MDIS Fall School 2017

Introduction to Gaussian Process Surrogate models

Clermont-Ferrand, Besse-en-Chandesse, 16th of October 2017

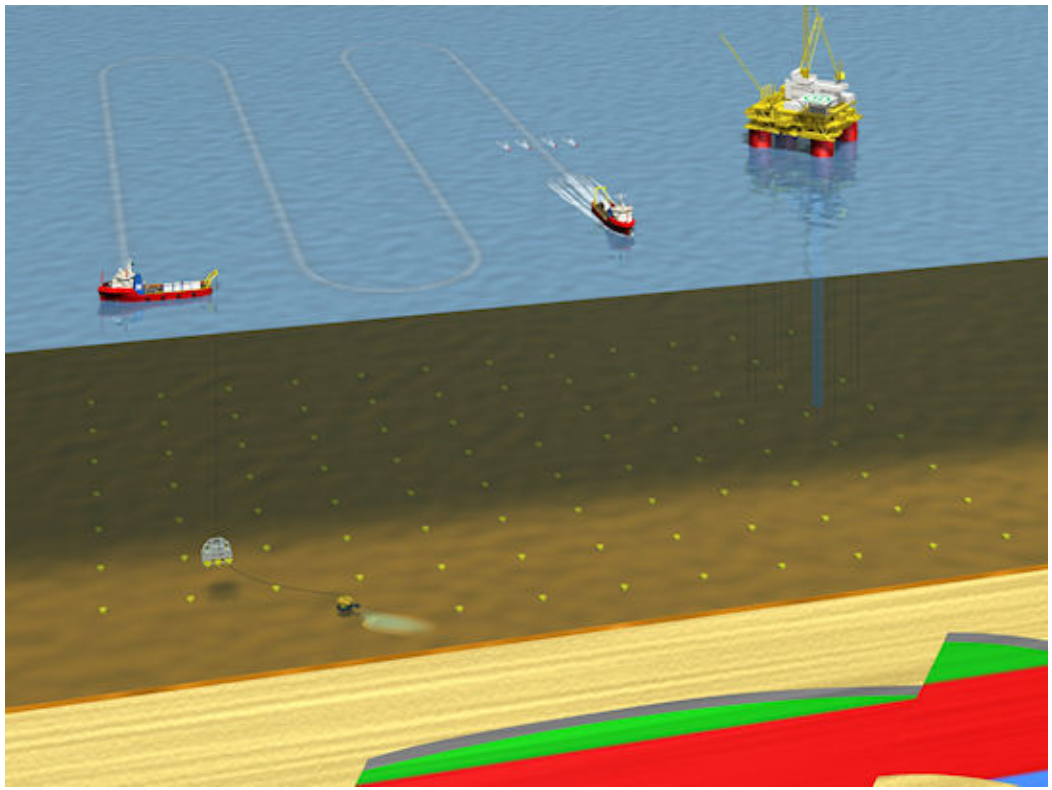
Nicolas Durrande, PROWLER.io (nicolas@prowler.io)
Rodolphe Le Riche, CNRS – Mines St-Étienne – LIMOS (leriche@emse.fr)

Why are statistical models relevant in engineering?

There is a wide variety of situations where getting data about a system performance can be extremely expensive.

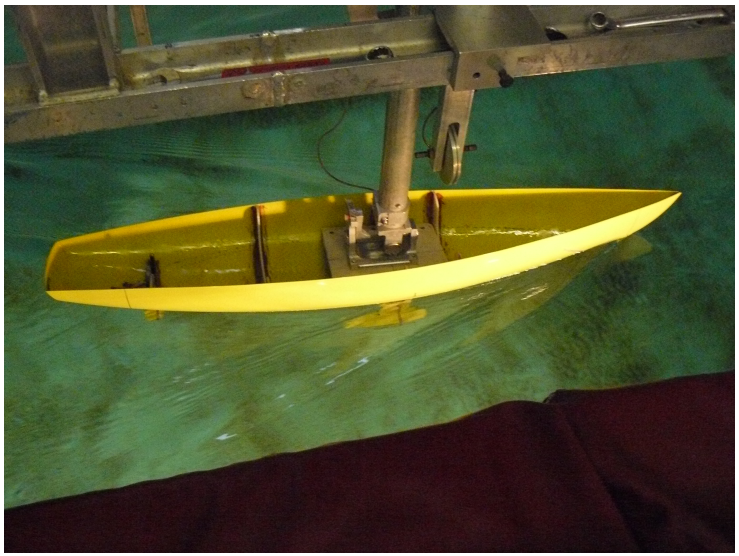
- real world experiments
- destructive tests
- prototyping
- numerical experiments

Example: real world experiments



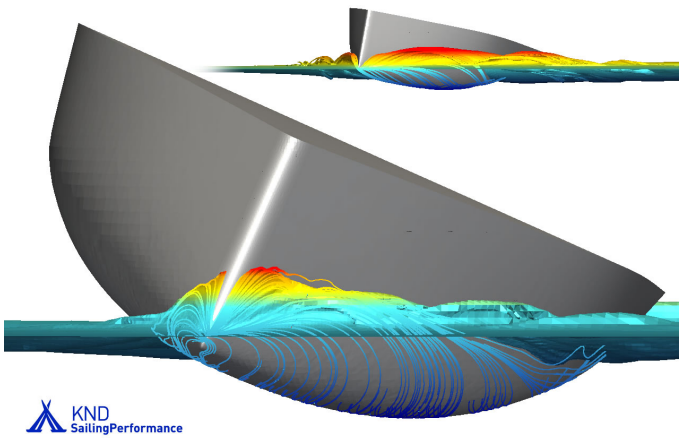
Example: Destructive tests

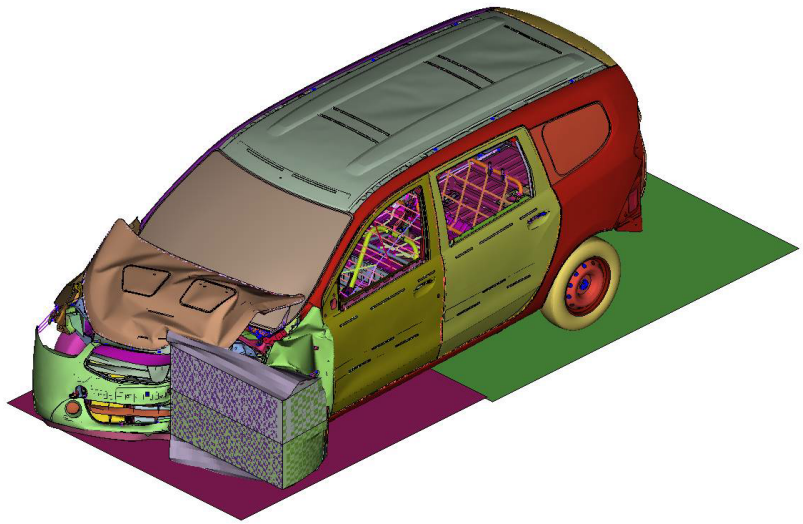
Example: Prototyping of a boat shape



Knowing the drag for a given design requires costly experiments

Example: Numerical experiments





Numerical experiments are less expensive but can be very time consuming!

In all these cases, the variable of interest can be seen as a function of the input parameters

$$y = f(x).$$

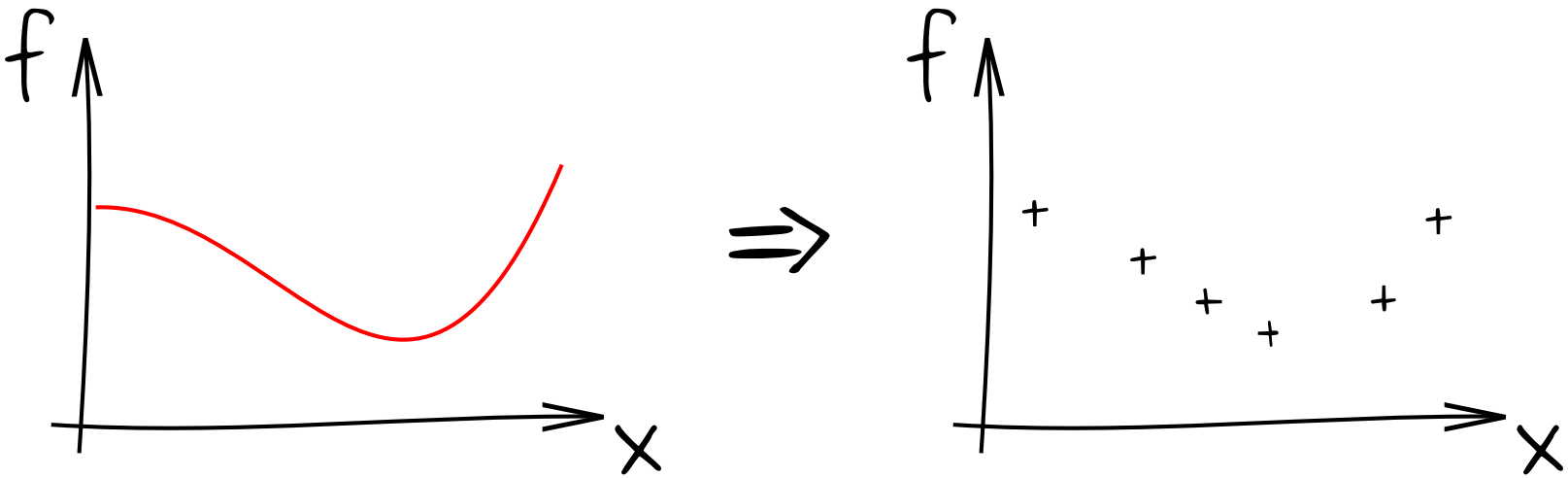
where f is a **costly to evaluate function**.

In the following, we will assume that

- $x \in \mathbb{R}^d$: There are many input parameters
- $y \in \mathbb{R}$: The output is a scalar.

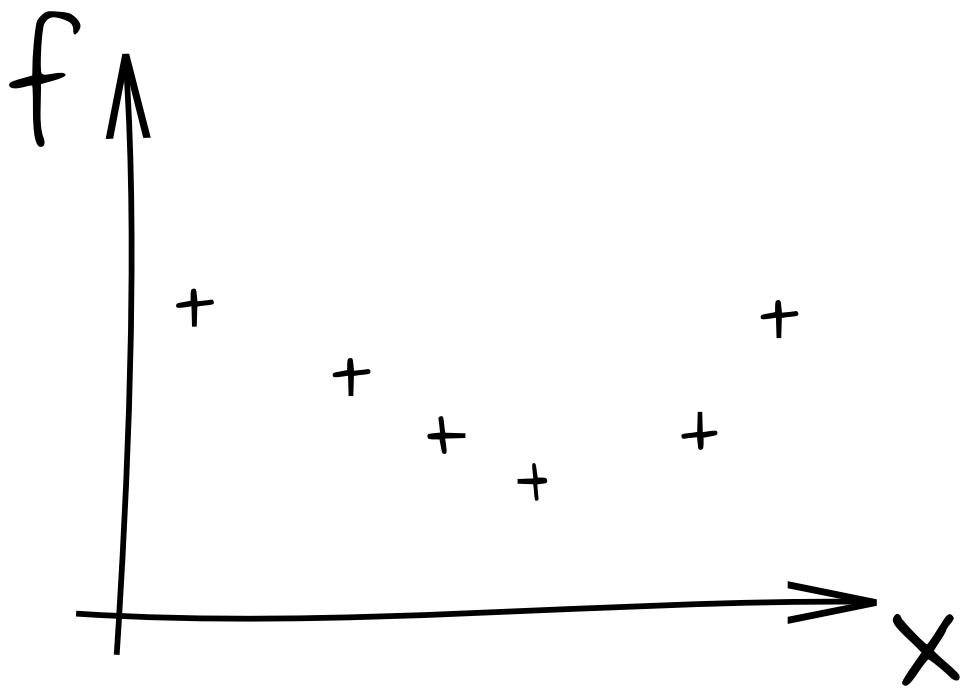
The fact that f is **costly to evaluate** changes a lot of things...

1. Representing the function is not possible...



The fact that f is **costly to evaluate** changes a lot of things...

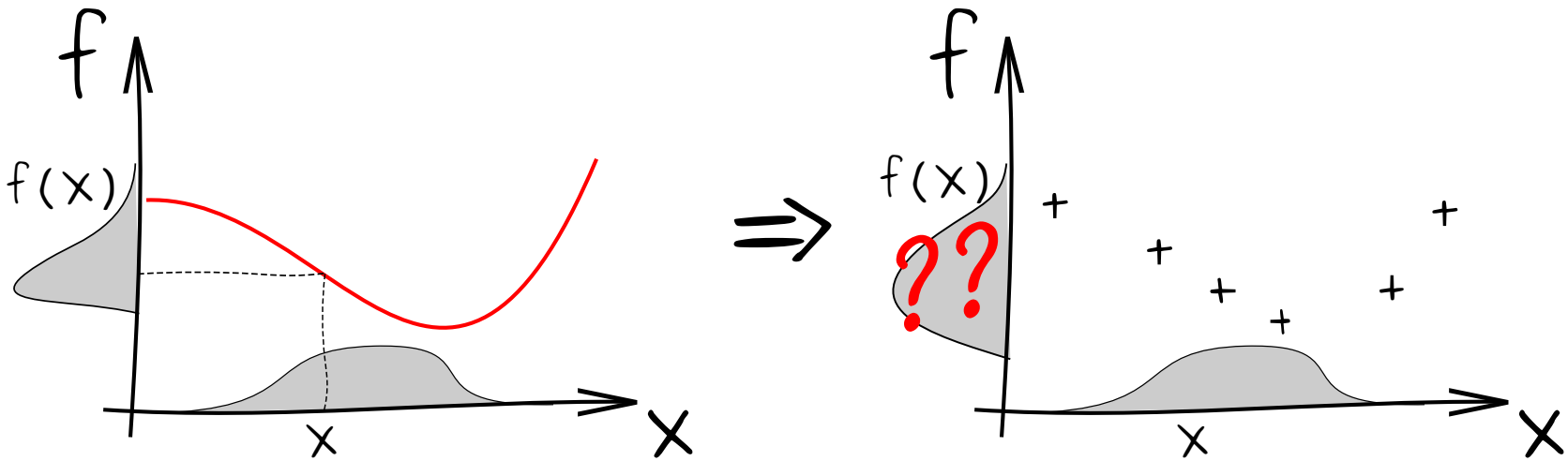
2. Computing integrals is not possible...



What is the mean value of f?

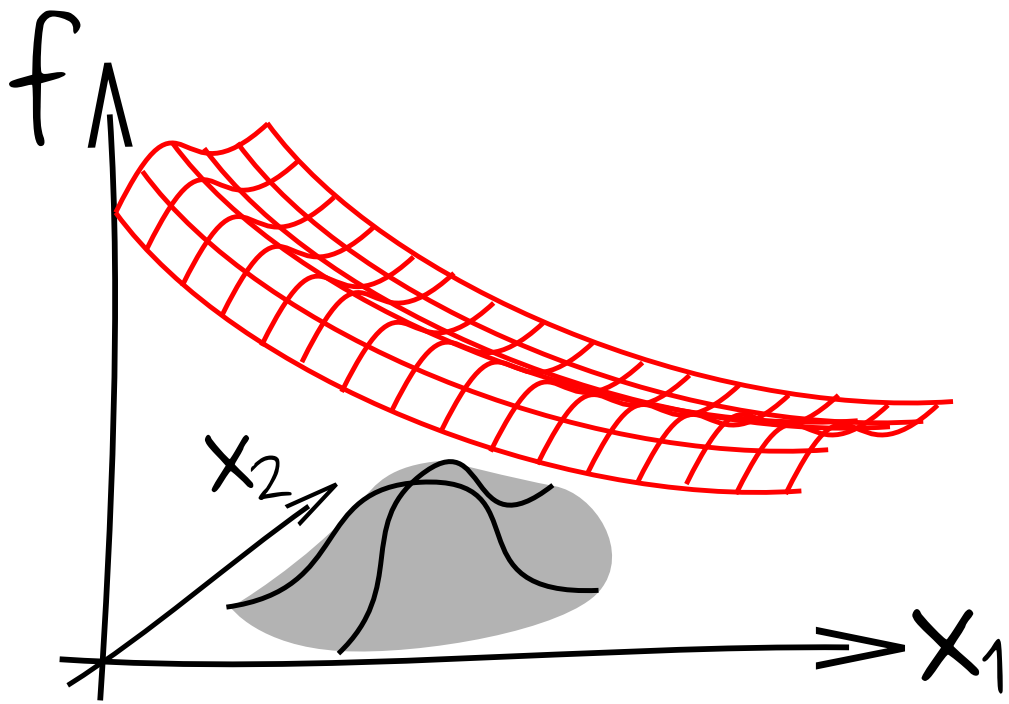
The fact that f is **costly to evaluate** changes a lot of things...

3. Uncertainty propagation is not possible...



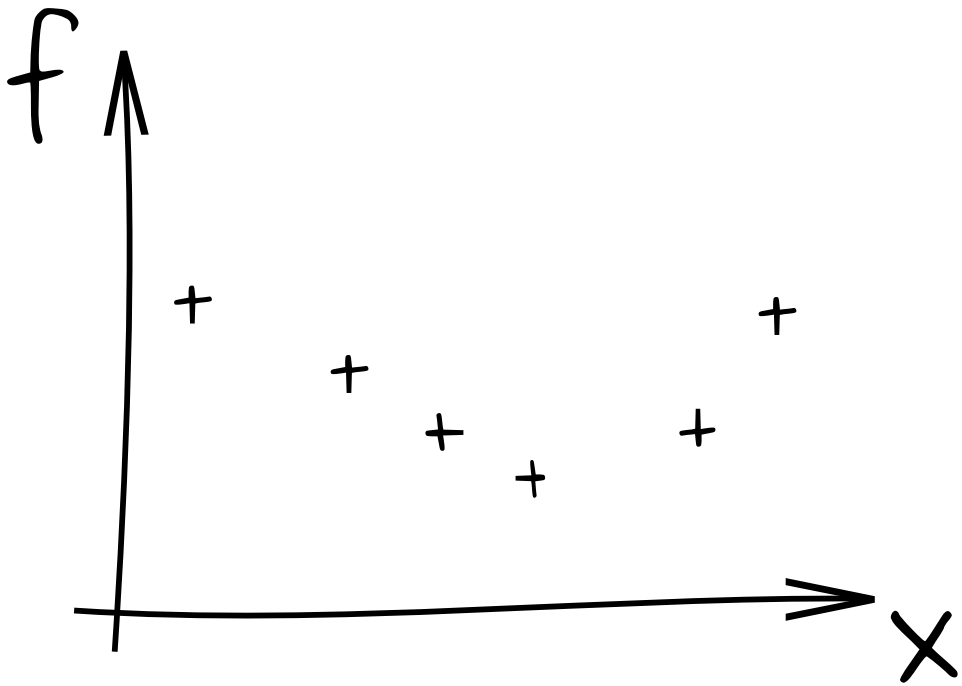
The fact that f is **costly to evaluate** changes a lot of things...

4. Sensitivity analysis is not possible...



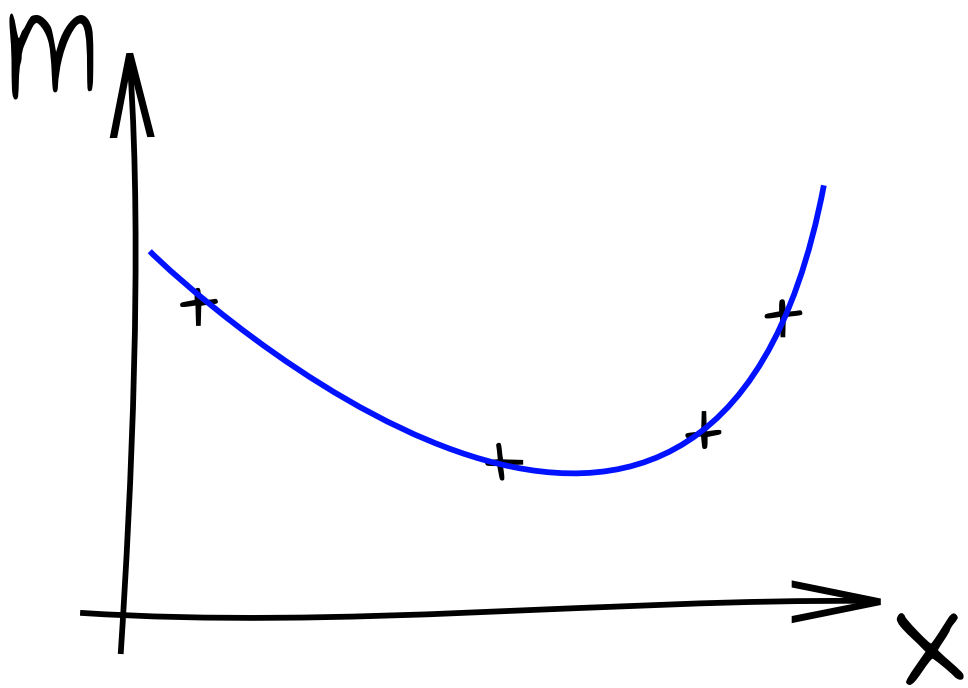
The fact that f is **costly to evaluate** changes a lot of things...

5. Optimisation is also tricky...



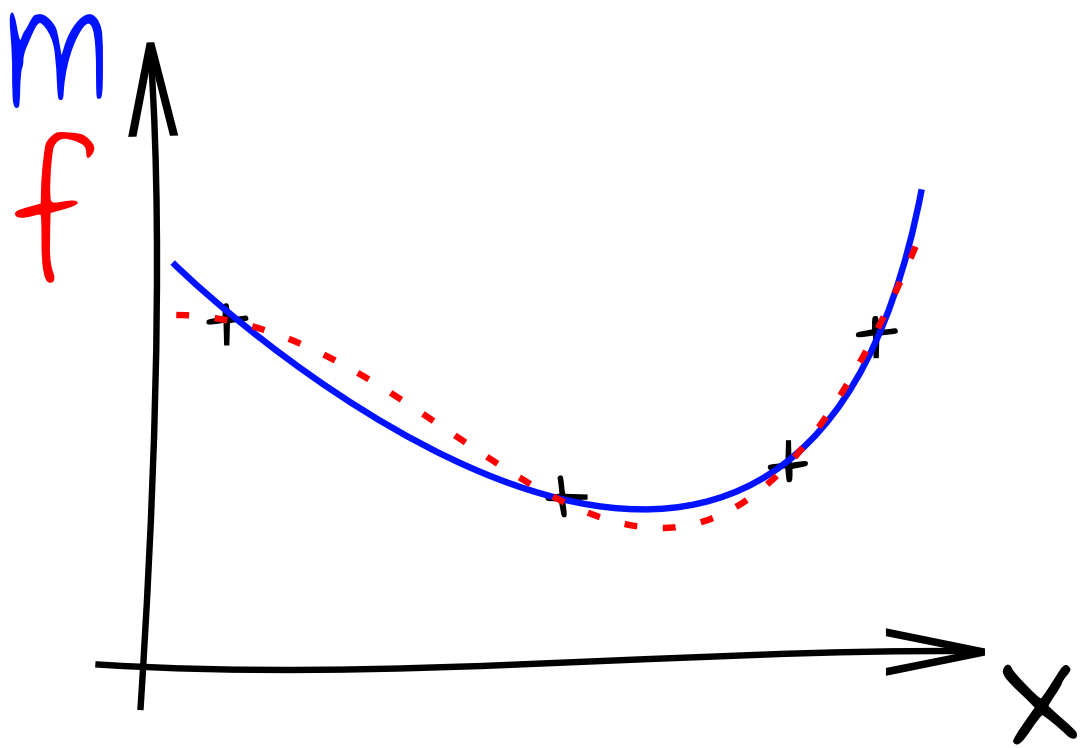
Statistical models

The principle of statistical modelling is to use the data to build a mathematical approximation of the function.

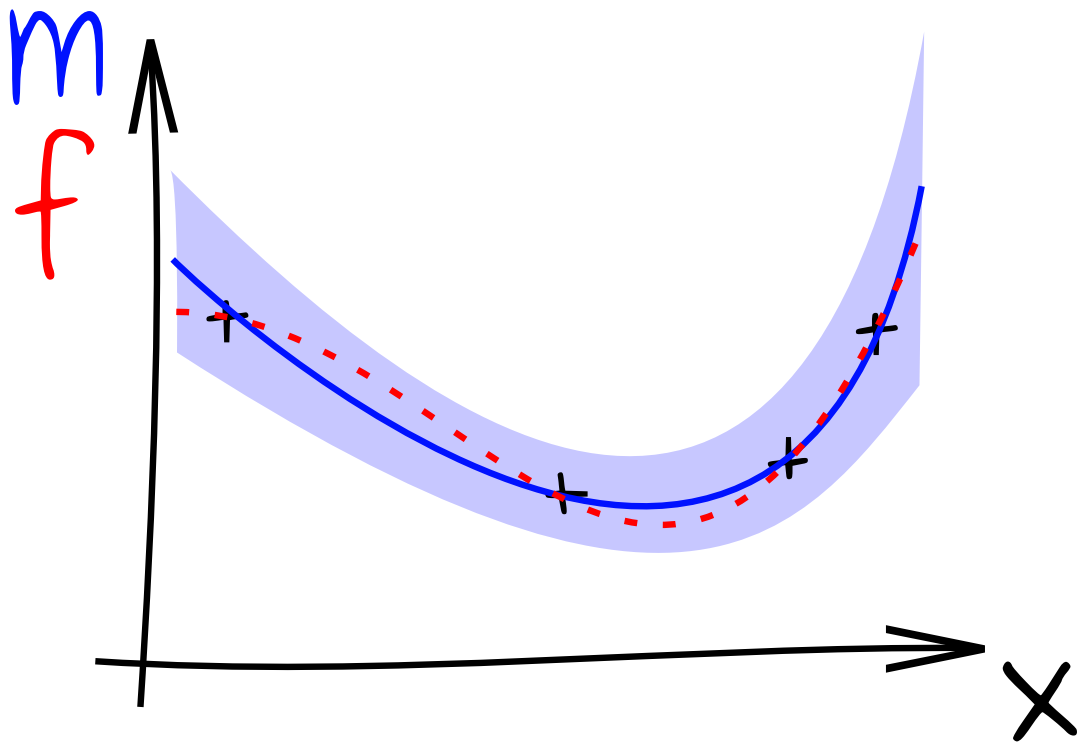


The model can then be used to answer all previous questions

Of course, there is a difference between f and m...



Why **statistical models**?
We want to be able to quantify the model error:



The confidence intervals can be used to obtain a **measure of uncertainty on the value of interest**.

In the sequel, we will use the following notations :

- The set of observation points will be represented by a $n \times d$ matrix $X = (X_1, ..., X_n)^t$
- The vector of observations will be denoted by F : $F_i = f(X_i)$ (or $F = f(X)$).

We will now discuss two types of statistical models:

- Linear regression
- Gaussian process regression

Linear Regression

Linear regression is probably the most commonly used statistical model.

Given a set of basis functions $B = (b_0, \ldots, b_p)$, we assume that the observations come from the probabilistic model

$$F = B(X)\beta + \varepsilon \qquad \left(\text{i.e. } F_i = \sum_{k=1}^{p} \beta_k b_k(X_i) + \varepsilon_i \right)$$

where the vector β is unknown, ε_i are independent and identically distributed and

$$B(X)_{(i,j)} = b_j(X_i)$$

If we consider a model of the form

$$m(x) = B(x)\hat{\beta}$$

the prediction error (Residual Sum of Square) is given by

$$RSS = (B(X)\hat{\beta} - F)^t(\hat{\beta}B(X) - F) \qquad \left(\text{i.e. } \sum_{k=1}^{n}(B(X_i)\hat{\beta} - F_i)^2\right)$$

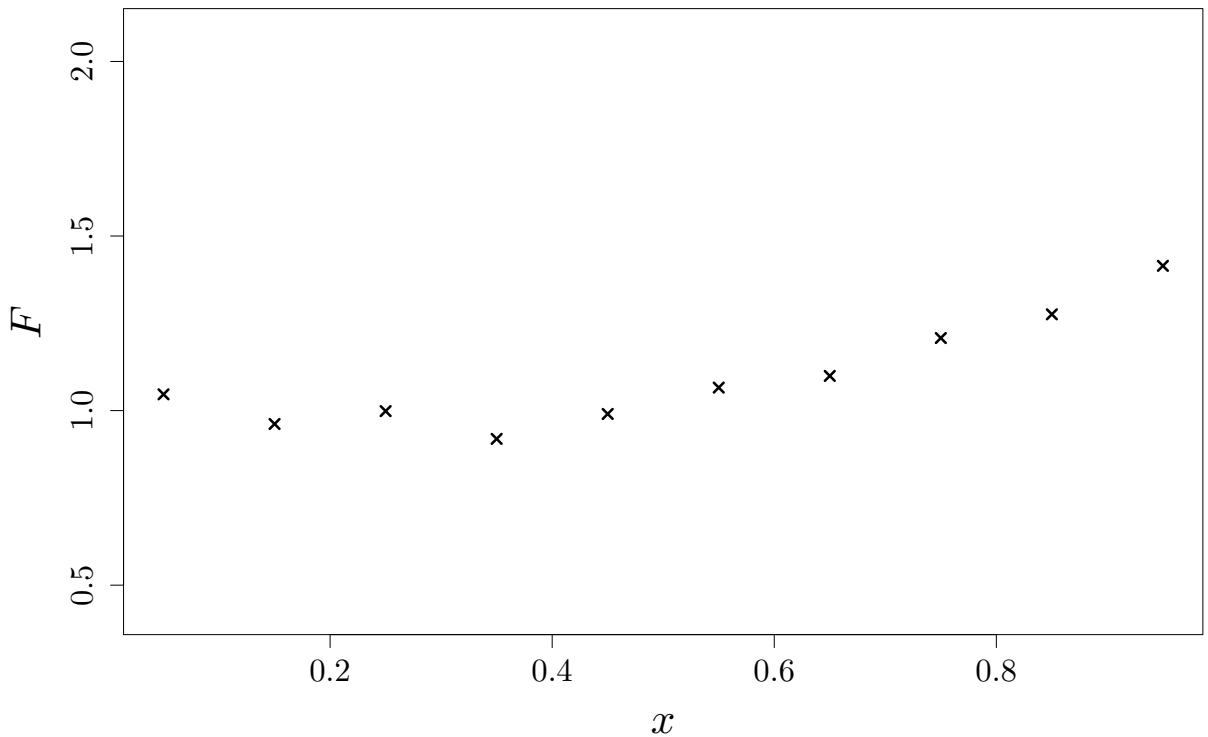
Finding the optimal value of $\hat{\beta}$ means minimizing a quadratic form. This can be done analytically and we obtain $\hat{\beta} = (B(X)^t B(X))^{-1} B(X)^t F$.

The associated linear regression model is thus

$$m(x) = B(x)(B(X)^t B(X))^{-1} B(X)^t F.$$

Example

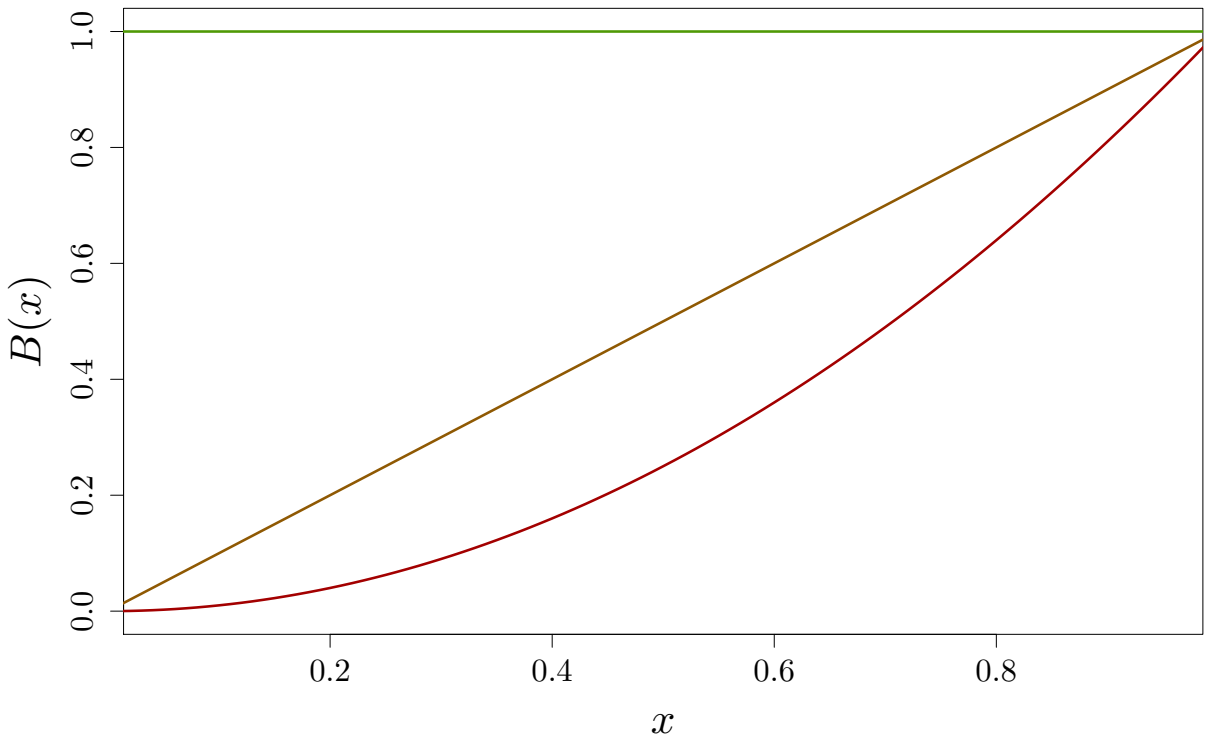
If we consider the following observations:



Example

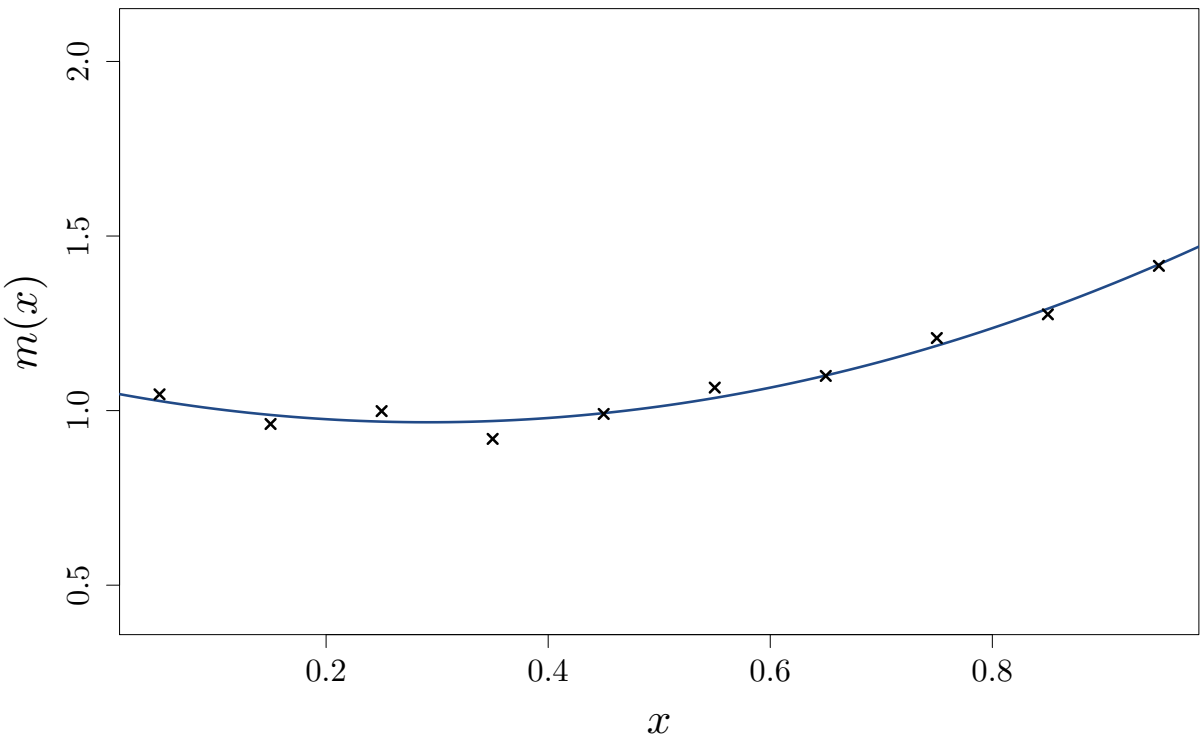
and a set of 3 basis functions:

$$b_0(x) = 1, \; b_1(x) = x, \; b_2(x) = x^2$$



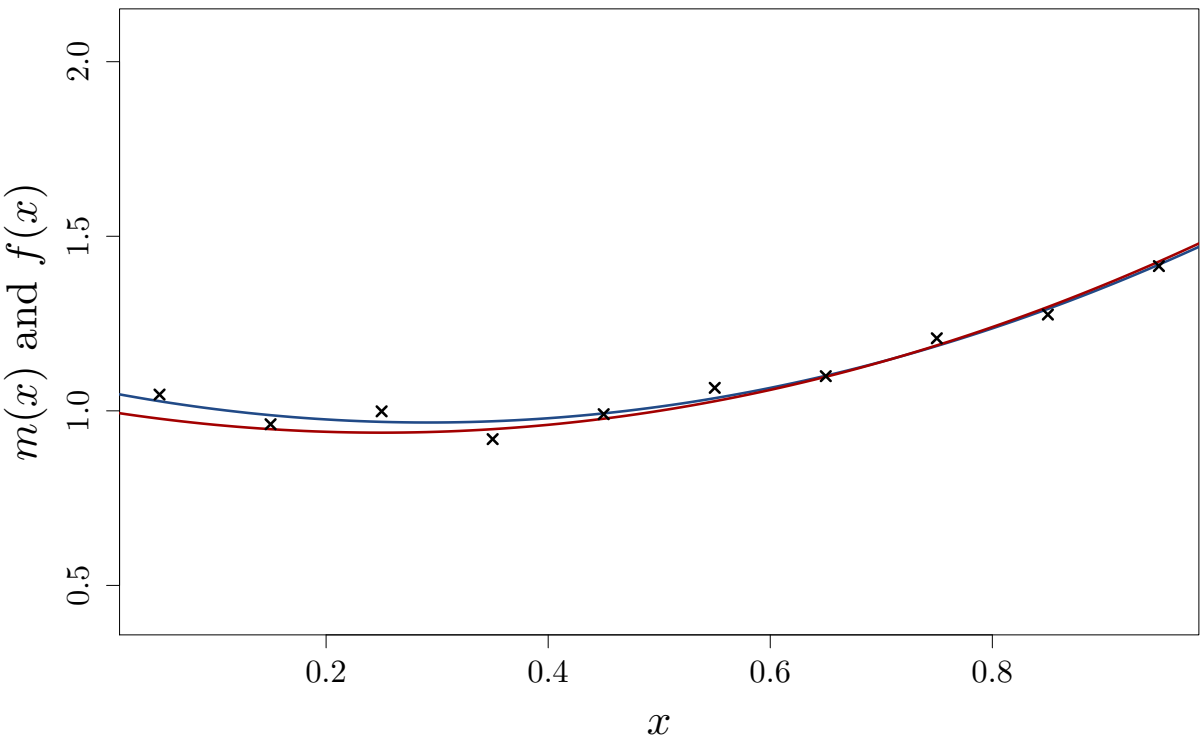
Example

We obtain $\hat{\beta} = (1.06, -0.61, 1.04)$ and the model is:



Example

There is of course an error between the true generative function and the model



Can this error be quantified?

The initial assumption is $F = B(X)\beta + \varepsilon$ and we have computed an estimator of β:

$$\hat{\beta} = (B(X)^t B(X))^{-1} B(X)^t F.$$

$\hat{\beta}$ can thus be seen as a sample from the random variable:

$$\hat{\beta} = (B(X)^t B(X))^{-1} B(X)^t (B(X)\beta + \varepsilon).$$

What about the distribution of $\hat{\beta}$?

The initial assumption is $F = B(X)\beta + \varepsilon$ and we have computed an estimator of β:

$$\hat{\beta} = (B(X)^t B(X))^{-1} B(X)^t F.$$

$\hat{\beta}$ can thus be seen as a sample from the random variable:

$$\hat{\beta} = (B(X)^t B(X))^{-1} B(X)^t (B(X)\beta + \varepsilon).$$

What about the distribution of $\hat{\beta}$?

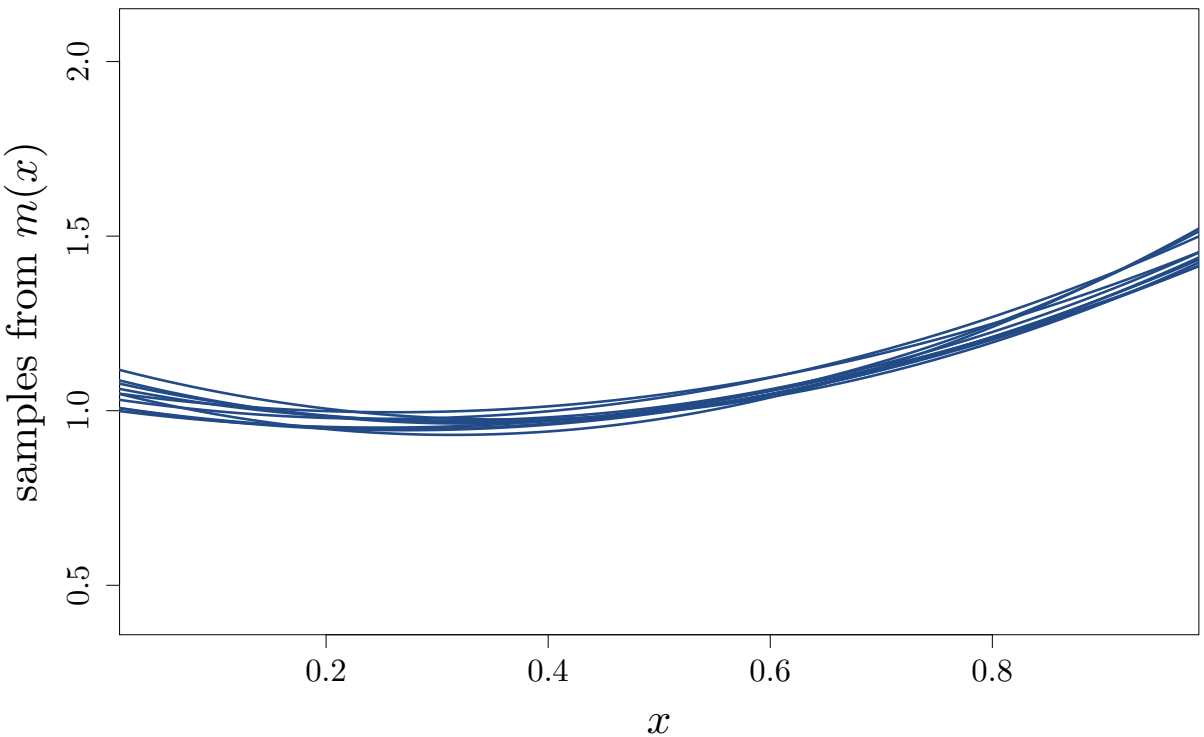
- Its expectation is $\beta \Rightarrow$ The estimator is unbiased
- Its covariance matrix is

$$(B(X)^t B(X))^{-1} B(X)^t \operatorname{cov}[\varepsilon, \varepsilon^t] B(X)(B(X)^t B(X))^{-1}$$

- If ε is multivariate normal, then $\hat{\beta}$ is also multivariate normal (not necessarily i.i.d.).

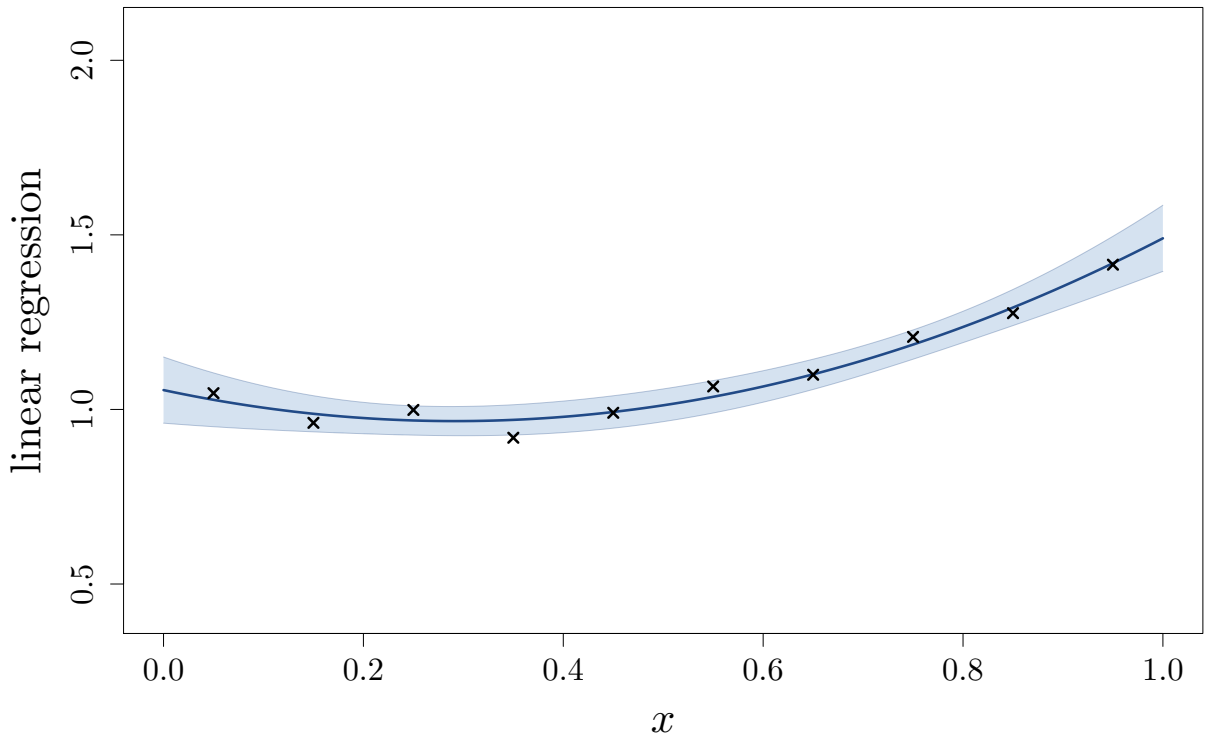
Sampling in the distribution of $\hat{\beta}$ gives us a large variety of models which represent the uncertainty about our estimation:

Back to the example



Back to the example

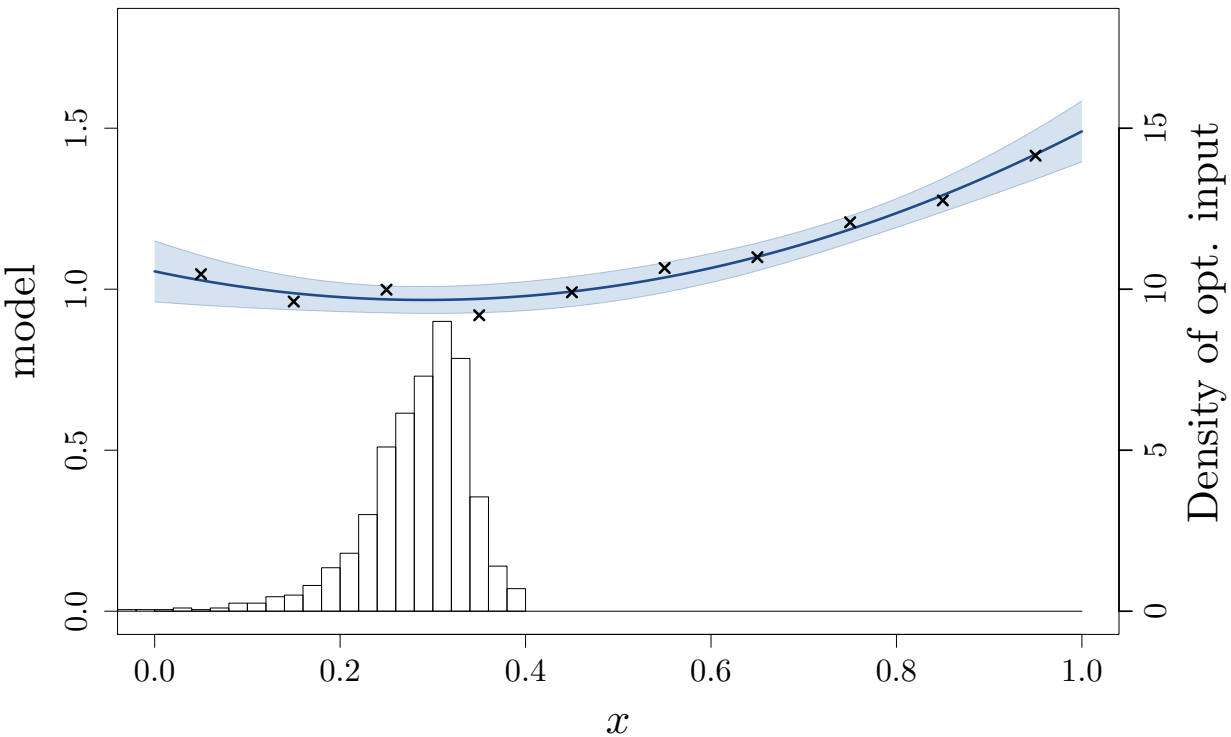
The previous picture can be summarized by showing the mean of m and 95% confidence intervals



Knowing the uncertainty on the model allows to compute an uncertainty on the quantity of interest.

