# AutoML: Gaussian Processes

## Gaussian Processes

Bernd Bischl  Frank Hutter  Lars Kotthoff
Marius Lindauer  Joaquin Vanschoren

# Weight-Space View

- So far, we have considered a hypothesis space $\mathcal{H}$ of parameterized functions $f(\mathbf{x} \mid \boldsymbol{\theta})$ (in particular, the space of linear functions).
- Using Bayesian inference, we derived distributions for $\boldsymbol{\theta}$ after having observed data $\mathcal{D}_{\text{train}}$.
- Prior believes about the parameter are expressed via a prior distribution $q(\boldsymbol{\theta})$, which is updated according to Bayes' rule

$$\underbrace{p(\boldsymbol{\theta} \mid \mathbf{X}, \mathbf{y})}_{\text{posterior}} = \frac{\overbrace{p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})}^{\text{likelihood}} \overbrace{q(\boldsymbol{\theta})}^{\text{prior}}}{\underbrace{p(\mathbf{y} \mid \mathbf{X})}_{\text{marginal}}}.$$

# Function-Space View I

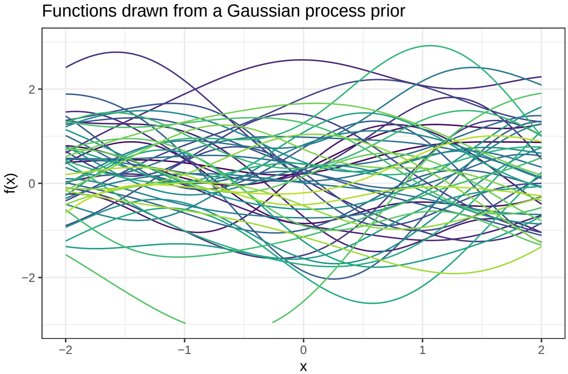
Let us change our point of view:

- Instead of "searching" for a parameter $\boldsymbol{\theta}$ in the parameter space, we directly search in a space of "allowed" functions $\mathcal{H}$.

- We will still use Bayesian inference, but instead of specifying a prior distribution over a parameter, we will specify a prior distribution **over functions** and will update it according to the data points that we observe.

# Function-Space View II

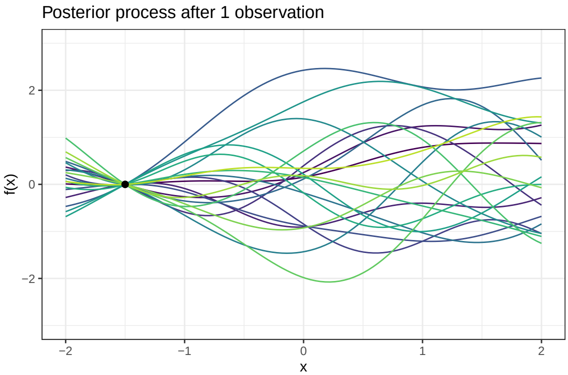
Intuitively, imagine we could draw a huge number of functions from some prior distribution over functions $^{(*)}$.




$^{(*)}$ We will see in a minute how distributions over functions can be specified.

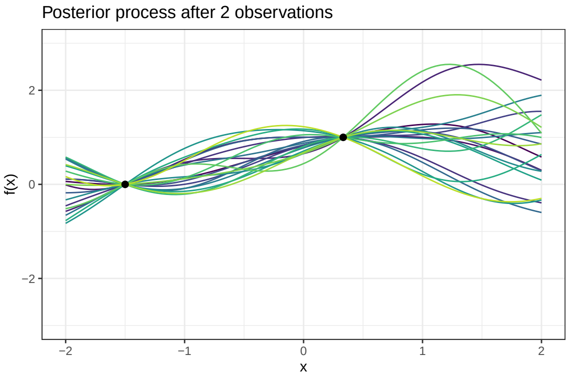
# Function-Space View III

After observing some data points, we are allowed to sample only those functions that are consistent with the data.



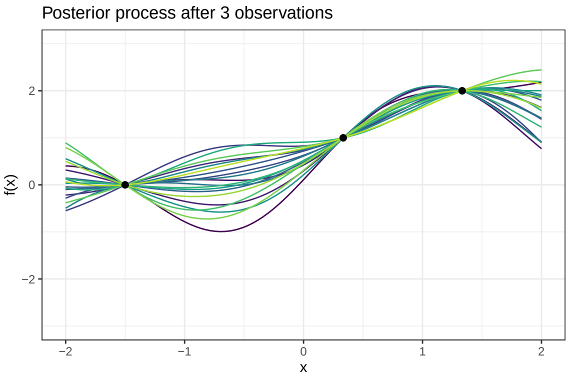
# Function-Space View IV

After observing some data points, we are allowed to sample only those functions that are consistent with the data.



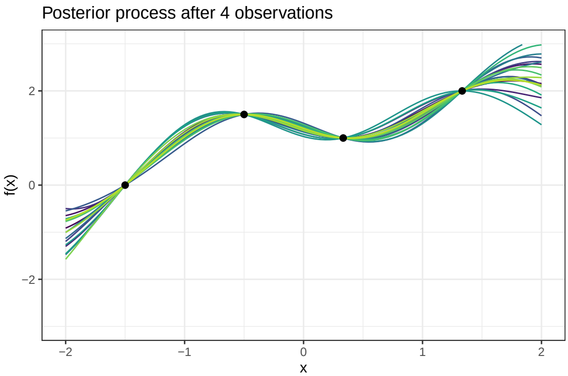
# Function-Space View V

After observing some data points, we are allowed to sample only those functions that are consistent with the data.



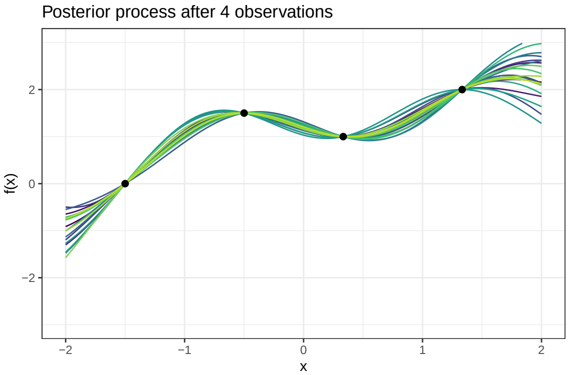
# Function-Space View VI

As we observe more and more data points, the number of functions that consistent with the data shrinks.



# Function-Space View VII

Intuitively, there is something like the "mean" and "variance" of a distribution over functions.



# Weight-Space View vs. Function-Space View

| **Weight-Space View** | **Function-Space View** |
| --- | --- |
| Parameterize functions<br>Example: $f(\mathbf{x} \mid \boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{x}$ | |
| Define distributions on $\boldsymbol{\theta}$ | Define distributions on $f$ |
| Inference in parameter space $\Theta$ | Inference in function space $\mathcal{H}$ |

Next, we will see how we can define distributions over functions mathematically.

# Distributions on Functions

# Discrete Functions I

For simplicity, we will firstly consider functions with finite domains.

- Let $\mathcal{X} = \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right\}$ be a finite set of elements and $\mathcal{H}$ the set of all functions $h : \mathcal{X} \to \mathbb{R}$.

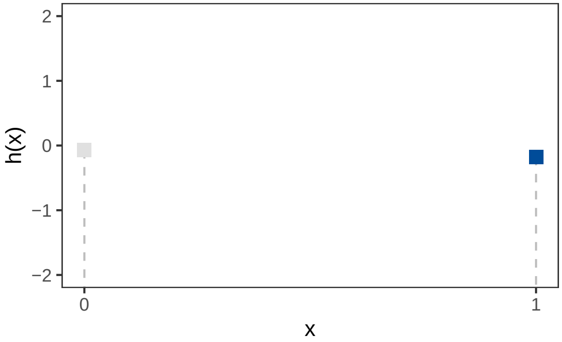
- Since the domain of any $h(\cdot) \in \mathcal{H}$ has only $n$ elements, we can represent the function $h(\cdot)$ compactly as a $n$-dimensional vector

$$\boldsymbol{h} = \left[h\left(\mathbf{x}^{(1)}\right), \ldots, h\left(\mathbf{x}^{(n)}\right)\right].$$

# Discrete Functions II

**Example 1:** Consider function $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **two** points $\mathcal{X} = \{0, 1\}$.

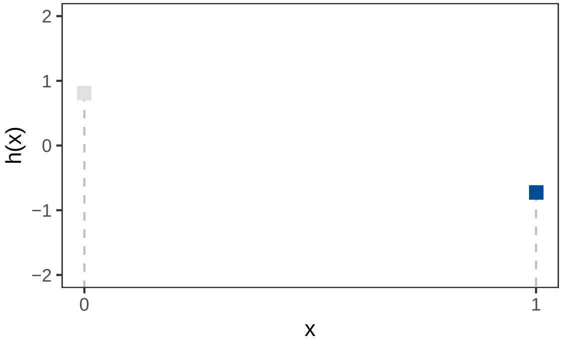
Examples for functions that live in this space:



# Discrete Functions III

**Example 1:** Consider function $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **two** points $\mathcal{X} = \{0, 1\}$.

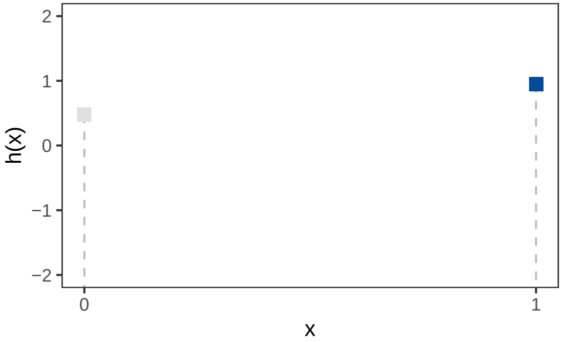
Examples for functions that live in this space:



# Discrete Functions IV

**Example 1:** Consider function $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **two** points $\mathcal{X} = \{0, 1\}$.

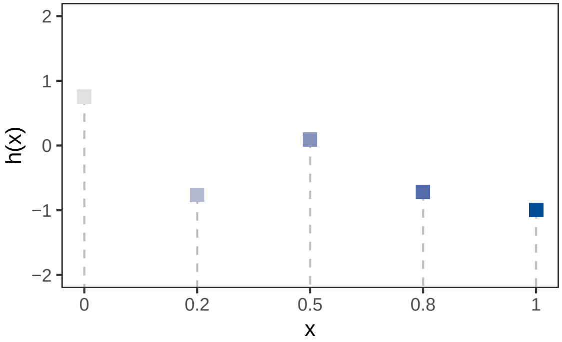
Examples for functions that live in this space:



# Discrete Functions V

**Example 2:** Consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **five** points $\mathcal{X} = \{0, 0.25, 0.5, 0.75, 1\}$.

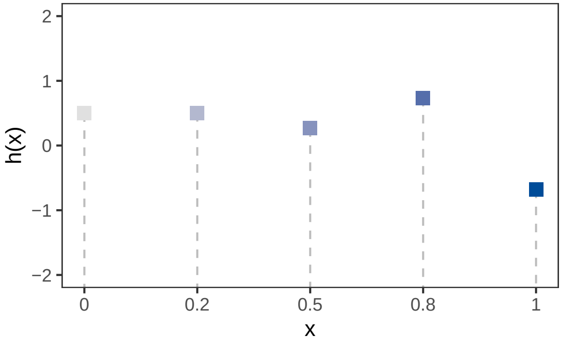
Examples for functions that live in this space:



# Discrete Functions VI

**Example 2:** Consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **five** points $\mathcal{X} = \{0, 0.25, 0.5, 0.75, 1\}$.

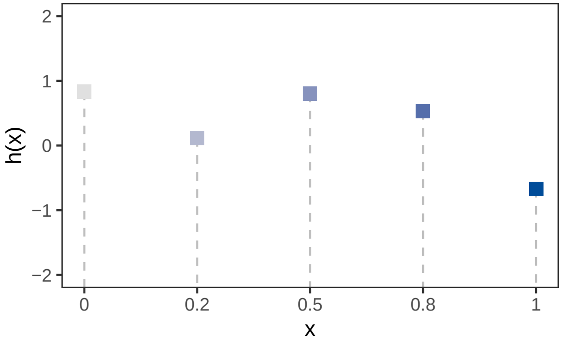
Examples for functions that live in this space:



# Discrete Functions VII

**Example 2:** Consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **five** points $\mathcal{X} = \{0, 0.25, 0.5, 0.75, 1\}$.

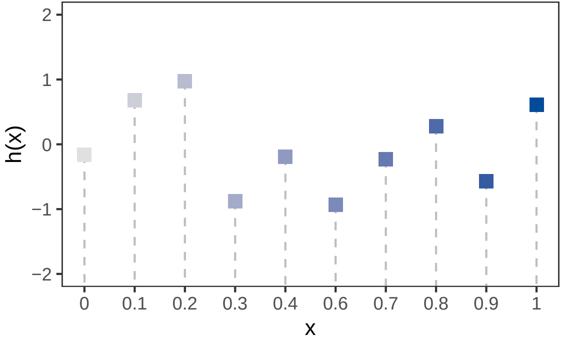
Examples for functions that live in this space:



# Discrete Functions VIII

**Example 3:** Consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **ten** points.

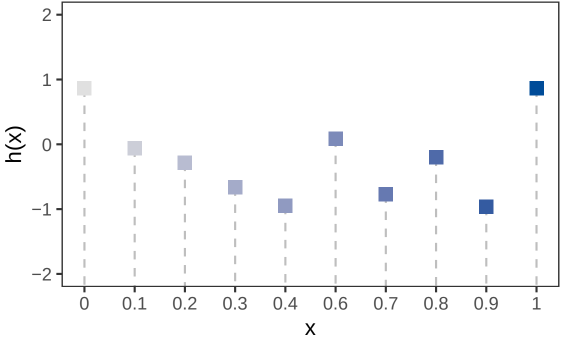
Examples for functions that live in this space:



# Discrete Functions IX

**Example 3:** Consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **ten** points.

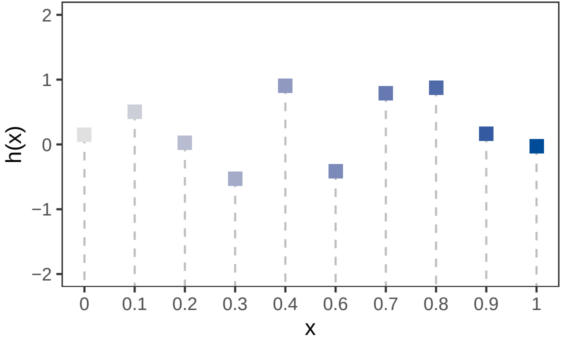
Examples for functions that live in this space:



# Discrete Functions X

**Example 3:** Consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **ten** points.

Examples for functions that live in this space:



# Distributions on Discrete Functions I

- One natural way to specify a probability distribution on a discrete function $h \in \mathcal{H}$ is to use the vector representation of the function:

$$\boldsymbol{h} = \left[h\left(\mathbf{x}^{(1)}\right), h\left(\mathbf{x}^{(2)}\right), \ldots, h\left(\mathbf{x}^{(n)}\right)\right].$$

- Let us consider $\boldsymbol{h}$ as a $n$-dimensional random variable. We will further assume the following normal distribution:

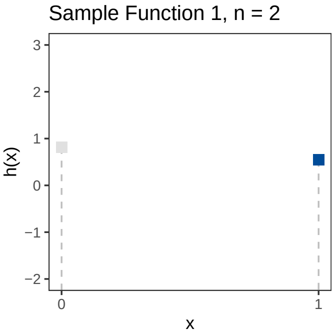
$$\boldsymbol{h} \sim \mathcal{N}\left(\boldsymbol{m}, \boldsymbol{K}\right).$$

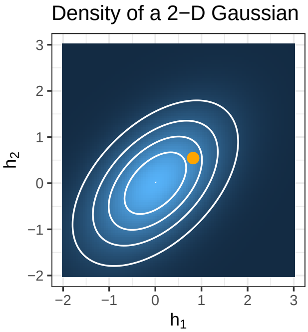
**Note:** For now, we set $\boldsymbol{m} = \mathbf{0}$ and take the covariance matrix $\boldsymbol{K}$ as given. We will see later how they are chosen / estimated.

