.LazyLinear(4096), nn.ReLU(),
    nn.Dropout(p=0.5),
    nn.LazyLinear(num_classes))
```

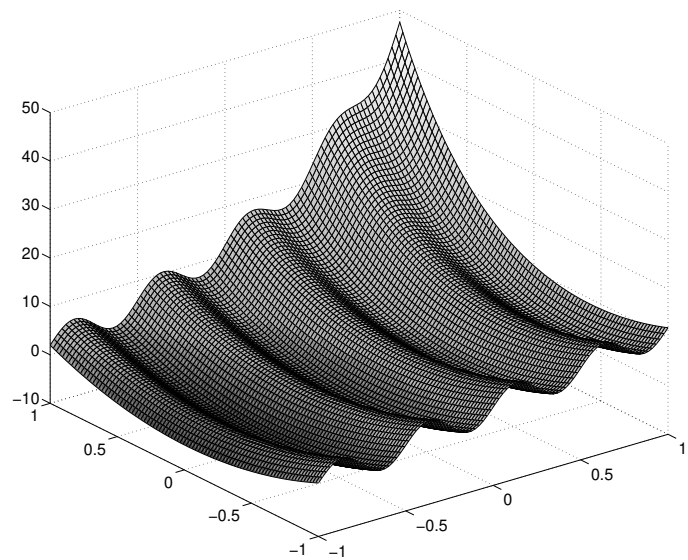
Old(!) AlexNet (Krizhevsky et al., 2012) is a 8-layer CNN. 20–30 parameters
Won the ImageNet Large Scale Visual Recognition Challenge 2012.

2 Local and grid search optimization for expensive-to-evaluate functions

Illustrative example

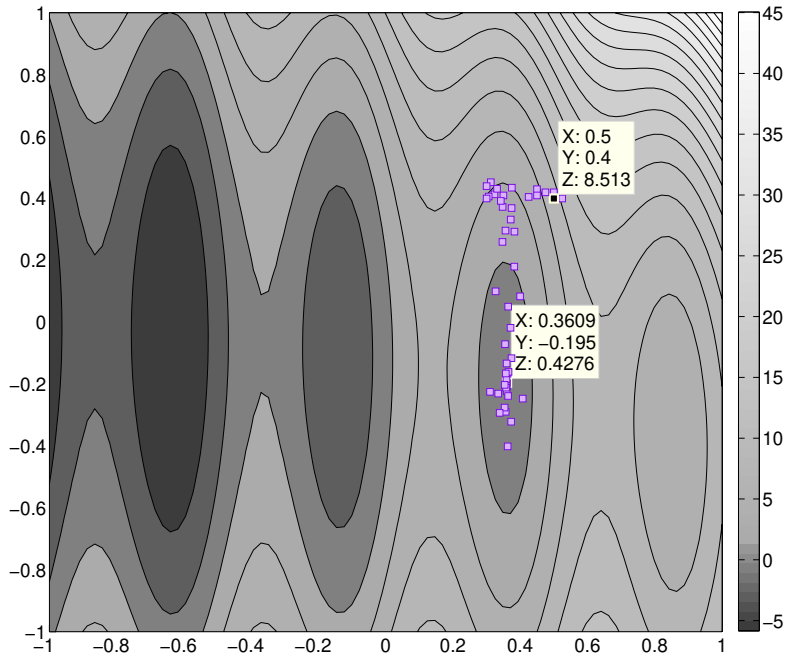
- Consider

$$\begin{array}{rccl} f: & \mathbb{R}^2 & \to & \mathbb{R} \\ & x & \mapsto & f(x) = \exp\left(1.8\left(x_{[1]} + x_{[2]}\right)\right) + 5x_{[1]} + 6{x_{[2]}}^2 + 3\sin\left(4\pi x_{[1]}\right) \end{array}$$



- Objective: find an approximation of $x^\star = \arg\min_{x\in[-1,1]^2} f(x)$ with a budget of $N = 60$ experiments

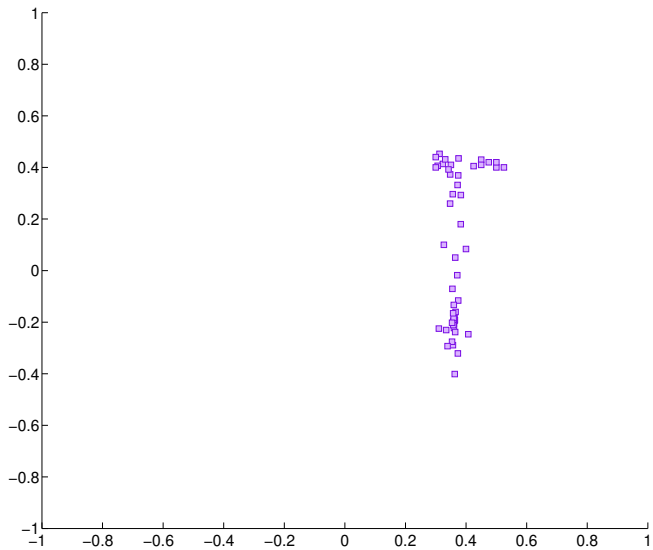
Evaluations points using a Nelder-Mead algorithm



$\rightarrow$ the algorithm converges to a local minimum (≈ 0.427, global minimum is ≈ -5.845))

This comes as no surprise (local search algorithm). But above all...

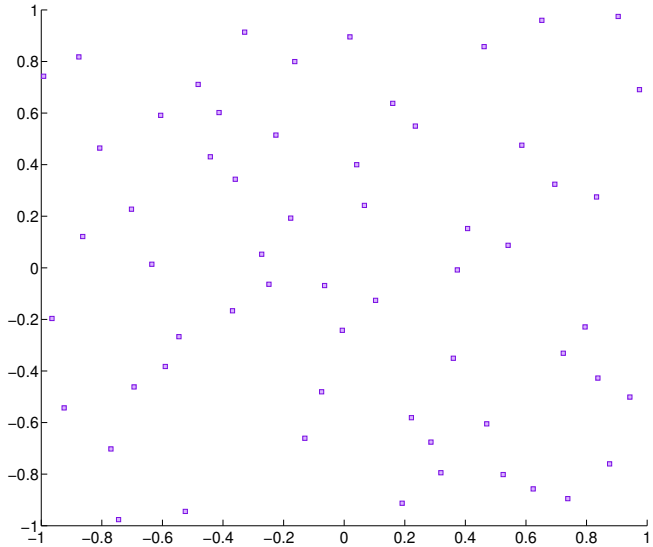
$\rightsquigarrow$ after having spent the budget of (possibly expensive) evaluations, the behavior of the function is only known in a small region of the search domain



$\rightsquigarrow$ the global behavior of the function is unknown

$\rightsquigarrow$ potentially interesting regions have not been explored

How about sampling f uniformly on the search domain?



$\rightarrow$ minimum of evaluation results is ≈ -5.823 (global minimum ≈ -5.845)

- Uniformly sampling tends to minimize the fill-distance $h_N = \max_{x \in \mathbb{X}} \min_i \|x - x_i\|$
- From a theoretical perspective, minimizing h_N on a search domain is minimax optimal for approximating the function's optimum.

- However, random search or grid search strategies are generally inefficient because they do not focus on the more promising regions of the search space
- In situations where functions are expensive to evaluate and there is a limited budget for evaluations $\rightarrow$ strike a balance between local search and exploration of the search domain.

  $\rightsquigarrow$ exploration vs exploitation trade-off

3 Bayesian black-box modeling

- Let $f : \mathbb{X} \to \mathbb{R}$ be a real function defined on $\mathbb{X} \subseteq \mathbb{R}^d$, where $\mathbb{X}$ is the input/parameter domain of the computer code under study
- f is a black-box, only known through evaluation/simulation results: query an evaluation at x, observe the result
- Given n simulations points $x_1, \ldots, x_n \in \mathbb{X}$, denote by

$$z_1 = f(x_1), \ldots, z_n = f(x_n)$$

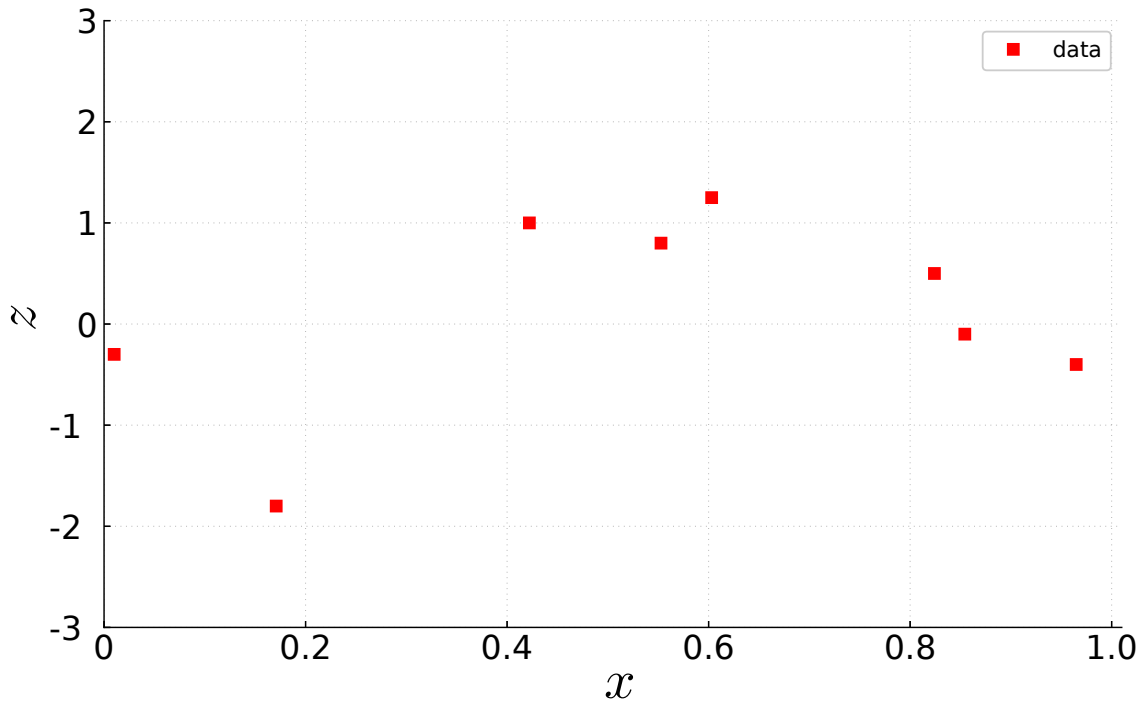
the corresp. simulation results (observations/evaluations of f)

- **Our objective**: use the data $\mathcal{D}_n = (x_i, z_i)_{i=1\ldots n}$ to infer properties about f
- Examples:
  - given a new $x \in \mathbb{R}^d$, predict the value $f(x)$,
  - predict $x^\star = \arg\max_x f(x)$, or $M = \max_x f(x)$
- Predict the value of f at a given x?

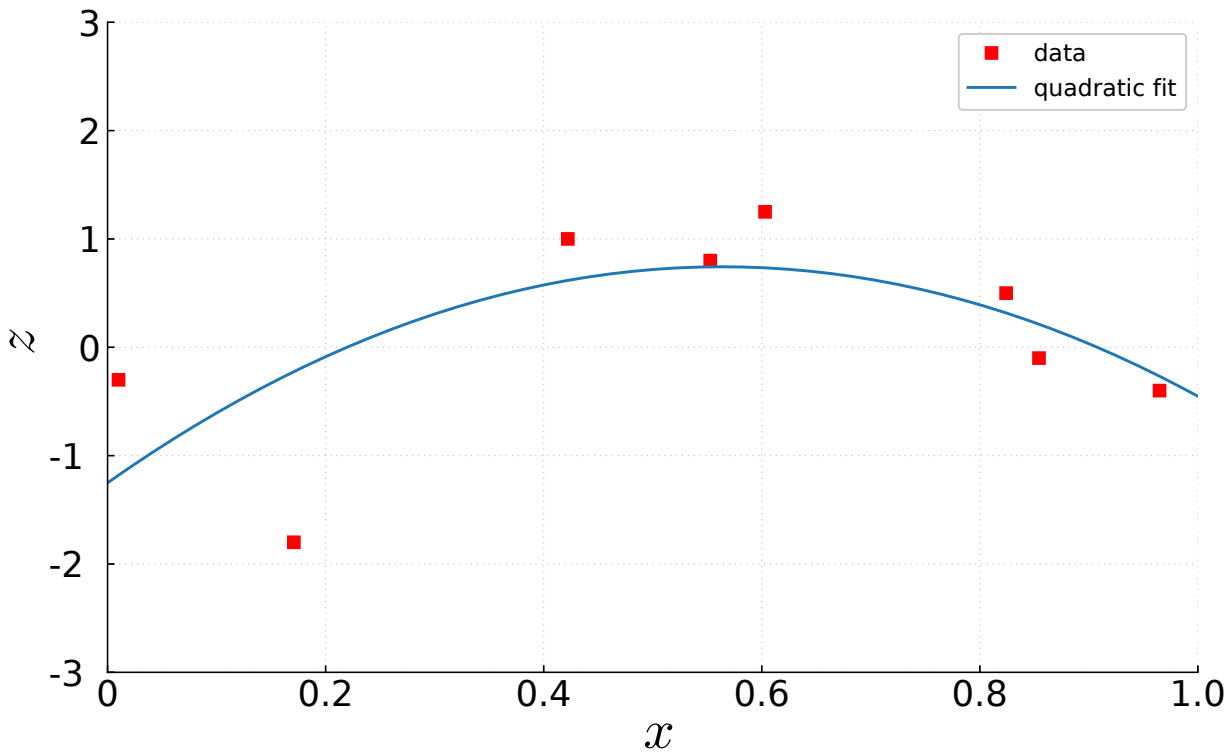
$\rightarrow$ the problem is that of constructing an approximation / an estimator $\widehat{f}_n$ of f from $\mathcal{D}_n$

A “curve fitting” problem

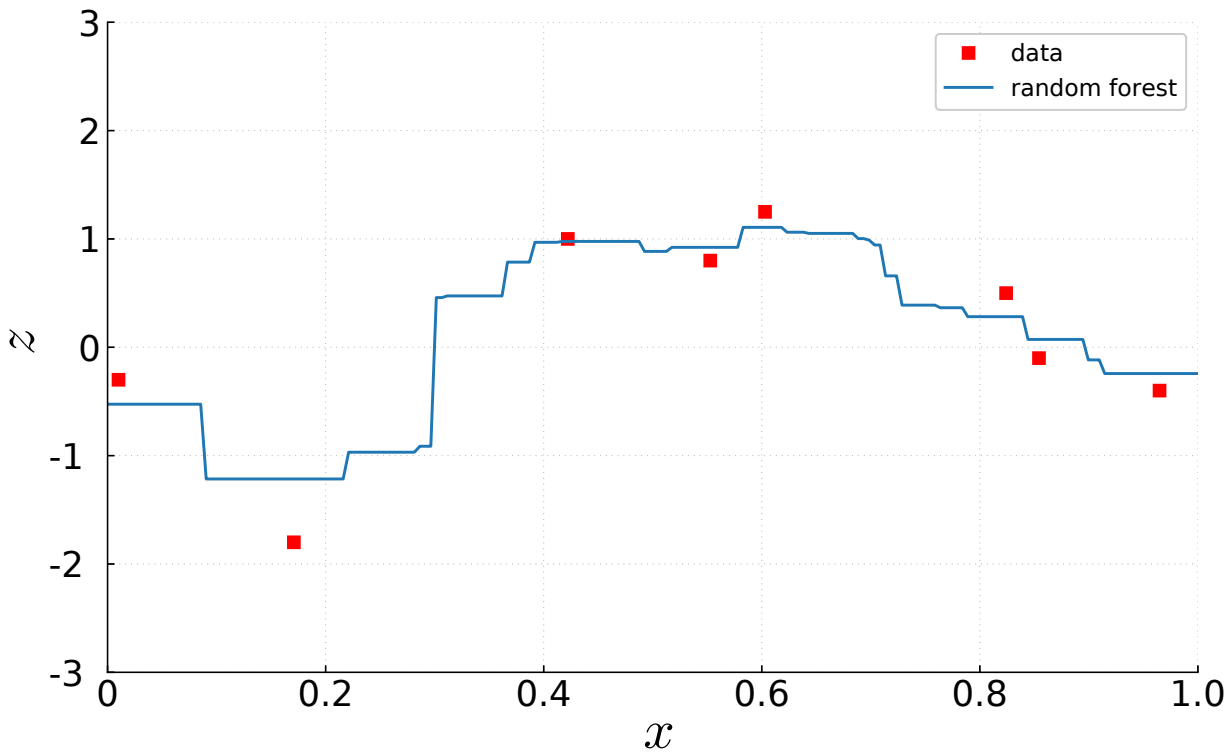
- We are given a data set of n simulation results, i.e., evaluations results of an unknown function $f : [0, 1] \to \mathbb{R}$, at points $x_1, \ldots, x_n$.
