Gaussian Processes

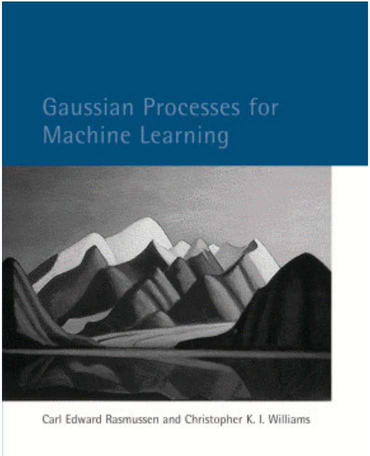
Marc Deisenroth
Centre for Artificial Intelligence
Department of Computer Science
University College London

@mpd37
m.deisenroth@ucl.ac.uk
https://deisenroth.cc

AIMS Rwanda and AIMS Ghana

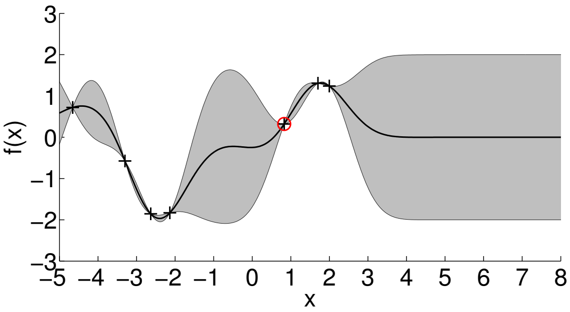
March/April 2020

Reference



`http://www.gaussianprocess.org/`

Problem Setting

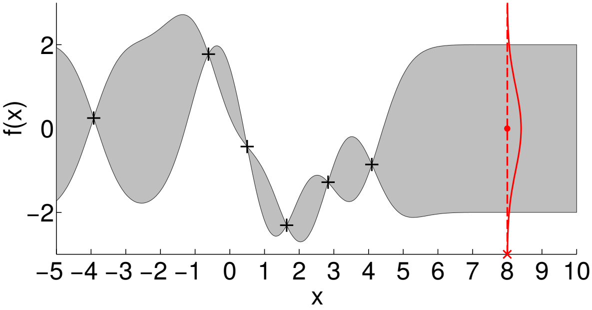


Objective

For a set of observations $y_i = f(\boldsymbol{x}_i) + \varepsilon, \quad \varepsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$, find a distribution over functions $p(f)$ that explains the data

▶▶ Probabilistic regression problem

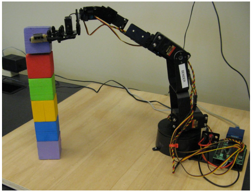
Problem Setting

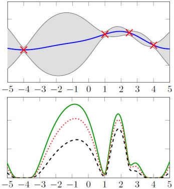


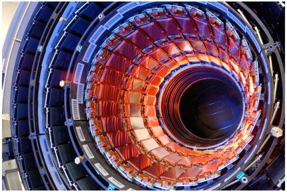
Objective

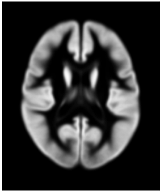
For a set of observations $y_i = f(\boldsymbol{x}_i) + \varepsilon, \quad \varepsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$, find a distribution over functions $p(f)$ that explains the data

▶▶ Probabilistic regression problem







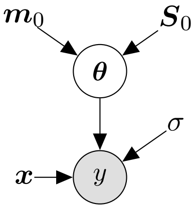


- Reinforcement learning and robotics
- Bayesian optimization (experimental design)
- Geostatistics
- Sensor networks
- Time-series modeling and forecasting
- High-energy physics
- Medical applications

Prior $p(\boldsymbol{\theta}) = \mathcal{N}\big(\boldsymbol{m}_0,\, \boldsymbol{S}_0\big)$

Likelihood $p(y|\boldsymbol{x}, \boldsymbol{\theta}) = \mathcal{N}\big(y \,|\, \boldsymbol{\phi}^\top(\boldsymbol{x})\boldsymbol{\theta},\, \sigma_n^2\big)$

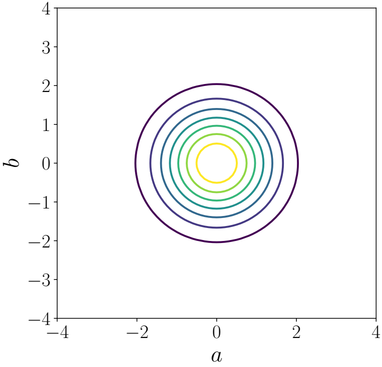
$$\implies y = \boldsymbol{\phi}^\top(\boldsymbol{x})\boldsymbol{\theta} + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$$



- Parameter $\boldsymbol{\theta}$ becomes a latent (random) variable
- Distribution $p(\boldsymbol{\theta})$ induces a distribution over plausible functions
- Choose a conjugate Gaussian prior
  - Gaussian posterior $p(\boldsymbol{\theta}|\boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}\big(\boldsymbol{\theta} \,|\, \boldsymbol{m}_N,\, \boldsymbol{S}_N\big)$
  - Closed-form computations (e.g., predictions, marginal likelihood)

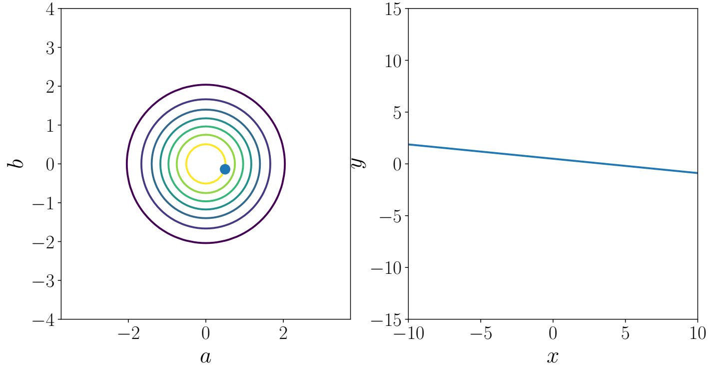
Consider a linear regression setting

$$y = a + bx + \epsilon\,, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$$
$$p(a,b) = \mathcal{N}\big(\mathbf{0},\, \boldsymbol{I}\big)$$



Consider a linear regression setting

$$y = f(x) + \epsilon = a + bx + \epsilon\,, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$$
$$p(a, b) = \mathcal{N}\big(\mathbf{0},\, \boldsymbol{I}\big)$$
$$f_i(x) = a_i + b_i x, \quad [a_i, b_i] \sim p(a, b)$$



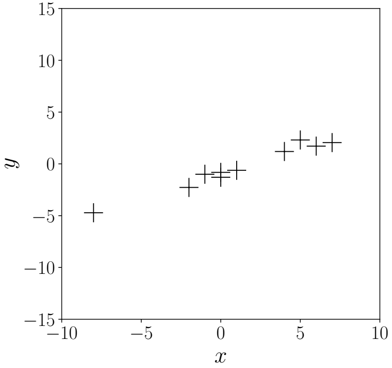
Sampling from the Posterior over Functions

Consider a linear regression setting

$$y = f(x) + \epsilon = a + bx + \epsilon\,, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$$

$$p(a,b) = \mathcal{N}\big(\mathbf{0},\, \boldsymbol{I}\big)$$

$$\boldsymbol{X} = [x_1, \ldots, x_N],\ \boldsymbol{y} = [y_1, \ldots, y_N] \quad \text{Training data}$$

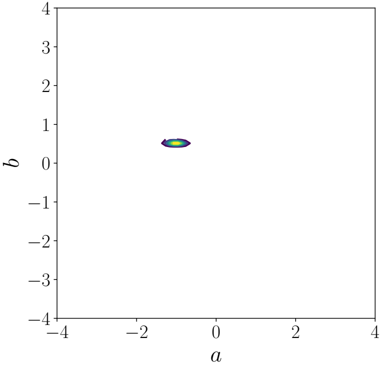


Sampling from the Posterior over Functions

Consider a linear regression setting

$$y = f(x) + \epsilon = a + bx + \epsilon\,, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$$
$$p(a, b) = \mathcal{N}\big(\mathbf{0},\, \boldsymbol{I}\big)$$
$$p(a, b|\boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}\big(\boldsymbol{m}_N,\, \boldsymbol{S}_N\big)$$

Posterior

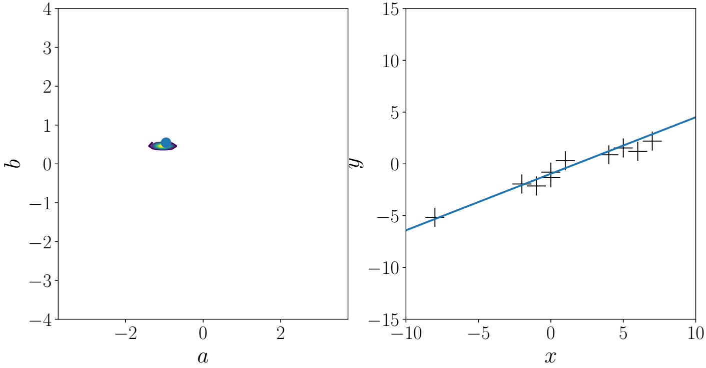


Consider a linear regression setting

$$y = f(x) + \epsilon = a + bx + \epsilon\,, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$$

$$[a_i, b_i] \sim p(a, b|\boldsymbol{X}, \boldsymbol{y})$$

$$f_i = a_i + b_i x$$



Approach

- Instead of sampling parameters, which induce a distribution over functions, sample functions directly
  - Place a prior on functions
  - Make assumptions on the distribution of functions

Approach

- Instead of sampling parameters, which induce a distribution over functions, sample functions directly
  - Place a prior on functions
  - Make assumptions on the distribution of functions
- Intuition: function = infinitely long vector of function values
  - Make assumptions on the distribution of function values

Approach

- Instead of sampling parameters, which induce a distribution over functions, sample functions directly
  - ▶▶ Place a prior on functions
  - ▶▶ Make assumptions on the distribution of functions
- Intuition: function = infinitely long vector of function values
  - ▶▶ Make assumptions on the distribution of function values

▶▶ **Gaussian process**

Overview

1. Gaussian Process: Definition
2. Regression as Inference
   - GP Prior
   - Likelihood
   - Marginal Likelihood
   - Posterior
   - Predictions
3. Model Selection
   - GP Training
   - Hyper-Parameters
   - Inspection of the Marginal Likelihood
   - Covariance Function
4. Limitations and Guidelines
5. Application Areas

Gaussian Process: Definition

Gaussian Process

- We will place a distribution $p(f)$ on functions f
- Informally, a function can be considered an infinitely long vector of function values $f = [f_1, f_2, f_3, ...]$

Gaussian Process

- We will place a distribution $p(f)$ on functions f
- Informally, a function can be considered an infinitely long vector of function values $f = [f_1, f_2, f_3, ...]$
- A Gaussian process is a generalization of a multivariate Gaussian distribution to infinitely many variables.

- We will place a distribution $p(f)$ on functions f
- Informally, a function can be considered an infinitely long vector of function values $f = [f_1, f_2, f_3, ...]$
- A Gaussian process is a generalization of a multivariate Gaussian distribution to infinitely many variables.

Definition (Rasmussen & Williams, 2006)

A Gaussian process (GP) is a collection of random variables $f_1, f_2, \dots,$ any finite number of which is Gaussian distributed.

Gaussian Process

- We will place a distribution $p(f)$ on functions f
- Informally, a function can be considered an infinitely long vector of function values $f = [f_1, f_2, f_3, ...]$
- A Gaussian process is a generalization of a multivariate Gaussian distribution to infinitely many variables.

Definition (Rasmussen & Williams, 2006)

A Gaussian process (GP) is a collection of random variables $f_1, f_2, \ldots,$ any finite number of which is Gaussian distributed.

- A Gaussian distribution is specified by a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$
- A Gaussian process is specified by a mean function $m(\cdot)$ and a covariance function (kernel) $k(\cdot, \cdot)$ ▶▶ More on this later

Regression as Inference

GP Regression as Bayesian Inference

Objective

For a set of observations $y_i = f(\boldsymbol{x}_i) + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$, find a (posterior) distribution over functions $p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y})$ that explains the data. Here: $\boldsymbol{X}$ training inputs, $\boldsymbol{y}$ training targets

GP Regression as Bayesian Inference

Objective

For a set of observations $y_i = f(\boldsymbol{x}_i) + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$, find a (posterior) distribution over functions $p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y})$ that explains the data. Here: $\boldsymbol{X}$ training inputs, $\boldsymbol{y}$ training targets

Training data: $\boldsymbol{X}, \boldsymbol{y}$. Bayes' theorem yields

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\; p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

GP Regression as Bayesian Inference

Objective

For a set of observations $y_i = f(\boldsymbol{x}_i) + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$, find a (posterior) distribution over functions $p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y})$ that explains the data. Here: $\boldsymbol{X}$ training inputs, $\boldsymbol{y}$ training targets

Training data: $\boldsymbol{X}, \boldsymbol{y}$. Bayes' theorem yields

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Prior: $p(f(\cdot)) = GP(m, k)$ ▶▶ Specify mean m function and kernel k.

GP Regression as Bayesian Inference

Objective

For a set of observations $y_i = f(\boldsymbol{x}_i) + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$, find a (posterior) distribution over functions $p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y})$ that explains the data. Here: $\boldsymbol{X}$ training inputs, $\boldsymbol{y}$ training targets

Training data: $\boldsymbol{X}, \boldsymbol{y}$. Bayes' theorem yields

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\; p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Prior: $p(f(\cdot)) = GP(m, k)$ ▶▶ Specify mean m function and kernel k.

Likelihood (noise model): $p(\boldsymbol{y}|f(\boldsymbol{X})) = \mathcal{N}\big(f(\boldsymbol{X}),\, \sigma_n^2 \boldsymbol{I}\big)$

GP Regression as Bayesian Inference

Objective

For a set of observations $y_i = f(\boldsymbol{x}_i) + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$, find a (posterior) distribution over functions $p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y})$ that explains the data. Here: $\boldsymbol{X}$ training inputs, $\boldsymbol{y}$ training targets

Training data: $\boldsymbol{X}, \boldsymbol{y}$. Bayes' theorem yields

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\; p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Prior: $p(f(\cdot)) = GP(m, k)$ ▶▶ Specify mean m function and kernel k.

Likelihood (noise model): $p(\boldsymbol{y}|f(\boldsymbol{X})) = \mathcal{N}\big(f(\boldsymbol{X}),\, \sigma_n^2 \boldsymbol{I}\big)$

Marginal likelihood (evidence): $p(\boldsymbol{y}|\boldsymbol{X}) = \int p(\boldsymbol{y}|f(\cdot), \boldsymbol{X}) p(f(\cdot)|\boldsymbol{X}) df$

GP Regression as Bayesian Inference

Objective

For a set of observations $y_i = f(\boldsymbol{x}_i) + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$, find a (posterior) distribution over functions $p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y})$ that explains the data. Here: $\boldsymbol{X}$ training inputs, $\boldsymbol{y}$ training targets

Training data: $\boldsymbol{X}, \boldsymbol{y}$. Bayes' theorem yields

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\; p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Prior: $p(f(\cdot)) = GP(m, k)$ ▶▶ Specify mean m function and kernel k.