# Distributions on Discrete Functions II

**Example 1 (continued):** Let $h : \mathcal{X} \to \mathcal{Y}$ be a function that is defined on **two** points $\mathcal{X}$. We sample functions by sampling from a two-dimensional normal variable

$$\boldsymbol{h} = [h(1), h(2)] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$



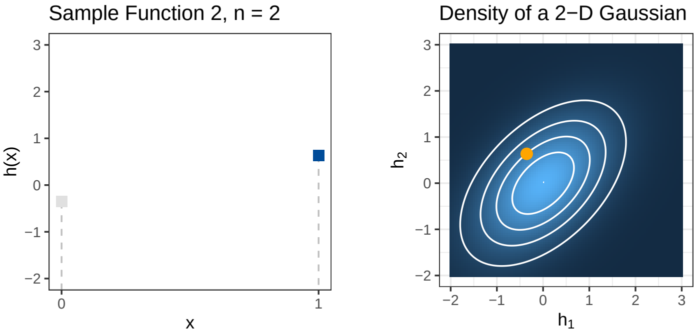




In this example, $m = (0, 0)$ and $K = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$.

# Distributions on Discrete Functions III

**Example 1 (continued):** Let $h : \mathcal{X} \to \mathcal{Y}$ be a function that is defined on **two** points $\mathcal{X}$. We sample functions by sampling from a two-dimensional normal variable

$$\boldsymbol{h} = [h(1), h(2)] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$

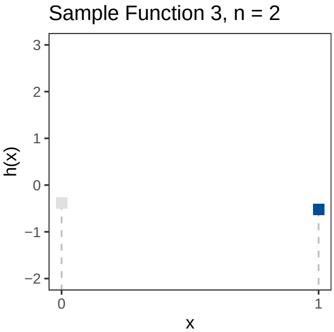


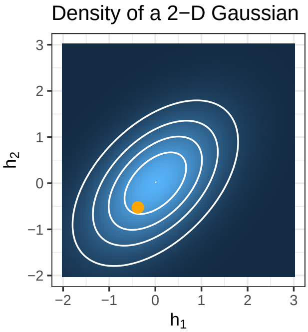

In this example, $m = (0, 0)$ and $K = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$.

# Distributions on Discrete Functions IV

**Example 1 (continued):** Let $h : \mathcal{X} \to \mathcal{Y}$ be a function that is defined on **two** points $\mathcal{X}$. We sample functions by sampling from a two-dimensional normal variable

$$\boldsymbol{h} = [h(1), h(2)] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$



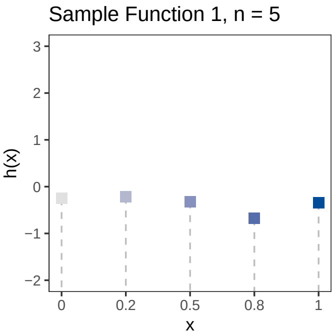



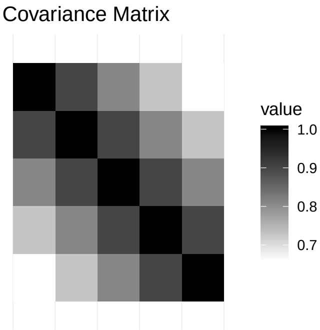

In this example, $m = (0, 0)$ and $K = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$.

# Distributions on Discrete Functions V

**Example 2 (continued):** Let us consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **five** points. We sample functions by sampling from a five-dimensional normal variable

$$\boldsymbol{h} = [h(1), h(2), h(3), h(4), h(5)] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$

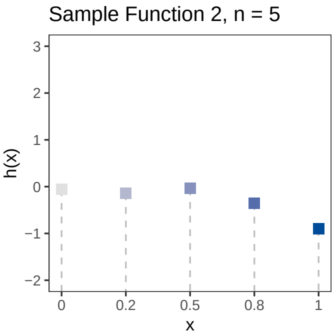


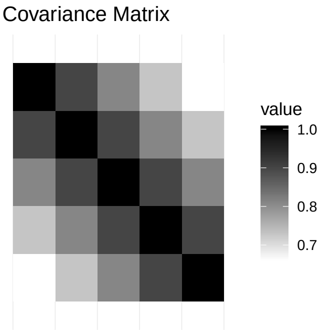



# Distributions on Discrete Functions VI

**Example 2 (continued):** Let us consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **five** points. We sample functions by sampling from a five-dimensional normal variable

$$\boldsymbol{h} = [h(1), h(2), h(3), h(4), h(5)] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$

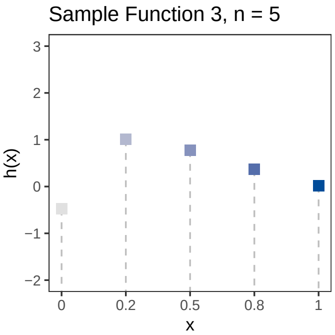


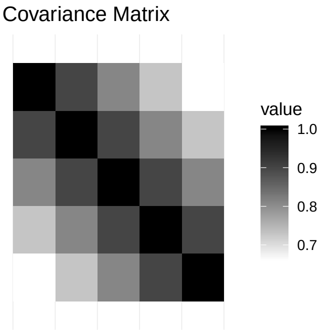



# Distributions on Discrete Functions VII

**Example 2 (continued):** Let us consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **five** points. We sample functions by sampling from a five-dimensional normal variable

$$\boldsymbol{h} = [h(1), h(2), h(3), h(4), h(5)] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$

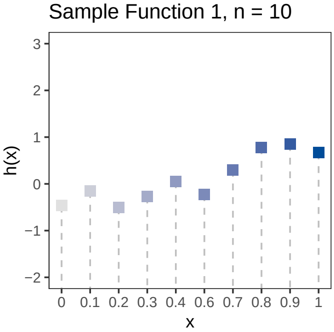


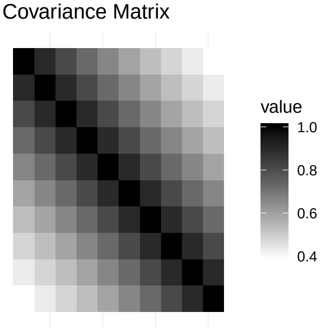



# Distributions on Discrete Functions VIII

**Example 3 (continued):** Let us consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **ten** points. We sample functions by sampling from a ten-dimensional normal variable

$$\boldsymbol{h} = [h(1), h(2), \ldots, h(10)] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$

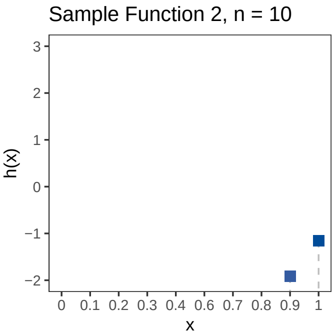


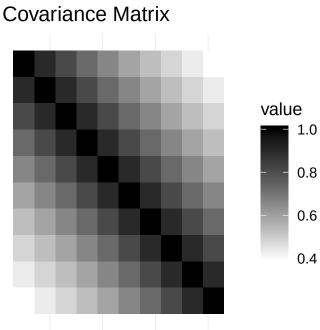



# Distributions on Discrete Functions IX

**Example 3 (continued):** Let us consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **ten** points. We sample functions by sampling from a ten-dimensional normal variable

$$\boldsymbol{h} = [h(1), h(2), \ldots, h(10)] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$

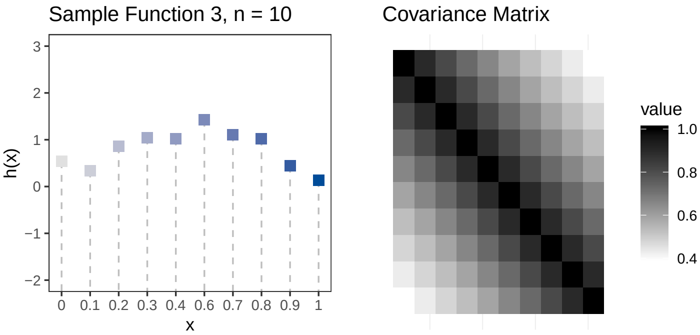


# Distributions on Discrete Functions X

**Example 3 (continued):** Let us consider $h : \mathcal{X} \to \mathcal{Y}$ where the input space consists of **ten** points. We sample functions by sampling from a ten-dimensional normal variable

$$\boldsymbol{h} = [h(1), h(2), \ldots, h(10)] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$

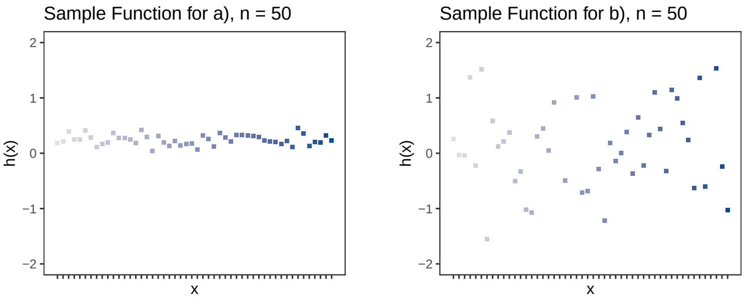


# The Role of Covariance Function I

The covariance controls the "shape" of drawn functions. Consider two extreme cases where function values are:

a) strongly correlated: $\boldsymbol{K} = \begin{pmatrix} 1 & 0.99 & \dots & 0.99 \\ 0.99 & 1 & \dots & 0.99 \\ 0.99 & 0.99 & \ddots & 0.99 \\ 0.99 & \dots & 0.99 & 1 \end{pmatrix}$

b) uncorrelated: $\boldsymbol{K} = \boldsymbol{I}$.



# The Role of Covariance Function II

- On a numeric space $\mathcal{X}$, "meaningful" functions may be characterized by the following spatial property:

  *If* $\mathbf{x}^{(i)}$ *and* $\mathbf{x}^{(j)}$ *are close in the* $\mathcal{X}$*-space, their function values* $f(\mathbf{x}^{(i)})$ *and* $f(\mathbf{x}^{(j)})$ *should be close in* $\mathcal{Y}$*-space.*

- In other words, if two data points are close in $\mathcal{X}$-space, their corresponding values should be **correlated**!

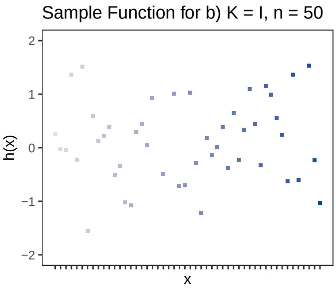
- We can enforce this condition by choosing a covariance function for which,

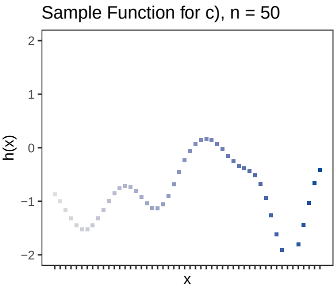
$$\boldsymbol{K}_{ij} \text{ is high, if } \mathbf{x}^{(i)} \text{ and } \mathbf{x}^{(j)} \text{ are close.}$$

# The Role of Covariance Function III

We can compute the entries of the covariance matrix by a function that is based on the distance between $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$. For example:

c) spatial correlation: $K_{ij} = k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(-\frac{1}{2}\left|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right|^2\right)$




**Note**: $k(\cdot, \cdot)$ is known as the **covariance function** or **kernel**. It will be studied in more detail later on.

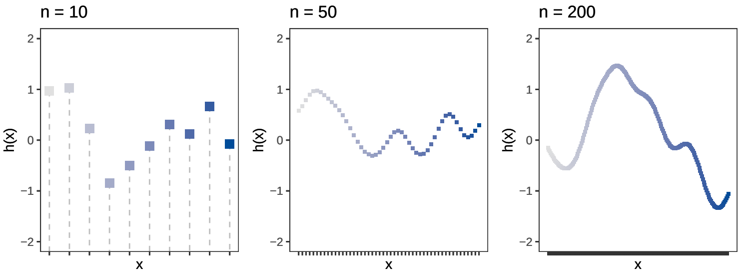
# Gaussian Processes

# From Discrete to Continuous Functions

- We have already considered distributions on functions with discrete domain. We did so, by defining Gaussian distributions on the vector of the respective function values

$$\mathbf{h} = [h(\mathbf{x}^{(1)}), h(\mathbf{x}^{(2)}), \ldots, h(\mathbf{x}^{(n)})] \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}).$$

- We can generalize this idea for $n \to \infty$.



# Gaussian Processes: Intuition I

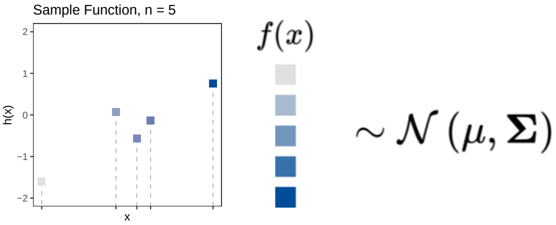
- No matter how large $n$ is, we consider functions with discrete domains.
- But, how can we extend our definition to functions with **continuous** domains $\mathcal{X} \subset \mathbb{R}$?
- Intuitively, a function $f$ drawn from a **Gaussian process** can be understood as an "infinite" long Gaussian random vector.
- It is unclear how to handle an "infinite" long Gaussian random vector!