Back to the example

For example, if we are interested in the value $x^\star$ minimizing $f(x)$:

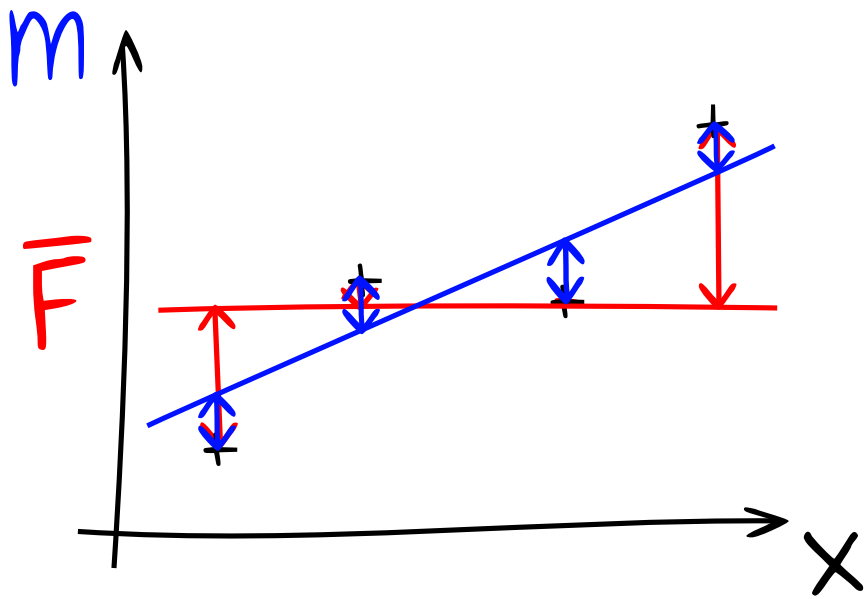


The expectation of $x^\star$ **is not** the input minimizing $m(x)$.

Model validation is always of upper importance.

The goodness of fit can be measured by the **coefficient of determination**:

$$R^2 = 1 - \frac{\text{var}[\textit{prediction errors}]}{\text{var}[\textit{data}]} = 1 - \frac{\sum_i (F_i - m(X_i))^2}{\sum_i (F_i - \textit{mean}(F))^2}$$



Be careful! A good R^2 (ie close to one) does not necessarily mean that the model is good.

We've seen that the law of $\hat{\beta}$ is multivariate Gaussian, with its covariance depending on ε:

$$(B(X)^t B(X))^{-1} B(X)^t \operatorname{cov}[\varepsilon, \varepsilon^t] B(X)(B(X)^t B(X))^{-1}$$

Ordinary regression: suppose now $\operatorname{cov}[\varepsilon, \varepsilon^t] = \sigma^2 I_d$ then

$$\operatorname{cov}[\hat{\beta}, \hat{\beta}^t] = \sigma^2 (B(X)^t B(X))^{-1}$$

and from this, we now that

$$\frac{\hat{\beta}_i}{\sqrt{\hat{\sigma}^2 (B(X)^t B(X))^{-1}_{(i,i)}}} \text{ is student}$$

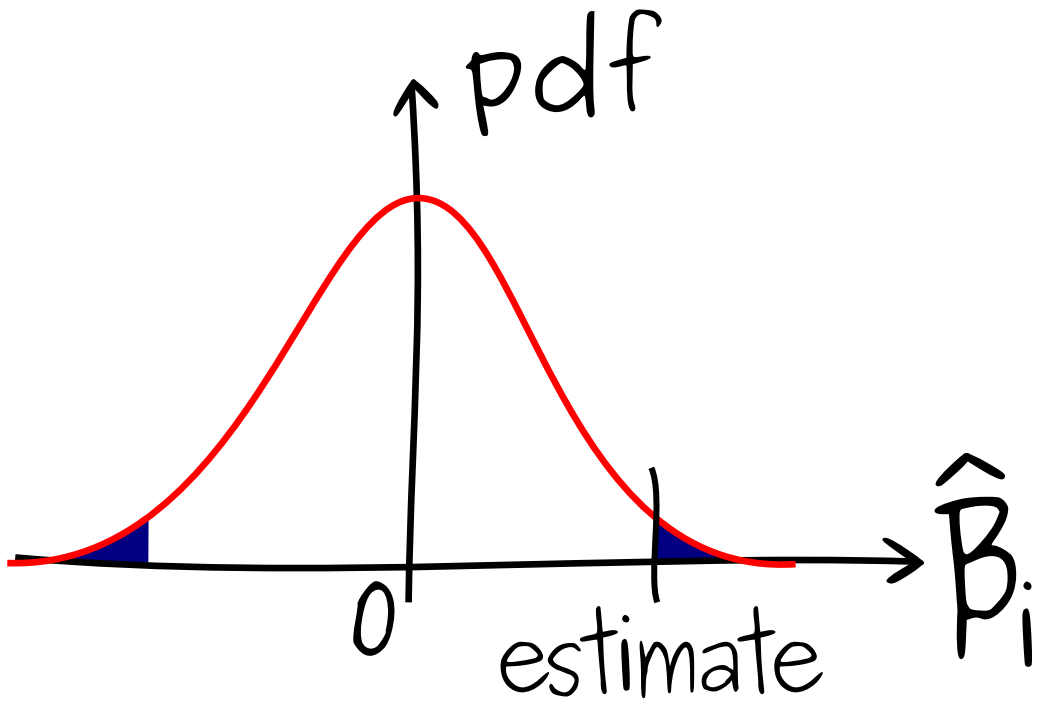
with the standard variance estimator

$$\hat{\sigma}^2 = \frac{1}{n-p} \|F - m(X)\|^2$$

The influence of one basis function can be tested using **p-values**:

H_0: The basis function b_i has no influence (i.e. $\beta_i = 0$)

p-values: probability of observing a larger estimate



The smaller the p-value is, the less $\hat{\beta}$ is likely to be due to chance.

We could dedicate the entire course to linear regression models...

- model validation
- choice of basis functions
- influence of input locations
- ...

We will just stress a few **pros and cons of these models**:

+ provide a good noise filtering
+ are easy to interpret
− are not flexible (need to choose the basis functions)
− do not interpolate
− may explode when using high order polynomials (over-fitting)

Gaussian Process Regression

This section will be organised in 3 subsections:

1. Reminders on Multivariate normal distribution
2. Gaussian processes
3. Gaussian process regression

1. Multivariate normal distribution

The usual one dimensional normal distribution $\mathcal{N}(\mu, \sigma^2)$ has the following pdf:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \text{ for } x \in \mathbb{R}$$

It can be generalised to vectors:

Definition

We say that a vector $Y = (Y_1, \ldots, Y_n)$ follows a multivariate normal distribution if any linear combination of Y follows a normal distribution:

$$\forall \alpha \in \mathbb{R}^n,\ \alpha^t Y \sim \mathcal{N}(m, s^2)$$

The distribution of a Gaussian vector is characterised by

- a mean vector $\mu = (\mu_1, \ldots, \mu_d)$
- a $d \times d$ covariance matrix Σ : $\Sigma_{i,j} = \text{cov}(Y_i, Y_j)$

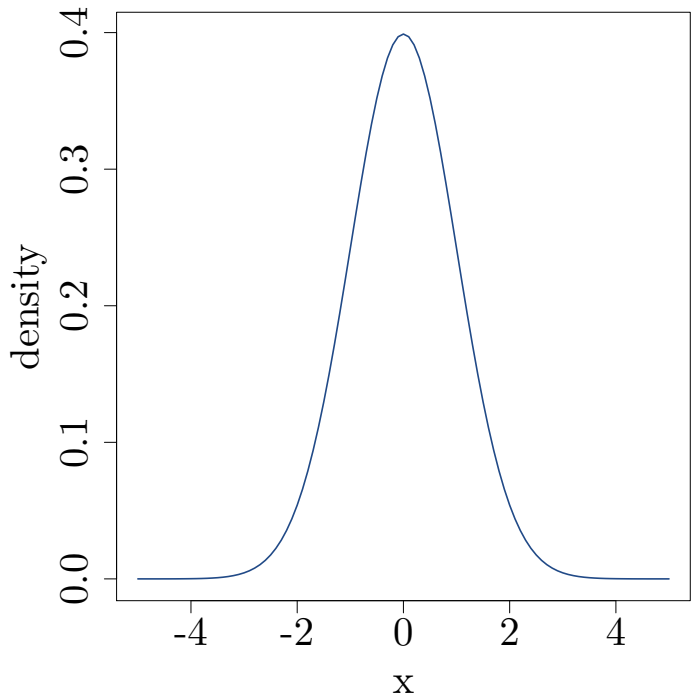
Property:

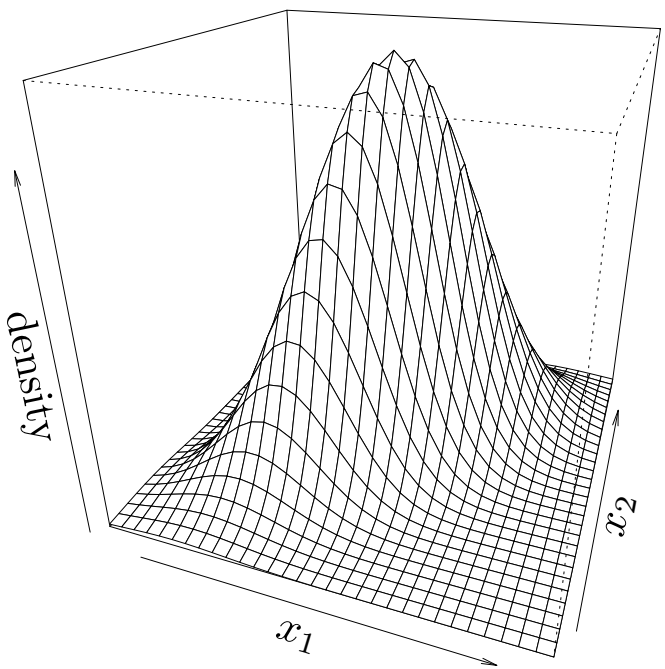
A covariance matrix is

- **symmetric** $K_{i,j} = K_{j,i}$
- **positive semi-definite** $\forall \alpha \in \mathbb{R}^d, \alpha^t K \alpha \geq 0$.

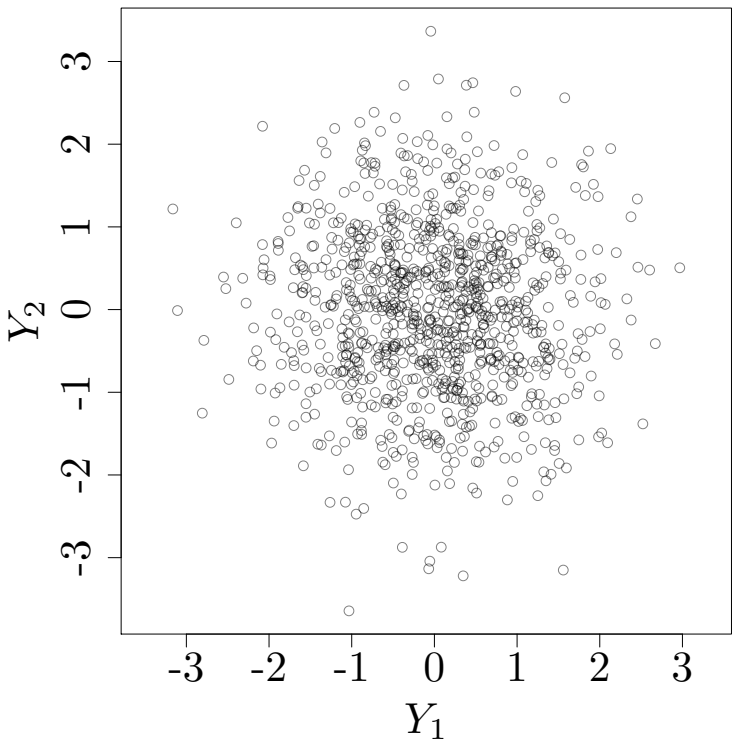
The density of a multivariate Gaussian is:

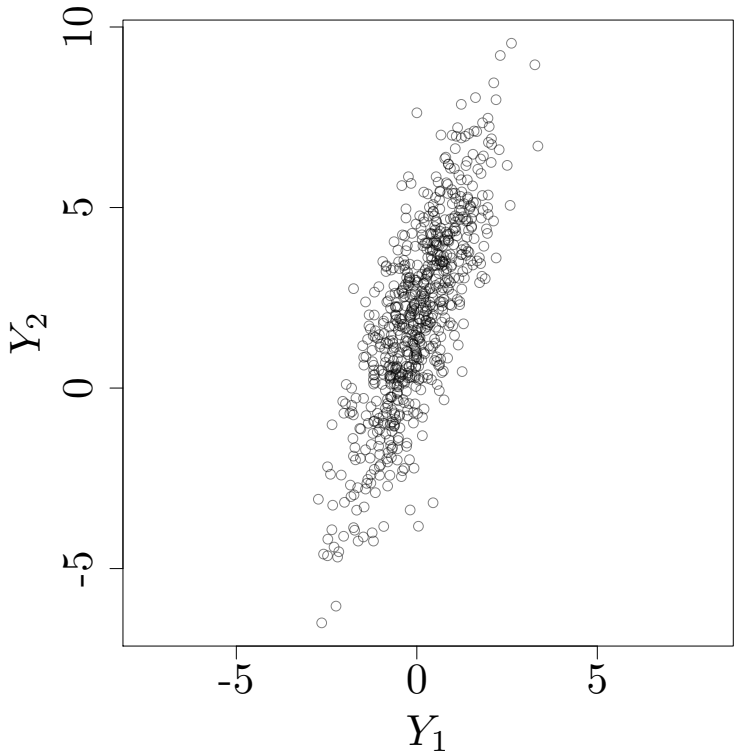
$$f_Y(x) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^t\Sigma^{-1}(x-\mu)\right).$$



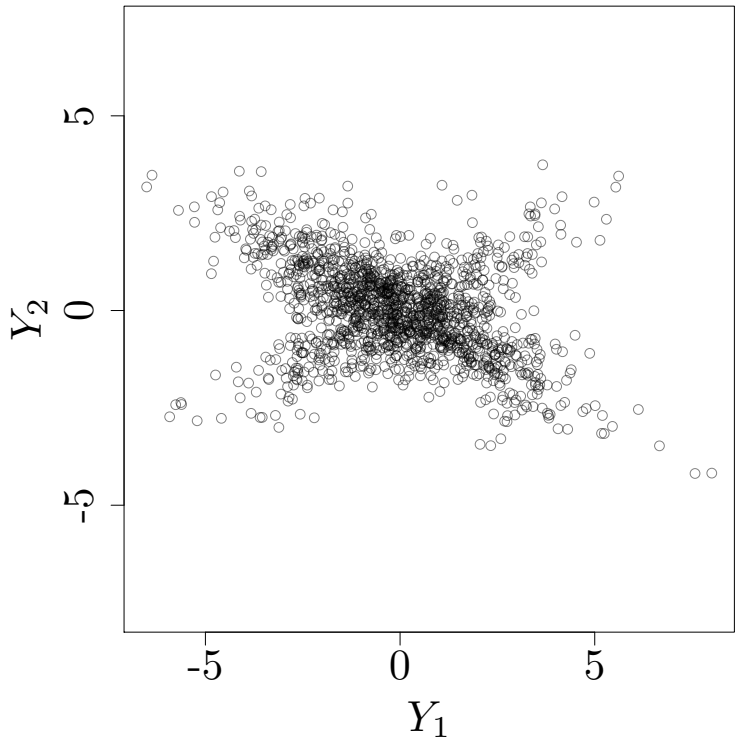


Example





Counter example



Y_1 and Y_2 are normally distributed but **the couple** (Y_1, Y_2) **is not**.

Conditional distribution

Let (Y, Z) be a Gaussian vector (Y and Z may both be vectors) with mean $(\mu_Y, \mu_Z)^t$ and covariance matrix

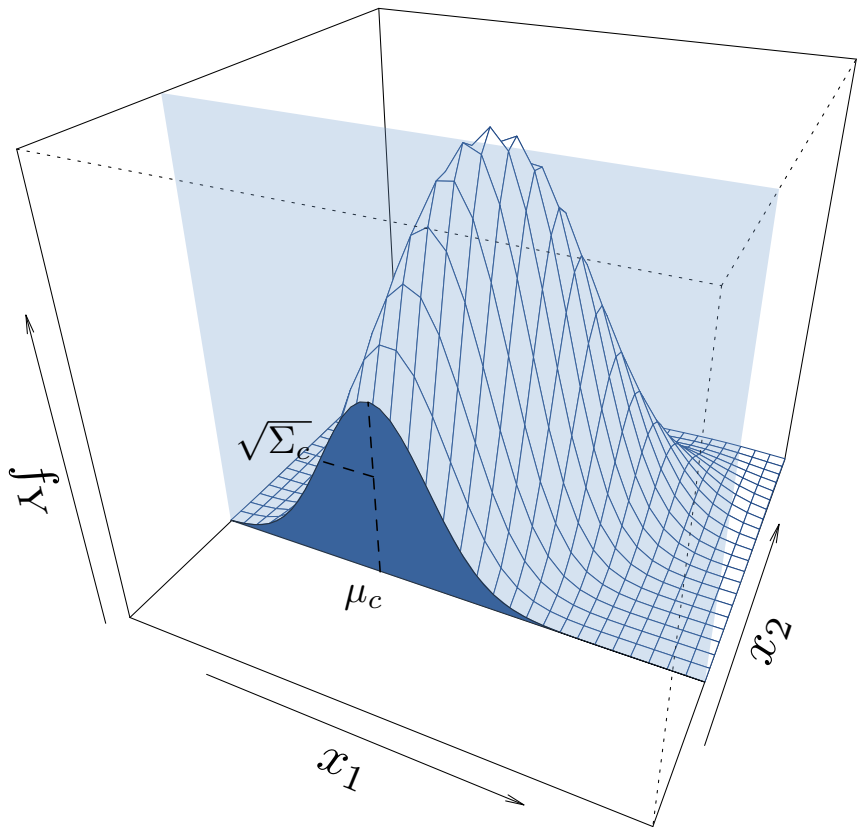
$$\begin{pmatrix} \text{cov}(Y,Y) & \text{cov}(Y,Z) \\ \text{cov}(Z,Y) & \text{cov}(Z,Z) \end{pmatrix}.$$

The conditional distribution of Y knowing Z is still multivariate normal $Y|Z \sim \mathcal{N}(\mu_{cond}, \Sigma_{cond})$ with

$$\mu_{cond} = \text{E}[Y|Z] = \mu_Y + \text{cov}(Y,Z)\,\text{cov}(Z,Z)^{-1}(Z - \mu_Z)$$
$$\Sigma_{cond} = \text{cov}[Y,Y|Z] = \text{cov}(Y,Y) - \text{cov}(Y,Z)\,\text{cov}(Z,Z)^{-1}\,\text{cov}(Z,Y)$$

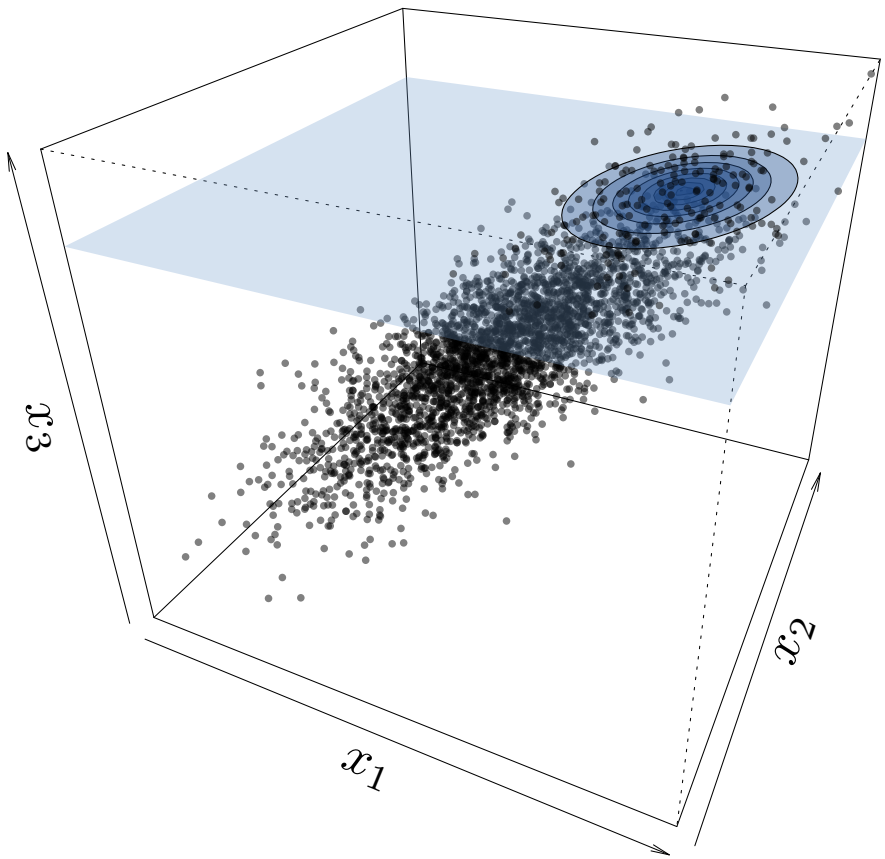
Example 1

2D multivariate Gaussian conditional distribution:



Example 2

3D multivariate Gaussian conditional distribution:



Exercise

Starting from the density function, prove the previous property using the Schur bloc inverse:

$$\begin{pmatrix} \Sigma_{1,1} & \Sigma_{1,2} \\ \Sigma_{2,1} & \Sigma_{2,2} \end{pmatrix}^{-1} = \begin{pmatrix} A & B \\ B^t & C \end{pmatrix}$$

where:

$$A = (\Sigma_{1,1} - \Sigma_{1,2}\Sigma_{2,2}^{-1}\Sigma_{2,1})^{-1}$$

$$B = -(\Sigma_{1,1} - \Sigma_{1,2}\Sigma_{2,2}^{-1}\Sigma_{2,1})^{-1}\Sigma_{1,2}\Sigma_{2,2}^{-1}$$

$$C = \Sigma_{2,2}^{-1} + \Sigma_{2,2}^{-1}\Sigma_{2,1}(\Sigma_{1,1} - \Sigma_{1,2}\Sigma_{2,2}^{-1}\Sigma_{2,1})^{-1}\Sigma_{1,2}\Sigma_{2,2}^{-1}$$

2. Gaussian processes

The multivariate Gaussian distribution can be generalised to random processes:

Definition

A random process Z over $D \subset \mathbb{R}^d$ is said to be Gaussian if

$$\forall n \in \mathbb{N}, \forall x_i \in D, (Z(x_1), \ldots, Z(x_n)) \text{ is a Gaussian vector.}$$

The distribution of a GP is fully characterised by:

- its mean function m defined over D
- its covariance function (or kernel) k defined over $D \times D$: $k(x, y) = \text{cov}(Z(x), Z(y))$

We will use the notation $Z \sim \mathcal{N}(m(.), k(.,.))$.

A kernel satisfies the following properties:

- It is symmetric: $k(x, y) = k(y, x)$
- It is positive semi-definite (psd):

$$\forall n \in \mathbb{N}, \forall x_i \in D, \forall \alpha \in \mathbb{R}^n, \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \geq 0$$

Furthermore any symmetric psd function can be seen as the covariance of a Gaussian process. This equivalence is known as the Loeve theorem.

Proving that a function is psd is often intractable. However there are a lot of functions that have already been proven to be psd:

constant	$k(x, y) = 1$
white noise	$k(x, y) = \delta_{x,y}$
Brownian	$k(x, y) = \min(x, y)$
exponential	$k(x, y) = \exp(-\lvert x - y \rvert)$
Matern 3/2	$k(x, y) = (1 + \lvert x - y \rvert) \exp(-\lvert x - y \rvert)$
Matern 5/2	$k(x, y) = \left(1 + \lvert x - y \rvert + 1/3 \lvert x - y \rvert^2\right) \exp(-\lvert x - y \rvert)$
squared exponential	$k(x, y) = \exp\left(-(x - y)^2\right)$
	$\vdots$

When k is a function of $x - y$, the kernel is called **stationary**.

Can we look at the sample paths associated to these kernels?

In order to simulate sample paths from a GP $Z \sim \mathcal{N}(m(.), k(.,.))$, we will consider samples of the GP discretised on a fine grid.

Exercise: Simulating sample paths

Let X be a set 100 regularly spaced points over the input space of Z.

- What is the distribution of $Z(X)$?
- How to simulate samples from $Z(X)$?

$\Rightarrow$ Shiny App

Furthermore, we can include some scaling parameters into the kernels:

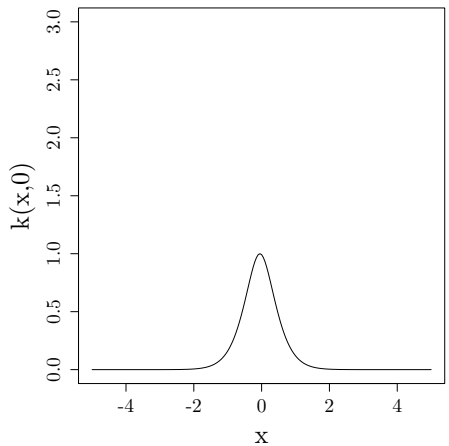
Exercise:
If Z is a GP $\mathcal{N}(0, k(.,.))$, what is the distribution of $Y(x) = \sigma Z(x/\theta)$?

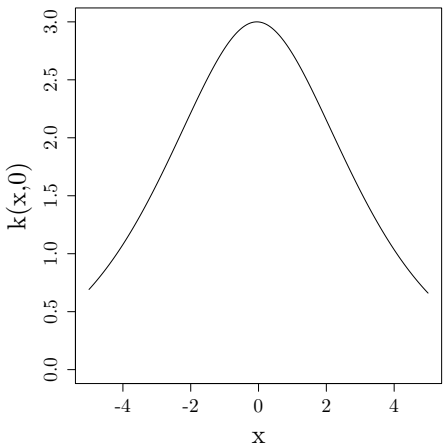
σ^2 is called the **variance** and θ the **length-scale**

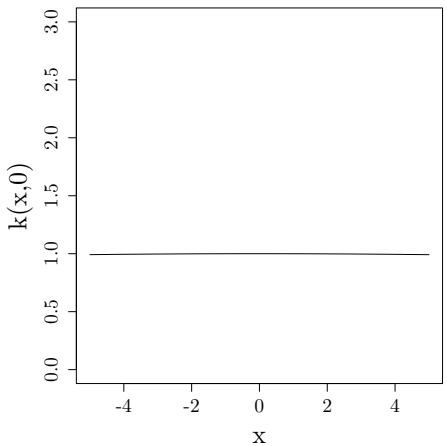
Exercise:

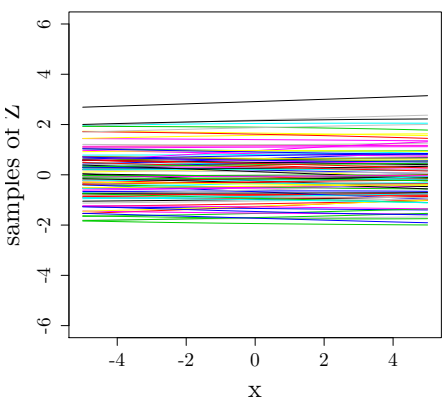
The kernel is Matern 5/2. Can you put each line in the right order?

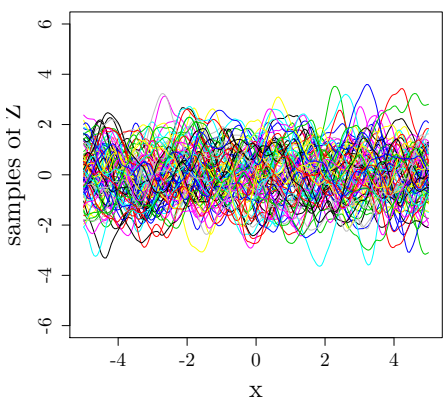
$$(\sigma^2, \theta) = (3, 3) \qquad (\sigma^2, \theta) = (1, 0.5) \qquad (\sigma^2, \theta) = (1, 50)$$

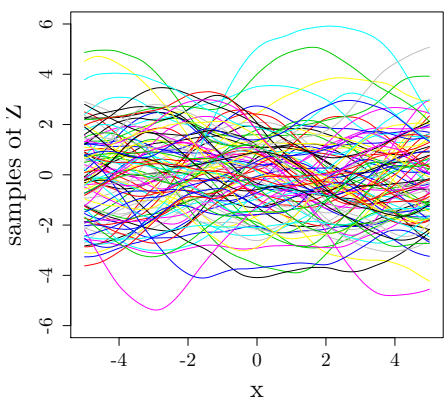








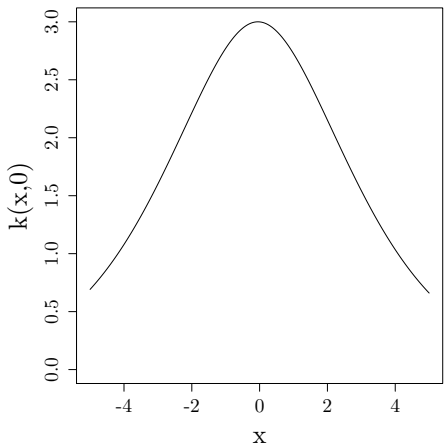


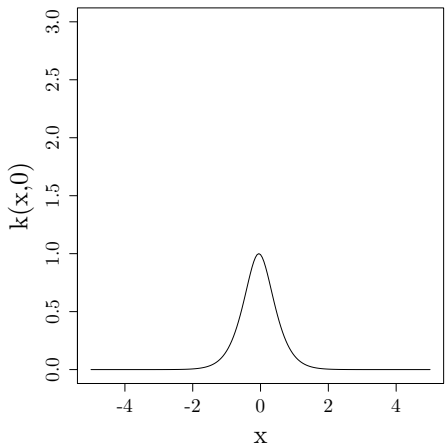


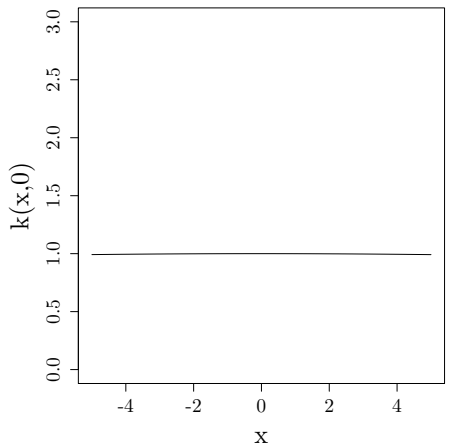
Exercise:

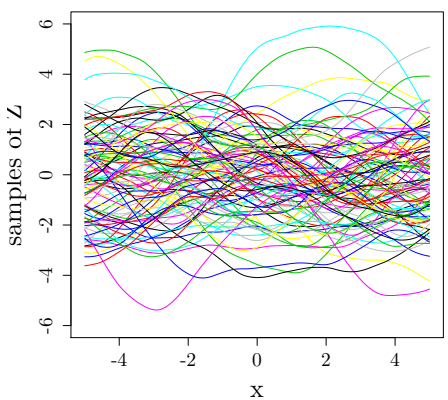
Answer is:

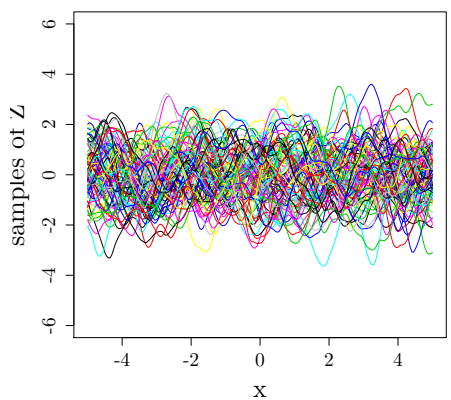
$(\sigma^2, \theta) = (3, 3)$ $(\sigma^2, \theta) = (1, 0.5)$ $(\sigma^2, \theta) = (1, 50)$

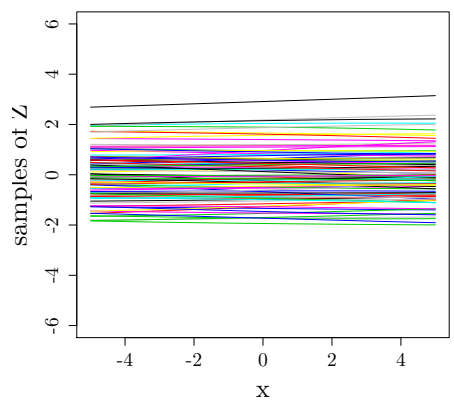












In higher dimension one can introduce one length-scale parameter per dimension. The usual Euclidean distance between two points $||x - y|| = (\sum(x_i - y_i)^2)^{1/2}$ is thus replaced by

$$||x - y||_\theta = \left(\sum_{i=1}^{d} \frac{(x_i - y_i)^2}{\theta_i^2} \right)^{1/2}.$$

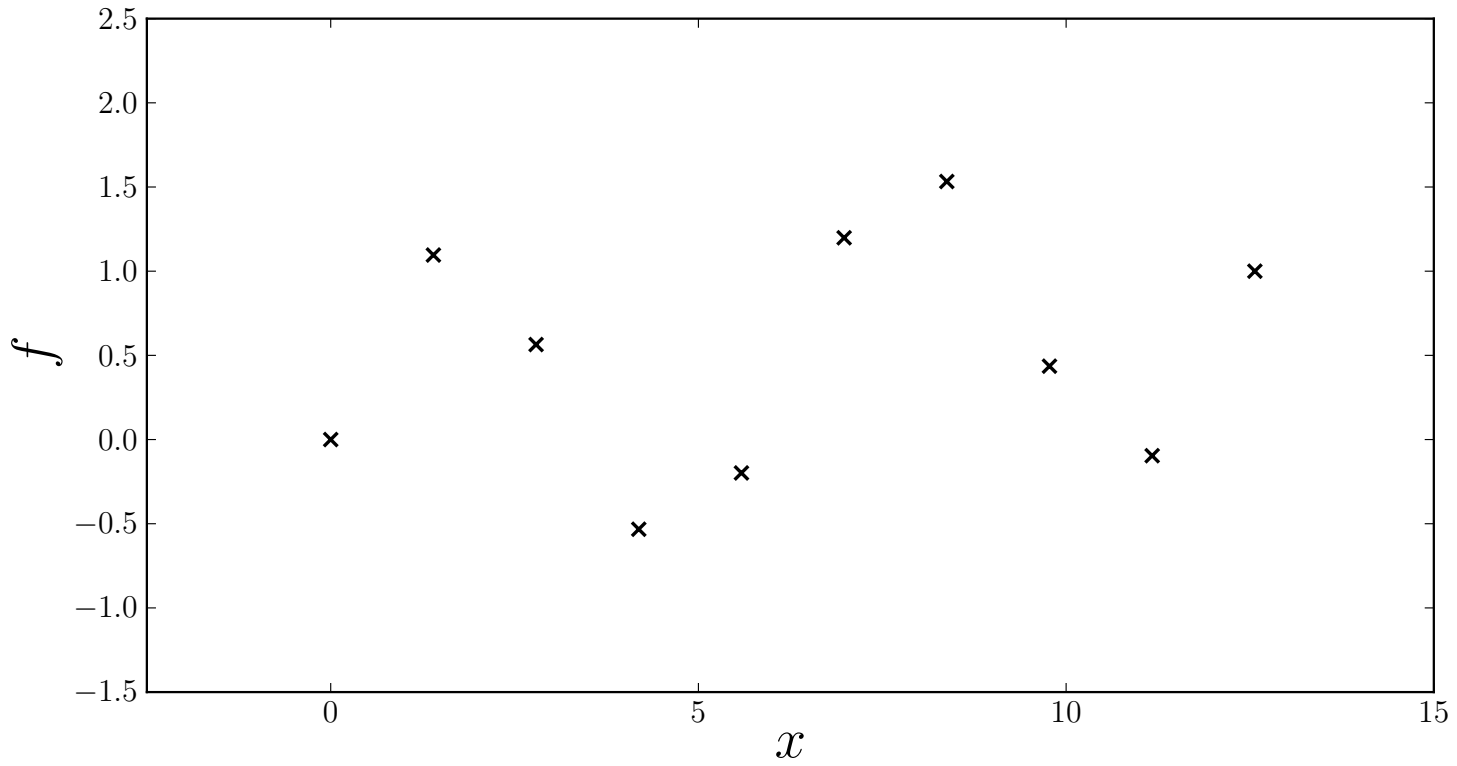
If the parameters θ_i are equal for all the dimensions, the covariance (or the process) is called **isotropic**.

Here is a list of the most common kernels:

constant	$k(x,y) = \sigma^2$
white noise	$k(x,y) = \sigma^2 \delta_{x,y}$
exponential	$k(x,y) = \sigma^2 \exp\left(-\|\|x-y\|\|_\theta\right)$
Matern 3/2	$k(x,y) = \sigma^2 \left(1+\sqrt{3}\|\|x-y\|\|_\theta\right) \exp\left(-\sqrt{3}\|\|x-y\|\|_\theta\right)$
Matern 5/2	$k(x,y) = \sigma^2 \left(1+\sqrt{5}\|\|x-y\|\|_\theta + \frac{5}{3}\|\|x-y\|\|_\theta^2\right) \exp\left(-\sqrt{5}\|\|x-y\|\|_\theta\right)$
Gaussian	$k(x,y) = \sigma^2 \exp\left(-\frac{1}{2}\|\|x-y\|\|_\theta^2\right)$

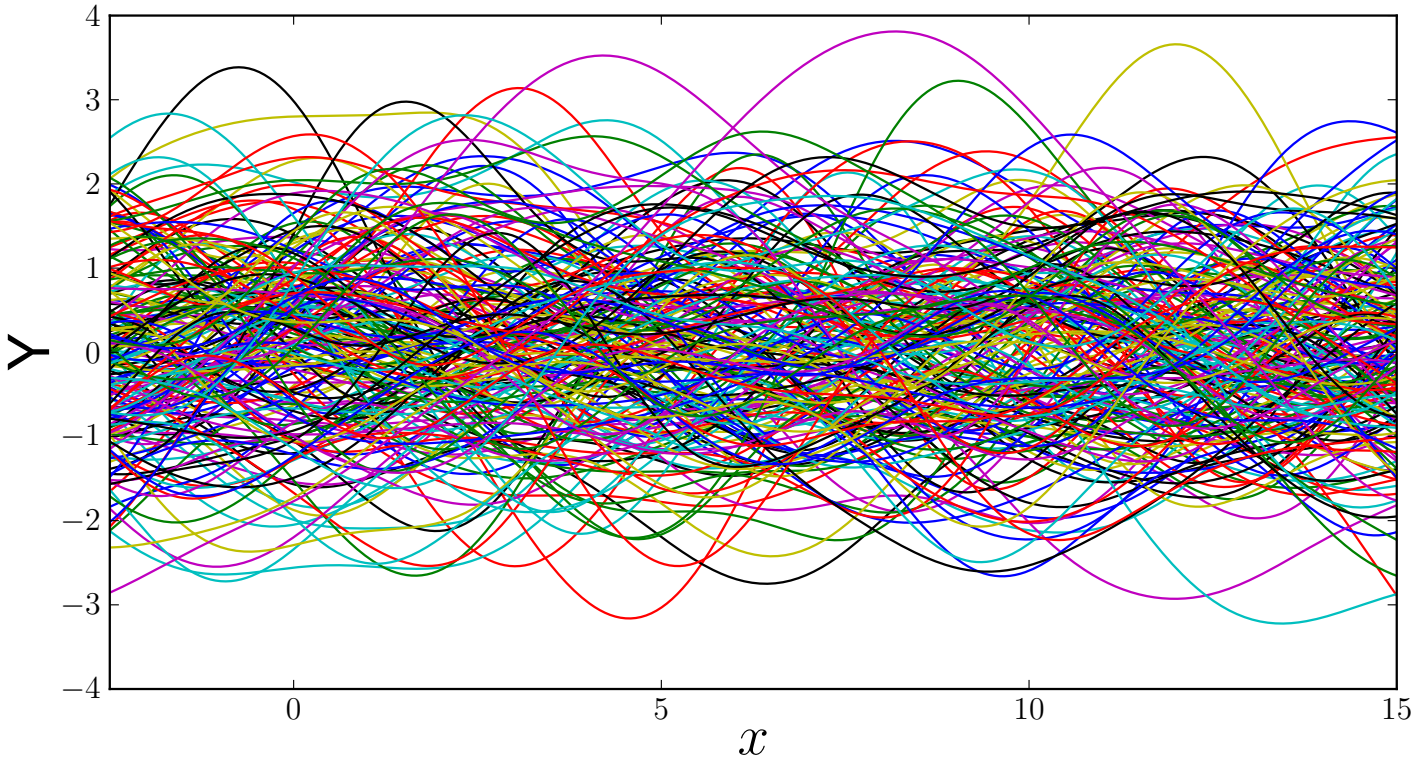
3. Gaussian process regression

We assume we have observed a function f for a set of points $X = (X_1, \ldots, X_n)$:



The vector of observations is $F = f(X)$ (ie $F_i = f(X_i)$).

Since f in unknown, we make the general assumption that it is the sample path of a Gaussian process $Z \sim \mathcal{N}(0, k)$:



What would be the next step?

We can look at the conditional distribution of Z knowing that it interpolates the data points:

Exercise

1. What is the conditional distribution of $Z(x)|Z(X) = F$?
2. Compute the conditional mean m and covariance $c(.,.)$.
3. Compute $m(X_1)$ and $c(X_1, X_1)$.