Likelihood (noise model): $p(\boldsymbol{y}|f(\boldsymbol{X})) = \mathcal{N}\big(f(\boldsymbol{X}),\, \sigma_n^2 \boldsymbol{I}\big)$

Marginal likelihood (evidence): $p(\boldsymbol{y}|\boldsymbol{X}) = \int p(\boldsymbol{y}|f(\cdot), \boldsymbol{X}) p(f(\cdot)|\boldsymbol{X}) df$

Posterior: $p(f(\cdot)|\boldsymbol{y}, \boldsymbol{X}) = GP(m_{\text{post}}, k_{\text{post}})$

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Bayesian linear regression:

- Prior $p(\boldsymbol{\theta})$ on the parameters $\boldsymbol{\theta}$ allows us to encode some properties of the parameters (e.g., range, reasonable values, ...)
- Every sample $\boldsymbol{\theta}_i \sim p(\boldsymbol{\theta})$ induces a function $f_i(\cdot) := \boldsymbol{\theta}_i^\top \boldsymbol{\phi}(\cdot)$

GP Prior

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Bayesian linear regression:

- Prior $p(\boldsymbol{\theta})$ on the parameters $\boldsymbol{\theta}$ allows us to encode some properties of the parameters (e.g., range, reasonable values, ...)
- Every sample $\boldsymbol{\theta}_i \sim p(\boldsymbol{\theta})$ induces a function $f_i(\cdot) := \boldsymbol{\theta}_i^\top \boldsymbol{\phi}(\cdot)$

Gaussian process:

- GP prior: $p(f(\cdot))$
- Function plays the role of the parameters
  - ▶▶ Every sample $f_i(\cdot) \sim GP$ is a function

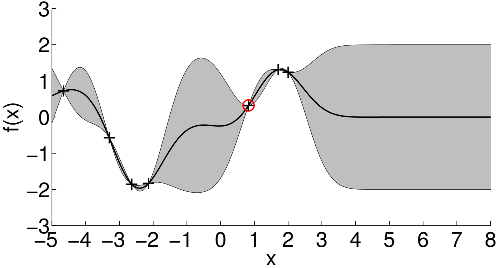
GP Prior (2)

- Bayesian prior specifies assumptions on the quantity of interest
- What assumptions could we make on the underlying function?
- What characterizes the function we want to model?

GP Prior (2)

- Bayesian prior specifies assumptions on the quantity of interest
- What assumptions could we make on the underlying function?
- What characterizes the function we want to model?
  - Mean function
  - Covariance function

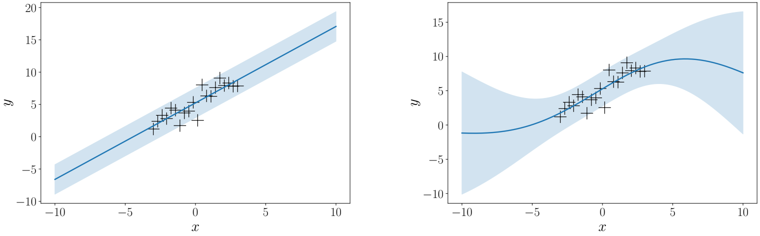
Mean Function



$$m(\boldsymbol{x}) = \mathbb{E}_f[f(\boldsymbol{x})], \quad f \sim GP$$

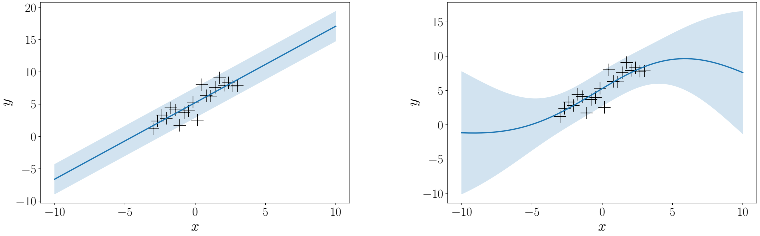
- The average function of the distribution over functions
- Allows us to bias the model (can make sense in application-specific settings)

Mean Function (2)



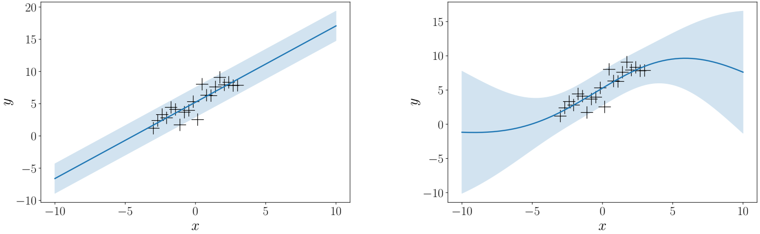
- Can be a parametrized function, e.g., linear, exponential, or neural network. Example: $m_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{\phi}(\boldsymbol{x})$

Mean Function (2)



- Can be a parametrized function, e.g., linear, exponential, or neural network. Example: $m_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{\phi}(\boldsymbol{x})$
- Prior mean function $m_{\boldsymbol{\theta}}$ can incorporate problem-specific prior knowledge (e.g., in robotics, natural sciences)
- Can simplify the learning problem

Mean Function (2)



- Can be a parametrized function, e.g., linear, exponential, or neural network. Example: $m_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{\phi}(\boldsymbol{x})$
- Prior mean function $m_{\boldsymbol{\theta}}$ can incorporate problem-specific prior knowledge (e.g., in robotics, natural sciences)
- Can simplify the learning problem
- Often: “Agnostic” mean function in the absence of data or prior knowledge: $m(\cdot) \equiv 0$ everywhere (for symmetry reasons)

Covariance Function

- Covariance function (kernel) is symmetric and positive semi-definite

Covariance Function

- Covariance function (kernel) is symmetric and positive semi-definite
- Compute covariances/correlations between (unknown) function values by just looking at the corresponding inputs:

$$\mathrm{Cov}[f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)] = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

▸▸ Kernel trick (Schölkopf & Smola, 2002)

Covariance Function

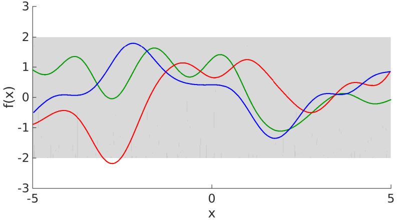
- Covariance function (kernel) is symmetric and positive semi-definite
- Compute covariances/correlations between (unknown) function values by just looking at the corresponding inputs:

$$\mathrm{Cov}[f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)] = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

  ▸▸ Kernel trick (Schölkopf & Smola, 2002)
- Encodes high-level structural assumptions (e.g., smoothness, periodicity) of the function we want to model

$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top (\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$

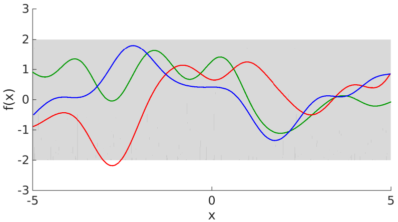
- Assumption on latent function: Smooth (∞ differentiable)



Gaussian Covariance Function

$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top (\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$

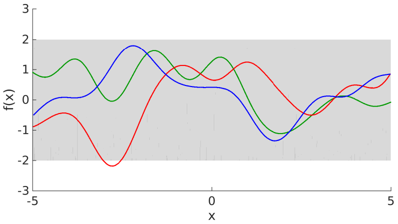
- Assumption on latent function: Smooth (∞ differentiable)
- σ_f: Amplitude of the latent function



$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top (\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$

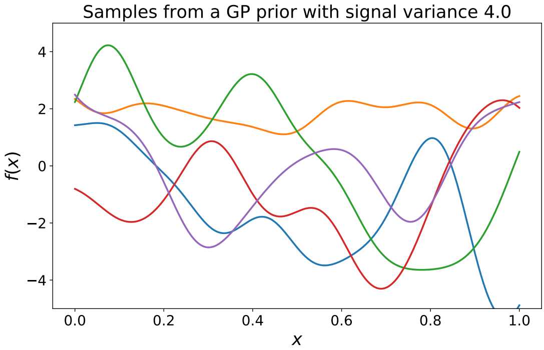
- Assumption on latent function: Smooth (∞ differentiable)
- σ_f: Amplitude of the latent function
- ℓ: Length-scale. How far do we have to move in input space before the function value changes significantly, i.e., when do function values become uncorrelated?

  ▶▶ **Smoothness parameter**



Amplitude Parameter σ_f^2

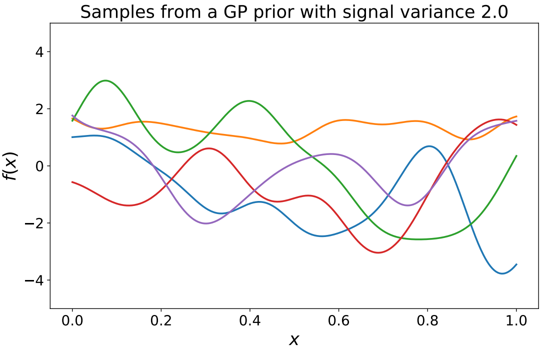
$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top (\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$



- Controls the amplitude (vertical magnitude) of the function we wish to model

Amplitude Parameter σ_f^2

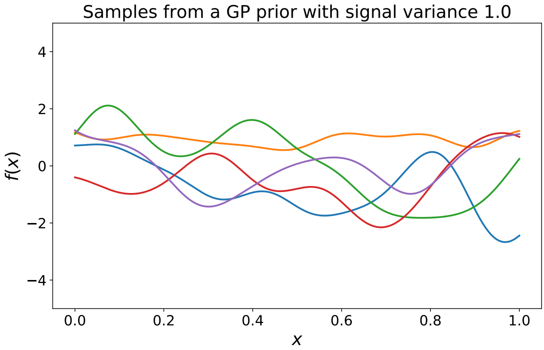
$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$



- Controls the amplitude (vertical magnitude) of the function we wish to model

Amplitude Parameter σ_f^2

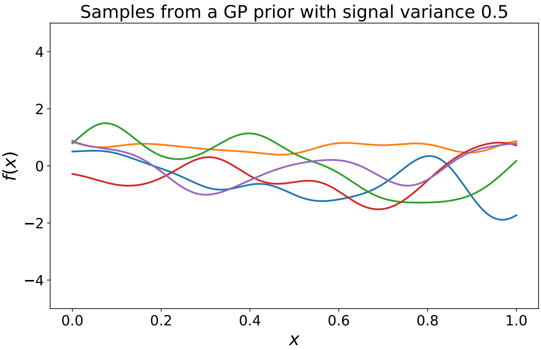
$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$



- Controls the amplitude (vertical magnitude) of the function we wish to model

Amplitude Parameter σ_f^2

$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$



- Controls the amplitude (vertical magnitude) of the function we wish to model

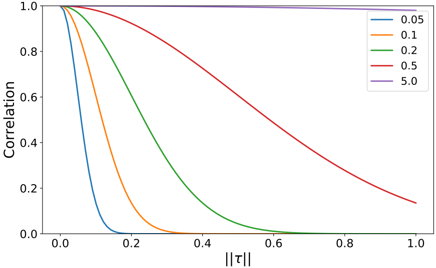
Length-Scale ℓ

$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top (\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$

- How “wiggly” is the function?
- How much information we can transfer to other function values?
  - ▸▸ Correlation between function values
- How far do we have to move in input space from x to x' to make $f(x)$ and $f(x')$ uncorrelated?

Length-Scale ℓ (2)

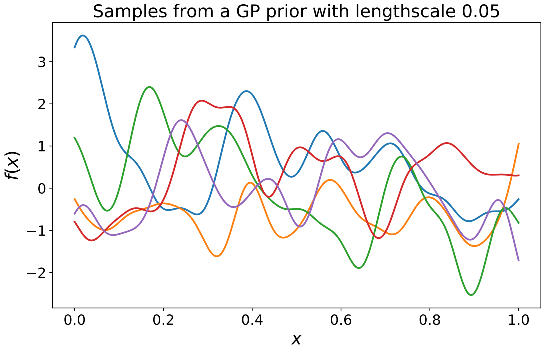
$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$



- Correlation between function values $f(\boldsymbol{x})$ and $f(\boldsymbol{x}')$ depends on the (scaled) distance $\|\tau\|/\ell = \|\boldsymbol{x} - \boldsymbol{x}'\|/\ell$ of the corresponding inputs.
- What does a short/long length-scale ℓ imply?

Length-Scale ℓ (3)

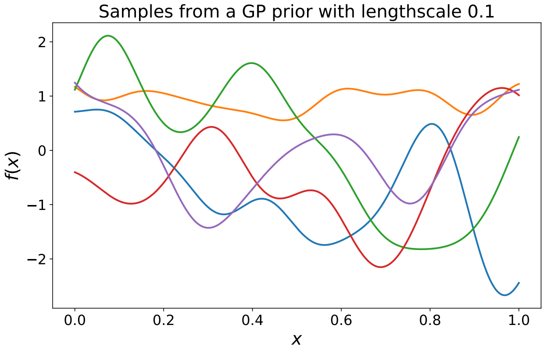
$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$



▶▶ Explore interactive diagrams at **https://drafts.distill.pub/gp/**

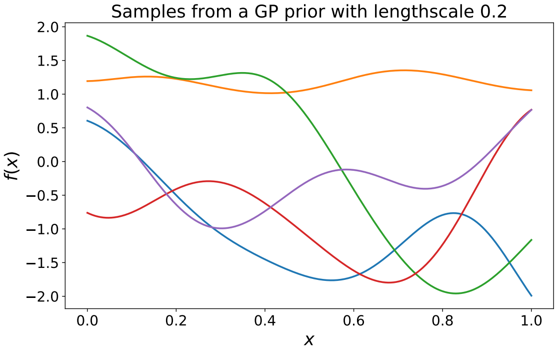
Length-Scale ℓ (3)

$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top (\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$



Explore interactive diagrams at
https://drafts.distill.pub/gp/

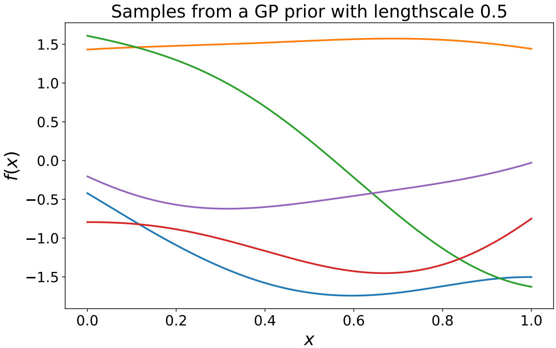
$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$



▶▶ Explore interactive diagrams at **https://drafts.distill.pub/gp/**

Length-Scale ℓ (3)

$$k_{Gauss}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\big(-(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)/\ell^2\big)$$



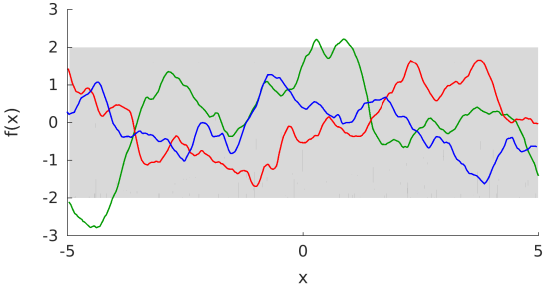
Explore interactive diagrams at

`https://drafts.distill.pub/gp/`

Matérn Covariance Function

$$k_{Mat,3/2}(x_i, x_j) = \sigma_f^2 \left(1 + \frac{\sqrt{3}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\ell}\right) \exp\left(-\frac{\sqrt{3}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\ell}\right)$$

- Assumption on latent function: 1-times differentiable
- σ_f: Amplitude of the latent function
- ℓ: Length-scale. How far do we have to move in input space before the function value changes significantly?