# Gaussian Processes: Intuition II

- Thus, it is required that for **any finite set** of inputs $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)}\} \subset \mathcal{X}$, the vector $\mathbf{f}$ has a Gaussian distribution with $\boldsymbol{m}$ and $\boldsymbol{K}$ being calculated by a mean function $m(\cdot)$ and a covariance function $k(\cdot, \cdot)$:

$$\boldsymbol{f} = \left[f\left(\mathbf{x}^{(1)}\right), \dots, f\left(\mathbf{x}^{(n)}\right)\right] \sim \mathcal{N}\left(\boldsymbol{m}, \boldsymbol{K}\right).$$

- This property is called the **Marginalization Property**.

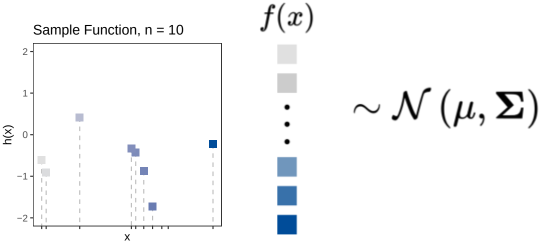


# Gaussian Processes: Intuition III

- Thus, it is required that for **any finite set** of inputs $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)}\} \subset \mathcal{X}$, the vector $\mathbf{f}$ has a Gaussian distribution with $\boldsymbol{m}$ and $\boldsymbol{K}$ being calculated by a mean function $m(\cdot)$ and a covariance function $k(\cdot, \cdot)$:

$$\boldsymbol{f} = \left[ f\left(\mathbf{x}^{(1)}\right), \dots, f\left(\mathbf{x}^{(n)}\right) \right] \sim \mathcal{N}\left(\boldsymbol{m}, \boldsymbol{K}\right).$$

- This property is called the **Marginalization Property**.

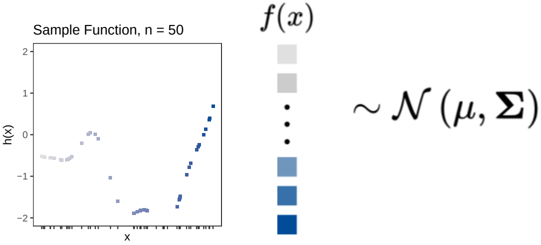


# Gaussian Processes: Intuition IV

- Thus, it is required that for **any finite set** of inputs $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)}\} \subset \mathcal{X}$, the vector $\mathbf{f}$ has a Gaussian distribution with $\boldsymbol{m}$ and $\boldsymbol{K}$ being calculated by a mean function $m(\cdot)$ and a covariance function $k(\cdot, \cdot)$:

$$\boldsymbol{f} = \left[f\left(\mathbf{x}^{(1)}\right), \dots, f\left(\mathbf{x}^{(n)}\right)\right] \sim \mathcal{N}\left(\boldsymbol{m}, \boldsymbol{K}\right).$$

- This property is called the **Marginalization Property**.



# Gaussian Processes: Formal Definitions I

- The above intuitive explanation is formally defined as follows.

*A function $f(\mathbf{x})$ is generated by a Gaussian process $\mathcal{G}$ if for* **any finite** *set of inputs $\left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right\}$, the associated vector of function values has a Gaussian distribution:*

$$\boldsymbol{f} = \left(f(\mathbf{x}^{(1)}), \ldots, f(\mathbf{x}^{(n)})\right) \sim \mathcal{N}\Big(\mathbf{m}, \mathbf{K}\Big),$$

*with*

$$\mathbf{m} := \left(m\left(\mathbf{x}^{(i)}\right)\right)_i, \quad \mathbf{K} := \left(k\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right)\right)_{i,j},$$

*where $m(\mathbf{x})$ is called mean function and $k(\mathbf{x}, \mathbf{x}')$ is called covariance function.*

# Gaussian Processes: Formal Definitions II

- A GP is **completely specified** by its mean and covariance functions.
- The mean function $m(\mathbf{x})$ and the covariance function $k(\mathbf{x}, \mathbf{x}')$ of a real process $f(\mathbf{x})$ are defined as:

$$
\begin{aligned}
m(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})] \\
k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}\Big[(f(\mathbf{x}) - \mathbb{E}[f(\mathbf{x})])\left(f(\mathbf{x}') - \mathbb{E}[f(\mathbf{x}')]\right)\Big]
\end{aligned}
$$

- We denote a GP by

$$f(\mathbf{x}) \sim \mathcal{G}\left(m(\mathbf{x}), k\left(\mathbf{x}, \mathbf{x}'\right)\right)$$

**Note:** For now, we assume $m(\mathbf{x}) \equiv 0$. This is not a drastic limitation. In fact, it is common to consider GPs with a zero mean function.

# Sampling from a Gaussian Process Prior I

- We can draw functions from a Gaussian process prior. To do so, consider $f(\mathbf{x}) \sim \mathcal{G}\left(0, k(\mathbf{x}, \mathbf{x}')\right)$ with the squared exponential covariance function $^{(*)}$

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2\ell^2}\|\mathbf{x} - \mathbf{x}'\|^2\right), \quad \ell = 1.$$

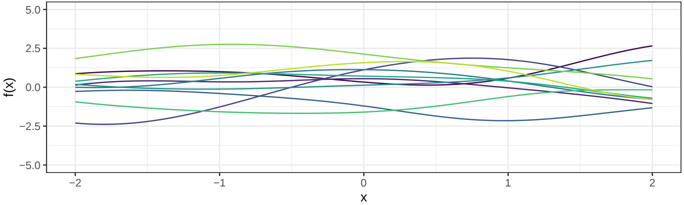
- This covariance function specifies the Gaussian process completely.

$^{(*)}$ We will talk later about different choices of covariance functions.

# Sampling from a Gaussian Process Prior II

To visualize a sample function, we

- choose a large number of equidistant points: $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$,
- compute their corresponding covariance matrix by plugging in all pairs of $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ in $\mathbf{K} = \left(k\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right)\right)_{i,j}$,
- sample from a Gaussian $\boldsymbol{f} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K})$.




We draw $10$ times from the Gaussian, to get $10$ different samples. Since we specified the mean function to be zero, the drawn functions have a zero mean.

# Gaussian Processes as an Indexed Family

# Gaussian Processes as an Indexed Family

- A Gaussian process is a special case of a **stochastic process** which is defined as a collection of random variables indexed by some index set (also called an **indexed family**).

- What does it mean?

- An **indexed family** is a mathematical function (or "rule") that maps indices $t \in T$ to objects in $\mathcal{S}$.

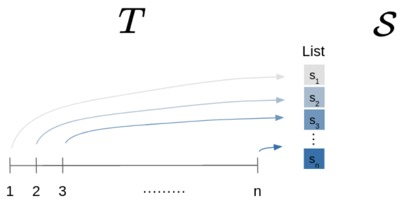
**Definition:** *an* **index family** *(or a family of elements in $\mathcal{S}$ indexed by $T$) is a surjective function that is defined as follows:*

$$\begin{aligned} s : T &\rightarrow \mathcal{S} \\ t &\mapsto s_t = s(t) \end{aligned}$$

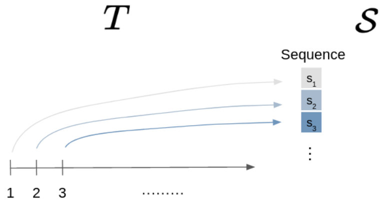
# Index Family I

Some simple examples for indexed families are:

- Finite sequences (lists): $T = \{1, 2, \ldots, n\}$ and $(s_t)_{t \in T} \in \mathbb{R}$



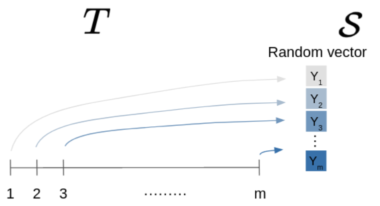

- Infinite sequences: $T = \mathbb{N}$ and $(s_t)_{t \in T} \in \mathbb{R}$

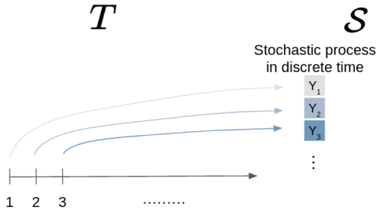


# Index Family II

But the indexed set $\mathcal{S}$ can be something more complicated, for example functions or **random variables** (RV):

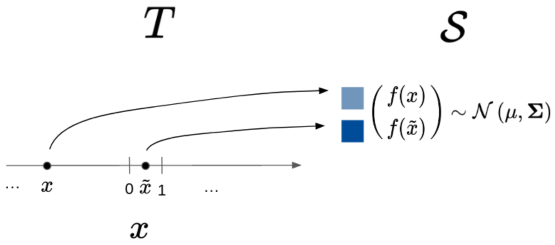
- $T = \{1, \dots, m\}$, $Y_t$'s are RVs: Indexed family is a random vector.
- $T = \{1, \dots, m\}$, $Y_t$'s are RVs: Indexed family is a stochastic process in discrete time.
- $T = \mathbb{Z}^2$, $Y_t$'s are RVs: Indexed family is a 2D-random walk.



# Index Family III

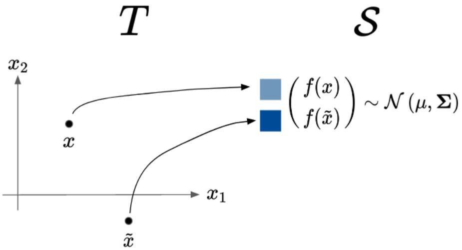
- A Gaussian process is also an indexed family, where the random variables $f(\mathbf{x})$ are indexed by the input values $\mathbf{x} \in \mathcal{X}$.
- Importantly, any indexed (finite) random vector has a multivariate Gaussian distribution (which comes with all the nice properties of Gaussianity!).




Visualization for a one-dimensional $\mathcal{X}$.

# Index Family IV

- A Gaussian process is also an indexed family, where the random variables $f(\mathbf{x})$ are indexed by the input values $\mathbf{x} \in \mathcal{X}$.
- Importantly, any indexed (finite) random vector has a multivariate Gaussian distribution (which comes with all the nice properties of Gaussianity!).




Visualization for a two-dimensional $\mathcal{X}$.

# AutoML: Gaussian Processes

## Covariance Functions for GPs

Bernd Bischl    Frank Hutter    Lars Kotthoff
Marius Lindauer    Joaquin Vanschoren

# Covariance function of a GP I

The marginalization property of the Gaussian process implies that for any finite set of input values, the corresponding vector of function values is Gaussian:

$$\boldsymbol{f} = \left[f\left(\mathbf{x}^{(1)}\right), \ldots, f\left(\mathbf{x}^{(n)}\right)\right] \sim \mathcal{N}\left(\boldsymbol{m}, \boldsymbol{K}\right).$$

- The covariance matrix $\boldsymbol{K}$ is constructed according to the chosen inputs $\left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right\}$.
- Each entry $\boldsymbol{K}_{ij}$ is computed by $k\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right)$.
- Technically, to be a valid covariance matrix, $\boldsymbol{K}$ needs to be positive semi-definite for **every** choice of inputs $\left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right\}$.
- A function $k(\cdot, \cdot)$ that satisfies this condition is called **positive definite**.

# Covariance function of a GP II

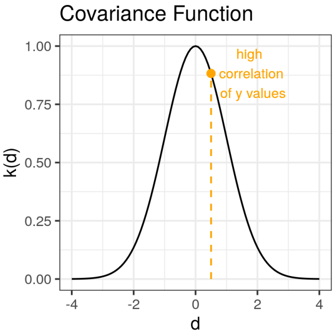
- Recall that the purpose of the covariance function is to control to which degree the following condition is fulfilled:

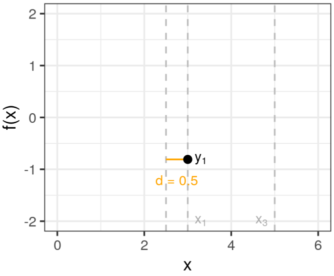
  *If* $\mathbf{x}^{(i)}$ *and* $\mathbf{x}^{(j)}$ *are close in the* $\mathcal{X}$*-space, their function values* $f(\mathbf{x}^{(i)})$ *and* $f(\mathbf{x}^{(j)})$ *should be close in* $\mathcal{Y}$*-space.*

Closeness of $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ in the input space $\mathcal{X}$ is measured by $\boldsymbol{d} = \mathbf{x}^{(i)} - \mathbf{x}^{(j)}$.

# Covariance function of a GP: Example I

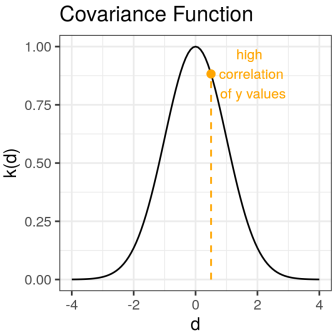
- Let $f(\mathbf{x})$ be a GP with $k(\mathbf{x}, \mathbf{x}') = \exp(-\frac{1}{2}\|\boldsymbol{d}\|^2)$ where $\boldsymbol{d} = \mathbf{x} - \mathbf{x}'$.
- Consider two points $\mathbf{x}^{(1)} = 3$ and $\mathbf{x}^{(2)} = 2.5$. To investigate how correlated their function values are, compute their correlation!

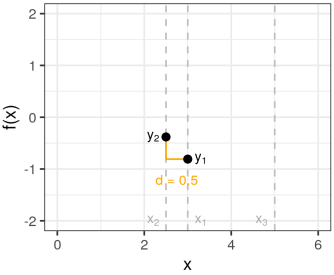


# Covariance function of a GP: Example II

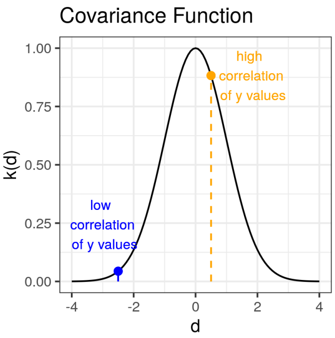
- Assume that we observe a value of $y^{(1)} = -0.8$. Under the said assumption for the Gaussian process, the value of $y^{(2)}$ should be close to $y^{(1)}$.