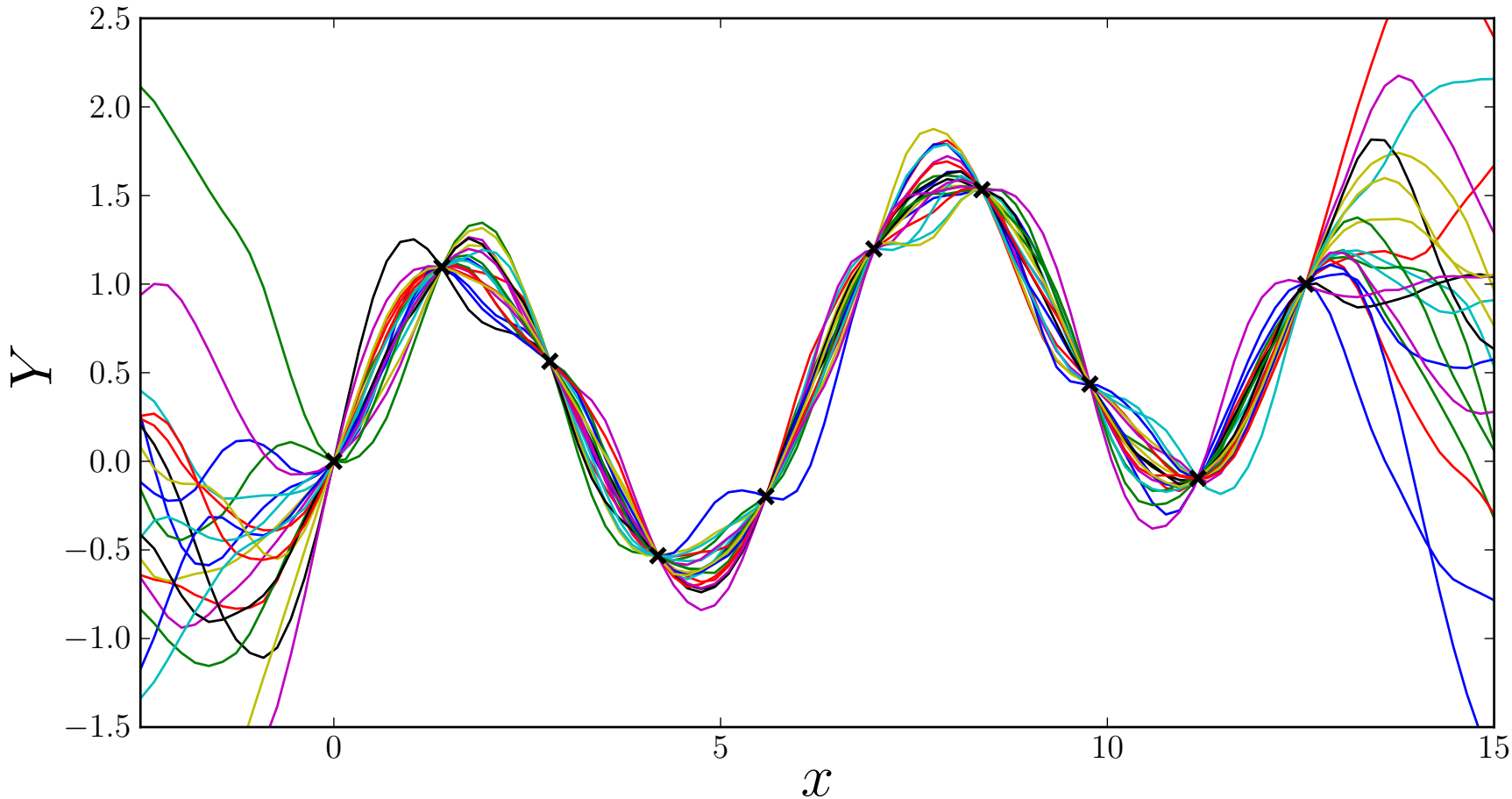
Solution

1. The conditional distribution is Gaussian.
2. It has mean and variance

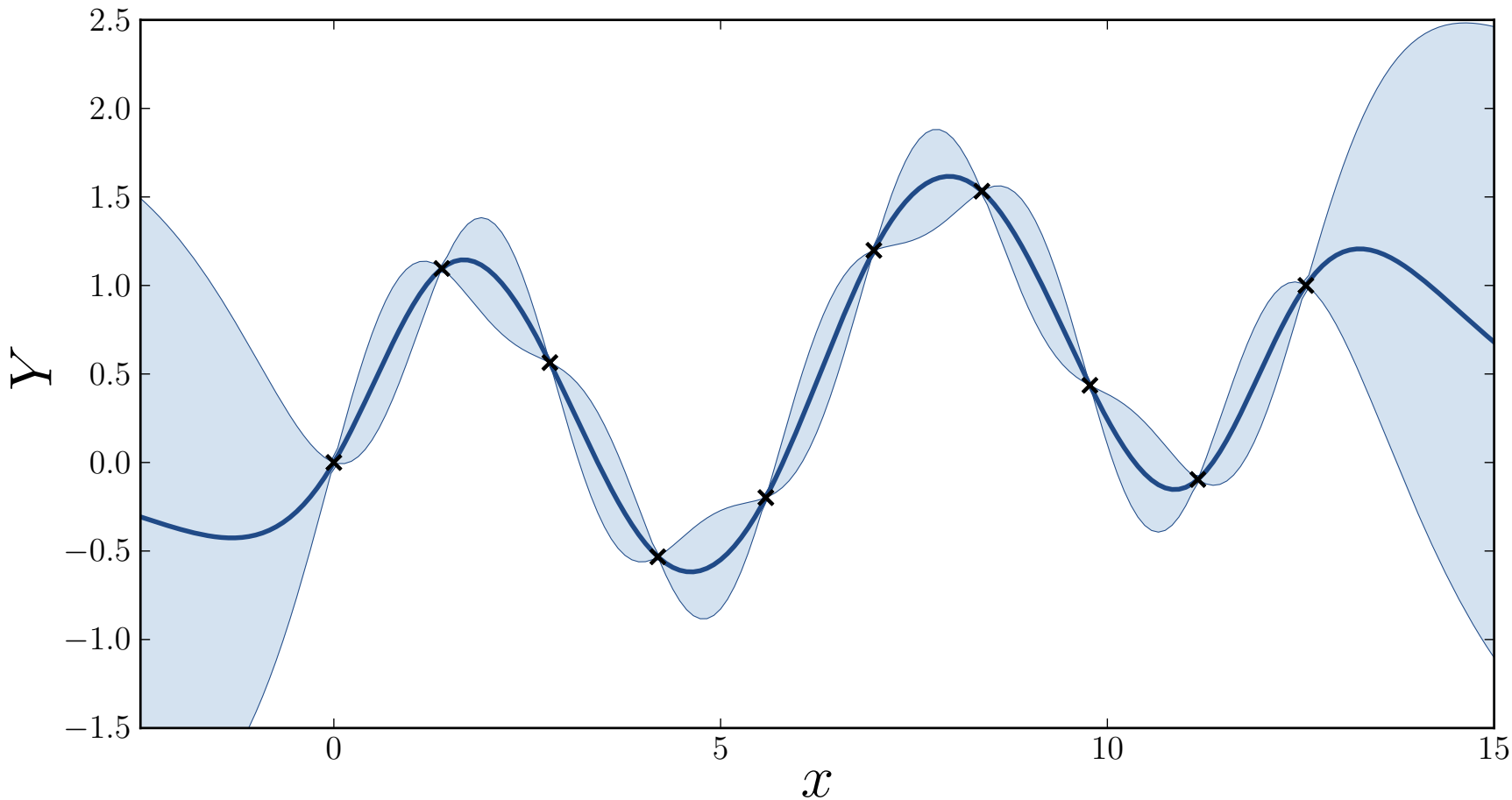
$$\begin{aligned} m(x) &= \mathrm{E}[Z(x)|Z(X)=F] \\ &= k(x,X)k(X,X)^{-1}F \\ c(x,y) &= \mathrm{cov}[Z(x),Z(y)|Z(X)=F] \\ &= k(x,y) - k(x,X)k(X,X)^{-1}k(X,y) \end{aligned}$$

3. We have $m(X_1) = F_1$ and $c(X_1, X_1) = 0$

We can look at sample paths from the conditional distribution



It can summarized by a mean function and 95% confidence intervals.



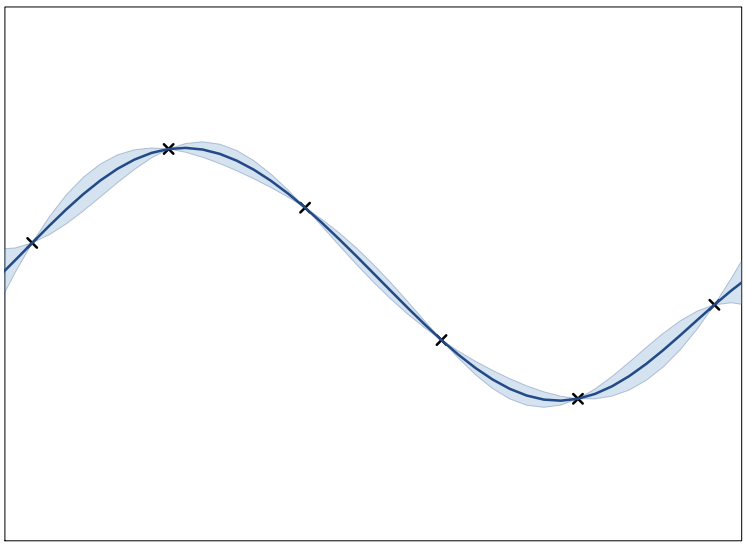
A few remarkable properties of GPR models

- They (can) interpolate the data-points
- The prediction variance does not depend on the observations
- The mean predictor does not depend on the variance
- They (usually) come back to zero when we are far away from the observations.

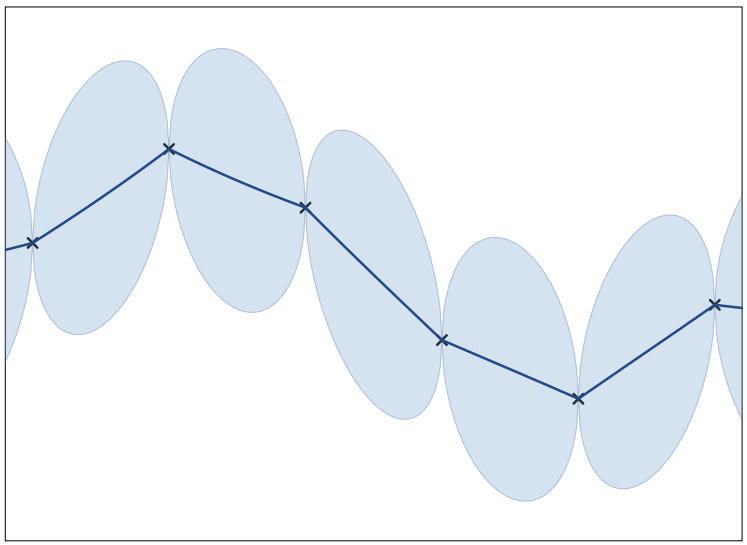
Can you prove them?

Changing the kernel has a huge impact on the model:

Gaussian kernel:

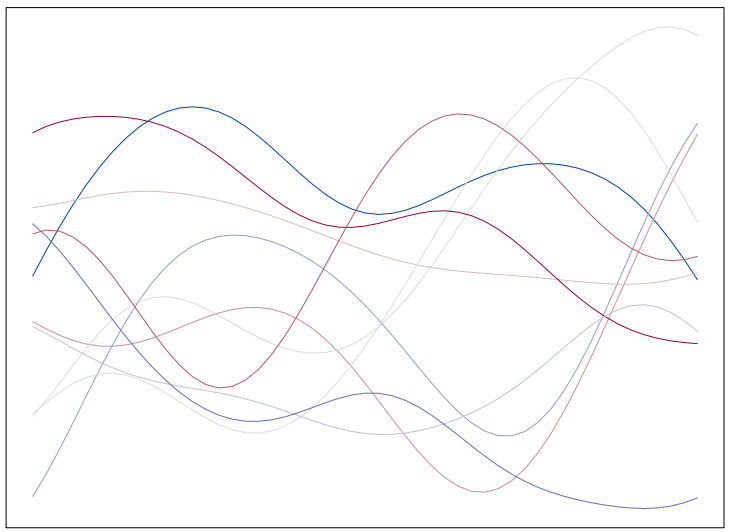


Exponential kernel:

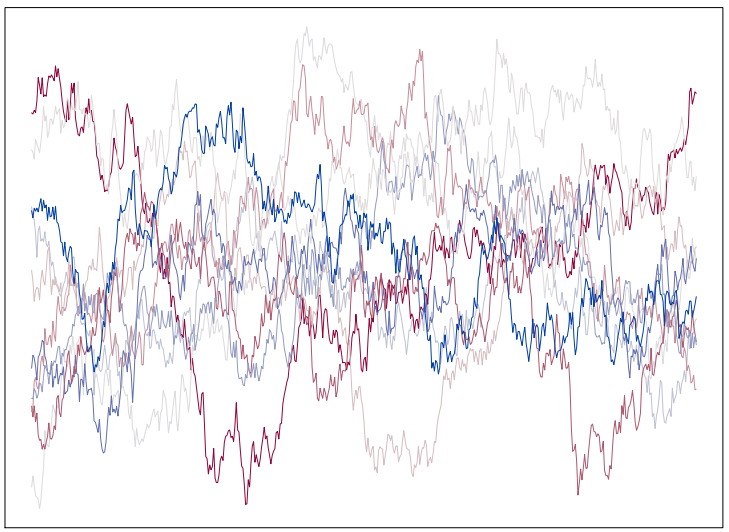


This is because changing the kernel means changing the prior on f

Gaussian kernel:

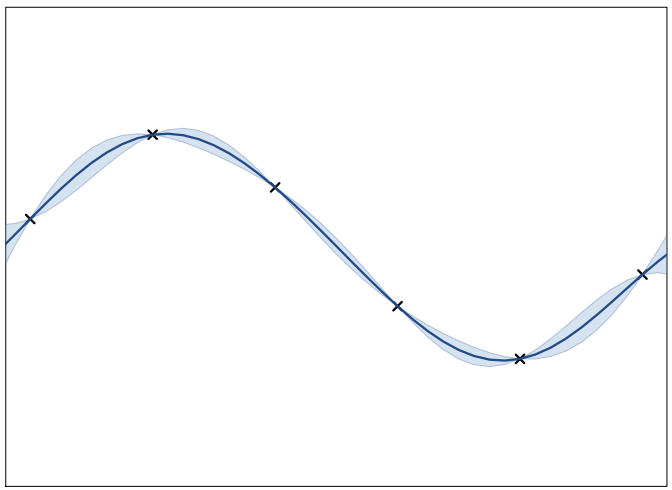


Exponential kernel:

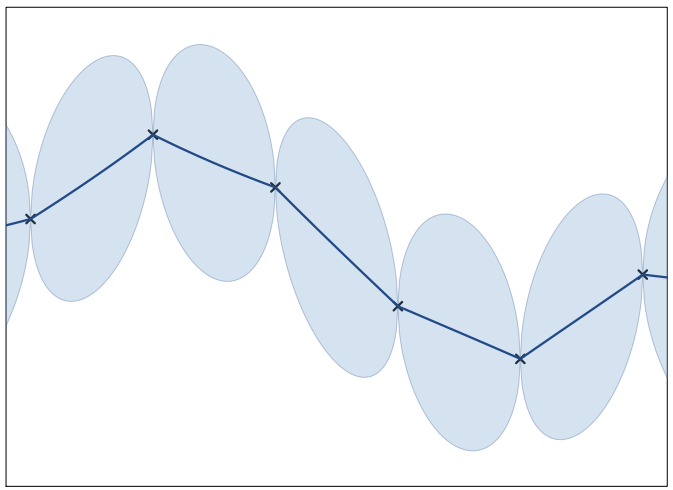


There is no model/kernel that is intrinsically better... it depends on the data!

Gaussian kernel:



Exponential kernel:



The kernel has to be chosen accordingly to our prior belief on the behaviour of the function to study:

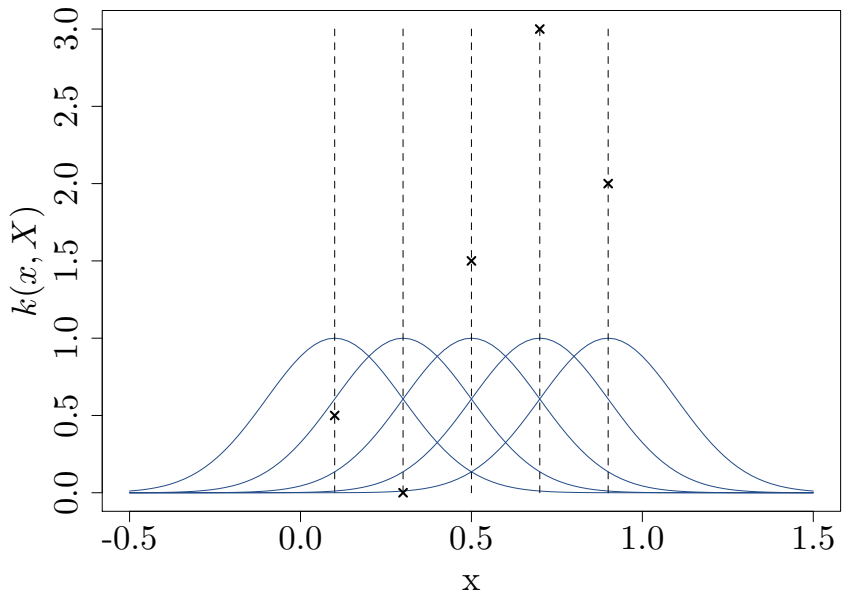
- is it continuous, differentiable, how many times?
- is it stationary ?
- ...

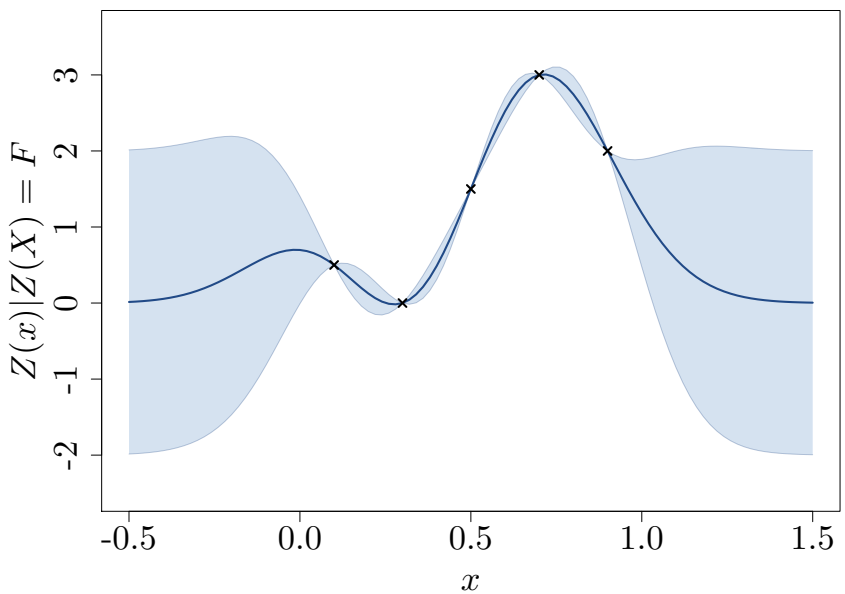
The best predictor can be seen either as a linear combination of

- the observations: $m(x) = \alpha^t F$
- the kernel evaluated at X: $m(x) = k(x, X)\beta$

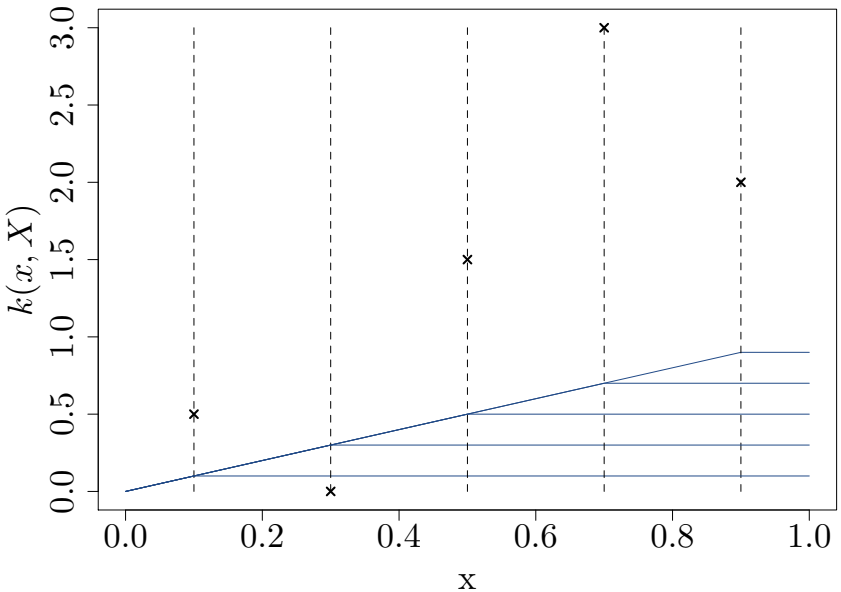
The later is interesting to understand the model shape and behaviour.

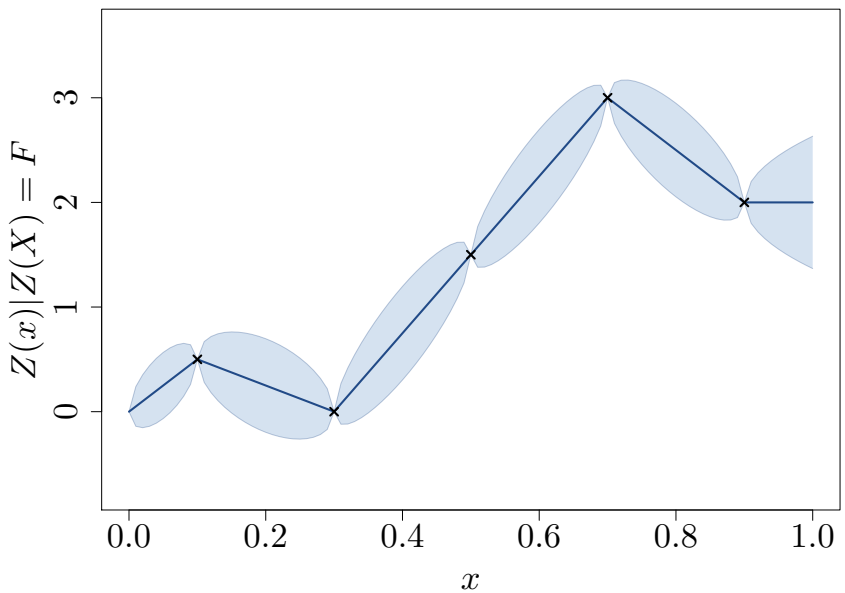
For example, we have for a squared exponential kernel





and for a Brownian kernel:





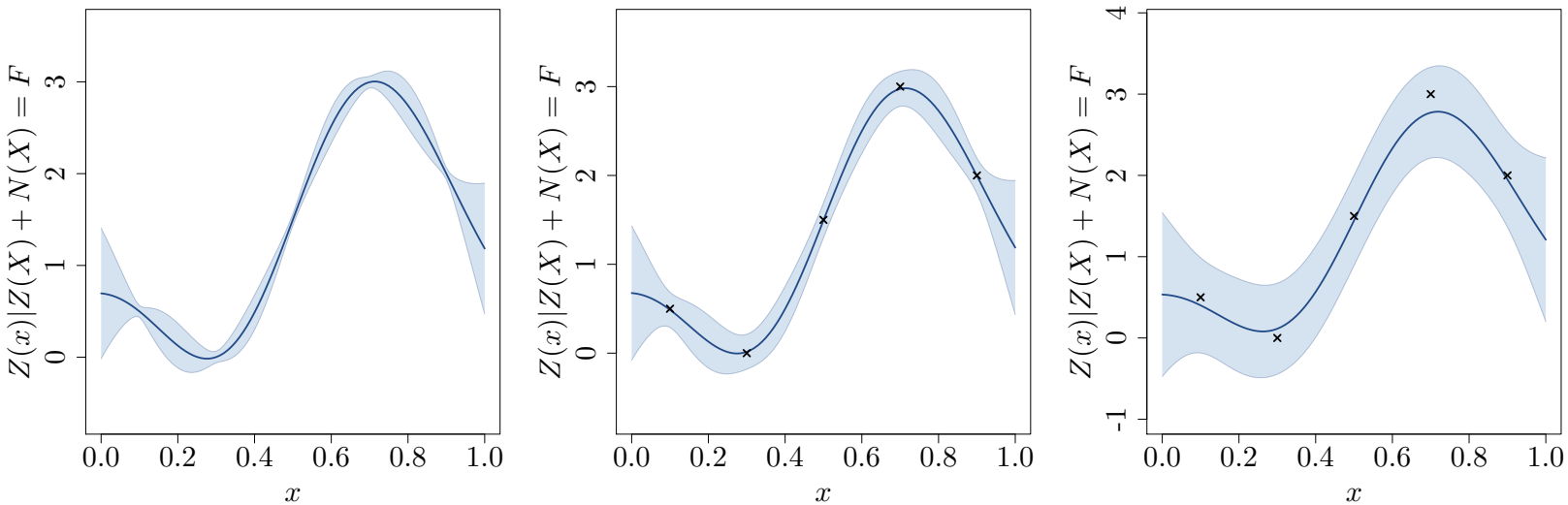
We are not always interested in models that interpolate the data.
For example, if there is some observation noise: $F = f(X) + \varepsilon$.

Let N be a process $\mathcal{N}(0, n)$ that represent the observation noise.
The expressions of GPR with noise are

$$\begin{aligned} m(x) &= \mathrm{E}[Z(x)|Z(X) + N(X)=F] \\ &= k(x, X)(k(X, X) + n(X, X))^{-1} F \end{aligned}$$

$$\begin{aligned} c(x, y) &= \mathrm{cov}[Z(x), Z(y)|Z(X) + N(X)=F] \\ &= k(x, y) - k(x, X)(k(X, X) + n(X, X))^{-1} k(X, y) \end{aligned}$$

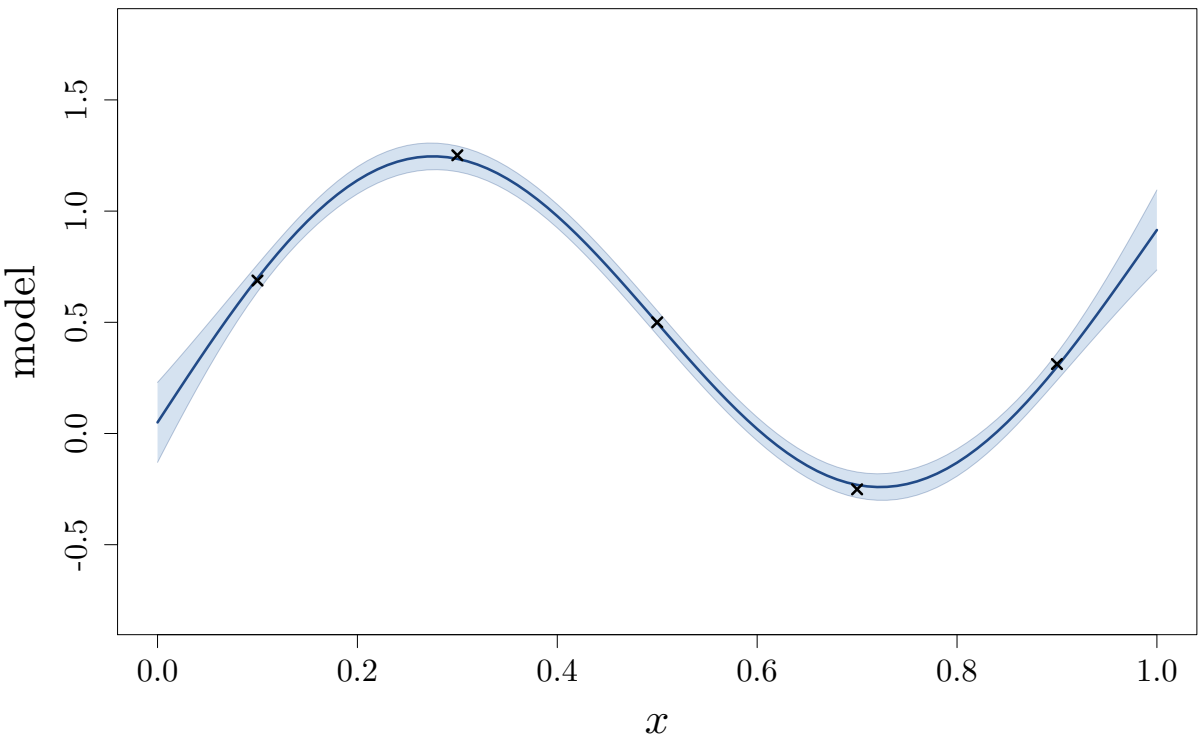
Examples of models with observation noise for $n(x, y) = \tau^2 \delta_{x,y}$:



The values of τ^2 are respectively 0.001, 0.01 and 0.1.

Design of experiments

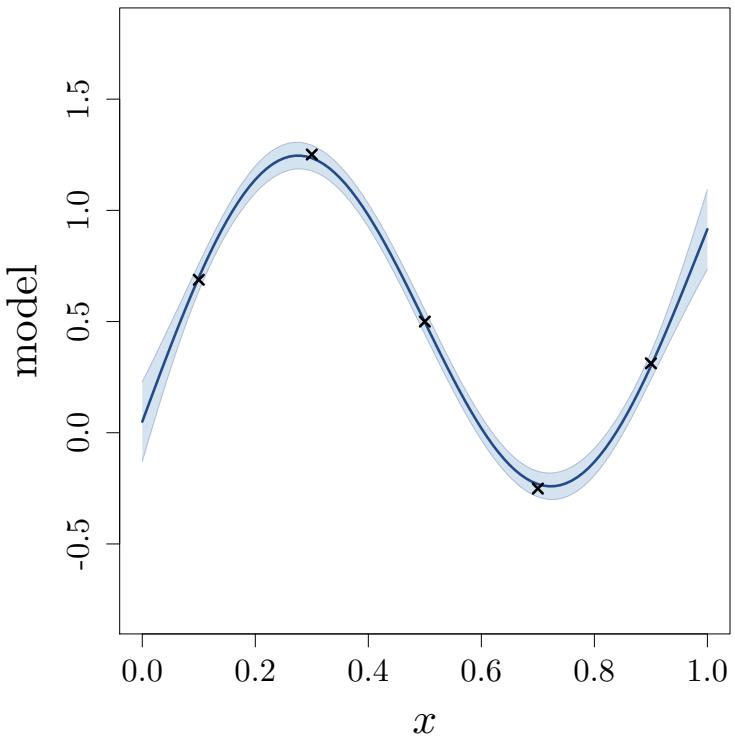
We have seen yesterday how to build a model from a given set of input/output tuples:



Today's question is: How to choose the input points to get the best model?

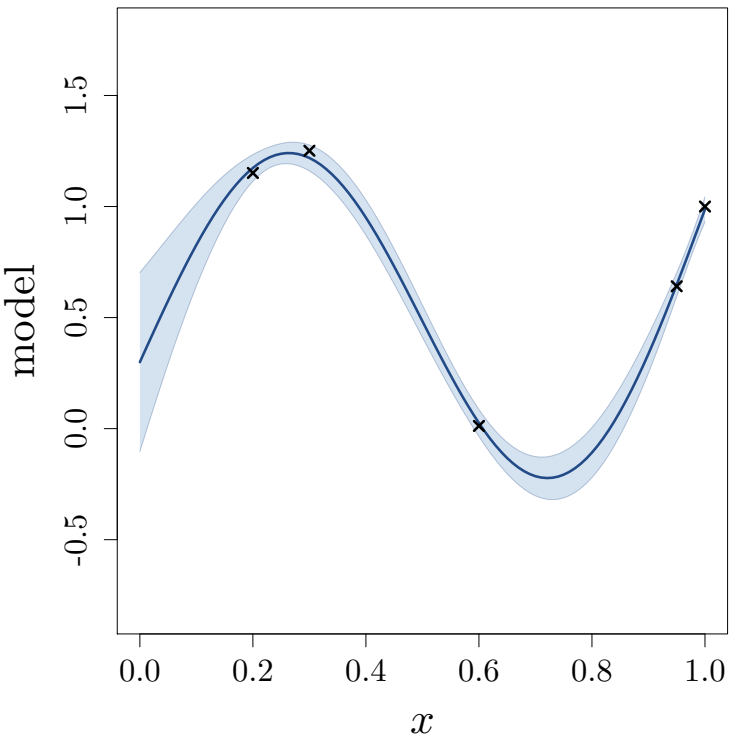
Motivating example

Same number of points but different input locations



$RSS = 0.054$

$IMSE = 0.001$



$RSS = 0.68$

$IMSE = 0.004$

Outline of the lecture

- Classical designs
- Space filling designs
- Optimal design for a given model

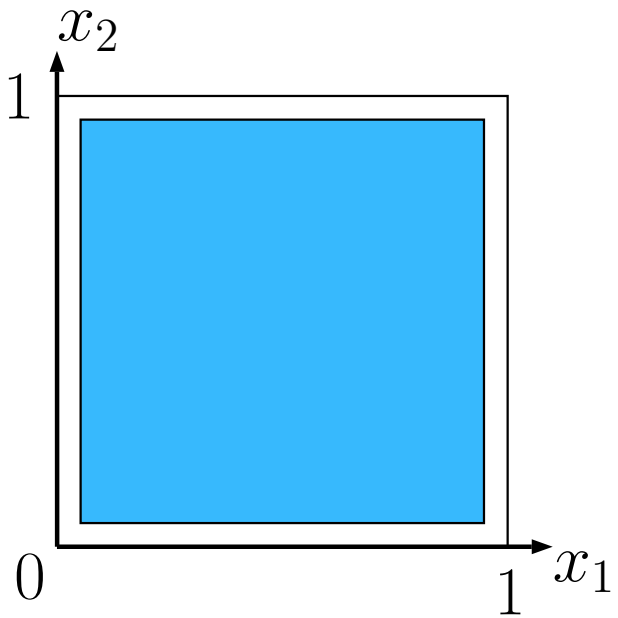
This lecture is based on a course of Victor Picheny (INRA) at Mines St-Étienne.

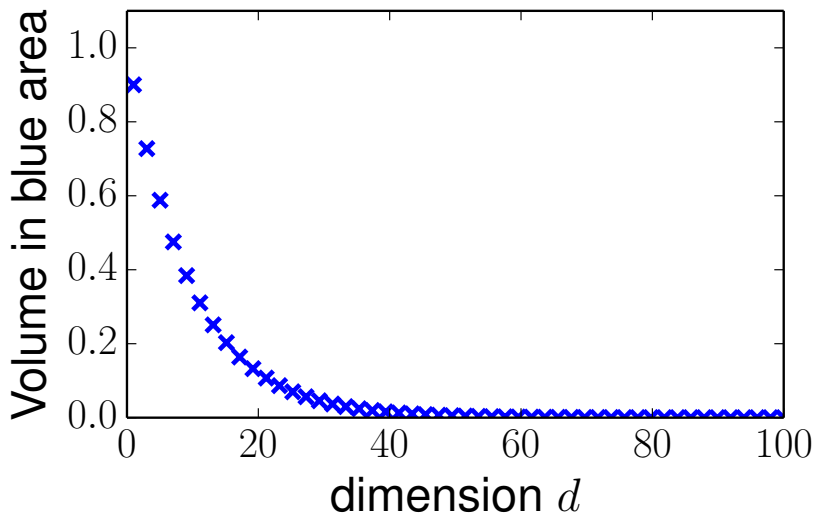
We have focus in the past lecture on 1-dimensional examples, we have to bear in mind that the input space is often of high dimension (d from 5 to 100).

Intuition is often misleading in high-dimension:

Examples 1/2

- Points in the unit cube can be far away
  $\rightarrow$ the diagonal of the unit cube is of length $\sqrt{d}$
- All the volume is near the domain boundaries
  $\rightarrow$ let us consider a hypercube of size 0.9 included in the the unit cube:

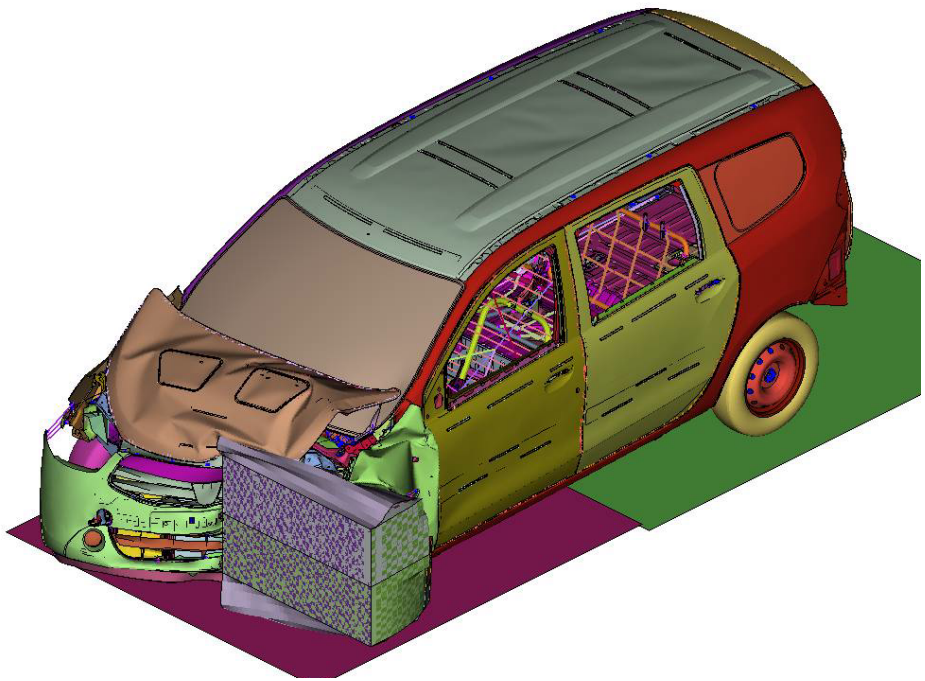




Intuition is often misleading in high-dimension:

Examples 2/2

- The number of vertices of an hypercube increases faster than we usually think

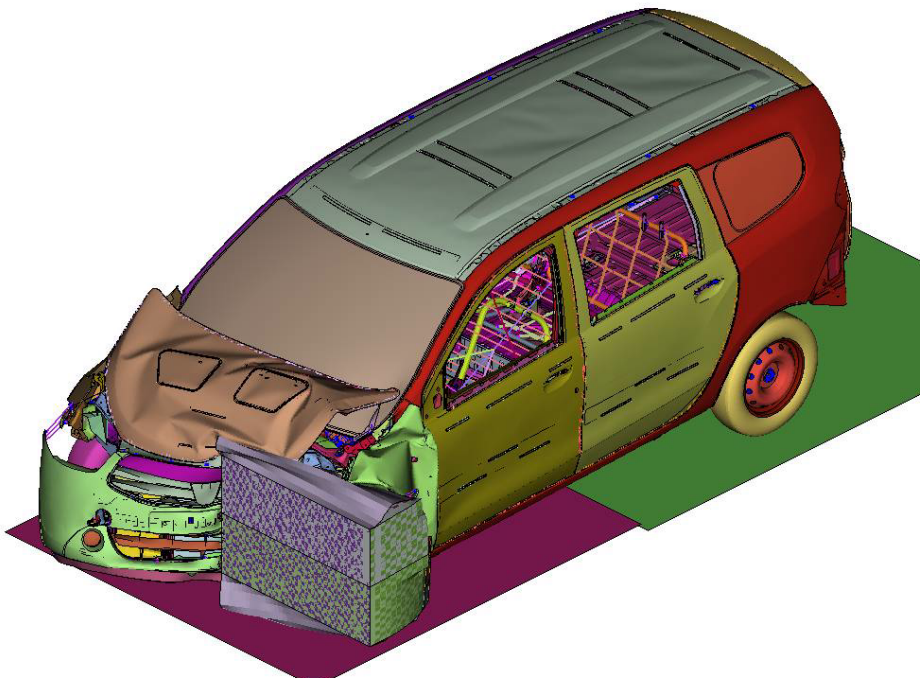


Testing all combinations of min and max input values for the 50 parameters would require...

Intuition is often misleading in high-dimension:

Examples 2/2

- The number of vertices of an hypercube increases faster than we usually think



Testing all combinations of min and max input values for the 50 parameters would require...
3000 times the age of the universe!
($d = 50 \rightarrow 2^d \approx 1.e15$)

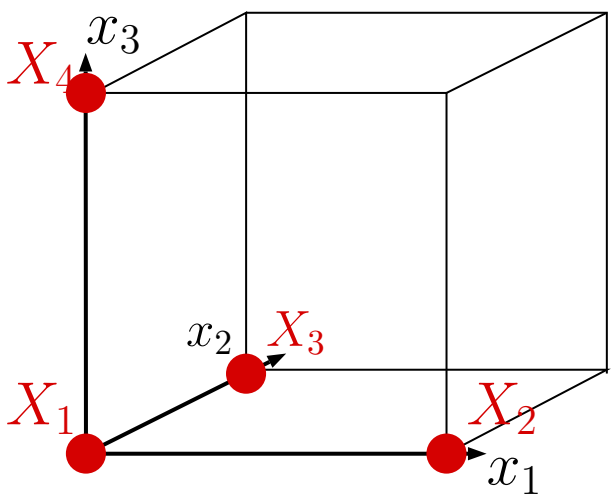
Classical designs

One at a time design

An intuitive way to check the influence of various variable is to make them change one at the time.

- All variables are fixed at a reference value (0 for example)
- One variable is changed at a time to see if there is an influence

Example



point	x_1	x_2	x_3
X_1	0	0	0
X_2	1	0	0
X_3	0	1	0
X_4	0	0	1

pros and cons of this kind of design:

+ require only $d + 1$ observations
+ are easy to interpret

– they can only see linear effects:

$$m(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

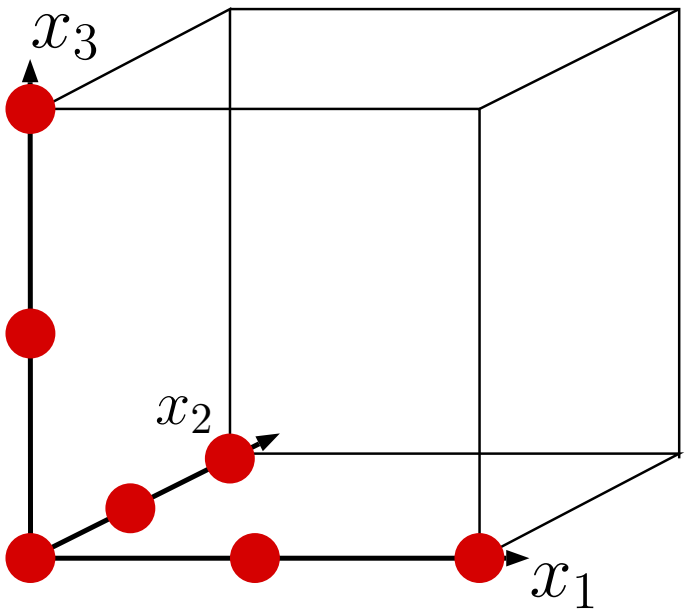
– they do not cover the space

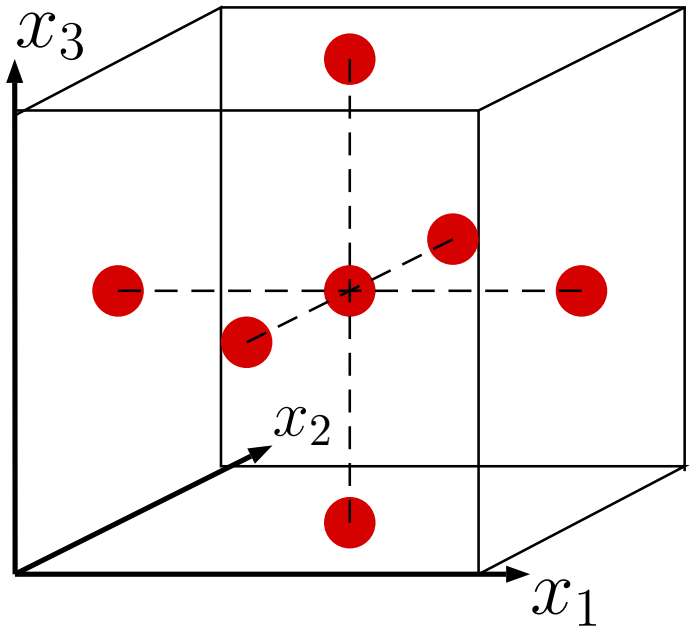
Exercise

How can this kind of design be adapted to estimate quadratic effect?

Solution

Quadratic effects can be estimated with either

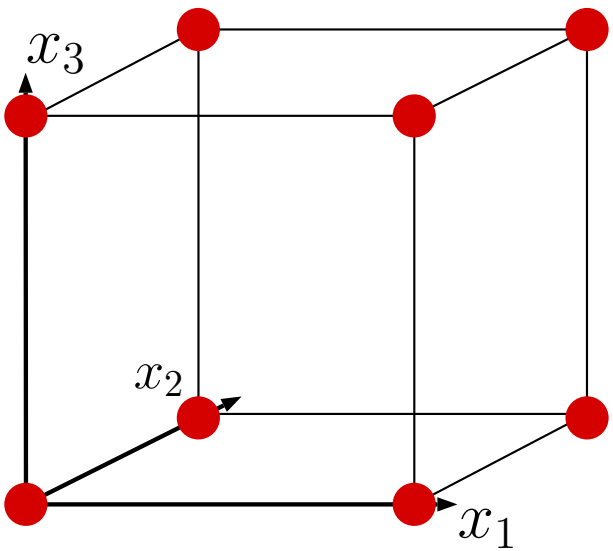




we sometime talk about “star shaped” design.

Factorial designs

The principle of factorial design is to consider all combinations for $x_i \in \{0, 1\}$:



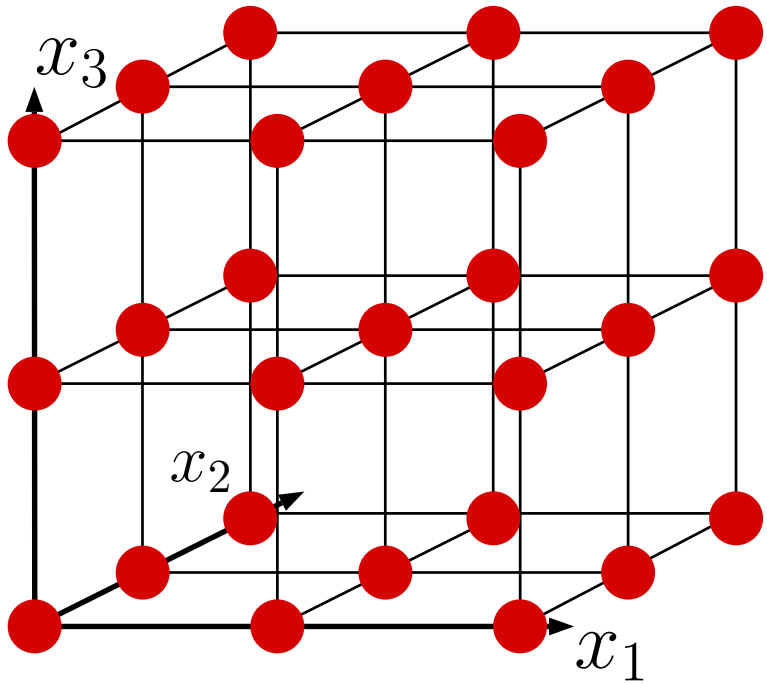
pros They allow to get all interaction terms:

$$\beta_0 + \sum_k \beta_k x_k + \sum_{j,k} \beta_{j,k} x_j x_k + \beta_{1,2,3} x_1 x_2 x_3$$

cons The number of evaluation is unrealistic when d is large

Factorial designs

It is also possible to build factorial designs with k levels:



This allows to compute quadratic effects but the number of evaluations k^d is even less realistic...

