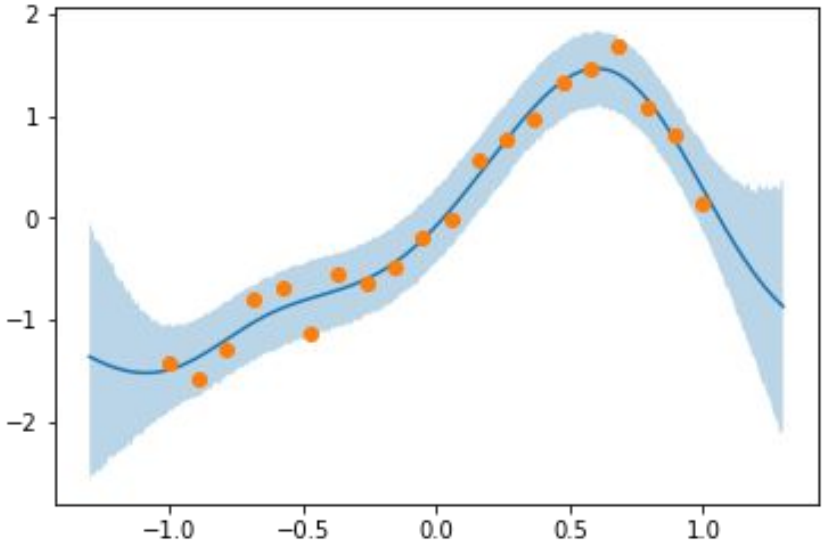
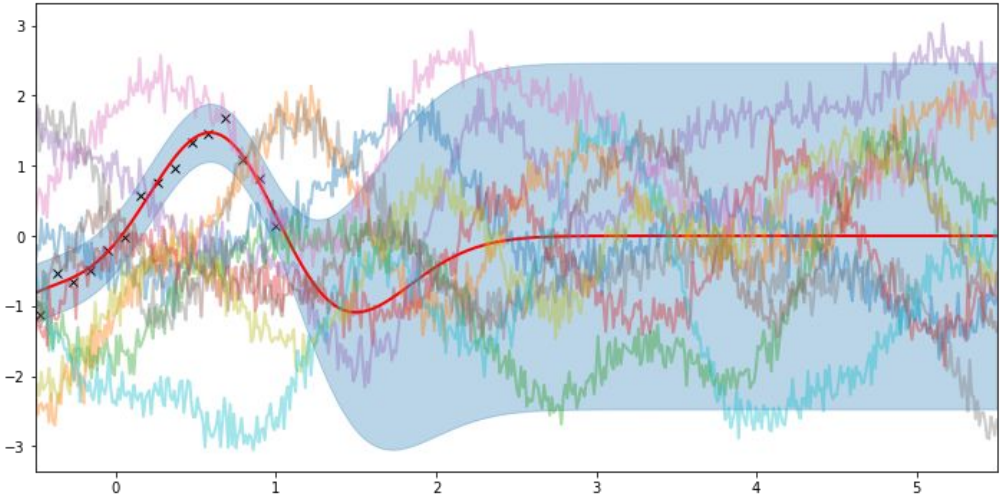
MISIS National University of Science and Technology
University of Zurich[UZH]

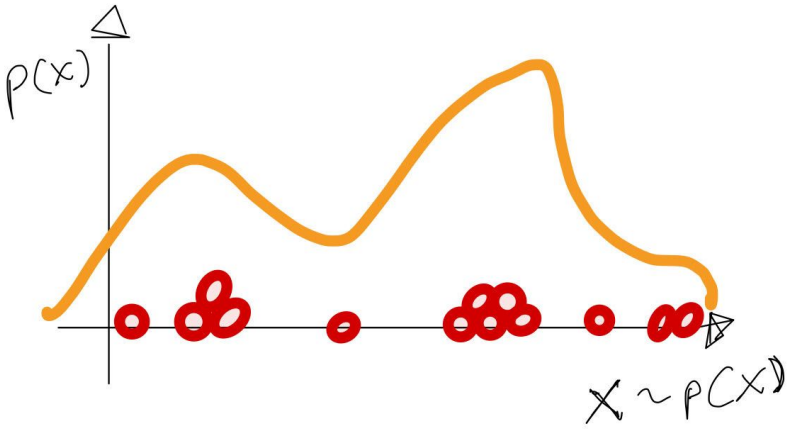
Gaussian processes and Bayesian Optimization

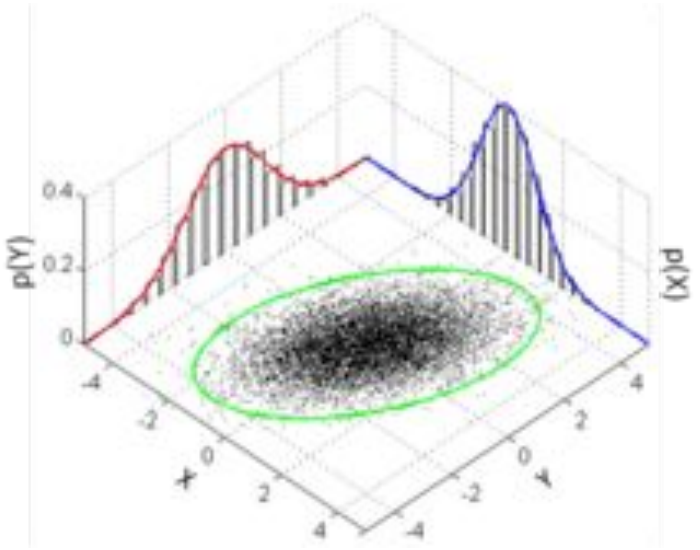
Prof. Dr. Nico Serra - University of Zurich





Sampling from a distribution

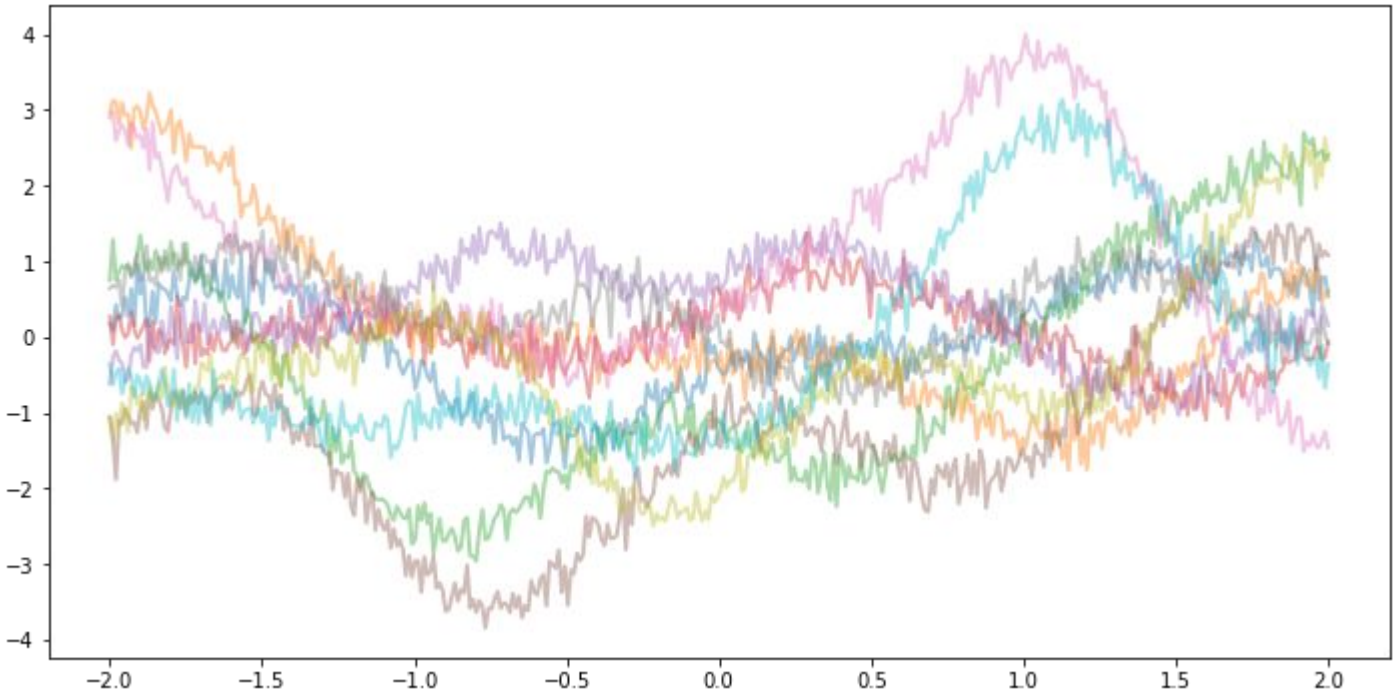




- We know how to draw samples from a n-dimensional distribution
- We also know how to draw from an unnormalized distribution (MCMC)
- Each point is a n-dim vector