- An example with $n = 8$:



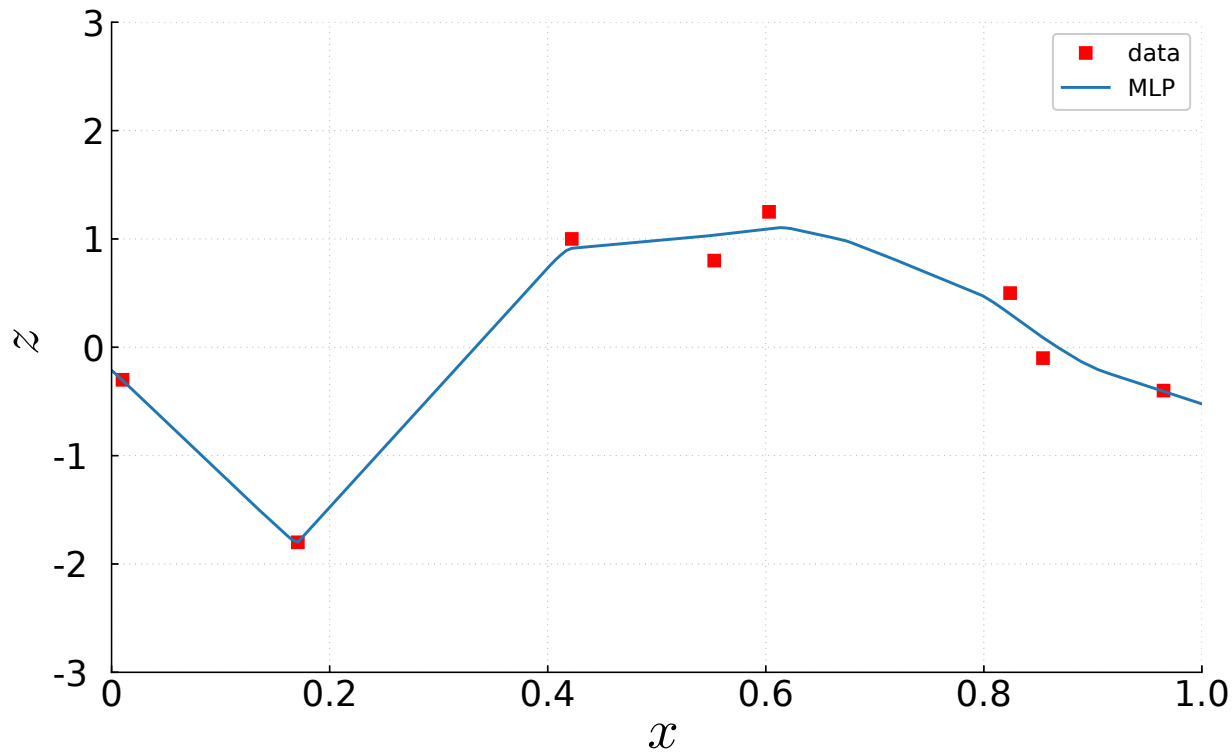
- Any approximation procedure of f consists in building a function $\widehat{f}_n = h(\cdot\,;\,\theta)$, where θ is a vector of parameters, to be chosen using $\mathcal{D}_n$ and available prior information
- e.g., quadratic fit using least-squares regression (try it in colab)



- Any approximation procedure of f consists in building a function $\widehat{f}_n = h(\cdot\,;\,\theta)$, where θ is a vector of parameters, to be chosen using $\mathcal{D}_n$ and available prior information
- e.g., random forest regression (try it in colab)



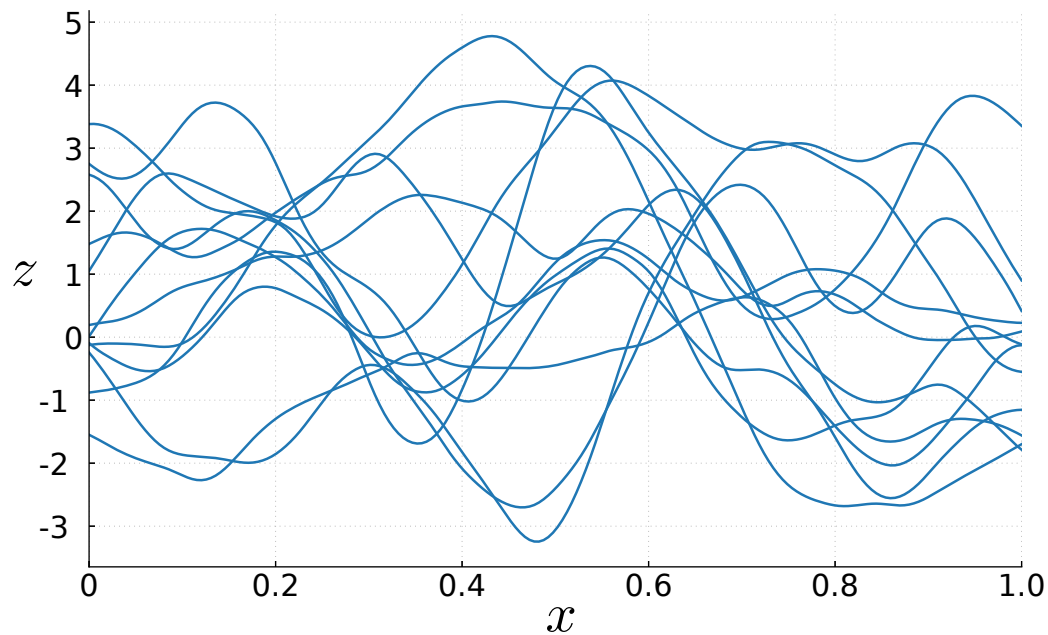
- Any approximation procedure of f consists in building a function $\widehat{f}_n = h(\cdot\,;\,\theta)$, where θ is a vector of parameters, to be chosen using $\mathcal{D}_n$ and available prior information
- e.g., neural network (multilayer perceptron) (try it in colab)



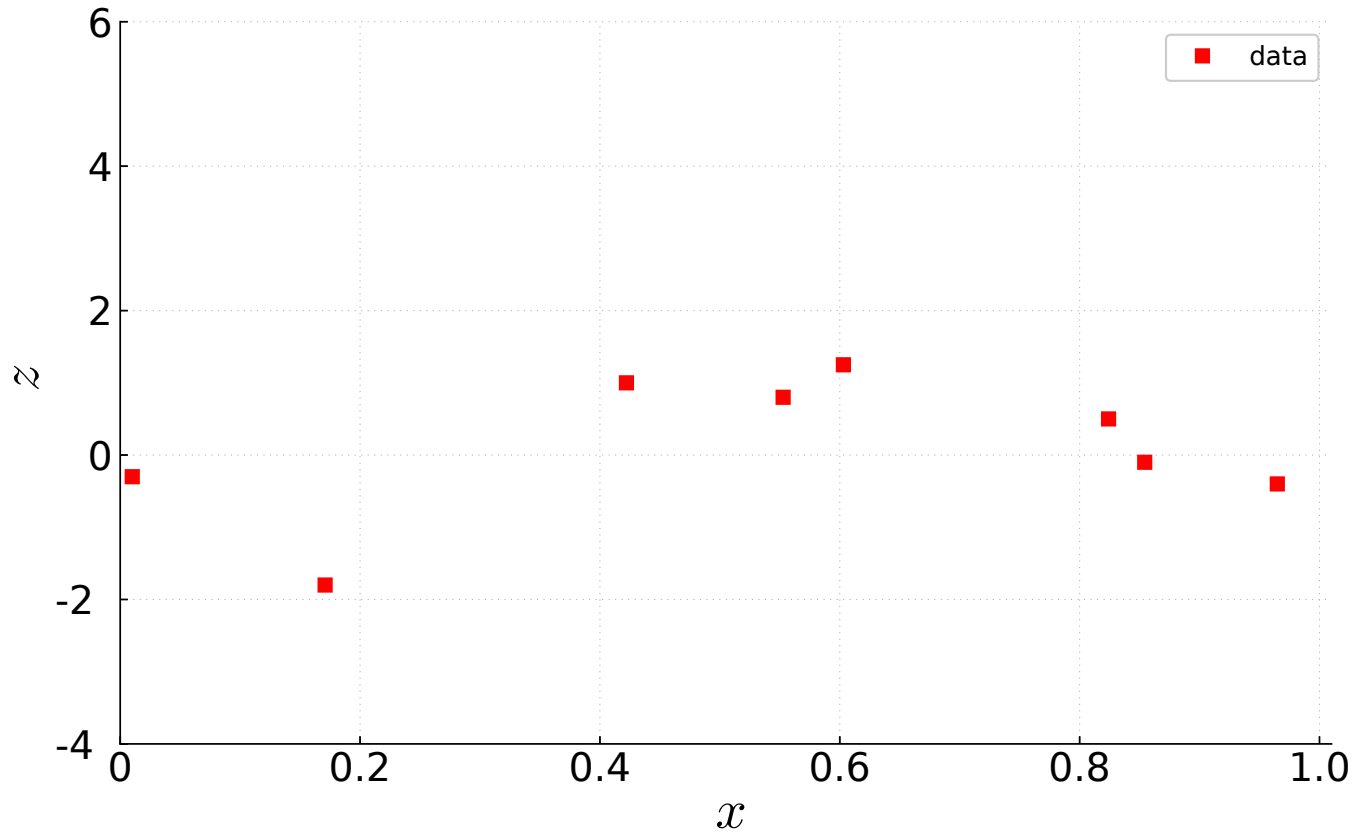
- Why choose one rather the other?
- Moreover, no uncertainty quantification in these approaches!

Bayesian approach: Gaussian processes

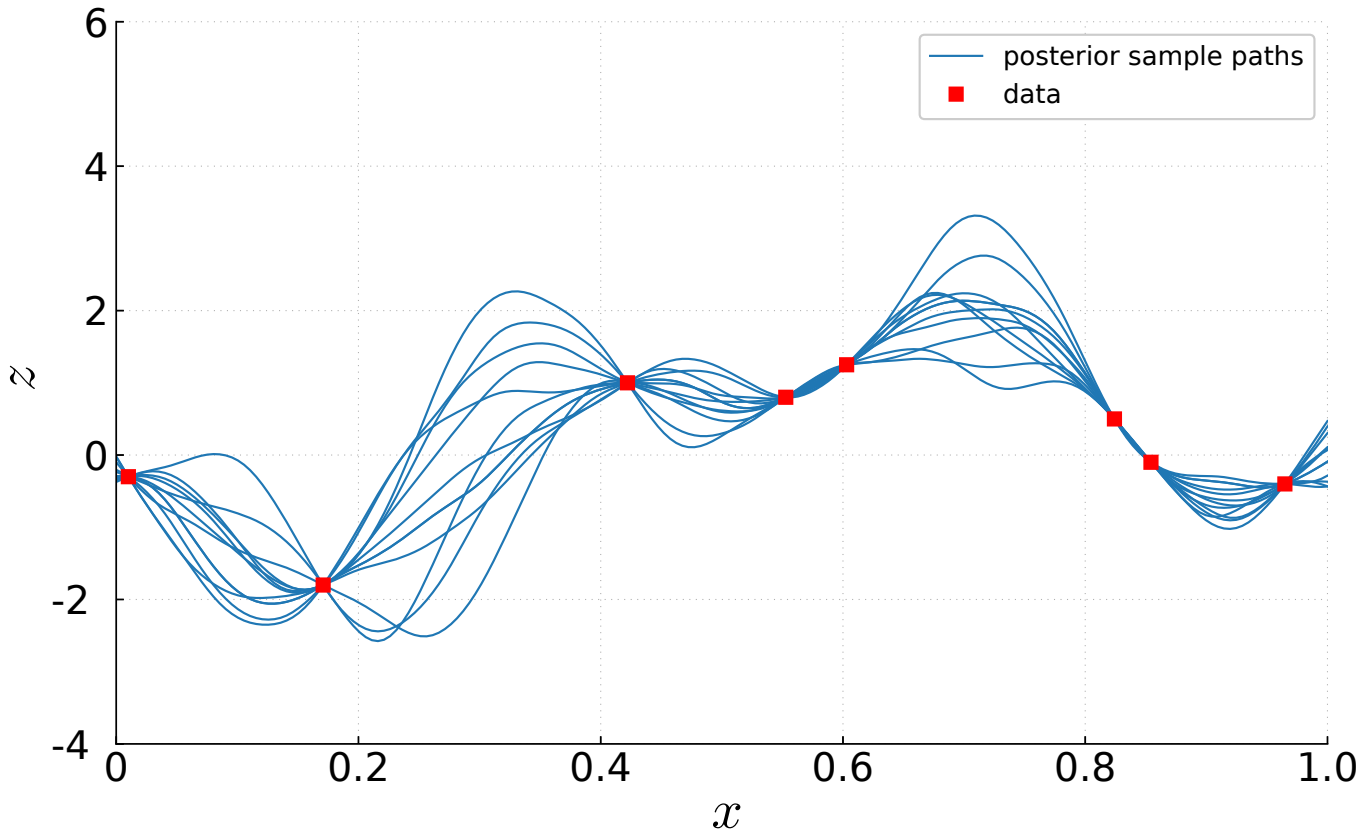
- A non-parametric route (for higher model capacity)
- f is modeled by a random Gaussian process, hereafter denoted by ξ, which encodes our prior knowledge about f



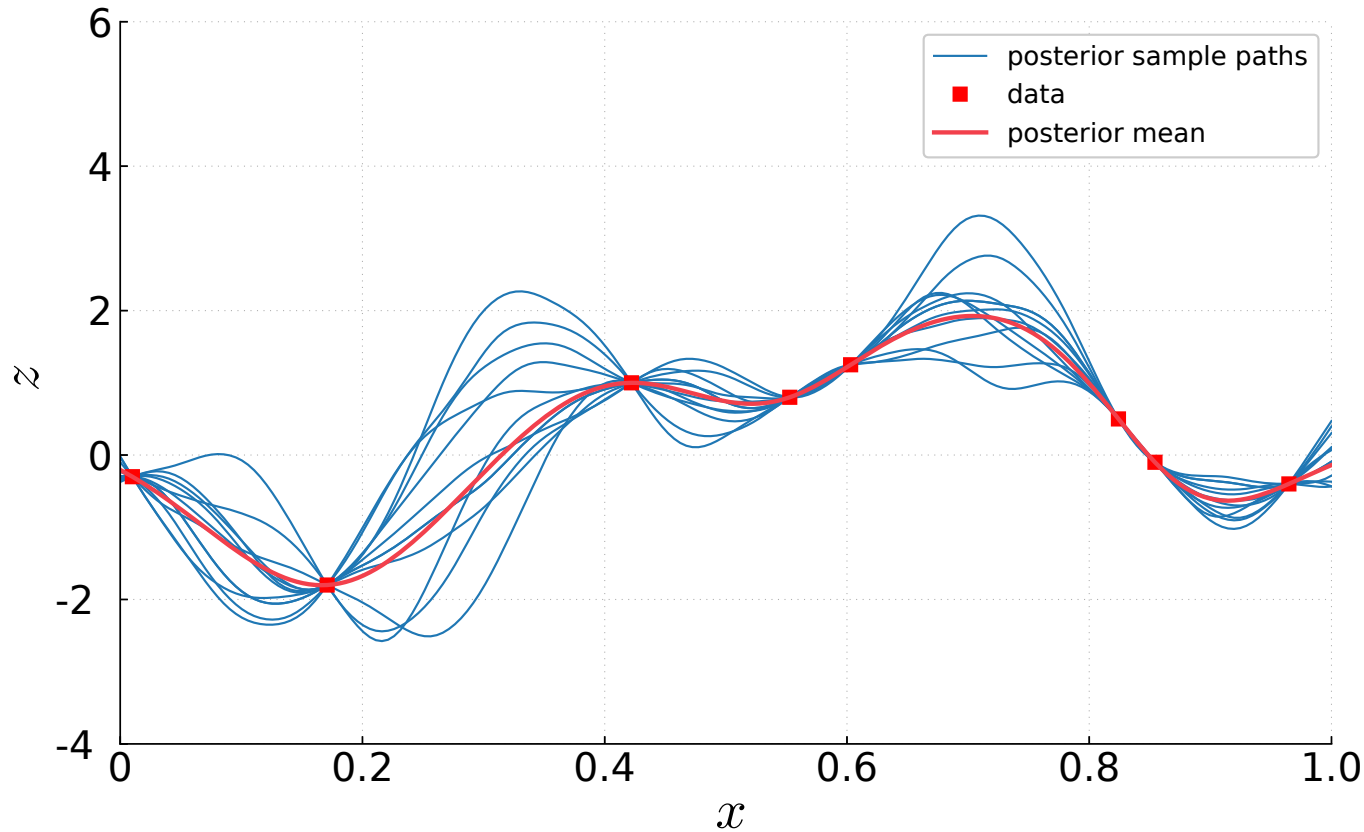
Data points / observations



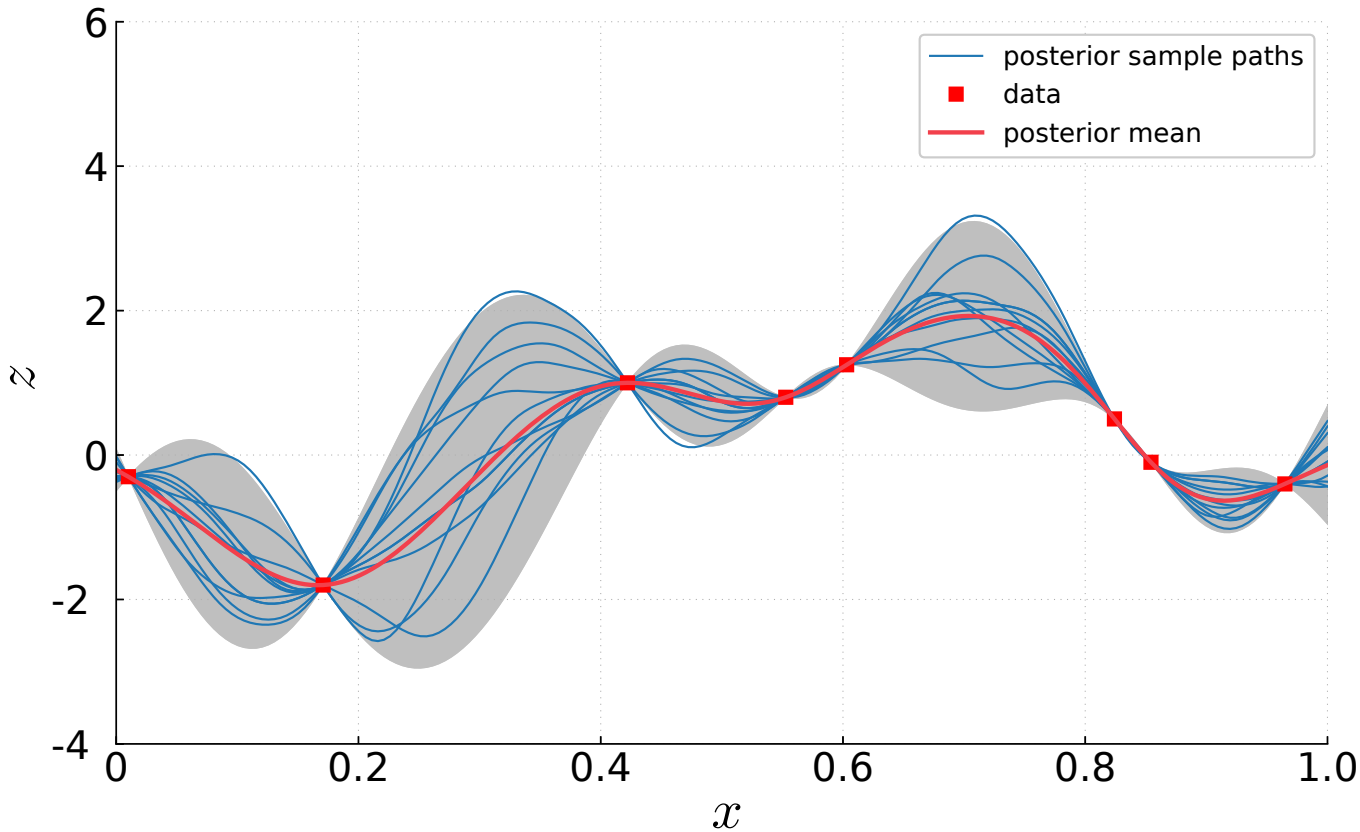
Posterior sample paths of ξ conditioned on observations
(try it in colab)



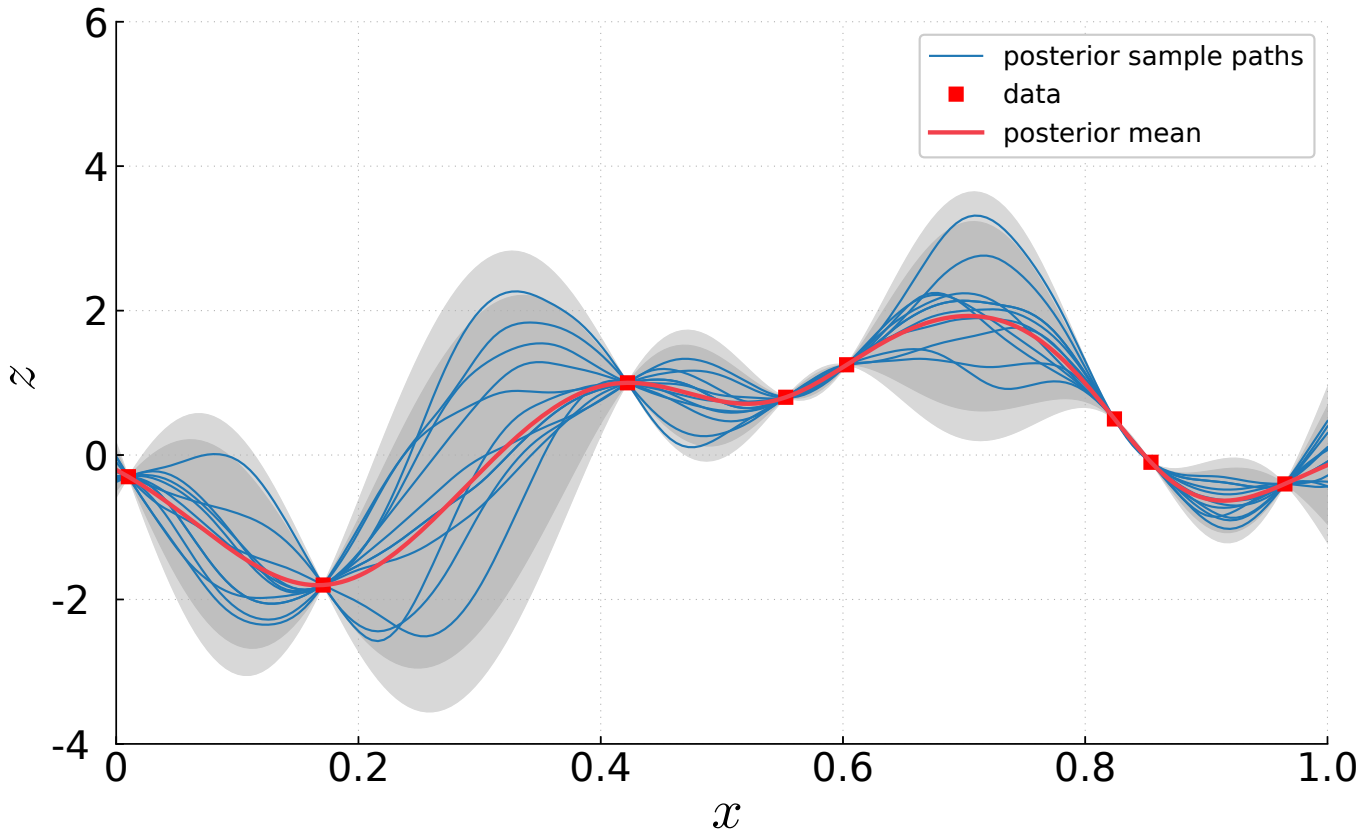
Posterior sample paths of ξ conditioned on observations
(try it in colab)



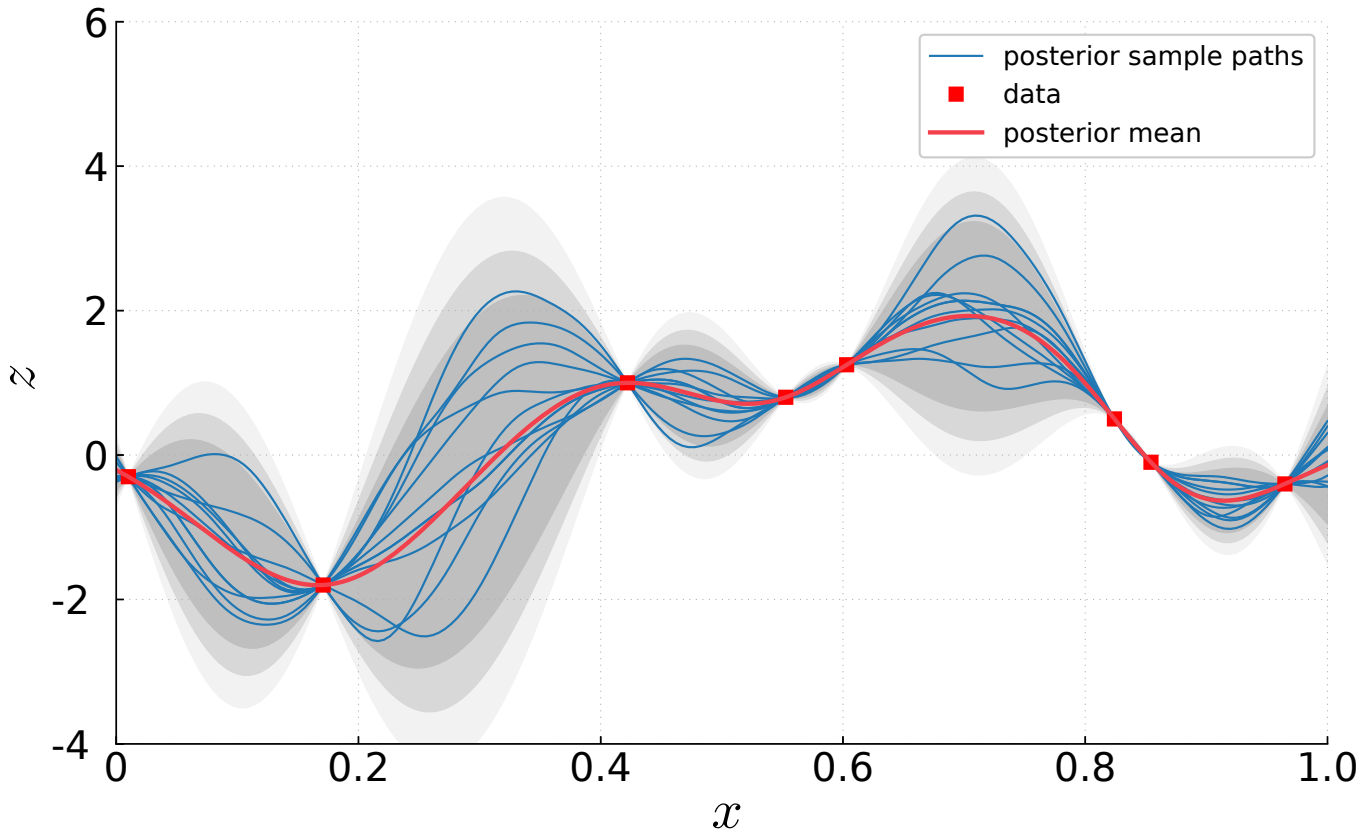
Posterior sample paths of ξ conditioned on observations
(try it in colab)



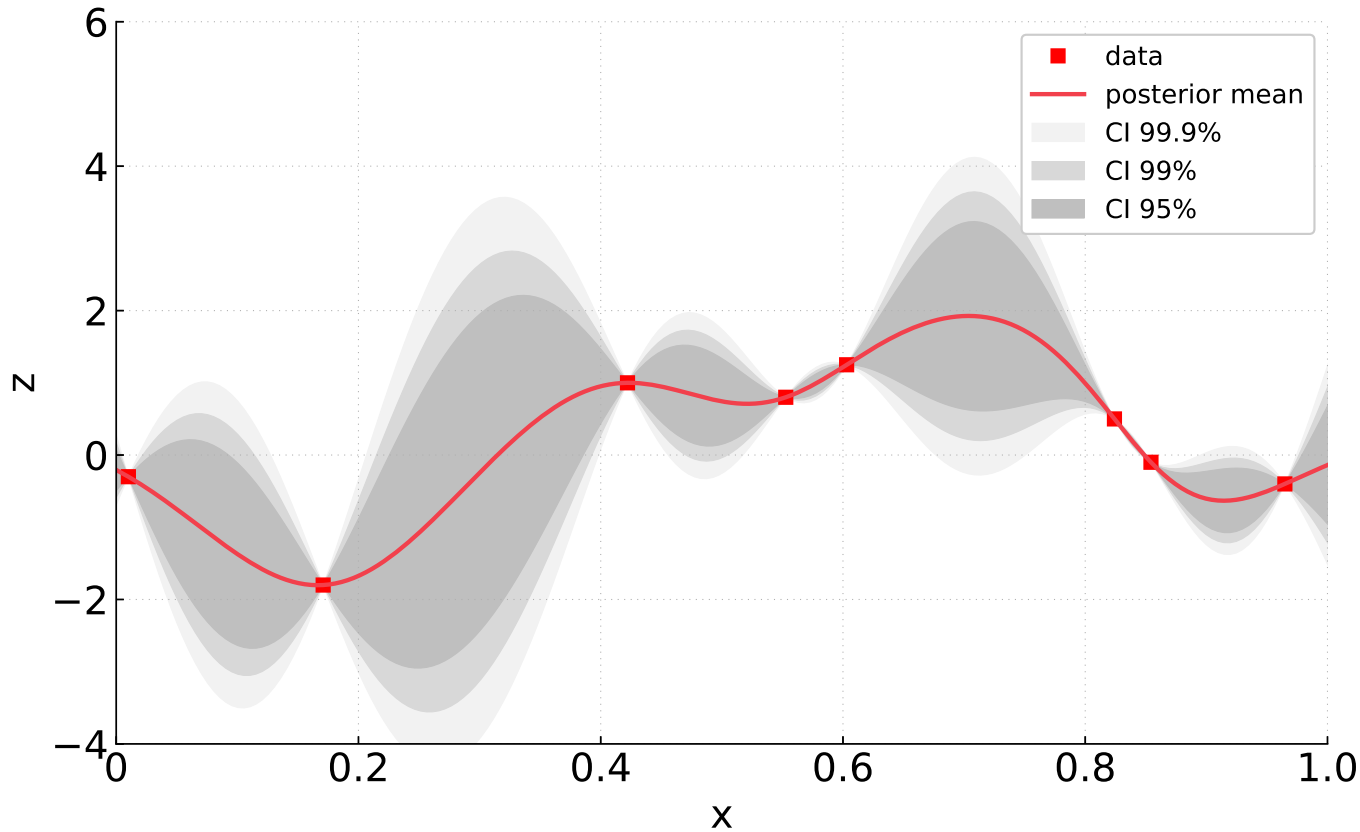
Posterior sample paths of ξ conditioned on observations
(try it in colab)