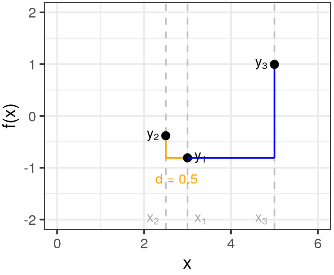


# Covariance function of a GP: Example III

- Now, let us take a new point $\mathbf{x}^{(3)}$ which is not too close to $\mathbf{x}^{(1)}$.
- Their function values should not be so correlated. That is, $y^{(1)}$ and $y^{(3)}$ are probably far away from each other.



# Covariance Functions

Three types of properties are commonly used in covariance functions:

- $k$ is **stationary** if it depends only on $\boldsymbol{d} = \mathbf{x} - \mathbf{x}'$ and is denoted by $k(\boldsymbol{d})$.
- $k$ is **isotropic** if it depends only on $r = \|\mathbf{x} - \mathbf{x}'\|$ and is denoted by $k(r)$.
- $k$ is a **dot product** if it depends only on $\mathbf{x}^T\mathbf{x}'$.

- Isotropy implies stationarity.
- Isotropic functions are rotationally invariant.
- Stationary functions are translationally invariant:

$$k(\mathbf{x}, \mathbf{x} + \boldsymbol{d}) = k(\mathbf{0}, \boldsymbol{d}) = k(\boldsymbol{d})$$

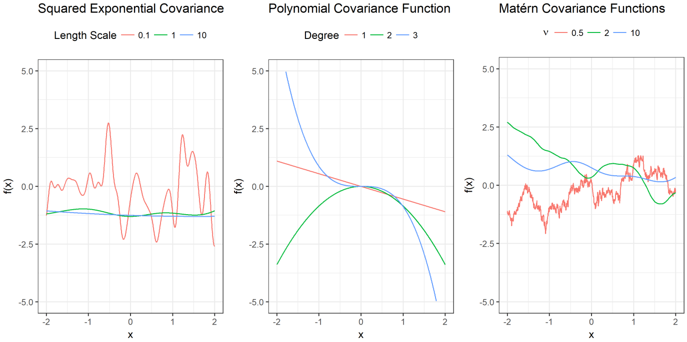
# Commonly Used Covariance Functions I

| Name | $k(\mathbf{x}, \mathbf{x}')$ |
|---|---|
| constant | $\sigma_0^2$ |
| linear | $\sigma_0^2 + \mathbf{x}^T \mathbf{x}'$ |
| polynomial | $(\sigma_0^2 + \mathbf{x}^T \mathbf{x}')^p$ |
| squared exponential | $\exp(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\ell^2})$ |
| Matérn | $\frac{1}{2^\nu \Gamma(\nu)} \left( \frac{\sqrt{2\nu}}{\ell} \|\mathbf{x} - \mathbf{x}'\| \right)^\nu K_\nu \left( \frac{\sqrt{2\nu}}{\ell} \|\mathbf{x} - \mathbf{x}'\| \right)$ |
| exponential | $\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|}{\ell}\right)$ |

$K_\nu(\cdot)$ is the modified Bessel function of the second kind.

# Commonly Used Covariance Functions II

- Some random functions drawn from Gaussian processes with a Squared Exponential Kernel (left), Polynomial Kernel (middle), and a Matérn Kernel (right, $\ell = 1$).
- The length-scale hyperparameter determines the "wiggliness" of the function.
- For Matérn, the $\nu$ parameter determines how differentiable the process is.



# Squared Exponential Covariance Function

The squared exponential function is one of the most commonly used covariance functions.

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

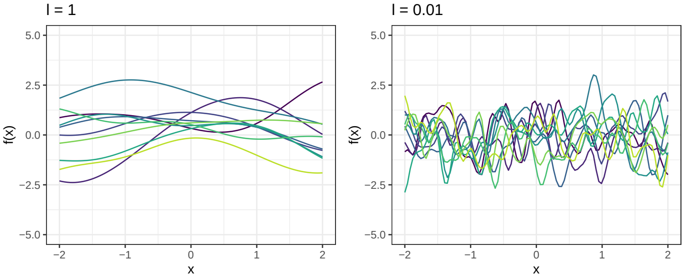
**Properties**:

- It depends merely on the distance $r = \|\mathbf{x} - \mathbf{x}'\| \rightarrow$ isotropic and stationary.
- Infinitely differentiable $\rightarrow$ the corresponding GP is too smooth.
- It utilizes strong smoothness assumptions $\rightarrow$ unrealistic for modeling most of the physical processes.

# Characteristic Length-Scale I

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2\ell^2}\|\mathbf{x} - \mathbf{x}'\|^2\right)$$

$\ell$ is called **characteristic length-scale**. Loosely speaking, the characteristic length-scale describes how far you need to move in input space for the function values to become uncorrelated. Higher $\ell$ induces smoother functions, lower $\ell$ induces more wiggly functions.



# Characteristic Length-Scale II

For more than $p = 2$ dimensions, the squared exponential can be parameterized as follows:

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \sigma_f^2 \exp\left(-\frac{1}{2}\left(\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right)^\top \boldsymbol{M}\left(\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right)\right)$$

Possible choices for the matrix $\boldsymbol{M}$ include

$$\boldsymbol{M}_1 = \ell^{-2}\boldsymbol{I} \qquad \boldsymbol{M}_2 = \mathsf{diag}(\boldsymbol{\ell})^{-2} \qquad \boldsymbol{M}_3 = \Gamma\Gamma^\top + \mathsf{diag}(\boldsymbol{\ell})^{-2}$$

where $\boldsymbol{\ell}$ is a $p$-vector of positive values and $\Gamma$ is a $p \times k$ matrix.

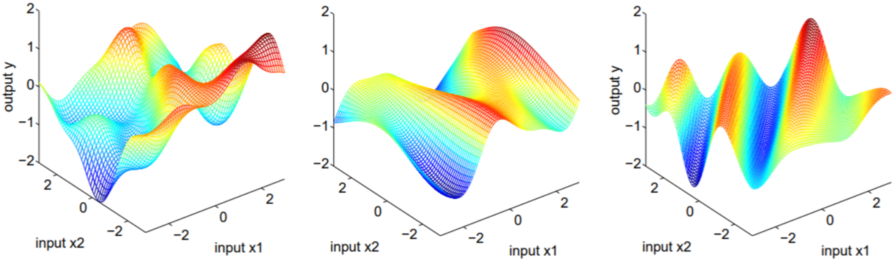
Here again, $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_p)$ are characteristic length-scales for each dimension.

# Characteristic Length-Scale III

What is the benefit of having an individual hyperparameter $\ell_i$ for each dimension?

- The $\ell_1, \dots, \ell_p$ hyperparameters play the role of **characteristic length-scales**.
- Loosely speaking, $\ell_i$ describes how far you need to move along axis $i$ in input space for the function values to be uncorrelated.
- Such a covariance function implements **automatic relevance determination** (ARD), since the inverse of the length-scale $\ell_i$ determines the relevancy of input feature $i$ to the regression.
- If $\ell_i$ is very large, the covariance will become almost independent of that input, effectively removing it from inference.
- If the features are on different scales, the data can be automatically **rescaled** by estimating $\ell_1, \dots, \ell_p$.

# Characteristic Length-Scale IV




For the first plot, we have chosen $\boldsymbol{M} = \boldsymbol{I}$: the function varies the same in all directions. The second plot is for $\boldsymbol{M} = \text{diag}(\boldsymbol{\ell})^{-2}$ and $\boldsymbol{\ell} = (1, 3)$: The function varies less rapidly as a function of $x_2$ than $x_1$ as the length-scale for $x_1$ is less. In the third plot $\boldsymbol{M} = \Gamma\Gamma^T + \text{diag}(\boldsymbol{\ell})^{-2}$ for $\Gamma = (1, -1)^\top$ and $\boldsymbol{\ell} = (6, 6)^\top$. Here $\Gamma$ gives the direction of the most rapid variation.

[Rasmussen and Williams. 2006 ]

# AutoML: Gaussian Processes

## Gaussian Process Prediction

Bernd Bischl  Frank Hutter  Lars Kotthoff
Marius Lindauer  Joaquin Vanschoren

# Motivation

- So far, we have learned how to **sample** from a Gaussian process prior.
- However, most of the time, we are not interested in drawing random functions from the prior. Instead, we usually like to use the knowledge provided by the training data to predict values of $f$ at a new test point $\mathbf{x}_*$.
- In what follows, we will investigate how to update the Gaussian process prior ($\rightarrow$ posterior process) and how to make predictions.

# Gaussian Posterior Process and Prediction

# Posterior Process I

- Let us distinguish between **observed training** inputs (also denoted by a design matrix $\mathbf{X}$), their corresponding values

$$\boldsymbol{f} = \left[ f\left(\mathbf{x}^{(1)}\right), \ldots, f\left(\mathbf{x}^{(n)}\right) \right],$$

  and one single **unobserved test point** $\mathbf{x}_*$ with $f_* = f\left(\mathbf{x}_*\right).$
- We now want to infer the distribution of $f_* | x_*, X, \boldsymbol{f}$.
- Assuming a zero-mean GP prior $\mathcal{G}\left(\mathbf{0}, k(\mathbf{x}, \mathbf{x}')\right)$, we can assert that

$$\begin{bmatrix} \boldsymbol{f} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{K} & \boldsymbol{k}_* \\ \boldsymbol{k}_*^T & \boldsymbol{k}_{**} \end{bmatrix}\right),$$

  where, $\boldsymbol{K} = \left(k\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right)\right)_{i,j}$, $\boldsymbol{k}_* = \left[k\left(\mathbf{x}_*, \mathbf{x}^{(1)}\right), \ldots, k\left(\mathbf{x}_*, \mathbf{x}^{(n)}\right)\right]$ and $\boldsymbol{k}_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$.

# Posterior Process II

(∗) *A General Rule of Conditioning for Gaussian Random Variables*

If the $m$-dimensional Gaussian vector $\boldsymbol{z} \sim \mathcal{N}(\mu, \Sigma)$ can be partitioned with $\boldsymbol{z} = (\boldsymbol{z}_1, \boldsymbol{z}_2)$ where $\boldsymbol{z}_1$ is $m_1$-dimensional and $\boldsymbol{z}_2$ is $m_2$-dimensional, and:

$$(\mu_1, \mu_2)\,, \quad \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix},$$

then the conditional distribution $\boldsymbol{a} = \boldsymbol{z}_2 \mid \boldsymbol{z}_1$ will be a multivariate normal distribution:

$$\mathcal{N}\left(\mu_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{a} - \mu_1)\,, \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)$$

# Posterior Process III

- Given that $\boldsymbol{f}$ is observed, we can exploit the general rule $(*)$ to obtain the following formula:

$$f_* \mid \mathbf{x}_*, \mathbf{X}, \boldsymbol{f} \sim \mathcal{N}(\boldsymbol{k}_*^T \boldsymbol{K}^{-1} \boldsymbol{f}, \boldsymbol{k}_{**} - \boldsymbol{k}_*^T \boldsymbol{K}^{-1} \boldsymbol{k}_*).$$

- As the posterior is Gaussian, the maximum a-posteriori estimate (i.e., the mode of the posterior distribution) is:

$$\boldsymbol{k}_*^T \boldsymbol{K}^{-1} \boldsymbol{f}.$$

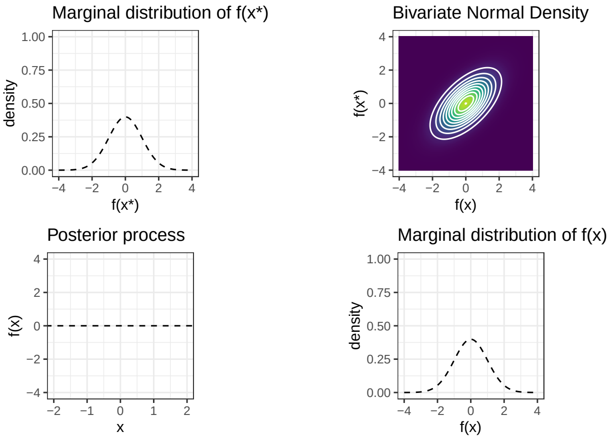
# GP Prediction: Two Points I

To visualize the above idea, assume that we have observed a single training point $\mathbf{x} = -0.5$. Based on this point, we intend to make a prediction at the test point $\mathbf{x}_* = 0.5$.

- Under a zero-mean $\mathcal{G}$ with $k(\mathbf{x}, \mathbf{x}') = \exp(-\frac{1}{2}\|\mathbf{x} - \mathbf{x}'\|^2)$, we compute the cov-matrix:
$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} 1 & 0.61 \\ 0.61 & 1 \end{bmatrix}\right).$$
- Let us assume that we observe the point $f(\mathbf{x}) = 1$. We can compute the posterior distribution:
$$\begin{aligned} f_* \mid \mathbf{x}_*, \mathbf{x}, f \quad &\sim \quad \mathcal{N}(\boldsymbol{k}_*^T \boldsymbol{K}^{-1} f, k_{**} - \boldsymbol{k}_*^T \boldsymbol{K}^{-1} \boldsymbol{k}_*) \\ &\sim \quad \mathcal{N}(0.61 \cdot 1 \cdot 1, 1 - 0.61 \cdot 1 \cdot 0.61) \\ &\sim \quad \mathcal{N}\left(0.61, 0.6279\right) \end{aligned}$$
- The MAP-estimate for $\mathbf{x}_*$ is $f(\mathbf{x}_*) = 0.61$, and the uncertainty estimate is $0.6279$.

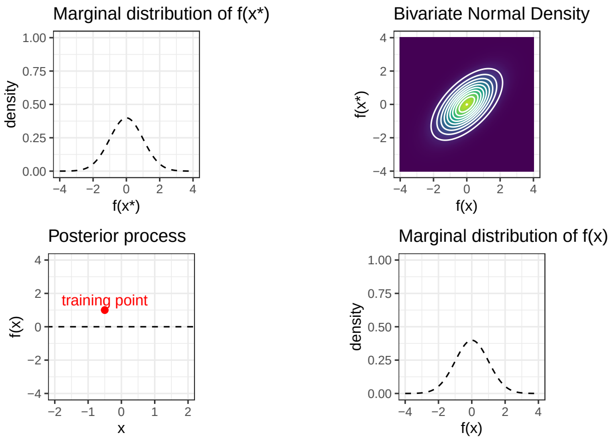
# GP Prediction: Two Points II

The figures show the bivariate normal density as well as the corresponding marginals.



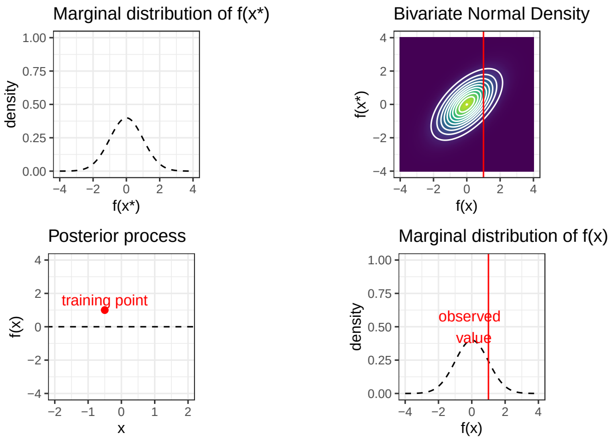
# GP Prediction: Two Points III

We observe $f(\mathbf{x}) = 1$ at training point $\mathbf{x} = -0.5$.



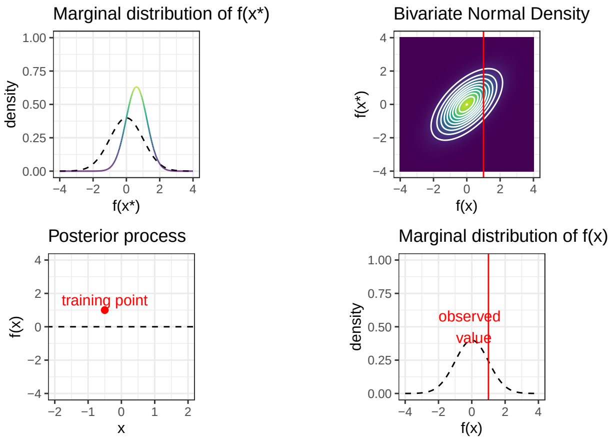
# GP Prediction: Two Points IV

We condition the Gaussian on $f(\mathbf{x}) = 1$.



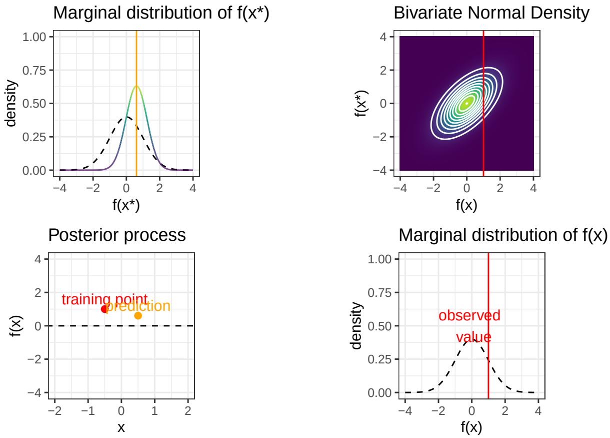
# GP Prediction: Two Points V

We then compute the posterior distribution of $f(\mathbf{x}_*)$ given that $f(\mathbf{x}) = 1$.



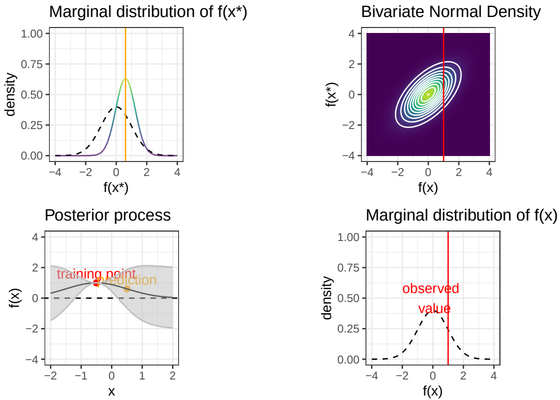
# GP Prediction: Two Points VI

A possible predictor for $f$ at $\mathbf{x}_*$ is the MAP of the posterior distribution.



# GP Prediction: Two Points VII

We can repeat this process for different $\mathbf{x}_*$ and find the respective mean (grey line) and standard deviation (grey area). Note that the grey area is mean $\pm 2\times$ standard deviation.



# Posterior Process I

- The previous discussion was restricted to a single test point. However, one can generalize it to posterior processes with multiple unobserved test points:

$$\boldsymbol{f}_* = \left[ f\left(\mathbf{x}_*^{(1)}\right), \ldots, f\left(\mathbf{x}_*^{(m)}\right) \right].$$

- Under a zero-mean Gaussian process, we have:

$$\begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{K} & \boldsymbol{K}_* \\ \boldsymbol{K}_*^T & \boldsymbol{K}_{**} \end{bmatrix}\right),$$

where $\boldsymbol{K}_* = \left(k\left(\mathbf{x}^{(i)}, \mathbf{x}_*^{(j)}\right)\right)_{i,j}$ and $\boldsymbol{K}_{**} = \left(k\left(\mathbf{x}_*^{(i)}, \mathbf{x}_*^{(j)}\right)\right)_{i,j}$

# Posterior Process II

- Similar to the single test point situation, to get the posterior distribution, we exploit the general rule of conditioning for Gaussians:

$$\boldsymbol{f}_* \mid \mathbf{X}_*, \mathbf{X}, \boldsymbol{f} \sim \mathcal{N}(\boldsymbol{K}_*^T \boldsymbol{K}^{-1} \boldsymbol{f}, \boldsymbol{K}_{**} - \boldsymbol{K}_*^T \boldsymbol{K}^{-1} \boldsymbol{K}_*).$$

- This formula enables us to talk about correlations among different test points and sample functions from the posterior process.

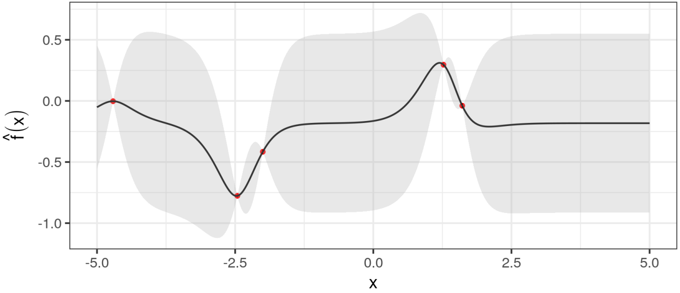
# Properties of a Gaussian Process

# GP as an Interpolator

- The "prediction" for a training point $\mathbf{x}^{(i)}$ is the exact function value $f(\mathbf{x}^{(i)})$. That is,

$$\boldsymbol{f} \mid \mathbf{X}, \boldsymbol{f} \sim \mathcal{N}(\boldsymbol{K}\boldsymbol{K}^{-1}\boldsymbol{f}, \boldsymbol{K} - \boldsymbol{K}^T\boldsymbol{K}^{-1}\boldsymbol{K}) = \mathcal{N}(\boldsymbol{f}, \mathbf{0}).$$

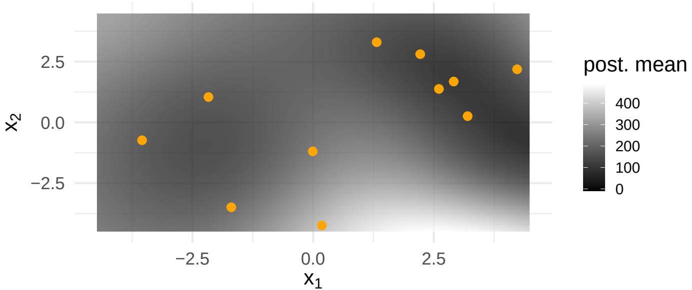
- Thus, a Gaussian process is a function **interpolator**.




After observing the training points (red), the posterior process (black) interpolates the training points.
(k(x,x') is Matèrn with nu = 2.5, the default for DiceKriging::km)

# GP as a Spatial Model I

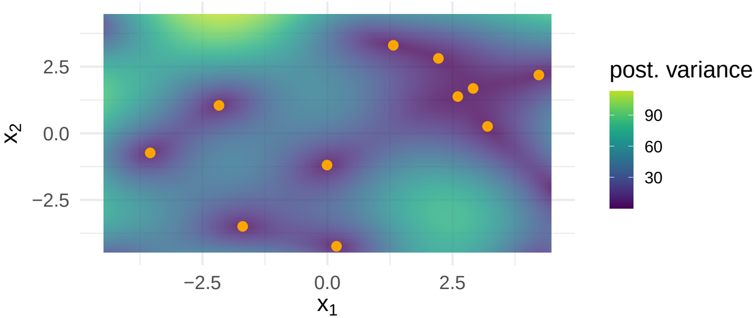
- The correlation among two outputs depends on the distance of the coresponding input points $\mathbf{x}$ and $\mathbf{x}'$. For instance, the Gaussian covariance kernel is $k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-\|\mathbf{x}-\mathbf{x}'\|^2}{2\ell^2}\right)$.
- Hence, close data points with high spatial similarity $k(\mathbf{x}, \mathbf{x}')$ enter into more strongly correlated predictions: $\boldsymbol{k}_*^\top \boldsymbol{K}^{-1} \boldsymbol{f}$ $(\boldsymbol{k}_* := \left(k(\mathbf{x}, \mathbf{x}^{(1)}), \ldots, k(\mathbf{x}, \mathbf{x}^{(n)})\right))$.




**Example:** the posterior mean of a GP that is fitted with the Gaussian covariance kernel with $\ell = 1$.

# GP as a Spatial Model II

- Posterior uncertainty increases if the new data points are far from the design points.
- The uncertainty is minimal at the design points, since the posterior variance is zero at these points.

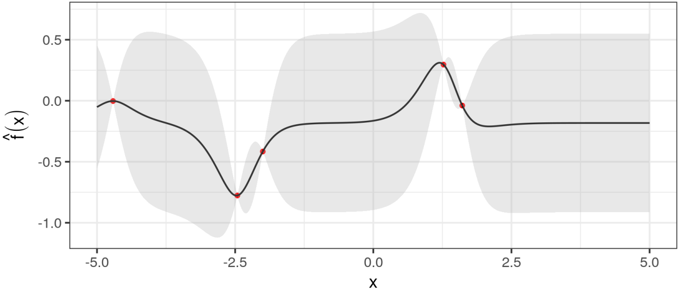



**Example (continued):** posterior variance

# Noisy Gaussian Process

# Noisy Gaussian Process I

- So far, we have implicitely assumed that we access the true function values $f(\mathbf{x})$.
- For the squared exponential kernel, for example, we had $cov\left(f(\mathbf{x}^{(i)}), f(\mathbf{x}^{(j)})\right) = 1$.
- Consequently, the posterior Gaussian process was an interpolator.




After observing the training points (red), the posterior process (black) interpolates the training points.
(k(x,x') is Matèrn with nu = 2.5, the default for DiceKriging::km)

# Noisy Gaussian Process II

- However, in reality that is not often the case. Rather, we often only have access to a noisy version of the true function values:

$$y = f(\mathbf{x}) + \epsilon, \text{ where } \epsilon \sim \mathcal{N}\left(0, \sigma^2\right).$$

- Let us assume that $f(\mathbf{x})$ is still a Gaussian process. Then, we would have the following:

$$\begin{aligned} & cov\left(y^{(i)}, y^{(j)}\right) = cov\left(f\left(\mathbf{x}^{(i)}\right) + \epsilon^{(i)}, f\left(\mathbf{x}^{(j)}\right) + \epsilon^{j}\right) \\ = \quad & cov\left(f\left(\mathbf{x}^{(i)}\right), f\left(\mathbf{x}^{(j)}\right)\right) + 2 \cdot cov\left(f\left(\mathbf{x}^{(i)}\right), \epsilon^{(j)}\right) + cov\left(\epsilon^{(i)}, \epsilon^{(j)}\right) \\ = \quad & k\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) + \sigma^2 \delta_{ij}. \end{aligned}$$

- $\sigma^2$ is called **nugget**.

# Noisy Gaussian Process III

- We can now derive the predictive distribution for the case of noisy observations.
- Assuming that $f$ is modeled by a Gaussian process, the prior distribution of $y$ is

$$\boldsymbol{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{pmatrix} \sim \mathcal{N}\left(\boldsymbol{m}, \boldsymbol{K} + \sigma^2 I_n\right),$$

with

$$\mathbf{m} \quad := \quad \left(m\left(\mathbf{x}^{(i)}\right)\right)_i, \quad \mathbf{K} := \left(k\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right)\right)_{i,j}.$$

# Noisy Gaussian Process IV

We distinguish again between:

- Observed training points and their corresponding values, i.e, $\mathbf{X}$ and $\boldsymbol{y}$.
- Unobserved test points and their corresponding values, i.e, $\mathbf{X}_*$ and $\boldsymbol{f}_*$.

and get:

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{K} + \sigma^2 I_n & \boldsymbol{K}_* \\ \boldsymbol{K}_*^T & \boldsymbol{K}_{**} \end{bmatrix}\right).$$

# Noisy Gaussian Process V

- Similar to the noise-free case, we condition according to the rule of conditioning for Gaussians to get the posterior distribution for the test outputs $\boldsymbol{f}_*$ at $\mathbf{X}_*$:

$$\boldsymbol{f}_* \mid \mathbf{X}_*, \mathbf{X}, \boldsymbol{y} \sim \mathcal{N}(\boldsymbol{m}_{\text{post}}, \boldsymbol{K}_{\text{post}}),$$

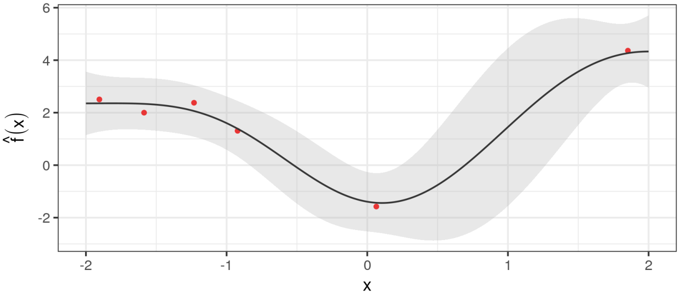
  with

$$\begin{aligned} \boldsymbol{m}_{\text{post}} &= \boldsymbol{K}_*^T \left(\boldsymbol{K} + \sigma^2 \cdot \boldsymbol{I}\right)^{-1} \boldsymbol{y} \\ \boldsymbol{K}_{\text{post}} &= \boldsymbol{K}_{**} - \boldsymbol{K}_*^T \left(\boldsymbol{K}^{-1} + \sigma^2 \cdot \boldsymbol{I}\right) \boldsymbol{K}_*. \end{aligned}$$

- This converts back to the noise-free formula if $\sigma^2 = 0$.

# Noisy Gaussian Process VI

- The noisy Gaussian process is not an interpolator any more.
- A larger nugget term leads to a wider "band" around the observed training points.
- The nugget term is estimated during training.




After observing the training points (red), we have a nugget-band around the oberved points.
(k(x,x') is the squared exponential)

# AutoML: Gaussian Processes

## Gaussian Process Training

Bernd Bischl Frank Hutter Lars Kotthoff
Marius Lindauer Joaquin Vanschoren

# Training of a Gaussian Process

- To make predictions for a regression task by a Gaussian process, one simply needs to perform matrix computations.
- But for this to work out, we assume that the covariance functions is fully given, including all of its hyperparameters.
- A very nice property of GPs is that we can learn the numerical hyperparameters of a selected covariance function directly during GP training.

# Training a GP via the Maximum Likelihood I

- Let us assume $y = f(\mathbf{x}) + \epsilon,\ \epsilon \sim \mathcal{N}\left(0, \sigma^2\right)$, where $f(\mathbf{x}) \sim \mathcal{G}\left(\mathbf{0}, k\left(\mathbf{x}, \mathbf{x}' \mid \boldsymbol{\theta}\right)\right)$.

- Noticing that $\boldsymbol{y} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{K} + \sigma^2 \boldsymbol{I}\right)$, we can find the marginal log-likelihood (or evidence):

$$
\begin{aligned}
log\, p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\theta}) &= log\left[(2\pi)^{-n/2} |\boldsymbol{K}_y|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_y^{-1} \boldsymbol{y}\right)\right] \\
&= -\frac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_y^{-1} \boldsymbol{y} - \frac{1}{2} log\, |\boldsymbol{K}_y| - \frac{n}{2} log\, 2\pi.
\end{aligned}
$$

with $\boldsymbol{K}_y := \boldsymbol{K} + \sigma^2 \boldsymbol{I}$ and $\boldsymbol{\theta}$ denoting the parameters of the covariance function (i.e., the hyperparameters).

# Training a GP via the Maximum Likelihood II

Recalling that the increase of the length-scale reduces the model flexibility, the three terms of the marginal likelihood can be interpreted as follows.

- The data fit $-\frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}_y^{-1}\boldsymbol{y}$. The data fit tends to decrease by increasing the length-scale.
- The complexity penalty $-\frac{1}{2}\, log\, |\boldsymbol{K}_y|$, which depends on the covariance function. This term decreases with the increase of the length-scale (the model gets less complex as the length-scale grows).
- The normalization constant $-\frac{n}{2}\, log\, 2\pi$.

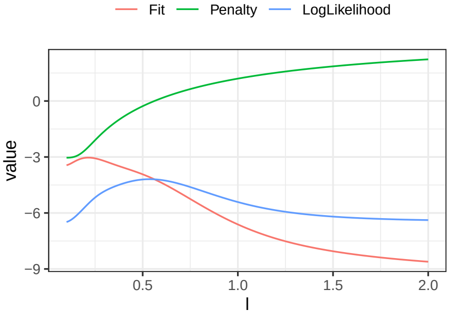
# Training a GP: Example I

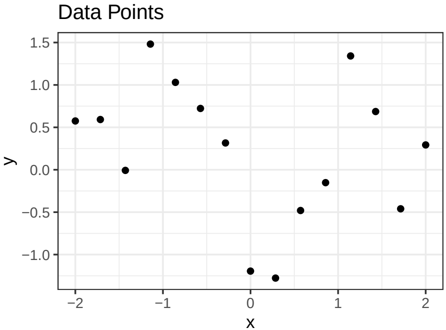
To visualize this, let us consider a zero-mean GP with a squared exponential kernel:

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2\ell^2}\|\mathbf{x} - \mathbf{x}'\|^2\right).$$

- Recall that the model becomes smoother and less complex as the length-scale $\ell$ increases.
- We will show how each of the following terms behaves if the value of $\ell$ increases:
  - the data fit $-\frac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_y^{-1}\boldsymbol{y}$,
  - the complexity penalty $-\frac{1}{2}\, log\, |\boldsymbol{K}_y|$,
  - the overall value of the marginal likelihood $log\, p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\theta})$.

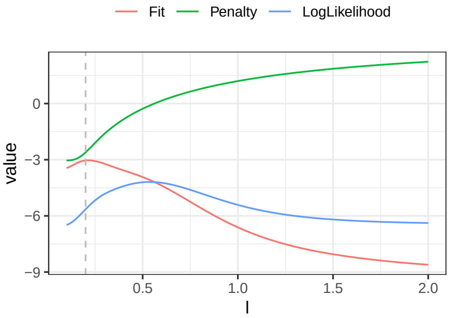
# Training a GP: Example II

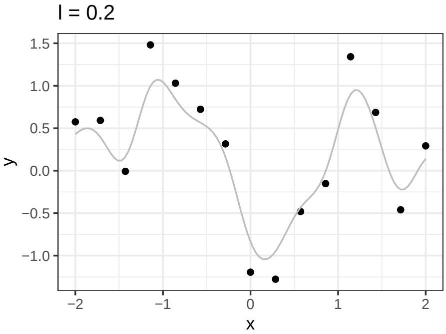



ℹ The left plot depicts how the data fit, the complexity penalty (a higher value means less penalization), and the overall marginal likelihood behave for increasing values of the length-scale.

# Training a GP: Example III

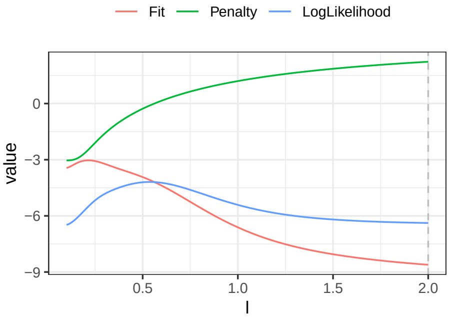


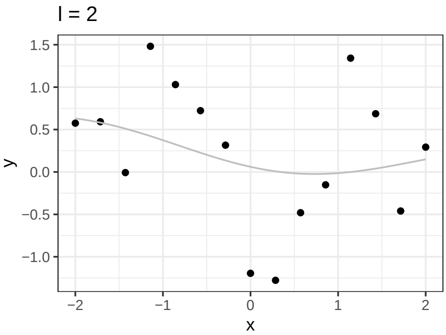




The left plot depicts how the data fit, the complexity penalty (a higher value means less penalization), and the overall marginal likelihood behave for increasing values of the length-scale.

A small $\ell$ leads to a good fit, but, to a high complexity penalty.

# Training a GP: Example IV

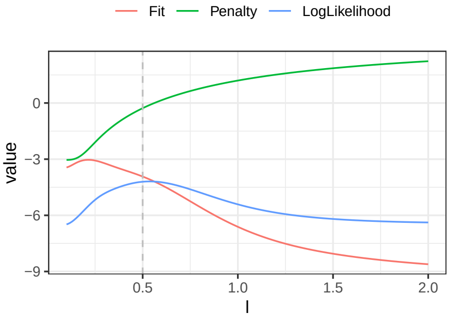


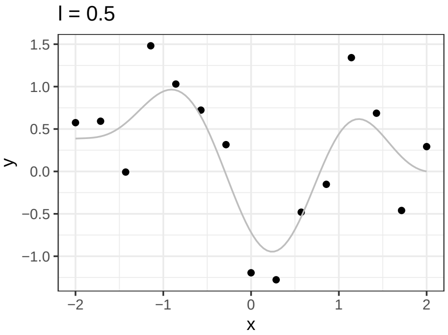




ℹ The left plot depicts how the data fit, the complexity penalty (a higher value means less penalization), and the overall marginal likelihood behave for increasing values of the length-scale.

💡 A large $\ell$ results in a poor fit.

# Training a GP: Example V




The left plot depicts how the data fit, the complexity penalty (a higher value means less penalization), and the overall marginal likelihood behave for increasing values of the length-scale.

The maximizer of the log-likelihood ($\ell = 0.5$) balances the complexity and data the fit.

# Training a GP via the Maximum Likelihood I

To choose the hyperparameters by maximizing the marginal likelihood, we need to find the partial derivatives of the likelihood w.r.t. the hyperparameters:

$$
\begin{aligned}
\frac{\partial}{\partial \theta_j} log\, p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\theta}) &= \frac{\partial}{\partial \theta_j} \left( -\frac{1}{2} \boldsymbol{y}^\top \boldsymbol{K}_y^{-1} \boldsymbol{y} - \frac{1}{2} log\, |\boldsymbol{K}_y| - \frac{n}{2} log\, 2\pi \right) \\
&= \frac{1}{2} \boldsymbol{y}^\top \boldsymbol{K}^{-1} \frac{\partial \boldsymbol{K}}{\partial \theta_j} \boldsymbol{K}^{-1} \boldsymbol{y} - \frac{1}{2} \mathrm{tr} \left( \boldsymbol{K}^{-1} \frac{\partial \boldsymbol{K}}{\partial \boldsymbol{\theta}} \right) \\
&= \frac{1}{2} \mathrm{tr} \left( (\boldsymbol{K}^{-1} \boldsymbol{y} \boldsymbol{y}^\top \boldsymbol{K}^{-1} - \boldsymbol{K}^{-1}) \frac{\partial \boldsymbol{K}}{\partial \theta_j} \right)
\end{aligned}
$$

Above, we used the following identities:

$$
\frac{\partial}{\partial \theta_j} \boldsymbol{K}^{-1} = -\boldsymbol{K}^{-1} \frac{\partial \boldsymbol{K}}{\partial \theta_j} \boldsymbol{K}^{-1} \text{ and } \frac{\partial}{\partial \boldsymbol{\theta}} log|\boldsymbol{K}| = \mathrm{tr} \left( \boldsymbol{K}^{-1} \frac{\partial \boldsymbol{K}}{\partial \boldsymbol{\theta}} \right)
$$

# Training a GP via the Maximum Likelihood II

- The complexity and the runtime of training a Gaussian process is dominated by the computational task of inverting $\boldsymbol{K}$ - or let's rather say for decomposing it.
- Standard methods require $\mathcal{O}(n^3)$ time (!) for this.
- Once $\boldsymbol{K}^{-1}$ - or rather the decomposition -is known, the computation of the partial derivatives requires only $\mathcal{O}(n^2)$ time per hyperparameter.
- Thus, the computational overhead of computing derivatives is small, and using a gradient based optimizer is advantageous.

# Training a GP via the Maximum Likelihood III

Workarounds to make GP estimation feasible for big data include:

- Using kernels that yield sparse $\boldsymbol{K}$: cheaper to invert.
- Subsampling the data to estimate $\theta$; $\mathcal{O}(m^3)$ for subset of size $m$.
- Combining estimates on different subsets of size $m$: **Bayesian committee**; $\mathcal{O}(nm^2)$.
- Exploiting low-rank approximations of $\boldsymbol{K}$ by using only a representative subset (inducing points) of $m$ training data $\boldsymbol{X}_m$:**Nyström approximation** $\boldsymbol{K} \approx \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-}\boldsymbol{K}_{mn}$, with $\mathcal{O}(nmk + m^3)$ for a rank-k-approximate inverse of $\boldsymbol{K}_{mm}$.
- Utilizing structure in $\boldsymbol{K}$ induced by the kernel: exact solutions but complicated maths, not applicable for all kernels.

... this is still an active area of research.

# AutoML: Gaussian Processes

## Mean Functions for Gaussian Processes

Bernd Bischl Frank Hutter Lars Kotthoff
Marius Lindauer Joaquin Vanschoren

# The Role of Mean Functions I

- It is common but by no means necessary to consider GPs with zero-mean functions:

$$m(\mathbf{x}) \equiv 0$$

- This is not necessarily a drastic limitation, since the mean of the posterior process is not confined to be zero:

$$\boldsymbol{f}_* | \mathbf{X}_*, \mathbf{X}, \boldsymbol{f} \sim \mathcal{N}(\mathbf{K}_*^\top \mathbf{K}^{-1} \boldsymbol{f}, \mathbf{K}_{**} - \mathbf{K}_*^\top \mathbf{K}^{-1} \mathbf{K}_*).$$

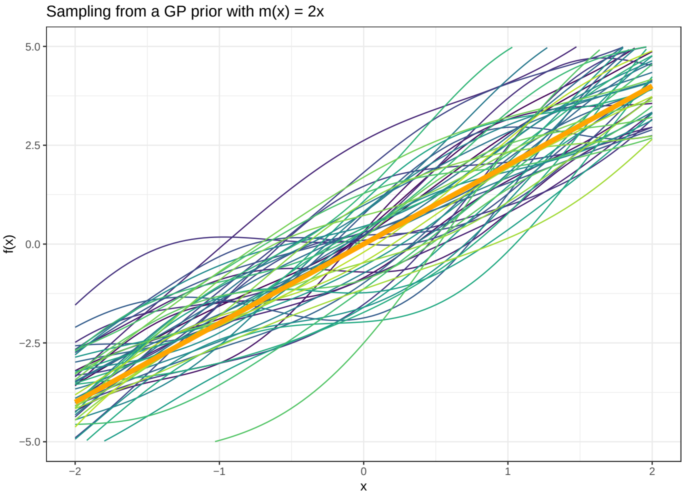
- There are several reasons why one might wish to explicitly model a mean function, including interpretability, convenience of expressing prior informations, etc.

# The Role of Mean Functions II

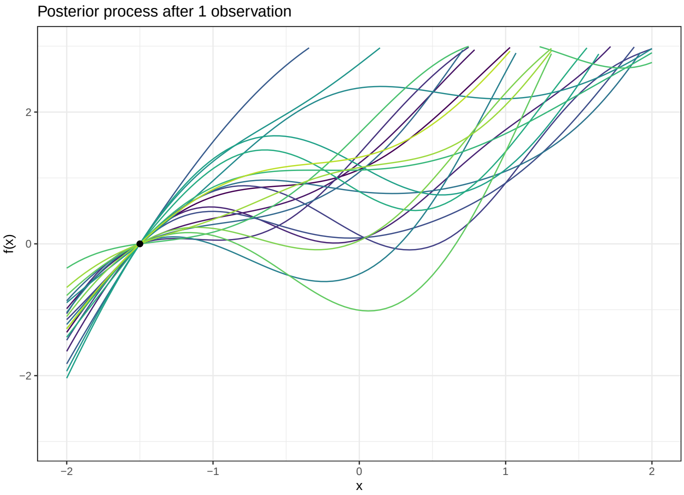
- When assuming a non-zero mean GP prior $\mathcal{G}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')$ with mean $m(\mathbf{x})$, the predictive mean becomes:
$$m(\mathbf{X}_*) + \mathbf{K}_*\mathbf{K}_y^{-1}\left(\boldsymbol{y} - m(\mathbf{X})\right)$$
but the predictive variance remains unchanged.

- Gaussian processes with non-zero mean Gaussian process priors are also called **Gaussian processes with trend**.

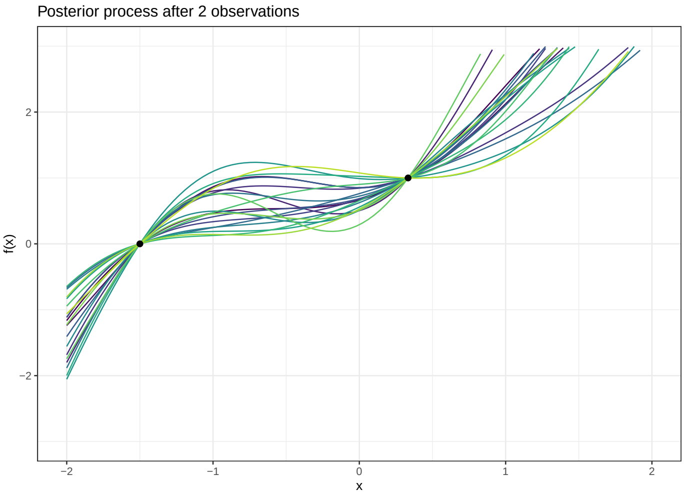
# The Role of Mean Functions III



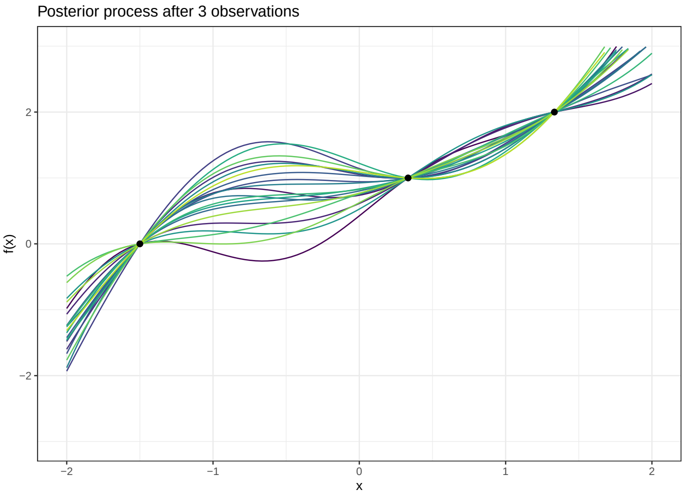
# The Role of Mean Functions IV



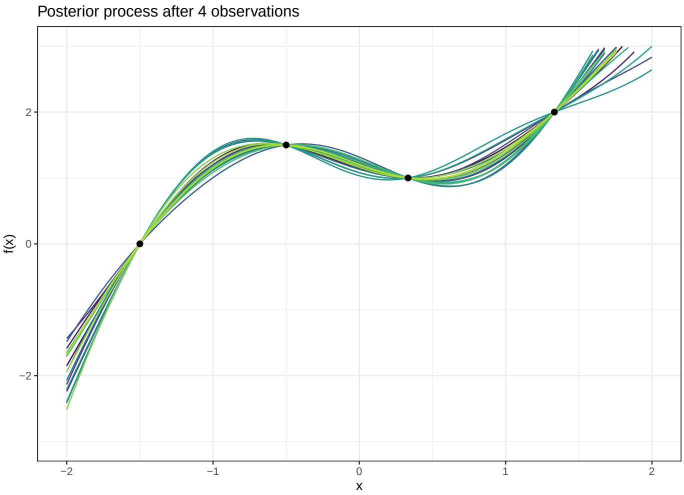
# The Role of Mean Functions V



# The Role of Mean Functions VI



# The Role of Mean Functions VII



# The Role of Mean Functions VIII

- In practice it is often difficult to specify a fixed mean function. Instead, it may be more convenient to specify a few fixed basis functions, whose coefficients, $\boldsymbol{\beta}$, are to be inferred from the data.

- Consider
$$g(\mathbf{x}) = f(\mathbf{x}) + \boldsymbol{h}(\mathbf{x})^\top \boldsymbol{\beta}, \text{ with } f(\mathbf{x}) \sim \mathcal{G}\left(0, k(\mathbf{x}, \mathbf{x}')\right)$$
where $f(\mathbf{x})$ is a zero mean GP, $\boldsymbol{h}(\mathbf{x})$ are a set of fixed basis functions, and $\boldsymbol{\beta}$ are additional parameters.

- This formulation expresses that the data is close to a global linear model with the residuals being modelled by a GP.

- By taking the prior on $\boldsymbol{\beta}$ to be Gaussian, $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{b}, B)$, another GP with an added contribution in the covariance function can be obtained:
$$g(\mathbf{x}) \sim \mathcal{G}\left(\mathbf{h}(\mathbf{x}^\top \mathbf{b}, k\left(\mathbf{x}, \mathbf{x}'\right) + \mathbf{h}(\mathbf{x})^\top B \ \mathbf{h}\left(\mathbf{x}'\right)\right)$$

# The Role of Mean Functions IX

- The mean and covariance functions of $g(\mathbf{x})$ will be then:

$$\overline{\mathbf{g}}\left(X_*\right) = \overline{\mathbf{f}}\left(X_*\right) + R^\top \overline{\boldsymbol{\beta}},$$
$$\operatorname{cov}\left(\mathbf{g}_*\right) = \operatorname{cov}\left(\mathbf{f}_*\right) + R^\top (B^{-1} + H K_y^{-1} H^\top)^{-1} R,$$

  where $\overline{\boldsymbol{\beta}} = (B^{-1} + H K_y^{-1} H^\top)^{-1} (H K_y^{-1} \mathbf{y} + B^{-1}\mathbf{b})$ and $R = H_* - H K_y^{-1} K_*$.

- Notice that $\overline{\boldsymbol{\beta}}$ is the mean of the global linear model parameters and the $H$ matrix collects the $\mathbf{h}(\mathbf{x})$ vectors for all training and test cases.

- Hence, the predictive mean is the sum of the mean linear output and what the GP model predicts from the residuals. The covariance is the usual covariance term plus a non negative contribution.

# The Role of Mean Functions X

- In the limiting case where the prior on the $\boldsymbol{\beta}$ becomes vague $(B^{-1} \to O)$, the predictive distribution becomes independent of $\boldsymbol{b}$:

$$\overline{\mathbf{g}}\left(X_*\right) = \overline{\mathbf{f}}\left(X_*\right) + R^\top \overline{\boldsymbol{\beta}},$$
$$\mathrm{cov}\left(\mathbf{g}_*\right) = \mathrm{cov}(\mathbf{f}_*) + R^\top (H K_y^{-1} H^\top)^{-1} R,$$

  where the limiting $\overline{\boldsymbol{\beta}} = (H K_y^{-1} H^\top)^{-1} H K_y^{-1} \mathbf{y}$.

- To make predictions under the limit $B^{-1} \to O$, it is important to use the above equation. Otherwise, by naïvely plugging the modified covariance function into the standard prediction equations, the entries of the covariance function tend to infinity making it unsuitable for numerical implementation.

# The Role of Mean Functions XI

- The marginal likelihood for the model with a Gaussian prior $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{b}, B)$ can be expressed as what follows (the explicit mean has been included).

$$\log p(\mathbf{y}|X, \mathbf{b}, B) = -\frac{1}{2}\left(H^\top \mathbf{b} - \mathbf{y}\right)^\top \left(K_y + H^\top B H\right)^{-1} \left(H^\top \mathbf{b} - \mathbf{y}\right) - \frac{1}{2}\log\left|K_y + H^\top B H\right| - \frac{n}{2}\log 2\pi.$$

- We are interested in exploring the limit where $B^{-1} \to O$, i.e. when the prior is vague. In this limit the mean of the prior is irrelevant, so without loss of generality we assume for now that the mean is zero, $\mathbf{b} = \mathbf{0}$.

# The Role of Mean Functions XII

- This assumption gives:

$$\log p(\mathbf{y}|X, \mathbf{b}=\mathbf{0}, B) = -\frac{1}{2}\mathbf{y}^\top K_y^{-1}\mathbf{y} + \frac{1}{2}\mathbf{y}^\top C\mathbf{y} - \frac{1}{2}\log|K_y| - \frac{1}{2}\log|B| - \frac{1}{2}\log|A| - \frac{n}{2}\log 2\pi,$$

  where $A = B^{-1} + HK_y^{-1}H^\top$ and $C = K_y^{-1}H^\top A^{-1}HK_y^{-1}$.

- The behaviour of the log marginal likelihood in the limiting case can be explored now. The variances of the Gaussian in the directions spanned by columns of $H^\top$ will become infinite, and it will require special treatment.

- The log marginal likelihood consists of three terms: a quadratic form in $\mathbf{y}$, a log determinant term, and a term involving $\log 2\pi$.

# The Role of Mean Functions XIII

- Performing an eigendecomposition of the covariance matrix, the contributions to quadratic form term from the infinite-variance directions will be zero. However, the log determinant term will tend to minus infinity.

- The standard solution in this case is to project $\mathbf{y}$ onto the directions orthogonal to the span of $H^\top$ and compute the marginal likelihood in this subspace.

- By taking $m$ to denote the rank of $H^\top$, we can discard the terms $-\frac{1}{2}\log|B| - \frac{m}{2}\log 2\pi$ from the previous equation to give:

$$\log p(\mathbf{y}|X) = -\frac{1}{2}\mathbf{y}^\top K_y^{-1}\mathbf{y} + \frac{1}{2}\mathbf{y}^\top C\mathbf{y} - \frac{1}{2}\log|K_y| - \frac{1}{2}\log|A| - \frac{n-m}{2}\log 2\pi,$$

where $A = HK_y^{-1}H^\top$ and $C = K_y^{-1}H^\top A^{-1}HK_y^{-1}$.

# AutoML: Gaussian Processes

## Covariance Functions for GPs - Advanced

Bernd Bischl    Frank Hutter    Lars Kotthoff
Marius Lindauer    Joaquin Vanschoren

# MS-Continuity and Differentiability I

We wish to describe a Gaussian process in terms of its smoothness. There are several notions of continuity for random variables. One is the continuity/differentiability in mean square (MS).

## Definition

A Gaussian process $f(\mathbf{x})$ is said to be **MS continuous** at $\mathbf{x}_*$, if

$$\mathbb{E}[|f(\mathbf{x}^{(k)}) - f(\mathbf{x}_*)|^2] \xrightarrow{k\to\infty} 0 \text{ for all converging sequences } \mathbf{x}^{(k)} \xrightarrow{k\to\infty} \mathbf{x}_*.$$

A Gaussian process $f(\mathbf{x})$ is said to be **MS differentiable** along the $i$ direction, if the following limit exists, with $\boldsymbol{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0)^\top$ being the unit vector along the $i$-th axis.

$$\lim_{h\to 0} \mathbb{E}[|\frac{f(\mathbf{x} + h\,\boldsymbol{e}_i) - f(\mathbf{x})}{h}|]$$

# MS-Continuity and Differentiability II

- The MS continuity/differentiability do not necessarily lead to the continuity/differentiability of sampled functions!
- The MS continuity/differentiability of a Gaussian process can be derived from the smoothness properties of the kernel.
- The GP is continuous in MS iff the covariance function $k(\mathbf{x}, \mathbf{x}')$ is continuous.
- The MS derivative of a Gaussian process exists iff the second derivative $\frac{\partial^2 k(\mathbf{x},\mathbf{x}')}{\partial \mathbf{x} \partial \mathbf{x}'}$ exists.

# Squared Exponential Covariance Function

The squared exponential function is one of the most commonly used covariance functions.

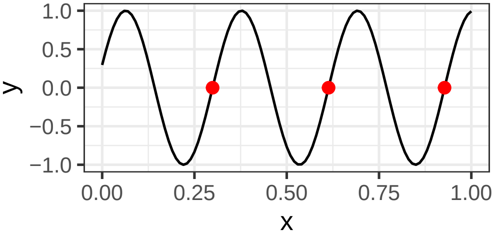
$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\ell^2}\right).$$

**Properties**:

- It depends merely on the distance $r = \|\mathbf{x} - \mathbf{x}'\| \rightarrow$ isotropic and stationary.
- Infinitely differentiable $\rightarrow$ the corresponding GP is too smooth.
- It utilizes strong smoothness assumptions $\rightarrow$ unrealistic for modeling most of the physical processes.

# Upcrossing Rate and Characteristic Length-Scale I

- Another way to describe a Gaussian process is the expected number of up-crossings at level-0 on the unit interval, which we denote by $N_0$.




- For an isotropic covariance function $k(r)$, the expected number of up-crossings can be calculated explicitly:

$$\mathbb{E}[N_0] = \frac{1}{2\pi}\sqrt{\frac{-k''(0)}{k(0)}}.$$

# Upcrossing Rate and Characteristic Length-Scale II

Example (squared exponential):

$$
\begin{aligned}
k(r) &= \exp\left(-\frac{r^2}{2\ell^2}\right) \\
k'(r) &= -k(r) \cdot \frac{r}{\ell^2} \\
k''(r) &= k(r) \cdot \frac{r^2}{\ell^4} - k(r) \cdot \frac{1}{\ell^2}
\end{aligned}
$$

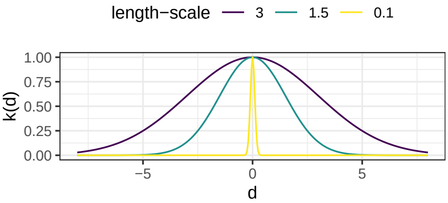
The expected number of upcrossings at level-0 is

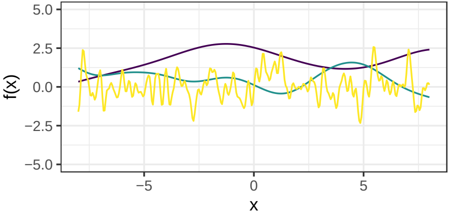
$$
\mathbb{E}[N_0] = \frac{1}{2\pi}\sqrt{\frac{-k''(0)}{k(0)}} = \frac{1}{2\pi}\sqrt{\frac{1}{\ell^2}} = (2\pi\ell)^{-1}.
$$

# Upcrossing Rate and Characteristic Length-Scale III

$\ell$ is called **characteristic length-scale**. Loosely speaking, the characteristic length-scale describes how far you need to move in input space for the function values to become uncorrelated.

- Left plot: for higher $\ell$ the correlation between function values (for unchanged distance of input points) is also higher
- Right plot: a higher $\ell$ induces a smoother function and thus fewer level-0 upcrossings



# Upcrossing Rate and Characteristic Length-Scale IV

For more than $p = 2$ dimensions, the squared exponential can be parameterized as follows:

$$k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \sigma_f^2 \exp\left(-\frac{1}{2}\left(\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right)^\top \boldsymbol{M}\left(\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right)\right)$$

Possible choices for the matrix $\boldsymbol{M}$ include

$$\boldsymbol{M}_1 = \ell^{-2}\boldsymbol{I} \qquad \boldsymbol{M}_2 = \mathrm{diag}(\boldsymbol{\ell})^{-2} \qquad \boldsymbol{M}_3 = \Gamma\Gamma^\top + \mathrm{diag}(\boldsymbol{\ell})^{-2}$$

where $\boldsymbol{\ell}$ is a $p$-vector of positive values and $\Gamma$ is a $p \times k$ matrix.

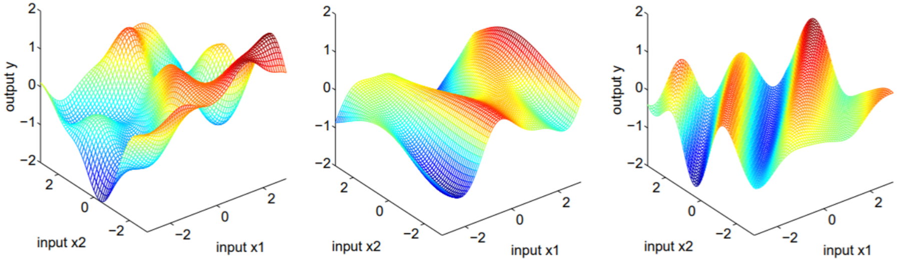
Here again, $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_p)$ are characteristic length-scales for each dimension.

# Upcrossing Rate and Characteristic Length-Scale V

What is the benefit of learning an individual hyperparameter $\ell_i$ for each dimension?

- The $\ell_1, \ldots, \ell_p$ hyperparameters play the role of **characteristic length-scales**.
- Losely speaking, $\ell_i$ describes how far you need to move along axis $i$ in input space for the function values to be uncorrelated.
- Such a covariance function implements **automatic relevance determination** (ARD), since the inverse of the length-scale $\ell_i$ determines the relevancy of input feature $i$ to the regression.
- If $\ell_i$ is very large, the covariance will become almost independent of that input, effectively removing it from inference.
- If the features are on different scales, the data can be automatically **rescaled** by estimating $\ell_1, \ldots, \ell_p$

# Upcrossing Rate and Characteristic Length-Scale VI




For the first plot, we have chosen $\boldsymbol{M} = \boldsymbol{I}$: the function varies the same in all directions. The second plot is for $\boldsymbol{M} = \text{diag}(\boldsymbol{\ell})^{-2}$ and $\boldsymbol{\ell} = (1, 3)$: The function varies less rapidly as a function of $x_2$ than $x_1$ as the length-scale for $x_1$ is less. In the third plot $\boldsymbol{M} = \Gamma\Gamma^T + \text{diag}(\boldsymbol{\ell})^{-2}$ for $\Gamma = (1, -1)^\top$ and $\boldsymbol{\ell} = (6, 6)^\top$. Here $\Gamma$ gives the direction of the most rapid variation.

[Rasmussen and Williams. 2006 ]

# AutoML: Gaussian Processes

## Gaussian Proccesses: Additional Material

Bernd Bischl    Frank Hutter    Lars Kotthoff
Marius Lindauer    Joaquin Vanschoren

# Notation

In this part,

- $(\mathbf{x}_*, y_*)$ denotes a single test observation, excluded from the training data.
- $\mathbf{X}_* \in \mathbb{R}^{n_* \times p}$ denotes a set of $n_*$ test observations.
- $\mathbf{y}_* \in \mathbb{R}^{n_* \times p}$ denotes the corresponding outcomes, excluded from the training data.

# Noisy Gaussian Processes

# Noisy Gaussian Processes

- In the previous slides, we implicitly assumed that we access the true function values $f(\mathbf{x})$. However, in many practical cases, we only have a noisy version of the values:

$$y = f(\mathbf{x}) + \epsilon.$$

- By assuming an additive i.i.d. Gaussian noise, the covariance function becomes:

$$cov\left(y^{(i)}, y^{(j)}\right) = k\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) + \sigma_n^2 \delta_{ij}, \text{ where } \delta_{ij} = 1 \text{ if } i = j.$$

- In the matrix notation, this becomes:

$$cov\left(\mathbf{y}\right) = \mathbf{K} + \sigma_n^2 \boldsymbol{I} =: \mathbf{K}_y, \text{ where } \sigma_n^2 \text{ is called } \textbf{nugget}.$$

# GP vs. Kernelized Ridge Regression

- The predictive function is then

$$\boldsymbol{f}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{y} \sim \mathcal{N}(\bar{\boldsymbol{f}}_*, \, cov\,(\bar{\boldsymbol{f}}_*)),$$

  with $\bar{\boldsymbol{f}}_* = \mathbf{K}_*^T \mathbf{K}_y^{-1} \mathbf{y}$ and $cov\,(\bar{\boldsymbol{f}}_*) = \mathbf{K}_{**} - \mathbf{K}_*^\top \mathbf{K}_y^{-1} \mathbf{K}_*$.

- The predicted mean value at the training points $\bar{\boldsymbol{f}} = \boldsymbol{K}\mathbf{K}_y^{-1}\boldsymbol{y}$ is a **linear combination** of the $\boldsymbol{y}$ values.

Predicting the posterior mean corresponds exactly to the predictions obtained by kernelized Ridge regression. However, a GP as a Bayesian model provides us with much more information (i.e., a posterior distribution), whilst the kernelized Ridge regression does not.

# Bayesian Linear Regression as a GP

# Bayesian Linear Regression as a GP

- One example for a Gaussian process is the Bayesian linear regression model, and we already discuss it.

- For $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \tau^2 \boldsymbol{I})$, the joint distribution of any set of function values is Gaussian:

$$f(\mathbf{x}^{(i)}) = \boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \epsilon.$$

- The corresponding mean function is $m(\mathbf{x}) = \mathbf{0}$, and the covariance function is

$$\begin{aligned} cov\left(f(\mathbf{x}), f(\mathbf{x}')\right) &= \mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] - \underbrace{\mathbb{E}[f(\mathbf{x})]\mathbb{E}[f(\mathbf{x}']}_{=0} \\ &= \mathbb{E}[(\boldsymbol{\theta}^\top \mathbf{x} + \epsilon)^\top (\boldsymbol{\theta}^\top \mathbf{x}' + \epsilon)] \\ &= \tau^2 \mathbf{x}^\top \mathbf{x}' + \sigma^2 =: k(\mathbf{x}, \mathbf{x}'). \end{aligned}$$

# Feature Spaces and the Kernel Trick I

- If one relaxes the linearity assumption by projecting the features into a higher dimensional feature space $\mathcal{Z}$ using a basis function $\phi : \mathcal{X} \rightarrow \mathcal{Z}$, the corresponding covariance function becomes:

$$k(\mathbf{x}, \mathbf{x}') = \tau^2 \phi(\mathbf{x})^\top \phi(\mathbf{x}') + \sigma^2.$$

- To get arbitrarily complicated functions, we would have to handle high-dimensional feature vectors $\phi(\mathbf{x})$.

- Fortunately, all we need to know is the inner product $\phi(\mathbf{x})^T \phi(\boldsymbol{x}')$. That is, the feature vector itself never occurs in calculations.

# Feature Spaces and the Kernel Trick II

If we can get the inner product directly and without calculating the infinite feature vectors, we can infer an infinitely complicated model with a finite amount of computation. This idea is known as **kernel trick**.

- A Gaussian process can then be defined by either:
  - deriving the covariance function from the inner products of the basis functions evaluations, or
  - choosing a positive definite kernel function (Mercer Kernel), which- according to Mercer's theorem - corresponds to taking the inner products in some (possibly infinite) feature space.

# Summary: Gaussian Process Regression

- The Gaussian process regression is equivalent to the **kernelized** Bayesian linear regression.

- The covariance function describes the shape of the Gaussian process. Hence, with the right choice of covariance function, remarkably flexible models can be built.

- Naive implementations of Gaussian process models scale poorly with large datasets, as
  - the kernel matrix has to be inverted / factorized, which is $\mathcal{O}(n^3)$,
  - computing the kernel matrix uses $\mathcal{O}(n^2)$ memory - running out of memory places a hard limit on the size of problems
  - generating predictions is $\mathcal{O}(n)$ for the mean, but $\mathcal{O}(n^2)$ for the variance.

(...special tricks are needed.)

# AutoML: Gaussian Processes

## Gaussian Process Classification

Bernd Bischl Frank Hutter Lars Kotthoff
Marius Lindauer Joaquin Vanschoren

# Training a GP via the Maximum Likelihood I

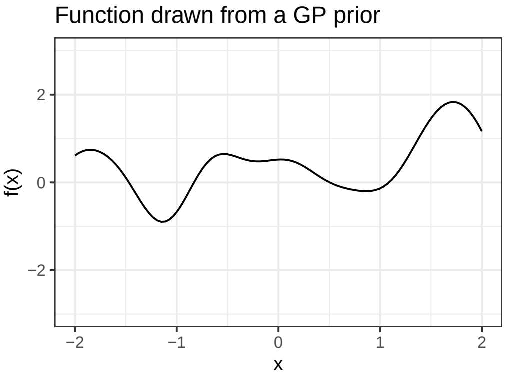
- Consider a binary classification problem, in which we want to learn $h : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{Y} = \{0, 1\}$.

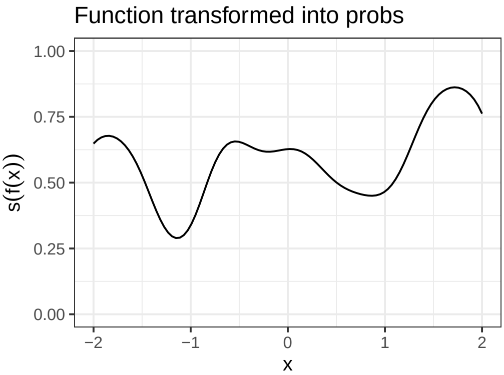
- The idea behind Gaussian process classification is straightforward: a GP prior is placed over the score function $f(\mathbf{x})$ and then transformed to a class probability via a sigmoid function $s(t)$:
$$p(y = 1 \mid f(\mathbf{x})) = s(f(\mathbf{x})).$$

- Since this is a non-Gaussian likelihood, we need to use approximate inference methods, such as Laplace approximation, expecation propagation, MCMC.

- For more details see [Rasmussen and Williams. 2006: Chapter 3]

# Training a GP via the Maximum Likelihood II



# Training a GP via the Maximum Likelihood III

- According to Bayes' rule, the posterior of the score function $\boldsymbol{f}$ takes the following form:

$$p(\boldsymbol{f} \mid \mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y} \mid \boldsymbol{f}, \mathbf{X}) \cdot p(\boldsymbol{f} \mid \mathbf{X})}{p(\mathbf{y} \mid \mathbf{X})} \propto p(\mathbf{y} \mid \boldsymbol{f}) \cdot p(\boldsymbol{f} \mid \mathbf{X}),$$

  where, the denominator is independent of $\boldsymbol{f}$ and hence has been dropped.

- From the GP assumption, we can assert that $p(\boldsymbol{f} \mid \mathbf{X}) \sim \mathcal{N}(0, \boldsymbol{K})$. Hence, we have:

$$\log p(\boldsymbol{f} \mid \mathbf{X}, y) \propto \log p(\mathbf{y} \mid \boldsymbol{f}) - \frac{1}{2}\boldsymbol{f}^\top \boldsymbol{K}^{-1} \boldsymbol{f} - \frac{1}{2} \log |\boldsymbol{K}| - \frac{n}{2} \log 2\pi.$$

# Training a GP via the Maximum Likelihood IV

- If the kernel is fixed, the last two terms will be fixed. To obtain the maximum a-posteriori estimate (MAP), we should minimize:

$$\frac{1}{2}\boldsymbol{f}^\top \boldsymbol{K}^{-1}\boldsymbol{f} - \sum_{i=1}^{n} \log p(y^{(i)} \mid f^{(i)}) + C.$$

- Note that $-\sum_{i=1}^{n} \log p(y^{(i)} \mid f^{(i)})$ is the logistic loss.

- It can be seen that the Gaussian process classification corresponds to the **kernel Bayesian logistic regression**!

# Comparison: GP vs. SVM I

- For the SVM, we have:
$$\frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{i=1}^{n} L(y^{(i)}, f(\mathbf{x}^{(i)})),$$
where $L(y, f(\mathbf{x})) = \max\{0, 1 - f(\mathbf{x}) \cdot y\}$ is the Hinge loss.

- Plugging that in, the optimization objective would be:
$$\frac{1}{2}\boldsymbol{f}^\top \boldsymbol{K}^{-1}\boldsymbol{f} + C\sum_{i=1}^{n} L(y^{(i)}, f(\mathbf{x}^{(i)})).$$

- From the representer theorem: $\boldsymbol{\theta} = \sum_{i=1}^{n} \beta_i \, y^{(i)} k\left(\mathbf{x}^{(i)}, \cdot\right)$, and thus:
$$\boldsymbol{\theta}^\top \boldsymbol{\theta} = \beta^\top \boldsymbol{K} \beta = \boldsymbol{f}^\top \boldsymbol{K}^{-1} \boldsymbol{f}$$

.

# Comparison: GP vs. SVM II

- For log-concave likelihoods $\log p(\mathbf{y} \mid \boldsymbol{f})$, there is a close correspondence between the MAP solution of the GP classifier and the SVM solution:

$$\arg\min_{f} \frac{1}{2} \boldsymbol{f}^\top \boldsymbol{K}^{-1} \boldsymbol{f} - \sum_{i=1}^{n} \log p(y^{(i)} \mid f^{(i)}) + C \quad \text{(GP classifier)}$$

$$\arg\min_{f} \frac{1}{2} \boldsymbol{f}^\top \boldsymbol{K}^{-1} \boldsymbol{f} + C \sum_{i=1}^{n} L(y^{(i)}, f(\mathbf{x}^{(i)})) \quad \text{(SVM classifier)}$$

# Comparison: GP vs. SVM III

- Both the Hinge loss and the Bernoulli loss are monotonically decreasing with increasing margin $yf(\mathbf{x})$.
- The key difference is that the Hinge loss takes on the value $0$ for $yf(\mathbf{x}) \geq 1$, while the Bernoulli loss decays slowly.
- It is this flat part of the Hinge function that gives rise to the sparsity of the SVM solution.
- The SVM classifier can be construed as a "sparse" GP classifier.