Sampling functions

$f(x)$

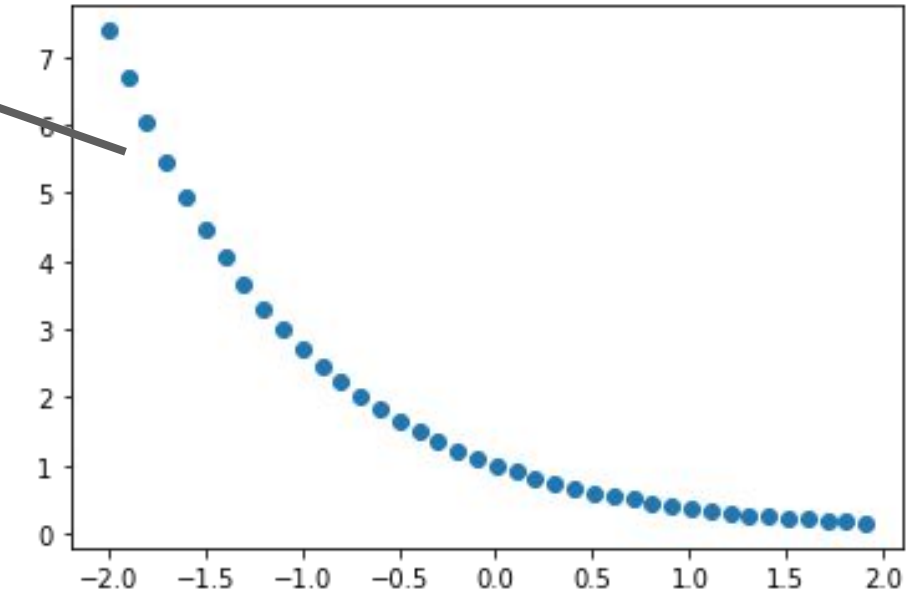


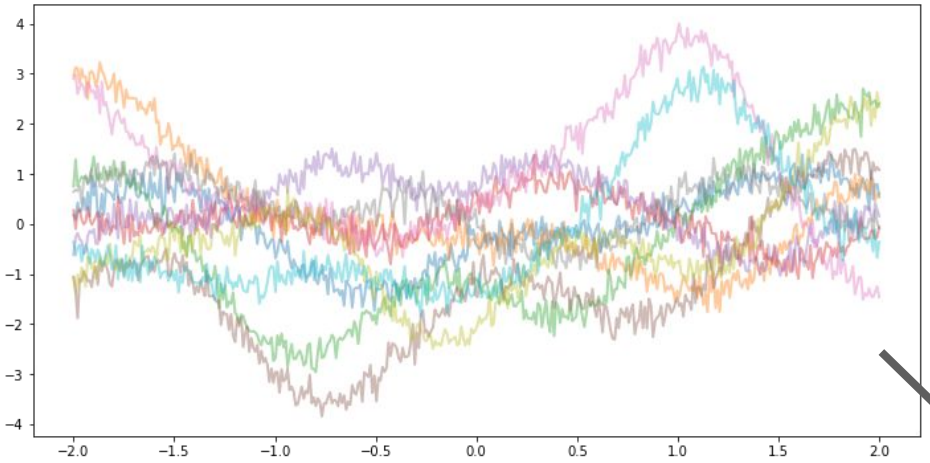
$$f(x) \sim GP(m(x), k(x, x'))$$

- Now we want to approximate data in a non-parameteric way
- We want to sample from functions

Sampling from functions

Each function can be thought as a discrete collection of points



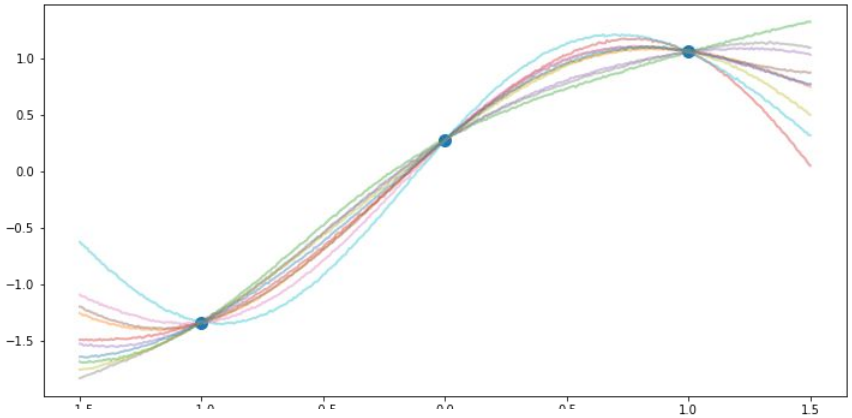


$$GP(\, m(\, x)\, ,\; k(\, x,\; x'\,)\,)$$

Each line is a sample from the process

Sampling from a function

We want to sample from all curves passing by the points I observe



Definition of Gaussian process:
every finite set of function values has a multivariate normal distribution

$$\forall n \quad \forall (x_1, \ldots, x_n) \quad (f(x_1), \ldots, f(x_n)) \sim \mathcal{N}(\mu, \Sigma)$$

Sampling for the points I can obtain continue curves because the correlations among the different points is not zero, but it is a function dependent on x

Sampling from a function

x,y = training data
x* = points for which I want to predict y*
To sample from general curves I need a mean and a variance, the general form is:

$$\begin{cases} \mu = K^* K^{-1} y \\ \mathbb{V} = K^{**} - K^* \cdot K^{-1} K^{*T} \end{cases}$$

K(X,X) = matrix correlation of X,X
K** (X*,X*)= matrix correlation X*,X*
K*(X*,X) = matrix correlation of X,X*

NB: $\mu = y \; and \; \mathbb{V} = 0 \quad for \; K = K^* \; , \; i.e. \; x^* = x$

$$f(x) \sim GP(m(x), k(x, x'))$$

$$\begin{cases} m(x) = 0 \\ k(x, x') = exp\left\{ -\frac{1}{2} \|x - x'\|^2 \right\} = \begin{cases} 1 & for \quad x' \to x \\ 0 & for \quad \|x - x'\| \to \infty \end{cases} \end{cases}$$

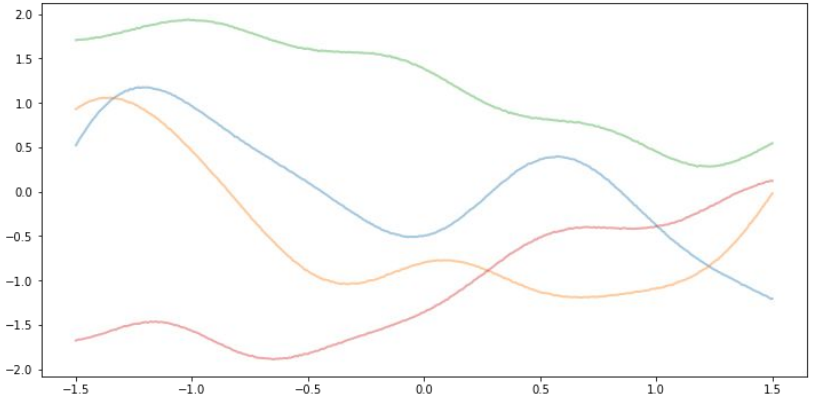
$$K = \begin{pmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{pmatrix}$$

Sampling from a function

$$\begin{cases} m(x) = 0 \\ k(x, x') = exp\left\{-\frac{1}{2}\|x - x'\|^2\right\} = \begin{cases} 1 & for \quad x' \to x \\ 0 & for \quad \|x - x'\| \to \infty \end{cases} \end{cases}$$

$$K = \begin{pmatrix} 1 & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & 1 \end{pmatrix}$$