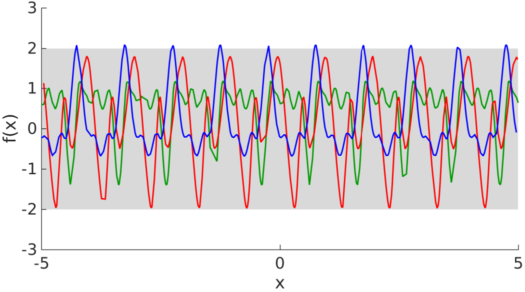


Periodic Covariance Function

$$k_{per}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{2\sin^2\left(\frac{\kappa(x_i - x_j)}{2\pi}\right)}{\ell^2}\right)$$

$$= k_{Gauss}(\boldsymbol{u}(x_i), \boldsymbol{u}(x_j)), \quad \boldsymbol{u}(x) = \begin{bmatrix}\cos(\kappa x)\\ \sin(\kappa x)\end{bmatrix}$$

- Assumption on latent function: periodic
- Periodicity parameter κ



Creating New Covariance Functions

Assume k_1 and k_2 are valid covariance functions and $u(\cdot)$ is a (nonlinear) transformation of the input space. Then

- $k_1 + k_2$ is a valid covariance function

Creating New Covariance Functions

Assume k_1 and k_2 are valid covariance functions and $u(\cdot)$ is a (nonlinear) transformation of the input space. Then

- $k_1 + k_2$ is a valid covariance function
- $k_1 k_2$ is a valid covariance function

Creating New Covariance Functions

Assume k_1 and k_2 are valid covariance functions and $u(\cdot)$ is a (nonlinear) transformation of the input space. Then

- $k_1 + k_2$ is a valid covariance function
- $k_1 k_2$ is a valid covariance function
- $k(u(\boldsymbol{x}), u(\boldsymbol{x}'))$ is a valid covariance function (MacKay, 1998)
  - Periodic covariance function
  - Manifold Gaussian process (Calandra et al., 2016)
  - Deep kernel learning (Wilson et al., 2016)

Creating New Covariance Functions

Assume k_1 and k_2 are valid covariance functions and $u(\cdot)$ is a (nonlinear) transformation of the input space. Then

- $k_1 + k_2$ is a valid covariance function
- $k_1 k_2$ is a valid covariance function
- $k(u(\boldsymbol{x}), u(\boldsymbol{x}'))$ is a valid covariance function (MacKay, 1998)
  - Periodic covariance function
  - Manifold Gaussian process (Calandra et al., 2016)
  - Deep kernel learning (Wilson et al., 2016)

Automatic Statistician (Lloyd et al., 2014)

(Gaussian) Likelihood

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

(Gaussian) Likelihood

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\; p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Gaussian likelihood in linear regression:

$$p(y|\boldsymbol{x}, \boldsymbol{\theta}) = \mathcal{N}\big(y \,|\, \boldsymbol{\theta}^\top \boldsymbol{x},\, \sigma_n^2\big)$$

- Function (not a distribution) of the parameters
- Describes how parameters and observed data are connected
- Tells us how to transform parameters into (noisy) data

(Gaussian) Likelihood

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Gaussian likelihood in linear regression:

$$p(y|\boldsymbol{x}, \boldsymbol{\theta}) = \mathcal{N}\big(y \,|\, \boldsymbol{\theta}^\top \boldsymbol{x},\, \sigma_n^2\big)$$

- Function (not a distribution) of the parameters
- Describes how parameters and observed data are connected
- Tells us how to transform parameters into (noisy) data

Gaussian likelihood in Gaussian processes:

$$p(y|f(\boldsymbol{x})) = \mathcal{N}\big(y \,|\, f(\boldsymbol{x}),\, \sigma_n^2\big)$$

- Intuition: Parameters are the function f itself

Marginal Likelihood

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Bayesian linear regression with a Gaussian prior $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{0},\, \boldsymbol{I})$:

$$p(\boldsymbol{y}|\boldsymbol{X}) = \int p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})p(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}$$

- Normalizes the posterior distribution

Marginal Likelihood

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\; p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Bayesian linear regression with a Gaussian prior $p(\boldsymbol{\theta}) = \mathcal{N}\big(\boldsymbol{0},\ \boldsymbol{I}\big)$:

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{X}) &= \int p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})p(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta} \\ &= \mathcal{N}\big(\boldsymbol{y}\,|\,\boldsymbol{0},\ \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2\boldsymbol{I}\big) \end{aligned}$$

- Normalizes the posterior distribution
- Can be computed analytically

Marginal Likelihood

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Bayesian linear regression with a Gaussian prior $p(\boldsymbol{\theta}) = \mathcal{N}\big(\boldsymbol{0},\ \boldsymbol{I}\big)$:

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{X}) &= \int p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} \\ &= \mathcal{N}\big(\boldsymbol{y}\,|\,\boldsymbol{0},\ \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \boldsymbol{I}\big) \\ &= \mathbb{E}_{\boldsymbol{\theta}}[p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})] \end{aligned}$$

- Normalizes the posterior distribution
- Can be computed analytically
- Expected likelihood (under the parameter prior)
- Expected predictive distribution of the training targets $\boldsymbol{y}$ (under the parameter prior)

Marginal Likelihood (2)

Gaussian process marginal likelihood

$$p(\boldsymbol{y}|\boldsymbol{X}) = \int p(\boldsymbol{y}|f(\boldsymbol{X}))p(f(\boldsymbol{X}))\mathrm{d}f$$

- Normalizes the posterior distribution

Marginal Likelihood (2)

Gaussian process marginal likelihood

$$p(\boldsymbol{y}|\boldsymbol{X}) = \int p(\boldsymbol{y}|f(\boldsymbol{X}))p(f(\boldsymbol{X}))\mathrm{d}f$$
$$= \mathcal{N}\big(\boldsymbol{y} \,|\, \boldsymbol{0},\, \boldsymbol{K} + \sigma^2\boldsymbol{I}\big)$$

- Normalizes the posterior distribution
- Can be computed analytically

Marginal Likelihood (2)

Gaussian process marginal likelihood

$$
\begin{aligned}
p(\boldsymbol{y}|\boldsymbol{X}) &= \int p(\boldsymbol{y}|f(\boldsymbol{X}))p(f(\boldsymbol{X}))\mathrm{d}f \\
&= \mathcal{N}\big(\boldsymbol{y}\,|\,\boldsymbol{0},\,\boldsymbol{K}+\sigma^2\boldsymbol{I}\big) \\
&= \mathbb{E}_f[p(\boldsymbol{y}|f(\boldsymbol{X}))]
\end{aligned}
$$

- Normalizes the posterior distribution
- Can be computed analytically
- Expected likelihood (under the GP prior)
- Expected predictive distribution of the training targets $\boldsymbol{y}$ (under the GP prior)

Marginal Likelihood (2)

Gaussian process marginal likelihood

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{X}) &= \int p(\boldsymbol{y}|f(\boldsymbol{X}))p(f(\boldsymbol{X}))\mathrm{d}f \\ &= \mathcal{N}\big(\boldsymbol{y}\,|\,\boldsymbol{0},\ \boldsymbol{K} + \sigma^2\boldsymbol{I}\big) \\ &= \mathbb{E}_f[p(\boldsymbol{y}|f(\boldsymbol{X}))] \end{aligned}$$

- Normalizes the posterior distribution
- Can be computed analytically
- Expected likelihood (under the GP prior)
- Expected predictive distribution of the training targets $\boldsymbol{y}$ (under the GP prior)

$$\log p(\boldsymbol{y}|\boldsymbol{X}) = -\frac{1}{2}\boldsymbol{y}^\top(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y} - \frac{1}{2}\log|\boldsymbol{K} + \sigma_n^2\boldsymbol{I}| - \frac{N}{2}\log(2\pi)$$

$$K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j), \quad i, j = 1, \ldots, N$$

GP Posterior

Posterior over functions (with training data $\boldsymbol{X}, \boldsymbol{y}$):

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

GP Posterior

Posterior over functions (with training data $\boldsymbol{X}, \boldsymbol{y}$):

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Using the properties of Gaussians, we obtain (with $\boldsymbol{K} := k(\boldsymbol{X}, \boldsymbol{X})$)

$$p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot)) = \mathcal{N}\big(\boldsymbol{y} \,|\, f(\boldsymbol{X}),\, \sigma_n^2 \boldsymbol{I}\big)\, GP(m(\cdot), k(\cdot, \cdot))$$

GP Posterior

Posterior over functions (with training data $\boldsymbol{X}, \boldsymbol{y}$):

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Using the properties of Gaussians, we obtain (with $\boldsymbol{K} := k(\boldsymbol{X}, \boldsymbol{X})$)

$$p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot)) = \mathcal{N}\big(\boldsymbol{y}\,|\,f(\boldsymbol{X}),\, \sigma_n^2 \boldsymbol{I}\big)\; GP(m(\cdot), k(\cdot, \cdot))$$

$$= Z \times GP\big(m_{\text{post}}(\cdot), k_{post}(\cdot, \cdot)\big)$$

$$m_{\text{post}}(\cdot) = m(\cdot) + k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}(\boldsymbol{y} - m(\boldsymbol{X}))$$

$$k_{\text{post}}(\cdot, \cdot) = k(\cdot, \cdot) - k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \cdot)$$

GP Posterior

Posterior over functions (with training data $\boldsymbol{X}, \boldsymbol{y}$):

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot))}{p(\boldsymbol{y}|\boldsymbol{X})}$$

Using the properties of Gaussians, we obtain (with $\boldsymbol{K} := k(\boldsymbol{X}, \boldsymbol{X})$)

$$p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\cdot)) = \mathcal{N}\big(\boldsymbol{y} \,|\, f(\boldsymbol{X}),\, \sigma_n^2 \boldsymbol{I}\big)\, GP(m(\cdot), k(\cdot, \cdot))$$

$$= Z \times GP\big(m_{\text{post}}(\cdot), k_{post}(\cdot, \cdot)\big)$$

$$m_{\text{post}}(\cdot) = m(\cdot) + k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}(\boldsymbol{y} - m(\boldsymbol{X}))$$

$$k_{\text{post}}(\cdot, \cdot) = k(\cdot, \cdot) - k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \cdot)$$

Marginal likelihood:

$$Z = p(\boldsymbol{y}|\boldsymbol{X}) = \int p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\boldsymbol{X}))\, \mathrm{d}f = \mathcal{N}\big(\boldsymbol{y} \,|\, m(\boldsymbol{X}),\, \boldsymbol{K} + \sigma_n^2 \boldsymbol{I}\big)$$

Sampling from the GP Prior

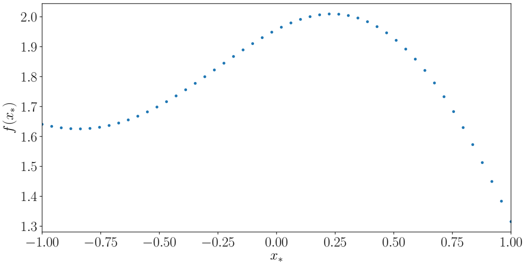
- GP is a distribution over functions
  - A sample from a GP will be an entire function

Sampling from the GP Prior

- GP is a distribution over functions
  - A sample from a GP will be an entire function
- In practice, we cannot sample functions directly

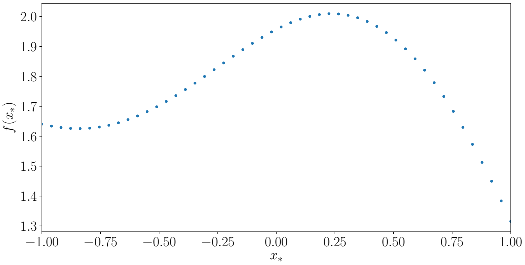
Sampling from the GP Prior

- GP is a distribution over functions
  - A sample from a GP will be an entire function
- In practice, we cannot sample functions directly
- Instead: function = collection of function values



Sampling from the GP Prior

- GP is a distribution over functions
  - A sample from a GP will be an entire function
- In practice, we cannot sample functions directly
- Instead: function = collection of function values
- Determine function values at a finite set of input locations $\boldsymbol{X}_* = [\boldsymbol{x}_*^{(1)}, \ldots, \boldsymbol{x}_*^{(K)}]$



Sampling from the GP Prior (2)

- Without any training data, the predictive distribution at test points $\boldsymbol{X}_*$ is

$$
\begin{aligned}
p(\boldsymbol{f}(\boldsymbol{X}_*)|\boldsymbol{X}_*) &= \mathcal{N}\big(\mathbb{E}_f[f(\boldsymbol{X}_*)],\ \mathbb{V}_f[f(\boldsymbol{X}_*)]\big) \\
&= \mathcal{N}\big(m_{\text{prior}}(\boldsymbol{X}_*),\ k_{\text{prior}}(\boldsymbol{X}_*, \boldsymbol{X}_*)\big)
\end{aligned}
$$

Sampling from the GP Prior (2)

- Without any training data, the predictive distribution at test points $\boldsymbol{X}_*$ is

$$
\begin{aligned}
p(\boldsymbol{f}(\boldsymbol{X}_*)|\boldsymbol{X}_*) &= \mathcal{N}\big(\mathbb{E}_f[f(\boldsymbol{X}_*)],\, \mathbb{V}_f[f(\boldsymbol{X}_*)]\big) \\
&= \mathcal{N}\big(m_{\text{prior}}(\boldsymbol{X}_*),\, k_{\text{prior}}(\boldsymbol{X}_*, \boldsymbol{X}_*)\big)
\end{aligned}
$$

- Exploited: Definition of GP that all function values are jointly Gaussian distributed

Sampling from the GP Prior (2)

- Without any training data, the predictive distribution at test points $\boldsymbol{X}_*$ is

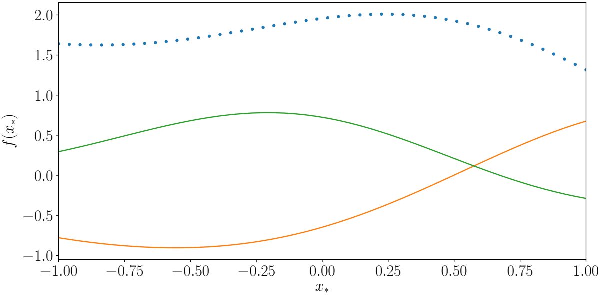
$$
\begin{aligned}
p(\boldsymbol{f}(\boldsymbol{X}_*)|\boldsymbol{X}_*) &= \mathcal{N}\big(\mathbb{E}_f[f(\boldsymbol{X}_*)],\ \mathbb{V}_f[f(\boldsymbol{X}_*)]\big) \\
&= \mathcal{N}\big(m_{\text{prior}}(\boldsymbol{X}_*),\ k_{\text{prior}}(\boldsymbol{X}_*, \boldsymbol{X}_*)\big)
\end{aligned}
$$