Conclusion on classical designs:

pros:

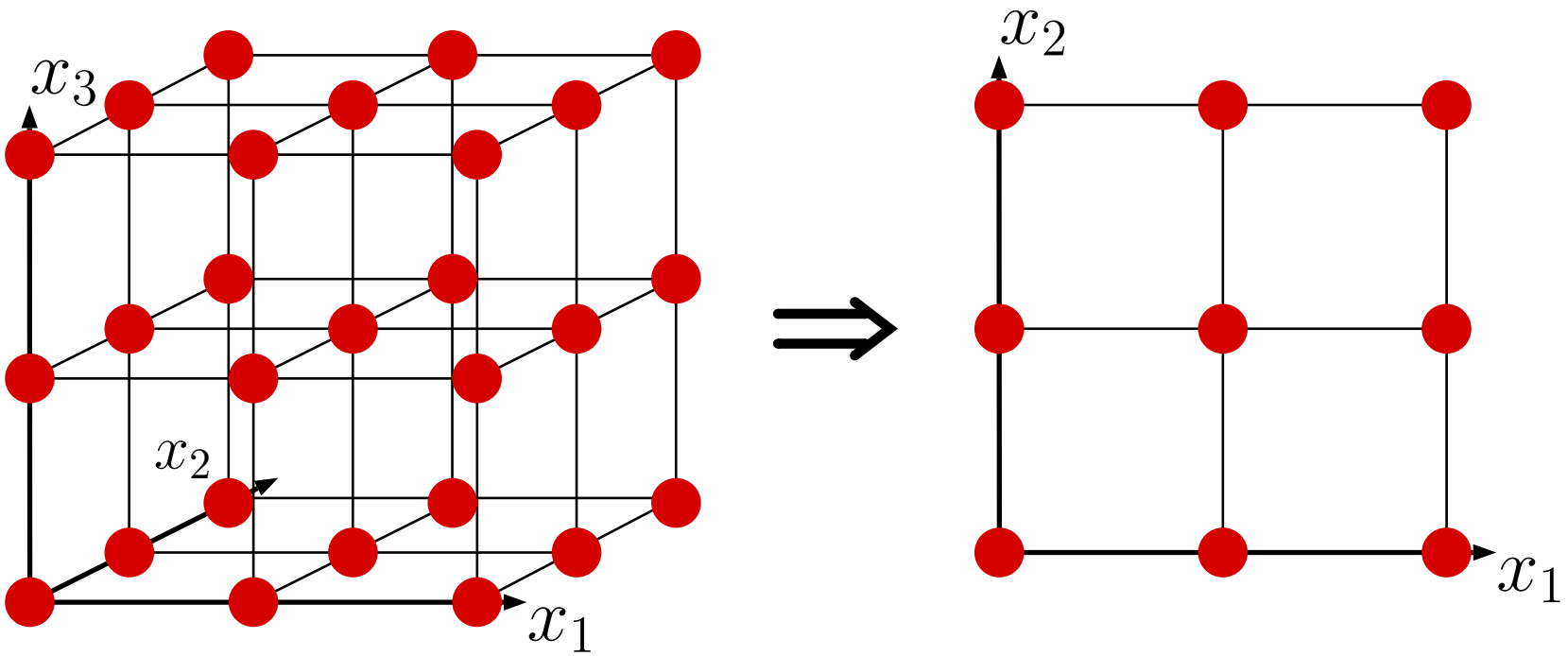
- Easy to use
- adapted to continuous or discrete variables
- Can be combined (star + factorial for example)
- Well suited (often optimal) for linear regression

cons:

- Number of evaluation is not flexible
- Number of evaluation too large in high dimension
- Points are on top of each other when projected

projection issues

Why don't we want points to be superimposed when projected?
If one of the variables has no influence, most observations become redundant...



From 27 observations, we end up with only 9...

Space filling DoE

We will now focus on designs of experiments that:

- are not model oriented
- give information about every domain of the input space
- have good projection on subspaces
- have a flexible number of points

How can we evaluate if a set of points fills the space?

Option 1. Compute distances between points

maximin the minimum distance between two points of the design should be large:

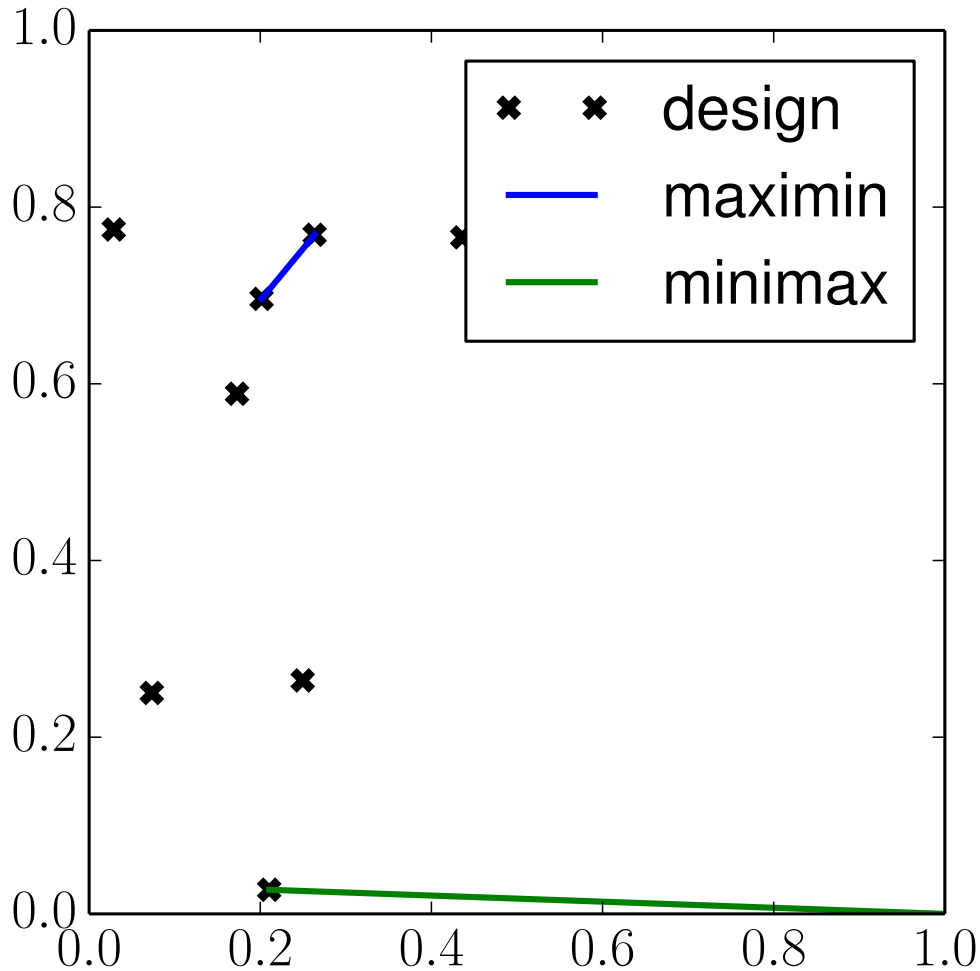
$$\text{Optimisation problem is:} \quad \max_{X_1,\dots,X_n} [\min_{i \neq j} dist(X_i, X_j)]$$

minimax the maximum distance between any point of the space and closest design point should be small:

$$\text{Optimisation problem is:} \quad \min_{X_1,\dots,X_n} (\max_{x \in D} [\min_{i} dist(x, X_i)])$$

The second criterion is much more difficult to optimise

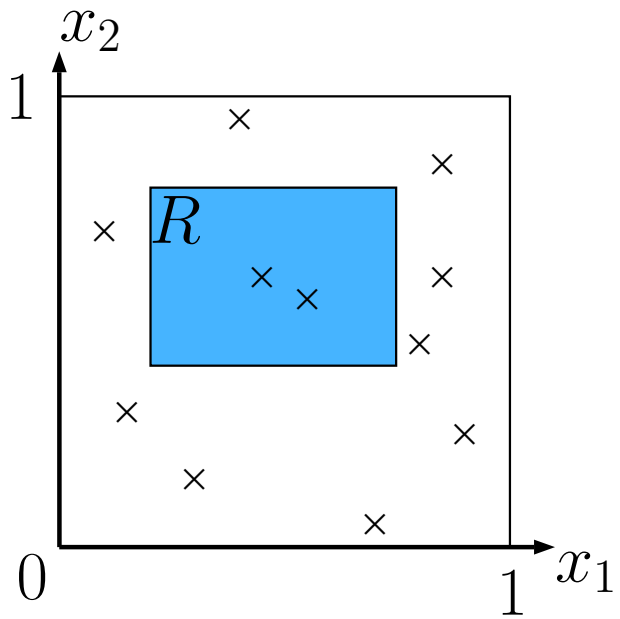
These criteria can be illustrated on a simple 2-dimensional example



How can we evaluate if a set of points fills the space?

Option 2. Compare the distribution with an uniform distribution

Discrepency is a measure of non uniformity. It compares the number of points in a hyper-rectangle with the expected number of samples from a uniform distribution



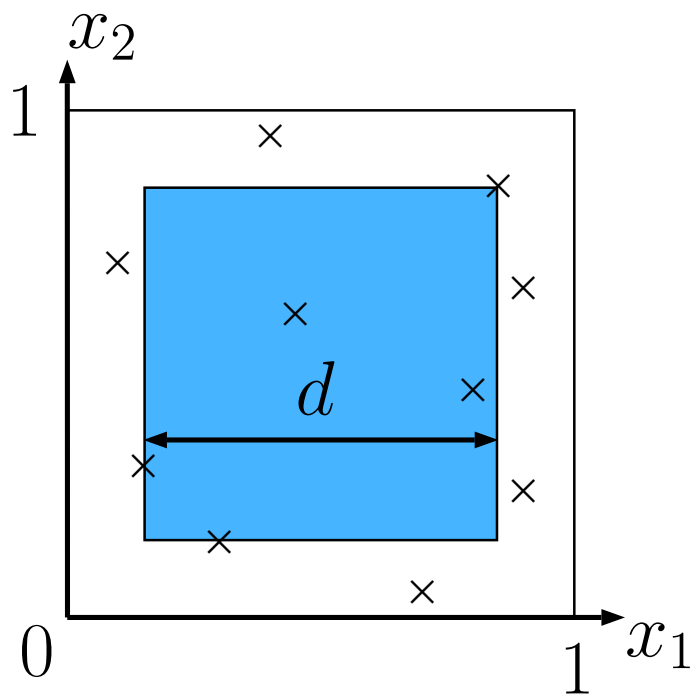
The probability for a uniform variable to be in R is 0.22 and we observe an empirical probability of 2/11. The discrepancy (w.r.t. R) is then:

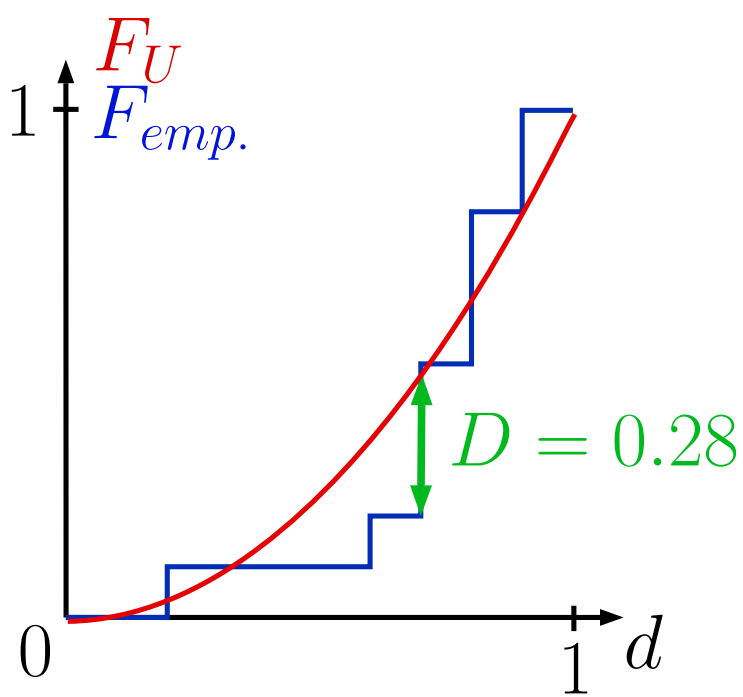
$$D_R = |0.22 - 2/11| = 0.038$$

Discrepency is defined as the sup of the distance between the empirical and analytical cdf.

Discrepency is often computed by:

- fixing one of the hyper-rectangle summit at the origin
- centering the hyper-rectangle





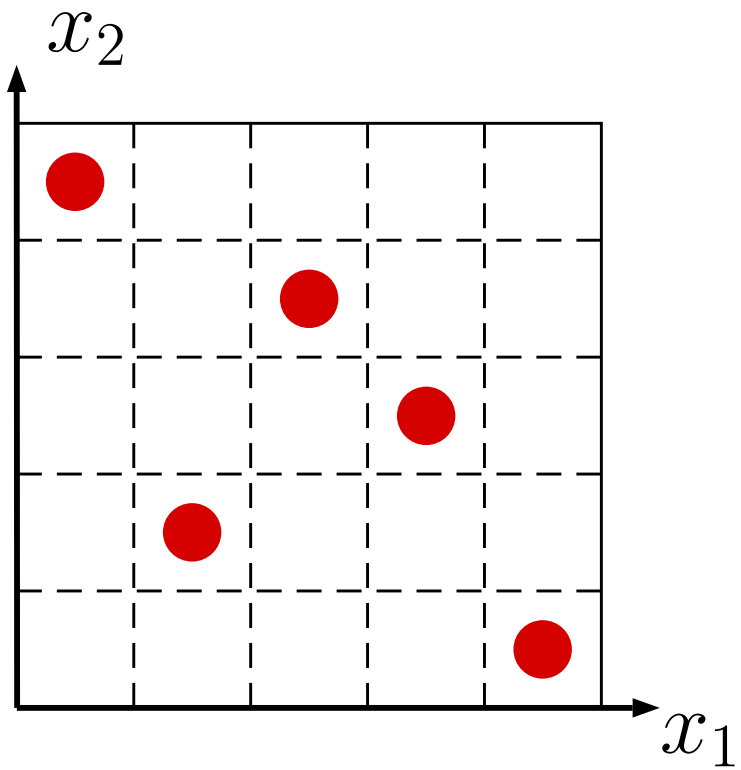
The maximum is located where the rectangle is tangent to points
$\rightarrow$ The optimisation is over a finite space

We will now give an overview on three types of space filling designs:

- Latin hypercubes
- low discrepancy sequences
- centroidal Voronoi tesselations

Latin hypercubes

Latin hypercubes are designs where the domain is sliced in n^d blocks and where there is only one point per "line" and "column":



These designs have good projection properties

A well known example of LHS in 2D is... Sudoku

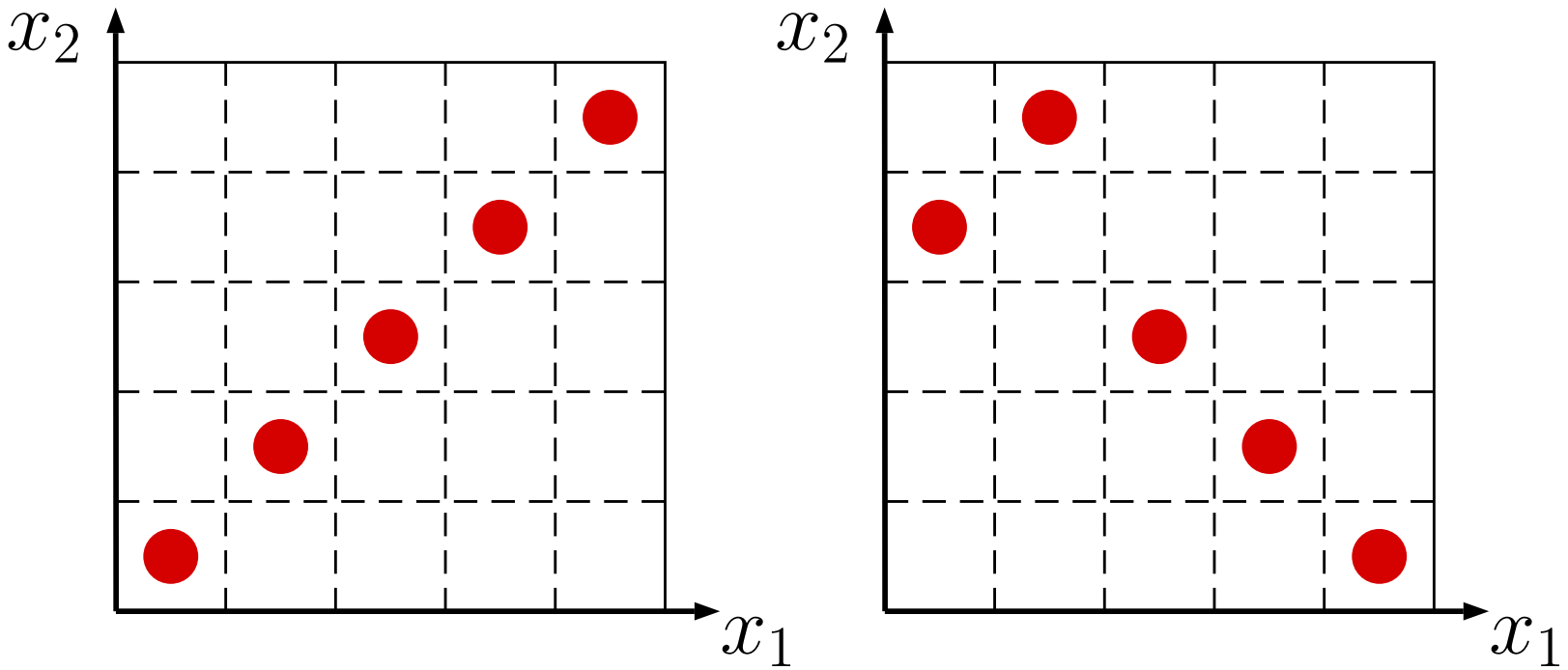
4	3	1	6	7	9	5	2	8
9	6	7	2	5	8	3	4	1
5	8	2	1	4	3	9	6	7
6	5	9	8	1	7	2	3	4
3	2	8	5	6	4	1	7	9
7	1	4	9	3	2	8	5	6
8	7	3	4	2	1	6	9	5
1	4	5	3	9	6	7	8	2
2	9	6	7	8	5	4	1	3

If we focus on one digit (say 4), we obtain a LHD:

●	3	1	6	7	9	5	2	8
9	6	7	2	5	8	3	●	1
5	8	2	1	●	3	9	6	7
6	5	9	8	1	7	2	3	●
3	2	8	5	6	●	1	7	9
7	1	●	9	3	2	8	5	6
8	7	3	●	2	1	6	9	5
1	●	5	3	9	6	7	8	2
2	9	6	7	8	5	●	1	3

Sudoku have more properties that LHD: the generalisation is called **orthogonal array**.

Latin hypercubes do not necessarily cover the space very well...



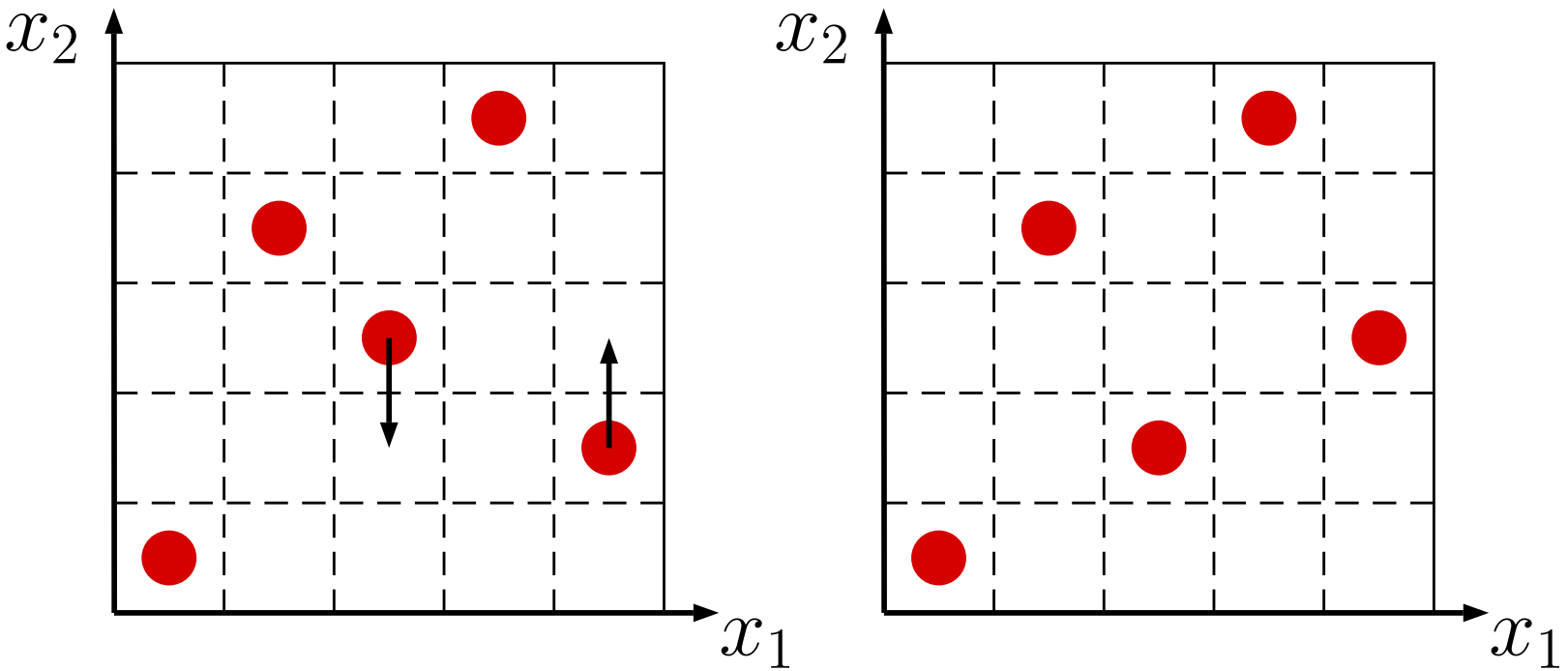
They have to be combined with a criterion such as maximin.

Exercise

- Generate a 5 points LHD in dimension 3.
- How would you program a function `LHD(n,d)`?
- How would you optimize a LHD to ensure it fills the space properly?

Solution

The coordinates of two points can be exchanged:



LHD optimization with simulated annealing:

Morris and Mitchell Algorithm

1. Generate LHD
2. find "bad" points according to maximin
3. choose randomly a column of this critical point and exchange it with an randomly selected other point
4. if the criteria is improved, the modification is accepted
5. otherwise, it is accepted with a probability of

$$\exp\left(\frac{maximin_{new} - maximin_{old}}{T}\right)$$

Low discrepancy sequences

Low discrepancy sequences are deterministic sequences that converge toward the uniform distribution.

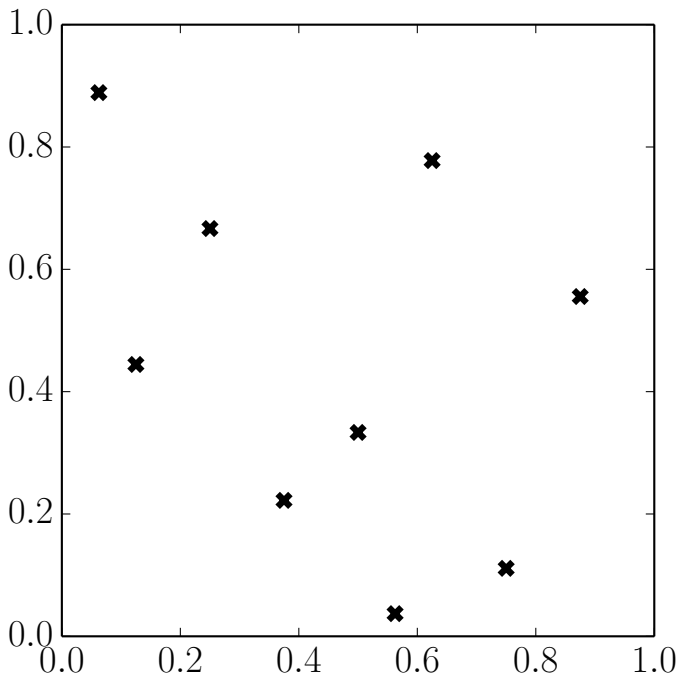
- They cover the space quickly and evenly
- They are easy to build
- It is easy to add new points

Many low discrepancy sequences can be found in the literature: Halton, Hammerley, Sobol', Faure, van der Corput, ...

Example (Halton sequence)

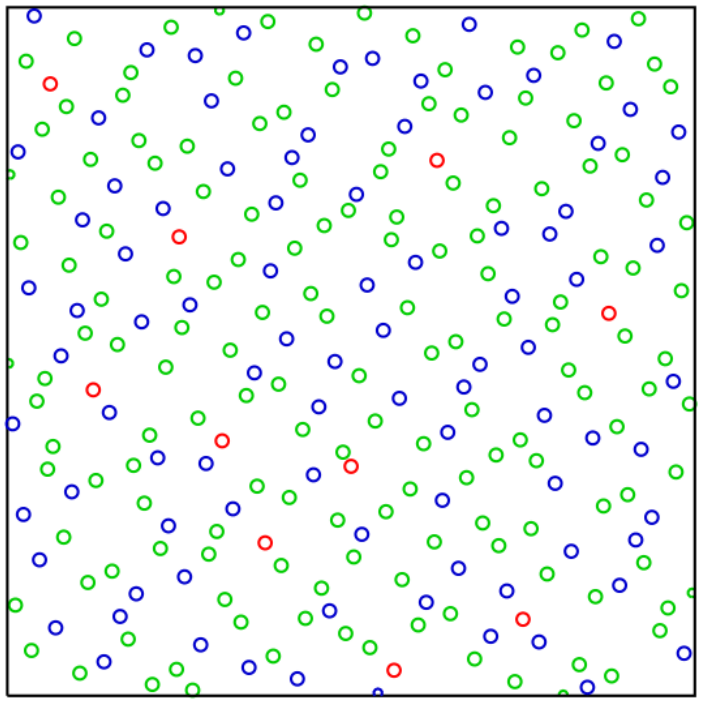
Let a and b be two integers with no common dividers (say 2 and 3). The x_1 and x_2 coordinates of the Halton sequence are:

$$x_1 = 1/2,\ 1/4,\ 3/4,\ 1/8,\ 5/8,\ 3/8,\ 7/8,\ 1/16,\ 9/16, \ldots$$
$$x_2 = 1/3,\ 2/3,\ 1/9,\ 4/9,\ 7/9,\ 2/9,\ 5/9,\ 8/9,\ 1/27, \ldots$$

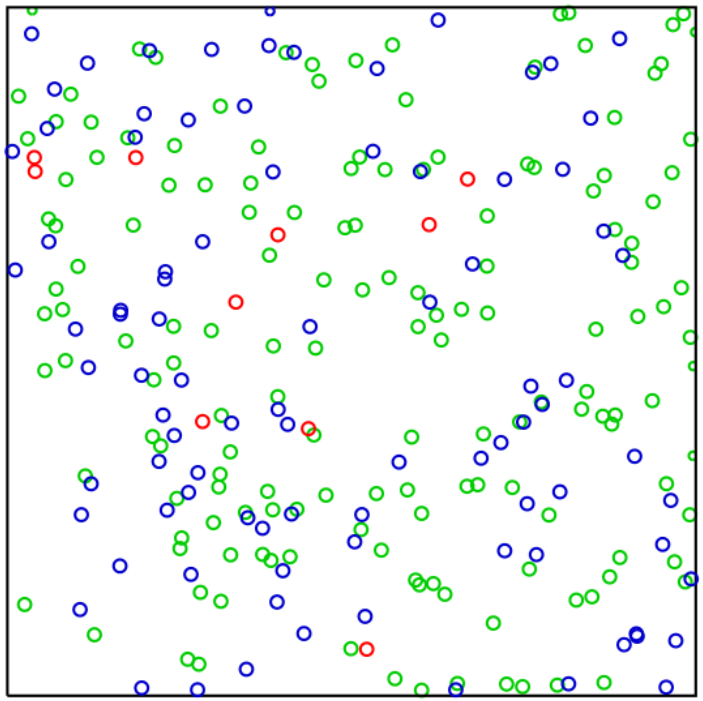


Example (Halton sequence)

Halton Sequence



uniform pseudo random



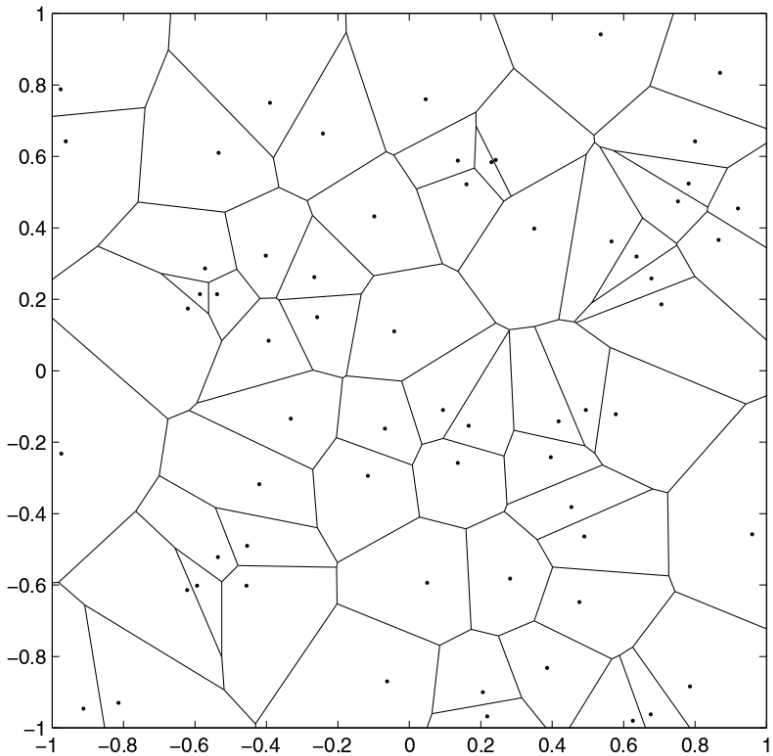
source: wikipedia

Issues with low discrepancy sequences:

- there can be alignments when projected
- there can be holes in subspaces
- points may be aligned (Example: 16 first points in basis (17,18))

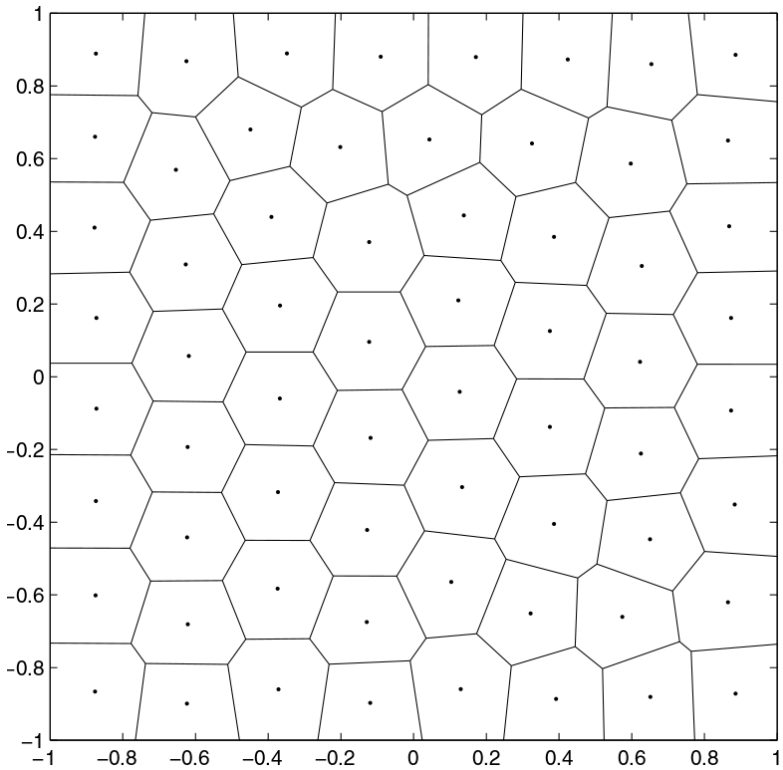
Centroidal Voronoi Tesselations

Given a set of generative points X, the **Voronoi Tesselations** (or Voronoi cells) associated to the point X_i is the region of the space such that X_i is the closest point from the set:



Source: Q. Du et Al., *Centroidal Voronoi Tessellations: Applications and Algorithms*, SIAM Review, 41-4, 1999.

Centroidal Voronoi Tesselations (CVT) is a special case of Voronoi Tesselations where the generative points correspond to the centre of mass of the cells



Source: Q. Du et Al., *Centroidal Voronoi Tessellations: Applications and Algorithms*, SIAM Review, 41-4, 1999.

Properties of CVT:

- Each point of the space is close to one generative points
- The generative points cover the space

$\Rightarrow$ The generative points of CVT can be used as design of experiment.

Generating CVT

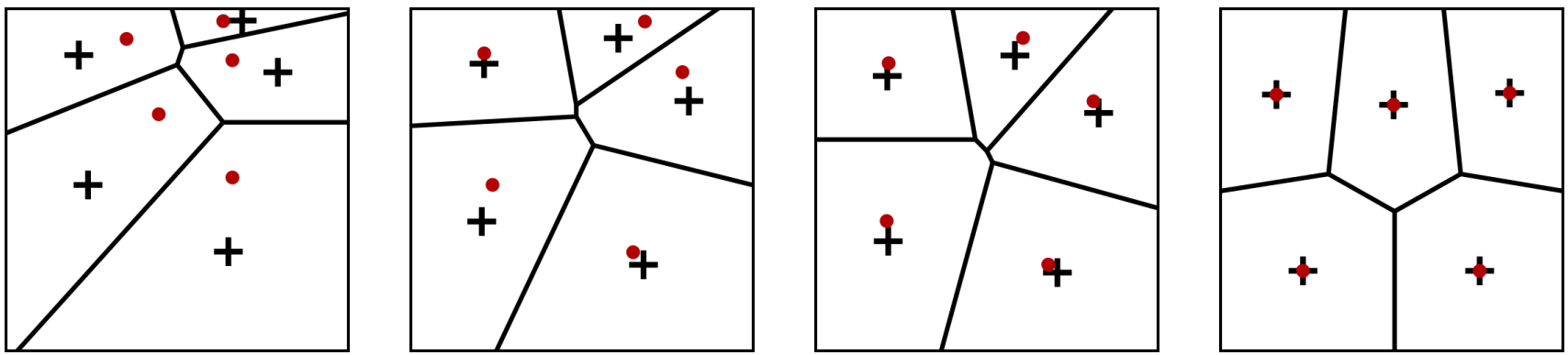
1. Lloyd's Algorithm

1 Initialize X as a set of n points

2 While $i < nb\_iter$

3 Compute the Voronoi diagram of X

4 X = centre of mass of each cell



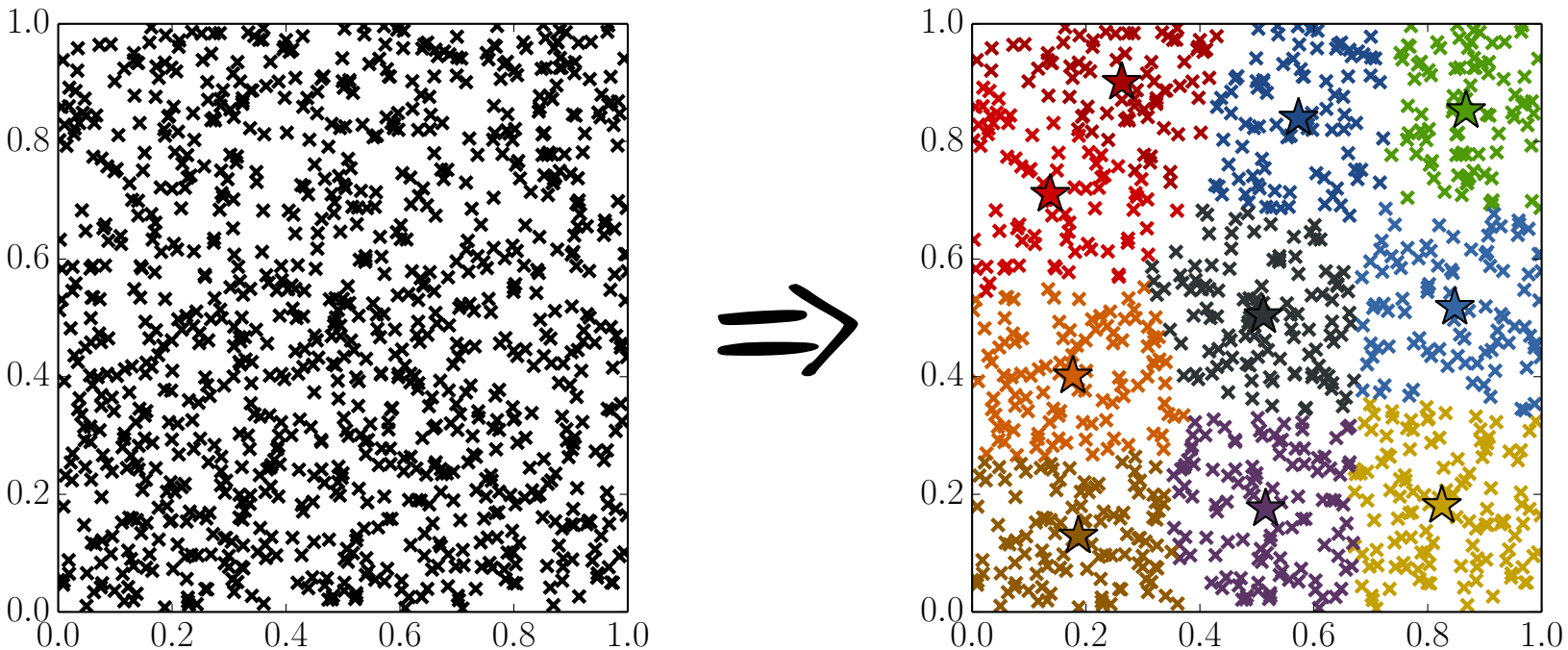
iteration 1 iteration 2 iteration 3 iteration 15

source: "Lloyd's algorithm" wikipedia page

Generating CVT

2. k-means

This algorithm is very similar to Lloyd but it uses a large set of points covering the input space instead of the full continuous domain:



Generating CVT

3. McQueen algorithm

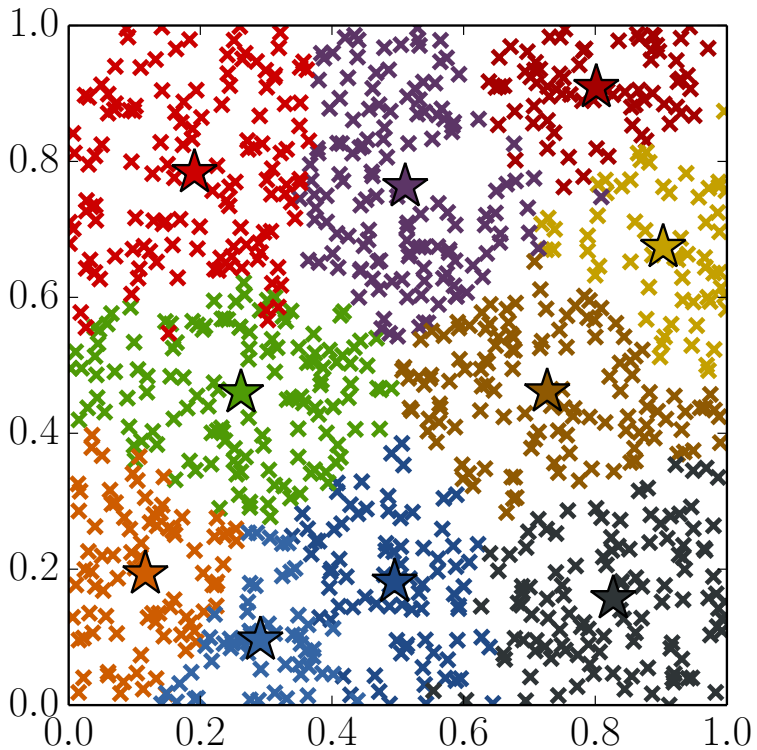
This algorithm is much faster than the previous ones and gives a good approximation

1. Initialize X as a set of n points
2. Initialize k as a vector of 1 with length n
3. While $i < nb\_iter$
4. generate one random point z in the input space
5. find the X_i closest to z
6. update $X_i = \frac{k_i X_i + z}{k_i + 1}$
7. $k_i = k_i + 1$

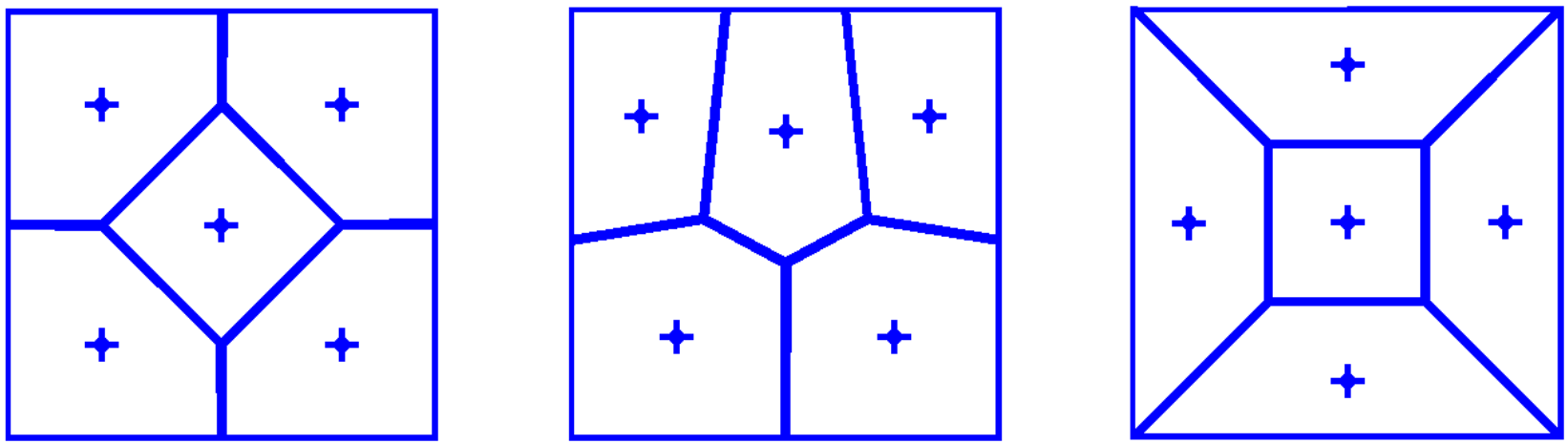
Generating CVT

3. McQueen algorithm

We obtain the following design:



CVT are not unique:



source: wikipedia page “Centroidal Voronoi Tesselations”

Optimal design for regression

Design for regression models

We have seen during yesterday's lecture the expression of the mean and variance of a linear regression model:

$$m(x) = B(x)(B(X)^t B(X))^{-1} B(X)^t F$$
$$v(x) = \sigma^2 B(x)(B(X)^t B(X))^{-1} B(x)^t$$

where B is a set of basis functions, X is the DoE, F is the vector of observations and σ^2 is the variance of the observation noise.

What would be the designs such that:

- $\hat{\beta}$ is a good estimate of β?
- the prediction variance is minimal?

Exercise

We consider a linear regression model over $(0, 1)$ with one basis function $b(x) = x$ and one observation X_1.

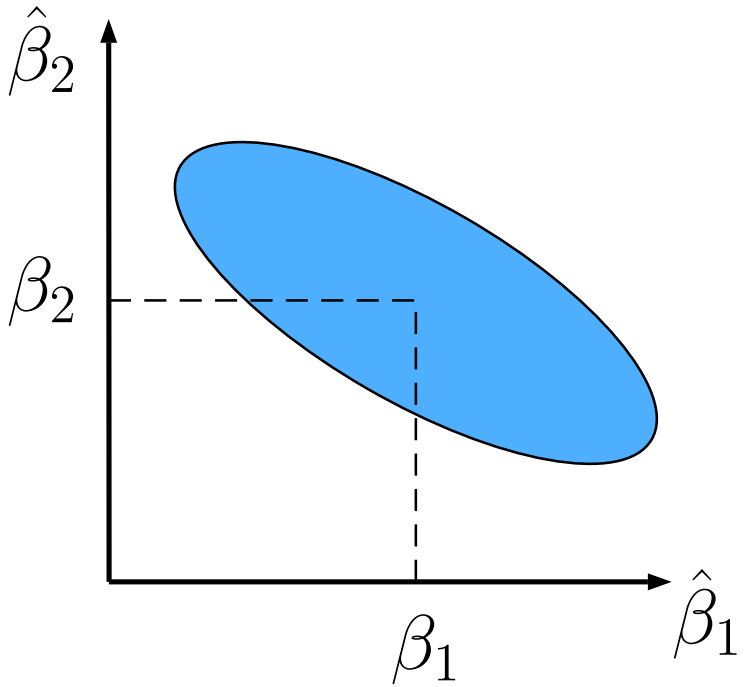
1. Give the expression of m and v.
2. What is the value of X_1 minimizing the maximum of v?
3. What is the value of X_1 minimizing the variance of $\hat{\beta}$?

What happen if we have two basis functions $b_0(x) = 1$, $b_1(x) = x$ and two observations?

From the previous example, we can see that

- Minimising beta variance $\Leftrightarrow$ Minimising prediction variance
- X only has an influence on the term $(B(X)^t B(X))^{-1}$, which is the covariance matrix of $\hat{\beta}$ (up to a σ^2 factor)

We thus want to minimise this uncertainty.