Drawing randomly from this distribution



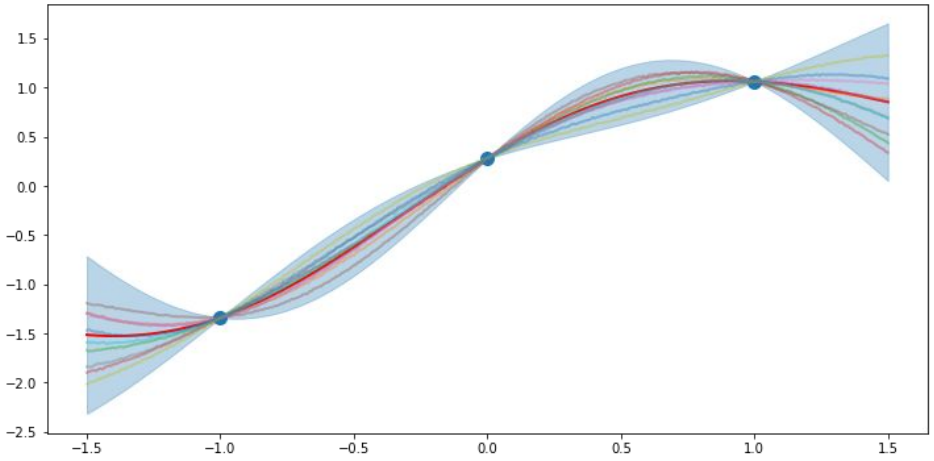
GP Prior

Sampling from a function

We want to use the data we have to predict the data we do not have

$$\begin{pmatrix} y \\ y^* \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_X \\ \mu_{X^*} \end{pmatrix}, \begin{bmatrix} K(X,X) & K(X,X^*) \\ K(X,X^*) & K(X^*,X^*) \end{bmatrix} \right)$$

$$p(y^* \mid x, y) \sim \mathcal{N}\left(\mu_X + K^* K^{-1} (y - \mu_X), K^{**} - K^* K^{-1} K^* \right)$$

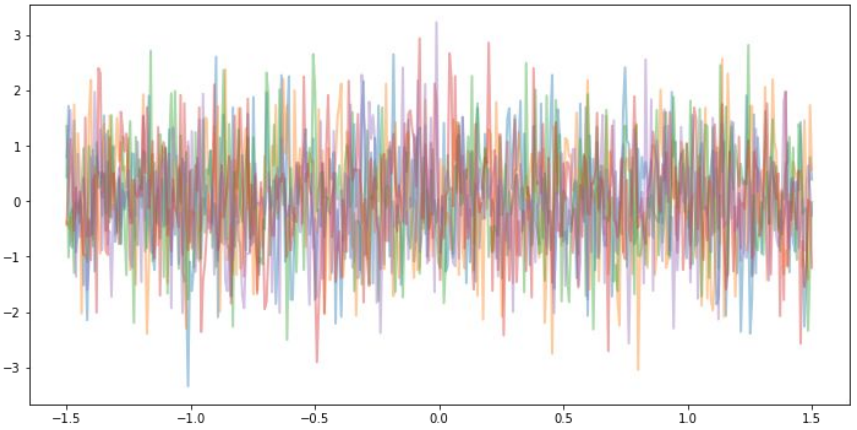


Example 1: white noise

$$f(x) \sim GP(m(x), k(x, x'))$$

$$m(x) = 0$$

$$\begin{cases} k(x, x') = \sigma^2 & \text{if } x = x' \\ 0 & \text{otherwise} \end{cases}$$



$$(f(x_1), \ldots, f(x_n)) \sim \mathcal{N}(\mu, \Sigma)$$

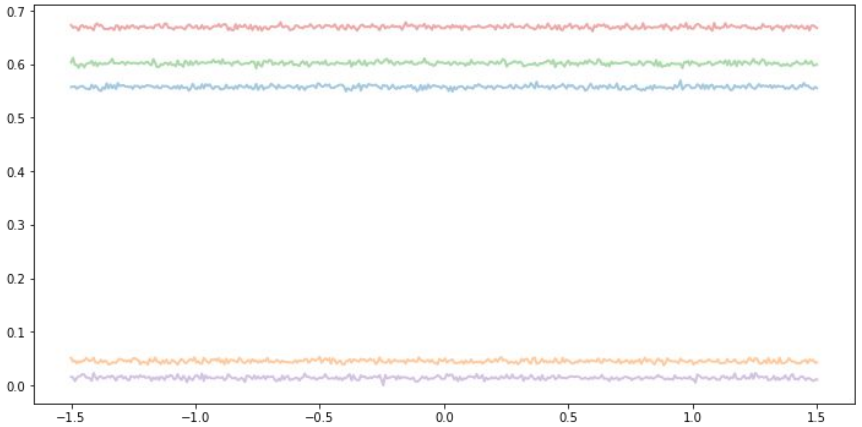
$$\mu = 0 \qquad \Sigma = \sigma^2 I$$

Example 2: Constant

$$f(x) \sim GP(m(x), k(x, x'))$$

$$m(x) = 0$$

$$k(x, x') = C$$



$$(f(x_1), \ldots, f(x_n)) \sim \mathcal{N}(\mu, \Sigma)$$

$$\mu = 0 \qquad \Sigma = \{C\}_{i,j=1}^{n,n}$$

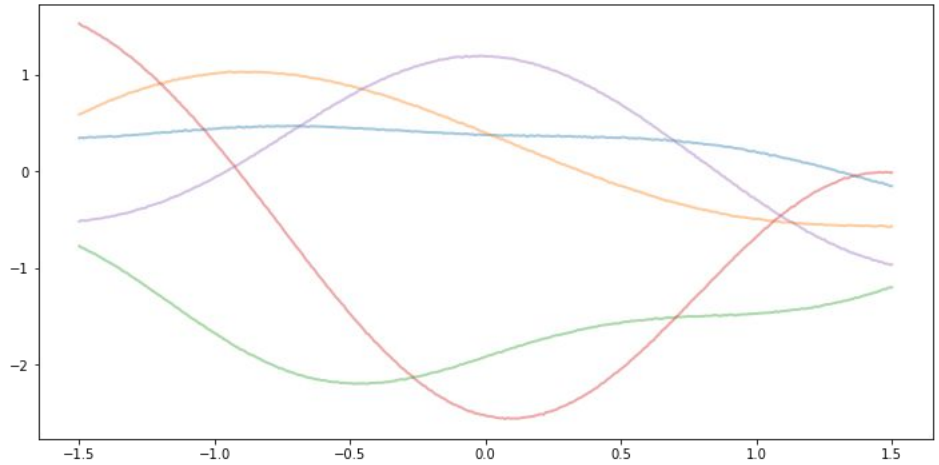
$$\left.\begin{array}{l} \forall i \neq j \\ \mathrm{Corr}(f(x_i), f(x_j)) = \dfrac{\mathrm{Cov}(f(x_i), f(x_j))}{\sqrt{\mathrm{Var}(f(x_i))\mathrm{Var}(f(x_j))}} = \dfrac{C}{\sqrt{C}^2} = 1 \\ \mathrm{Var}(f(x_i)) = \mathrm{Var}(f(x_j)) = C, \quad \mathbb{E}f(x_i) = \mathbb{E}f(x_j) = 0 \end{array}\right\} \Rightarrow f(x_i) = f(x_j)$$

Example 3: RBF-kernel

$$f(x) \sim GP(m(x), k(x, x'))$$

$$m(x) = 0$$

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



$$(f(x_1), \ldots, f(x_n)) \sim \mathcal{N}(\mu, \Sigma)$$

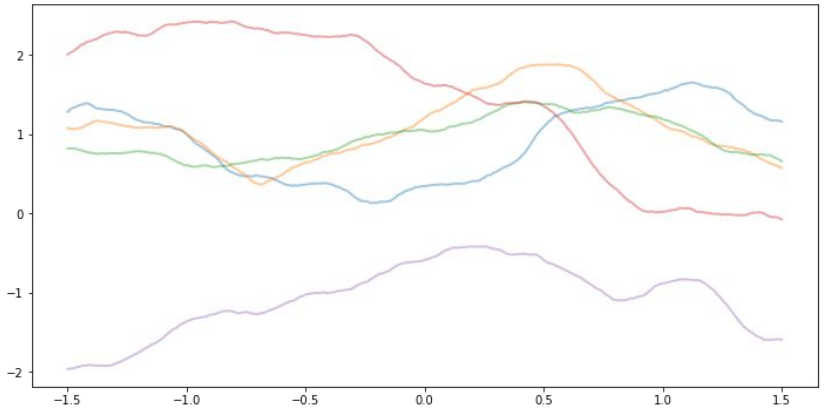
$$\mu = 0 \qquad \Sigma = \{k(x_i, x_j)\}_{i,j=1}^{n,n}$$

$$\text{if } \|x_i - x_j\| \approx 0 \quad \Rightarrow \quad \Sigma_{ij} \approx \sigma^2 = \Sigma_{ii} = \Sigma_{jj} \quad \Rightarrow \quad f(x_i) \approx f(x_j)$$