Posterior sample paths of ξ conditioned on observations
(try it in colab)



Posterior distribution of ξ, obtained by solving a system of linear equations $\leftrightarrow$ aka kriging prediction, G. Matheron ~ 1970



4 Bayesian optimization

- Objective: find an approximation of

$$\left\{ \begin{array}{lcl} M & = & \max_{x \in \mathbb{X}} f(x) \\ x^\star & \in & \arg\max_{x \in \mathbb{X}} f(x) \end{array} \right.$$

  using a sequence of evaluations of f at points $X_1, X_2, \ldots \in \mathbb{X}$

- The construction of an optimization strategy $\underline{X} : f \mapsto (X_1, X_2, X_3 \ldots)$ is a sequential decision problem: at iteration n, we must choose a new evaluation point X_{n+1} using evaluation results of f at $X_1, \ldots, X_n$

Bayesian optimization

- Start with a loss function: the efficiency of the optimization strategy $\underline{X}$ at iteration n can be measured using the loss

$$\varepsilon_n(\underline{X}, f) = M - M_n$$

with $M_n = f(X_1) \vee \cdots \vee f(X_n)$

- Bayesian optimization: f is considered as a sample path of a random process ξ defined on some probability space $(\Omega, \mathcal{B}, \mathsf{P}_0)$, with parameter $x \in \mathbb{X}$
- Information at iteration n is a realization of

$$\mathcal{F}_n = \{X_1, \xi(X_1) \ldots, X_n, \xi(X_n)\}$$

- Denote by E_n the conditional expectation $\mathsf{E}(\cdot \mid \mathcal{F}_n)$

- At each step, given the outcome for $\mathcal{F}_n$, we want to minimize the risk (expected loss)

$$H_n = \mathsf{E}_n[\varepsilon_n(\underline{X}, \xi)] = \mathsf{E}_n(M - M_n)$$

- H_n can be viewed as a measure of residual uncertainty about M
- At iteration n, we choose X_{n+1} to minimize the expectation of H_{n+1}:

$$X_{n+1} = \arg\min_{x \in \mathbb{X}} \quad \mathsf{E}_n\big(H_{n+1} \mid X_{n+1} = x\big)$$