Various criteria for the variability of the estimate:

D-optimality

The volume of the confidence ellipsoid is minimized

$$\min_X \det(B(X)^t B(X))^{-1} = \max \det B(X)^t B(X)$$

A-optimality

The sum of the coefficients variance is minimized

$$\min_X \mathrm{tr}(B(X)^t B(X))^{-1}$$

E-optimality

The maximum eigenvalue of $(B(X)^t B(X))^{-1}$ is minimized

$$\min_X \max_i \lambda_i \qquad \text{(where } \lambda_i \text{ is eigenvalue of } B(X)^t B(X))$$

Various criteria for the prediction variance:

G-optimality

maximum of the prediction variance is minimized

$$\min_X \max_x \sigma^2 B(x)(B(X)^t B(X))^{-1} B(x)^t$$

IMSE-optimality (or I-optimality)

the integrated variance is minimized

$$\min_X \int \sigma^2 B(x)(B(X)^t B(X))^{-1} B(x)^t \, \mathrm{d}x$$

In practice, the optimization of these criterion is difficult:

- Large number of variables ($n \times d$)
- multimodal function (lots of symmetries)

Some algorithms (such as Fedorov) are based on one at a time points replacement:

1. Find the worst point in the Design
2. Find a critic region (large variance)
3. Replace the "bad" point by a point in the critic region

Equivalence theorem (Kiefer and Wolfowitz)

The three conditions are equivalent

- A design is D-optimal
- A design is G-optimal
- The maximum prediction variance is p

Knowing a lower bound allows to define the efficiency of a DoE:

$$G_{eff} = 100 \times \sqrt{\frac{\max_x \sigma^2 B(x)(B(X)^t B(X))^{-1} B(x)^t}{p}}$$

Optimal design for Gaussian process regression

A GP Z with covariance k can be decomposed as a sum of two independent GPs:

$$Z(x) = \underbrace{k(x,X)k(X,X)^{-1}Z(X)}_{Z_X(x)} + \underbrace{Z(x) - k(x,X)k(X,X)^{-1}Z(X)}_{Z_{X^\perp}(x)}$$

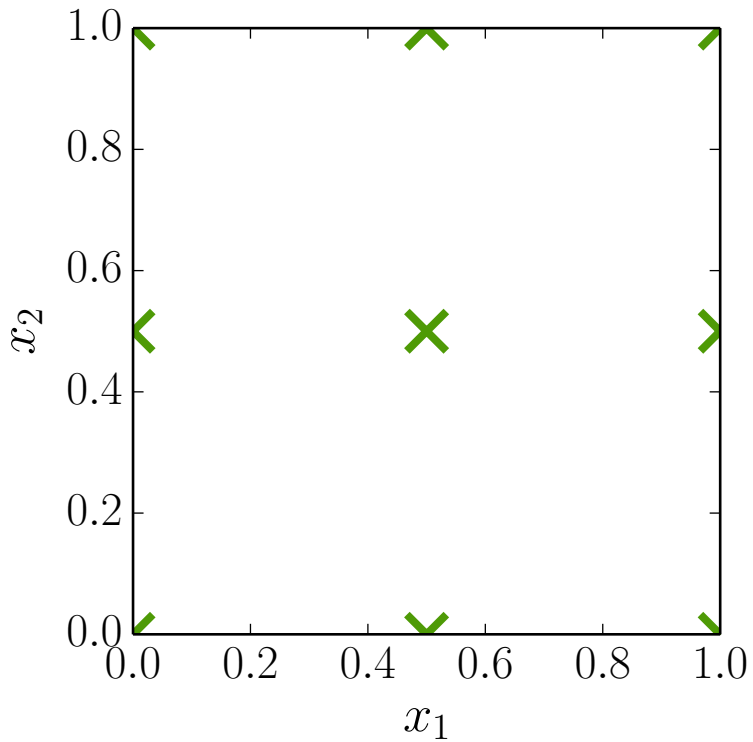
$$k(x,y) = \underbrace{k(x,X)k(X,X)^{-1}k(X,y)}_{k_X(x,y)} + \underbrace{k(x,y) - k_X(x,y)}_{k_{X^\perp}(x,y)}$$

In order to capture most of the variability of Z, we can:

- Maximize the variability of $Z_C(X)$ and apply previous D/A/E-optimality criterion to $k(X,X)$ instead of $B(X)^t B(X)$.
- Minimize the prediction error: I/G-optimality to $k_{X^\perp}(x,x)$.

known covariance parameters

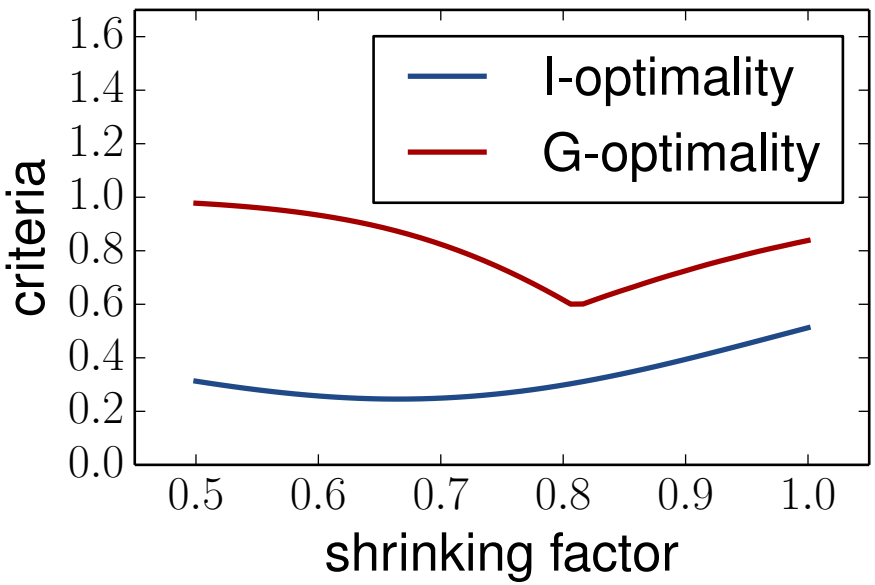
If we maximise the determinant of $k(X, X)$ for a 9 points DoE on $(0, 1)^2$, we find the following design:

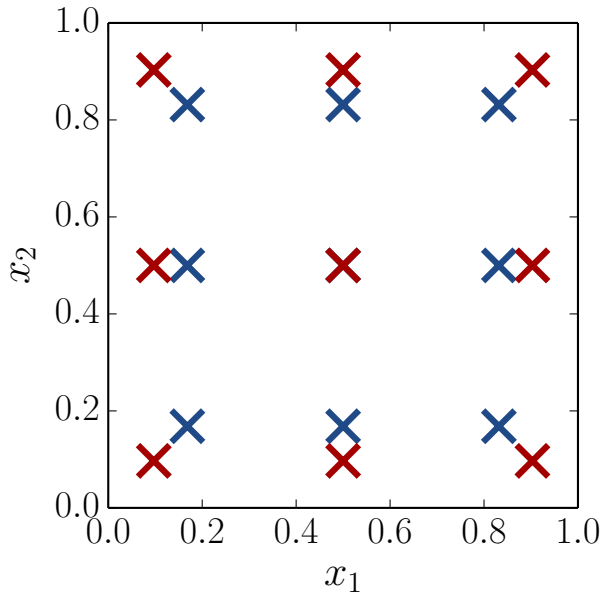


however, this design is not I-optimal nor G-optimal...

known covariance parameters

We can compute numerically the optimal shrinking factor:





$\Rightarrow$ They all give a different optimal DoE.

Small recap on optimal design for GPR

- All criteria are difficult to compute
- The optimization problem is tricky
- We don't have strong theoretical results as in regression

Good practice:

- Use of space filling designs
- Optimization inside a class of DoE

Remark: IMSE is more correlated to minimax than maximin

Adaptive Designs

The principle of **adaptive design** is to add the points in the design one after each other.

$\rightarrow$ the $n \times d$-dimensional optimisation is transformed into n optimisations in d dimensions.

This is still expensive

One new model has to be built for each candidate point.
Furthermore, for each candidate model:

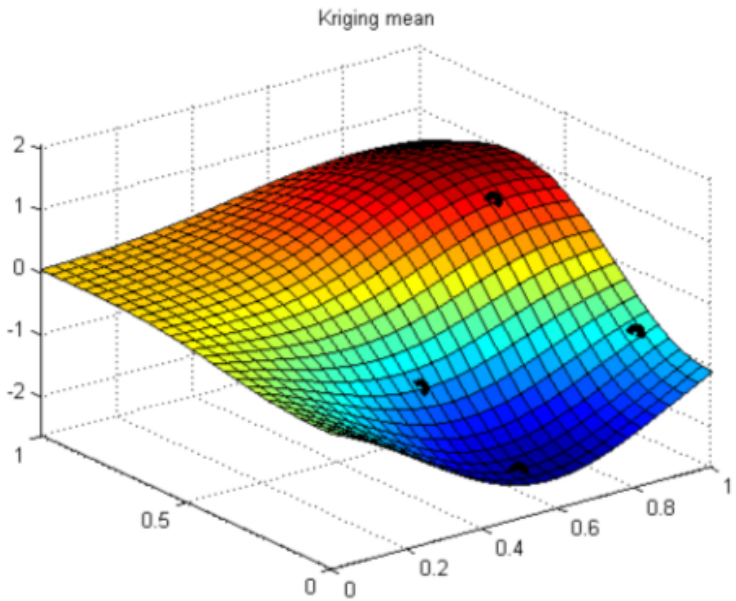
- I-optimality requires to compute a high dimensional integral
- G-optimality requires to optimize the variance

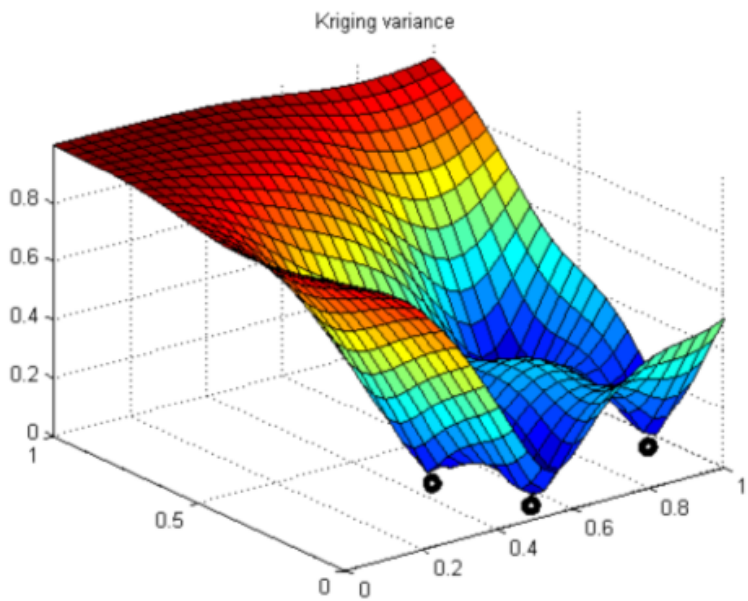
An approximation that is not computationaly expensive is to add the new point where the model variance is maximum:

Algorithm

1. Build an initial DoE X with k points
2. While $i < n - k$
3. find $x^* = \mathrm{argmax}(c(x, x))$
4. add x^* to the design and recompute $c(x, x)$

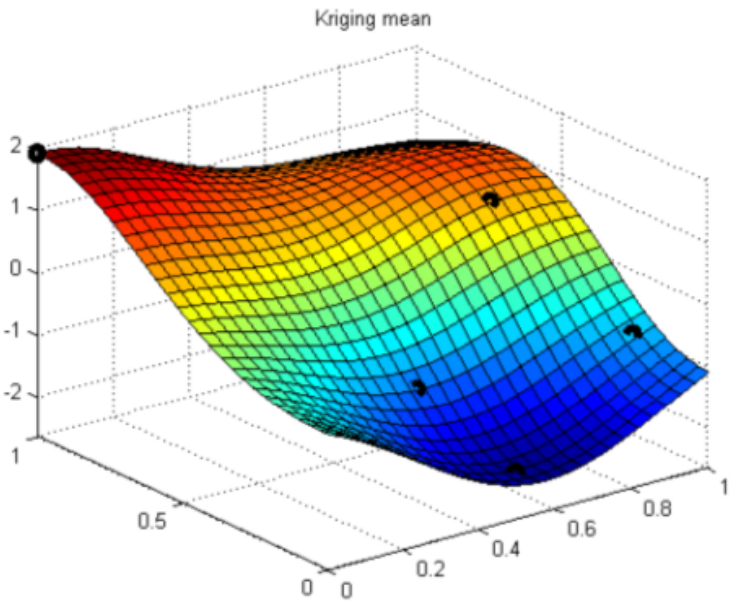
- IMSE = 0.5985
- maxMSE = 0.9991

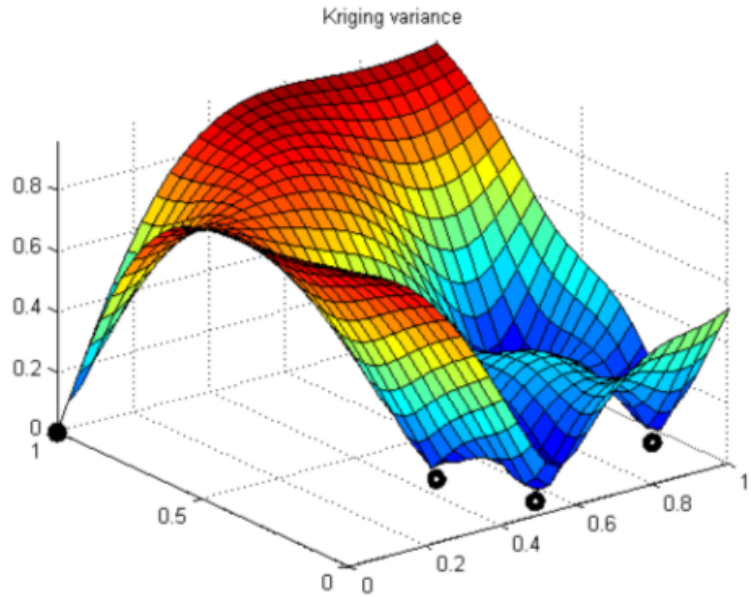




source: Lecture from V. Picheny at Mines St-Etienne

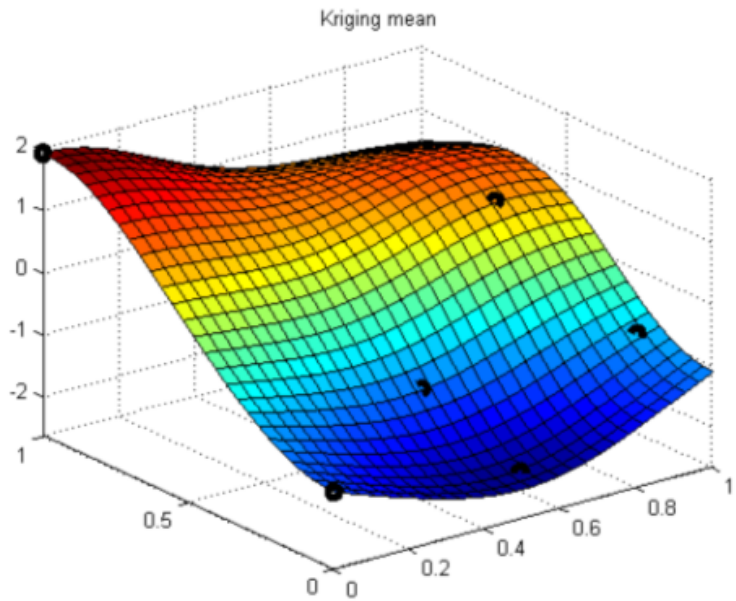
- IMSE = 0.5462
- maxMSE = 0.9665

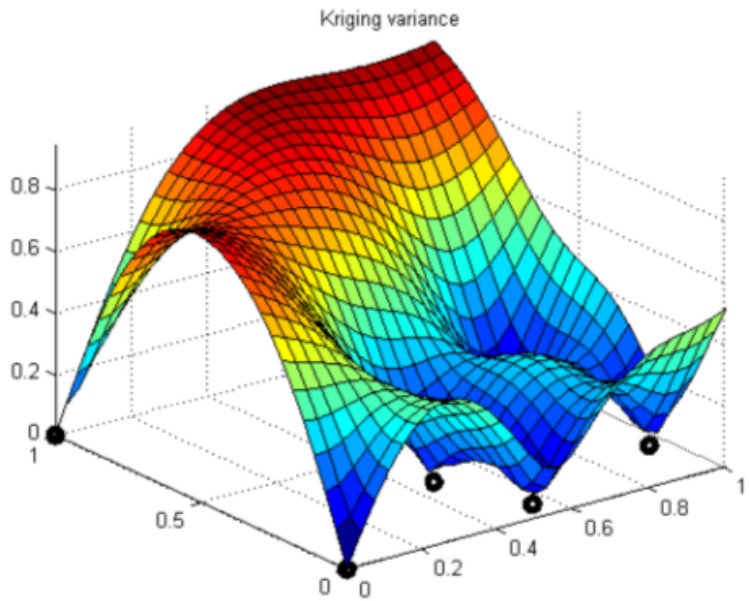




source: Lecture from V. Picheny at Mines St-Etienne

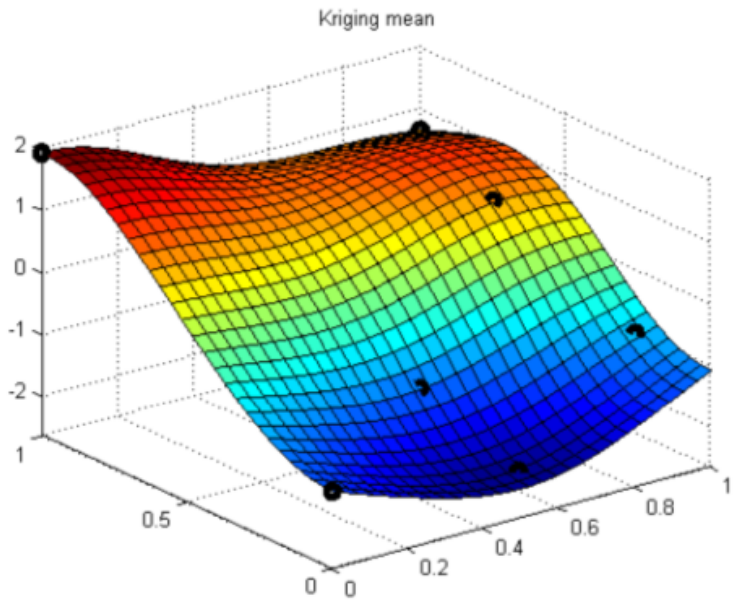
- IMSE = 0.5011
- maxMSE = 0.9466

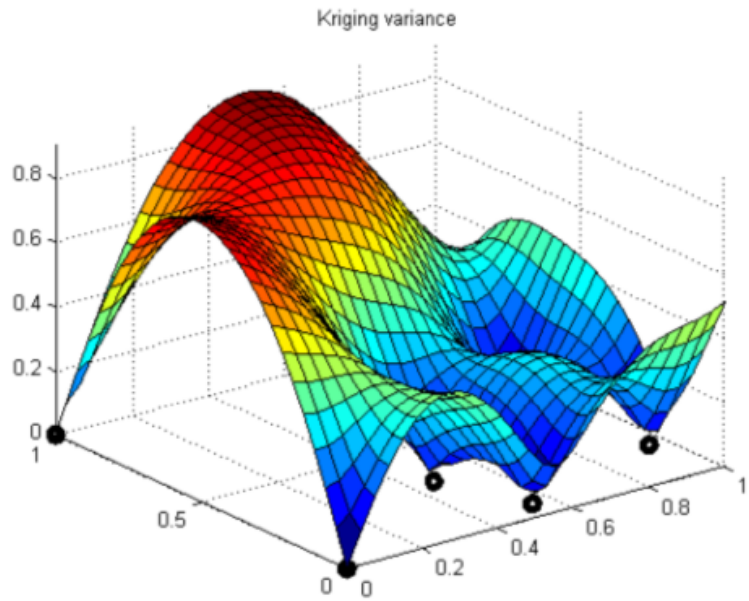




source: Lecture from V. Picheny at Mines St-Etienne

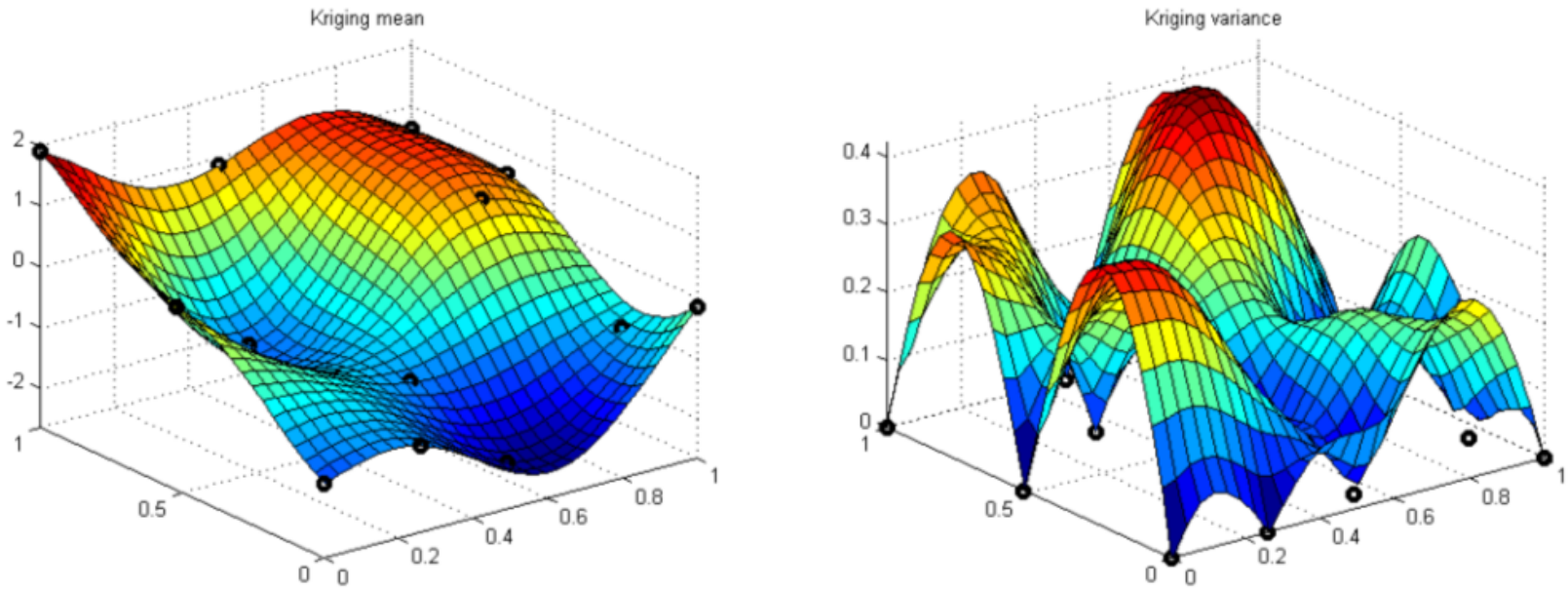
- IMSE = 0.4619
- maxMSE = 0.9035





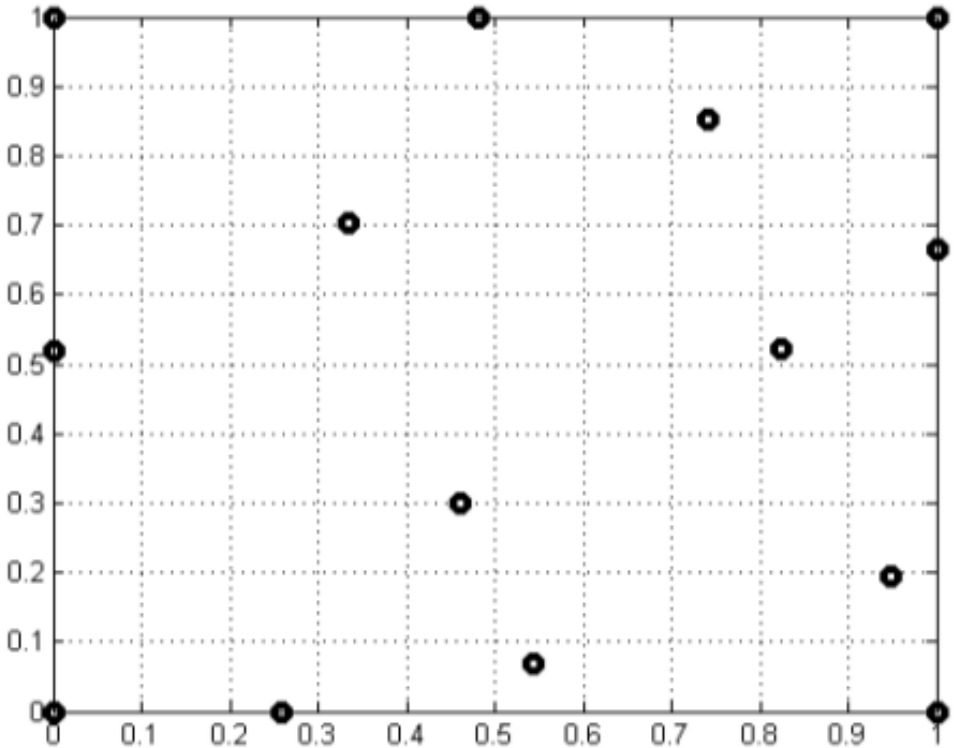
source: Lecture from V. Picheny at Mines St-Etienne

- IMSE = 0.2009
- maxMSE = 0.4226



source: Lecture from V. Picheny at Mines St-Etienne

We end-up with the following design:



It has:

+ Good space filling properties
− Too much points on the boundaries

Parameter estimation

We have seen previously that the choice of the kernel and its parameters have a great influence on the model.

In order to choose a prior that is suited to the data at hand, we can consider:

- minimising the model error
- Using maximum likelihood estimation

We will now detail the second one.

Definition

The **likelihood** of a distribution with a density f_X given some observations $X_1, \ldots, X_p$ is:

$$L = \prod_{i=1}^{p} f_X(X_i)$$

This quantity can be used to measure the adequacy between observations and a distribution.

In the GPR context, we often have only **one observation** of the vector F. The likelihood is then:

$$L = f_{Z(X)}(F) = \frac{1}{(2\pi)^{n/2}|k(X,X)|^{1/2}} \exp\left(-\frac{1}{2}F^t k(X,X)^{-1}F\right).$$

It is thus possible to maximise L – or $\log(L)$ – with respect to the kernel's parameters in order to find a well suited prior.

Example

We consider 100 sample from a Matern $5/2$ process with parameters $\sigma^2 = 1$ and $\theta = 0.2$, and n observation points. We then try to recover the kernel parameters using MLE.

n	5	10	15	20
σ^2	1.0 (0.7)	1.11 (0.71)	1.03 (0.73)	0.88 (0.60)
θ	0.20 (0.13)	0.21 (0.07)	0.20 (0.04)	0.19 (0.03)

MLE can be applied regardless to the dimension of the input space.

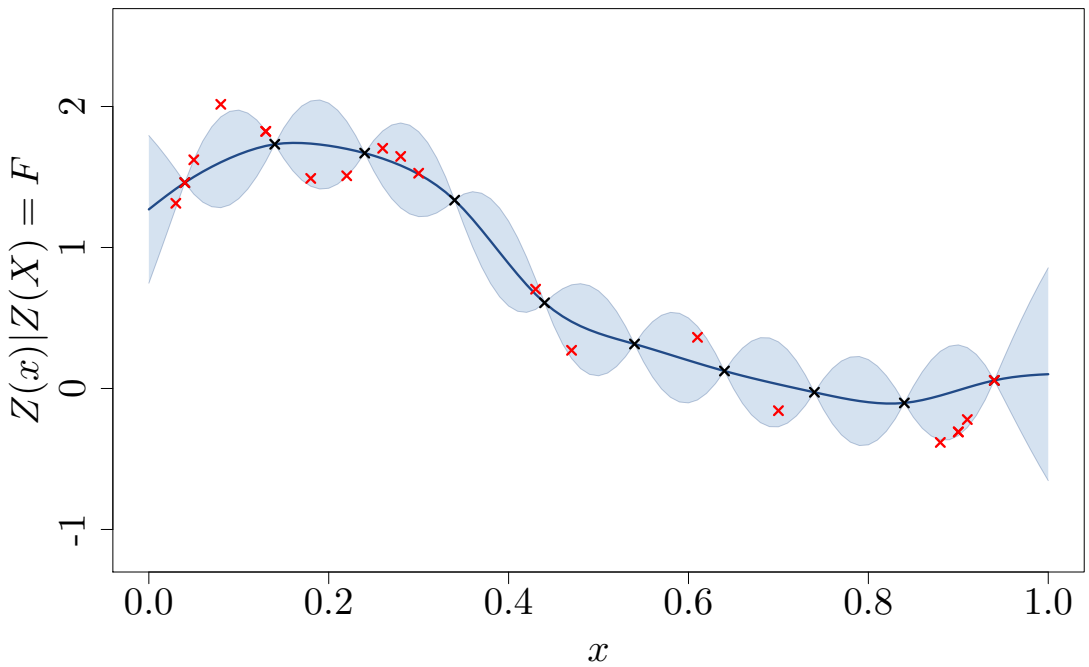
Model validation

We have seen that given some observations $F = f(X)$, it is very easy to build lots of models, either by changing the kernel parameters or the kernel itself.

The interesting question now is to know how to get a good model. To do so, we will need to answer the following questions:

- What is a good model?
- How to measure it?

The idea is to introduce new data and to compare the model prediction with reality



Since GPR models provide a mean and a covariance structure for the error they both have to be assessed.

Let X_t be the test set and $F_t = f(X_t)$ be the associated observations.

The accuracy of the mean can be measured by computing:

$$\text{Mean Square Error} \qquad MSE = \text{mean}((F_t - m(X_t))^2)$$

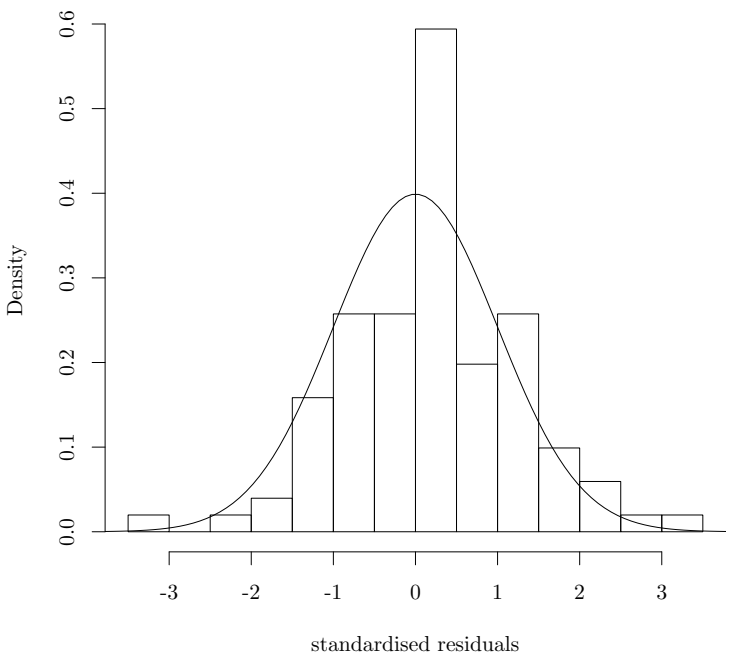
$$\text{A "normalised" criterion} \qquad Q_2 = 1 - \frac{\sum (F_t - m(X_t))^2}{\sum (F_t - \text{mean}(F_t))^2}$$

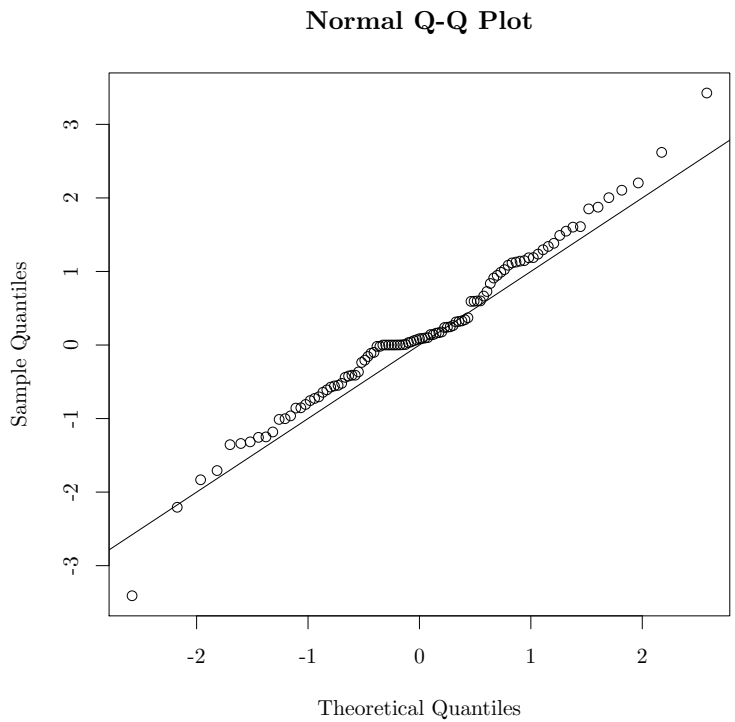
On the above example we get $MSE = 0.038$ and $Q_2 = 0.95$.

The predicted distribution can be tested by normalising the residuals.

According to the model, $F_t \sim \mathcal{N}(m(X_t), c(X_t, X_t))$.

$c(X_t, X_t)^{-1/2}(F_t - m(X_t))$ should thus be independents $\mathcal{N}(0, 1)$:





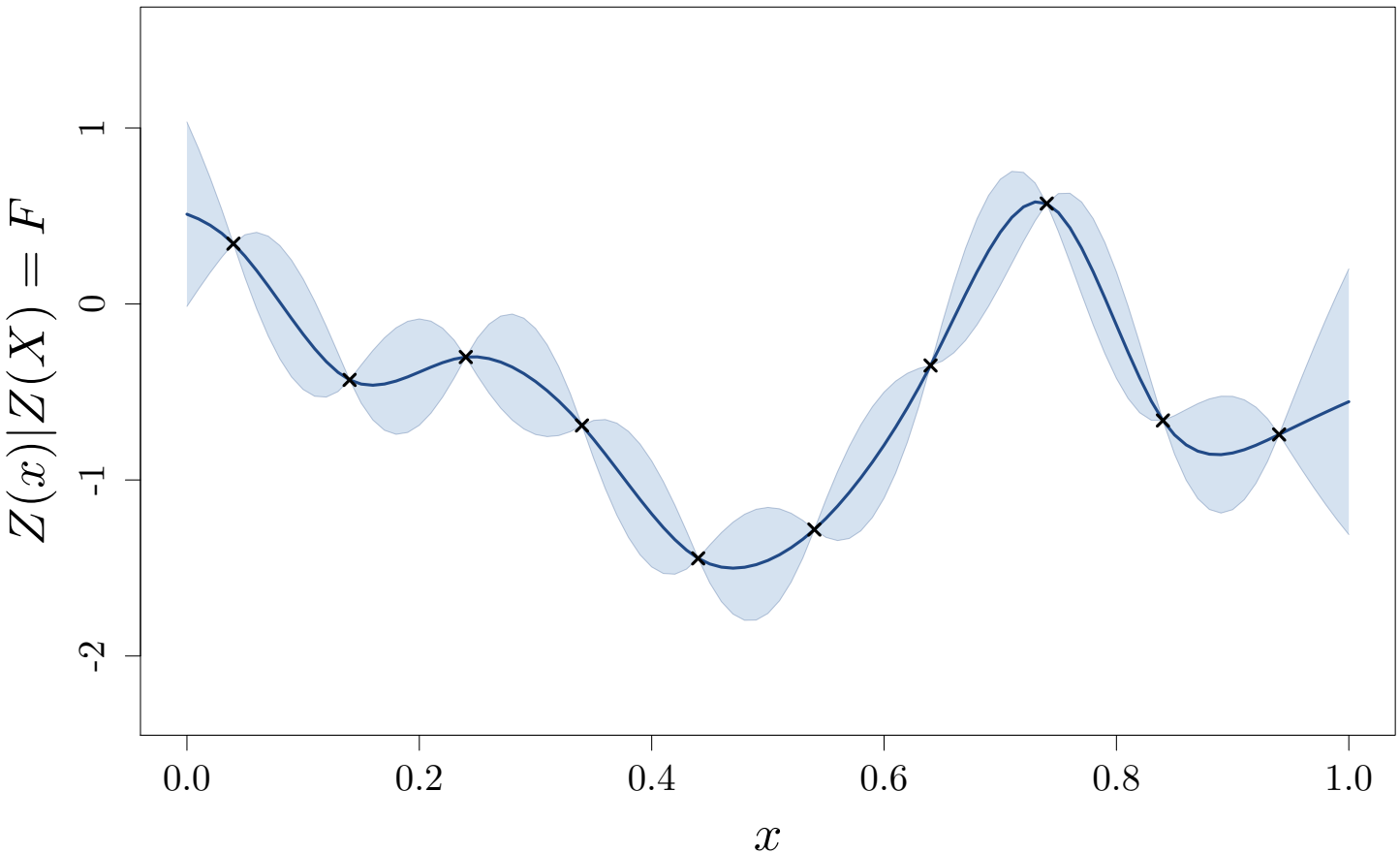
When no test set is available, another option is to consider cross validation methods such as leave-one-out.

The steps are:

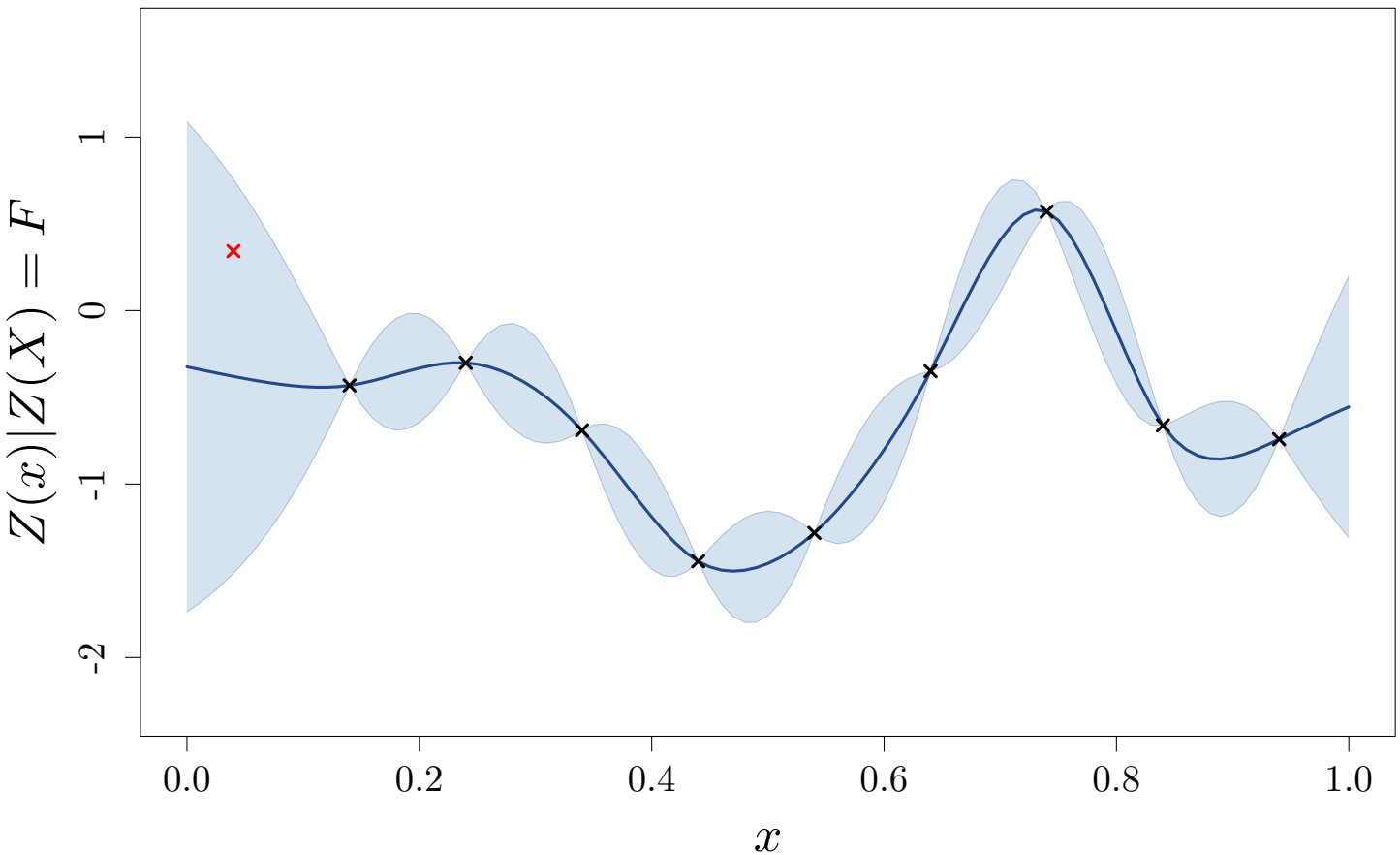
1. build a model based on all observations except one
2. compute the model error at this point

This procedure can be repeated for all the design points in order to get a vector of error.

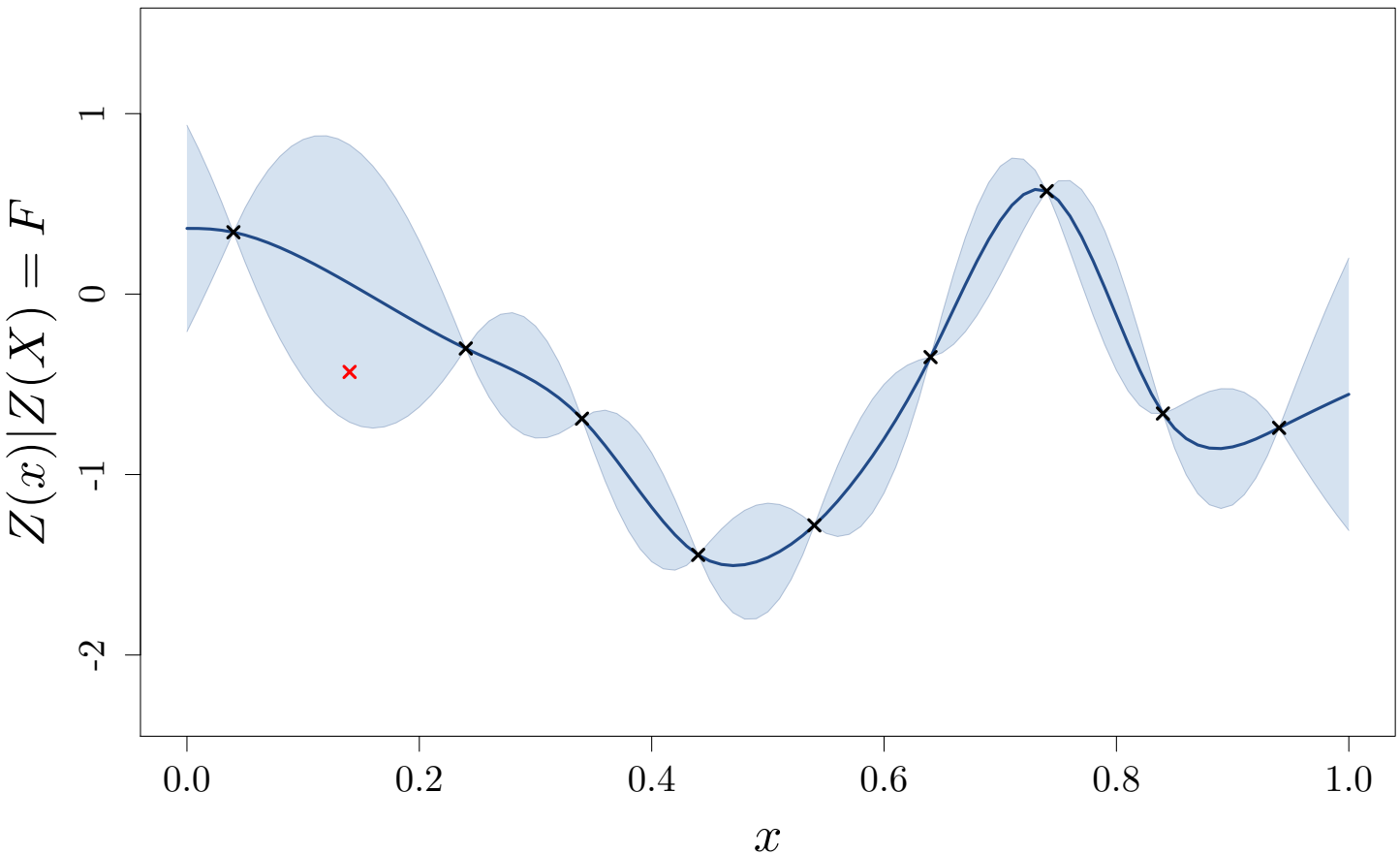
Model to be tested:



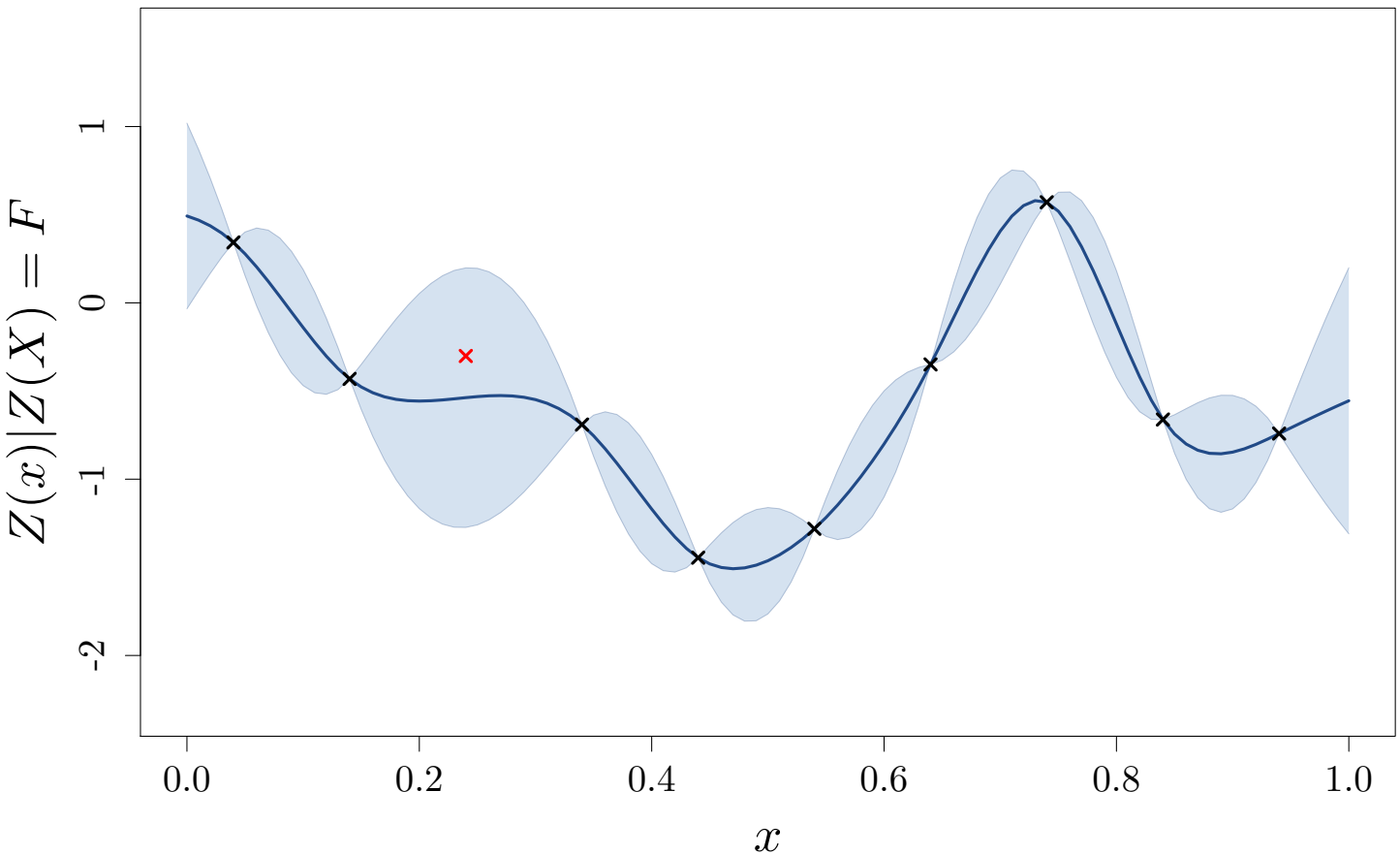
Step 1:



Step 2:



Step 3:



If we apply LOO to previous example we obtain:

$$MSE = 0.24 \text{ and } Q_2 = 0.34.$$

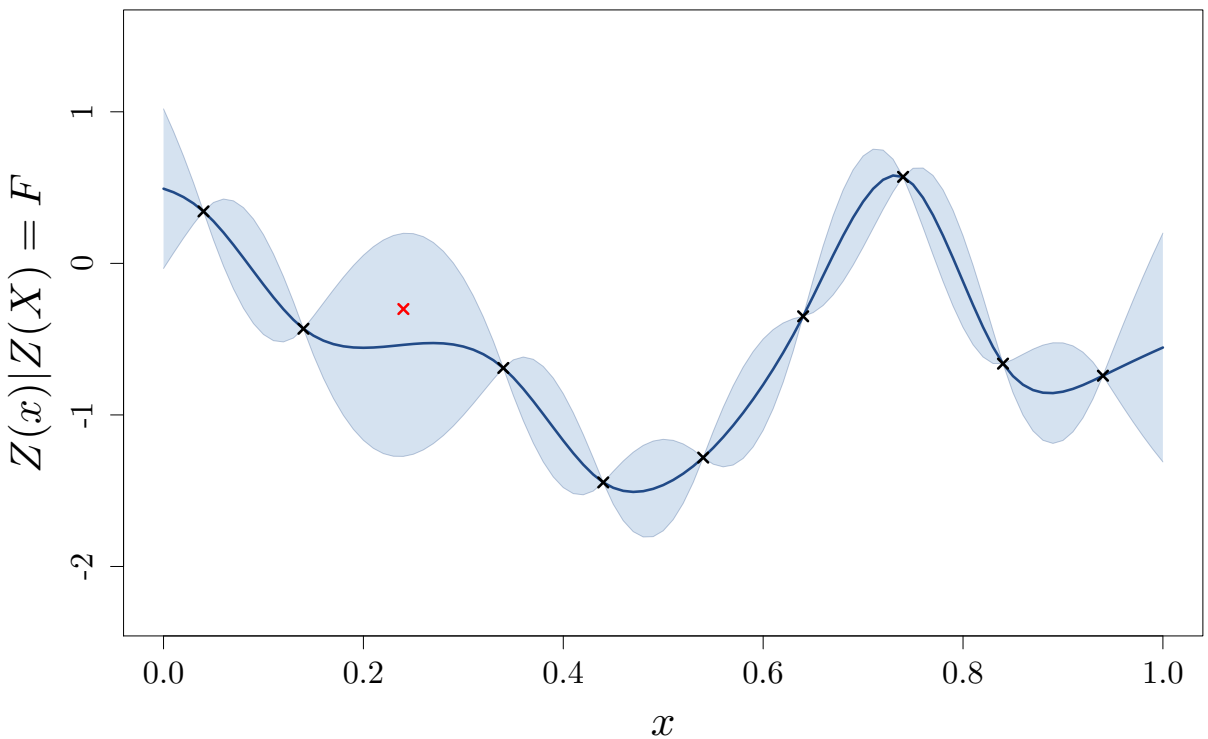
Why doesn't the model perform as good as previously?