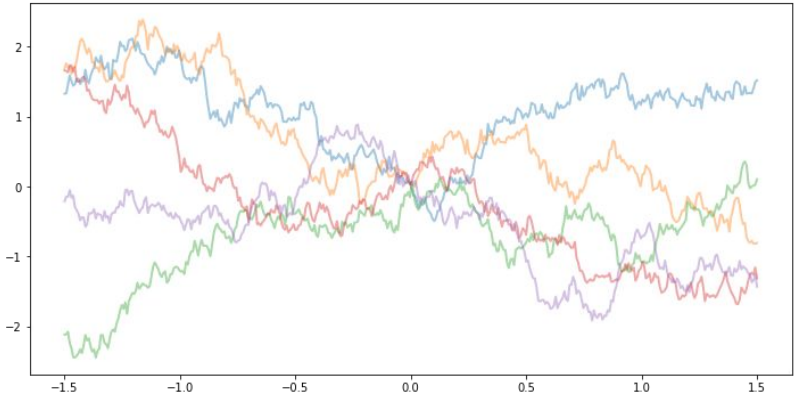
$$\text{if } \|x_i - x_j\| \gg 0 \quad \Rightarrow \quad \Sigma_{ij} \approx 0, \quad f(x_i) \text{ and } f(x_j) \text{ are not correlated}$$

Other Kernels

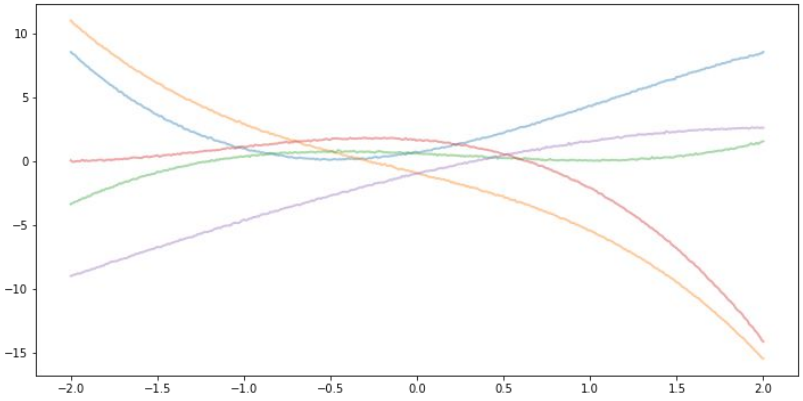
Matern



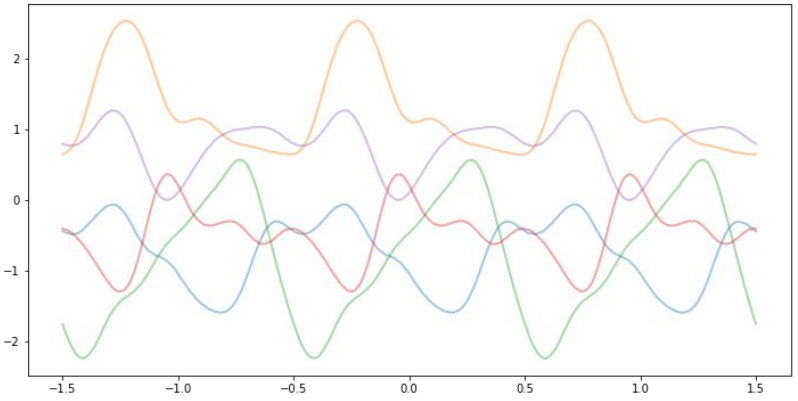
Brownian



Polynomial



Periodic

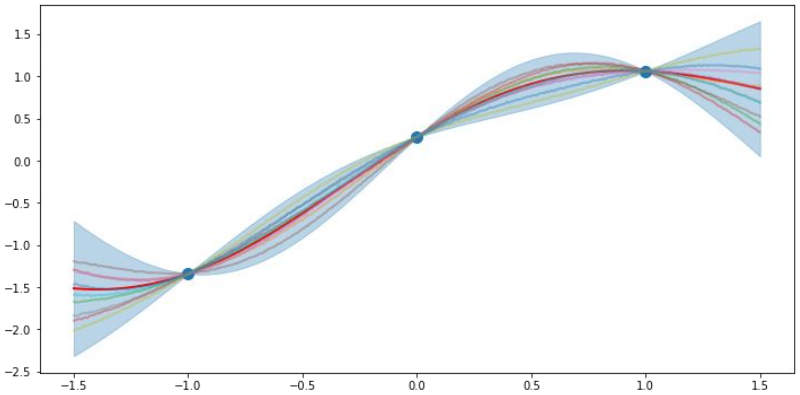


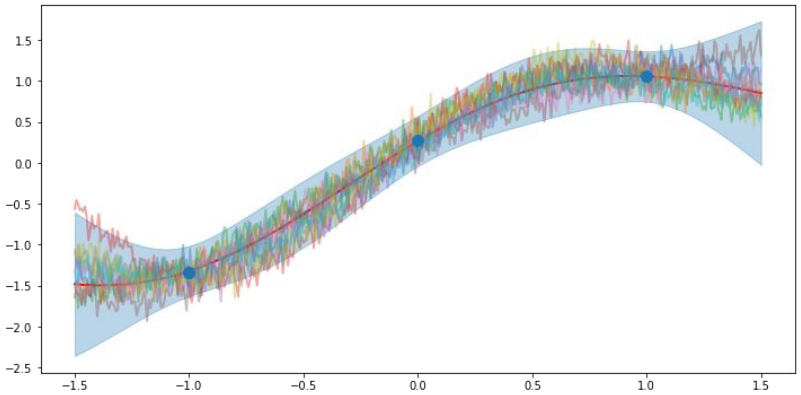
Adding Noise

Forcing data to pass exactly by the points, might be a too strong requirement, e.g. if you have noise or measurement errors on y

$$\begin{cases} y = f(x) + \varepsilon \\ \varepsilon \sim \mathcal{N}(0, \sigma^2) \\ f(x) \sim \mathcal{N}(0, K) \end{cases}$$

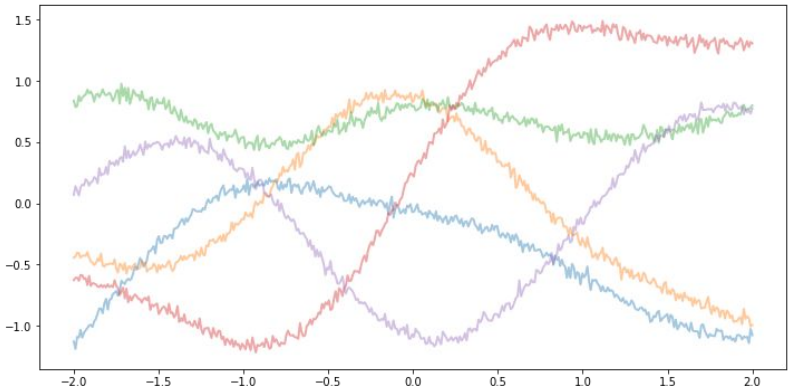
$$\begin{pmatrix} y \\ y^* \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_X \\ \mu_{X^*} \end{pmatrix}, \begin{bmatrix} K(X,X) + \sigma^2 I & K(X,X^*) \\ K(X,X^*) & K(X^*,X^*) \end{bmatrix} \right)$$



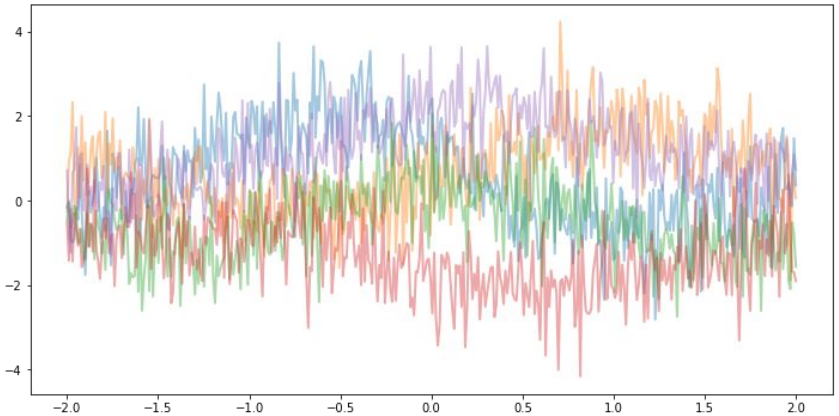


RBF-Kernel

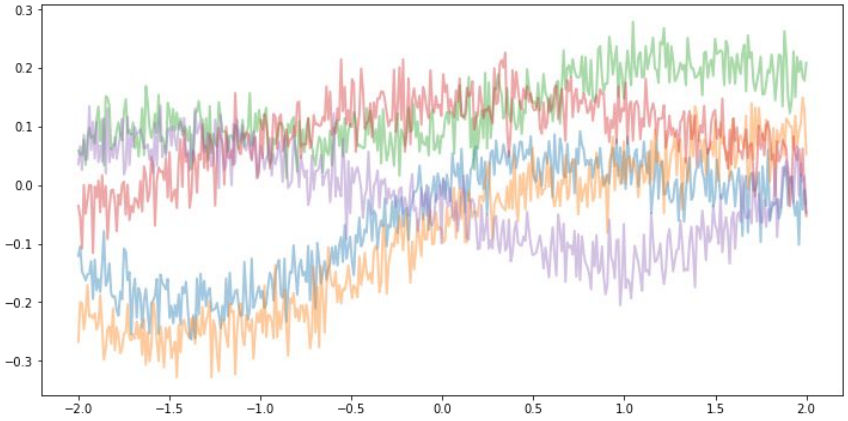
$\sigma = 1 \quad \ell = 1$ **noise=0.01**