- This is a Bayesian decision-theoretic point of view for optimization, explored by J. Mockus, A. Žilinskas and their coauthors ($\sim$ 1970–1990)
- Extended to other settings under the name of SUR: Sequential Uncertainty Reduction (V. & Martinez 2006, V. & Bect 2009, Bect et al. 2012, Bect et al. 2019...)

- We have

$$\begin{aligned} X_{n+1} &= \arg\min_{x\in\mathbb{X}} \mathsf{E}_n\left(M - M_{n+1} \,\middle|\, X_{n+1} = x\right) \\ &= \arg\max_{x\in\mathbb{X}} \mathsf{E}_n\left(M_{n+1} \,\middle|\, X_{n+1} = x\right) \\ &= \arg\max_{x\in\mathbb{X}} \mathsf{E}_n\big(M_n \vee \xi(x)\big) \\ &= \arg\max_{x\in\mathbb{X}} \rho_n(x) \end{aligned}$$

  with $\rho_n(x) := \mathsf{E}_n\big((\xi(x) - M_n)_+\big)$, and $z_+ = \max(z, 0)$.

- ρ_n corresponds to the average excursion of $\xi(x)$ above the current maximum of past evaluation results

- ρ_n is called the expected improvement (EI) sampling criterion

- The strategy $X_{n+1} = \arg\max_{x \in \mathbb{X}} \rho_n(x)$ using a Gaussian process for ξ has been popularized by D. Jones et al. 1998 under the name of EGO (Efficient Global Optimization)

- When ξ is a GP, with known mean and covariance functions, $\rho_n(x)$ has a closed-form expression:

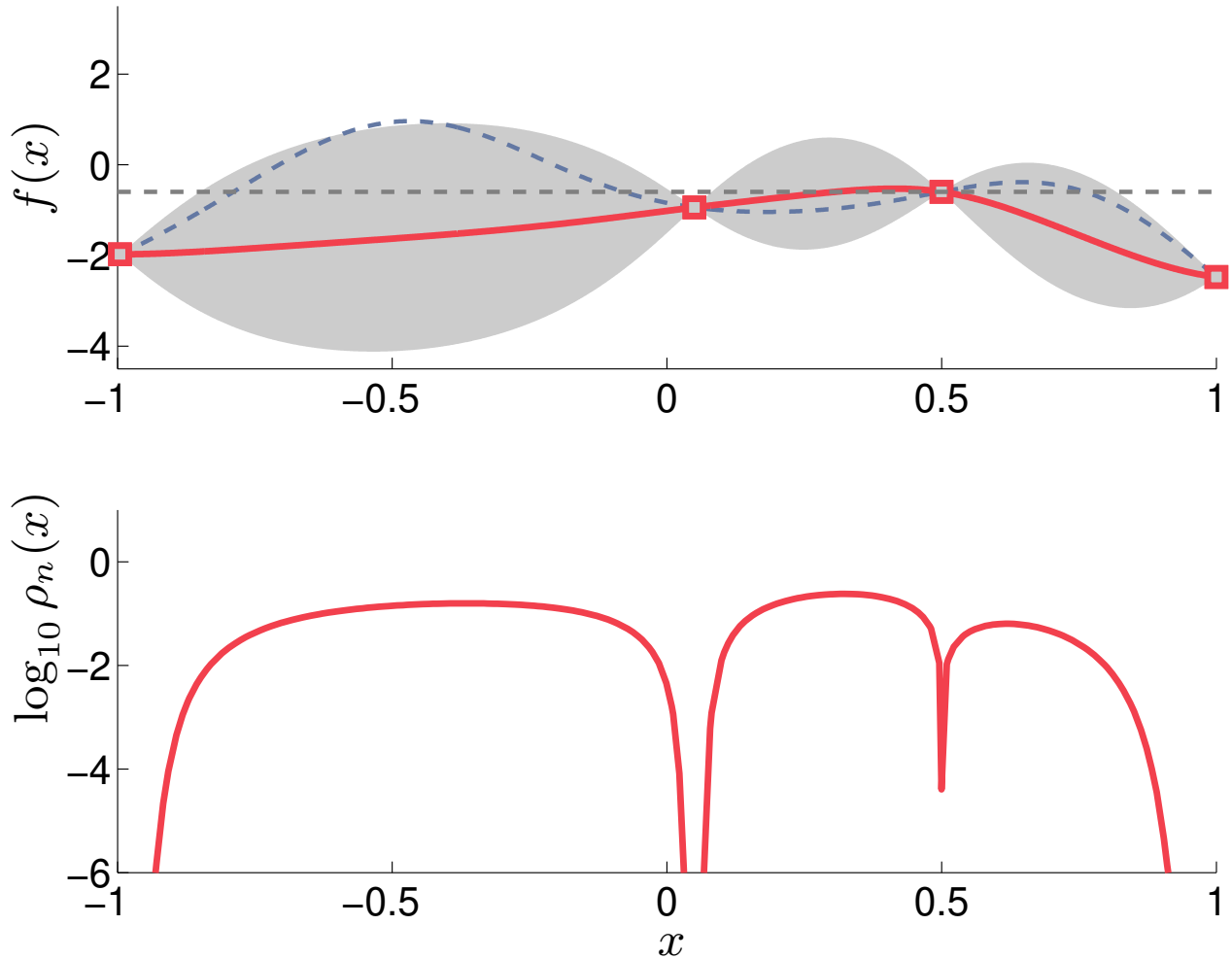
$$\rho_n(x) \;=\; \gamma\left(\widehat{\xi}_n(x;\underline{X}_n) - M_n,\, \sigma_n^2(x)\right),$$

where

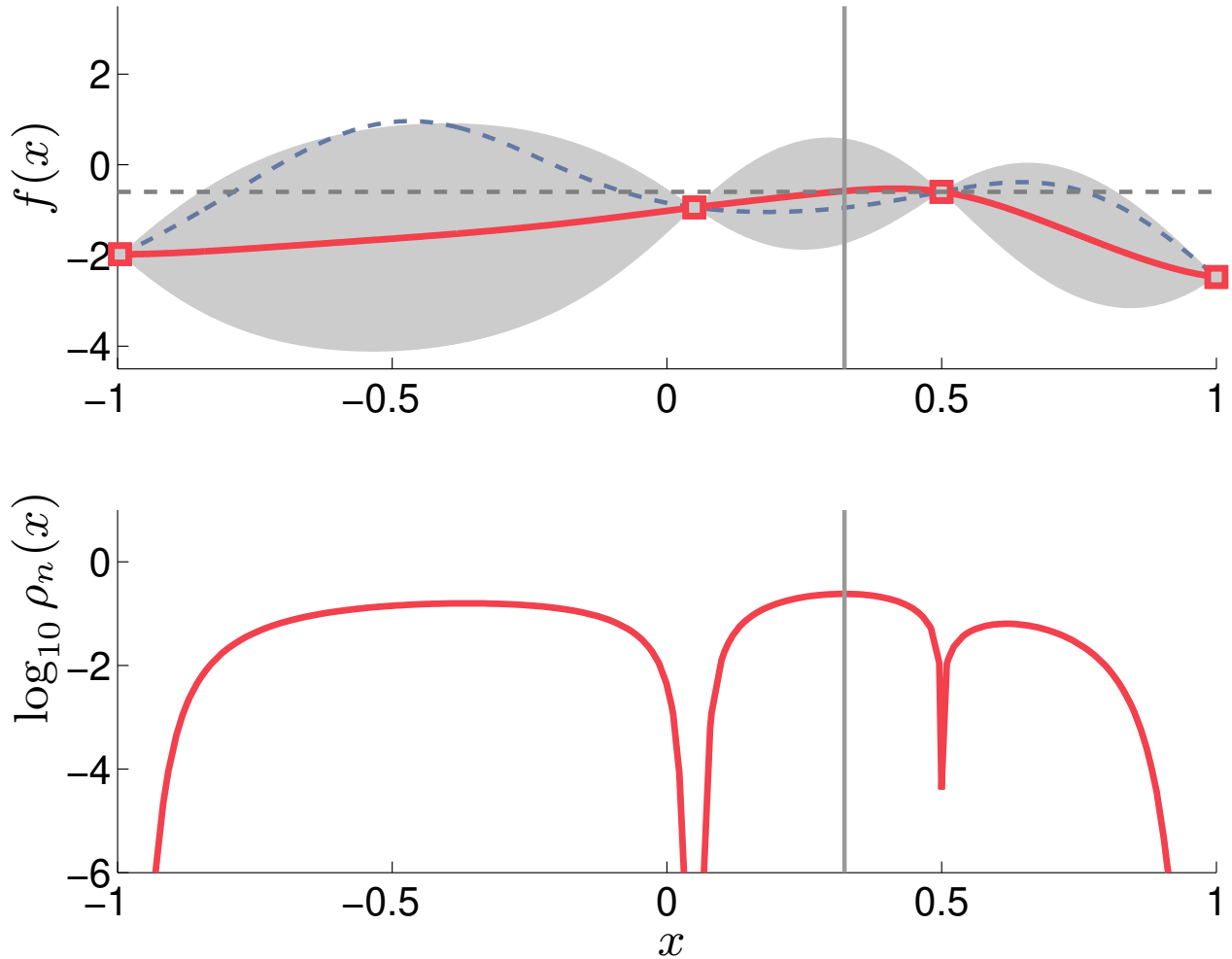
$$\gamma(z,s) \;=\; \begin{cases} \sqrt{s}\,\Phi'\left(\frac{z}{\sqrt{s}}\right) + z\,\Phi\left(\frac{z}{\sqrt{s}}\right) & \text{if } s > 0, \\ \max(z,0) & \text{if } s = 0. \end{cases}$$

and $\widehat{\xi}_n(x;\underline{X}_n)$ and $\sigma_n^2(x)$ are the posterior mean and the posterior variance of $\xi(x)$

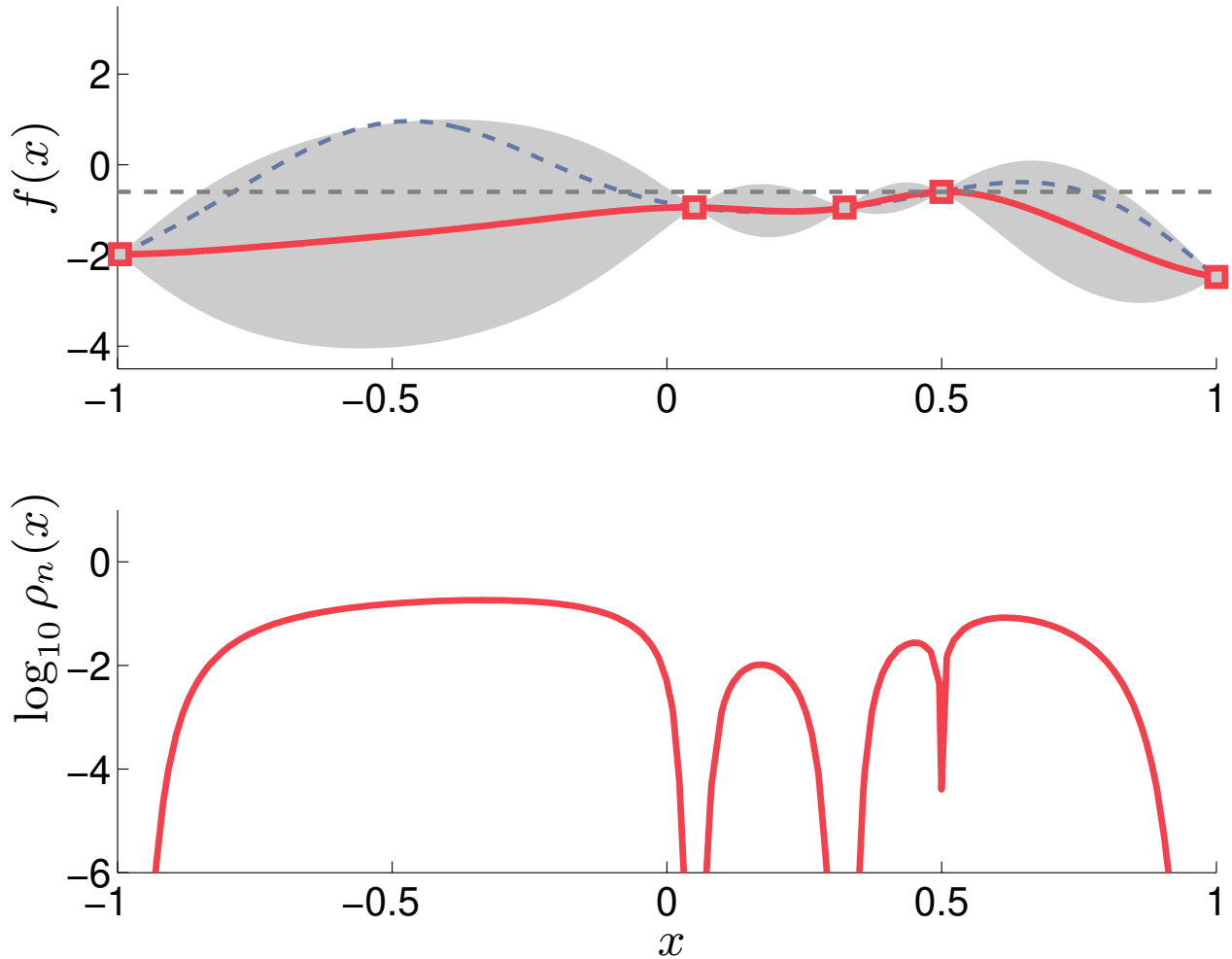
Function to be maximized (dashed blue line) and kriging prediction



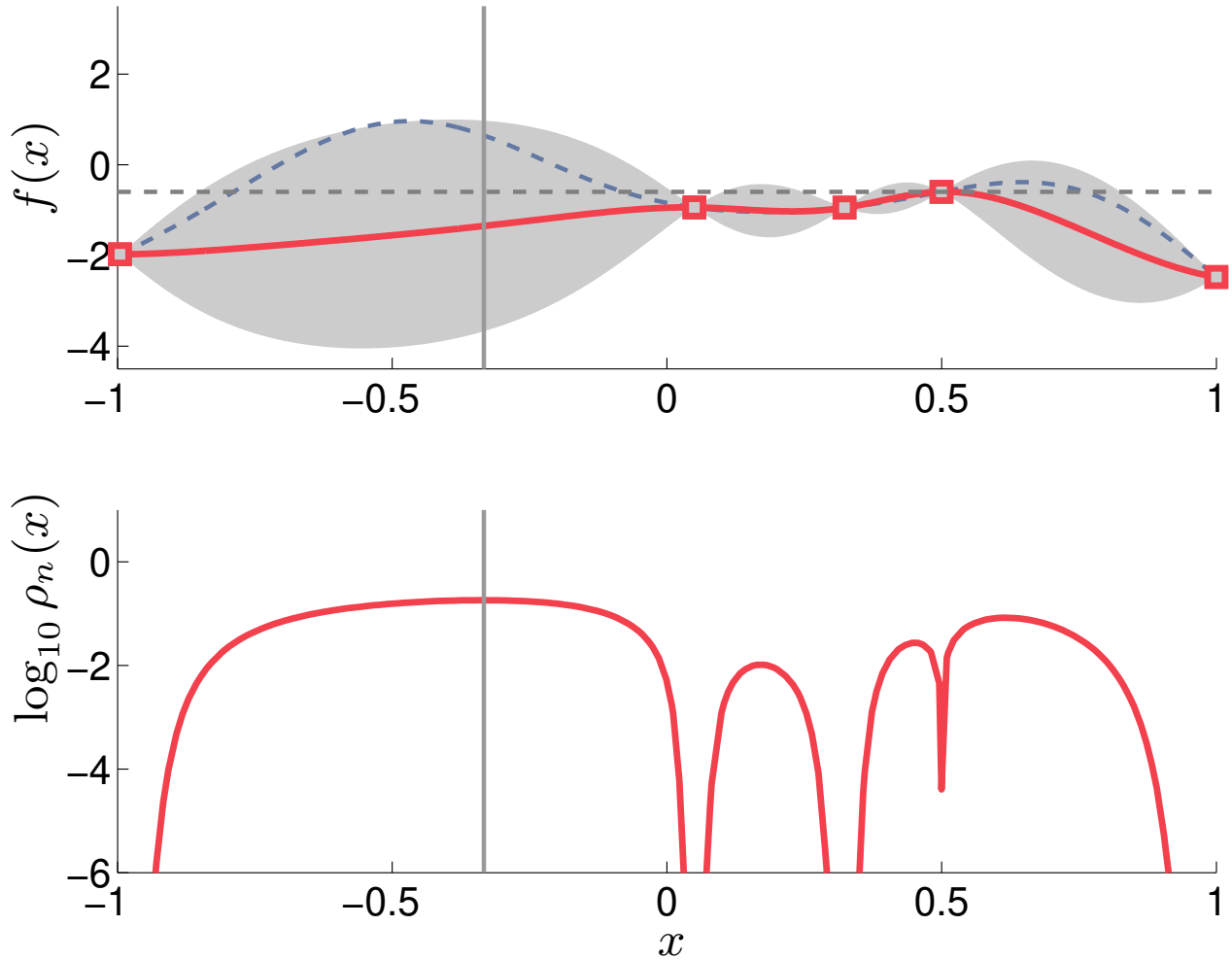
Function to be maximized (dashed blue line) and kriging prediction



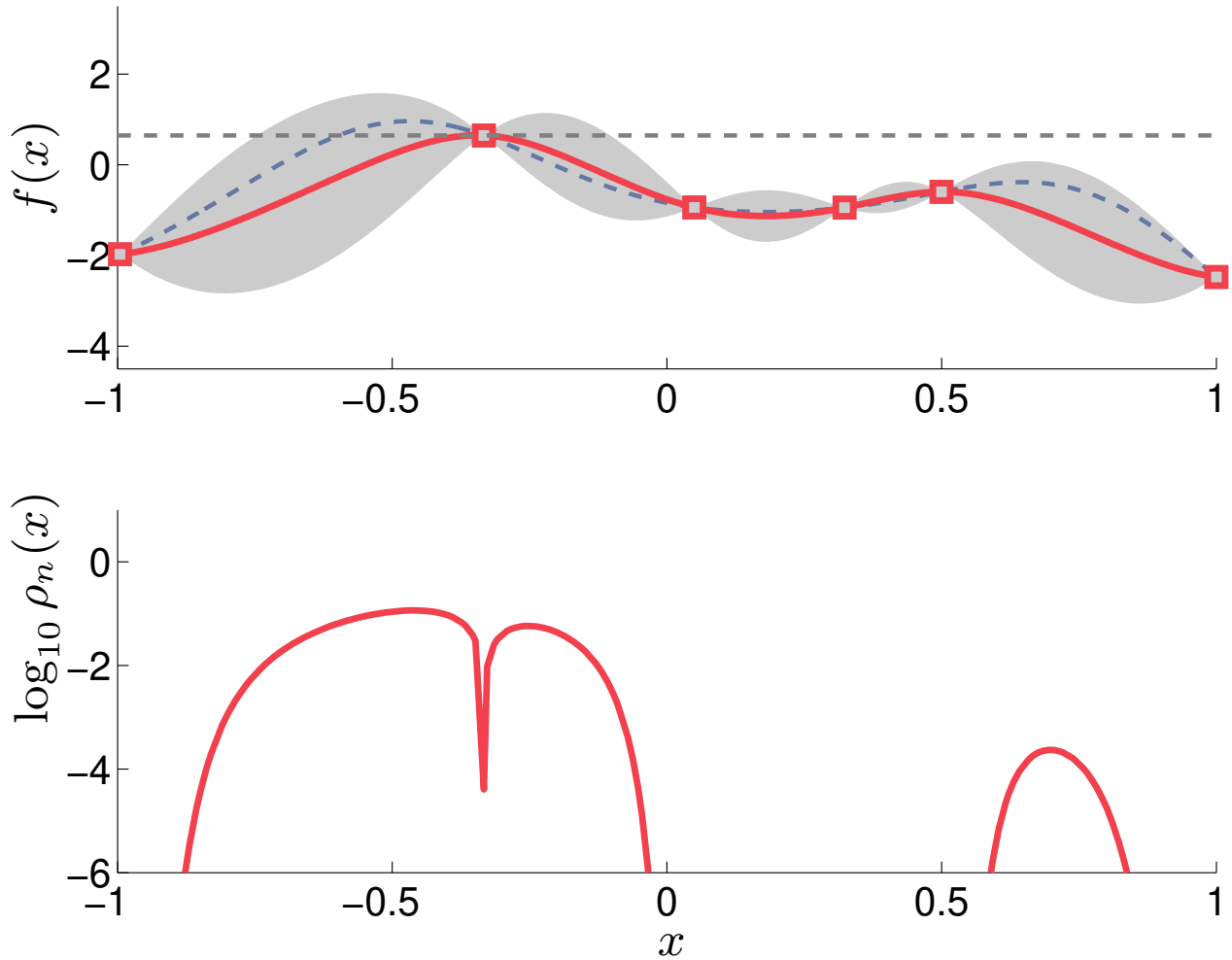
Function to be maximized (dashed blue line) and kriging prediction



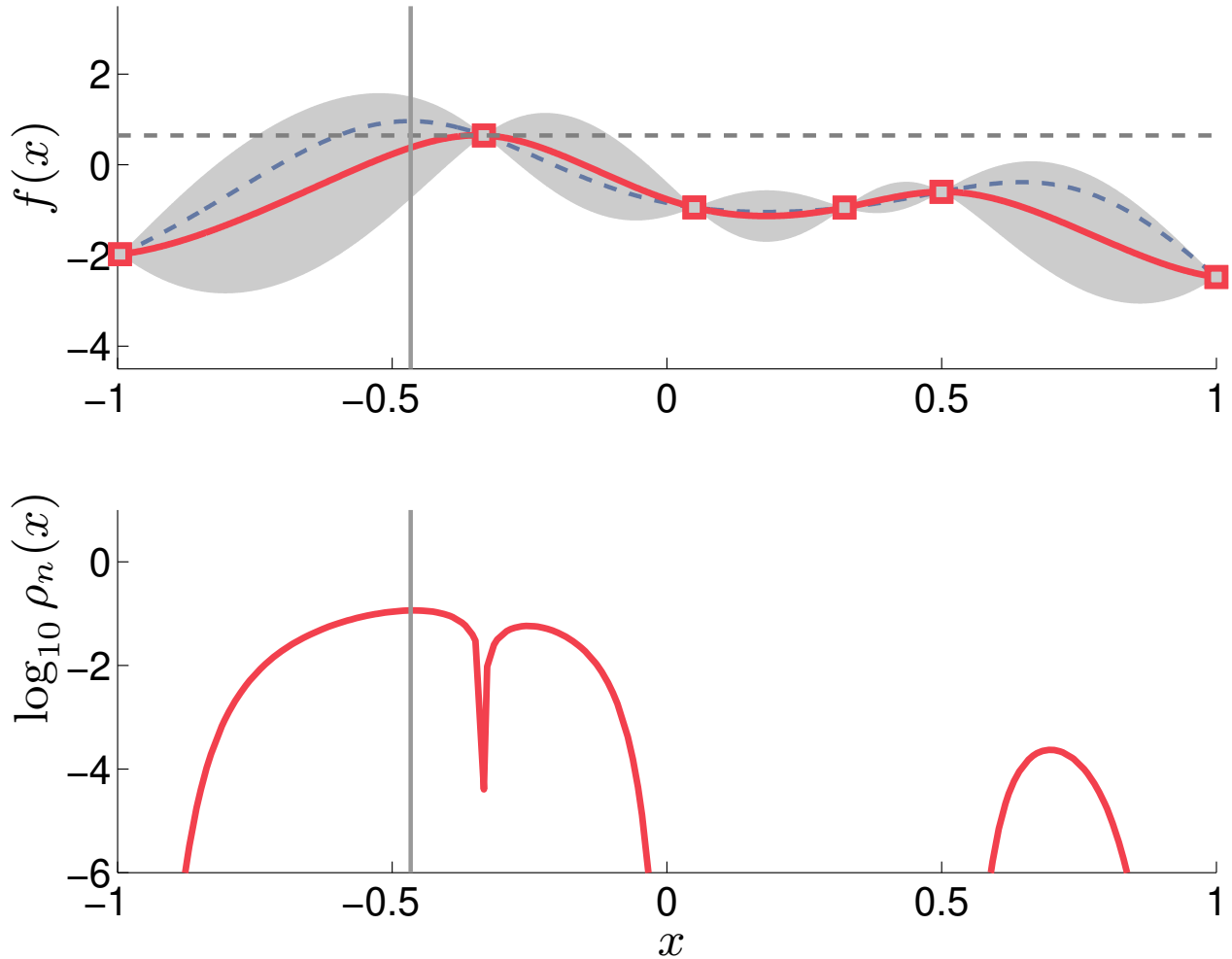
Function to be maximized (dashed blue line) and kriging prediction



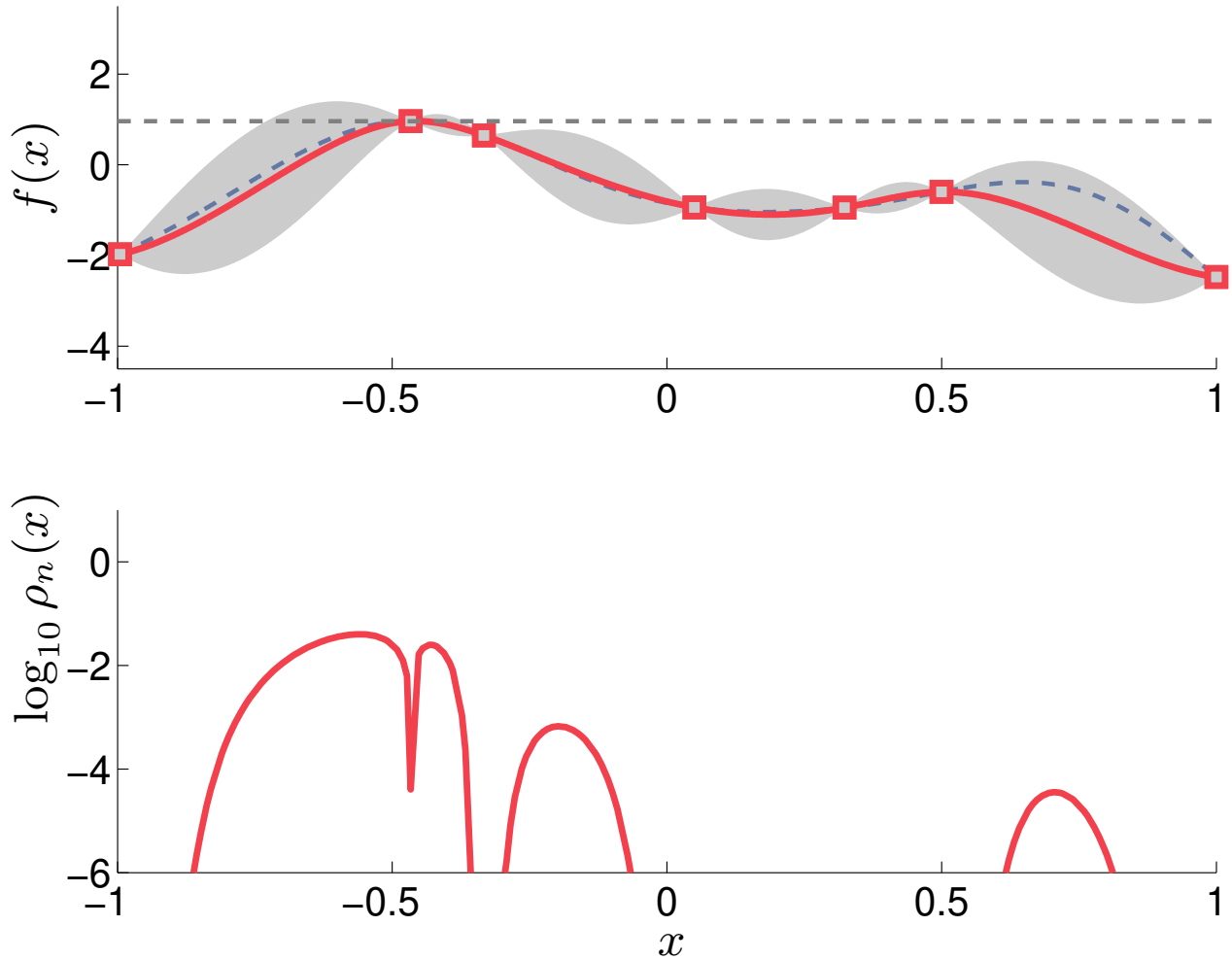
Function to be maximized (dashed blue line) and kriging prediction



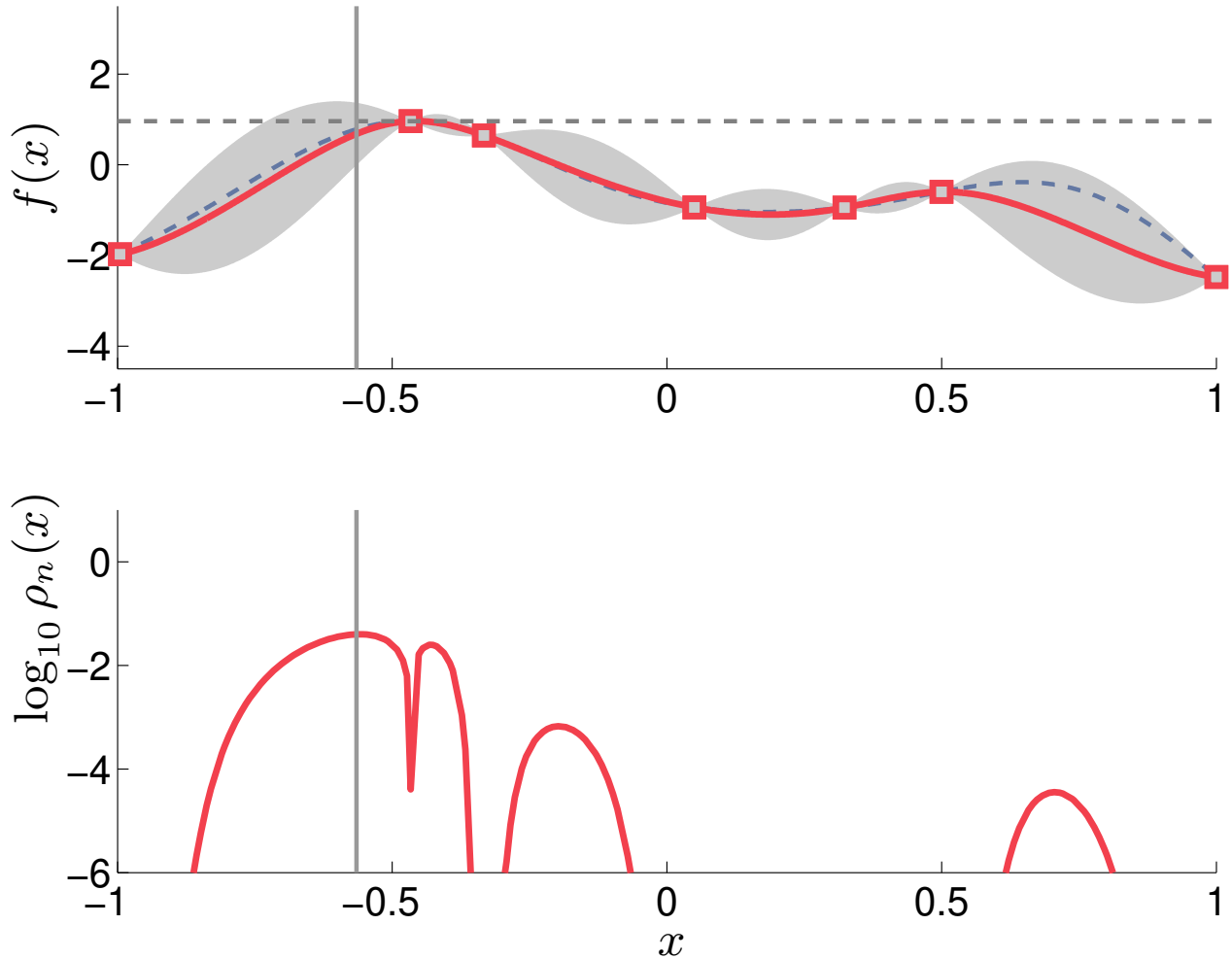
Function to be maximized (dashed blue line) and kriging prediction



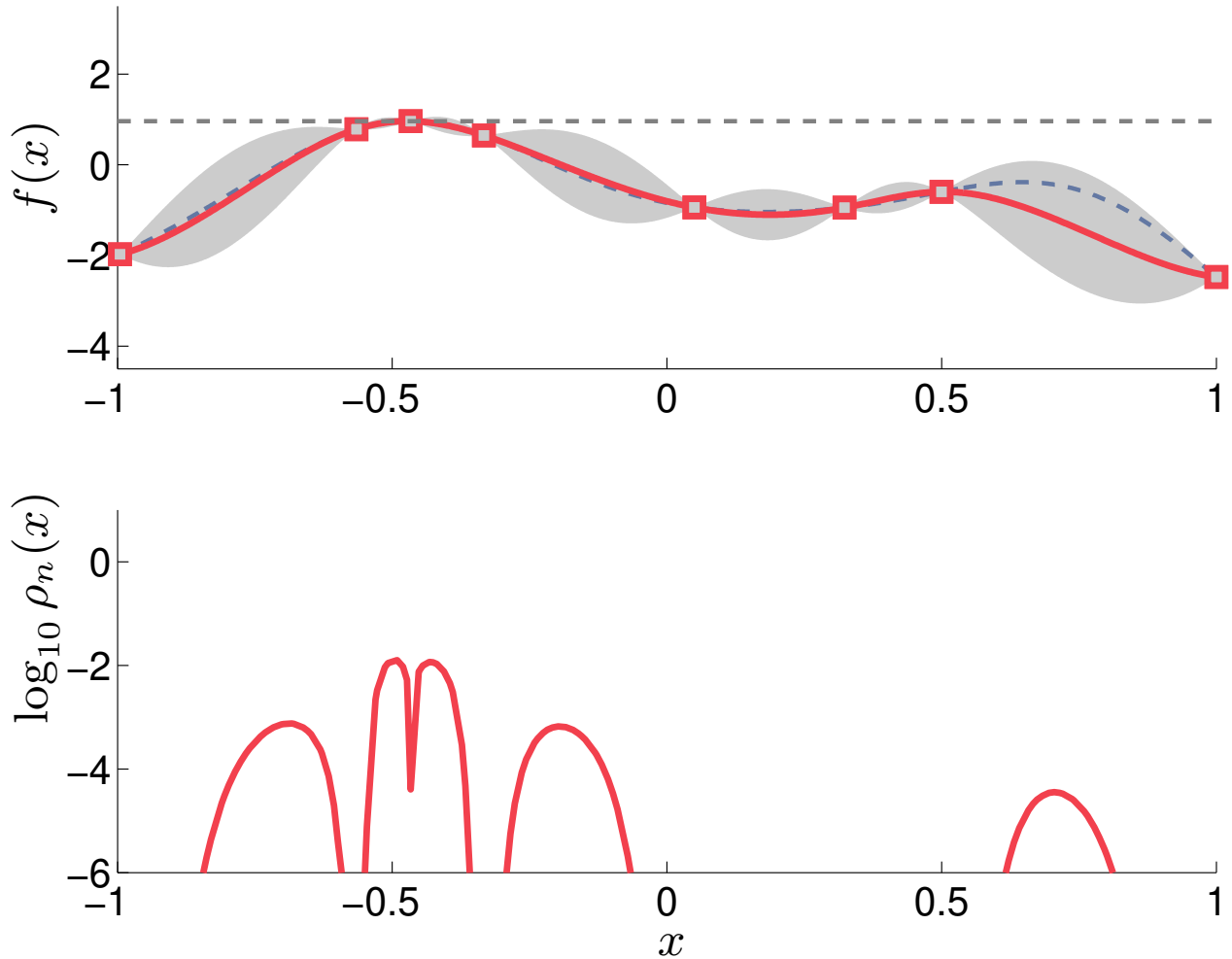
Function to be maximized (dashed blue line) and kriging prediction



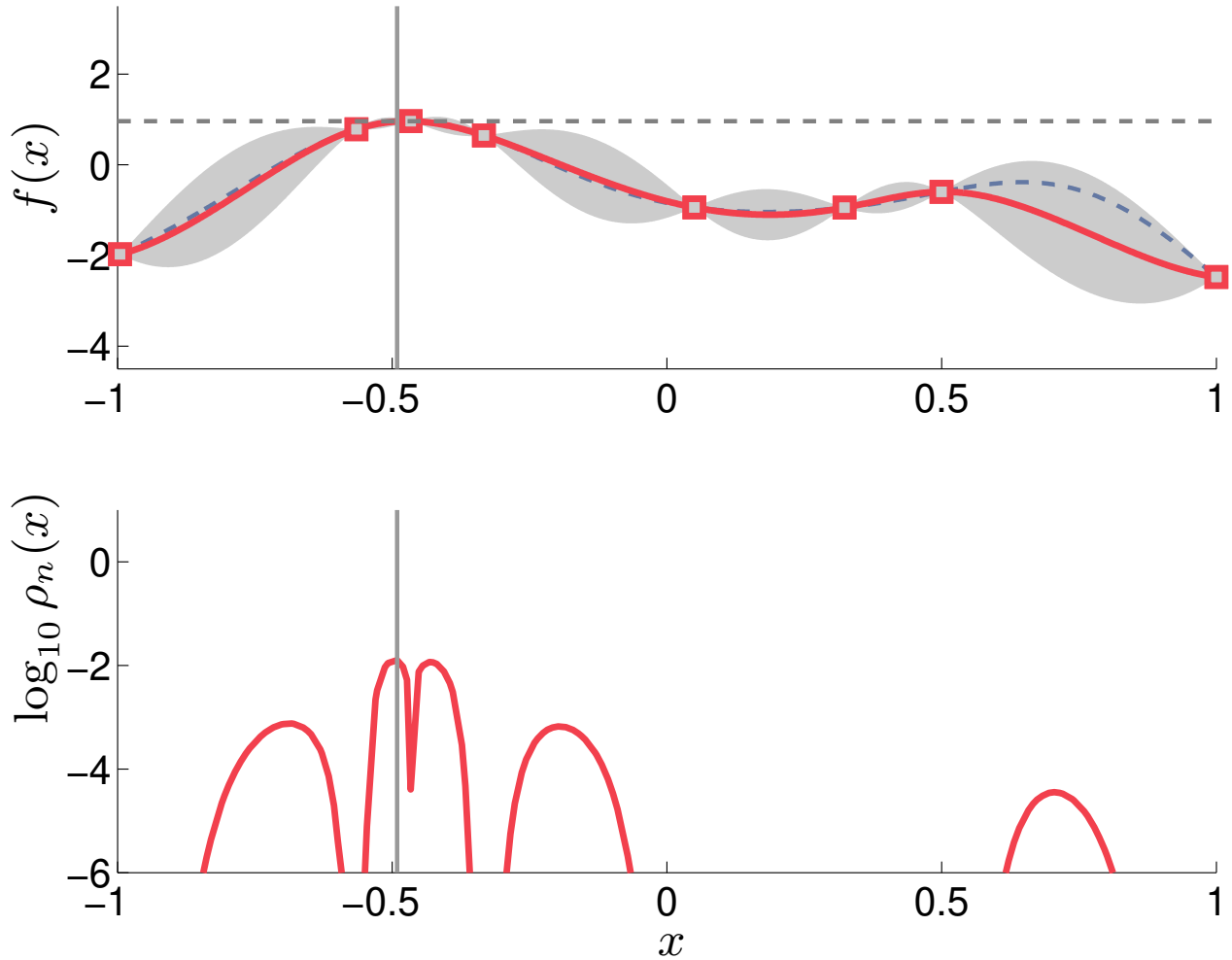
Function to be maximized (dashed blue line) and kriging prediction



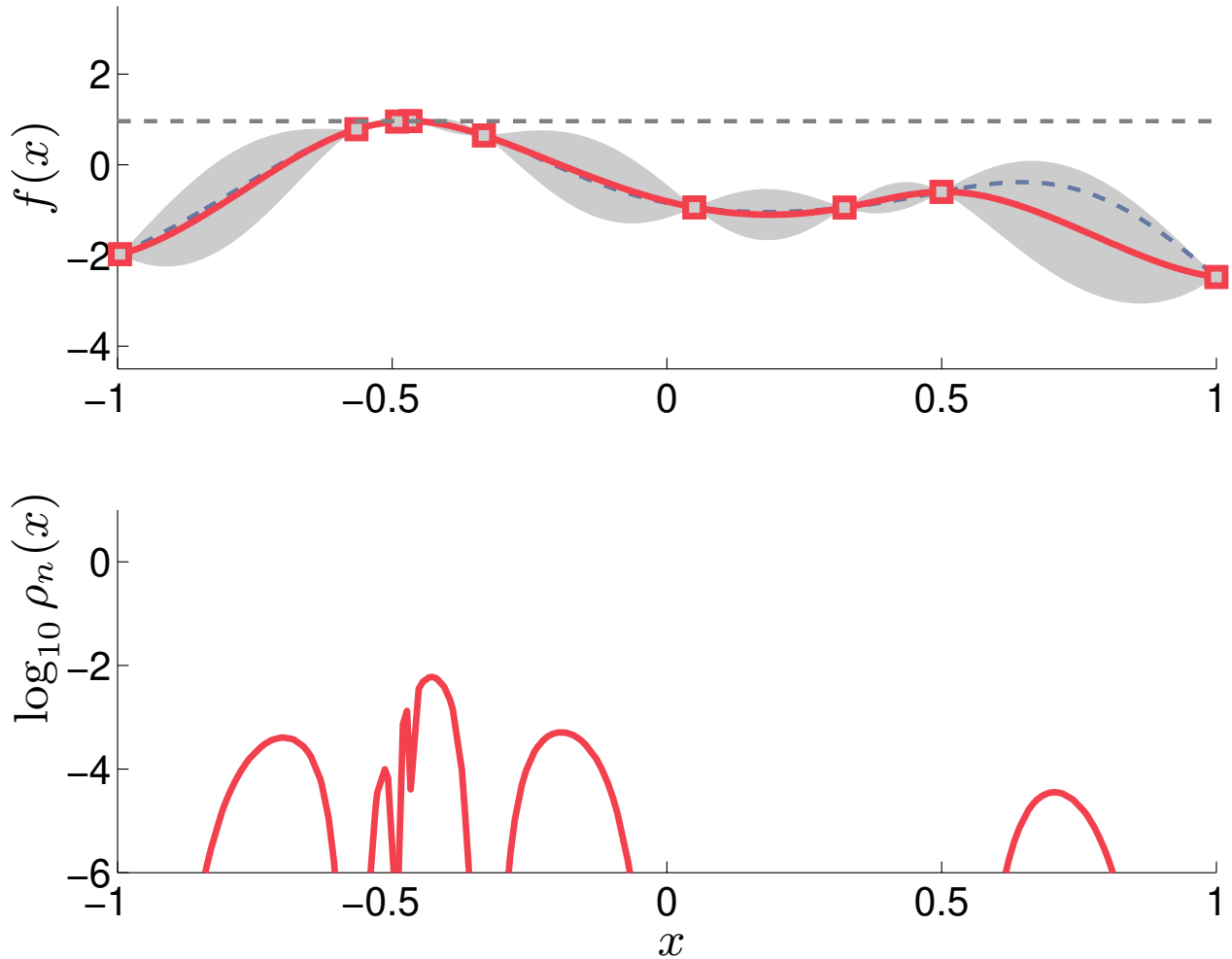
Function to be maximized (dashed blue line) and kriging prediction



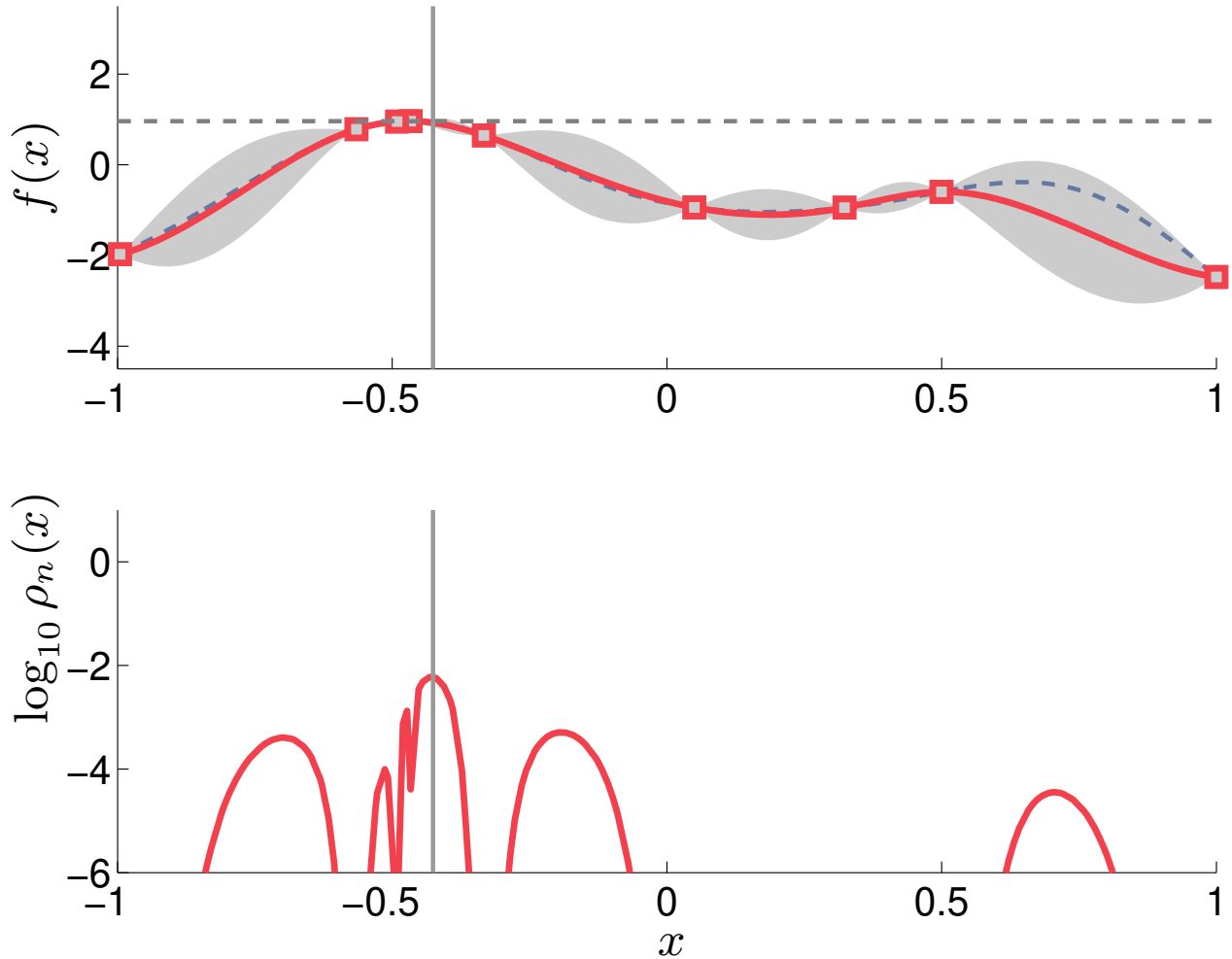
Function to be maximized (dashed blue line) and kriging prediction



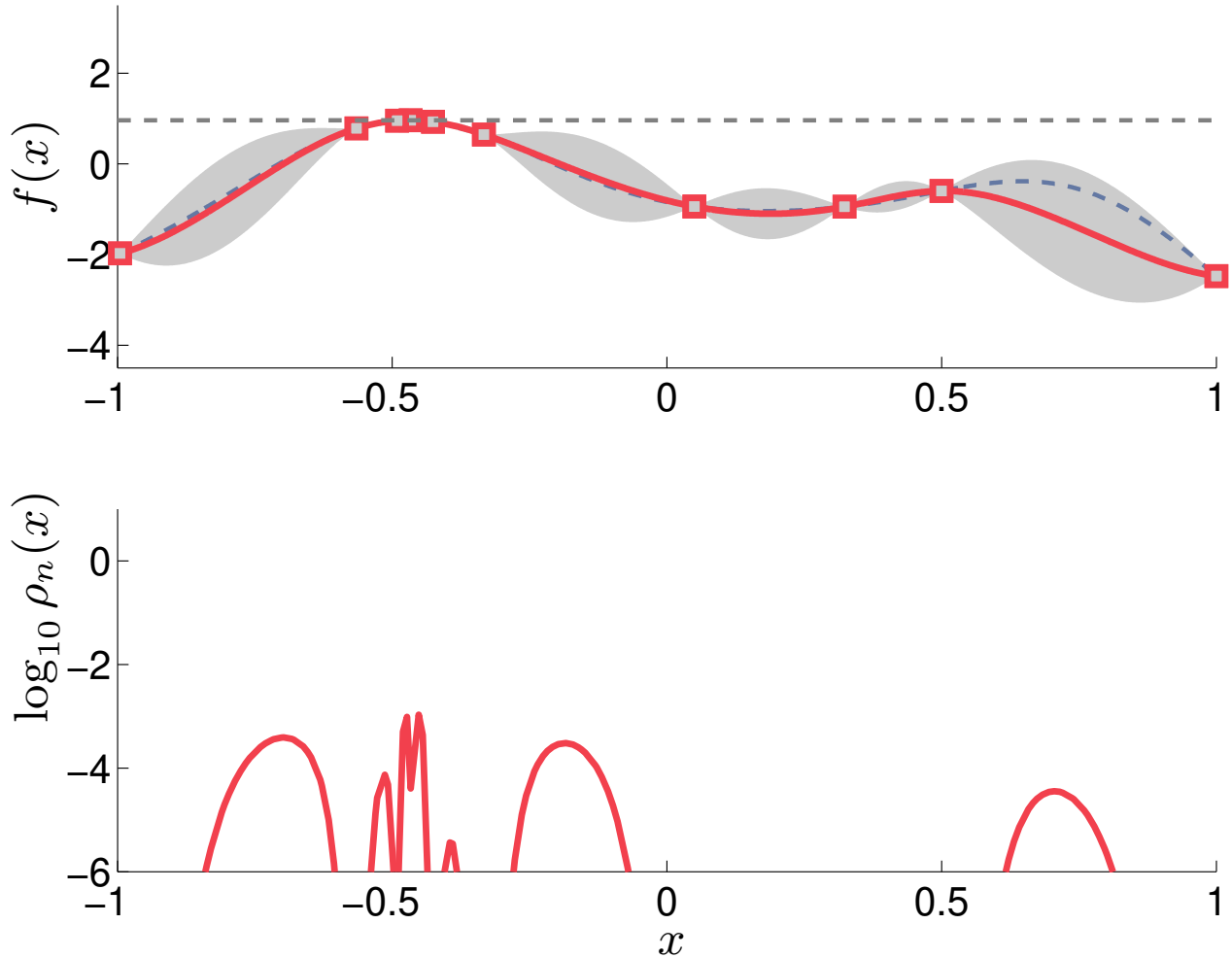
Function to be maximized (dashed blue line) and kriging prediction



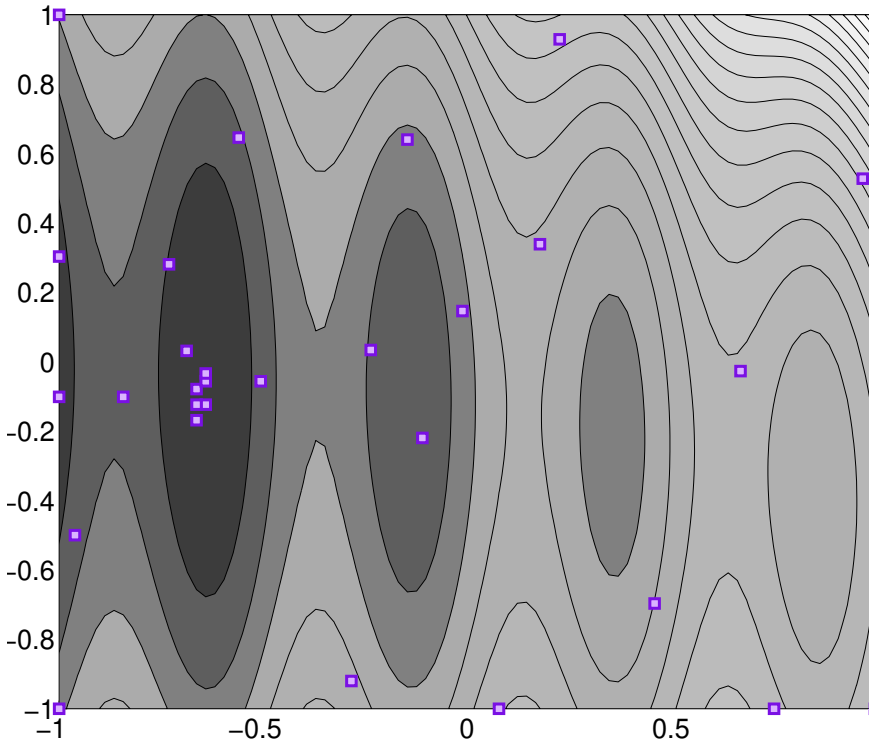
Function to be maximized (dashed blue line) and kriging prediction



Function to be maximized (dashed blue line) and kriging prediction



2d illustration



Exploration vs Exploitation (31 evaluations)

	M_n with $N = 60$
LHS	-5.823
DIRECT	-5.839
EI (31 evals)	-5.845
Global minimum	-5.845

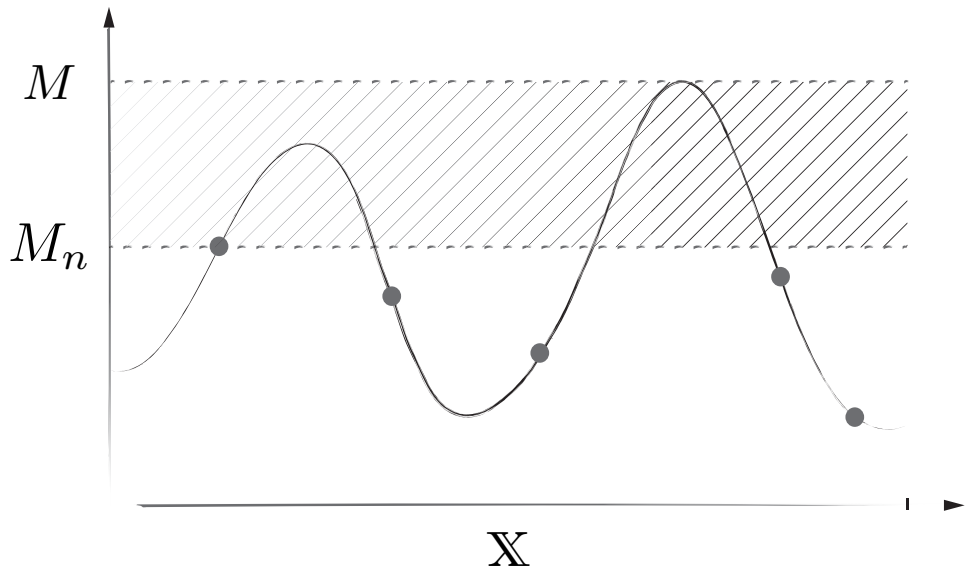
- The EGO/EI algorithm has been used in countless of engineering design problems and is widely available (Python, Matlab, R...)
- The convergence of the algorithm is well established (V. & Bect 2010)
- Convergence rates are partially known (Bull 2011, Ryzhov 2016...)

Other Bayesian strategies for mono-objective optimization?

- In a global optimization problem, it is generally of interest to obtain a good approximation of both M and $x^\star$
- The loss function $\varepsilon_n(\underline{X}, f) = M - M_n$ does not measure directly the distance of $x_n^\star$ to $x^\star$
- Better alternative(s)?

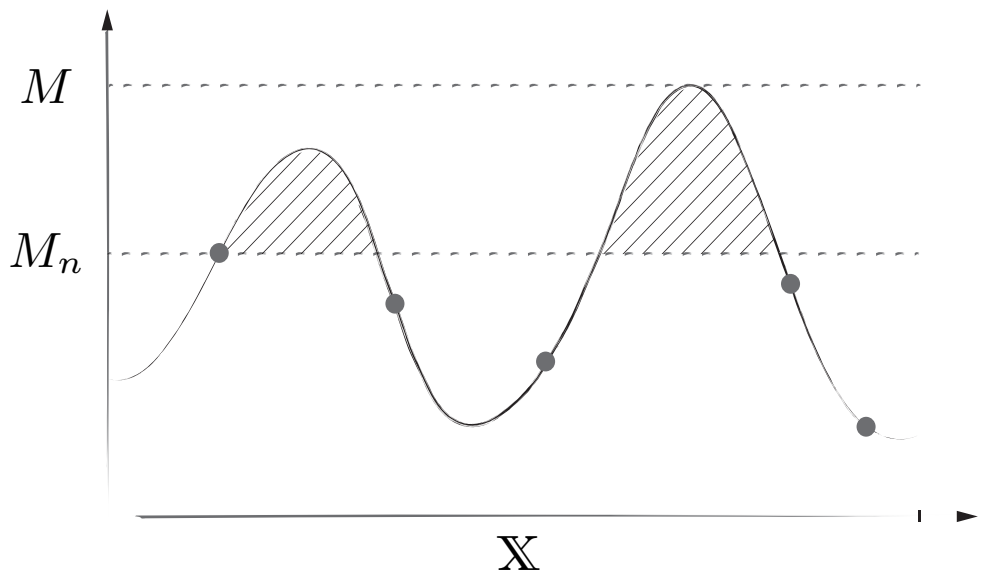
Expected integrated expected improvement

- Note that $\varepsilon_n(\underline{X}, f) = M - M_n \propto \lambda(\mathbb{X})\,(M - M_n)$



$\rightarrow$ coarse measure of the uncertainty about the pair $(M, x^\star)$

- Idea: use the integral loss $\varepsilon_n(\underline{X}, f) = \int_{\mathbb{X}} (f(x) - M_n)_+ \, \lambda(\mathrm{d}x)$



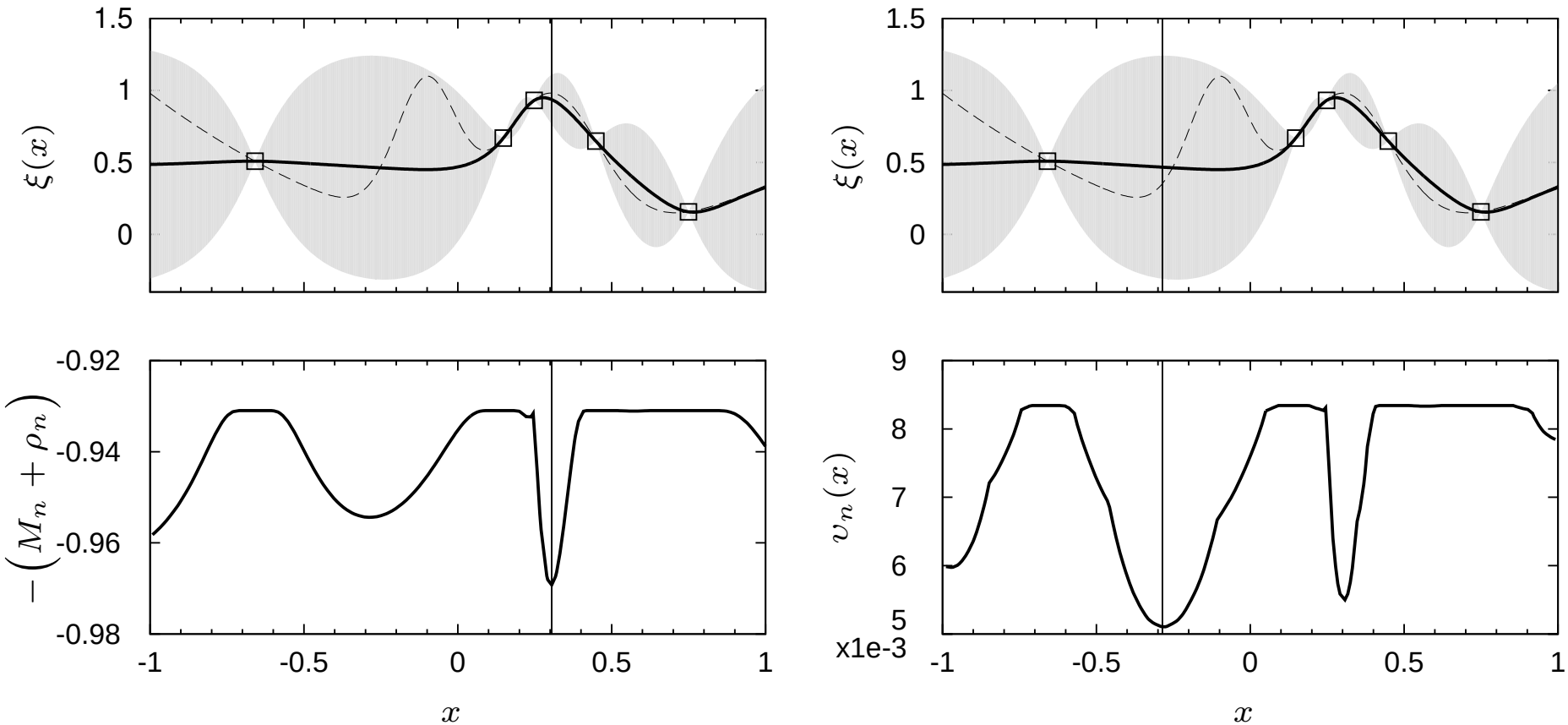
$\rightarrow$ ε_n gets smaller when uncertainty about $x^\star$ decreases

- SUR strategy:

$$
\begin{aligned}
X_{n+1} &= \arg\min_{x\in\mathbb{X}} \mathsf{E}_n\left(\int_{\mathbb{X}} (\xi(y)-M_{n+1})_+ \,\mathrm{d}y \;\middle|\; X_{n+1}=x\right)\\
&= \arg\min_{x\in\mathbb{X}} \mathsf{E}_n\left(\int_{\mathbb{X}} \mathsf{E}_{n+1}\big((\xi(y)-M_{n+1})_+\big) \,\mathrm{d}y \;\middle|\; X_{n+1}=x\right)\\
&= \arg\min_{x\in\mathbb{X}} \upsilon_n(x) := \mathsf{E}_n\left(\int_{\mathbb{X}} \rho_{n+1}(y)\mathrm{d}y \;\middle|\; X_{n+1}=x\right)
\end{aligned}
$$

- We call υ_n the Expected Integrated Expected Improvement (EI^2) (V. & Bect, 2014)

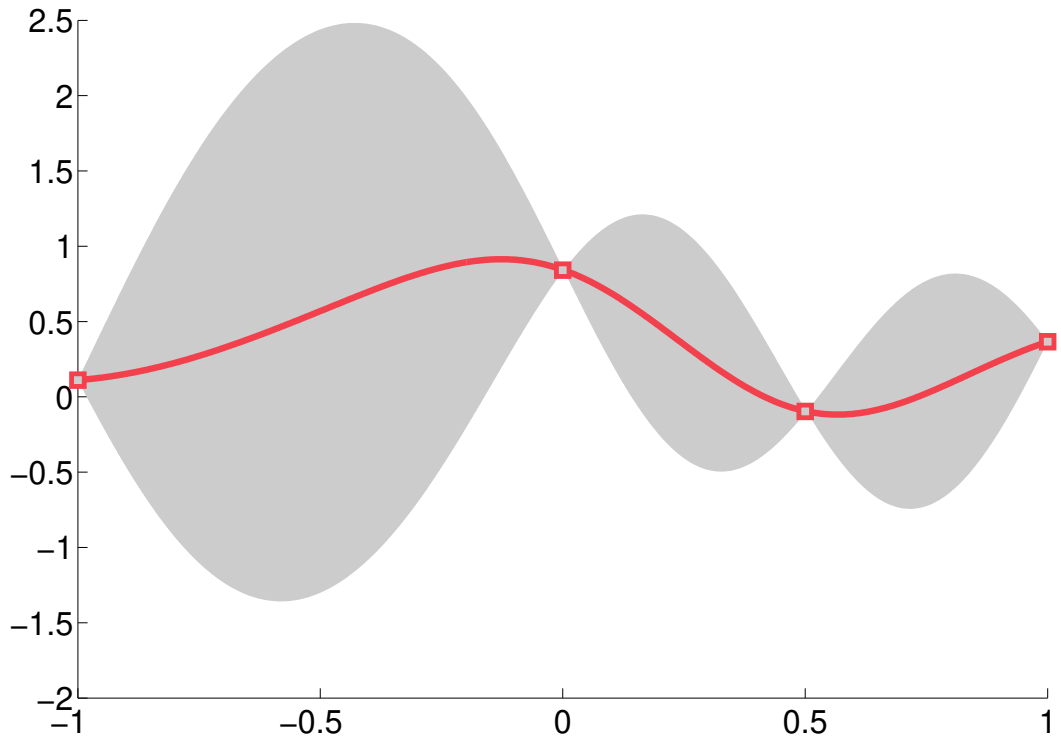
Large expected improvement in a small region, smaller expected improvement over a large region of the search domain $\rightarrow$ here, υ_n favors better exploration than ρ_n



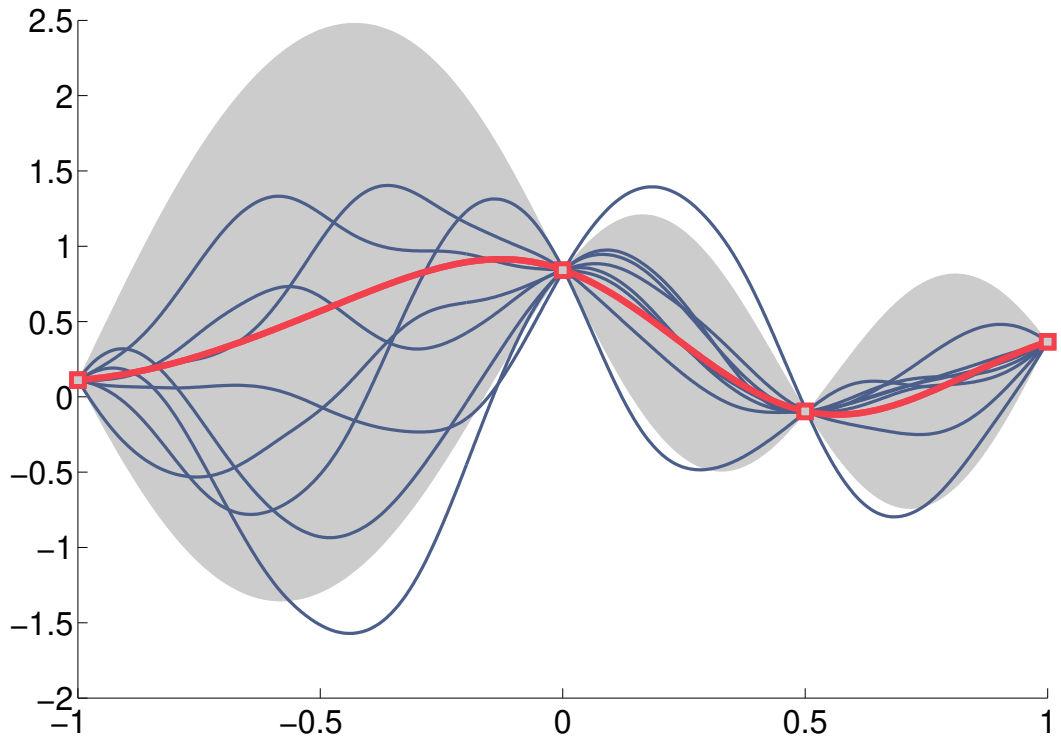
Entropy-based strategies

- Informational Approach to Global Optimization (IAGO)
  (Villemonteix, Walter & V., 2007–2008)
- Entropy Search for Information-Efficient Global Optimization
  (Hennig, Schuler, 2012)
- Predictive Entropy Search for Efficient Global Optimization of Black-box Functions
  (Hernandez-Lobato, Hoffman, Ghahramani, 2014)
- ...

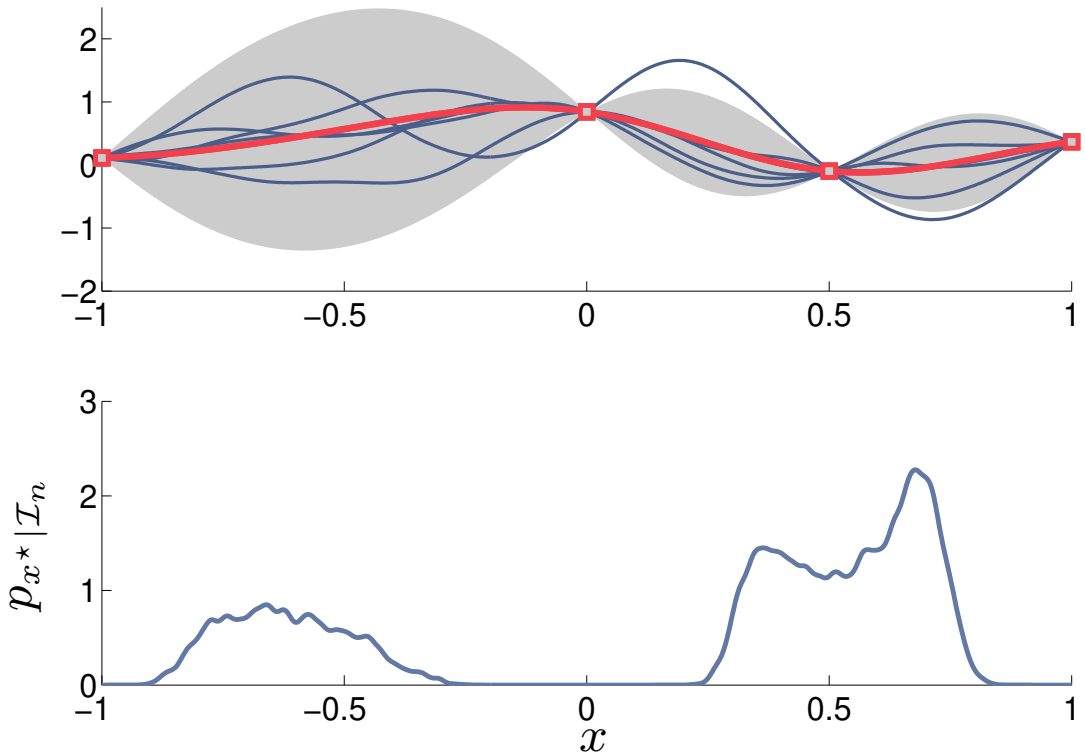
Empirical posterior density of the minimizer



Empirical posterior density of the minimizer



Empirical posterior density of the minimizer



- Assumption: search domain $\mathbb{X}$ is finite
- Define a loss function as the residual uncertainty about $x^\star$ measured using the Shannon entropy

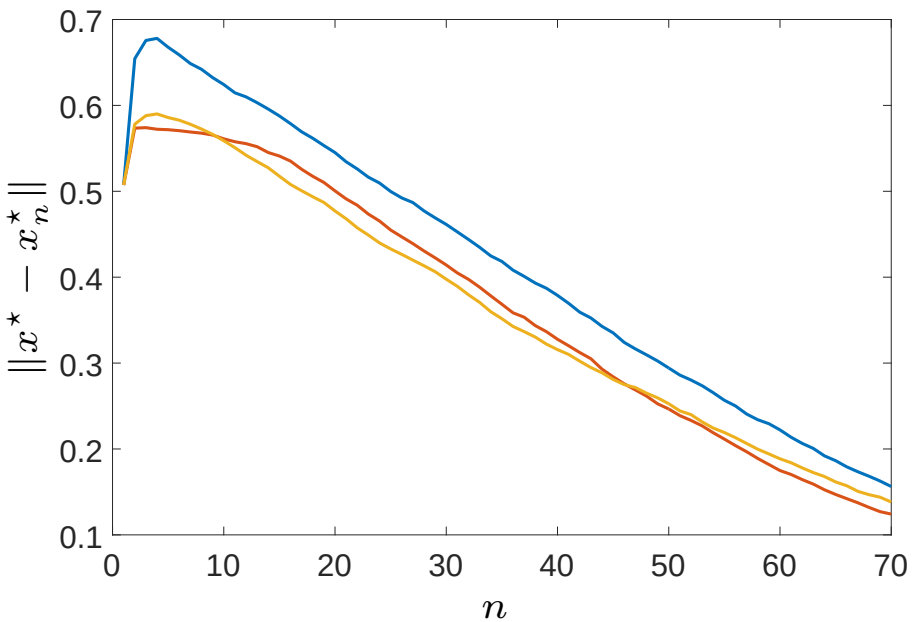
$$\varepsilon_n(\underline{X}, \xi) = H(x^\star; \mathcal{F}_n) = -\sum_{x \in \mathbb{X}} \mathsf{P}_n(x^\star = x) \log \mathsf{P}_n(x^\star = x)\,,$$

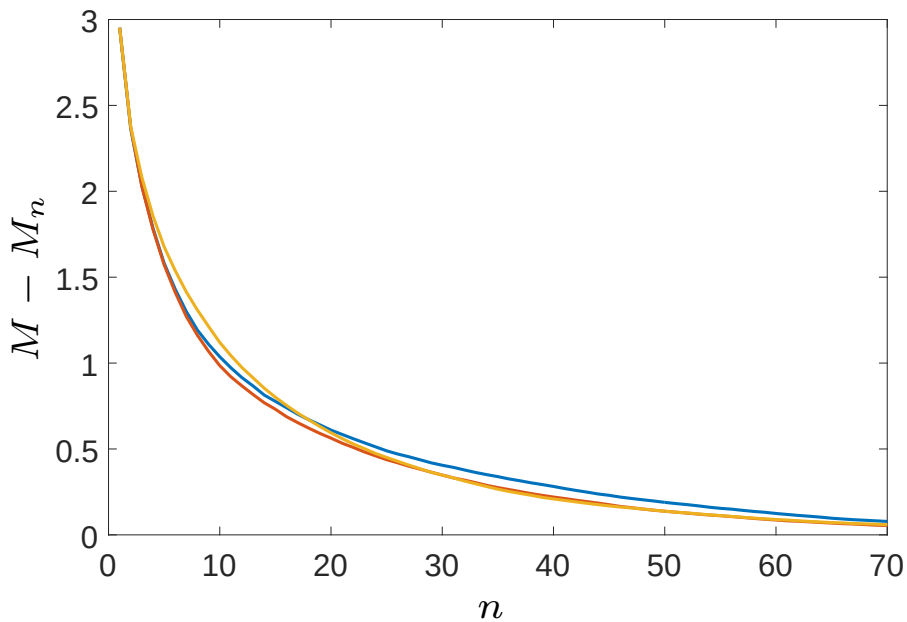
  with P_n the conditional distribution $\mathsf{P}_0(\,\cdot \mid \mathcal{F}_n)$
- SUR strategy:

$$X_{n+1} = \arg\min_{x \in \mathbb{X}} \mathsf{E}_n \left(\varepsilon_{n+1}(\underline{X}, \xi) \mid X_{n+1} = x \right)$$

Empirical comparison

Average errors using EI (blue), EI^2 (red), and IAGO (yellow), from 3000 sample paths of a GP on $\mathbb{R}^3$, with zero-mean and isotropic Matérn covariance function, simulated on a set of 1000 points in $[0,1]^3$.





Multi-objective optimization

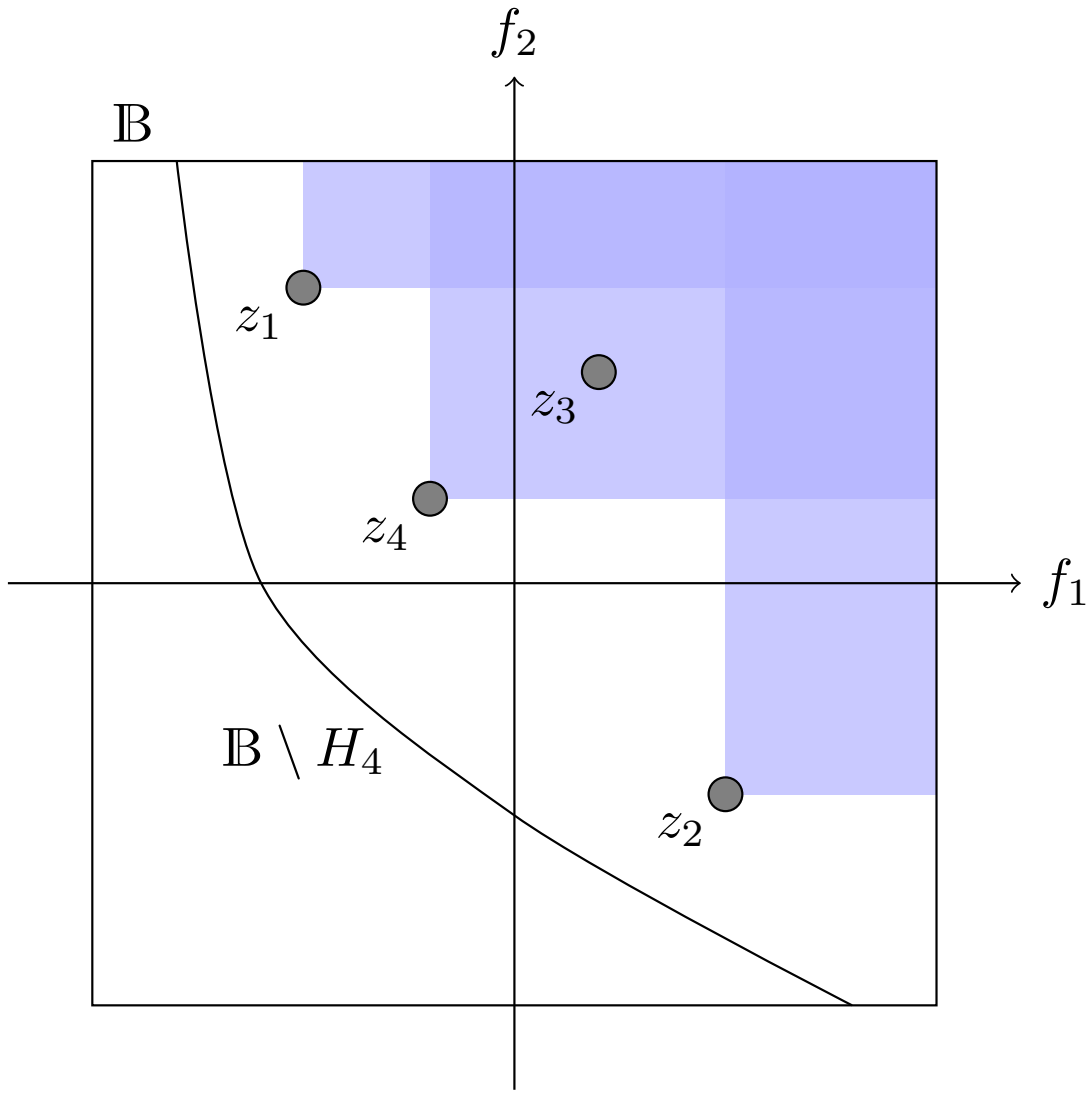
- Consider a set of objective functions $f_j : \mathbb{X} \to \mathbb{R}$, $j = 1, \ldots, p$, to be minimized
- Objective: build an approximation of the Pareto front

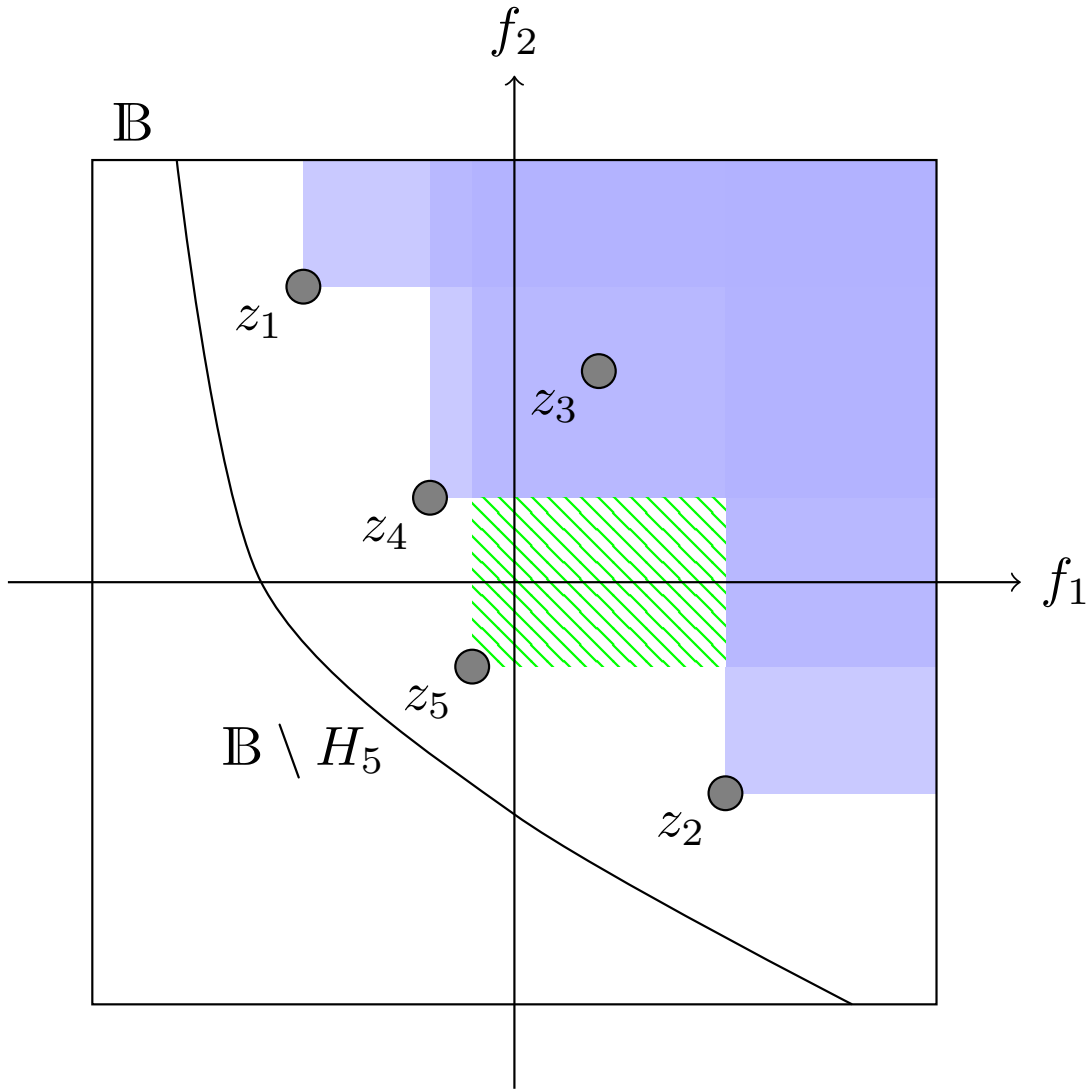
$$\Gamma = \{x \in \mathbb{X} : \nexists\, x' \in \mathbb{X} \text{ such that } \; f(x') \prec f(x)\}\,,$$

where $\prec$ stands for the Pareto domination rule defined by

$$z' = (z'_1, \ldots, z'_p) \prec z = (z_1, \ldots, z_p) \iff \begin{cases} \forall i \leq p, & z'_i \leq z_i\,, \\ \exists j \leq p, & z'_j < z_j\,. \end{cases}$$

- Define
  - $\mathbb{B} = \{z \in \mathbb{R}^p;\ z \leq z^{\mathsf{upp}}\}, z^{\mathsf{upp}} \in \mathbb{R}^p$
  - $H = \{z \in \mathbb{B}; \exists x \in \mathbb{X}, f(x) \prec z\}$
  - $H_n = \{z \in \mathbb{B}; \exists i \leq n,\ f(X_i) \prec z\}$


f_2
$\mathbb{B}$
z_1
z_3
z_4
f_1
$\mathbb{B} \setminus H_4$
z_2


f_2
$\mathbb{B}$
z_1
z_3
z_4
f_1
z_5
$\mathbb{B} \setminus H_5$
z_2

- Idea: use the volume of the non-dominated region as a loss function:

$$\varepsilon_n(\underline{X}, f) = |\mathbb{B} \setminus H_n| \,,$$

where

$$H_n = \{z \in \mathbb{B}; \exists i \leq n,\ f(X_i) \prec z\} \,,$$

- Define the improvement yielded by a new evaluation result

$$f(X_{n+1}) = (f_1(X_{n+1}), \ldots, f_p(X_{n+1}))$$

as the increase of the volume of the dominated region:

$$I_n\left(X_{n+1}\right) = |\mathbb{B} \setminus H_n| - |\mathbb{B} \setminus H_{n+1}| = |H_{n+1} \setminus H_n| = |H_{n+1}| - |H_n| \,,$$

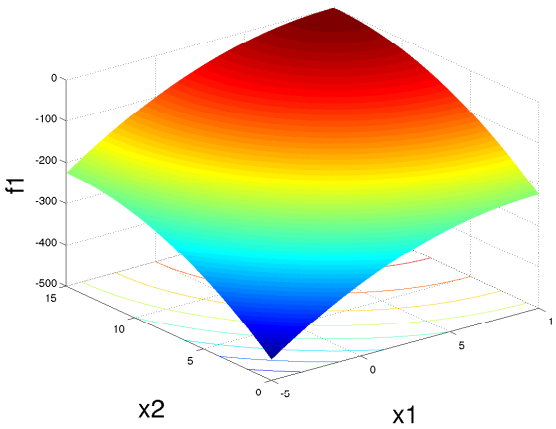
since $H_n \subset H_{n+1} \subset H$.