- Exploited: Definition of GP that all function values are jointly Gaussian distributed
- Generate "function draws" (samples from the GP prior)

$$
f_k(\boldsymbol{X}_*) \sim \mathcal{N}\big(m_{\text{prior}}(\boldsymbol{X}_*),\ k_{\text{prior}}(\boldsymbol{X}_*, \boldsymbol{X}_*)\big)
$$

- Goal: Generate random functions f_k, so that

$$f_k(\boldsymbol{X}_*) \sim \mathcal{N}\big(m_{\text{prior}}(\boldsymbol{X}_*),\, k_{\text{prior}}(\boldsymbol{X}_*, \boldsymbol{X}_*)\big)$$



- Goal: Generate random functions f_k, so that

$$f_k(\boldsymbol{X}_*) \sim \mathcal{N}\big(m_{\text{prior}}(\boldsymbol{X}_*),\, k_{\text{prior}}(\boldsymbol{X}_*, \boldsymbol{X}_*)\big)$$

- Define $\boldsymbol{m}_* := m_{\text{prior}}(\boldsymbol{X}_*)$ and $\boldsymbol{K}_{**} := k_{\text{prior}}(\boldsymbol{X}_*, \boldsymbol{X}_*)$. Then

$$f_k(\boldsymbol{X}_*) \sim \mathcal{N}\big(\boldsymbol{m}_*,\, \boldsymbol{K}_{**}\big)$$

▶▶ Sample from a multivariate Gaussian

GP Predictions (Posterior)

$$y = f(\boldsymbol{x}) + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$$

- **Objective:** Find $p(f(\boldsymbol{X}_*)|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*)$ for training data $\boldsymbol{X}, \boldsymbol{y}$ and test inputs $\boldsymbol{X}_*$.
- GP prior at training inputs: $p(f|\boldsymbol{X}) = \mathcal{N}\big(m(\boldsymbol{X}),\, \boldsymbol{K}\big)$
- Gaussian Likelihood: $p(\boldsymbol{y}|f, \boldsymbol{X}) = \mathcal{N}\big(f(\boldsymbol{X}),\, \sigma_n^2\boldsymbol{I}\big)$

GP Predictions (Posterior)

$$y = f(\boldsymbol{x}) + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$$

- **Objective:** Find $p(f(\boldsymbol{X}_*)|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*)$ for training data $\boldsymbol{X}, \boldsymbol{y}$ and test inputs $\boldsymbol{X}_*$.
- GP prior at training inputs: $p(f|\boldsymbol{X}) = \mathcal{N}\big(m(\boldsymbol{X}),\, \boldsymbol{K}\big)$
- Gaussian Likelihood: $p(\boldsymbol{y}|f, \boldsymbol{X}) = \mathcal{N}\big(f(\boldsymbol{X}),\, \sigma_n^2 \boldsymbol{I}\big)$
- With $f \sim GP$ it follows that $\boldsymbol{f}, \boldsymbol{f}_*$ are jointly Gaussian distributed:

$$p(\boldsymbol{f}, \boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{X}_*) = \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{X}) \\ m(\boldsymbol{X}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} & k(\boldsymbol{X}, \boldsymbol{X}_*) \\ k(\boldsymbol{X}_*, \boldsymbol{X}) & k(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right)$$

GP Predictions (Posterior)

$$y = f(\boldsymbol{x}) + \epsilon, \quad \epsilon \sim \mathcal{N}\big(0,\, \sigma_n^2\big)$$

- **Objective:** Find $p(f(\boldsymbol{X}_*)|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*)$ for training data $\boldsymbol{X}, \boldsymbol{y}$ and test inputs $\boldsymbol{X}_*$.
- GP prior at training inputs: $p(f|\boldsymbol{X}) = \mathcal{N}\big(m(\boldsymbol{X}),\, \boldsymbol{K}\big)$
- Gaussian Likelihood: $p(\boldsymbol{y}|f, \boldsymbol{X}) = \mathcal{N}\big(f(\boldsymbol{X}),\, \sigma_n^2 \boldsymbol{I}\big)$
- With $f \sim GP$ it follows that $\boldsymbol{f}, \boldsymbol{f}_*$ are jointly Gaussian distributed:

$$p(\boldsymbol{f}, \boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{X}_*) = \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{X}) \\ m(\boldsymbol{X}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} & k(\boldsymbol{X}, \boldsymbol{X}_*) \\ k(\boldsymbol{X}_*, \boldsymbol{X}) & k(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right)$$

- Due to the Gaussian likelihood, we also get ($\boldsymbol{f}$ is unobserved)

$$p(\boldsymbol{y}, \boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{X}_*) = \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{X}) \\ m(\boldsymbol{X}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} + \sigma_n^2 \boldsymbol{I} & k(\boldsymbol{X}, \boldsymbol{X}_*) \\ k(\boldsymbol{X}_*, \boldsymbol{X}) & k(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right)$$

GP Posterior Predictions

Prior evaluated at $\boldsymbol{X}, \boldsymbol{X}_*$:

$$p(\boldsymbol{y}, \boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{X}_*) = \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{X}) \\ m(\boldsymbol{X}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} + \sigma_n^2 \boldsymbol{I} & k(\boldsymbol{X}, \boldsymbol{X}_*) \\ k(\boldsymbol{X}_*, \boldsymbol{X}) & k(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right)$$

Posterior predictive distribution $p(\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*)$ at test inputs $\boldsymbol{X}_*$

GP Posterior Predictions

Prior evaluated at $\boldsymbol{X}, \boldsymbol{X}_*$:

$$p(\boldsymbol{y}, \boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{X}_*) = \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{X}) \\ m(\boldsymbol{X}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} + \sigma_n^2 \boldsymbol{I} & k(\boldsymbol{X}, \boldsymbol{X}_*) \\ k(\boldsymbol{X}_*, \boldsymbol{X}) & k(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right)$$

Posterior predictive distribution $p(\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*)$ at test inputs $\boldsymbol{X}_*$ obtained by Gaussian conditioning:

$$p(\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*) = \mathcal{N}\big(\mathbb{E}[\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*], \; \mathbb{V}[\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*]\big)$$

$$\mathbb{E}[\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*] = \underbrace{m(\boldsymbol{X}_*)}_{\text{prior mean}} + \underbrace{k(\boldsymbol{X}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}}_{\text{“Kalman gain”}} \underbrace{(\boldsymbol{y} - m(\boldsymbol{X}))}_{\text{error}}$$

GP Posterior Predictions

Prior evaluated at $\boldsymbol{X}, \boldsymbol{X}_*$:

$$p(\boldsymbol{y}, \boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{X}_*) = \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{X}) \\ m(\boldsymbol{X}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} + \sigma_n^2 \boldsymbol{I} & k(\boldsymbol{X}, \boldsymbol{X}_*) \\ k(\boldsymbol{X}_*, \boldsymbol{X}) & k(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right)$$

Posterior predictive distribution $p(\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*)$ at test inputs $\boldsymbol{X}_*$ obtained by Gaussian conditioning:

$$p(\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*) = \mathcal{N}\big(\mathbb{E}[\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*],\ \mathbb{V}[\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*]\big)$$

$$\mathbb{E}[\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*] = \underbrace{m(\boldsymbol{X}_*)}_{\text{prior mean}} + \underbrace{k(\boldsymbol{X}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}}_{\text{“Kalman gain”}} \underbrace{(\boldsymbol{y} - m(\boldsymbol{X}))}_{\text{error}}$$

$$\mathbb{V}[\boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*] = \underbrace{k(\boldsymbol{X}_*, \boldsymbol{X}_*)}_{\text{prior variance}} - \underbrace{k(\boldsymbol{X}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \boldsymbol{X}_*)}_{\geqslant 0}$$

Sanity Check

- GP posterior (from earlier):

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = GP\big(m_{\text{post}}(\cdot), k_{post}(\cdot, \cdot)\big)$$

$$m_{\text{post}}(\cdot) = m(\cdot) + k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}(\boldsymbol{y} - m(\boldsymbol{X}))$$

$$k_{\text{post}}(\cdot, \cdot) = k(\cdot, \cdot) - k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \cdot)$$

Sanity Check

- GP posterior (from earlier):

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = GP\big(m_{\text{post}}(\cdot), k_{post}(\cdot, \cdot)\big)$$

$$m_{\text{post}}(\cdot) = m(\cdot) + k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}(\boldsymbol{y} - m(\boldsymbol{X}))$$

$$k_{\text{post}}(\cdot, \cdot) = k(\cdot, \cdot) - k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}k(\boldsymbol{X}, \cdot)$$

- GP posterior predictions at $\boldsymbol{X}_*$:

$$p(\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*) = \mathcal{N}\big(\mathbb{E}[\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*],\ \mathbb{V}[\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*]\big)$$

$$\mathbb{E}[\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*] = m(\boldsymbol{X}_*) + k(\boldsymbol{X}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}(\boldsymbol{y} - m(\boldsymbol{X}))$$

$$\mathbb{V}[\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*] = k(\boldsymbol{X}_*, \boldsymbol{X}_*) - k(\boldsymbol{X}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}k(\boldsymbol{X}, \boldsymbol{X}_*)$$

Sanity Check

- GP posterior (from earlier):

$$p(f(\cdot)|\boldsymbol{X}, \boldsymbol{y}) = GP\big(m_{\text{post}}(\cdot), k_{post}(\cdot, \cdot)\big)$$

$$m_{\text{post}}(\cdot) = m(\cdot) + k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}(\boldsymbol{y} - m(\boldsymbol{X}))$$

$$k_{\text{post}}(\cdot, \cdot) = k(\cdot, \cdot) - k(\cdot, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \cdot)$$

- GP posterior predictions at $\boldsymbol{X}_*$:

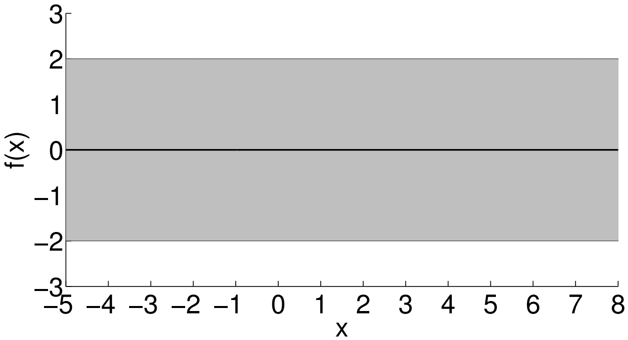
$$p(\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*) = \mathcal{N}\big(\mathbb{E}[\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*], \mathbb{V}[\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*]\big)$$

$$\mathbb{E}[\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*] = m(\boldsymbol{X}_*) + k(\boldsymbol{X}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}(\boldsymbol{y} - m(\boldsymbol{X}))$$

$$\mathbb{V}[\boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*] = k(\boldsymbol{X}_*, \boldsymbol{X}_*) - k(\boldsymbol{X}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \boldsymbol{X}_*)$$

Predictions

Make predictions by evaluating the GP posterior mean and covariance function at a finite number of inputs $\boldsymbol{X}_*$

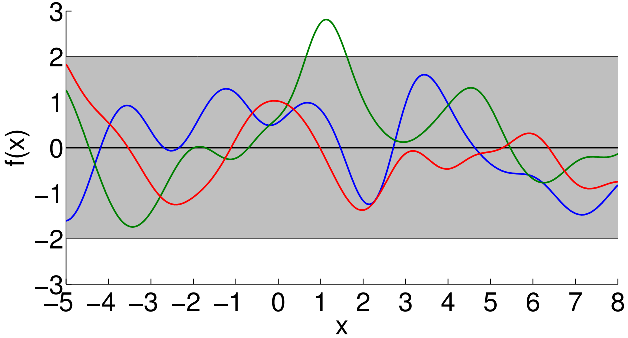


Prior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \varnothing] = m(\boldsymbol{x}_*) = 0$$
$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \varnothing] = \sigma^2(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$$

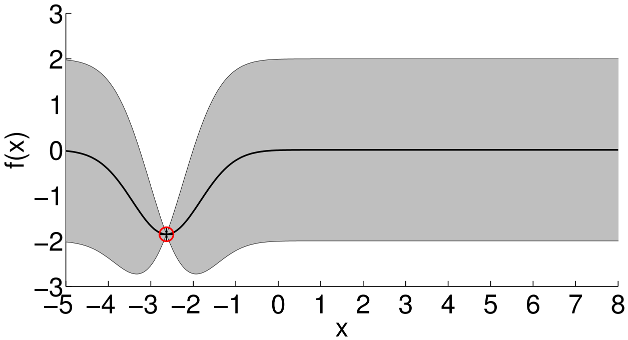
Illustration: Inference with Gaussian Processes



Prior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \varnothing] = m(\boldsymbol{x}_*) = 0$$
$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \varnothing] = \sigma^2(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$$

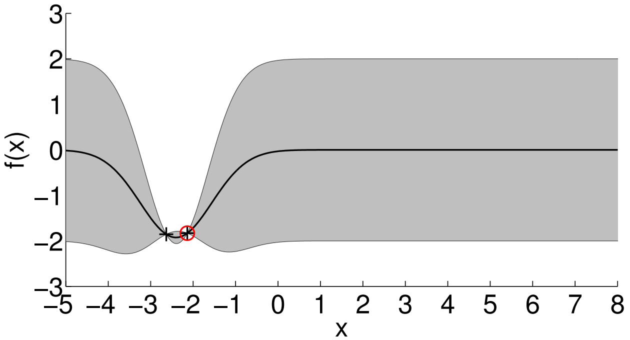


Posterior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = m(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y}$$
$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \boldsymbol{x}_*)$$

Illustration: Inference with Gaussian Processes



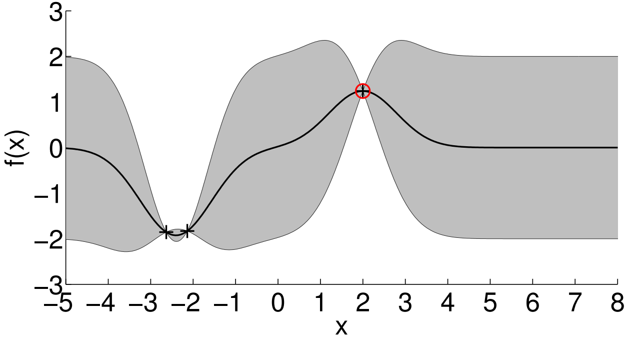
Posterior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = m(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y}$$

$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}k(\boldsymbol{X}, \boldsymbol{x}_*)$$