If we apply LOO to previous example we obtain:

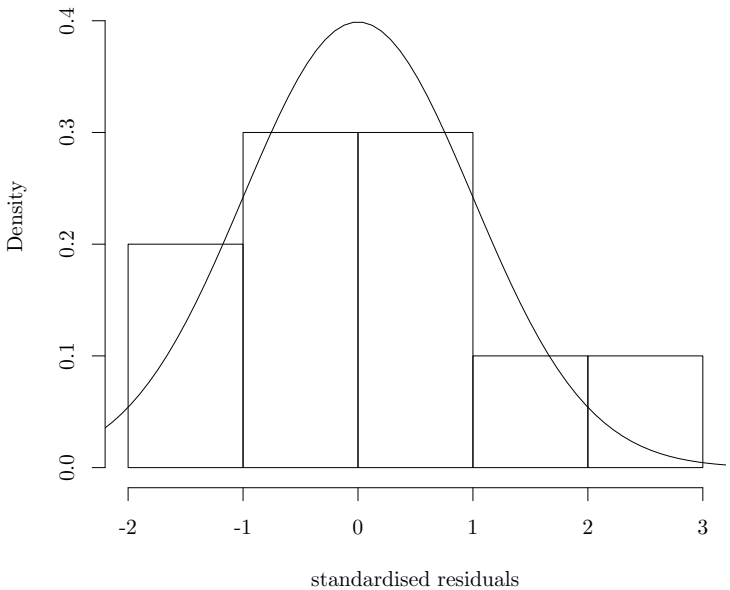
$$MSE = 0.24 \text{ and } Q_2 = 0.34.$$

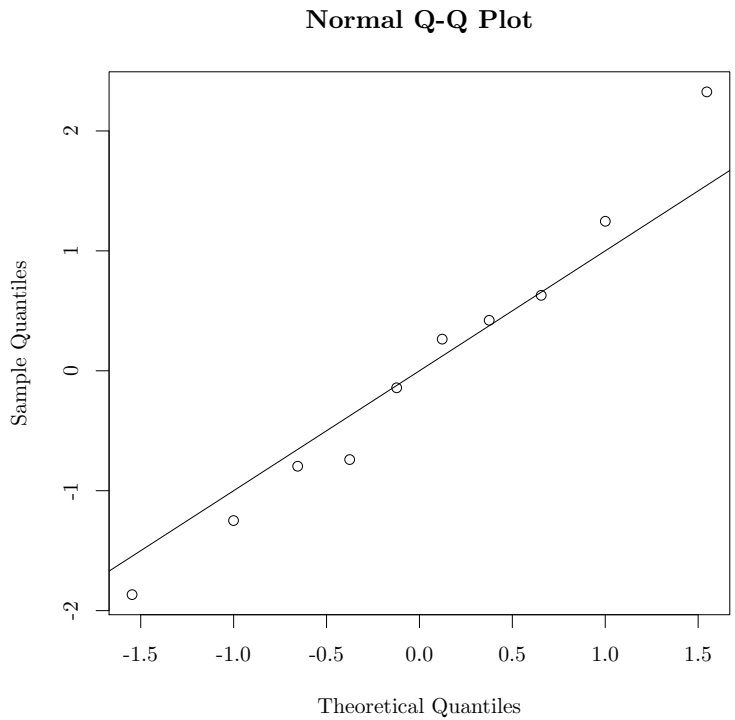
Why doesn't the model perform as good as previously?

It turns out that errors are always computed at the 'worst' location!



We can also look at the residual distribution. For leave-one-out, there is no joint distribution for the residuals so they have to be standardised independently. We obtain:





Making new from old

Making new from old: Many operations can be applied to psd functions while retaining this property

Kernels can be:

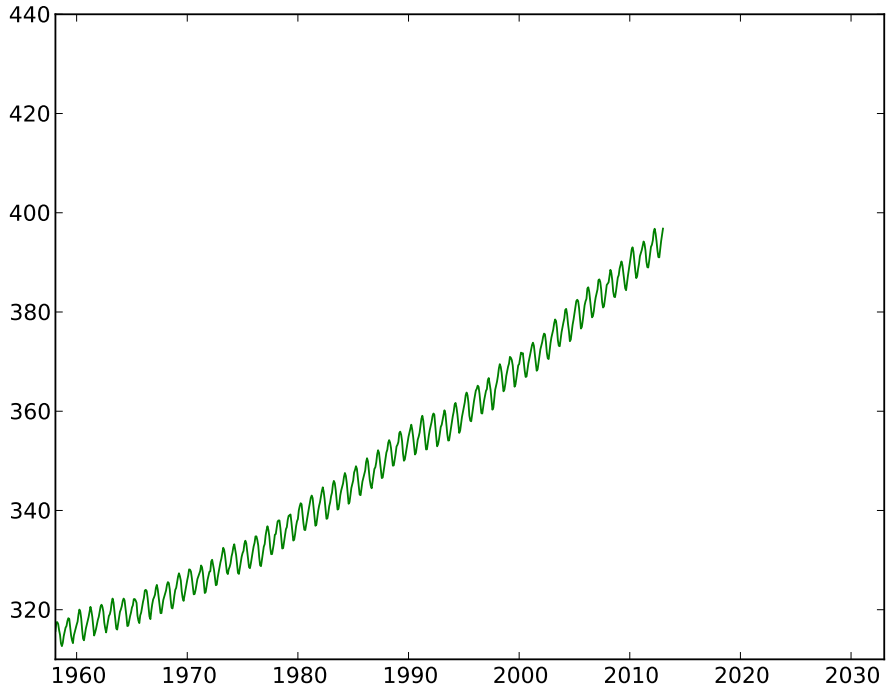
- Summed together
  - On the same space $k(x, y) = k_1(x, y) + k_2(x, y)$
  - On the tensor space $k(x, y) = k_1(x_1, y_1) + k_2(x_2, y_2)$
- Multiplied together
  - On the same space $k(x, y) = k_1(x, y) \times k_2(x, y)$
  - On the tensor space $k(x, y) = k_1(x_1, y_1) \times k_2(x_2, y_2)$
- Composed with a function
  - $k(x, y) = k_1(f(x), f(y))$

How can this be useful?

Sum of kernels over the same space

Example (The Mauna Loa observatory dataset)

This famous dataset compiles the monthly CO_2 concentration in Hawaii since 1958.

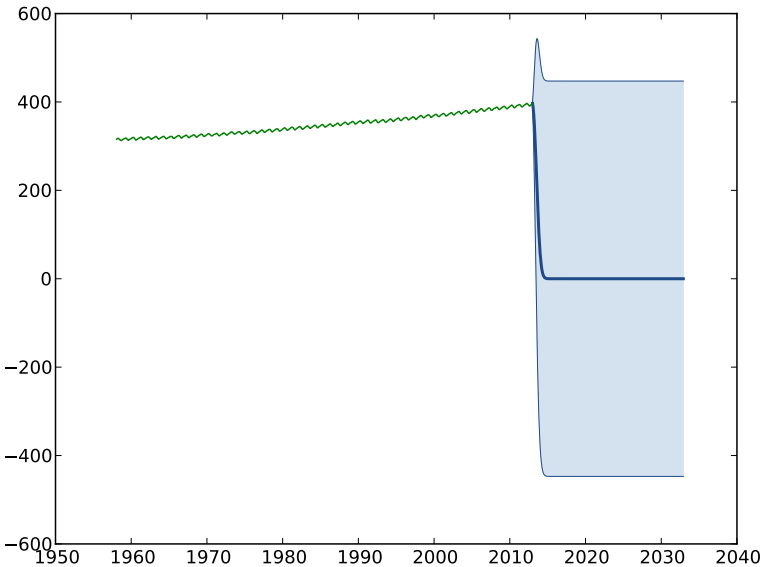


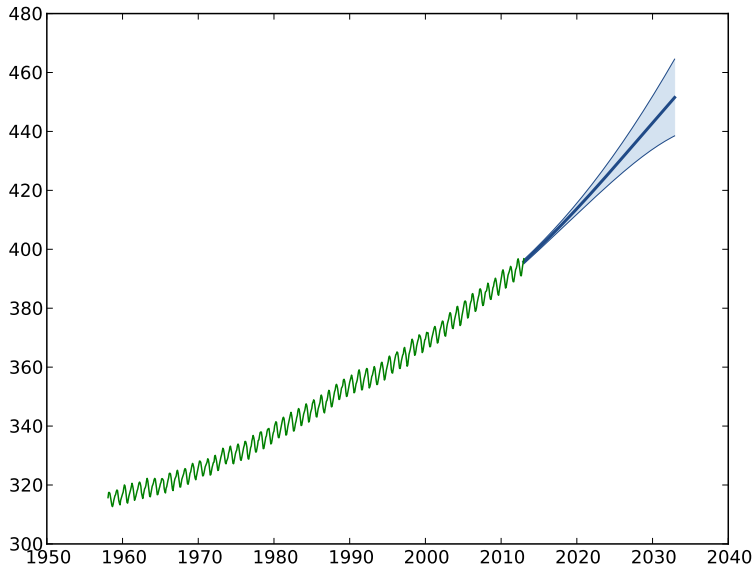
Let's try to predict the concentration for the next 20 years.

Sum of kernels over the same space

We first consider a squared-exponential kernel:

$$k(x, y) = \sigma^2 \exp\left(-\frac{(x-y)^2}{\theta^2}\right)$$





The results are terrible!

Sum of kernels over the same space

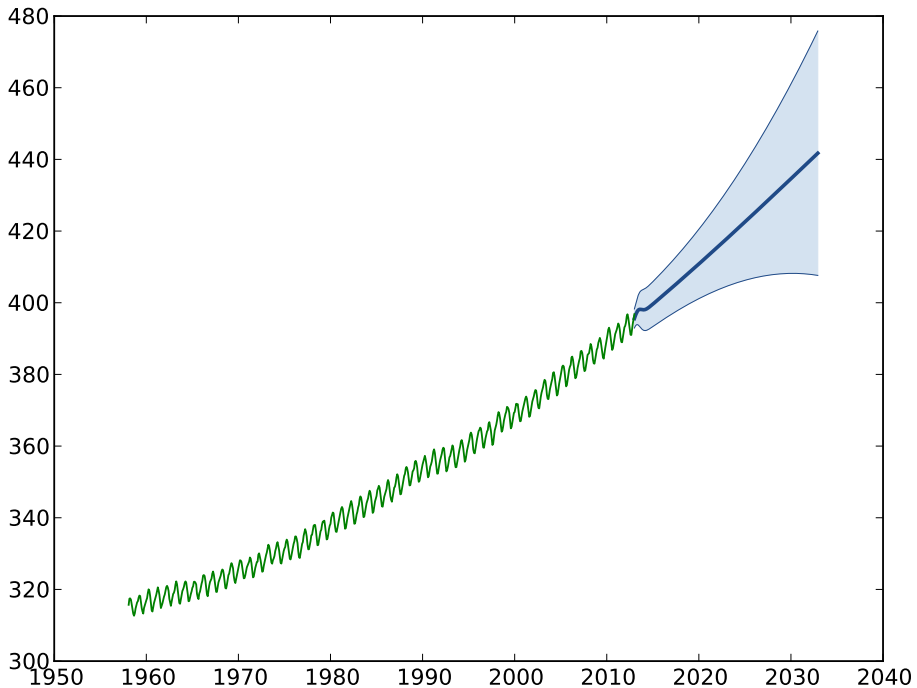
What happen if we sum both kernels?

$$k(x, y) = k_{rbf1}(x, y) + k_{rbf2}(x, y)$$

Sum of kernels over the same space

What happen if we sum both kernels?

$$k(x, y) = k_{rbf1}(x, y) + k_{rbf2}(x, y)$$



The model is drastically improved!

Sum of kernels over the same space

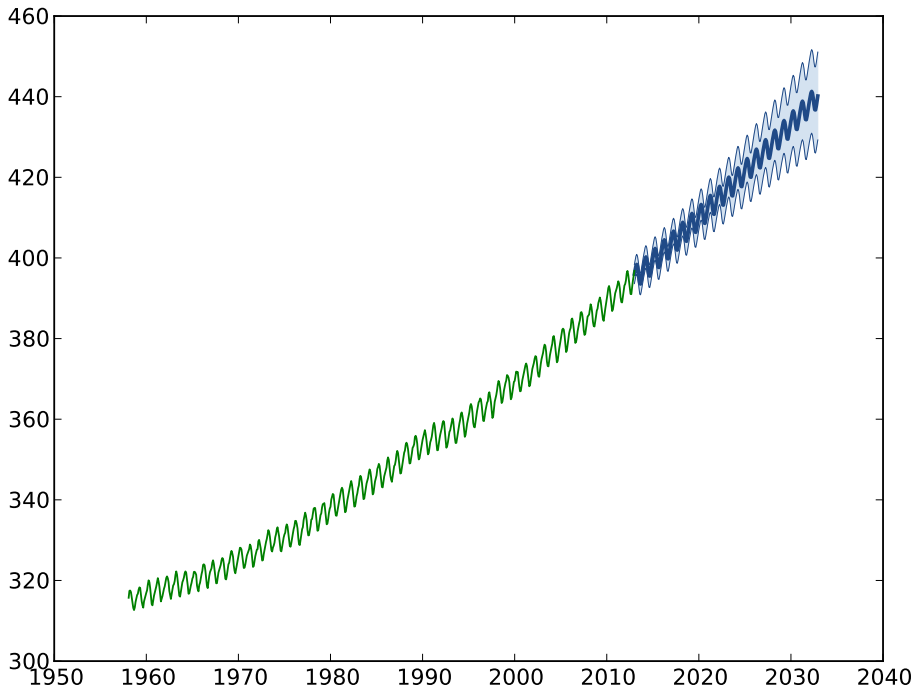
We can try the following kernel:

$$k(x, y) = \sigma_0^2 x^2 y^2 + k_{rbf1}(x, y) + k_{rbf2}(x, y) + k_{per}(x, y)$$

Sum of kernels over the same space

We can try the following kernel:

$$k(x, y) = \sigma_0^2 x^2 y^2 + k_{rbf1}(x, y) + k_{rbf2}(x, y) + k_{per}(x, y)$$



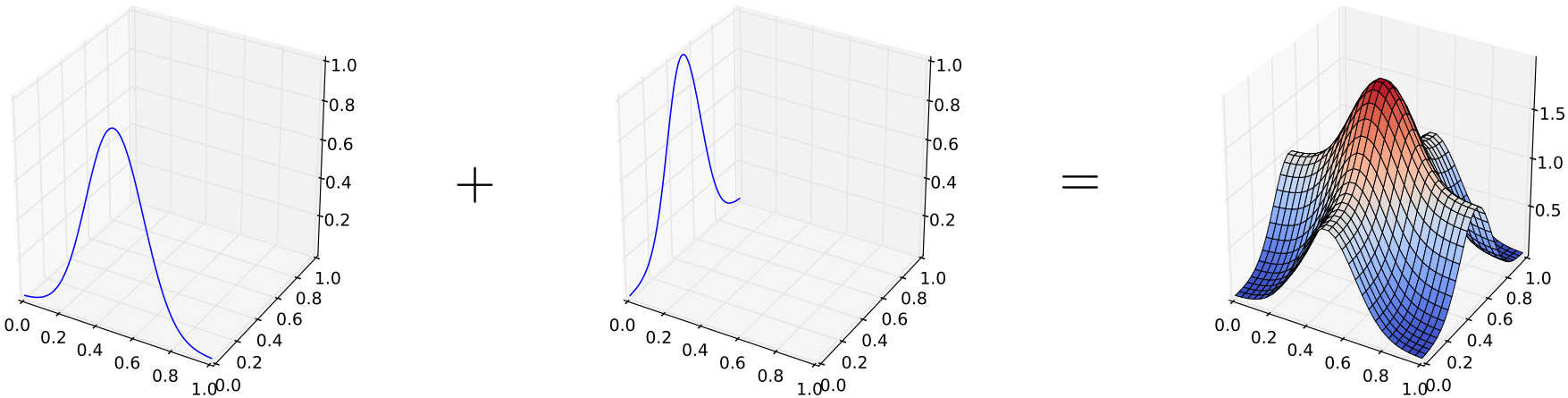
Once again, the model is significantly improved.

Sum of kernels over tensor space

Property

$$k(x, y) = k_1(x_1, y_1) + k_2(x_2, y_2)$$

is valid covariance structure.

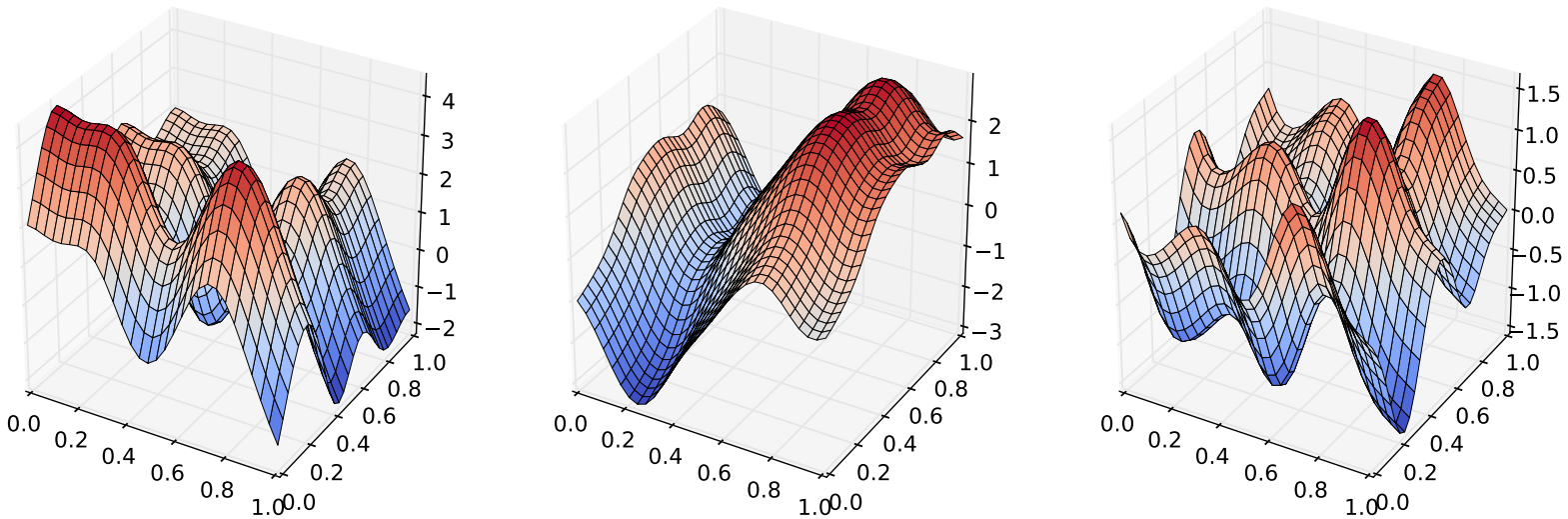


Remark: From a GP point of view, k is the kernel of

$$Z(x) = Z_1(x_1) + Z_2(x_2)$$

Sum of kernels over tensor space

We can have a look at a few sample paths from Z:



$\Rightarrow$ They are additive (up to a modification)

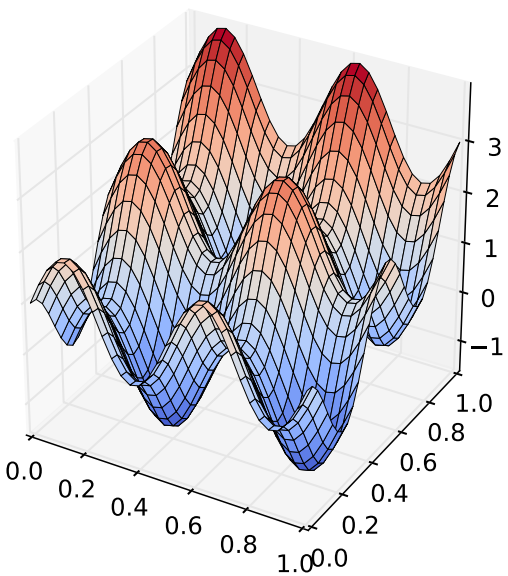
Tensor Additive kernels are very useful for

- Approximating additive functions
- Building models over high dimensional inputs spaces

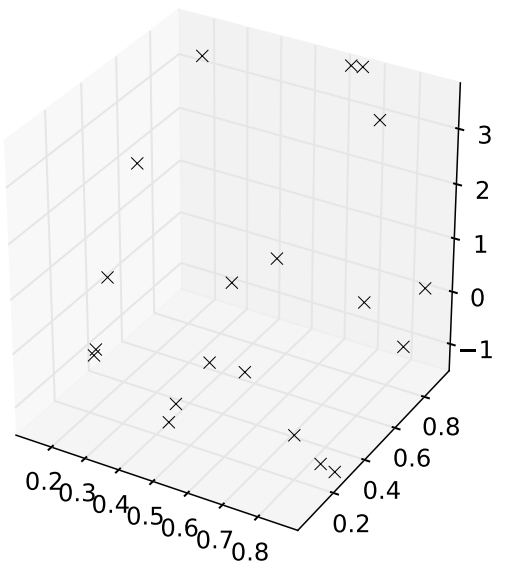
Sum of kernels over tensor space

We consider the test function $f(x) = \sin(4\pi x_1) + \cos(4\pi x_2) + 2x_2$ and a set of 20 observation in $[0, 1]^2$.

Test function



Observations

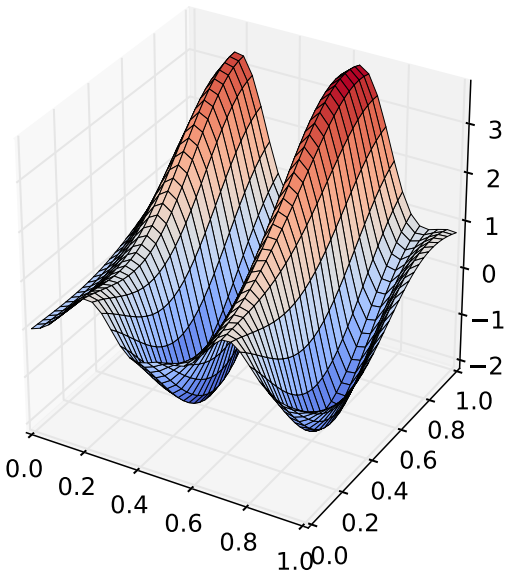


Sum of kernels over tensor space

We obtain the following models:

Gaussian kernel

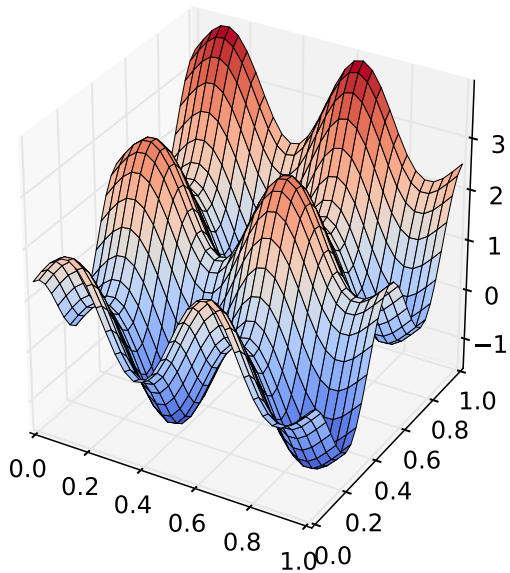
Mean predictor



RMSE is 1.06

Additive Gaussian kernel

Mean predictor



RMSE is 0.12

Sum of kernels over tensor space

Remark

- It is straightforward to show that the mean predictor is additive

$$\begin{aligned} m(x) &= (k_1(x_1, X_1) + k_2(x_2, X_2))k(X, X)^{-1}F \\ &= \underbrace{k_1(x_1, X_1)k(X, X)^{-1}F}_{m_1(x_1)} + \underbrace{k_2(x_2, X_2)k(X, X)^{-1}F}_{m_2(x_2)} \end{aligned}$$

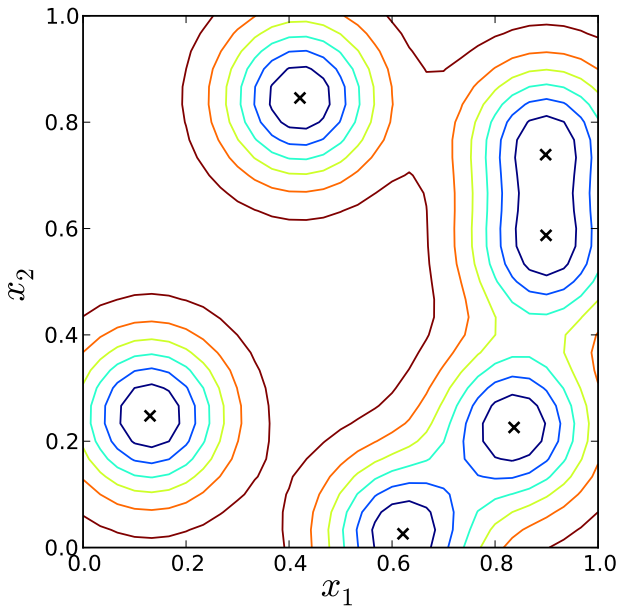
$\Rightarrow$ The mean predictor shares the prior behaviour.

Sum of kernels over tensor space

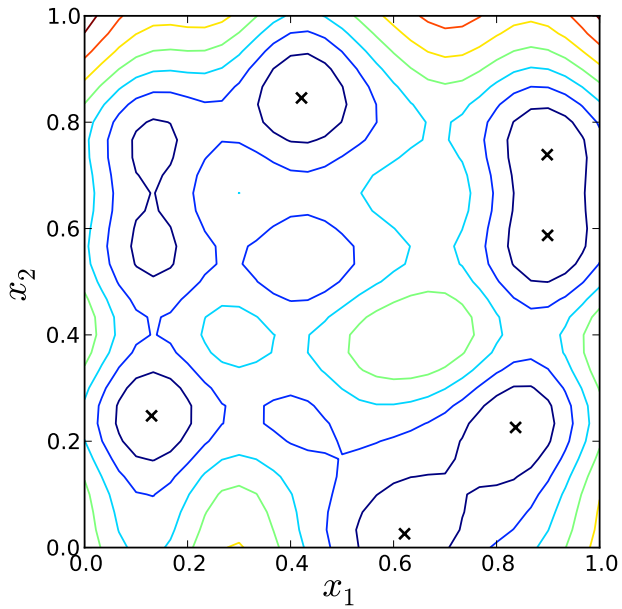
Remark

- The prediction variance has interesting features

pred. var. with kernel product

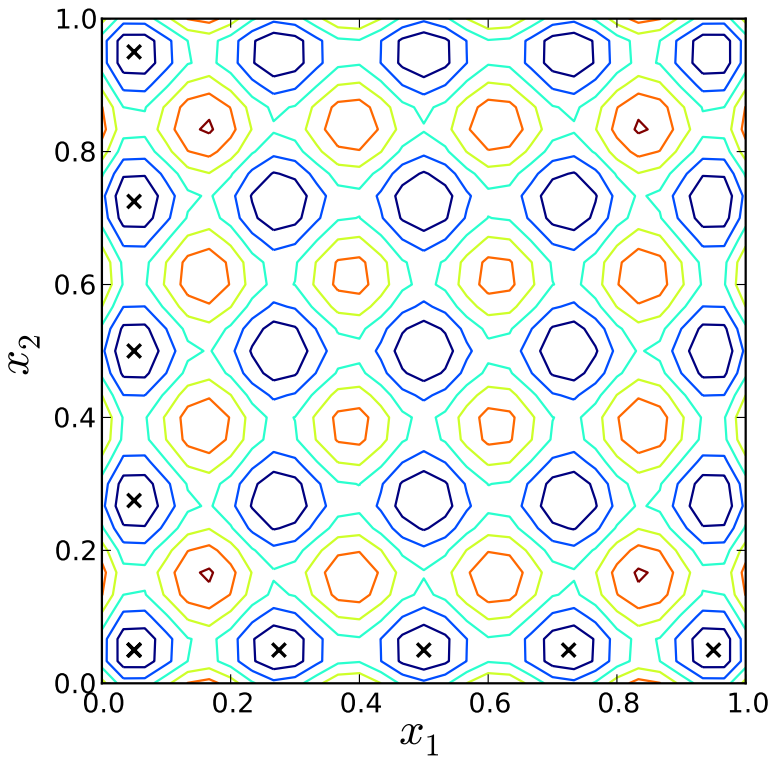


pred. var. with kernel sum



Sum of kernels over tensor space

This property can be used to construct a design of experiment that covers the space with only $cst \times d$ points.



Prediction variance

Product over the same space

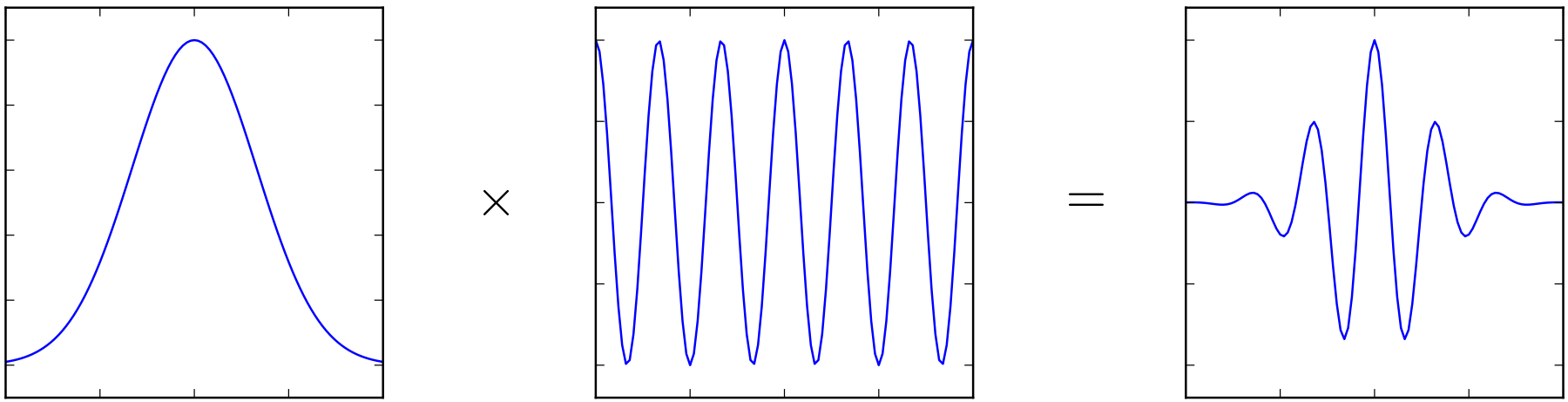
Property

$$k(x, y) = k_1(x, y) \times k_2(x, y)$$

is valid covariance structure.

Example

We consider the product of a squared exponential with a cosine:



Product over the tensor space

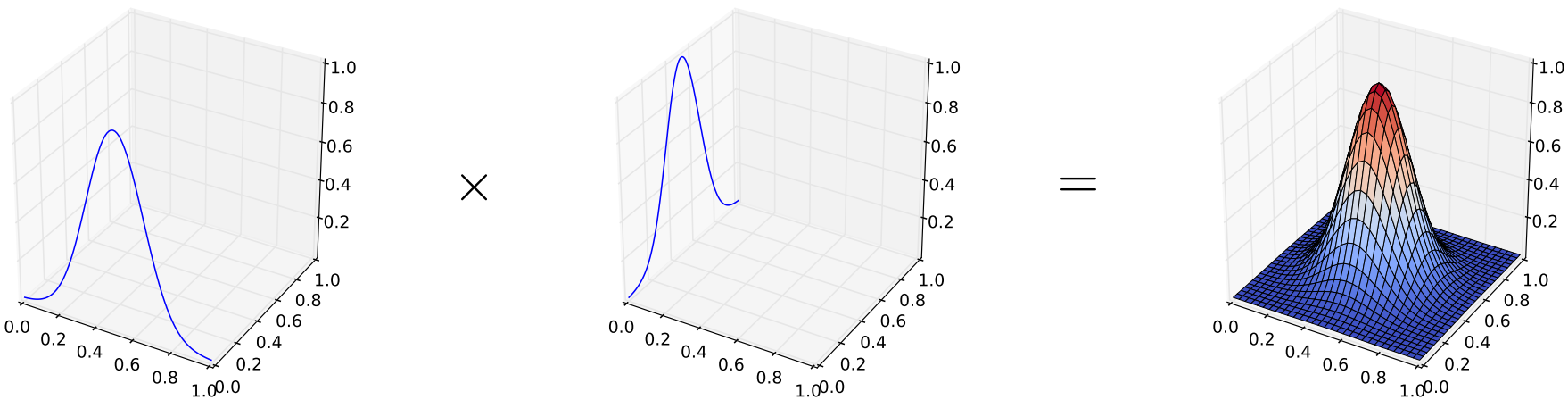
Property

$$k(x, y) = k_1(x_1, y_1) \times k_2(x_2, y_2)$$

is valid covariance structure.

Example

We multiply 2 squared exponential kernel



Calculation shows this is the usual 2D squared exponential kernel.

Composition with a function

Property

Let k_1 be a kernel over $D_1 \times D_1$ and f be an arbitrary function $D \to D_1$, then

$$k(x, y) = k_1(f(x), f(y))$$

is a kernel over $D \times D$.

proof

$$\sum\sum a_i a_j k(x_i, x_j) = \sum\sum a_i a_j k_1(\underbrace{f(x_i)}_{y_i}, \underbrace{f(x_j)}_{y_j}) \geq 0$$

Remarks:

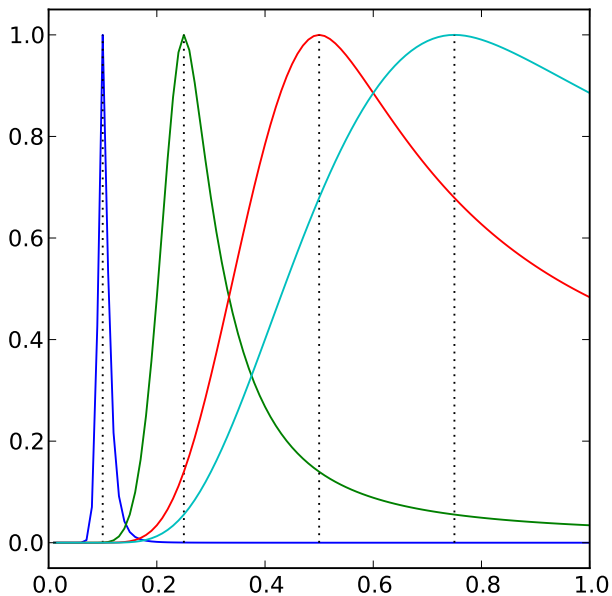
- k corresponds to the covariance of $Z(x) = Z_1(f(x))$
- This can be seen as a (non-linear) rescaling of the input space

Example

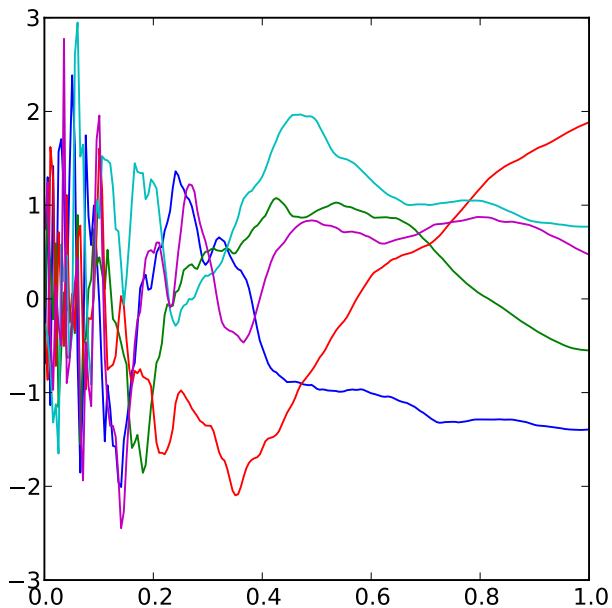
We consider $f(x) = \frac{1}{x}$ and a Matérn 3/2 kernel $k_1(x, y) = (1 + |x - y|)e^{-|x-y|}$.

We obtain:

Kernel



Sample paths



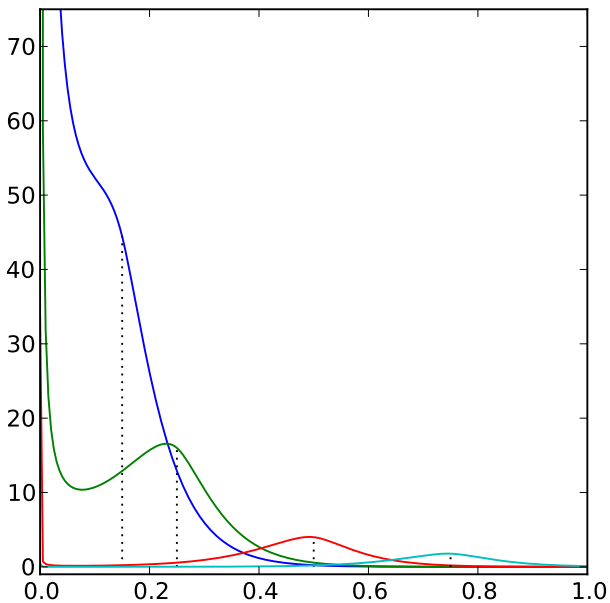
All these transformations can be combined!

Example

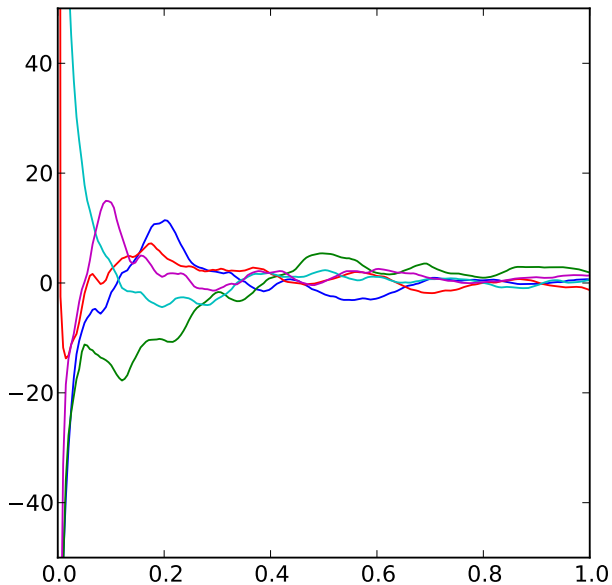
$k(x, y) = f(x)f(y)k_1(x, y)$ is a valid kernel.

This can be illustrated with $f(x) = \frac{1}{x}$ and $k_1(x, y) = (1 + |x - y|)e^{-|x-y|}$:

Kernel



Sample paths



Bochner's theorem

Theorem (Bochner)

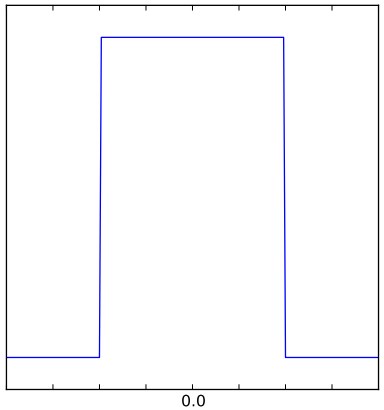
A continuous stationary function $k(x, y) = \tilde{k}(x - y)$ is positive definite if and only if $\tilde{k}$ is the Fourier transform of a finite positive measure:

$$\tilde{k}(t) = \int_{\mathbb{R}} e^{-i\omega t} \mathrm{d}\mu(\omega)$$

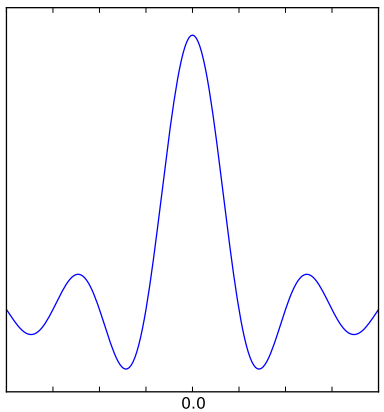
This result is very useful to prove the positive definiteness of stationary functions.

Example

We consider the following measure:



Its Fourier transform gives $\tilde{k}(t) = \dfrac{\sin(t)}{t}$:



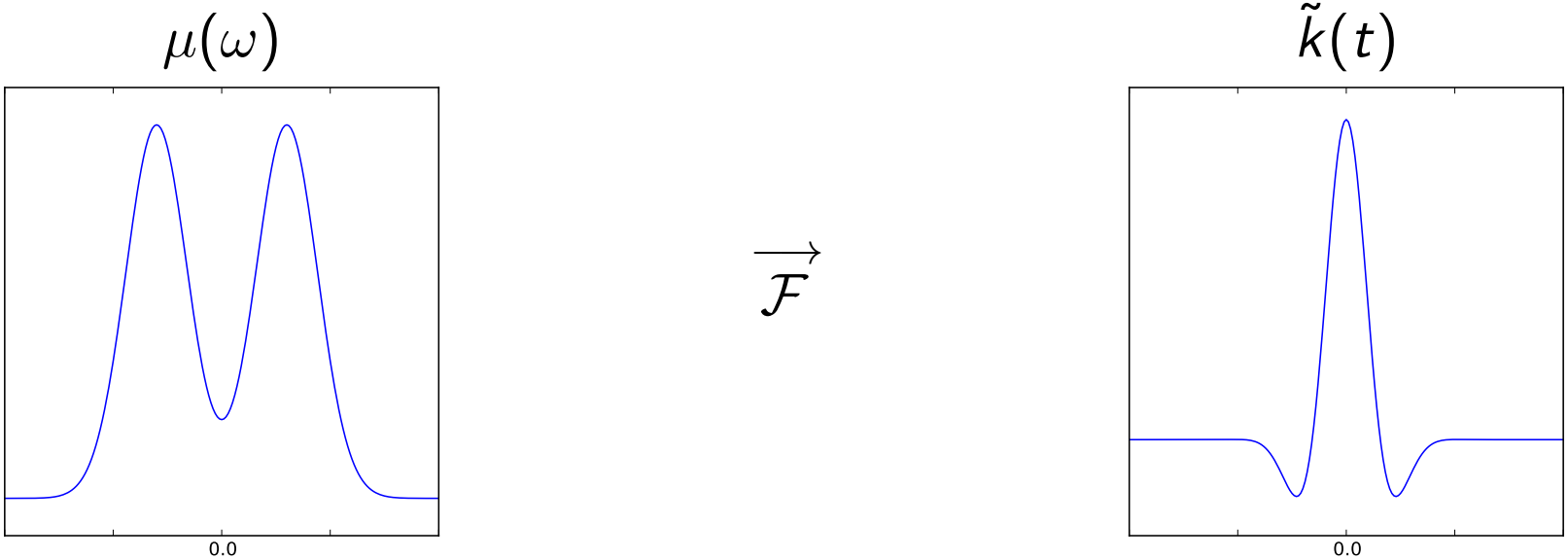
As a consequence, $k(x, y) = \dfrac{\sin(x - y)}{x - y}$ is a valid covariance function.

Bochner theorem can be used to prove the positive definiteness of many usual stationary kernels

- The Gaussian is the Fourier transform of itself
  $\Rightarrow$ it is psd.
- Matern kernels are the Fourier transforms of $\frac{1}{(1+\omega^2)^p}$
  $\Rightarrow$ they are psd.
- the constant function is the Fourier transform of $\delta_{x,y}$
  $\Rightarrow$ it is psd.

Spectral approximation with a mixture of Gaussian (A. Wilson, ICML 2013)

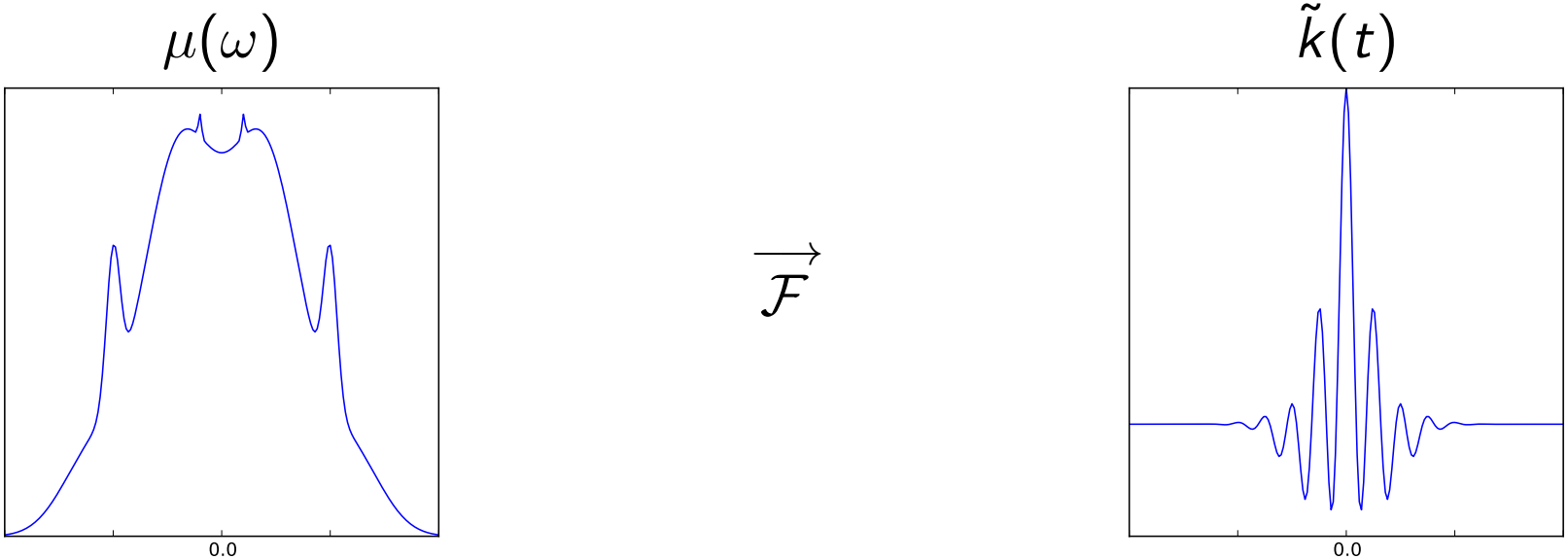
The inverse Fourier transform of a (symmetrised) non centred Gaussian is:



This can be generalised to a measure based on the sum of Gaussians.