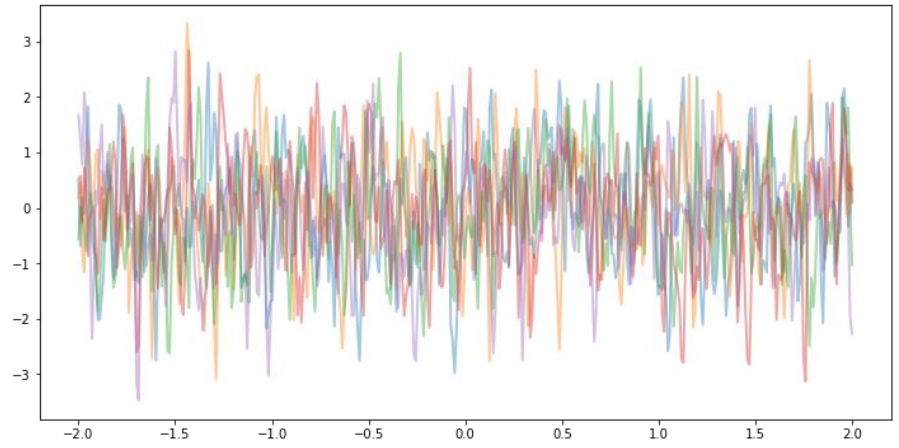
$\sigma = 1 \quad \ell = 1$ **noise=0.5**



$\sigma = 0.01 \quad \ell = 1$ **noise=0.01**



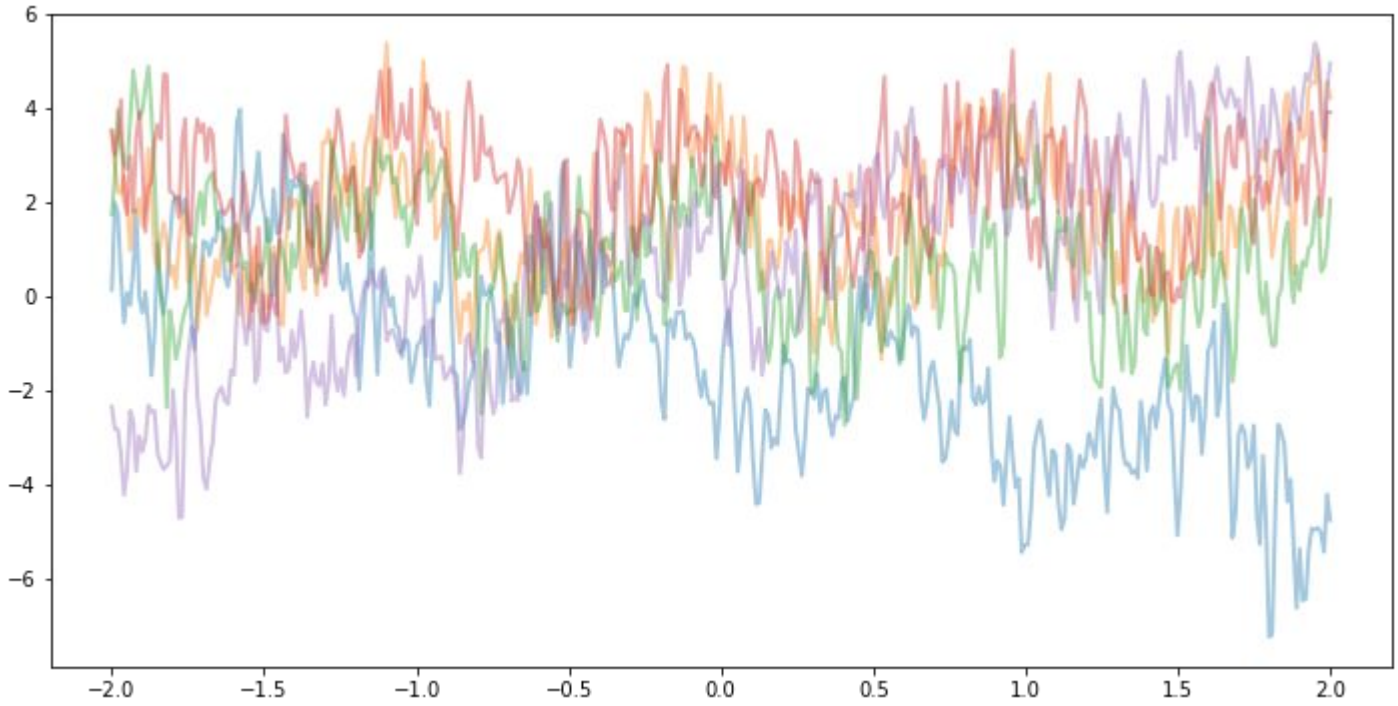
$\sigma = 1 \quad \ell = 0.01$ **noise=0.01**



Kernel Combination

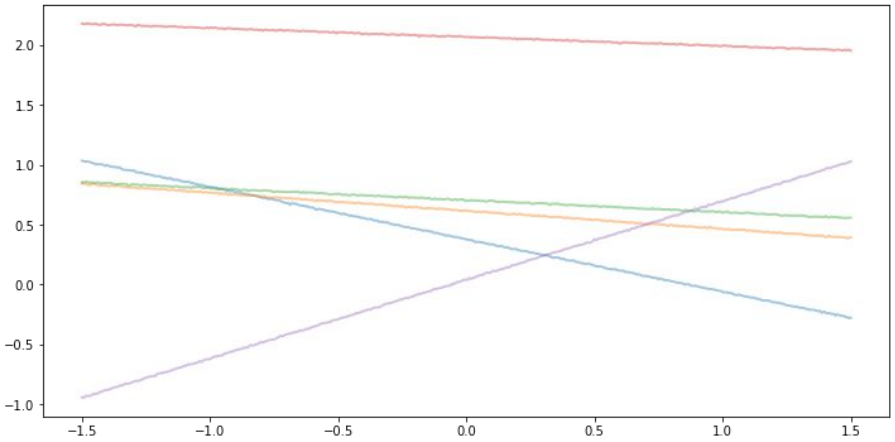
It is possible to combine the kernels, e.g. by summing

$$k(x, x') = \sigma^2[x = x'] \quad + \quad k(x, x') = x \cdot x' \quad + \quad k(x, x') = \sigma^2 \exp\left(-\frac{(x - x')^2}{2r^2}\right)$$



Bayesian view

Let's start with a linear family of curves



$$\begin{cases} y = \omega^T x + \varepsilon \\ \varepsilon \sim \mathcal{N}(0, \sigma^2) \\ f(x) \sim \mathcal{N}(0, K) \end{cases}$$