Illustration: Inference with Gaussian Processes

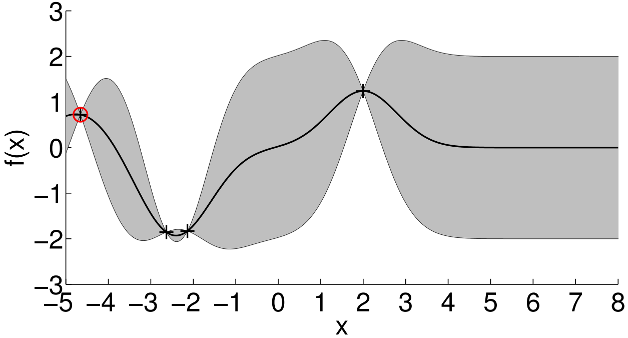


Posterior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = m(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y}$$
$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}k(\boldsymbol{X}, \boldsymbol{x}_*)$$

Illustration: Inference with Gaussian Processes



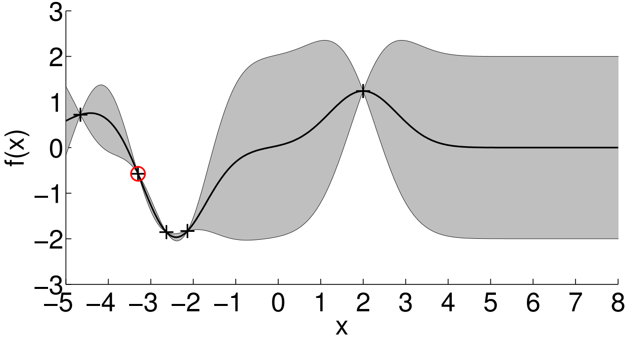
Posterior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = m(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y}$$

$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}k(\boldsymbol{X}, \boldsymbol{x}_*)$$

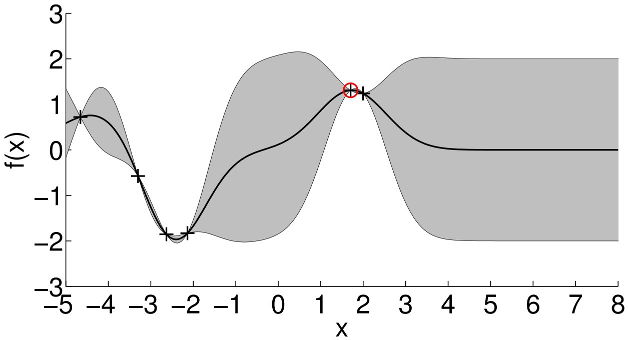
Illustration: Inference with Gaussian Processes



Posterior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = m(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y}$$
$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}k(\boldsymbol{X}, \boldsymbol{x}_*)$$

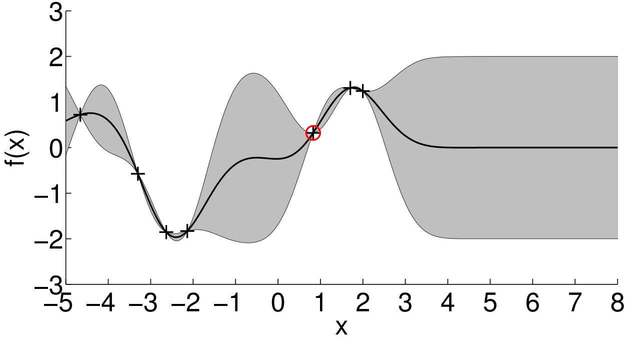


Posterior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = m(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y}$$
$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}k(\boldsymbol{X}, \boldsymbol{x}_*)$$

Illustration: Inference with Gaussian Processes



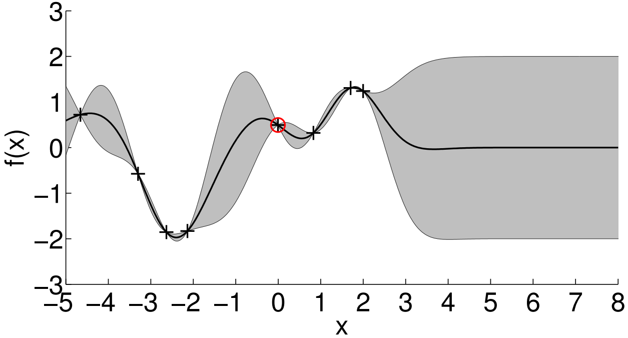
Posterior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = m(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y}$$

$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \boldsymbol{x}_*)$$

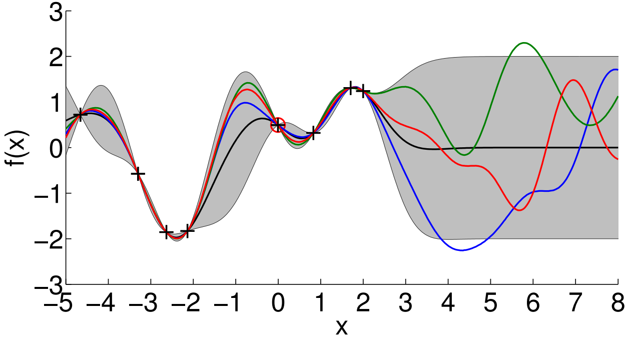
Illustration: Inference with Gaussian Processes



Posterior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = m(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y}$$
$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \boldsymbol{x}_*)$$



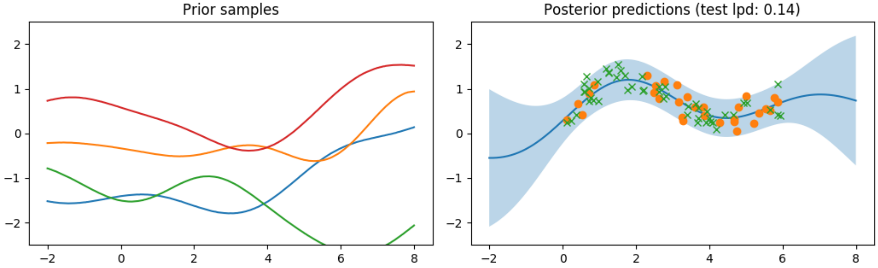
Posterior belief about the function

Predictive (marginal) mean and variance:

$$\mathbb{E}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = m(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y}$$
$$\mathbb{V}[f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k(\boldsymbol{x}_*, \boldsymbol{X})(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1}k(\boldsymbol{X}, \boldsymbol{x}_*)$$

Model Selection

Influence of Prior on Posterior

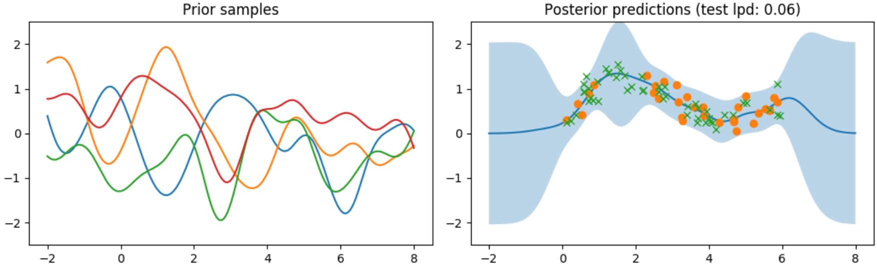


- Generalization error measured by log-predictive density (lpd)

$$\text{lpd} = \log p(y_* | \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}, \ell)$$

for different length-scales ℓ and different datasets

Influence of Prior on Posterior



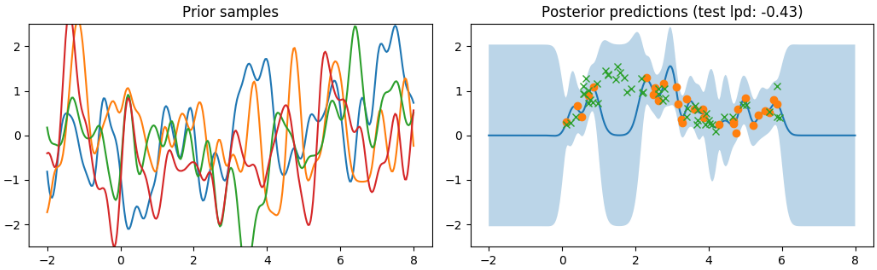
- Generalization error measured by log-predictive density (lpd)

$$\text{lpd} = \log p(y_* | \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}, \ell)$$

  for different length-scales ℓ and different datasets

- Shorter length-scale ⏩ More flexible model ⏩ Faster increase in uncertainty away from data ⏩ Bad generalization properties

Influence of Prior on Posterior



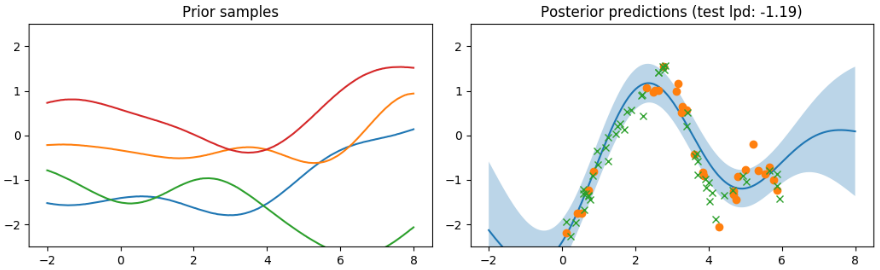
- Generalization error measured by log-predictive density (lpd)

$$\text{lpd} = \log p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}, \ell)$$

for different length-scales ℓ and different datasets

- Shorter length-scale ⏩ More flexible model ⏩ Faster increase in uncertainty away from data ⏩ Bad generalization properties

Influence of Prior on Posterior



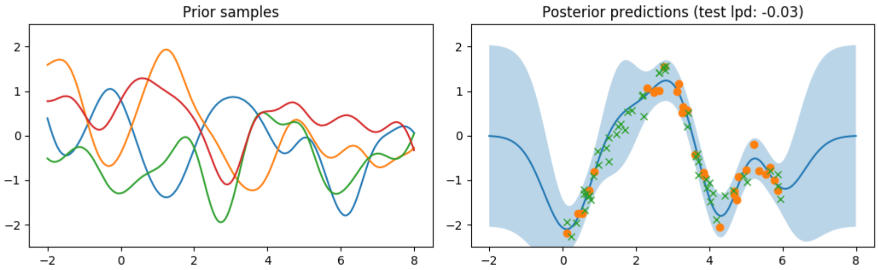
- Generalization error measured by log-predictive density (lpd)

$$\text{lpd} = \log p(y_* | \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}, \ell)$$

  for different length-scales ℓ and different datasets

- Shorter length-scale ⏩ More flexible model ⏩ Faster increase in uncertainty away from data ⏩ Bad generalization properties

Influence of Prior on Posterior

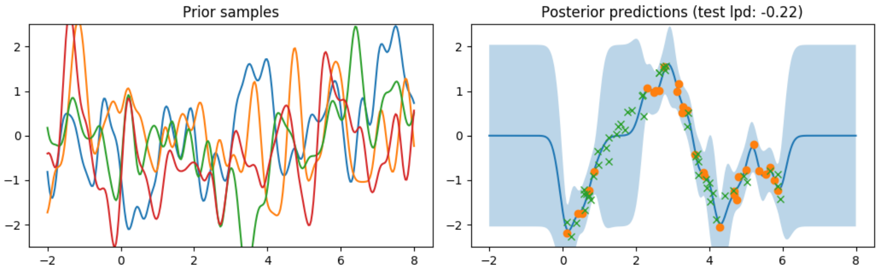


- Generalization error measured by log-predictive density (lpd)

$$\text{lpd} = \log p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}, \ell)$$

  for different length-scales ℓ and different datasets

- Shorter length-scale ⏩ More flexible model ⏩ Faster increase in uncertainty away from data ⏩ Bad generalization properties



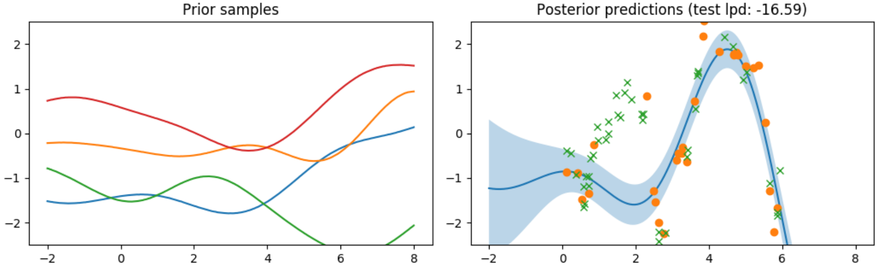
- Generalization error measured by log-predictive density (lpd)

$$\text{lpd} = \log p(y_* | \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}, \ell)$$

  for different length-scales ℓ and different datasets

- Shorter length-scale ⏩ More flexible model ⏩ Faster increase in uncertainty away from data ⏩ Bad generalization properties

Influence of Prior on Posterior



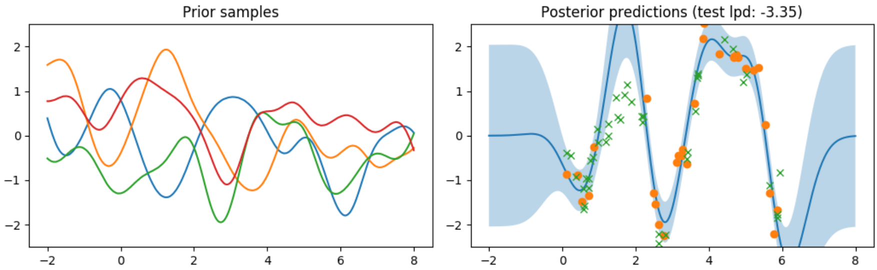
- Generalization error measured by log-predictive density (lpd)

$$\text{lpd} = \log p(y_* | \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}, \ell)$$

  for different length-scales ℓ and different datasets

- Shorter length-scale ⏩ More flexible model ⏩ Faster increase in uncertainty away from data ⏩ Bad generalization properties

Influence of Prior on Posterior



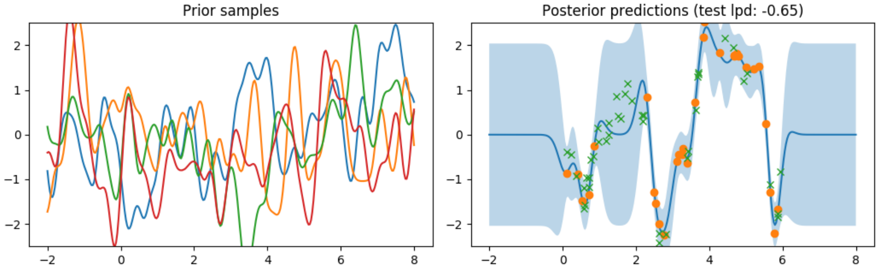
- Generalization error measured by log-predictive density (lpd)

$$\text{lpd} = \log p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}, \ell)$$

  for different length-scales ℓ and different datasets

- Shorter length-scale ⏩ More flexible model ⏩ Faster increase in uncertainty away from data ⏩ Bad generalization properties

Influence of Prior on Posterior



- Generalization error measured by log-predictive density (lpd)

$$\text{lpd} = \log p(y_* | \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}, \ell)$$

  for different length-scales ℓ and different datasets

- Shorter length-scale ⏩ More flexible model ⏩ Faster increase in uncertainty away from data ⏩ Bad generalization properties

Model Selection

- Make predictions equipped with uncertainty
- Choice of prior (e.g., length-scale) influences predictions
- Different tasks require different priors

- Make predictions equipped with uncertainty
- Choice of prior (e.g., length-scale) influences predictions
- Different tasks require different priors

How do we select a good prior?

Model Selection

- Make predictions equipped with uncertainty
- Choice of prior (e.g., length-scale) influences predictions
- Different tasks require different priors

How do we select a good prior?

Model Selection in GPs

- Choose hyper-parameters of the GP
- Choose good mean function and kernel

Hyper-Parameters of a GP

The GP possesses a set of hyper-parameters:

- Parameters of the mean function
- Parameters of the covariance function (e.g., length-scales and signal variance)
- Likelihood parameters (e.g., noise variance σ_n^2)

Hyper-Parameters of a GP

The GP possesses a set of hyper-parameters:

- Parameters of the mean function
- Parameters of the covariance function (e.g., length-scales and signal variance)
- Likelihood parameters (e.g., noise variance σ_n^2)

▶▶ Train a GP to find a good set of hyper-parameters

The GP possesses a set of hyper-parameters:

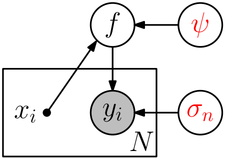
- Parameters of the mean function
- Parameters of the covariance function (e.g., length-scales and signal variance)
- Likelihood parameters (e.g., noise variance σ_n^2)

▸▸ Train a GP to find a good set of hyper-parameters

▸▸ Higher-level model selection to find good mean and covariance functions
(can also be automated: Automatic Statistician (Lloyd et al., 2014))

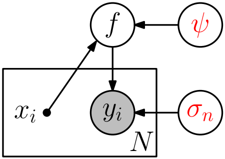
GP Training

Find good hyper-parameters $\boldsymbol{\theta}$ (kernel/mean function parameters ψ, noise variance σ_n^2)



GP Training

Find good hyper-parameters $\boldsymbol{\theta}$ (kernel/mean function parameters ψ, noise variance σ_n^2)



- Place a prior $p(\boldsymbol{\theta})$ on hyper-parameters
- Posterior over hyper-parameters:

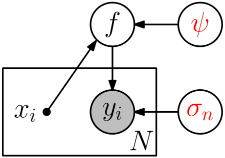
$$p(\boldsymbol{\theta}|\boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{\theta})\, p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})}{p(\boldsymbol{y}|\boldsymbol{X})}$$

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = \int p(\boldsymbol{y}|f, \boldsymbol{X}) p(f|\boldsymbol{X}, \boldsymbol{\theta}) \mathrm{d}f$$

- Posterior over hyper-parameters:

$$p(\boldsymbol{\theta}|\boldsymbol{X},\boldsymbol{y}) = \frac{p(\boldsymbol{\theta})\, p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})}{p(\boldsymbol{y}|\boldsymbol{X})}$$

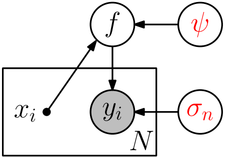
$$p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta}) = \int p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\boldsymbol{X})|\boldsymbol{\theta})\, \mathrm{d}f$$



- Posterior over hyper-parameters:

$$p(\boldsymbol{\theta}|\boldsymbol{X},\boldsymbol{y}) = \frac{p(\boldsymbol{\theta})\, p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})}{p(\boldsymbol{y}|\boldsymbol{X})}$$

$$p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta}) = \int p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\boldsymbol{X})|\boldsymbol{\theta})\, \mathrm{d}f$$



- Choose hyper-parameters $\boldsymbol{\theta}^*$, such that

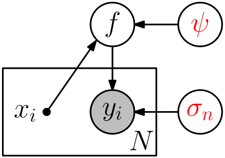
$$\boldsymbol{\theta}^* \in \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}) + \log p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})$$

Gaussian Process Training: Hyper-Parameters

- Posterior over hyper-parameters:

$$p(\boldsymbol{\theta}|\boldsymbol{X},\boldsymbol{y}) = \frac{p(\boldsymbol{\theta})\, p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})}{p(\boldsymbol{y}|\boldsymbol{X})}$$

$$p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta}) = \int p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\boldsymbol{X})|\boldsymbol{\theta})\, \mathrm{d}f$$



- Choose hyper-parameters $\boldsymbol{\theta}^*$, such that

$$\boldsymbol{\theta}^* \in \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}) + \log p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})$$

▶▶ Maximize marginal likelihood if $p(\boldsymbol{\theta}) = \mathcal{U}$ (uniform prior)

Training via Marginal Likelihood Maximization

GP Training

Maximize the evidence/marginal likelihood (probability of the data given the hyper-parameters, where the unwieldy f has been integrated out) ⏩ Also called Maximum Likelihood Type-II

Training via Marginal Likelihood Maximization

GP Training

Maximize the evidence/marginal likelihood (probability of the data given the hyper-parameters, where the unwieldy f has been integrated out) ⏩ Also called Maximum Likelihood Type-II

Marginal likelihood (with a prior mean function $m(\cdot) \equiv 0$):

$$
\begin{aligned}
p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) &= \int p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\boldsymbol{X})|\boldsymbol{\theta})\, \mathrm{d}f \\
&= \int \mathcal{N}\big(\boldsymbol{y} \,|\, f(\boldsymbol{X}),\, \sigma_n^2 \boldsymbol{I}\big)\, \mathcal{N}\big(f(\boldsymbol{X}) \,|\, \boldsymbol{0},\, \boldsymbol{K}\big)\, \mathrm{d}f \\
&= \mathcal{N}\big(\boldsymbol{y} \,|\, \boldsymbol{0},\, \boldsymbol{K} + \sigma_n^2 \boldsymbol{I}\big)
\end{aligned}
$$

Training via Marginal Likelihood Maximization

GP Training

Maximize the evidence/marginal likelihood (probability of the data given the hyper-parameters, where the unwieldy f has been integrated out) ▶▶ Also called Maximum Likelihood Type-II

Marginal likelihood (with a prior mean function $m(\cdot) \equiv 0$):

$$
\begin{aligned}
p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta}) &= \int p(\boldsymbol{y}|f(\boldsymbol{X}))\, p(f(\boldsymbol{X})|\boldsymbol{\theta})\,\mathrm{d}f \\
&= \int \mathcal{N}\big(\boldsymbol{y}\,|\,f(\boldsymbol{X}),\,\sigma_n^2\boldsymbol{I}\big)\,\mathcal{N}\big(f(\boldsymbol{X})\,|\,\boldsymbol{0},\,\boldsymbol{K}\big)\,\mathrm{d}f \\
&= \mathcal{N}\big(\boldsymbol{y}\,|\,\boldsymbol{0},\,\boldsymbol{K}+\sigma_n^2\boldsymbol{I}\big)
\end{aligned}
$$

Learning the GP hyper-parameters:

$$
\boldsymbol{\theta}^* \in \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})
$$

- Log-marginal likelihood:

$$\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = -\tfrac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\boldsymbol{y} - \tfrac{1}{2}\log|\boldsymbol{K}_{\boldsymbol{\theta}}| + \text{const}$$

$$\boldsymbol{K}_{\boldsymbol{\theta}} := \boldsymbol{K} + \sigma_n^2 \boldsymbol{I}$$

Gradient-based Optimization

- Log-marginal likelihood:

$$\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = -\tfrac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\boldsymbol{y} - \tfrac{1}{2}\log|\boldsymbol{K}_{\boldsymbol{\theta}}| + \text{const}$$

$$\boldsymbol{K}_{\boldsymbol{\theta}} := \boldsymbol{K} + \sigma_n^2 \boldsymbol{I}$$

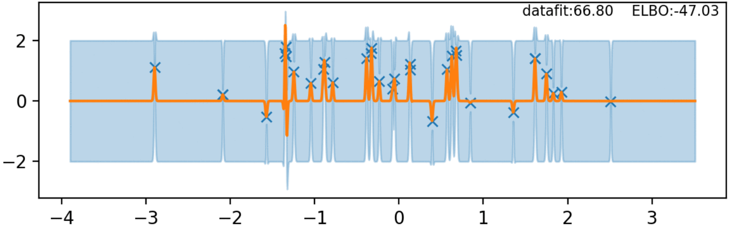
- Gradient-based optimization to get hyper-parameters $\boldsymbol{\theta}^*$:

$$\frac{\partial \log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})}{\partial \theta_i} = \tfrac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\frac{\partial \boldsymbol{K}_{\boldsymbol{\theta}}}{\partial \theta_i}\boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\boldsymbol{y} - \tfrac{1}{2}\text{tr}\big(\boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\frac{\partial \boldsymbol{K}_{\boldsymbol{\theta}}}{\partial \theta_i}\big)$$

$$= \tfrac{1}{2}\text{tr}\big((\boldsymbol{\alpha}\boldsymbol{\alpha}^\top - \boldsymbol{K}_{\boldsymbol{\theta}}^{-1})\frac{\partial \boldsymbol{K}_{\boldsymbol{\theta}}}{\partial \theta_i}\big),$$

$$\boldsymbol{\alpha} := \boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\boldsymbol{y}$$

Marginal Likelihood Illustration[1]



- "ELBO" refers to the log-marginal likelihood
- Data-fit term gets worse, but marginal likelihood increases

[1] Thanks to Mark van der Wilk

Inspecting the Marginal Likelihood

Log-marginal likelihood:

$$\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = -\tfrac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\boldsymbol{y} - \tfrac{1}{2}\log|\boldsymbol{K}_{\boldsymbol{\theta}}| + \text{const}\,, \quad \boldsymbol{K}_{\boldsymbol{\theta}} := \boldsymbol{K} + \sigma_n^2 \boldsymbol{I}$$

Inspecting the Marginal Likelihood

Log-marginal likelihood:

$$\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = -\tfrac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\boldsymbol{y} - \tfrac{1}{2}\log|\boldsymbol{K}_{\boldsymbol{\theta}}| + \text{const}\,, \quad \boldsymbol{K}_{\boldsymbol{\theta}} := \boldsymbol{K} + \sigma_n^2\boldsymbol{I}$$

- Quadratic term measures whether observation $\boldsymbol{y}$ is within the variation allowed by the prior

Inspecting the Marginal Likelihood

Log-marginal likelihood:

$$\log p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta}) = -\tfrac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\boldsymbol{y} - \tfrac{1}{2}\log|\boldsymbol{K}_{\boldsymbol{\theta}}| + \text{const}\,, \quad \boldsymbol{K}_{\boldsymbol{\theta}} := \boldsymbol{K} + \sigma_n^2\boldsymbol{I}$$

- Quadratic term measures whether observation $\boldsymbol{y}$ is within the variation allowed by the prior
- Determinant is the product of the variances of the prior (volume of the prior) ▸▸ Volume $\approx$ richness of model class

Inspecting the Marginal Likelihood

Log-marginal likelihood:

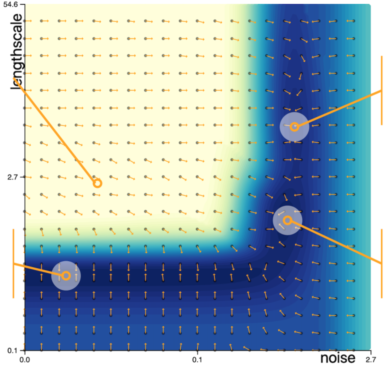
$$\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = -\tfrac{1}{2}\boldsymbol{y}^\top \boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\boldsymbol{y} - \tfrac{1}{2}\log|\boldsymbol{K}_{\boldsymbol{\theta}}| + \text{const}\,, \quad \boldsymbol{K}_{\boldsymbol{\theta}} := \boldsymbol{K} + \sigma_n^2\boldsymbol{I}$$

- Quadratic term measures whether observation $\boldsymbol{y}$ is within the variation allowed by the prior
- Determinant is the product of the variances of the prior (volume of the prior) ▸▸ Volume $\approx$ richness of model class

Marginal likelihood

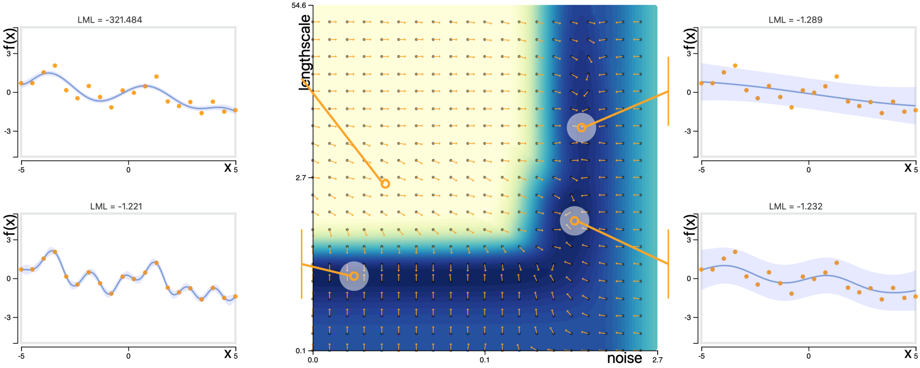
▸▸ Automatic trade-off between data fit and model complexity

Marginal Likelihood Surface



- Several plausible hyper-parameters (local optima)
- What do you expect to happen in each local optimum?

Marginal Likelihood Surface



- Several plausible hyper-parameters (local optima)
- What do you expect to happen in each local optimum?

https://drafts.distill.pub/gp/

Marginal Likelihood and Parameter Learning

- The marginal likelihood is non-convex

Marginal Likelihood and Parameter Learning

- The marginal likelihood is non-convex
- Especially in the very-small-data regime, a GP can end up in three different situations when optimizing the hyper-parameters:

Marginal Likelihood and Parameter Learning

- The marginal likelihood is non-convex
- Especially in the very-small-data regime, a GP can end up in three different situations when optimizing the hyper-parameters:
  - Short length-scales, low noise (highly nonlinear mean function with little noise)

Marginal Likelihood and Parameter Learning

- The marginal likelihood is non-convex
- Especially in the very-small-data regime, a GP can end up in three different situations when optimizing the hyper-parameters:
  - Short length-scales, low noise (highly nonlinear mean function with little noise)
  - Long length-scales, high noise (everything is considered noise)

Marginal Likelihood and Parameter Learning

- The marginal likelihood is non-convex
- Especially in the very-small-data regime, a GP can end up in three different situations when optimizing the hyper-parameters:
  - Short length-scales, low noise (highly nonlinear mean function with little noise)
  - Long length-scales, high noise (everything is considered noise)
  - Hybrid

Marginal Likelihood and Parameter Learning

- The marginal likelihood is non-convex
- Especially in the very-small-data regime, a GP can end up in three different situations when optimizing the hyper-parameters:
  - Short length-scales, low noise (highly nonlinear mean function with little noise)
  - Long length-scales, high noise (everything is considered noise)
  - Hybrid
- Re-start hyper-parameter optimization from random initialization to mitigate the problem

Marginal Likelihood and Parameter Learning

- The marginal likelihood is non-convex
- Especially in the very-small-data regime, a GP can end up in three different situations when optimizing the hyper-parameters:
  - Short length-scales, low noise (highly nonlinear mean function with little noise)
  - Long length-scales, high noise (everything is considered noise)
  - Hybrid
- Re-start hyper-parameter optimization from random initialization to mitigate the problem
- With increasing data set size the GP typically ends up in the "hybrid" mode. Other modes are unlikely.

Marginal Likelihood and Parameter Learning

- The marginal likelihood is non-convex
- Especially in the very-small-data regime, a GP can end up in three different situations when optimizing the hyper-parameters:
  - Short length-scales, low noise (highly nonlinear mean function with little noise)
  - Long length-scales, high noise (everything is considered noise)
  - Hybrid
- Re-start hyper-parameter optimization from random initialization to mitigate the problem
- With increasing data set size the GP typically ends up in the “hybrid” mode. Other modes are unlikely.
- Ideally, we would integrate the hyper-parameters out
  No closed-form solution ⏩ Markov chain Monte Carlo

Why Does the Marginal Likelihood Work?

- Overall goal: Good generalization performance on unseen test data

Why Does the Marginal Likelihood Work?

- Overall goal: Good generalization performance on unseen test data
- Minimizing training error is not a good idea (e.g., maximum likelihood) ⏩ Overfitting
- Just adding uncertainty does not help either if the model is wrong, but it makes predictions more cautious

Why Does the Marginal Likelihood Work?

- Overall goal: Good generalization performance on unseen test data
- Minimizing training error is not a good idea (e.g., maximum likelihood) ⏩ Overfitting
- Just adding uncertainty does not help either if the model is wrong, but it makes predictions more cautious
- Marginal likelihood seems to find a good balance between fitting the data and finding a simple model (Occam's razor)

Why does the marginal likelihood lead to models that generalize well?

Marginal Likelihood: Incremental Prediction

- "Probability of the training data" given the parameters
- General factorization (ignoring inputs $\boldsymbol{X}$):

$$p(\boldsymbol{y}|\boldsymbol{\theta}) = p(y_1, \dots, y_N|\boldsymbol{\theta})$$

Marginal Likelihood: Incremental Prediction

- "Probability of the training data" given the parameters
- General factorization (ignoring inputs $\boldsymbol{X}$):

$$
\begin{aligned}
p(\boldsymbol{y}|\boldsymbol{\theta}) &= p(y_1, \dots, y_N|\boldsymbol{\theta}) \\
&= p(y_1|\boldsymbol{\theta})p(y_2|y_1, \boldsymbol{\theta})p(y_3|y_1, y_2, \boldsymbol{\theta}) \cdot \ldots \cdot p(y_N|y_1, \dots, y_{N-1}, \boldsymbol{\theta})
\end{aligned}
$$

Marginal Likelihood: Incremental Prediction

- "Probability of the training data" given the parameters
- General factorization (ignoring inputs $\boldsymbol{X}$):

$$
\begin{aligned}
p(\boldsymbol{y}|\boldsymbol{\theta}) &= p(y_1, \ldots, y_N|\boldsymbol{\theta}) \\
&= p(y_1|\boldsymbol{\theta})p(y_2|y_1, \boldsymbol{\theta})p(y_3|y_1, y_2, \boldsymbol{\theta}) \cdot \ldots \cdot p(y_N|y_1, \ldots, y_{N-1}, \boldsymbol{\theta}) \\
&= p(y_1|\boldsymbol{\theta}) \prod_{n=2}^{N} p(y_n|y_1, \ldots, y_{n-1}, \boldsymbol{\theta})
\end{aligned}
$$

$$p(\boldsymbol{y}|\boldsymbol{\theta}) = p(y_1|\boldsymbol{\theta}) \prod_{n=2}^{N} p(y_n|y_1, \ldots, y_{n-1}, \boldsymbol{\theta})$$

Marginal Likelihood: Incremental Prediction (2)

$$p(\boldsymbol{y}|\boldsymbol{\theta}) = p(y_1|\boldsymbol{\theta}) \prod_{n=2}^{N} p(y_n|y_1, \ldots, y_{n-1}, \boldsymbol{\theta})$$

- If we think of this as a sequence model (where data arrives sequentially), the marginal likelihood predicts the nth training observation given all "previous" observations

Marginal Likelihood: Incremental Prediction (2)

$$p(\boldsymbol{y}|\boldsymbol{\theta}) = p(y_1|\boldsymbol{\theta}) \prod_{n=2}^{N} p(y_n|y_1, \ldots, y_{n-1}, \boldsymbol{\theta})$$

- If we think of this as a sequence model (where data arrives sequentially), the marginal likelihood predicts the nth training observation given all “previous” observations
- Predict training data y_n that has not been accounted for (we only condition on $y_1, \ldots, y_{n-1}$) ▸▸ Treat next data point as test data

Marginal Likelihood: Incremental Prediction (2)

$$p(\boldsymbol{y}|\boldsymbol{\theta}) = p(y_1|\boldsymbol{\theta}) \prod_{n=2}^{N} p(y_n|y_1, \dots, y_{n-1}, \boldsymbol{\theta})$$

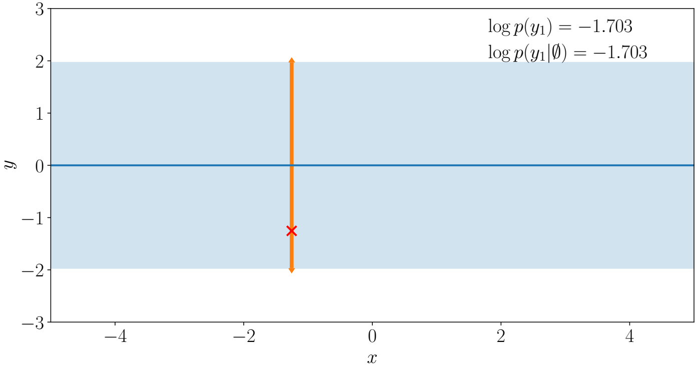
- If we think of this as a sequence model (where data arrives sequentially), the marginal likelihood predicts the nth training observation given all "previous" observations
- Predict training data y_n that has not been accounted for (we only condition on $y_1, \dots, y_{n-1}$) ▸▸ Treat next data point as test data
- Intuition: If it continuously predicted well on all N previous points, it probably will do well next time

Marginal Likelihood: Incremental Prediction (2)

$$p(\boldsymbol{y}|\boldsymbol{\theta}) = p(y_1|\boldsymbol{\theta}) \prod_{n=2}^{N} p(y_n|y_1, \ldots, y_{n-1}, \boldsymbol{\theta})$$

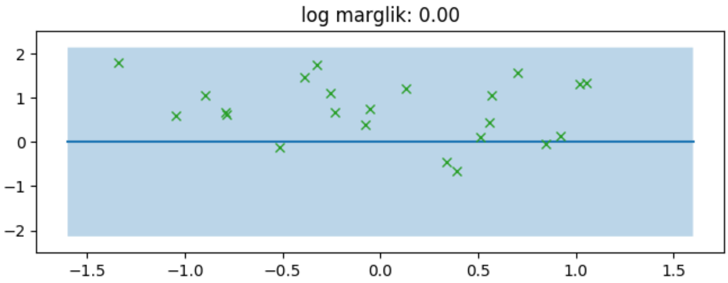
- If we think of this as a sequence model (where data arrives sequentially), the marginal likelihood predicts the nth training observation given all “previous” observations
- Predict training data y_n that has not been accounted for (we only condition on $y_1, \ldots, y_{n-1}$) ▶▶ Treat next data point as test data
- Intuition: If it continuously predicted well on all N previous points, it probably will do well next time
  ▶▶ Proxy for generalization error on unseen test data

Marginal Likelihood: Illustration



$$p(\boldsymbol{y}|\boldsymbol{\theta}) = p(y_1, \dots, y_N|\boldsymbol{\theta}) = p(y_1|\boldsymbol{\theta}) \prod_{n=2}^{N} p(y_n|y_1, \dots, y_{n-1}, \boldsymbol{\theta})$$

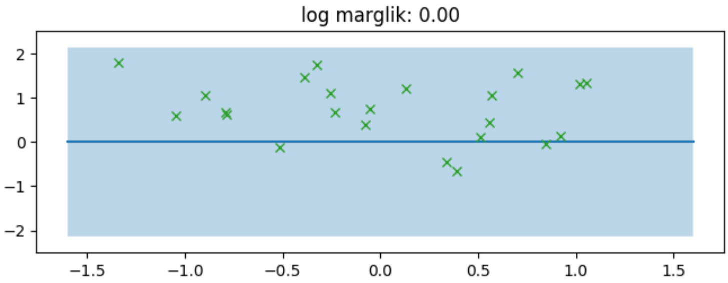
Marginal Likelihood Computation in Action[2]



- Short length-scale

[2] Thanks to Mark van der Wilk

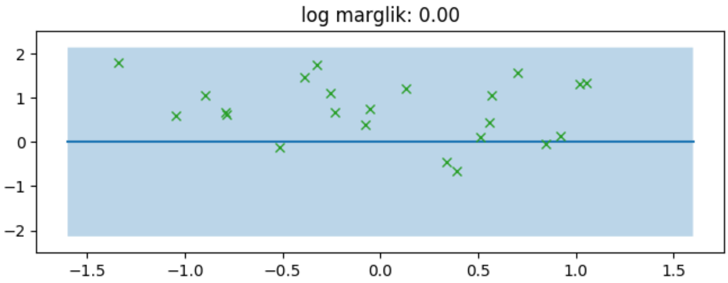
Marginal Likelihood Computation in Action[3]



- Long length-scale

[3]Thanks to Mark van der Wilk

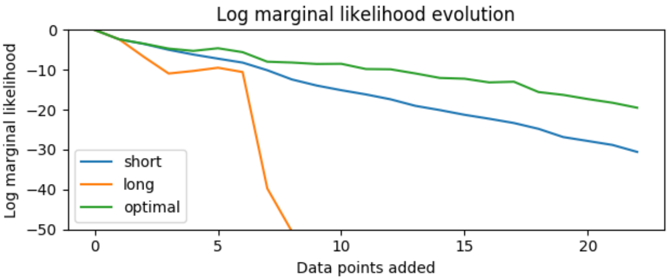
Marginal Likelihood Computation in Action[4]



- Optimal length-scale

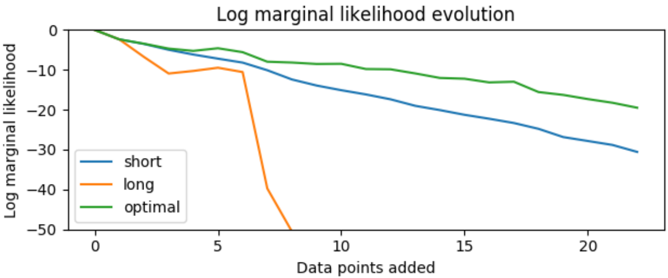
[4]Thanks to Mark van der Wilk

Marginal Likelihood Evolution



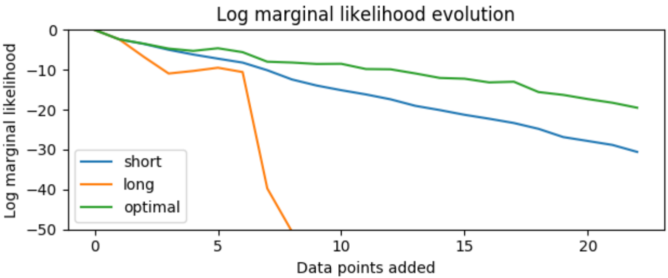
- Short lengthscale: consistently overestimates variance
  - No high density, even with observations inside the error bars

Marginal Likelihood Evolution



- Short lengthscale: consistently overestimates variance
  - No high density, even with observations inside the error bars
- Long lengthscale: consistently underestimates variance
  - Low density because observations are outside the error bars

Marginal Likelihood Evolution



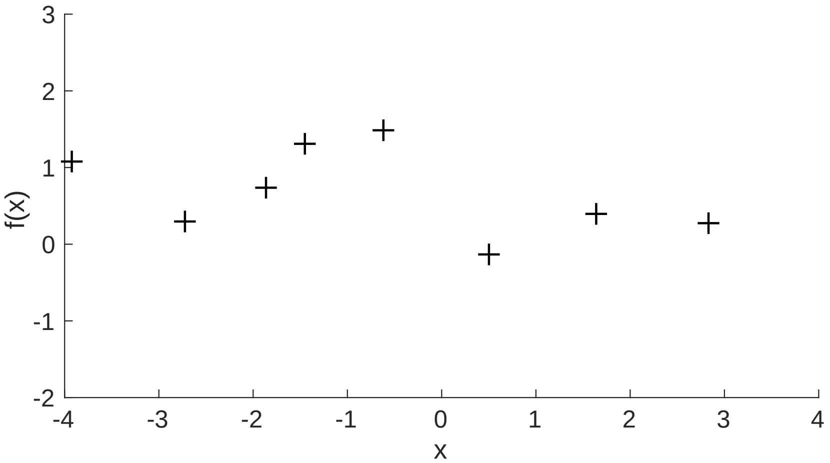
- Short lengthscale: consistently overestimates variance
  - No high density, even with observations inside the error bars
- Long lengthscale: consistently underestimates variance
  - Low density because observations are outside the error bars
- Optimal lengthscale: trades off both behaviors reasonably well

- Assume we have a finite set of models M_i, each one specifying a mean function m_i and a kernel k_i. How do we find the best one?

Model Selection—Mean Function and Kernel

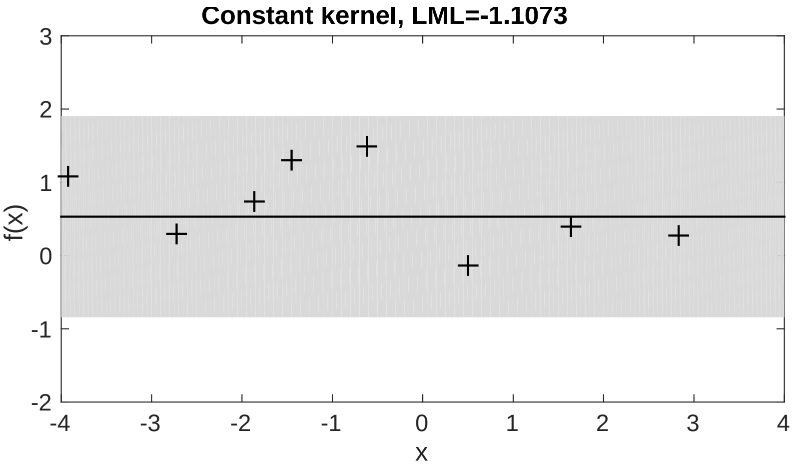
- Assume we have a finite set of models M_i, each one specifying a mean function m_i and a kernel k_i. How do we find the best one?
- Some options:
  - Cross validation
  - Bayesian Information Criterion, Akaike Information Criterion
  - Compare marginal likelihood values (assuming a uniform prior on the set of models)

Example



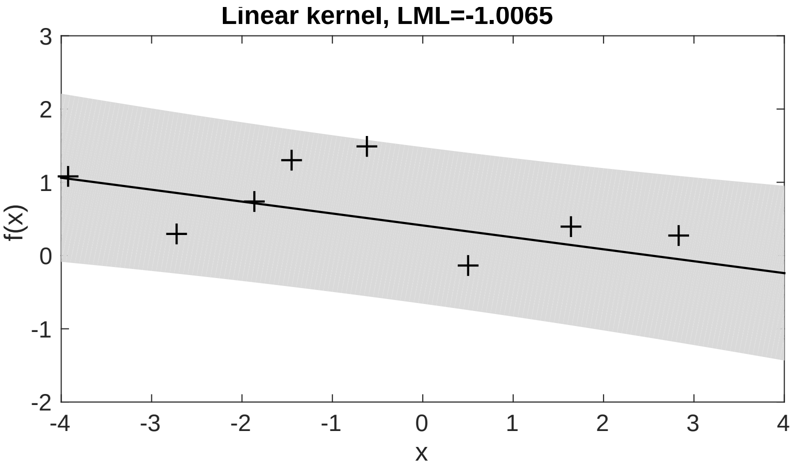
- Four different kernels (mean function fixed to $m \equiv 0$)
- MAP hyper-parameters for each kernel
- Log-marginal likelihood values for each (optimized) model

Example



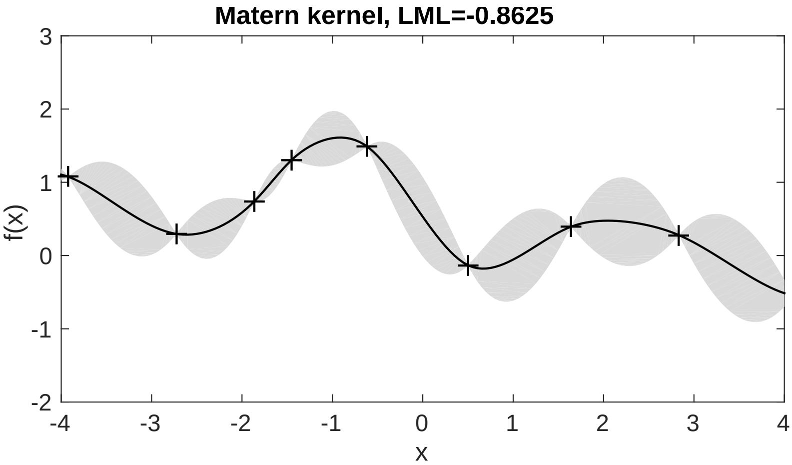
- Four different kernels (mean function fixed to $m \equiv 0$)
- MAP hyper-parameters for each kernel
- Log-marginal likelihood values for each (optimized) model

Example



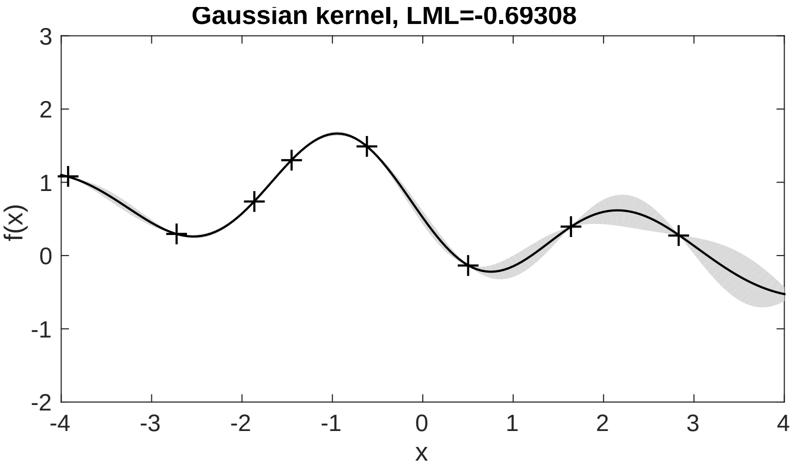
- Four different kernels (mean function fixed to $m \equiv 0$)
- MAP hyper-parameters for each kernel
- Log-marginal likelihood values for each (optimized) model

Example

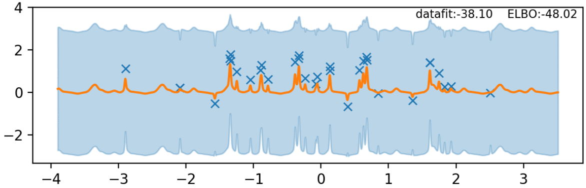


- Four different kernels (mean function fixed to $m \equiv 0$)
- MAP hyper-parameters for each kernel
- Log-marginal likelihood values for each (optimized) model

Example



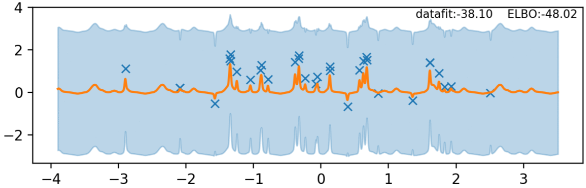
- Four different kernels (mean function fixed to $m \equiv 0$)
- MAP hyper-parameters for each kernel
- Log-marginal likelihood values for each (optimized) model



- Prior: $f(\boldsymbol{x}) = \theta_s f_{\text{smooth}}(\boldsymbol{x}) + \theta_p f_{\text{periodic}}(\boldsymbol{x})$, with smooth and periodic GP priors, respectively.

[5] Thanks to Mark van der Wilk

GP Training: Illustration[5]



- Prior: $f(\boldsymbol{x}) = \theta_s f_{\text{smooth}}(\boldsymbol{x}) + \theta_p f_{\text{periodic}}(\boldsymbol{x})$, with smooth and periodic GP priors, respectively.
- Amount of periodicity vs. smoothness is automatically chosen by selecting hyper-parameters θ_s, θ_p.
- Marginal likelihood learns how to generalize, not just to fit the data

[5] Thanks to Mark van der Wilk

Limitations and Guidelines

Computational and memory complexity

Training set size: N

- Training scales in $\mathcal{O}(N^3)$
- Prediction (variances) scales in $\mathcal{O}(N^2)$
- Memory requirement: $\mathcal{O}(ND + N^2)$

▶▶ **Practical limit** $N \approx 10,000$

Limitations of Gaussian Processes

Computational and memory complexity

Training set size: N

- Training scales in $\mathcal{O}(N^3)$
- Prediction (variances) scales in $\mathcal{O}(N^2)$
- Memory requirement: $\mathcal{O}(ND + N^2)$

Practical limit $N \approx 10,000$

Some solution approaches:

- Sparse GPs with inducing variables (e.g., Snelson & Ghahramani, 2006; Quiñonero-Candela & Rasmussen, 2005; Titsias 2009; Hensman et al., 2013; Matthews et al., 2016)
- Combination of local GP expert models (e.g., Tresp 2000; Cao & Fleet 2014; Deisenroth & Ng, 2015)
- Variational Fourier features (Hensman et al., 2018)

Tips and Tricks for Practitioners

- To set initial hyper-parameters, use domain knowledge.

⏩ `https://drafts.distill.pub/gp`

Tips and Tricks for Practitioners

- To set initial hyper-parameters, use domain knowledge.
- Standardize input data and set initial length-scales ℓ to ≈ 0.5.

https://drafts.distill.pub/gp

Tips and Tricks for Practitioners

- To set initial hyper-parameters, use domain knowledge.
- Standardize input data and set initial length-scales ℓ to ≈ 0.5.
- Standardize targets y and set initial signal variance to $\sigma_f \approx 1$.

▶▶ https://drafts.distill.pub/gp

Tips and Tricks for Practitioners

- To set initial hyper-parameters, use domain knowledge.
- Standardize input data and set initial length-scales ℓ to ≈ 0.5.
- Standardize targets y and set initial signal variance to $\sigma_f \approx 1$.
- Often useful: Set initial noise level relatively high (e.g., $\sigma_n \approx 0.5 \times \sigma_f$ amplitude), even if you think your data have low noise. The optimization surface for your other parameters will be easier to move in.

▶▶ `https://drafts.distill.pub/gp`

Tips and Tricks for Practitioners

- To set initial hyper-parameters, use domain knowledge.
- Standardize input data and set initial length-scales ℓ to ≈ 0.5.
- Standardize targets y and set initial signal variance to $\sigma_f \approx 1$.
- Often useful: Set initial noise level relatively high (e.g., $\sigma_n \approx 0.5 \times \sigma_f$ amplitude), even if you think your data have low noise. The optimization surface for your other parameters will be easier to move in.
- When optimizing hyper-parameters, try random restarts or other tricks to avoid local optima are advised.

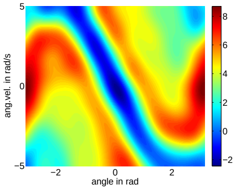
▶▶ `https://drafts.distill.pub/gp`

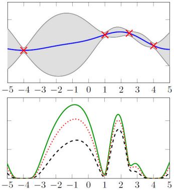
- To set initial hyper-parameters, use domain knowledge.
- Standardize input data and set initial length-scales ℓ to ≈ 0.5.
- Standardize targets y and set initial signal variance to $\sigma_f \approx 1$.
- Often useful: Set initial noise level relatively high (e.g., $\sigma_n \approx 0.5 \times \sigma_f$ amplitude), even if you think your data have low noise. The optimization surface for your other parameters will be easier to move in.
- When optimizing hyper-parameters, try random restarts or other tricks to avoid local optima are advised.
- Mitigate the problem of numerical instability (Cholesky decomposition of $\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}$) by penalizing high signal-to-noise ratios σ_f / σ_n

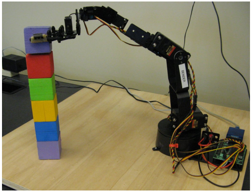
▶▶ https://drafts.distill.pub/gp

Application Areas

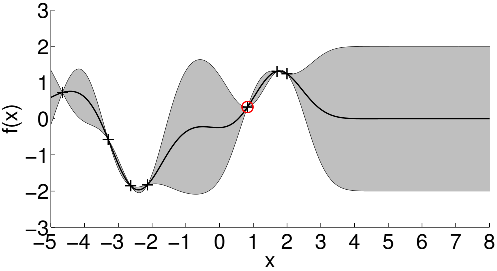
Application Areas

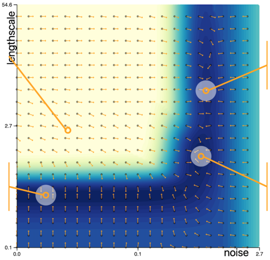






- Reinforcement learning and robotics
  - Model value functions and/or dynamics with GPs
- Bayesian optimization (Experimental Design)
  - Model unknown utility functions with GPs
- Geostatistics
  - Spatial modeling (e.g., landscapes, resources)
- Sensor networks
- Time-series modeling and forecasting





- Gaussian processes are the gold-standard for regression
- Closely related to Bayesian linear regression
- Computations boil down to manipulating multivariate Gaussian distributions
- Marginal likelihood objective automatically trades off data fit and model complexity

References I

[1] G. Bertone, M. P. Deisenroth, J. S. Kim, S. Liem, R. R. de Austri, and M. Welling. Accelerating the BSM Interpretation of LHC Data with Machine Learning. arXiv preprint arXiv:1611.02704, 2016.

[2] R. Calandra, J. Peters, C. E. Rasmussen, and M. P. Deisenroth. Manifold Gaussian Processes for Regression. In *Proceedings of the International Joint Conference on Neural Networks*, 2016.

[3] Y. Cao and D. J. Fleet. Generalized Product of Experts for Automatic and Principled Fusion of Gaussian Process Predictions. `http://arxiv.org/abs/1410.7827`, 2014.

[4] N. A. C. Cressie. *Statistics for Spatial Data*. Wiley-Interscience, 1993.

[5] M. Cutler and J. P. How. Efficient Reinforcement Learning for Robots using Informative Simulated Priors. In *Proceedings of the International Conference on Robotics and Automation*, 2015.

[6] M. P. Deisenroth and J. W. Ng. Distributed Gaussian Processes. In *Proceedings of the International Conference on Machine Learning*, 2015.

[7] M. P. Deisenroth, C. E. Rasmussen, and D. Fox. Learning to Control a Low-Cost Manipulator using Data-Efficient Reinforcement Learning. In *Proceedings of Robotics: Science and Systems*, 2011.

[8] M. P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian Process Dynamic Programming. *Neurocomputing*, 72(7–9):1508–1524, Mar. 2009.

[9] M. P. Deisenroth, R. Turner, M. Huber, U. D. Hanebeck, and C. E. Rasmussen. Robust Filtering and Smoothing with Gaussian Processes. *IEEE Transactions on Automatic Control*, 57(7):1865–1871, 2012.

[10] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen. Bayesian Inference and Learning in Gaussian Process State-Space Models with Particle MCMC. In *Advances in Neural Information Processing Systems*. 2013.

[11] N. HajiGhassemi and M. P. Deisenroth. Approximate Inference for Long-Term Forecasting with Periodic Gaussian Processes. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2014.

[12] J. Hensman, N. Durrande, and A. Solin. Variational Fourier Features for Gaussian Processes. *Journal of Machine Learning Research*, pages 1–52, 2018.

[13] J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian Processes for Big Data. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2013.

[14] A. Krause, A. Singh, and C. Guestrin. Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *Journal of Machine Learning Research*, 9:235–284, Feb. 2008.

[15] M. C. H. Lee, H. Salimbeni, M. P. Deisenroth, and B. Glocker. Patch Kernels for Gaussian Processes in High-Dimensional Imaging Problems. In *NIPS Workshop on Practical Bayesian Nonparametrics*, 2016.

[16] J. R. Lloyd, D. Duvenaud, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Automatic Construction and Natural-Language Description of Nonparametric Regression Models. In *AAAI Conference on Artificial Intelligence*, pages 1–11, 2014.

[17] D. J. C. MacKay. Introduction to Gaussian Processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, volume 168, pages 133–165. Springer, 1998.

[18] A. G. d. G. Matthews, J. Hensman, R. Turner, and Z. Ghahramani. On Sparse Variational Methods and the Kullback-Leibler Divergence between Stochastic Processes. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2016.

[19] M. A. Osborne, S. J. Roberts, A. Rogers, S. D. Ramchurn, and N. R. Jennings. Towards Real-Time Information Processing of Sensor Network Data Using Computationally Efficient Multi-output Gaussian Processes. In *Proceedings of the International Conference on Information Processing in Sensor Networks*, pages 109–120. IEEE Computer Society, 2008.

[20] J. Quiñonero-Candela and C. E. Rasmussen. A Unifying View of Sparse Approximate Gaussian Process Regression. *Journal of Machine Learning Research*, 6(2):1939–1960, 2005.

[21] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[22] S. Roberts, M. A. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain. Gaussian Processes for Time Series Modelling. *Philosophical Transactions of the Royal Society (Part A)*, 371(1984), Feb. 2013.

References III

[23] B. Schölkopf and A. J. Smola. *Learning with Kernels—Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, USA, 2002.

[24] E. Snelson and Z. Ghahramani. Sparse Gaussian Processes using Pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1257–1264. The MIT Press, Cambridge, MA, USA, 2006.

[25] M. K. Titsias. Variational Learning of Inducing Variables in Sparse Gaussian Processes. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2009.

[26] V. Tresp. A Bayesian Committee Machine. *Neural Computation*, 12(11):2719–2741, 2000.

[27] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing. Deep Kernel Learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2016.