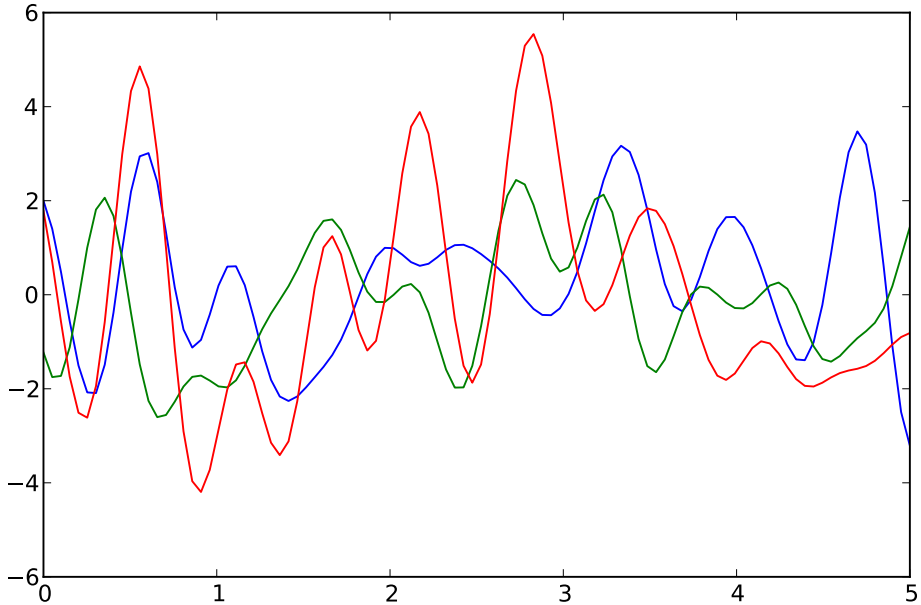
Spectral approximation with a mixture of Gaussian (A. Wilson, ICML 2013)

We obtain a kernel that is parametrised by the means and the bandwidths of Gaussian bells in the measure space:



Spectral approximation with a mixture of Gaussian (A. Wilson, ICML 2013)

The sample paths have the following aspect:



Effect of a linear operator

Effect of a linear operator

Property

Let L be a linear operator that commutes with the covariance, then $k(x, y) = L_x(L_y(k_1(x, y)))$ is a kernel.

Example

We want to approximate a function $[0, 1] \to \mathbb{R}$ that is symmetric with respect to 0.5. We will consider 2 linear operators:

$$L_1 : f(x) \to \begin{cases} f(x) & x < 0.5 \\ f(1-x) & x \geq 0.5 \end{cases}$$

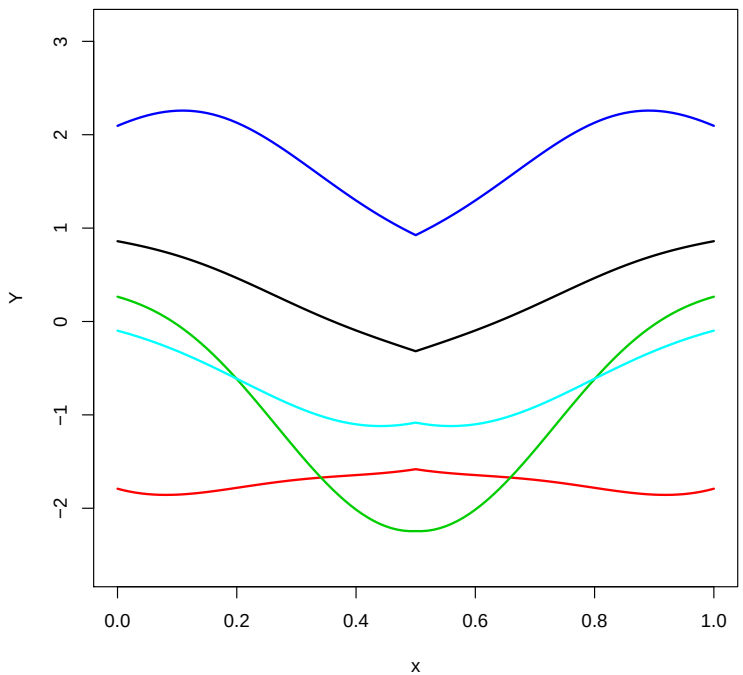
$$L_2 : f(x) \to \frac{f(x) + f(1-x)}{2}.$$

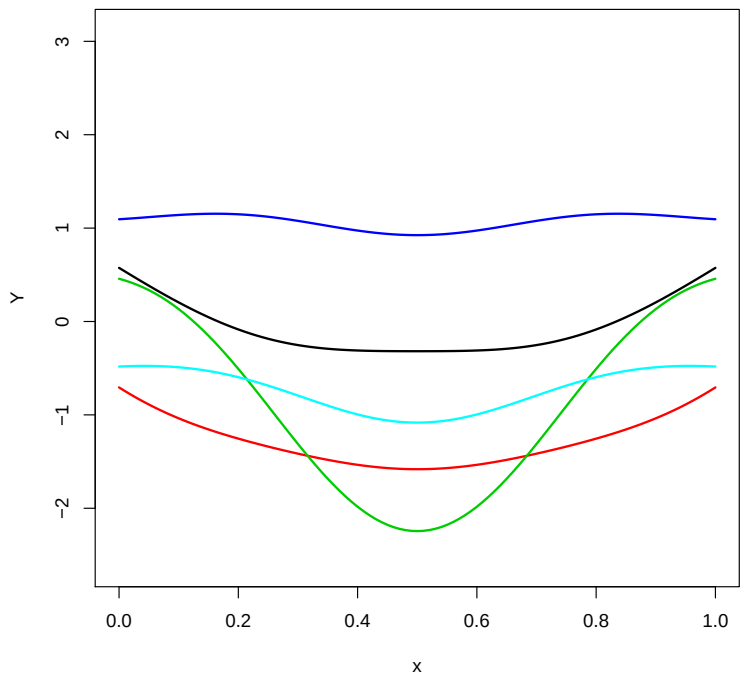
Exercice: Compute the kernel associated to the second operator.

Effect of a linear operator: example [Ginsbourger 2013]

Examples of associated sample paths are

$$k_1 = L_1(L_1(k)) \qquad k_2 = L_2(L_2(k))$$

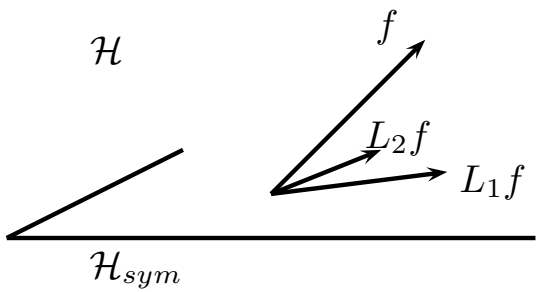




The differentiability is not always respected!

Effect of a linear operator

These linear operator are projections onto a space of symmetric functions:



Is there an optimal projection?

$\Rightarrow$ This can be difficult... but it raises interesting questions!

Effect of a linear operator

Can we construct a GP such that the integrals of the paths are null?

We can think of the following application:

$$L : f(x) \to f(x) - \int f(s)\,\mathrm{d}s.$$

More generally, for all $g : [0,1] \to \mathbb{R}$, the application

$$L_g : f(x) \to f(x) - \frac{g(x)}{\int g(s)\,\mathrm{d}s} \int f(s)\,\mathrm{d}s$$

will center f. It turns out that the optimal g is $g(x) = \int k(x,s)\,\mathrm{d}s$

Exercice

1. Compute the associated kernel.
2. What is the distribution of $Z | \int Z = 0$?

Adding "exotic" observations

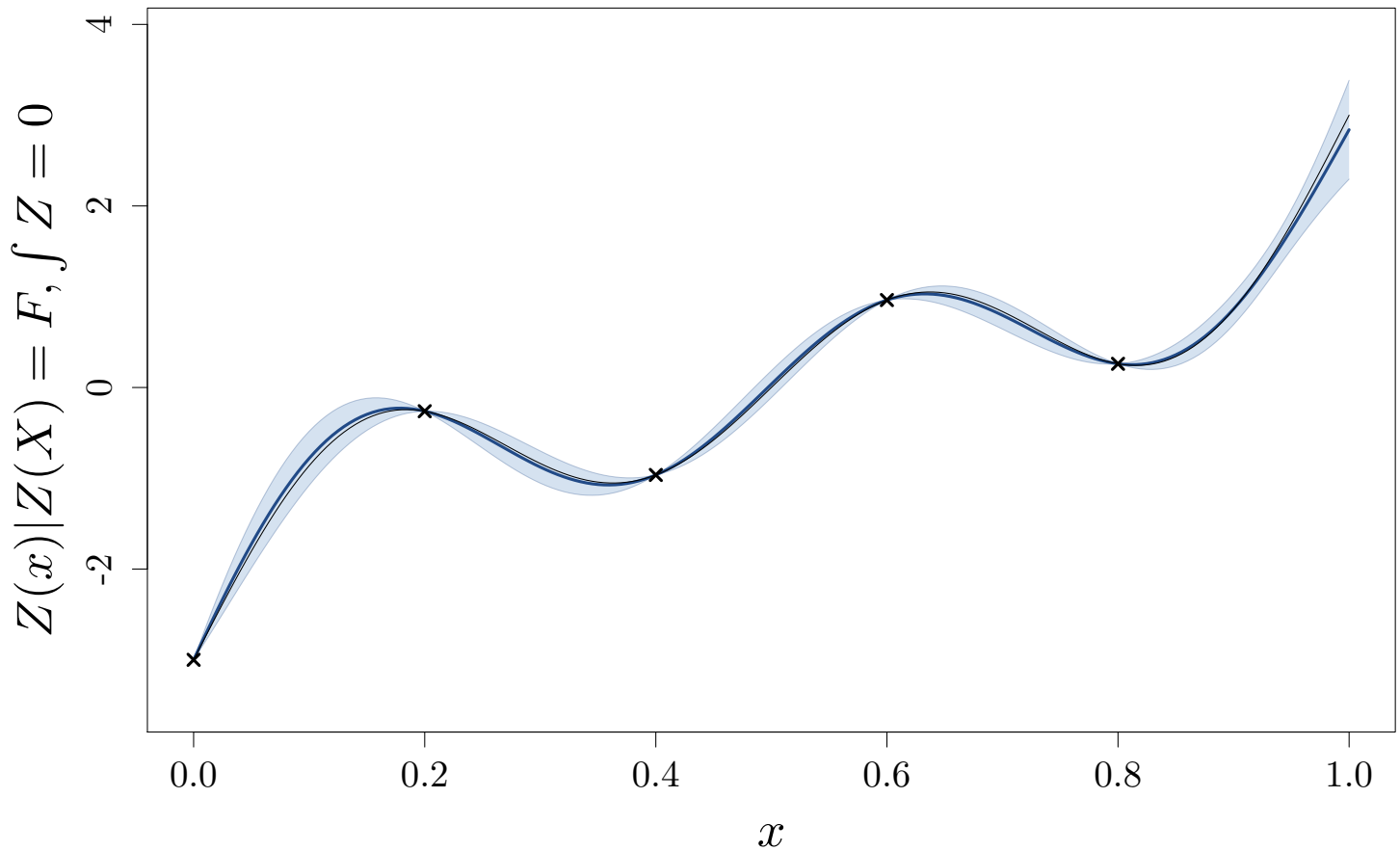
Up to now, we have only considered regular observation of the kind $f(X)$.

According to what we have just seen, the conditioning can also include observation of the integral. This can be generalised to other linear operators such as the derivative:

$$Z \;\Big|\; Z(X) = F, \int Z = a, \frac{\mathrm{d}Z}{\mathrm{d}x}(X') = F'$$

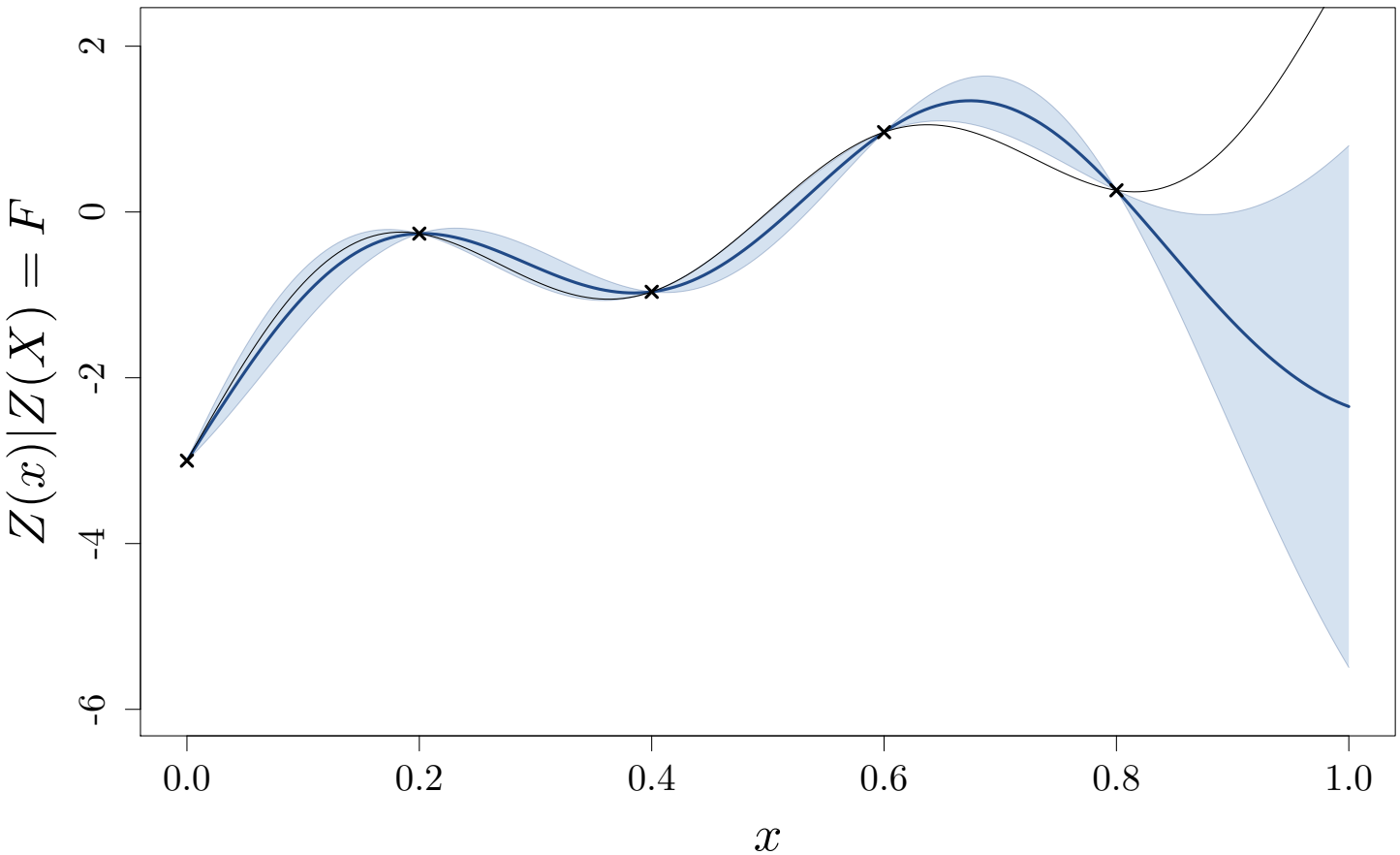
Example

If we take into account that the function is centred, we obtain:

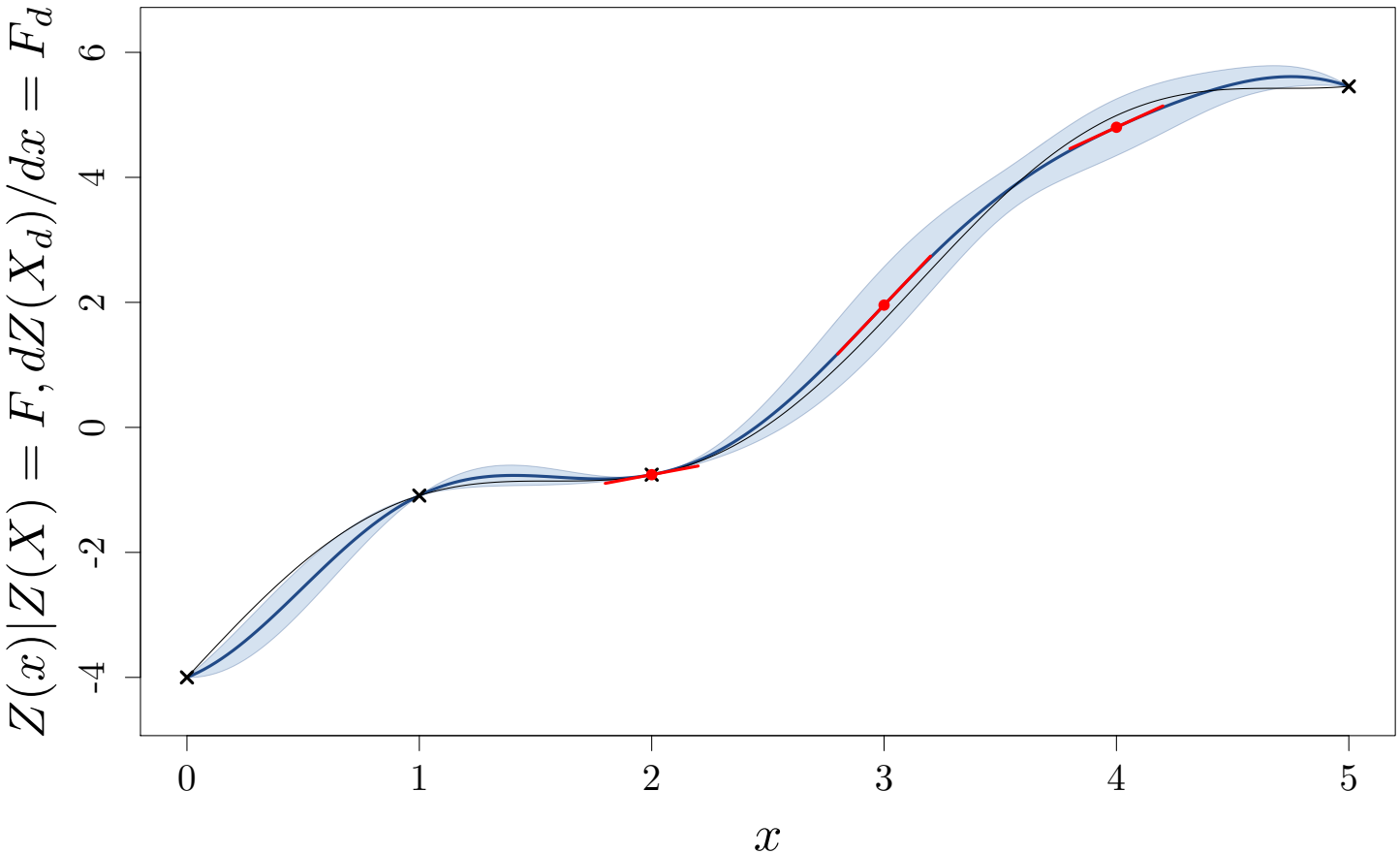


Example

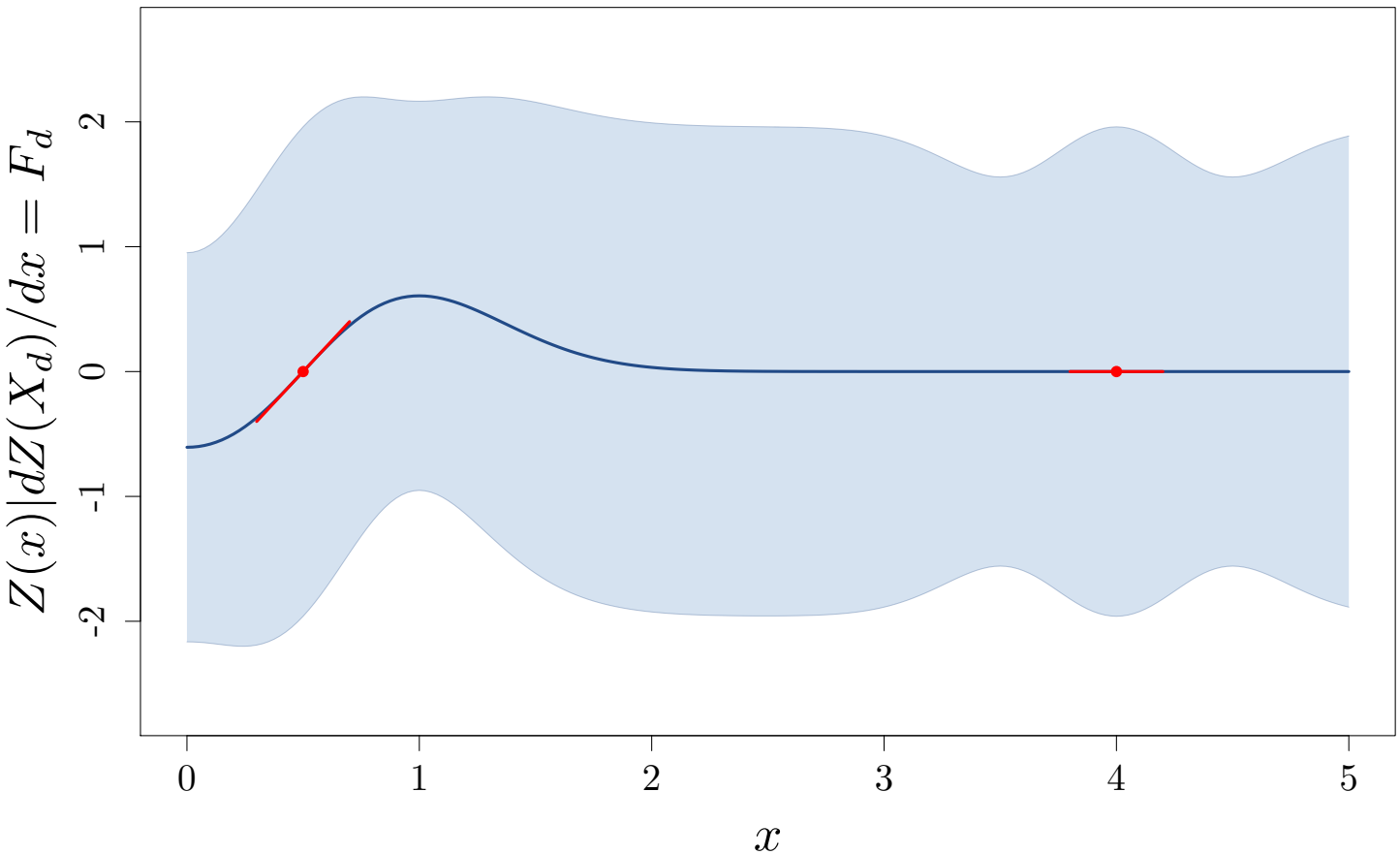
Whereas if we ignore it:



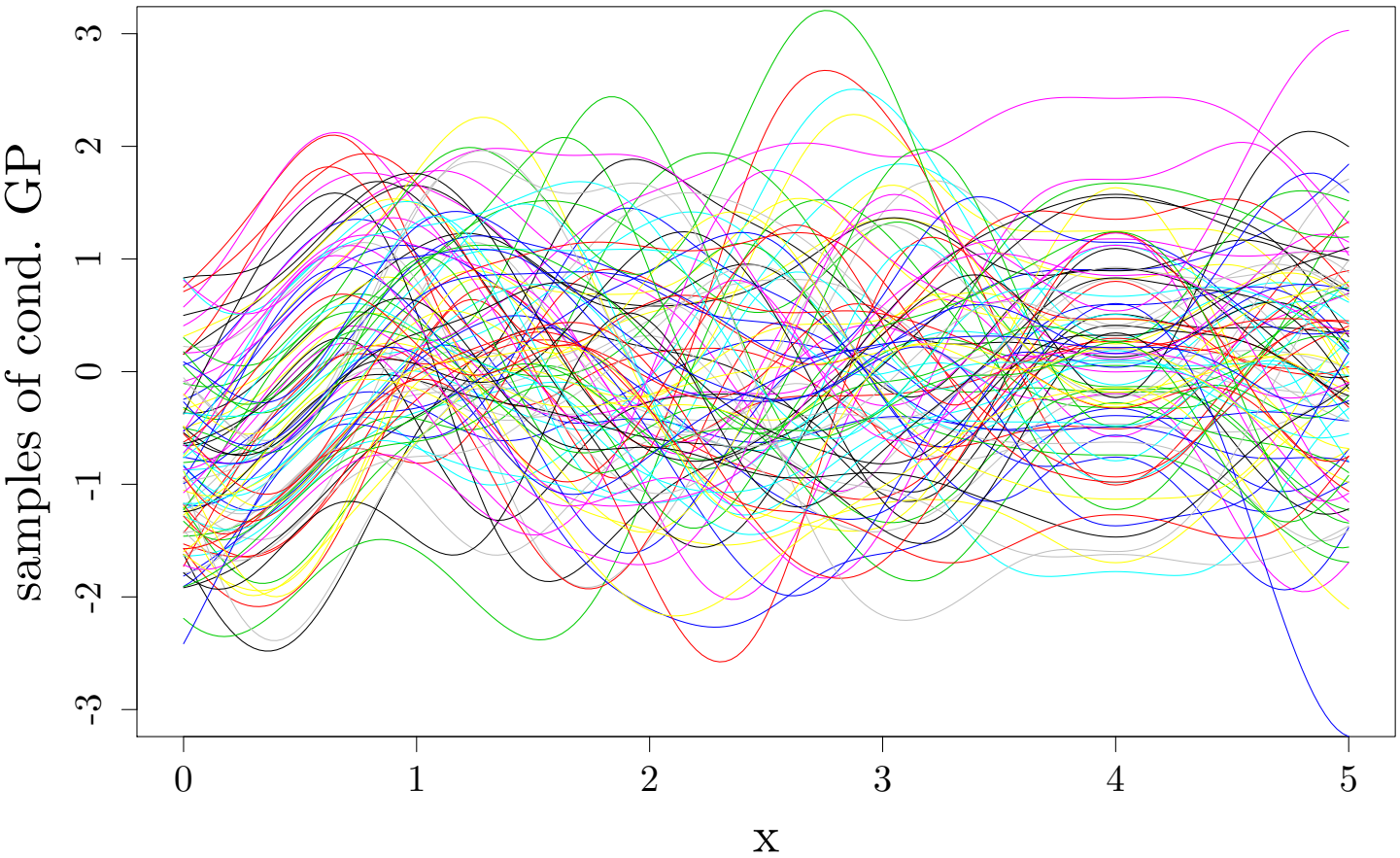
Similarly, we can include in the model some derivatives observations:



We can see interesting behaviour if we look at a model with only derivatives.

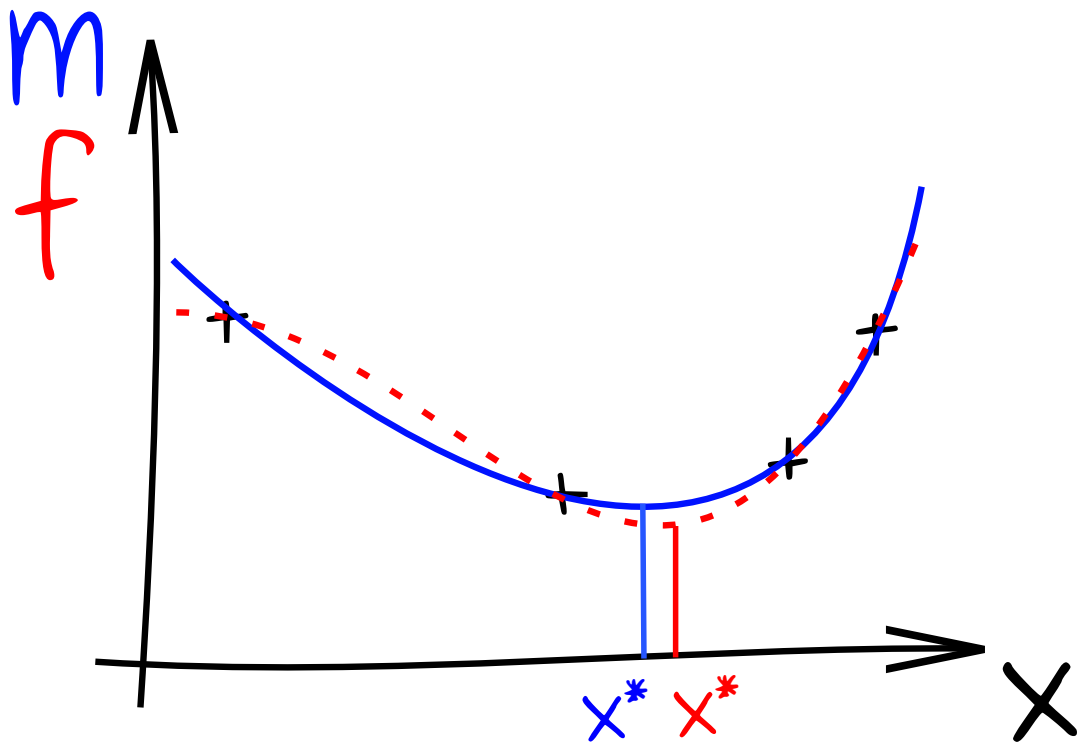


As always, we can simulate conditional paths:



Model based optimization methods

If the number of function evaluations are limited, we can run the optimization on the model instead of running it directly on the function

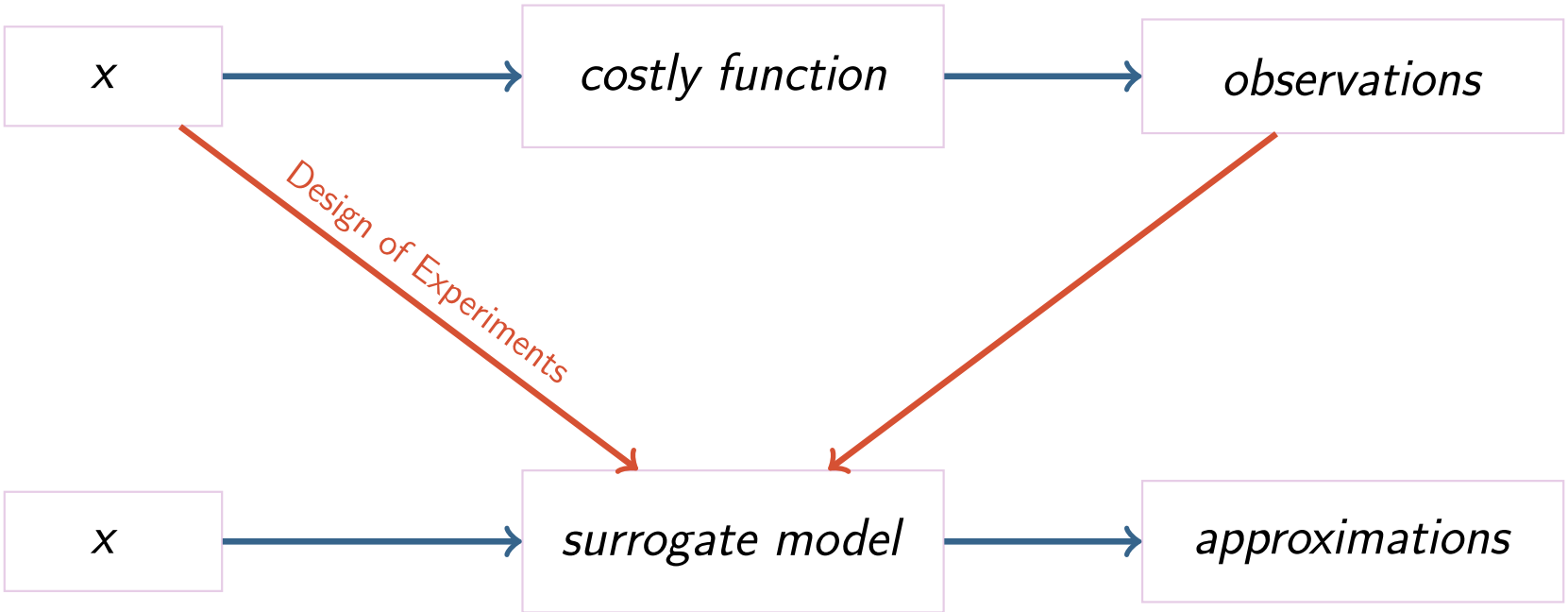


In the end, we hope that:

$$\text{argmin}(m) \approx \text{argmin}(f)$$

$$\min(m) \approx \min(f)$$

Overall framework



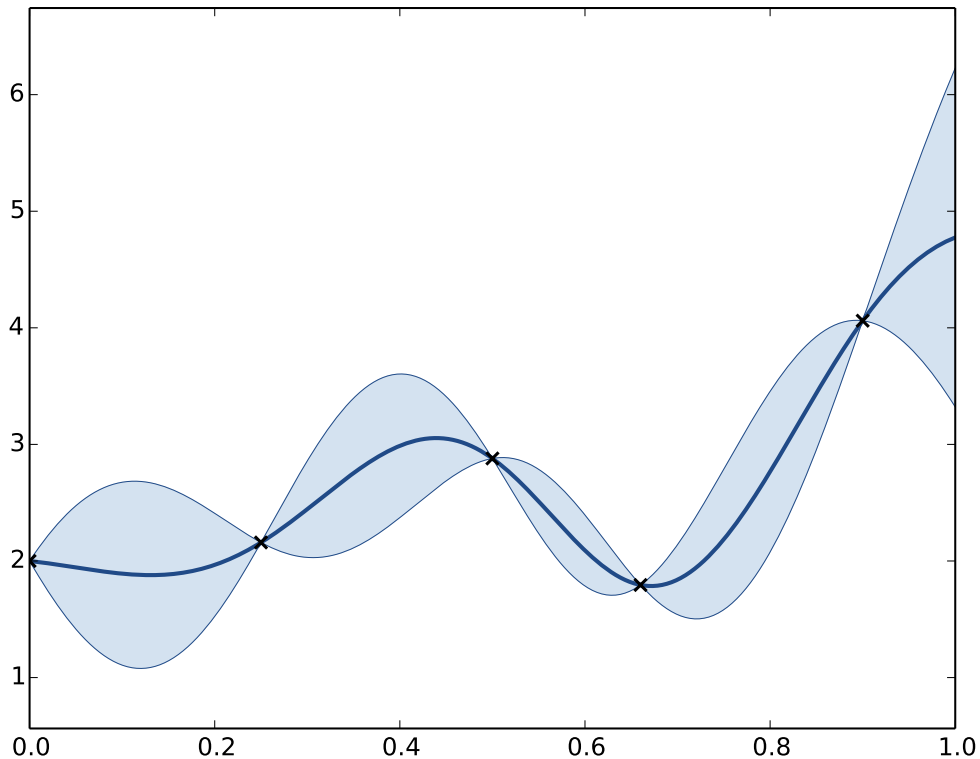
In practice, it is risky to take decisions based only on the model...

On the other hand, the model can be used to guide us in the search for the optimum.

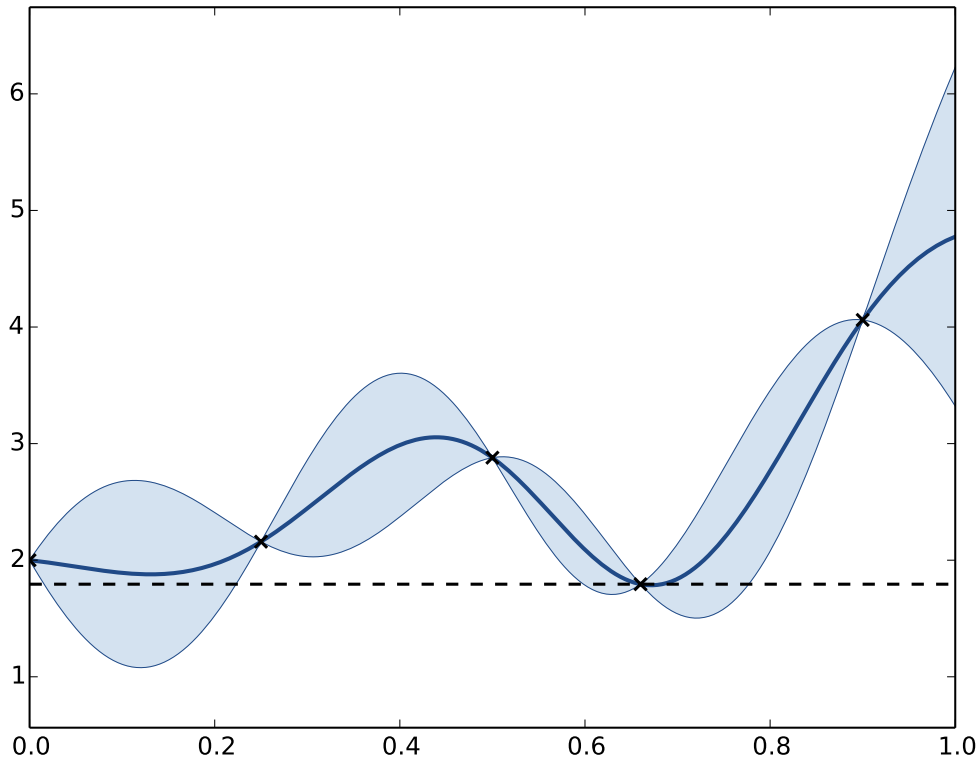
Global optimization methods are a trade-off between

- Exploitations of good results
- Exploration of the space

How can GPR models be helpful?



In our example, the best observed value is 1.79

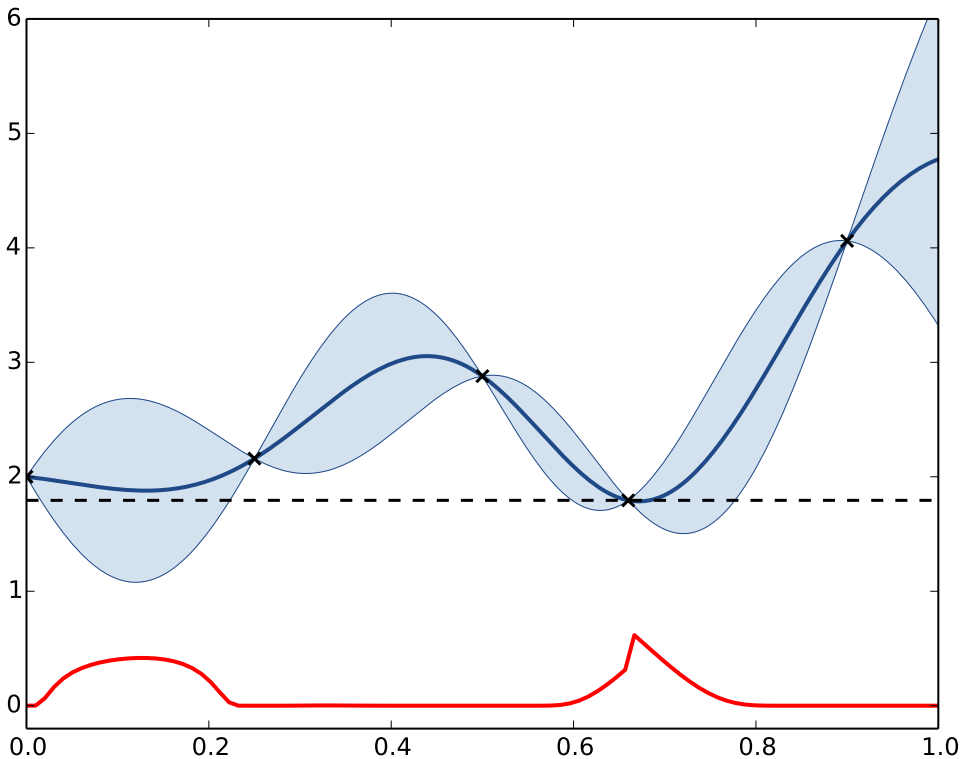


Various criteria can be studied

- probability of improvement
- Expected improvement

Probability of Improvement:

$$PI(x) = cdf\left(\frac{\min(F) - m(x)}{\sqrt{(c(x,x))}}\right)$$



The point with the highest PI is often very close to the best observed value. We can show that there is a x in the neighbourhood of x^* such that $PI(x) \geq 0.5$.

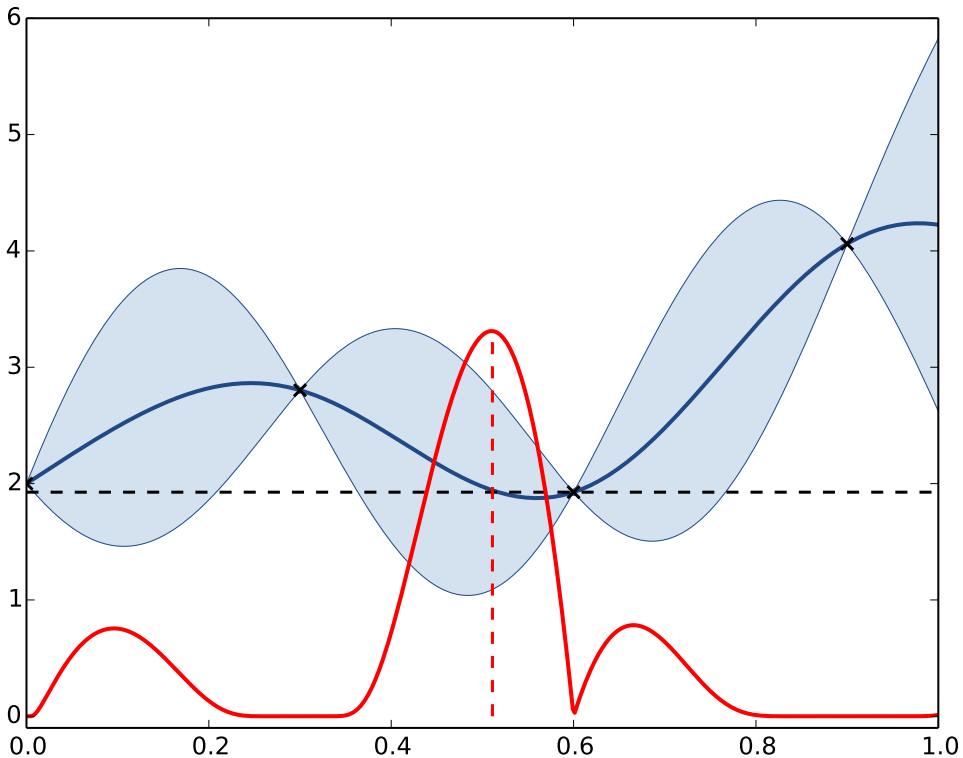
For such points, the improvement cannot be large...

Can we find another criterion?