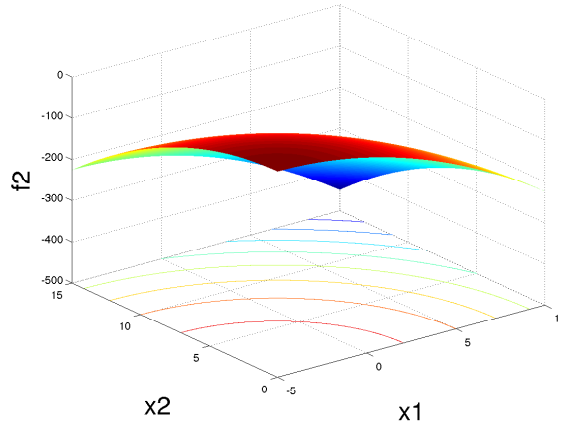
Expected hyper-volume improvement (EHVI)

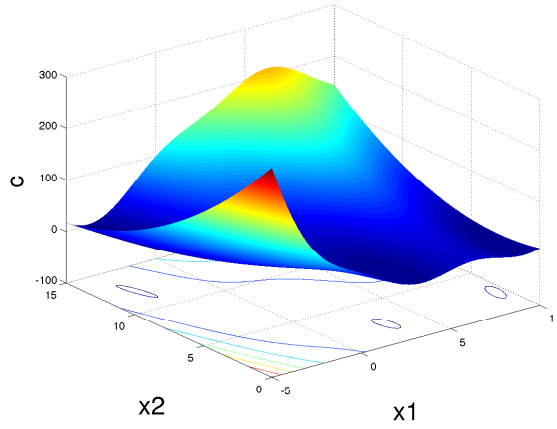
- Given a vector-valued Gaussian random process model $\xi = (\xi_1, \ldots, \xi_p)$ of $f = (f_1, \ldots, f_p)$, define a multi-objective EI criterion as

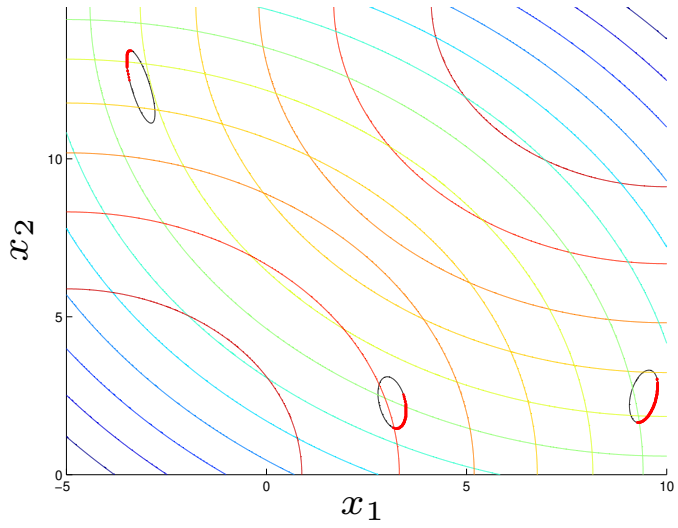
$$\begin{aligned}
\rho_n(x) &= \mathsf{E}_n\left(I_n(x)\right) \\
&= \mathsf{E}_n\left(\int_{\mathbb{B}\setminus H_n} \mathbb{1}_{\xi(x)\prec z}\,\mathrm{d}z\right) \\
&= \int_{\mathbb{B}\setminus H_n} \mathsf{E}_n\left(\mathbb{1}_{\xi(x)\prec z}\right)\,\mathrm{d}z \\
&= \int_{\mathbb{B}\setminus H_n} \mathsf{P}_n\left(\xi(x)\prec z\right)\,\mathrm{d}z\,,
\end{aligned}$$

- First proposed by Emmerich and coworkers (2005–2008)
- Extension to the constrained multi-objective setting (Feliot, Bect & V. 2017)









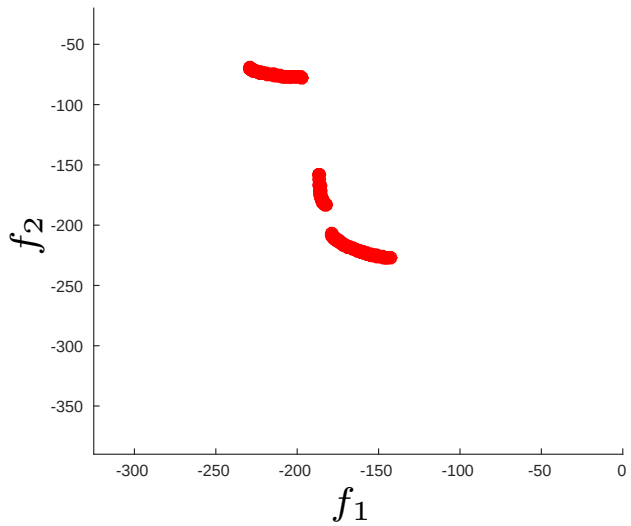
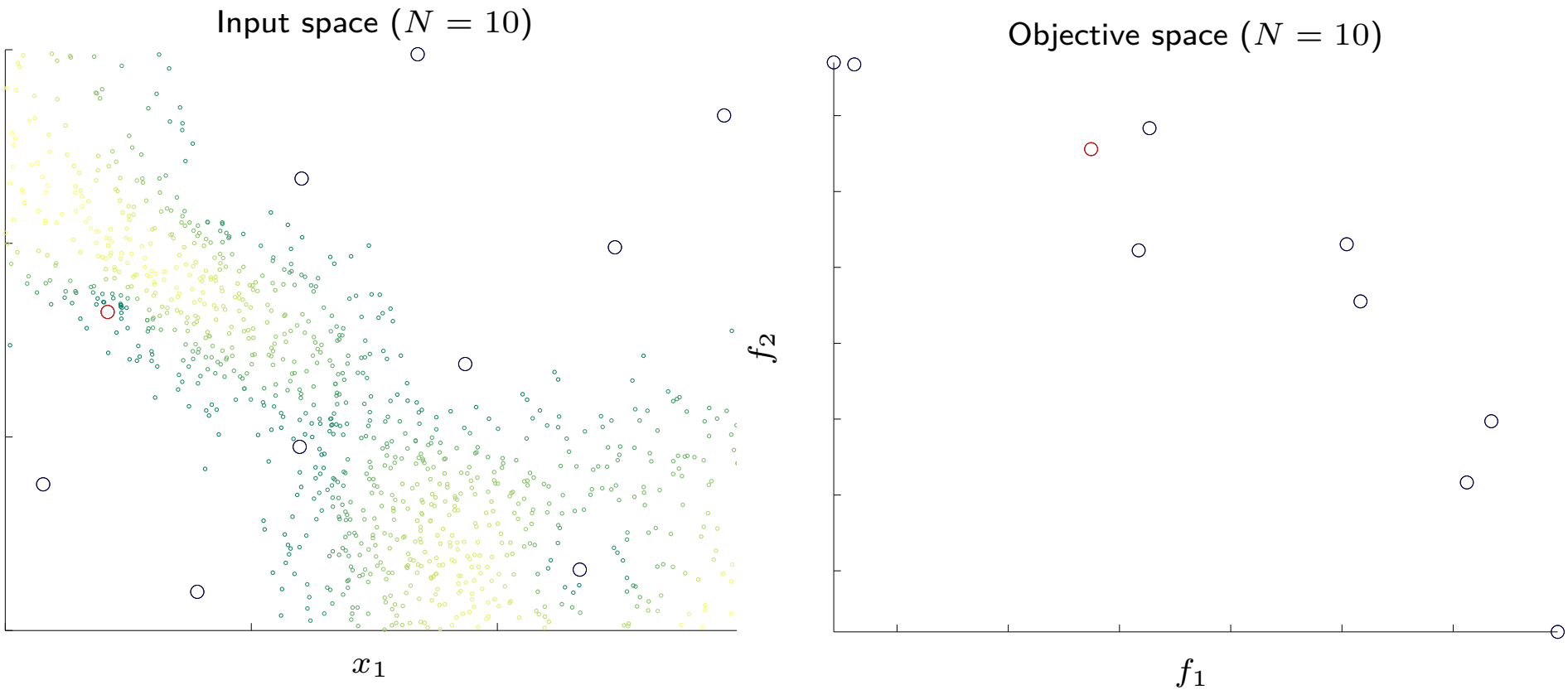
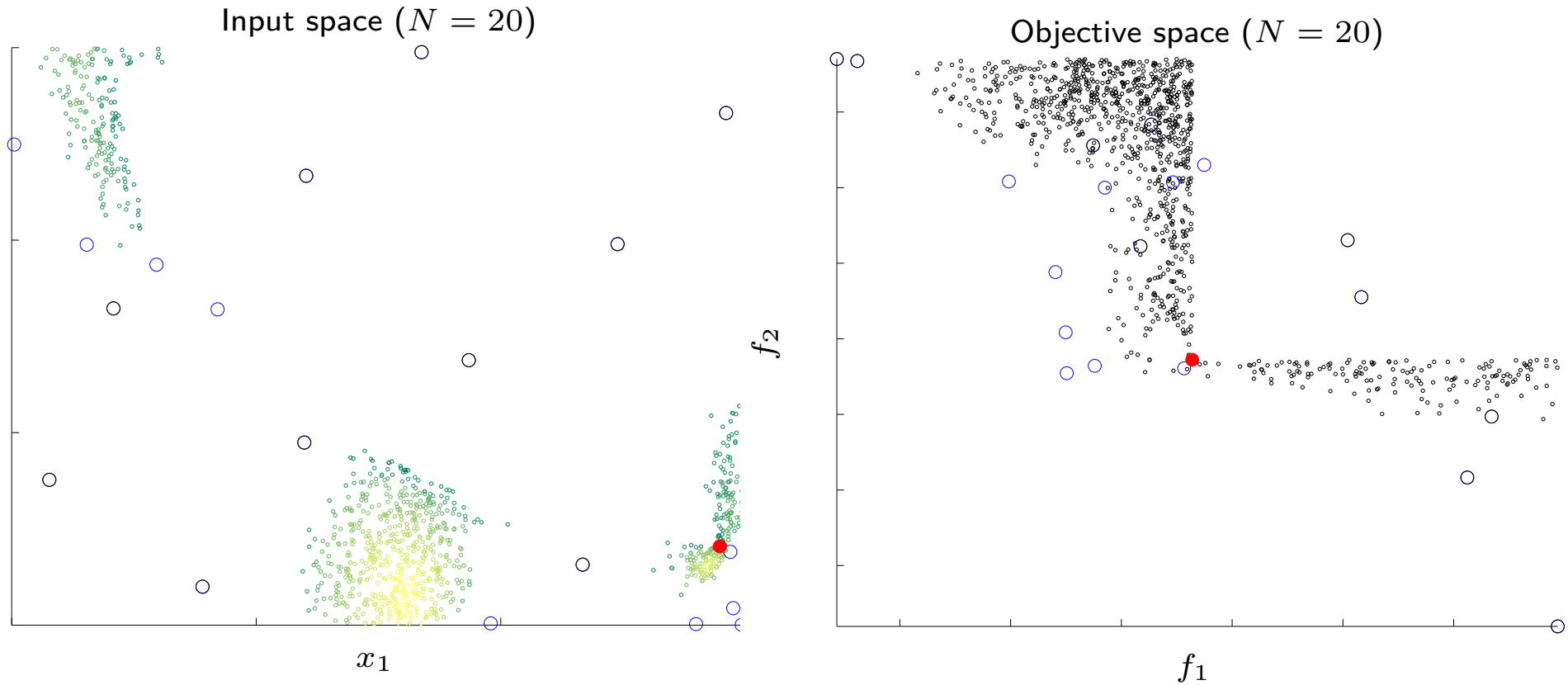


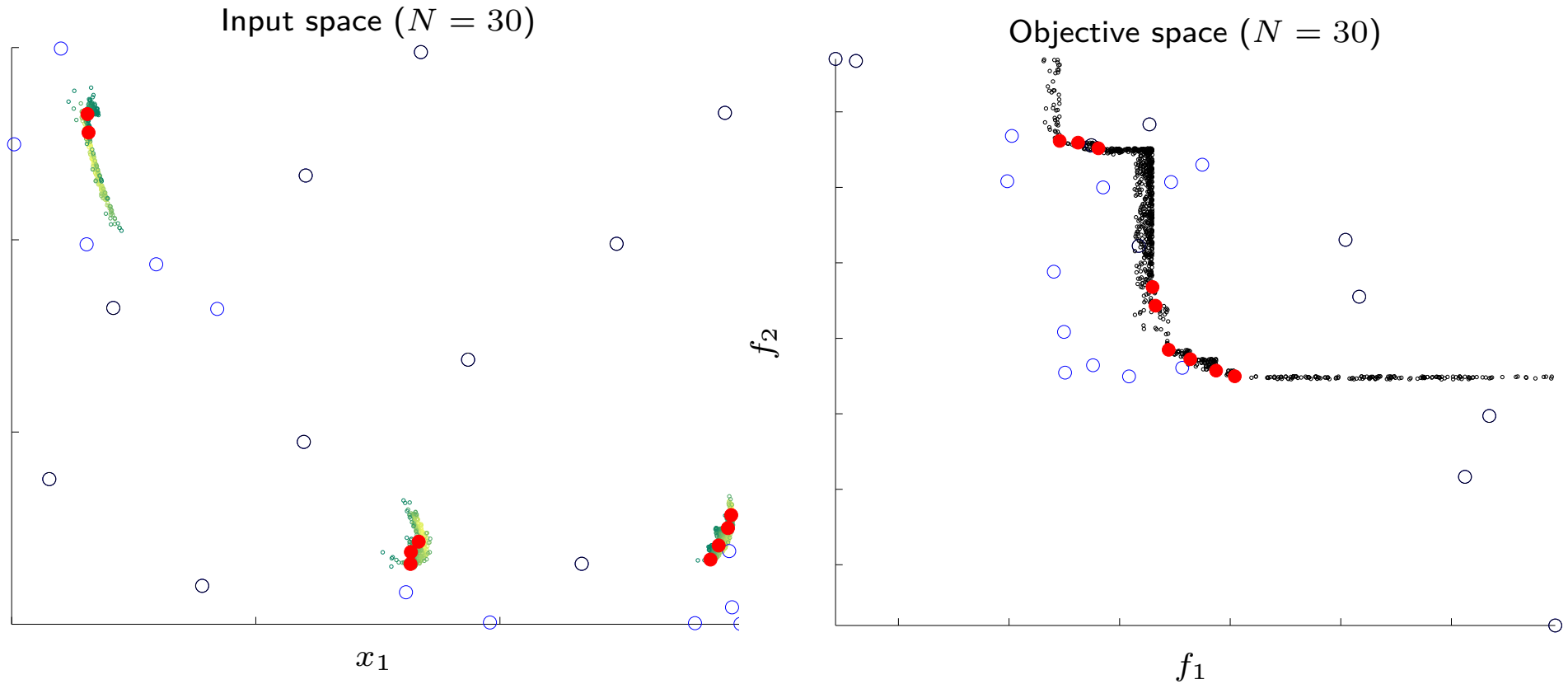
Illustration with $d = 2, p = 2, q = 1$



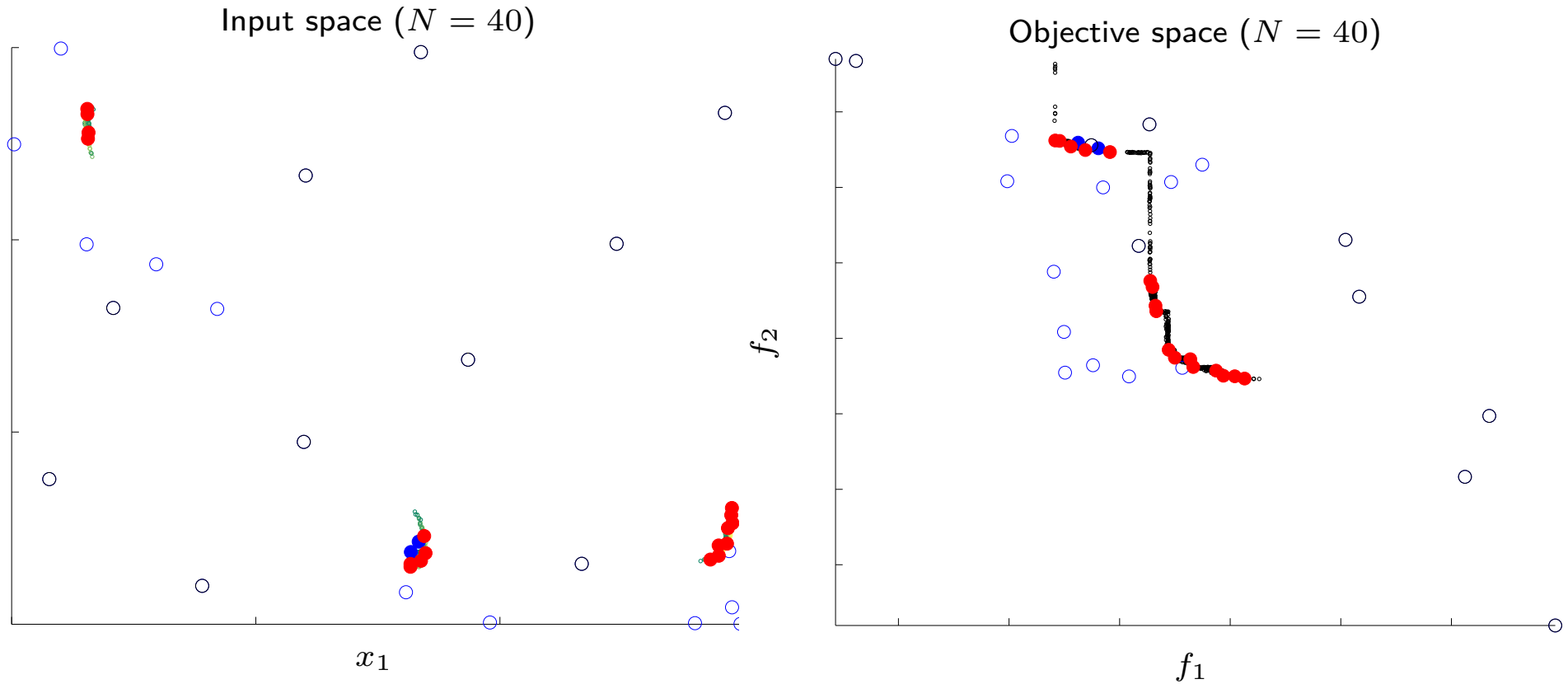
BMOO with Sequential Monte Carlo (Feliot, Bect & Vazquez 2017)



BMOO with Sequential Monte Carlo (Feliot, Bect & Vazquez 2017)



BMOO with Sequential Monte Carlo (Feliot, Bect & Vazquez 2017)



BMOO with Sequential Monte Carlo (Feliot, Bect & Vazquez 2017)

5 Concluding remarks

- Bayesian optimization, along with Bayesian techniques for the estimation of excursion sets (a closely related problem), have been prominent topics in recent years (2015–today)
- Numerous publications in the ML and UQ communities on applications, sampling criteria, high dimensionality...
- Software is available: scikit-learn, BoTorch...
- However, I strongly advise against using these implementations in black-box mode: performance depends on the GP model!
- Please feel free to contact me if you want to discuss about BO