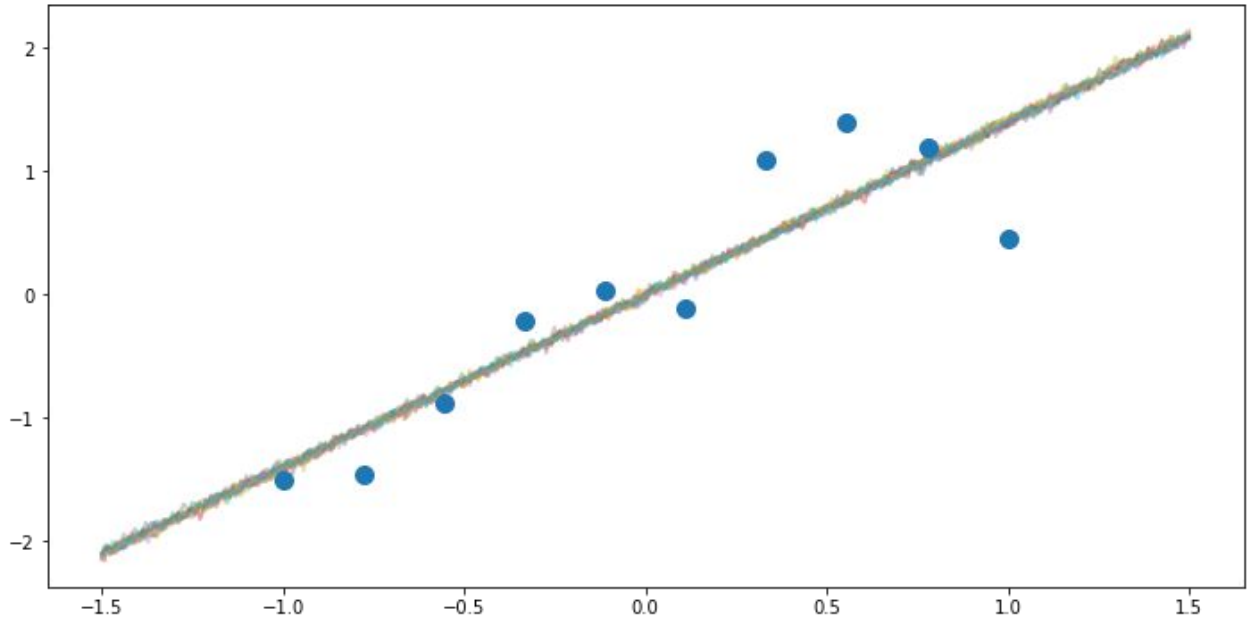
$$\pi(\omega) = \mathcal{N}(0, \Sigma_p)$$ → Gaussian Prior

$$\mathcal{L} \propto \prod_i exp\left(\frac{1}{2\sigma}\|y_i - x_i w\|^2\right) = \mathcal{N}(X\omega, \sigma^2 I)$$ → Gaussian Likelihood

$$p(\omega|X, y) = \mathcal{N}(\omega|X, y)\,\pi(\omega)$$ → Gaussian Posterior

Bayesian view

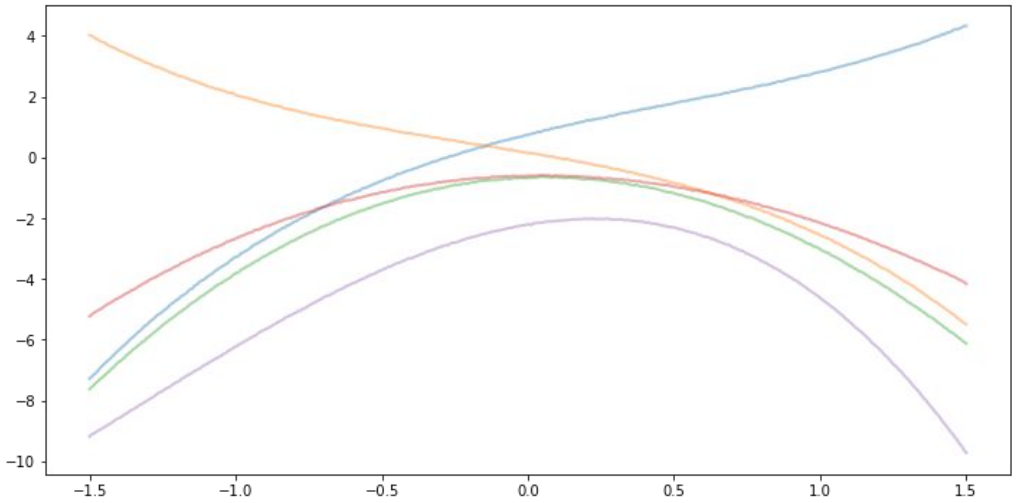
Once I have the posterior, I have found the best linear combination



Of course using linear functions is a limitation, however we can use the same technique with an arbitrary family of curves

Bayesian view

$$\begin{cases} y = \phi(x) + \varepsilon \\ \varepsilon \sim \mathcal{N}(0, \sigma^2) \\ f(x) \sim \mathcal{N}(0, K) \end{cases}$$

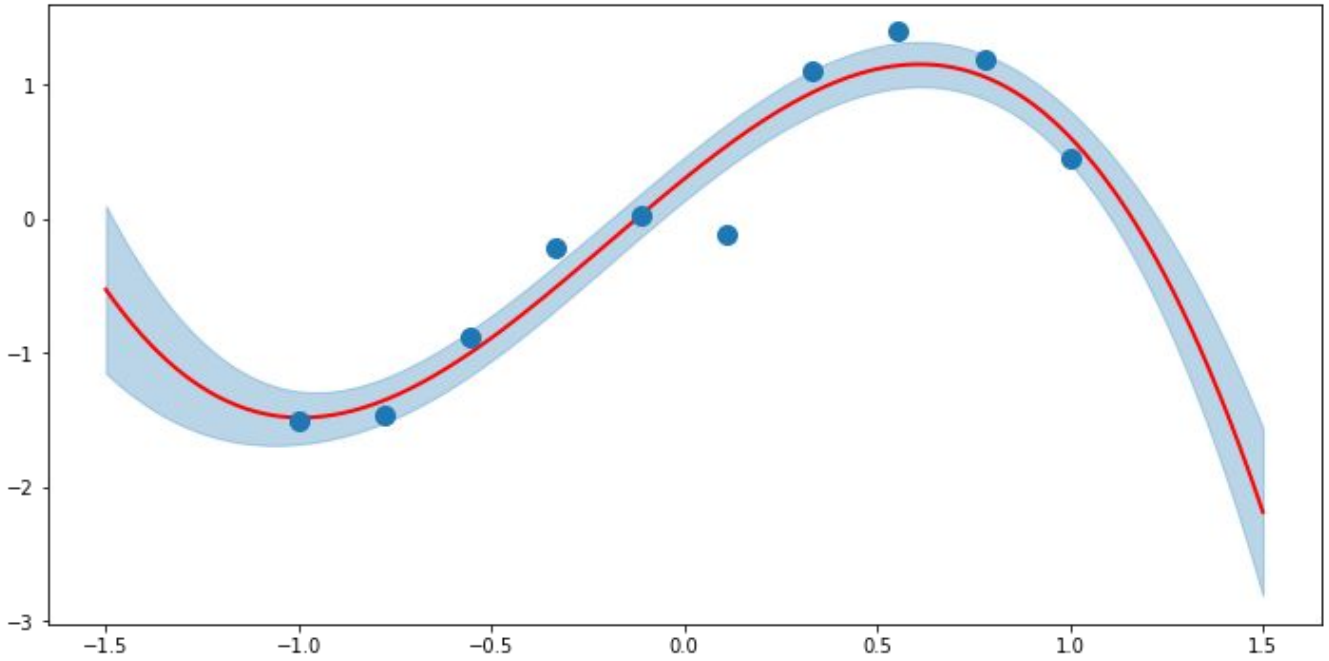


$$\mathcal{L} \propto \prod_i exp\left(\frac{1}{2\sigma}\left\|y_i - \phi(x_i)\right\|^2\right) = \mathcal{N}(\phi(x_i), \Sigma)$$

$$p(\omega | X, y) = \mathcal{N}(\omega | X, y)\, \pi(\omega)$$

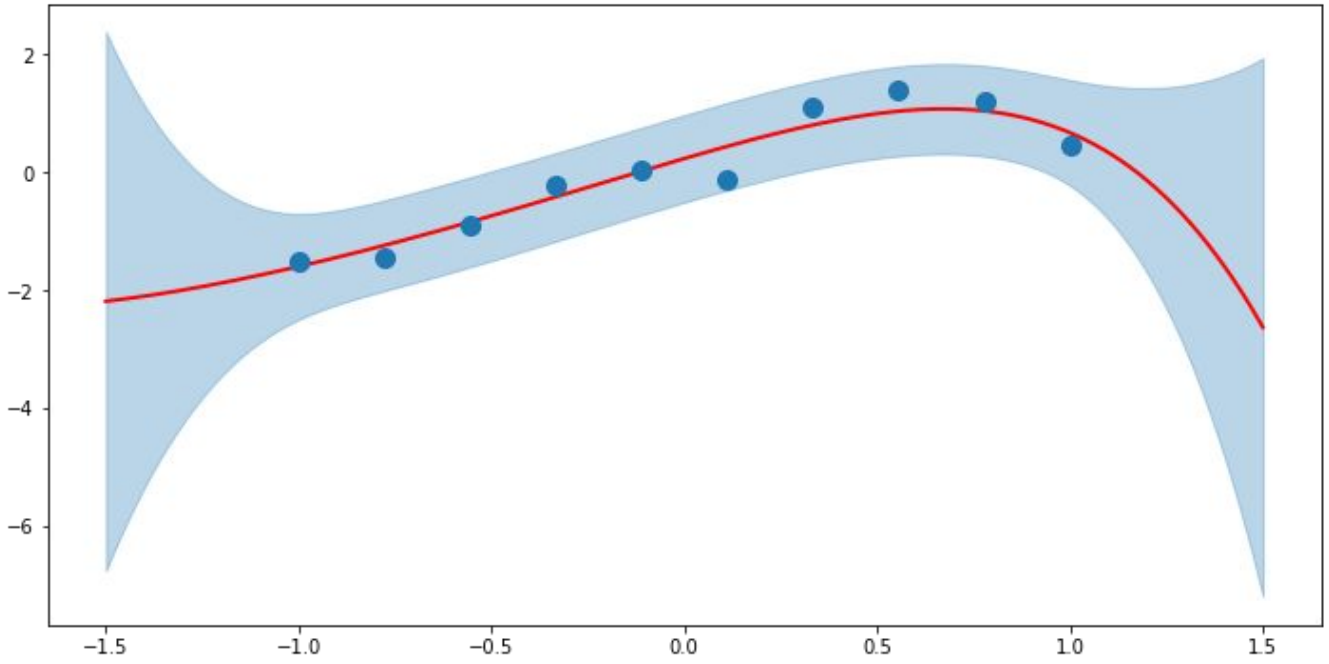
Bayesian view

More expressive basis fits better data…



Bayesian view

More expressive basis fits better data… but could lead to larger uncertainty when extrapolating

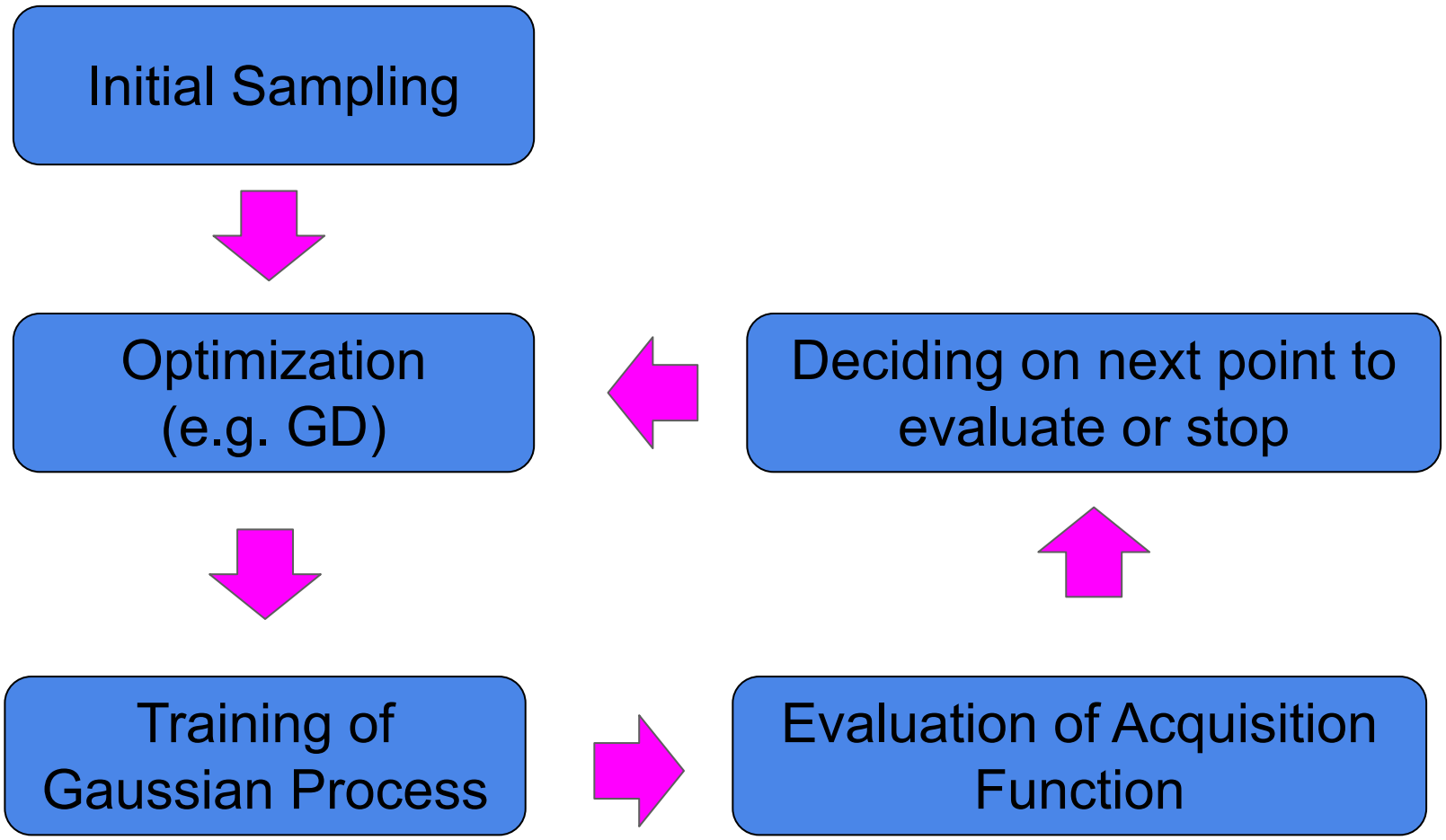


Bayesian Optimization

Bayesian Optimization

- Normally in addition to the network parameters (that you optimize with gradient descent) you have a set of hyperparameters
- Finding good hyperparameters might make the difference between a method working or not-working at all
- There are several strategies you can use to optimize the hyperparameters, e.g. random search
- Bayesian Optimization is an efficient strategy to optimize hyper-parameters

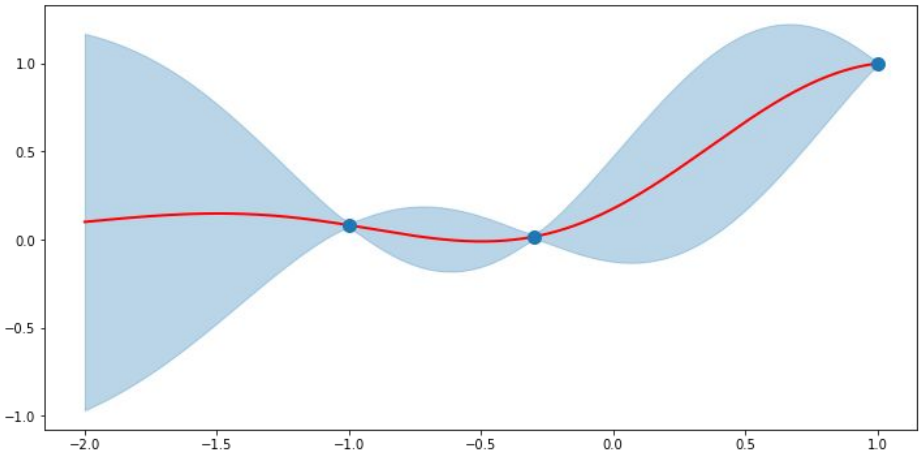
Bayesian Optimization



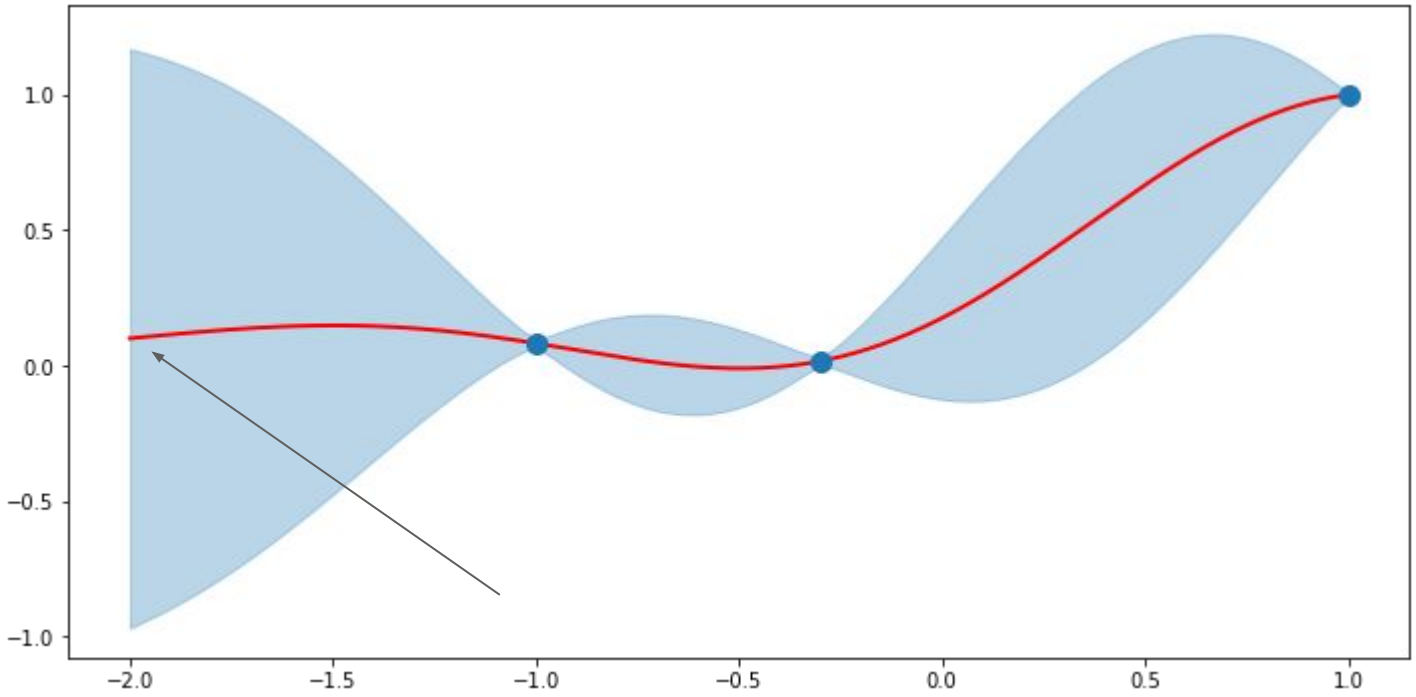
Acquisition Function

- We need a criterion to decide which point I will explore next
- To make this decision we should look at the prediction and also at the uncertainty, e.g. lower coefficient bound, entropy search, expected improvement, ...

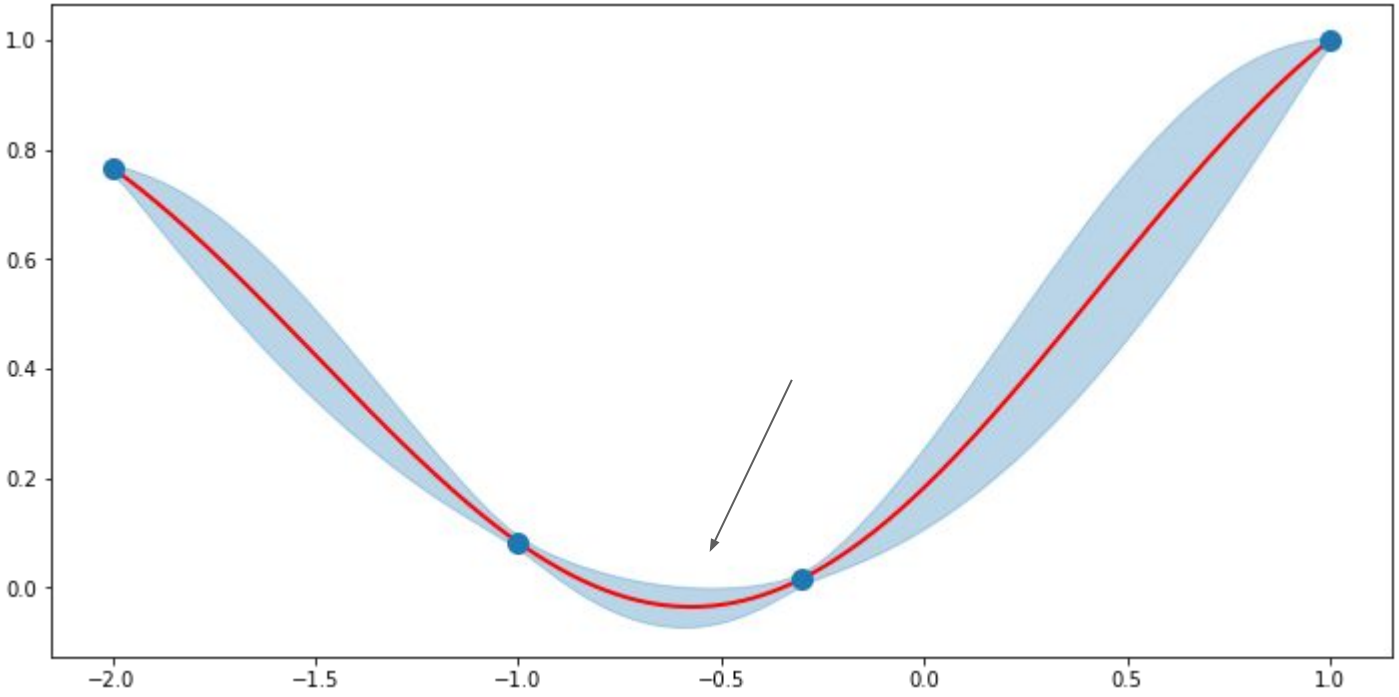
$$a(x) = f(x) - \kappa \cdot \sigma(f(x))$$



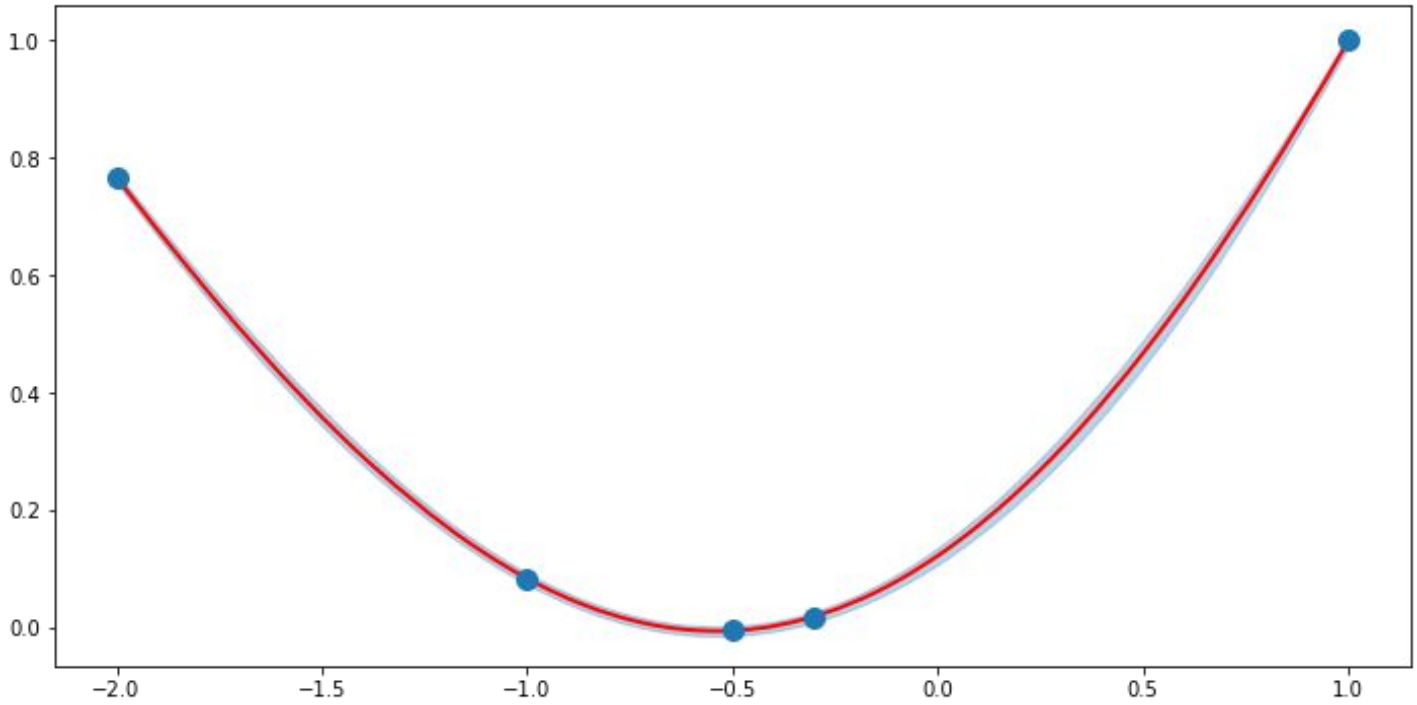
Bayesian Optimization Example



Bayesian Optimization



Bayesian Optimization



Conclusions

- We have learned how to approximate functions with Gaussian Processes
- This is a non-parametric technique, equivalent to KNN for functions
- GP allows to evaluate uncertainties
- We have used 1-dimensional function because it is easier to visualize, but all we said is valid in n-dimensions
- GP are very efficient up to 20-dim
- One very interesting application for Machine learning is bayesian optimization