Expected Improvement:

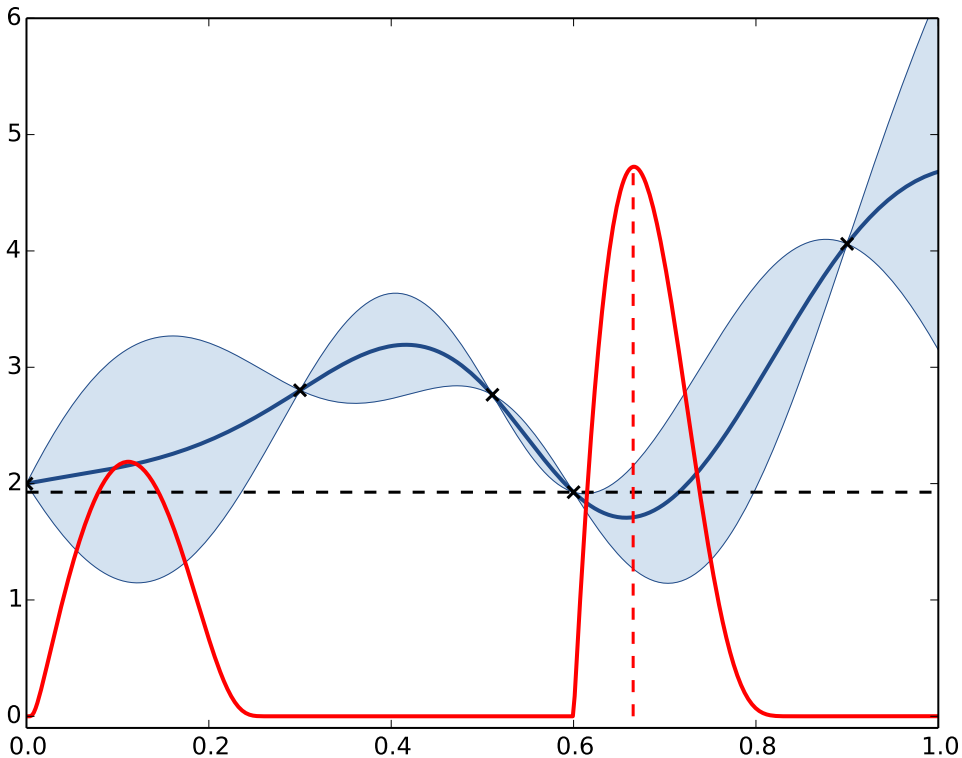
$$EI(x) = \sqrt{c(x,x)}(u(x)cdf(u(x)) + pdf(u(x)))$$

$$\text{with } u(x) = \frac{\min(F) - m(x)}{\sqrt{(c(x,x))}}$$



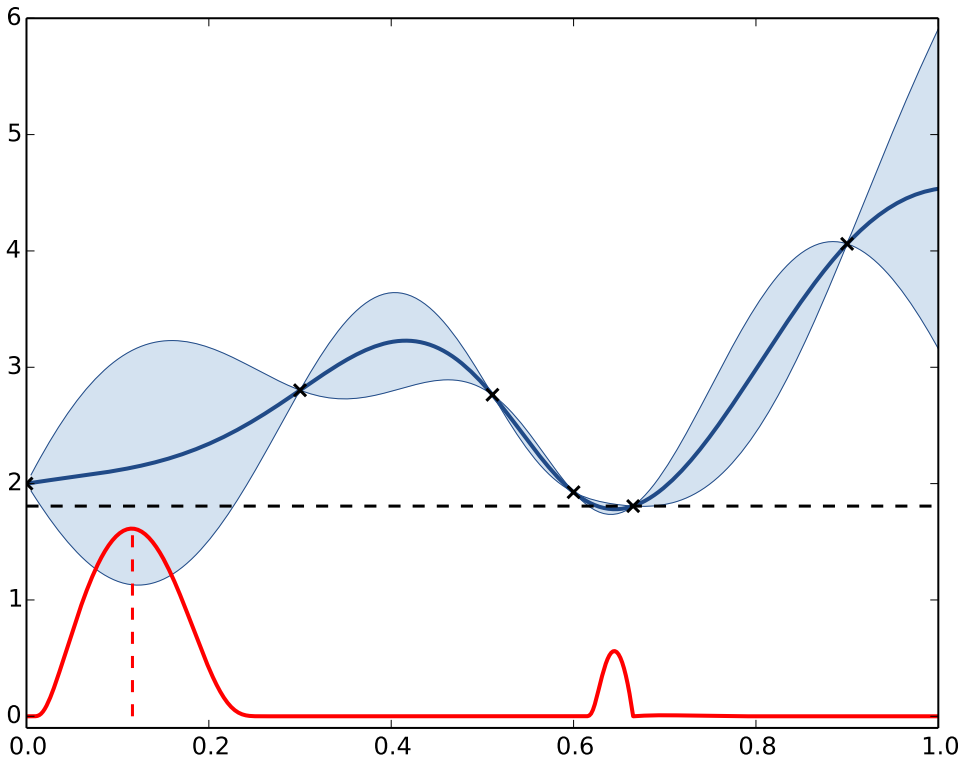
Expected Improvement

Let's see how it works... iteration 1



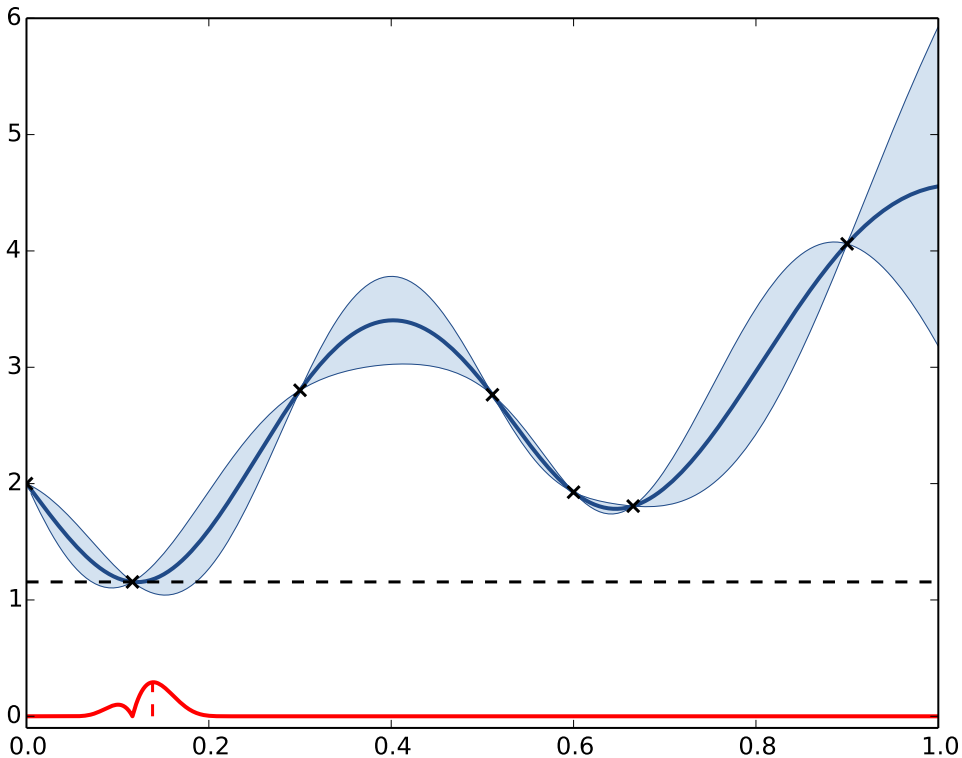
Expected Improvement

Let's see how it works... iteration 2



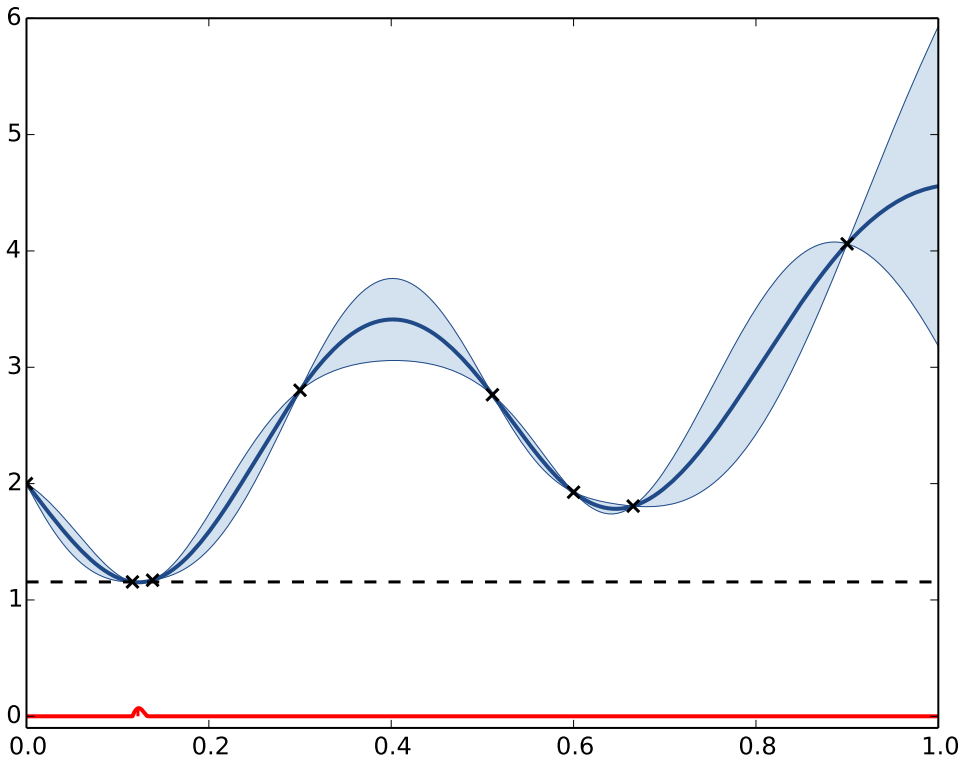
Expected Improvement

Let's see how it works... iteration 3



Expected Improvement

Let's see how it works... iteration 4



Expected Improvement

Let's see how it works... iteration 5

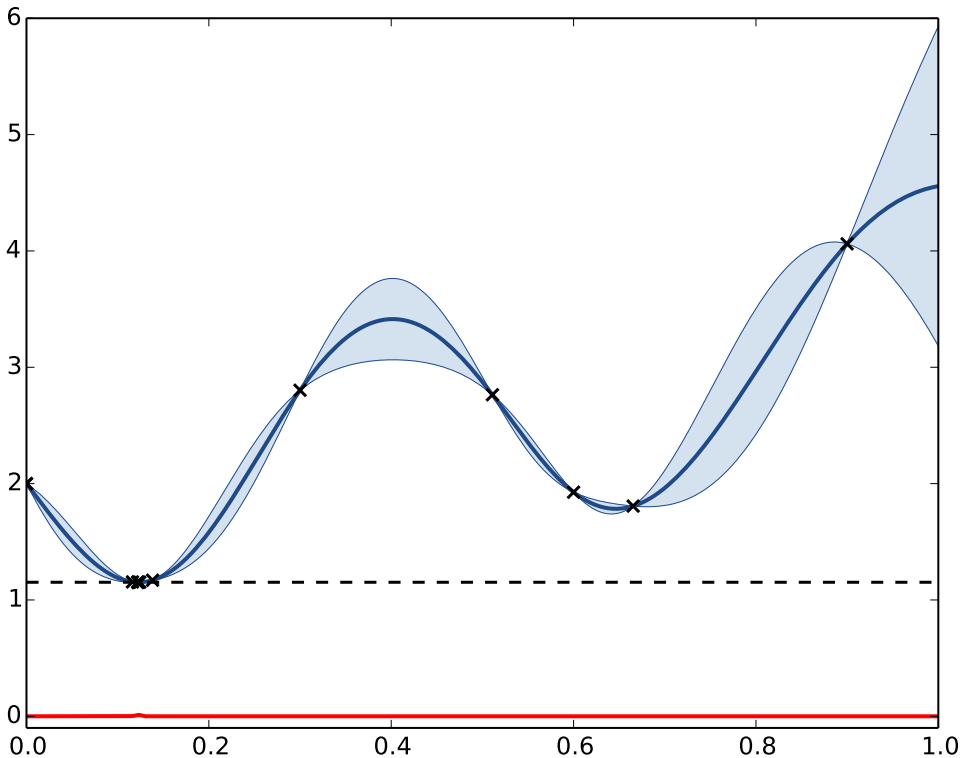
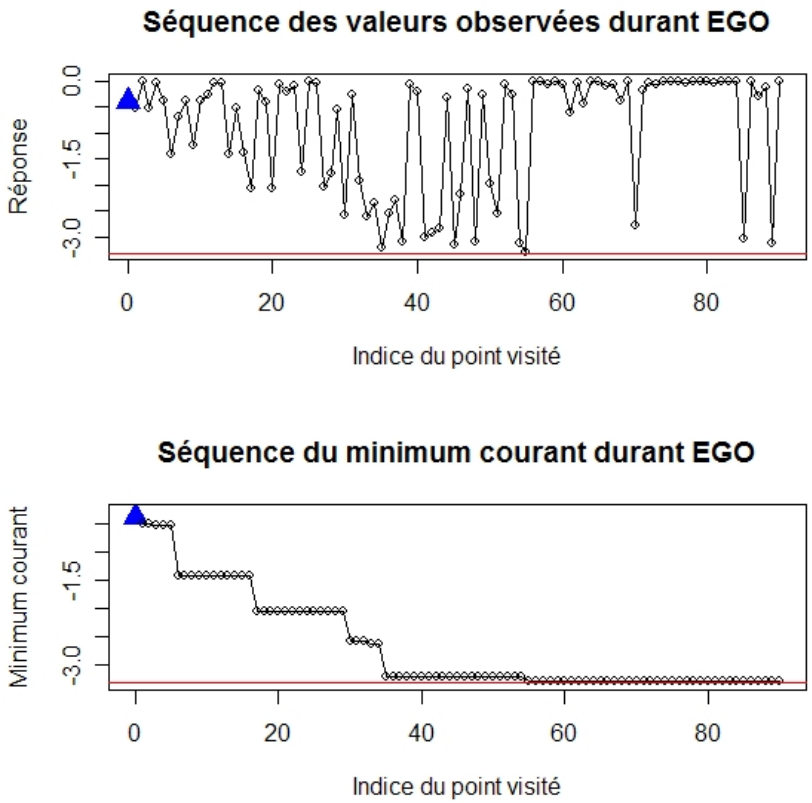
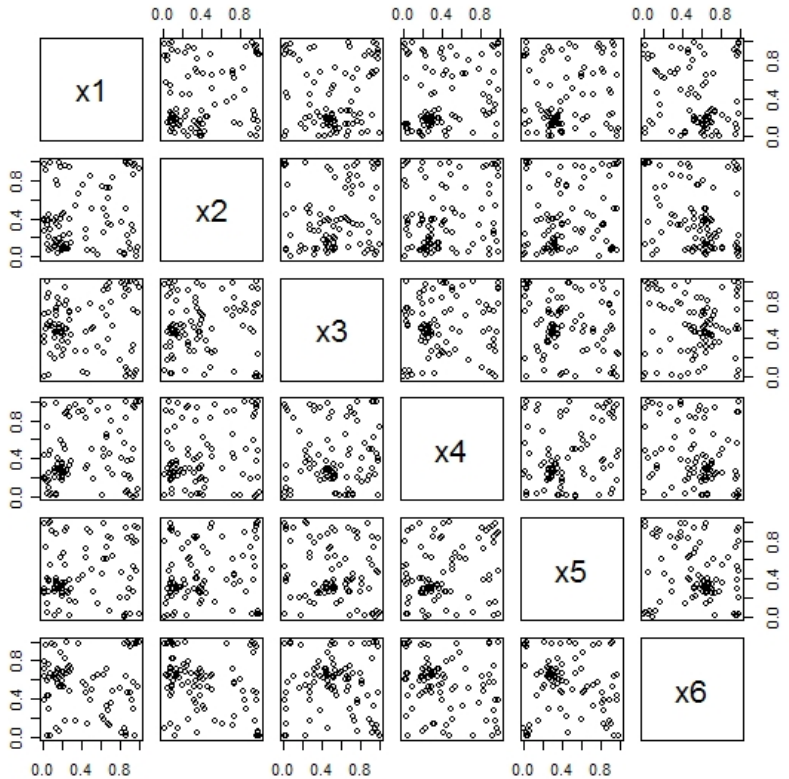


Illustration for $d = 6$ (Hartman)

Illustration in higher dimension





Source: DiceOptim, D. Ginsbourger, 2009.

Expected Improvement

This algorithm is called **Efficient Global Optimization** (EGO). It is famous since a paper of Jones et Al in 1998.

+ EGO provides a good trade-off between exploitation and exploration.
+ It only requires a few function observations (10 in the example)

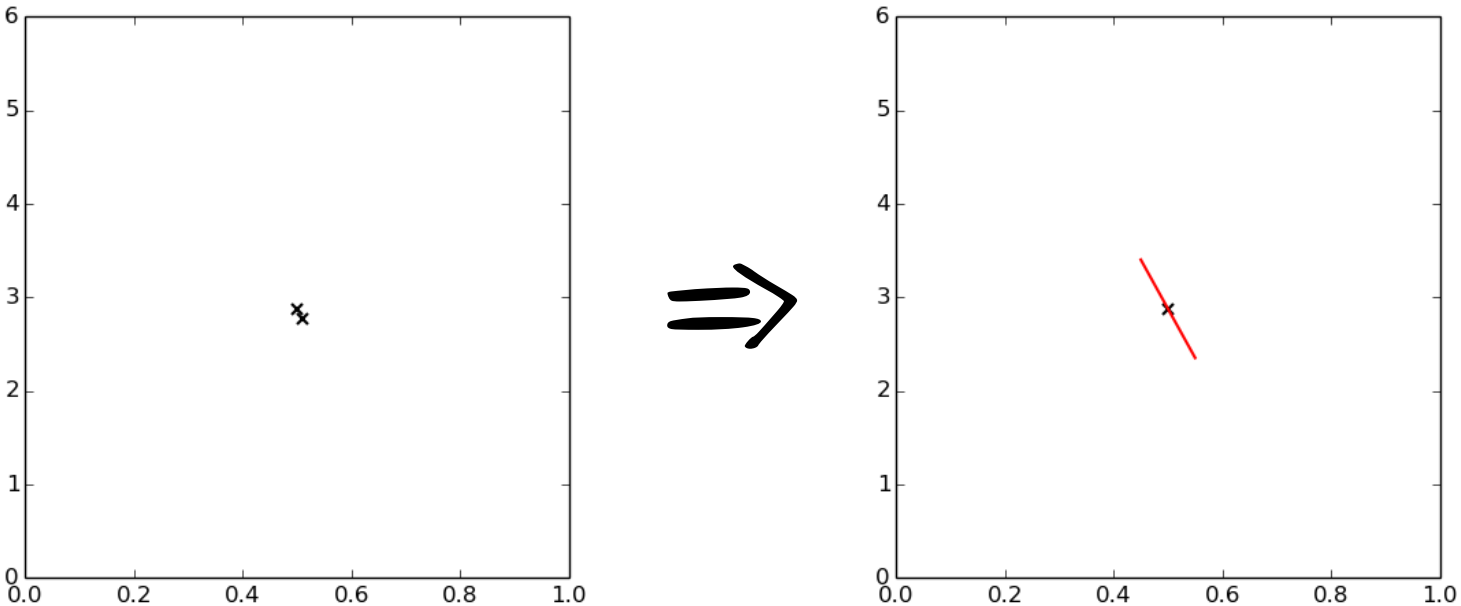
One issue is that we may have a model with observations very close one from each other

Example

From the previous 5 iterations, we obtain $1.44e39$ for the conditioning of the covariance matrix. Eigenvalues are

$$(67.70,\ 24.86,\ 5.13,\ 1.68,\ 0.45,\ 0.16,\ 0.01,\ 0.00,\ 0.00,\ 0.00)$$

One way to improve the conditioning of the covariance matrix is to replace two values that are close-by by one function value and one derivative:



Cond. $= 3842$ Cond. $= 10$

This can be generalised to higher orders $\rightarrow$ Taylor expansion
see articles from M. Osborn

If we know the computational budget in advance, adding new points at the **best one step ahead location** is not optimal.

Some improvements have been made toward this

- Batch EGO
- Parallelization of the algorithm

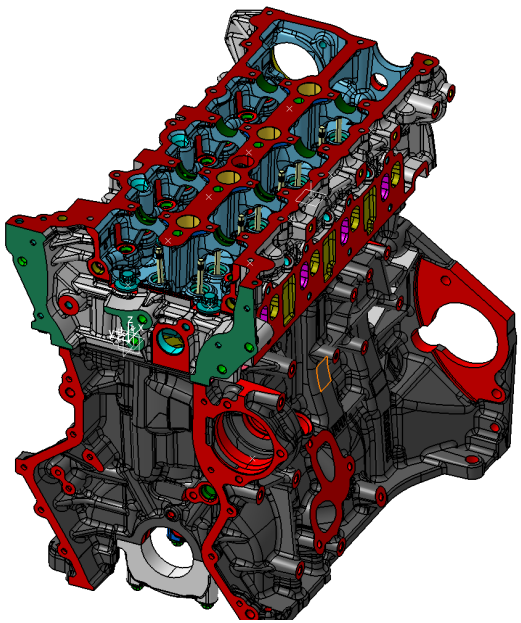
see works from D. Ginsbourger

Robust optimization

Robust optimization may mean various things:

- There is observation noise on the output
- Some input variables are uncertain
- Model is uncertain

Example

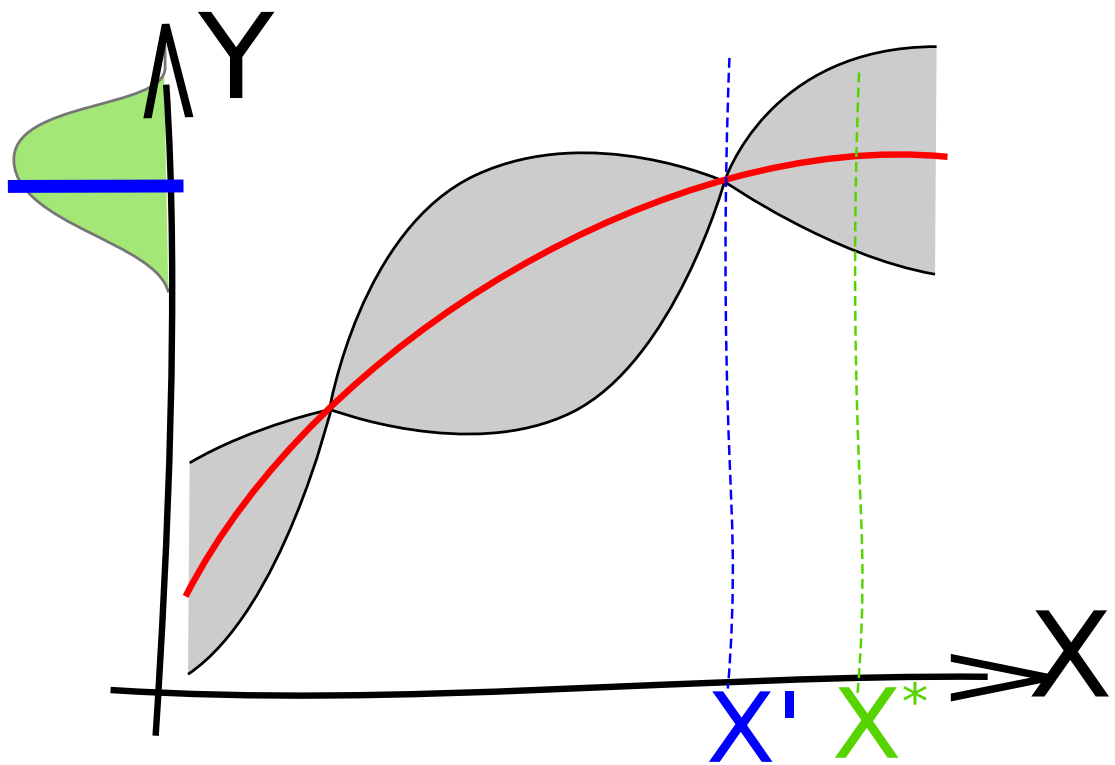


a +/- 1mm dispersion in the manufacturing of a car cylinder head can degrade its performance (g CO2/km) by -20% (worst case)

Source: Talk from R. Le Riche at the Porquerolles Summer School, 2014

Example

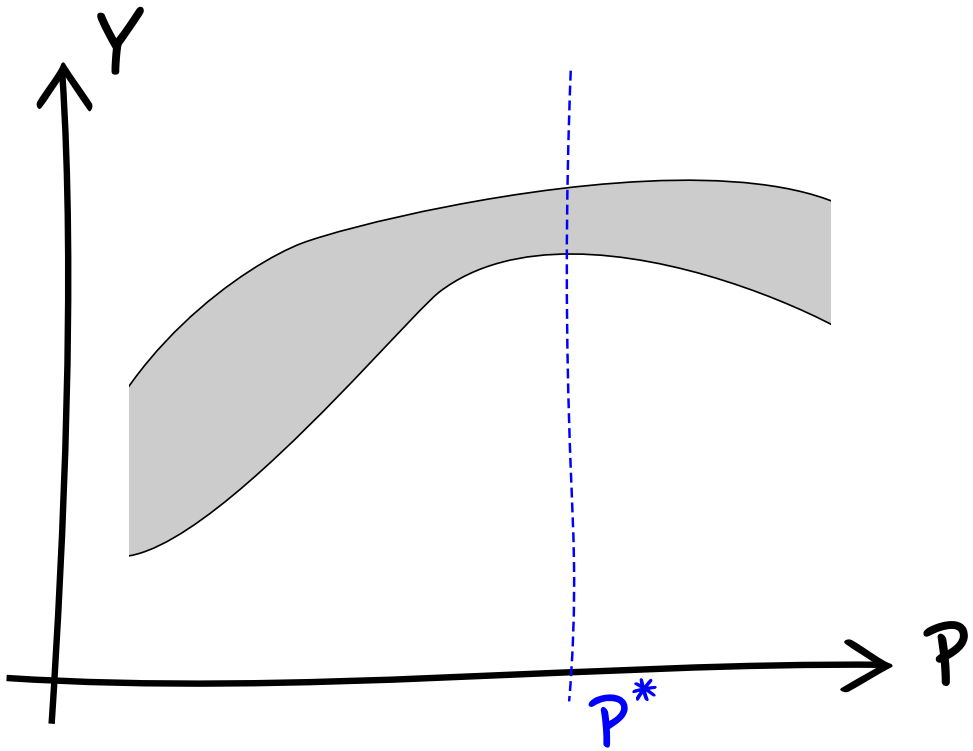
Here is a basic example:



Which input is the best?

Example

A non Gaussian example



In some cases, we may want to optimize the worst case scenario.

Can EGO be adapted when observations are noisy?

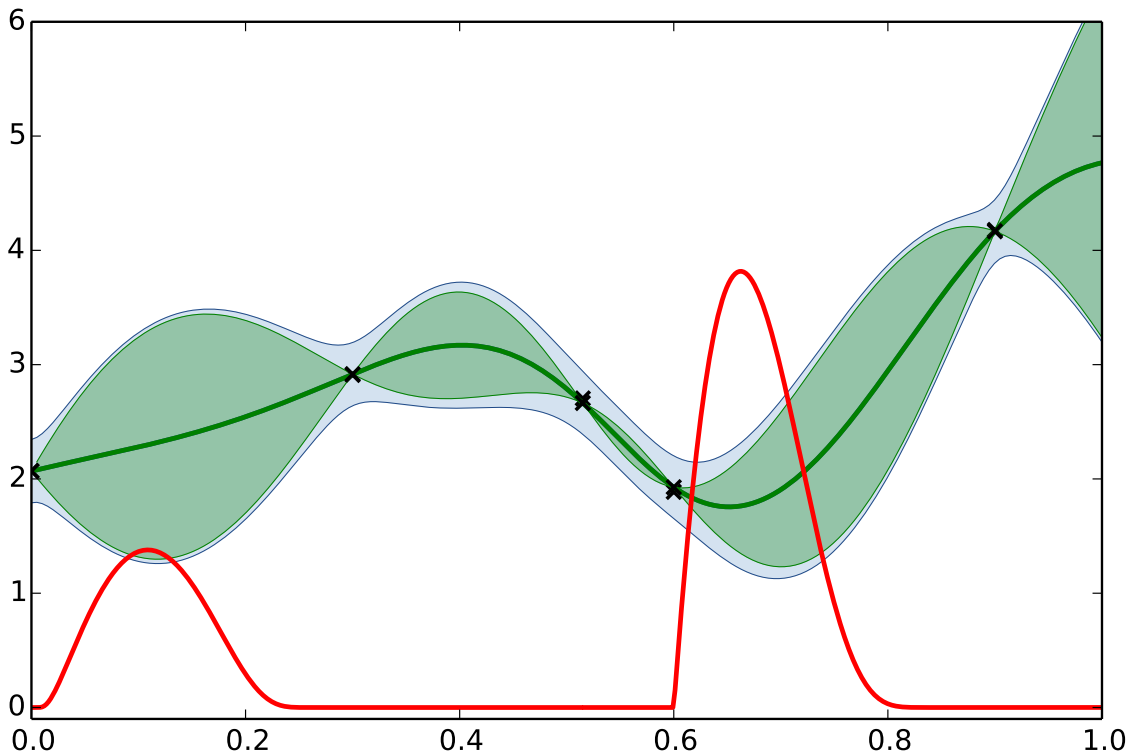
First of all, using the current best observation as a minimum does not make much sense...

Some solutions are

- S1 Build a new model that interpolates $m(X)$ at X.
- S2 Include observation noise and replace $\min(F)$ by $\min(m(X))$ in the EI expression
- S3 Similar to 2 but consider an Expected Mean Improvement.

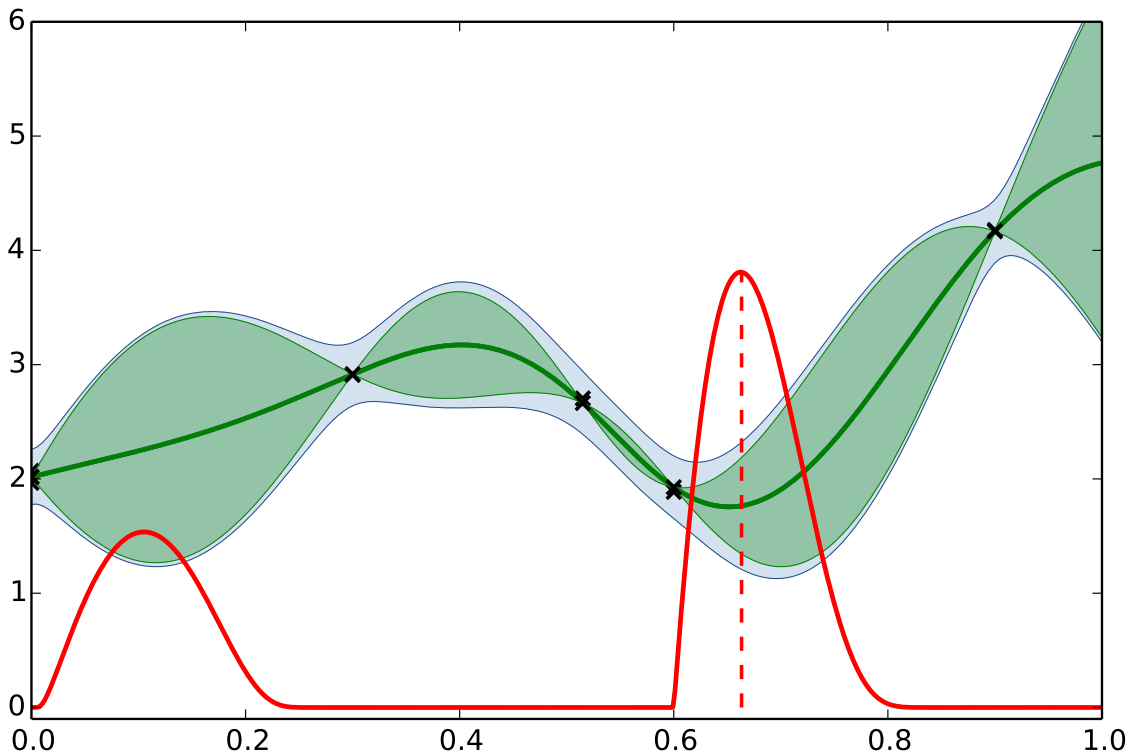
Solution 1

iteration 0



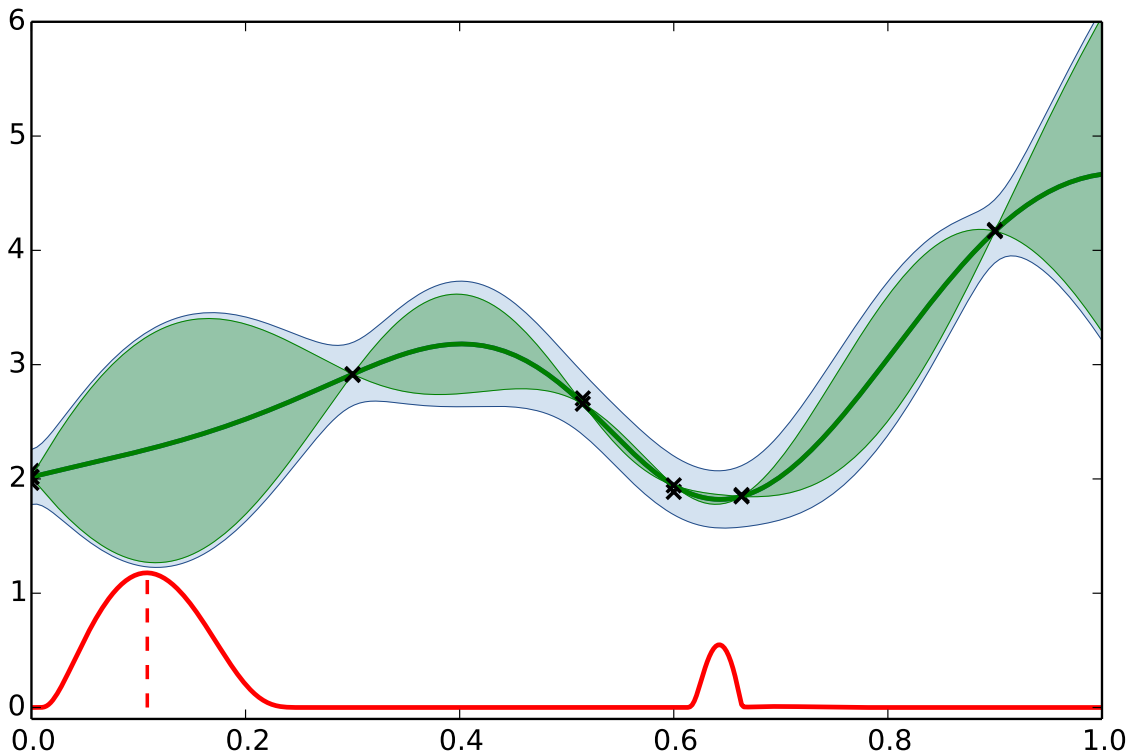
Solution 1

iteration 1



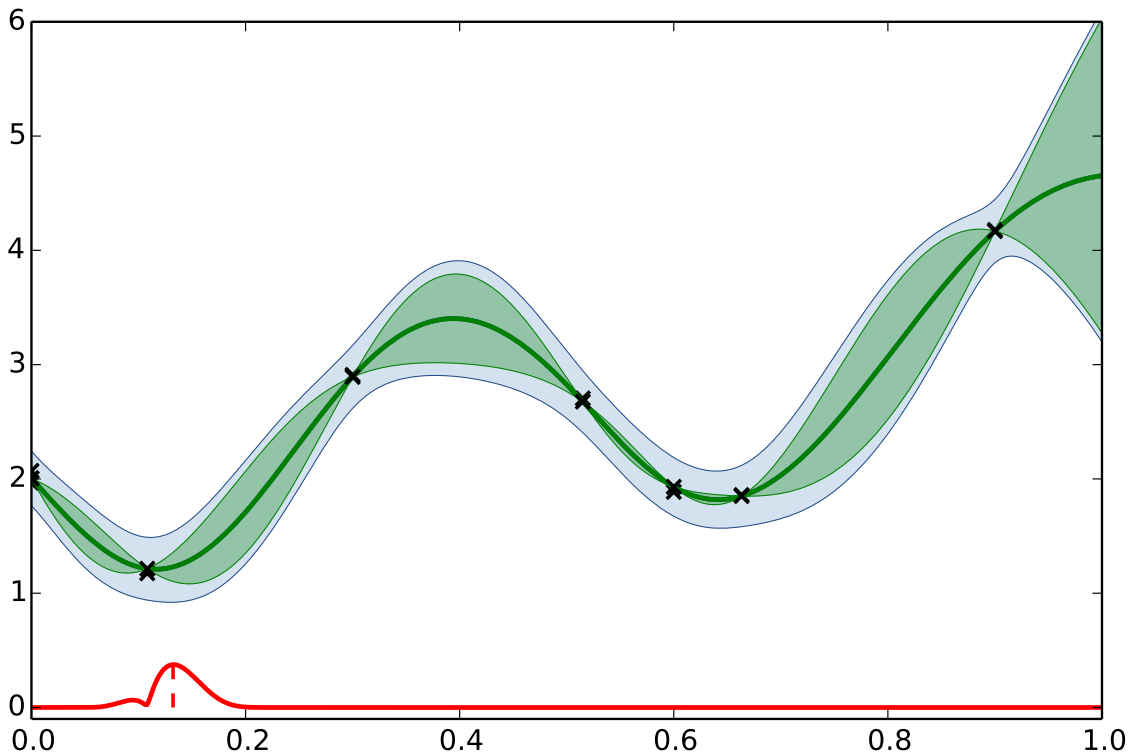
Solution 1

iteration 2



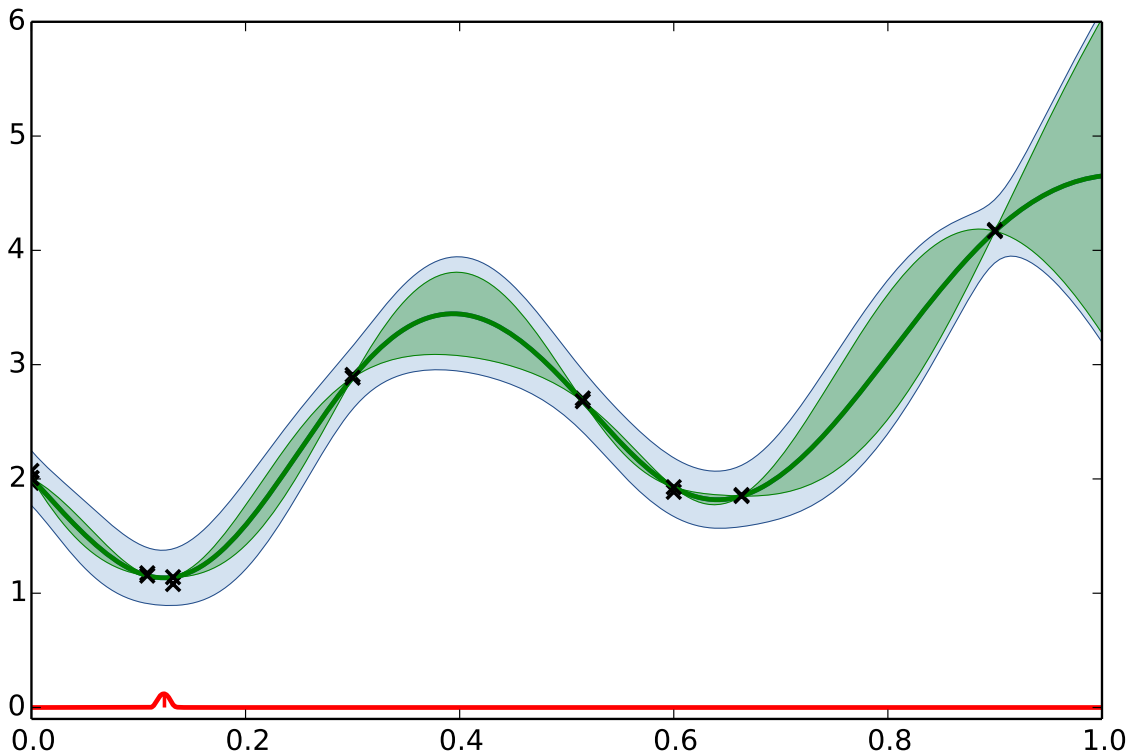
Solution 1

iteration 3



Solution 1

iteration 4



Related problems

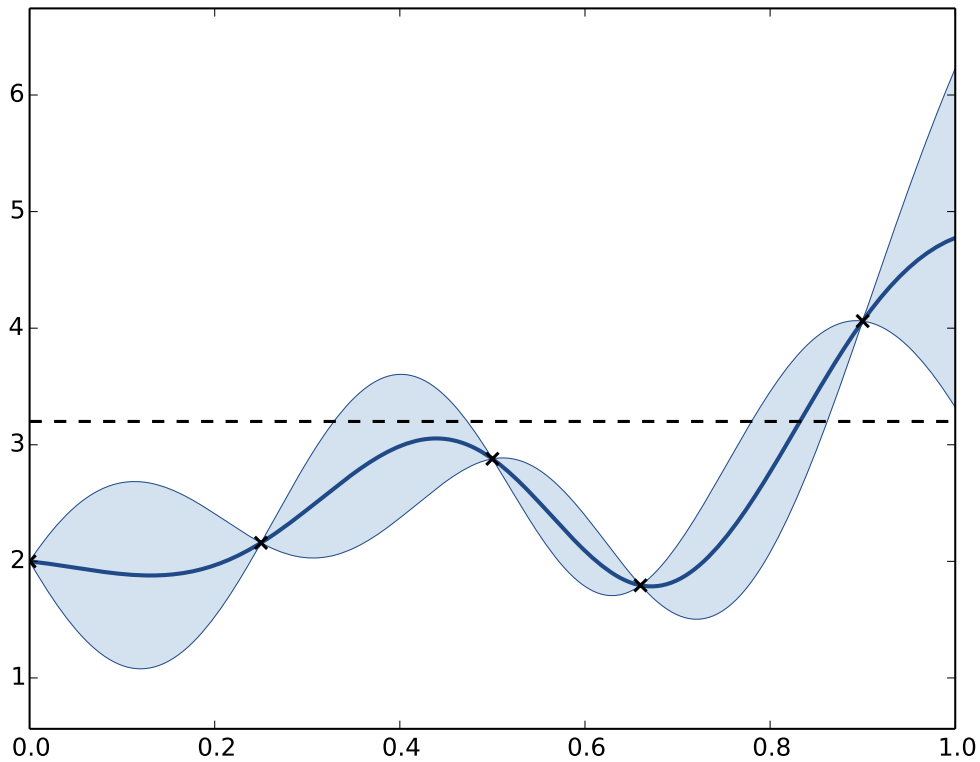
Some related optimization problems are:

- calibration problems
- probability computations

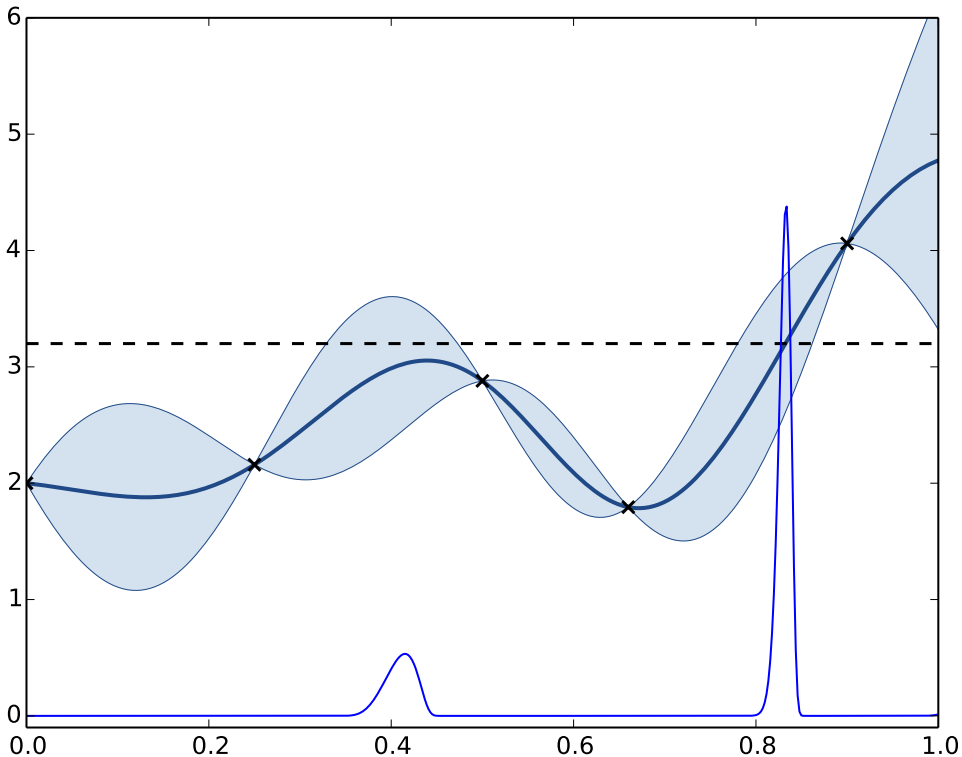
Some algorithms with an EGO spirit can be applied:

- SUR methods

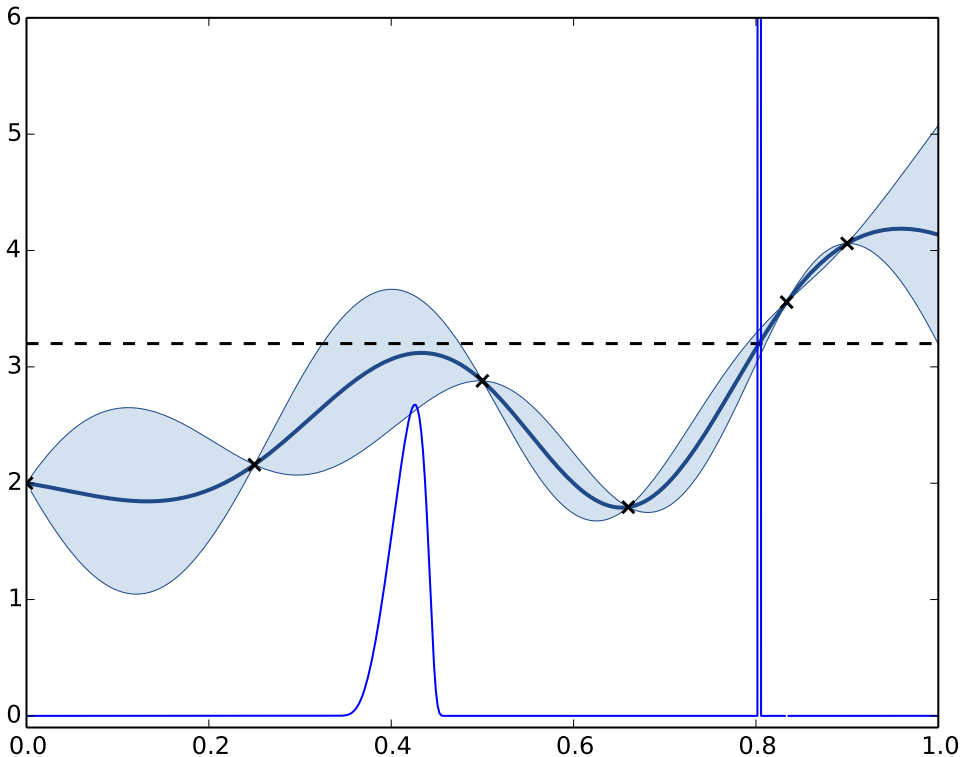
We want to find the input(s) such that f(x) = 3.2



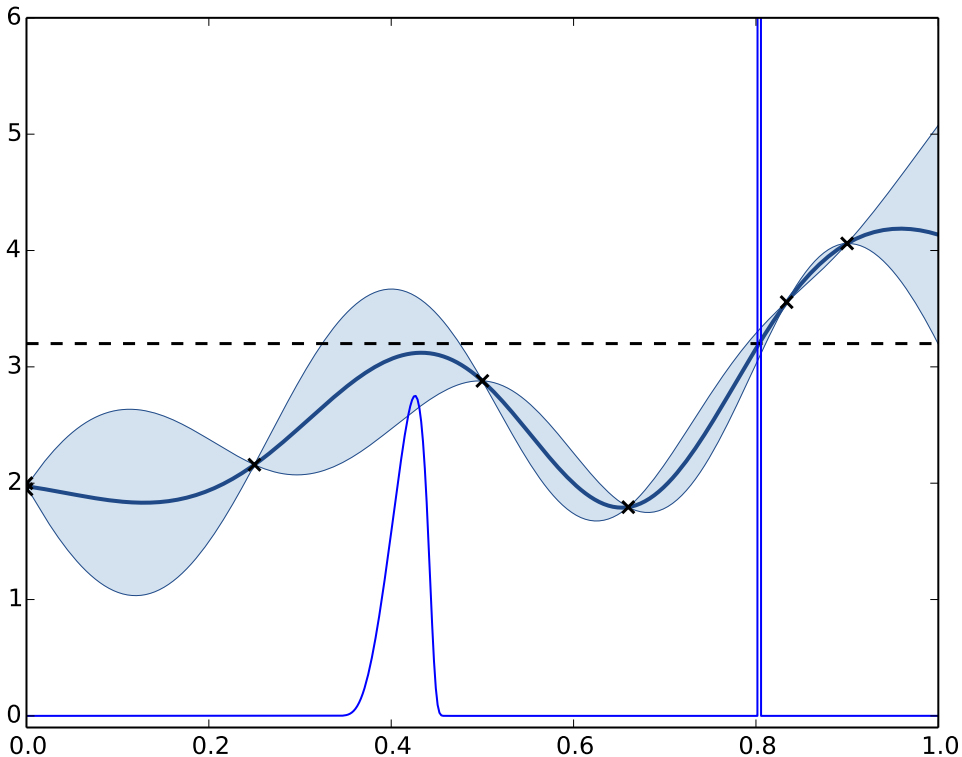
iteration 0:



iteration 1:



iteration 2:



iteration 3: