UNIVERSIDADE FEDERAL DO CEARÁ

DEPARTAMENTO DE COMPUTAÇÃO

The Art of Modeling with Gaussian Processes

César Lincoln Cavalcante Mattos

Department of Computer Science
Federal University of Ceará
Fortaleza, Ceará, Brazil
cesarlincoln@dc.ufc.br

2023

Table of Contents

The Art of Gaussian Processes: Classical and Contemporary

Part of this course was taken from the NeurIPS 2021 tutorial jointly presented with Prof. Felipe Tobar and available at `https://neurips.cc/virtual/2021/tutorial/21890`.

César Lincoln C. Mattos is an Associate Professor at the Department of Computer Science, at Federal University of Ceará (UFC), Brazil. He is also an associate researcher at the Logics and Artificial Intelligence Group (LOGIA). He has research interests in (deep) probabilistic learning, approximate inference, system identification and anomaly detection.

Felipe Tobar is an Associate Professor at the Initiative for Data & Artificial Intelligence, Universidad de Chile, and the Coordinator of the Master of Data Science at the same University. He is also an Associate Researcher at the Center for Mathematical Modeling and the Advanced Centre for Electrical and Electronic Engineering. He teaches courses on Probability, Statistics and Machine Learning, and his research interests include time series, Bayesian inference, computational optimal transport and the societal impacts of machine learning.

Table of Contents

Learning from data

- **Data**: partial observations of the world.
- **Model**: links observations and latent (unobserved) objects.
- **Uncertainty**: data is incomplete and noisy.
- **Learning**: estimate unknowns from known quantities.

(taken with permission from Thoughtfulnz on Twitter)

Probability **Statistics**

General $<$ – – – – – – – – – – – – – – – $>$ Particular

Population $<$ – – – – – – – – – – – – – – – $>$ Sample

Model $<$ – – – – – – – – – – – – – – – $>$ Data

The statistical model

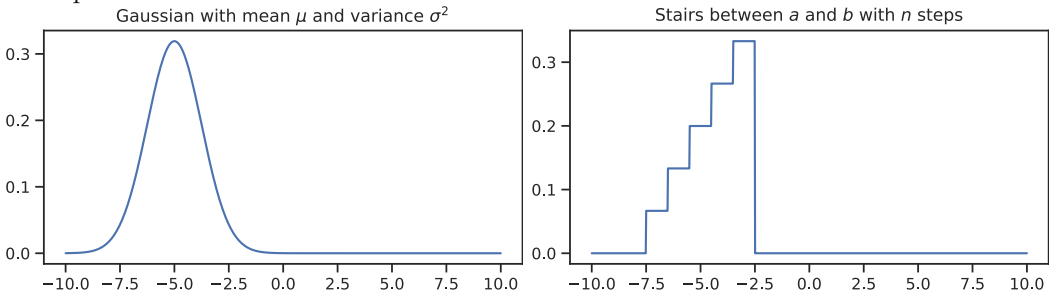
Notation

- **Sample space** $\mathcal{X}$: where the data lives.
- **Statistical model** $\mathcal{M}$: the set of probability distributions on $\mathcal{X}$ indexed by a set $\mathcal{T}$:

$$\mathcal{M} = \{P_\theta \mid \theta \in \mathcal{T}\}. \tag{1}$$

- **Parameter** θ and **parameter space** $\mathcal{T}$.

Examples:



The statistical model

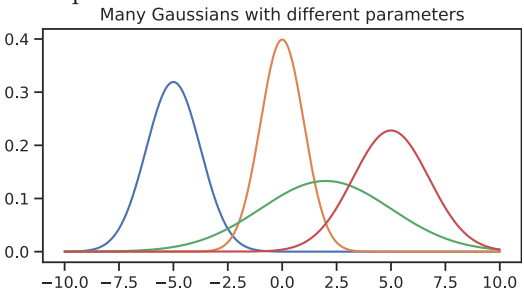
Notation

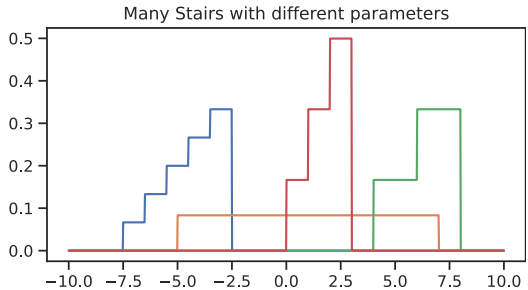
- **Sample space** $\mathcal{X}$: where the data lives.
- **Statistical model** $\mathcal{M}$: the set of probability distributions on $\mathcal{X}$ indexed by a set $\mathcal{T}$:

$$\mathcal{M} = \{P_\theta \mid \theta \in \mathcal{T}\}. \tag{1}$$

- **Parameter** θ and **parameter space** $\mathcal{T}$.

Examples:





Finding the parameters

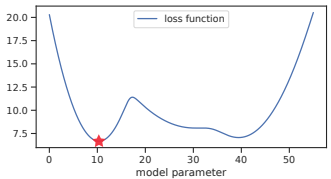
(and thus the model)

Alternative 1: best model

i) Define the model space

ii) Gather data

iii) Give each model a (fit) score:

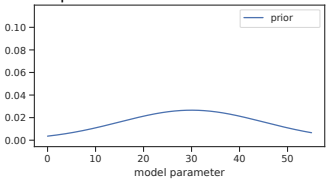


iv) Chose the model with the best score ★

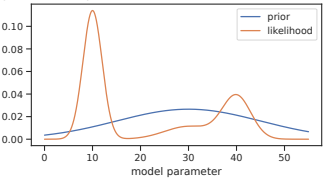
Predictions: "plug in" the chosen model into the predictor

Alternative 2: posterior belief

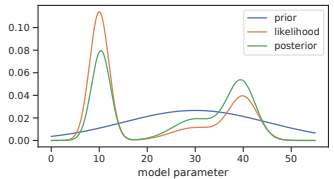
i) Define the model space and equip it with a prior belief



ii) Gather data and give each model a (likelihood) score



iii) Update your prior belief into a posterior one



Predictions: predict with each model and average predictions wrt to posterior belief

Bayesian inference: a one-slide cheat sheet

Model specification: Define some parameters as random variables (θ) and some as deterministic values (λ):

$$\lambda \in \Lambda, \quad \theta \sim p_\lambda(\theta) \quad \Rightarrow \quad X|\theta, \lambda \sim p_\lambda(x|\theta). \tag{2}$$

After observing some data ($\mathcal{D}$):

- **Set λ (model comparison):** e.g., by maximizing the marginal likelihood

$$p_\lambda(\mathcal{D}) = \int p_\lambda(\mathcal{D}|\theta) p_\lambda(\theta) \mathrm{d}\theta. \tag{3}$$

- **Bayesian update:** compute the posterior over θ

$$p_\lambda(\theta|\mathcal{D}) = \frac{p_\lambda(\mathcal{D}|\theta) p_\lambda(\theta)}{p_\lambda(\mathcal{D})}. \tag{4}$$

- **Prediction:** performed via posterior averages

$$p_\lambda(x|\mathcal{D}) = \int p_\lambda(x|\theta) p_\lambda(\theta|\mathcal{D}) \mathrm{d}\theta. \tag{5}$$

Important: The data is *summarized* in the posterior $p_\lambda(\theta|\mathcal{D})$ and is used for prediction.

Table of Contents

Bayesian linear regression

- Consider a linear regression model with Gaussian noise:

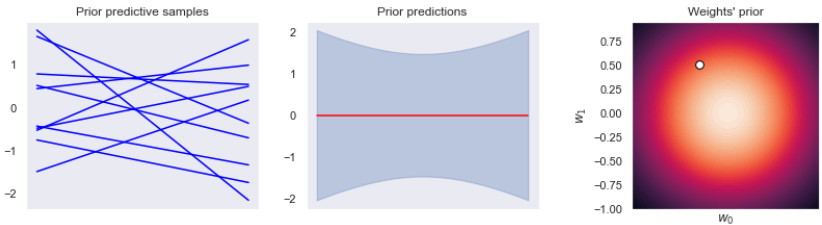
$$y = \boldsymbol{w}^\top \boldsymbol{x} + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

- The likelihood function is given by:

$$p(y|\boldsymbol{x}, \boldsymbol{w}, \sigma^2) = \mathcal{N}(y|\boldsymbol{w}^\top \boldsymbol{x}, \sigma^2).$$

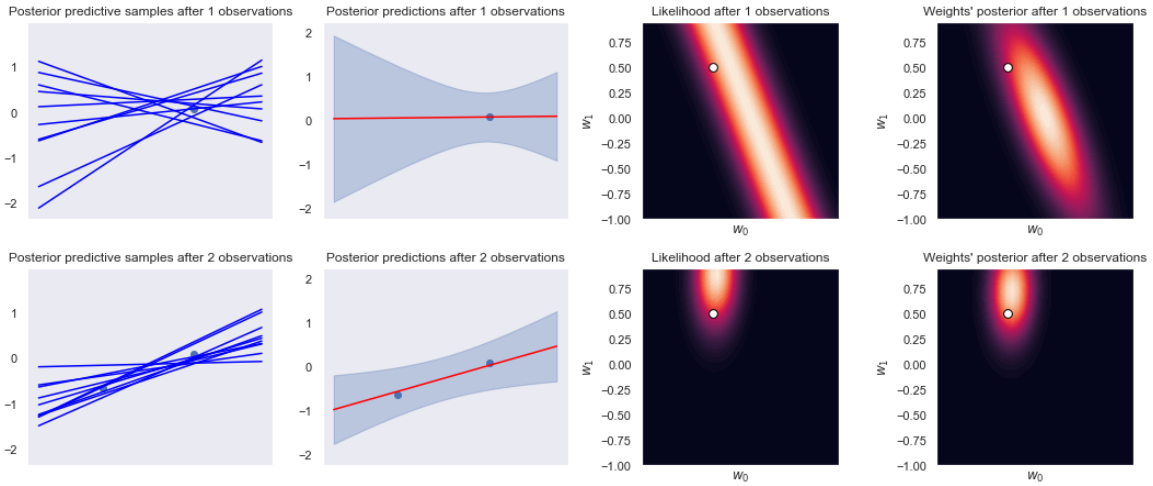
- The unknown parameter vector $\boldsymbol{w}$ has a prior distribution, e.g., a multivariate Gaussian:

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{m}_0, \boldsymbol{S}_0).$$



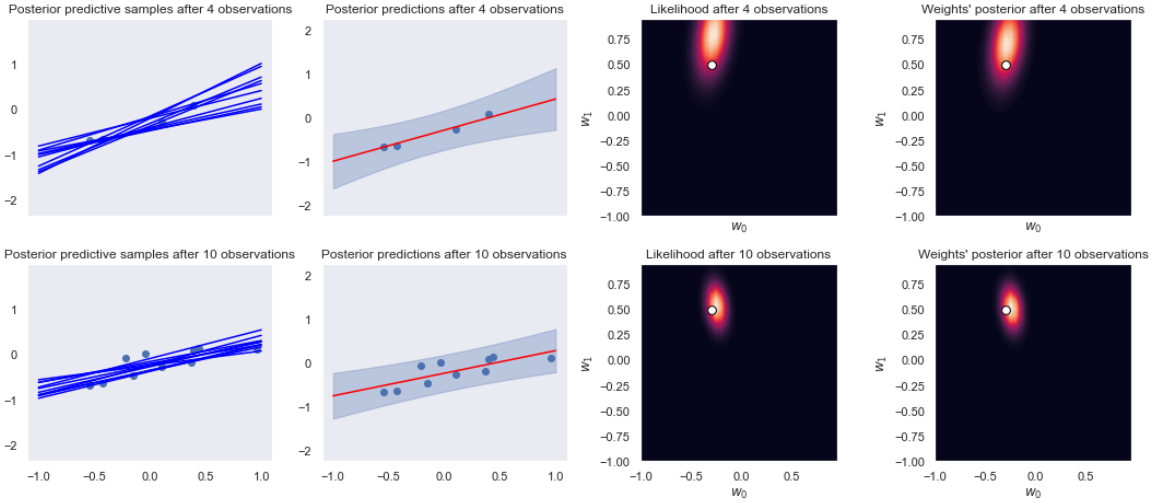
Bayesian linear regression

- After observing $\mathcal{D} = (\boldsymbol{X}, \boldsymbol{y})$, we calculate the posterior $p(\boldsymbol{w}|\mathcal{D})$ and perform prediction.



Bayesian linear regression

- After observing $\mathcal{D} = (\boldsymbol{X}, \boldsymbol{y})$, we calculate the posterior $p(\boldsymbol{w}|\mathcal{D})$ and perform prediction.

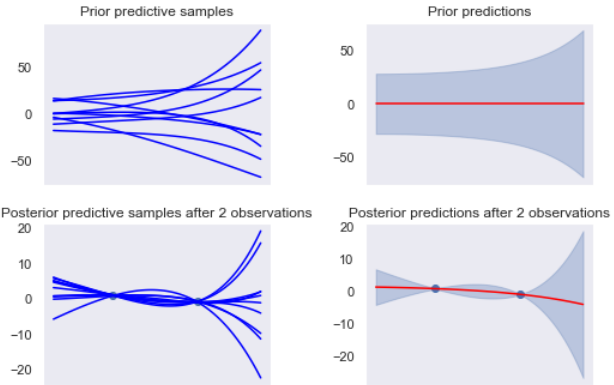


Bayesian polynomial regression

- Consider a 5-order polynomial regression model with Gaussian noise:

$$y = \boldsymbol{w}^\top \phi(x) + \epsilon, \quad \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{S}_0), \quad \epsilon \sim \mathcal{N}(0, \sigma^2),$$

where $\phi(x) = [1, x, x^2, x^3, x^4, x^5]^\top$.

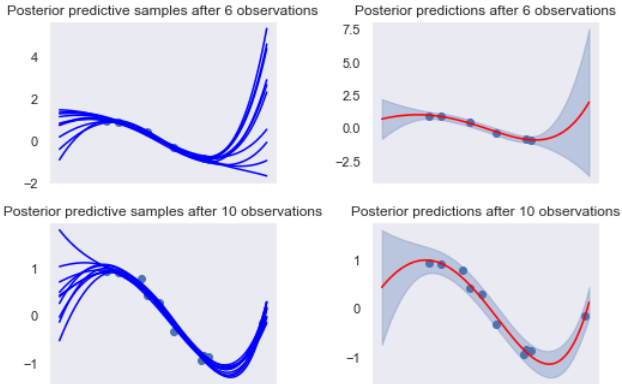


Bayesian polynomial regression

- Consider a 5-order polynomial regression model with Gaussian noise:

$$y = \boldsymbol{w}^\top \phi(x) + \epsilon, \quad \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{S}_0), \quad \epsilon \sim \mathcal{N}(0, \sigma^2),$$

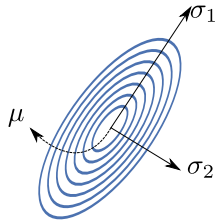
where $\phi(x) = [1, x, x^2, x^3, x^4, x^5]^\top$.

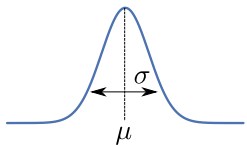


Why Gaussians?

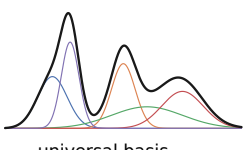
Many useful properties:

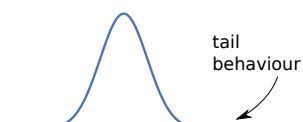
- central limit theorem;
- conjugacy;
- infinite support;
- infinitely differentiable;
- maximum entropy (fixed variance);
- convex loss (linear regression);
- conditional expectation is linear;
- equivalence to least squares;
- uncorrelatedness $\Leftrightarrow$ independence;
- universal basis.



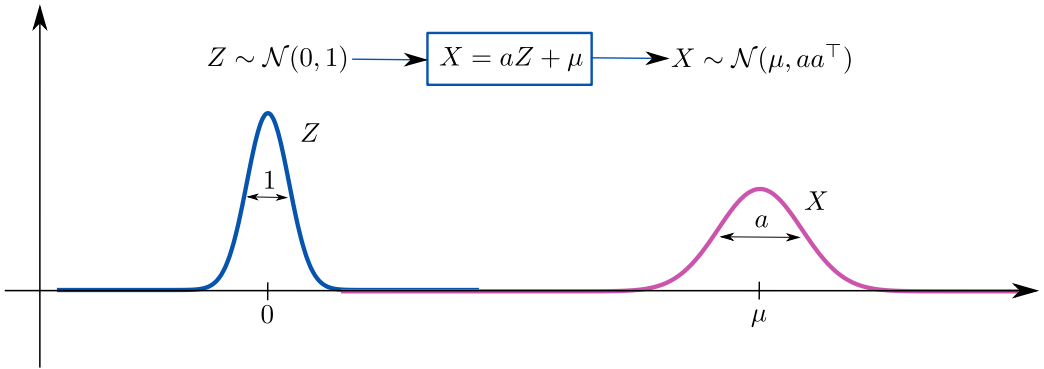


$$X \sim \mathcal{N}(\mu, \sigma^2) \Leftrightarrow p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$



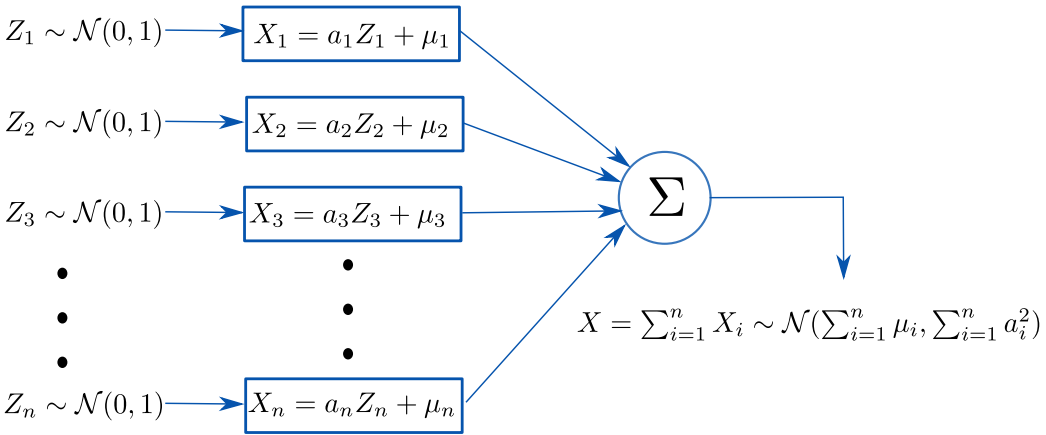


Linear transformation of Gaussians



- Closed under linear transformations.
- Any Gaussian RV can be produced from a linear transformation of $\mathcal{N}(0,1)$.

Linear combinations of Gaussians



(we have assumed the RVs to be independent but the result holds for the correlated case)

Summary of convenient Gaussian properties

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix}.$$

Marginalization

The observation of a larger collection of variables does not affect the distribution of smaller subsets:

$$\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}) \text{ and } \boldsymbol{x}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22}).$$

Conditioning

Gaussian conditioning results in another Gaussian:

$$p(\boldsymbol{x}_1|\boldsymbol{x}_2 = \boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}_1|\boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{z} - \boldsymbol{\mu}_2), \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}).$$

Linearity

Linear combination of Gaussians results in another Gaussian:

$$p(a\boldsymbol{x}_1 + b\boldsymbol{x}_2) = \mathcal{N}(a\boldsymbol{\mu}_1 + b\boldsymbol{\mu}_2, a^2\boldsymbol{\Sigma}_{11} + b^2\boldsymbol{\Sigma}_{22}).$$

Bayesian linear regression

Math details

- Considering inputs $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ and observed outputs $\boldsymbol{y} \in \mathbb{R}^{\mathbb{N}}$:

$$
\begin{aligned}
y_i &= \boldsymbol{w}^\top \boldsymbol{x}_i + \epsilon_i, \quad p(\epsilon_i) = \mathcal{N}(\epsilon_i | 0, \sigma^2), \quad i = 1, \cdots, N, \\
p(\boldsymbol{y} | \boldsymbol{X}, \boldsymbol{w}, \sigma^2) &= \mathcal{N}(\boldsymbol{y} | \boldsymbol{X}\boldsymbol{w}, \sigma^2 \boldsymbol{I}), \\
p(\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{w} | \boldsymbol{m}_0, \boldsymbol{S}_0).
\end{aligned}
$$

- The marginal likelihood is tractable:

$$
\begin{aligned}
p(\boldsymbol{y} | \boldsymbol{X}) &= \int p(\boldsymbol{y} | \boldsymbol{X}, \boldsymbol{w}, \sigma^2) p(\boldsymbol{w}) \mathrm{d}\boldsymbol{w} \\
&= \int \mathcal{N}(\boldsymbol{y} | \boldsymbol{X}\boldsymbol{w}, \sigma^2 \boldsymbol{I}) \mathcal{N}(\boldsymbol{w} | \boldsymbol{m}_0, \boldsymbol{S}_0) \mathrm{d}\boldsymbol{w} \\
&= \mathcal{N}(\boldsymbol{y} | \boldsymbol{X}\boldsymbol{m}_0, \boldsymbol{X}\boldsymbol{S}_0\boldsymbol{X}^\top + \sigma^2 \boldsymbol{I}).
\end{aligned}
$$

- The covariance between $\boldsymbol{w}$ and $\boldsymbol{y}$ is calculated by

$$
\begin{aligned}
\mathrm{cov}[\boldsymbol{w}, \boldsymbol{y}] &= \mathrm{cov}[\boldsymbol{w}, \boldsymbol{X}\boldsymbol{w} + \boldsymbol{\epsilon}] \\
&= \mathrm{cov}[\boldsymbol{w}, \boldsymbol{X}\boldsymbol{w}] = \boldsymbol{X}\,\mathrm{cov}[\boldsymbol{w}, \boldsymbol{w}] = \boldsymbol{X}\boldsymbol{S}_0.
\end{aligned}
$$

Bayesian linear regression

Math details

- Considering inputs $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ and observed outputs $\boldsymbol{y} \in \mathbb{R}^{\mathbb{N}}$:

$$
\begin{aligned}
y_i &= \boldsymbol{w}^\top \boldsymbol{x}_i + \epsilon_i, \quad p(\epsilon_i) = \mathcal{N}(\epsilon_i | 0, \sigma^2), \quad i = 1, \cdots, N, \\
p(\boldsymbol{y} | \boldsymbol{X}, \boldsymbol{w}, \sigma^2) &= \mathcal{N}(\boldsymbol{y} | \boldsymbol{X}\boldsymbol{w}, \sigma^2 \boldsymbol{I}), \\
p(\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{w} | \boldsymbol{m}_0, \boldsymbol{S}_0).
\end{aligned}
$$

- The marginal likelihood is tractable:

$$
\begin{aligned}
p(\boldsymbol{y} | \boldsymbol{X}) &= \int p(\boldsymbol{y} | \boldsymbol{X}, \boldsymbol{w}, \sigma^2) p(\boldsymbol{w}) \mathrm{d}\boldsymbol{w} \\
&= \int \mathcal{N}(\boldsymbol{y} | \boldsymbol{X}\boldsymbol{w}, \sigma^2 \boldsymbol{I}) \mathcal{N}(\boldsymbol{w} | \boldsymbol{m}_0, \boldsymbol{S}_0) \mathrm{d}\boldsymbol{w} \\
&= \mathcal{N}(\boldsymbol{y} | \boldsymbol{X}\boldsymbol{m}_0, \boldsymbol{X}\boldsymbol{S}_0\boldsymbol{X}^\top + \sigma^2 \boldsymbol{I}).
\end{aligned}
$$

- The covariance between $\boldsymbol{w}$ and $\boldsymbol{y}$ is calculated by

$$
\begin{aligned}
\mathrm{cov}[\boldsymbol{w}, \boldsymbol{y}] &= \mathrm{cov}[\boldsymbol{w}, \boldsymbol{X}\boldsymbol{w} + \boldsymbol{\epsilon}] \\
&= \mathrm{cov}[\boldsymbol{w}, \boldsymbol{X}\boldsymbol{w}] = \boldsymbol{X}\,\mathrm{cov}[\boldsymbol{w}, \boldsymbol{w}] = \boldsymbol{X}\boldsymbol{S}_0.
\end{aligned}
$$

Bayesian linear regression

Math details

- Considering inputs $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ and observed outputs $\boldsymbol{y} \in \mathbb{R}^{\mathbb{N}}$:

$$
\begin{aligned}
y_i &= \boldsymbol{w}^\top \boldsymbol{x}_i + \epsilon_i, \quad p(\epsilon_i) = \mathcal{N}(\epsilon_i | 0, \sigma^2), \quad i = 1, \cdots, N, \\
p(\boldsymbol{y} | \boldsymbol{X}, \boldsymbol{w}, \sigma^2) &= \mathcal{N}(\boldsymbol{y} | \boldsymbol{X}\boldsymbol{w}, \sigma^2 \boldsymbol{I}), \\
p(\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{w} | \boldsymbol{m}_0, \boldsymbol{S}_0).
\end{aligned}
$$

- The marginal likelihood is tractable:

$$
\begin{aligned}
p(\boldsymbol{y} | \boldsymbol{X}) &= \int p(\boldsymbol{y} | \boldsymbol{X}, \boldsymbol{w}, \sigma^2) p(\boldsymbol{w}) \mathrm{d}\boldsymbol{w} \\
&= \int \mathcal{N}(\boldsymbol{y} | \boldsymbol{X}\boldsymbol{w}, \sigma^2 \boldsymbol{I}) \mathcal{N}(\boldsymbol{w} | \boldsymbol{m}_0, \boldsymbol{S}_0) \mathrm{d}\boldsymbol{w} \\
&= \mathcal{N}(\boldsymbol{y} | \boldsymbol{X}\boldsymbol{m}_0, \boldsymbol{X}\boldsymbol{S}_0\boldsymbol{X}^\top + \sigma^2 \boldsymbol{I}).
\end{aligned}
$$

- The covariance between $\boldsymbol{w}$ and $\boldsymbol{y}$ is calculated by

$$
\begin{aligned}
\mathrm{cov}[\boldsymbol{w}, \boldsymbol{y}] &= \mathrm{cov}[\boldsymbol{w}, \boldsymbol{X}\boldsymbol{w} + \boldsymbol{\epsilon}] \\
&= \mathrm{cov}[\boldsymbol{w}, \boldsymbol{X}\boldsymbol{w}] = \boldsymbol{X}\mathrm{cov}[\boldsymbol{w}, \boldsymbol{w}] = \boldsymbol{X}\boldsymbol{S}_0.
\end{aligned}
$$

Bayesian linear regression

Math details

- The joint distribution $p(\boldsymbol{w}, \boldsymbol{y})$ can be readily built:

$$p\left(\left[\begin{array}{c} \boldsymbol{w} \\ \boldsymbol{y} \end{array}\right] \middle| \boldsymbol{X}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2\right) = \mathcal{N}\left(\left[\begin{array}{c} \boldsymbol{w} \\ \boldsymbol{y} \end{array}\right] \middle| \left[\begin{array}{c} \boldsymbol{m}_0 \\ \boldsymbol{X}\boldsymbol{m}_0 \end{array}\right], \left[\begin{array}{cc} \boldsymbol{S}_0 & \boldsymbol{S}_0\boldsymbol{X}^\top \\ \boldsymbol{X}\boldsymbol{S}_0 & \boldsymbol{X}\boldsymbol{S}_0\boldsymbol{X}^\top + \sigma^2\boldsymbol{I} \end{array}\right]\right)$$

- Using the Gaussian conditioning property, the posterior over $\boldsymbol{w}$ after observing $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$ is

$$\begin{aligned} p(\boldsymbol{w}|\mathcal{D}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}), \\ \boldsymbol{\mu} &= \boldsymbol{m}_0 + (\boldsymbol{S}_0\boldsymbol{X}^\top\boldsymbol{X} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{S}_0\boldsymbol{X}^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{m}_0), \\ \boldsymbol{\Sigma} &= \boldsymbol{S}_0 - (\boldsymbol{S}_0\boldsymbol{X}^\top\boldsymbol{X} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{S}_0\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{S}_0. \end{aligned}$$

- Predictions for new inputs $\boldsymbol{X}_*$ are given by

$$\begin{aligned} p(\boldsymbol{y}_*|\boldsymbol{X}_*, \mathcal{D}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2) &= \int p(\boldsymbol{y}_*|\boldsymbol{w}, \sigma^2)p(\boldsymbol{w}|\mathcal{D}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2)\mathrm{d}\boldsymbol{w} \\ &= \int \mathcal{N}(\boldsymbol{y}_*|\boldsymbol{X}_*\boldsymbol{w}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\mathrm{d}\boldsymbol{w} \\ &= \mathcal{N}(\boldsymbol{y}_*|\boldsymbol{X}_*\boldsymbol{\mu}, \boldsymbol{X}_*\boldsymbol{\Sigma}\boldsymbol{X}_*^\top + \sigma^2\boldsymbol{I}). \end{aligned}$$

Bayesian linear regression

Math details

- The joint distribution $p(\boldsymbol{w}, \boldsymbol{y})$ can be readily built:

$$p\left(\left[\begin{array}{c} \boldsymbol{w} \\ \boldsymbol{y} \end{array}\right] \middle| \boldsymbol{X}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2\right) = \mathcal{N}\left(\left[\begin{array}{c} \boldsymbol{w} \\ \boldsymbol{y} \end{array}\right] \middle| \left[\begin{array}{c} \boldsymbol{m}_0 \\ \boldsymbol{X}\boldsymbol{m}_0 \end{array}\right], \left[\begin{array}{cc} \boldsymbol{S}_0 & \boldsymbol{S}_0\boldsymbol{X}^\top \\ \boldsymbol{X}\boldsymbol{S}_0 & \boldsymbol{X}\boldsymbol{S}_0\boldsymbol{X}^\top + \sigma^2\boldsymbol{I} \end{array}\right]\right)$$

- Using the Gaussian conditioning property, the posterior over $\boldsymbol{w}$ after observing $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$ is

$$\begin{aligned} p(\boldsymbol{w}|\mathcal{D}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}), \\ \boldsymbol{\mu} &= \boldsymbol{m}_0 + (\boldsymbol{S}_0\boldsymbol{X}^\top\boldsymbol{X} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{S}_0\boldsymbol{X}^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{m}_0), \\ \boldsymbol{\Sigma} &= \boldsymbol{S}_0 - (\boldsymbol{S}_0\boldsymbol{X}^\top\boldsymbol{X} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{S}_0\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{S}_0. \end{aligned}$$

- Predictions for new inputs $\boldsymbol{X}_*$ are given by

$$\begin{aligned} p(\boldsymbol{y}_*|\boldsymbol{X}_*, \mathcal{D}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2) &= \int p(\boldsymbol{y}_*|\boldsymbol{w}, \sigma^2)p(\boldsymbol{w}|\mathcal{D}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2)\mathrm{d}\boldsymbol{w} \\ &= \int \mathcal{N}(\boldsymbol{y}_*|\boldsymbol{X}_*\boldsymbol{w}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\mathrm{d}\boldsymbol{w} \\ &= \mathcal{N}(\boldsymbol{y}_*|\boldsymbol{X}_*\boldsymbol{\mu}, \boldsymbol{X}_*\boldsymbol{\Sigma}\boldsymbol{X}_*^\top + \sigma^2\boldsymbol{I}). \end{aligned}$$

Bayesian linear regression

Math details

- The joint distribution $p(\boldsymbol{w}, \boldsymbol{y})$ can be readily built:

$$p\left(\left[\begin{array}{c}\boldsymbol{w}\\ \boldsymbol{y}\end{array}\right]\middle|\boldsymbol{X}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2\right) = \mathcal{N}\left(\left[\begin{array}{c}\boldsymbol{w}\\ \boldsymbol{y}\end{array}\right]\middle|\left[\begin{array}{c}\boldsymbol{m}_0\\ \boldsymbol{X}\boldsymbol{m}_0\end{array}\right], \left[\begin{array}{cc}\boldsymbol{S}_0 & \boldsymbol{S}_0\boldsymbol{X}^\top\\ \boldsymbol{X}\boldsymbol{S}_0 & \boldsymbol{X}\boldsymbol{S}_0\boldsymbol{X}^\top + \sigma^2\boldsymbol{I}\end{array}\right]\right)$$

- Using the Gaussian conditioning property, the posterior over $\boldsymbol{w}$ after observing $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$ is

$$\begin{aligned} p(\boldsymbol{w}|\mathcal{D}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}),\\ \boldsymbol{\mu} &= \boldsymbol{m}_0 + (\boldsymbol{S}_0\boldsymbol{X}^\top\boldsymbol{X} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{S}_0\boldsymbol{X}^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{m}_0),\\ \boldsymbol{\Sigma} &= \boldsymbol{S}_0 - (\boldsymbol{S}_0\boldsymbol{X}^\top\boldsymbol{X} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{S}_0\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{S}_0. \end{aligned}$$

- Predictions for new inputs $\boldsymbol{X}_*$ are given by

$$\begin{aligned} p(\boldsymbol{y}_*|\boldsymbol{X}_*, \mathcal{D}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2) &= \int p(\boldsymbol{y}_*|\boldsymbol{w}, \sigma^2)p(\boldsymbol{w}|\mathcal{D}, \boldsymbol{m}_0, \boldsymbol{S}_0, \sigma^2)\mathrm{d}\boldsymbol{w}\\ &= \int \mathcal{N}(\boldsymbol{y}_*|\boldsymbol{X}_*\boldsymbol{w}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\mathrm{d}\boldsymbol{w}\\ &= \mathcal{N}(\boldsymbol{y}_*|\boldsymbol{X}_*\boldsymbol{\mu}, \boldsymbol{X}_*\boldsymbol{\Sigma}\boldsymbol{X}_*^\top + \sigma^2\boldsymbol{I}). \end{aligned}$$

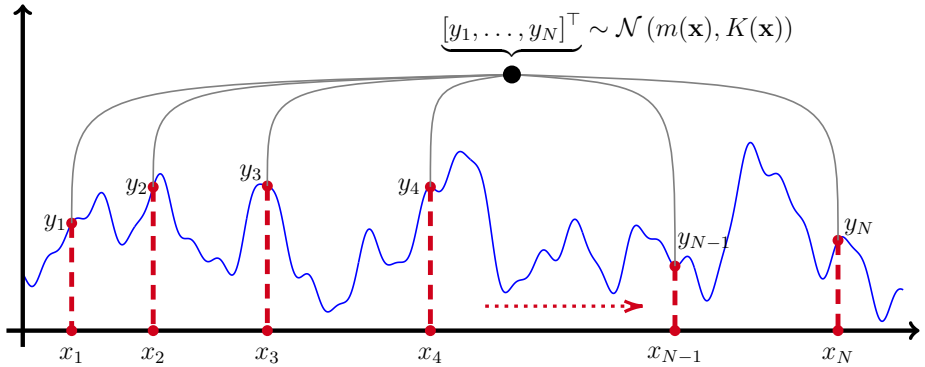
Table of Contents

What do we seek?

- Avoid a finite parametric formulation that acts as a **bottleneck** between data and predictions.
- Proper **uncertainty handling** and predictive distributions.
- Balance between **data fit** and **regularization**.
- (Few) hyperparameters that can be **directly** adjusted.
- Solid theoretical framework with **useful properties** from Gaussians.
- **Gaussian process**: a Bayesian nonparametric modeling framework.

The Gaussian process

Definition: A GP is a stochastic process such that any finite collection of values follows a multivariate Gaussian distribution.



The Gaussian process

- Notation for N observations:

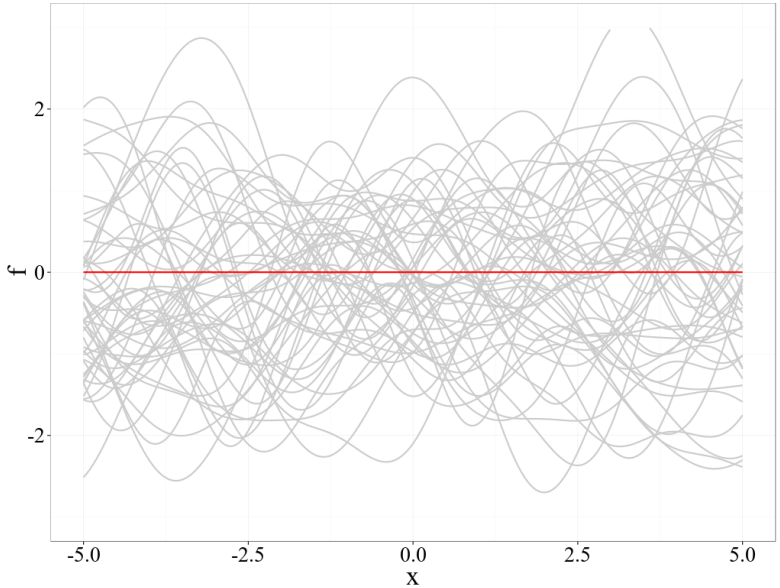
$$f \sim \mathcal{GP}\left(\boldsymbol{\mu}, \boldsymbol{K}\right) \iff f(\boldsymbol{x}) \sim \mathcal{N}(m(\mathbf{x}), K(\mathbf{x}, \mathbf{x}')),$$

$$p(\boldsymbol{f}|\boldsymbol{\mu}, \boldsymbol{K}) = \frac{1}{(2\pi)^{\frac{N}{2}}|\boldsymbol{K}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\boldsymbol{f}-\boldsymbol{\mu})^\top \boldsymbol{K}^{-1}(\boldsymbol{f}-\boldsymbol{\mu})\right).$$

Important

- The GP is a distribution over **functions**.
- The likelihood $p(\boldsymbol{y}|\boldsymbol{f})$ relates the noisy observations $\boldsymbol{y}$ and the latent function values $\boldsymbol{f}$.
- The GP is entirely defined by its mean ($m(\boldsymbol{x})$) and covariance ($K(\boldsymbol{x}, \boldsymbol{x'})$) functions.
- We usually choose a zero mean prior ($\boldsymbol{\mu} = \mathbf{0}$).
- The GP allows us to analyze functions (**infinite objects**) from finite distributions (**multivariate Gaussians**).
- The function f is a single (infinite dimensional) GP sample, and, accordingly, the vector $\boldsymbol{f}$ is a single (finite dimensional) multivariate Gaussian sample.

Samples from the GP prior



Nonlinear regression

- Consider a dataset
$$\mathcal{D} = \{(\boldsymbol{x}_i, y_i)|_{i=1}^N\} = (\boldsymbol{X}, \boldsymbol{y}),$$
where $\boldsymbol{x}_i \in \mathbb{R}^D$ are the inputs, $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ and $\boldsymbol{y} \in \mathbb{R}^N$ are the observed outputs.
- A general nonlinear regression task is expressed as
$$y_i = f(\boldsymbol{x}_i) + \epsilon_i,$$
where $\epsilon_i \in \mathbb{R}$ is the observation noise.
- The values $f_i = f(\boldsymbol{x}_i)$ are not directly observed (they are *latent*) and the true $f(\cdot)$ is unknown.

Nonlinear regression with GPs

- Choose a GP prior for the unknown function:

$$f \sim \mathcal{GP}(\mathbf{0}, \boldsymbol{K}),$$
$$\boldsymbol{f} = f(\boldsymbol{X}) \sim \mathcal{N}(\boldsymbol{f}|\mathbf{0}, \boldsymbol{K}),$$

  where $\boldsymbol{K} \in \mathbb{R}^{N \times N}$, $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, is the covariance matrix, obtained from the covariance, or *kernel* function, $k(\cdot, \cdot)$.

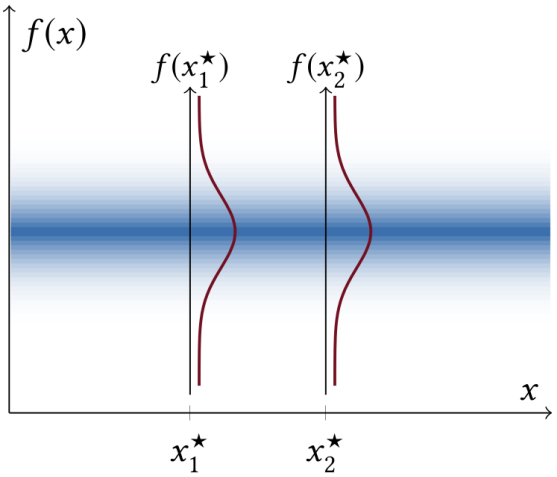
- Note that the output behavior is entirely explained by the covariance matrix $\boldsymbol{K}$, obtained from relations between the inputs.

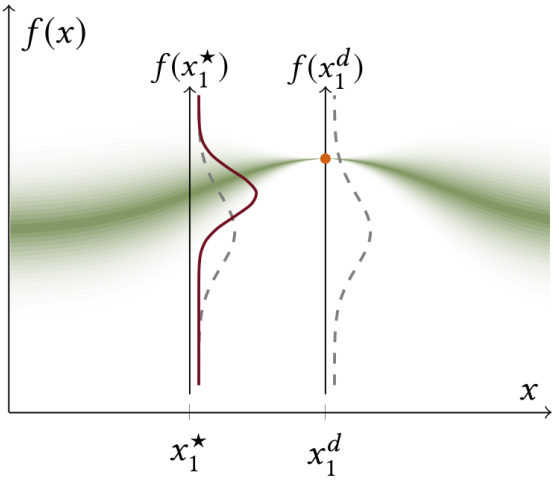
- A common choice is the *squared exponential* (or *Radial Basis Function* - RBF):

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_{id} - x_{jd})^2}{l_d^2}\right).$$

- The vector of hyperparameters $\boldsymbol{\theta} = [\sigma_f^2, l_1^2, \ldots, l_D^2]^\top$ characterizes the model covariance.

Nonlinear regression with GPs





Nonlinear regression with GPs

- If the observation noise is Gaussian, i.e., $\epsilon_i \sim \mathcal{N}(0, \sigma_y^2)$, all the expressions of interest are analytical.

Likelihood

Since the noise is Gaussian, the likelihood is also Gaussian:

$$p(\boldsymbol{y}|\boldsymbol{f}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{f}, \sigma_y^2 \boldsymbol{I}),$$

where $\boldsymbol{I} \in \mathbb{R}^{N \times N}$ is the N-th order identity matrix.

- **Important**: Different noise considerations lead to distinct likelihood functions and break the tractability (more on that later!).

Nonlinear regression with GPs

- If the observation noise is Gaussian, i.e., $\epsilon_i \sim \mathcal{N}(0, \sigma_y^2)$, all the expressions of interest are analytical.

Likelihood

Since the noise is Gaussian, the likelihood is also Gaussian:

$$p(\boldsymbol{y}|\boldsymbol{f}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{f}, \sigma_y^2 \boldsymbol{I}),$$

where $\boldsymbol{I} \in \mathbb{R}^{N \times N}$ is the N-th order identity matrix.

- **Important**: Different noise considerations lead to distinct likelihood functions and break the tractability (more on that later!).

Nonlinear regression with GPs

Marginal likelihood

The marginal likelihood is calculated by integrating $\boldsymbol{f}$:

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

Posteriori

The posterior over $\boldsymbol{f}$, after observing $\boldsymbol{y}$, is also Gaussian:

$$\underbrace{p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{X})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{y}|\boldsymbol{f})}^{\text{likelihood}}\overbrace{p(\boldsymbol{f}|\boldsymbol{X})}^{\text{priori}}}{\underbrace{p(\boldsymbol{y}|\boldsymbol{X})}_{\text{marginal likelihood}}} = \frac{\mathcal{N}(\boldsymbol{y}|\boldsymbol{f}, \sigma_y^2\boldsymbol{I})\mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K})}{\mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{K} + \sigma_y^2\boldsymbol{I})},$$
$$= \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}, \boldsymbol{K} - \boldsymbol{K}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{K}).$$

Nonlinear regression with GPs

Marginal likelihood

The marginal likelihood is calculated by integrating $\boldsymbol{f}$:

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

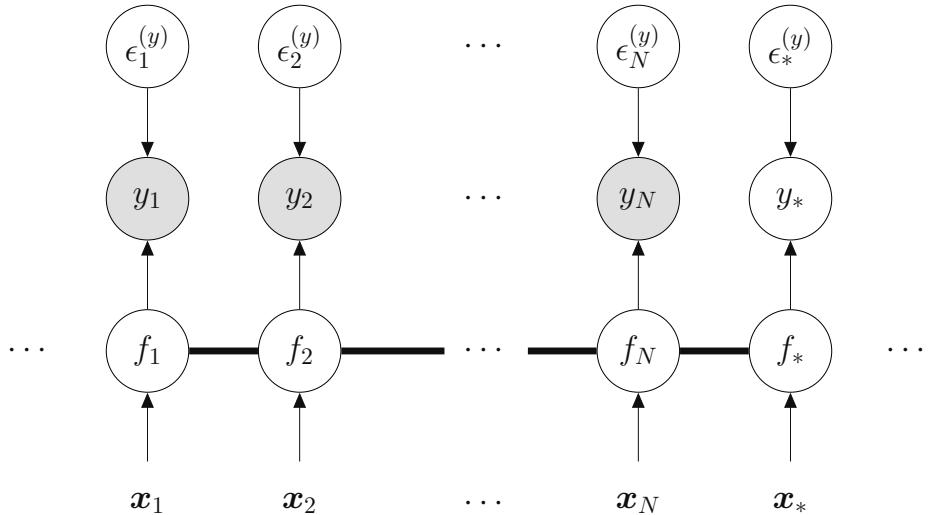
Posteriori

The posterior over $\boldsymbol{f}$, after observing $\boldsymbol{y}$, is also Gaussian:

$$\begin{aligned} \underbrace{p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{X})}_{\text{posterior}} &= \frac{\overbrace{p(\boldsymbol{y}|\boldsymbol{f})}^{\text{likelihood}}\overbrace{p(\boldsymbol{f}|\boldsymbol{X})}^{\text{priori}}}{\underbrace{p(\boldsymbol{y}|\boldsymbol{X})}_{\text{marginal likelihood}}} = \frac{\mathcal{N}(\boldsymbol{y}|\boldsymbol{f}, \sigma_y^2\boldsymbol{I})\mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K})}{\mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{K} + \sigma_y^2\boldsymbol{I})}, \\ &= \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}, \boldsymbol{K} - \boldsymbol{K}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{K}). \end{aligned}$$

Nonlinear regression with GPs

Graphical model



Predictions with GPs

- Inference for f_* (latent space) given $\boldsymbol{x}_*$ is obtained by using the posterior $p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{X})$:

$$
\begin{aligned}
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \int p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{X}) \mathrm{d}\boldsymbol{f}, \\
&= \int \mathcal{N}(f_*|\boldsymbol{k}_{*f}\boldsymbol{K}_f^{-1}\boldsymbol{f}, k_{**} - \boldsymbol{k}_{*f}\boldsymbol{K}_f^{-1}\boldsymbol{k}_{f*}) p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{X}) \mathrm{d}\boldsymbol{f}, \\
&= \mathcal{N}(f_*|\mu_*, \sigma_*^2), \\
\mu_* &= \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{y}, \\
\sigma_*^2 &= k_{**} - \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{f*}, \\
\text{where } \boldsymbol{k}_{f*} &= [k(\boldsymbol{x}_*, \boldsymbol{x}_1), \cdots, k(\boldsymbol{x}_*, \boldsymbol{x}_N)]^\top, \quad k_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*).
\end{aligned}
$$

- Inference for y_* (observation space) is obtained by marginalizing f_*:

$$
\begin{aligned}
p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \int p(y_*|f_*) p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \mathrm{d}\boldsymbol{f} \\
&= \mathcal{N}(y_*|\mu_*, \sigma_*^2 + \sigma_y^2).
\end{aligned}
$$

Predictions with GPs

- Inference for f_* (latent space) given $\boldsymbol{x}_*$ is obtained by using the posterior $p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{X})$:

$$
\begin{aligned}
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \int p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{X}) \mathrm{d}\boldsymbol{f}, \\
&= \int \mathcal{N}(f_*|\boldsymbol{k}_{*f}\boldsymbol{K}_f^{-1}\boldsymbol{f}, k_{**} - \boldsymbol{k}_{*f}\boldsymbol{K}_f^{-1}\boldsymbol{k}_{f*}) p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{X}) \mathrm{d}\boldsymbol{f}, \\
&= \mathcal{N}(f_*|\mu_*, \sigma_*^2), \\
\mu_* &= \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{y}, \\
\sigma_*^2 &= k_{**} - \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{f*}, \\
\text{where } \boldsymbol{k}_{f*} &= [k(\boldsymbol{x}_*, \boldsymbol{x}_1), \cdots, k(\boldsymbol{x}_*, \boldsymbol{x}_N)]^\top, \quad k_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*).
\end{aligned}
$$

- Inference for y_* (observation space) is obtained by marginalizing f_*:

$$
\begin{aligned}
p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \int p(y_*|f_*) p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \mathrm{d}\boldsymbol{f} \\
&= \mathcal{N}(y_*|\mu_*, \sigma_*^2 + \sigma_y^2).
\end{aligned}
$$

Predictions with GPs

Alternative derivation

- Inference for f_* given $\boldsymbol{x}_*$ may also be obtained from the joint distribution:

$$\begin{bmatrix} \boldsymbol{y} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K} + \sigma_y^2 \boldsymbol{I} & \boldsymbol{k}_{f*} \\ \boldsymbol{k}_{*f} & k_{**} \end{bmatrix}\right),$$

$$\begin{aligned} p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}(f_*|\mu_*, \sigma_*^2), \\ p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}(y_*|\mu_*, \sigma_*^2 + \sigma_y^2), \\ \mu_* &= \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{y}, \\ \sigma_*^2 &= k_{**} - \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{f*}, \\ \text{where } \boldsymbol{k}_{f*} &= [k(\boldsymbol{x}_*, \boldsymbol{x}_1), \cdots, k(\boldsymbol{x}_*, \boldsymbol{x}_N)]^\top, \quad k_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*). \end{aligned}$$

Stable implementation via Cholesky decomposition

$$\begin{aligned} \boldsymbol{K} + \sigma_y^2 \boldsymbol{I} &= \boldsymbol{L}\boldsymbol{L}^\top, \quad (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} = \boldsymbol{L}^{-\top} \boldsymbol{L}^{-1}, \\ \mu_* &= (\boldsymbol{L}^{-1} \boldsymbol{k}_{f*})^\top \boldsymbol{L}^{-1} \boldsymbol{y}, \\ \sigma_*^2 &= k_{**} - (\boldsymbol{L}^{-1} \boldsymbol{k}_{f*})^\top \boldsymbol{L}^{-1} \boldsymbol{k}_{f*}. \end{aligned}$$

Predictions with GPs

Alternative derivation

- Inference for f_* given $\boldsymbol{x}_*$ may also be obtained from the joint distribution:

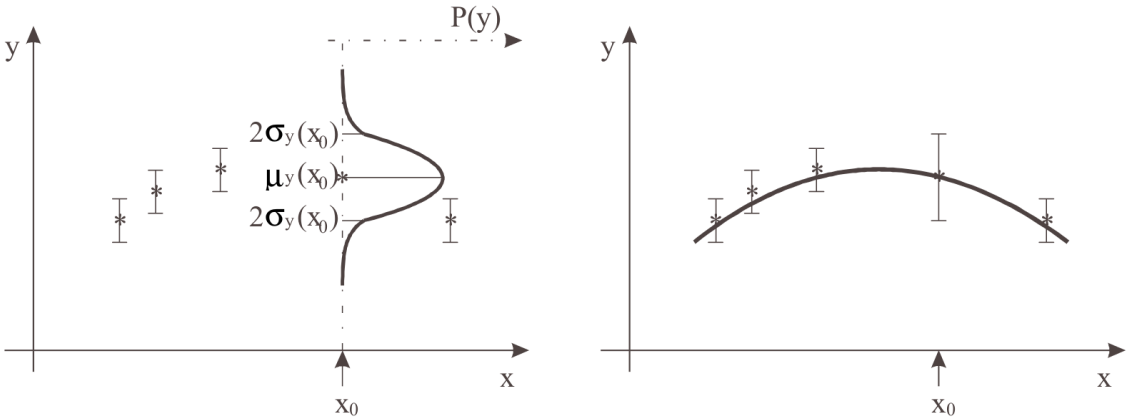
$$\begin{bmatrix} \boldsymbol{y} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K} + \sigma_y^2 \boldsymbol{I} & \boldsymbol{k}_{f*} \\ \boldsymbol{k}_{*f} & k_{**} \end{bmatrix}\right),$$

$$\begin{aligned} p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}(f_*|\mu_*, \sigma_*^2), \\ p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}(y_*|\mu_*, \sigma_*^2 + \sigma_y^2), \\ \mu_* &= \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{y}, \\ \sigma_*^2 &= k_{**} - \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{f*}, \\ \text{where } \boldsymbol{k}_{f*} &= [k(\boldsymbol{x}_*, \boldsymbol{x}_1), \cdots, k(\boldsymbol{x}_*, \boldsymbol{x}_N)]^\top, \quad k_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*). \end{aligned}$$

Stable implementation via Cholesky decomposition

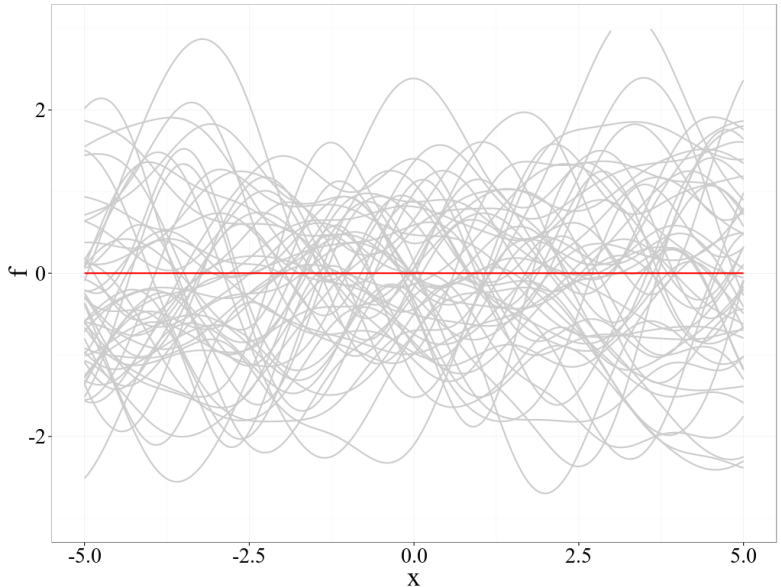
$$\begin{aligned} \boldsymbol{K} + \sigma_y^2 \boldsymbol{I} &= \boldsymbol{L}\boldsymbol{L}^\top, \quad (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} = \boldsymbol{L}^{-\top} \boldsymbol{L}^{-1}, \\ \mu_* &= (\boldsymbol{L}^{-1} \boldsymbol{k}_{f*})^\top \boldsymbol{L}^{-1} \boldsymbol{y}, \\ \sigma_*^2 &= k_{**} - (\boldsymbol{L}^{-1} \boldsymbol{k}_{f*})^\top \boldsymbol{L}^{-1} \boldsymbol{k}_{f*}. \end{aligned}$$

GP predictive distribution

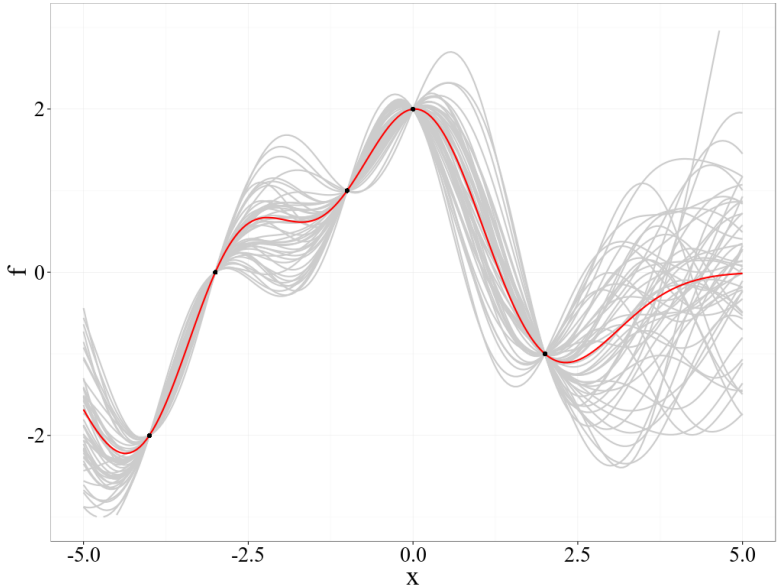


Note that each prediction is a fully defined Gaussian distribution.

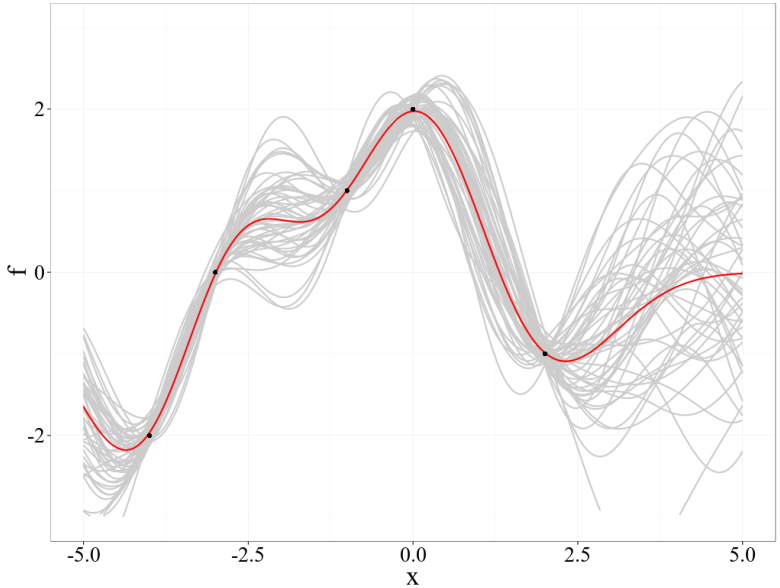
Samples from the GP posterior



Samples from the GP posterior



Samples from the GP posterior



From feature spaces to GPs

Math details

- Let $\phi : \mathbb{R}^D \to \mathbb{R}^Q$ be a mapping function and $\boldsymbol{w}_i \in \mathbb{R}^Q$ be a parameter vector:

$$y_i = \boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_y^2).$$

- If we consider the prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \boldsymbol{\Sigma}_w)$ and the matrix $\boldsymbol{\Phi} = \phi(\boldsymbol{X})$, where each row is given by $\phi(\boldsymbol{x}_i)$, the posterior is

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) = \frac{p(\boldsymbol{y}|\boldsymbol{w}, \boldsymbol{X})p(\boldsymbol{w})}{p(\boldsymbol{y}|\boldsymbol{X})} = \mathcal{N}\left(\boldsymbol{w} \middle| \frac{1}{\sigma_y^2}\boldsymbol{A}^{-1}\boldsymbol{\Phi}\boldsymbol{y}, \boldsymbol{A}^{-1}\right),$$

  where $\boldsymbol{A} \in \mathbb{R}^{Q\times Q} = \frac{1}{\sigma_y^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Sigma}_w^{-1}$.

- Predictions are made by marginalizing the parameters using the posterior:

$$\begin{aligned} p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \int p(f_*|\boldsymbol{x}_*, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X})\mathrm{d}\boldsymbol{w} \\ &= \mathcal{N}\left(f_* \middle| \frac{1}{\sigma_y^2}\phi(\boldsymbol{x}_*)^\top \boldsymbol{A}^{-1}\boldsymbol{\Phi}\boldsymbol{y}, \phi(\boldsymbol{x}_*)^\top \boldsymbol{A}^{-1}\phi(\boldsymbol{x}_*)\right). \end{aligned}$$

From feature spaces to GPs

Math details

- Let $\phi : \mathbb{R}^D \to \mathbb{R}^Q$ be a mapping function and $\boldsymbol{w}_i \in \mathbb{R}^Q$ be a parameter vector:

$$y_i = \boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_y^2).$$

- If we consider the prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \boldsymbol{\Sigma}_w)$ and the matrix $\boldsymbol{\Phi} = \phi(\boldsymbol{X})$, where each row is given by $\phi(\boldsymbol{x}_i)$, the posterior is

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) = \frac{p(\boldsymbol{y}|\boldsymbol{w}, \boldsymbol{X})p(\boldsymbol{w})}{p(\boldsymbol{y}|\boldsymbol{X})} = \mathcal{N}\left(\boldsymbol{w}\middle|\frac{1}{\sigma_y^2}\boldsymbol{A}^{-1}\boldsymbol{\Phi}\boldsymbol{y}, \boldsymbol{A}^{-1}\right),$$

  where $\boldsymbol{A} \in \mathbb{R}^{Q\times Q} = \frac{1}{\sigma_y^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Sigma}_w^{-1}$.

- Predictions are made by marginalizing the parameters using the posterior:

$$\begin{aligned} p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \int p(f_*|\boldsymbol{x}_*, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X})\mathrm{d}\boldsymbol{w} \\ &= \mathcal{N}\left(f_*\middle|\frac{1}{\sigma_y^2}\phi(\boldsymbol{x}_*)^\top \boldsymbol{A}^{-1}\boldsymbol{\Phi}\boldsymbol{y}, \phi(\boldsymbol{x}_*)^\top \boldsymbol{A}^{-1}\phi(\boldsymbol{x}_*)\right). \end{aligned}$$

From feature spaces to GPs

Math details

- Let $\phi : \mathbb{R}^D \to \mathbb{R}^Q$ be a mapping function and $\boldsymbol{w}_i \in \mathbb{R}^Q$ be a parameter vector:

$$y_i = \boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_y^2).$$

- If we consider the prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \boldsymbol{\Sigma}_w)$ and the matrix $\boldsymbol{\Phi} = \phi(\boldsymbol{X})$, where each row is given by $\phi(\boldsymbol{x}_i)$, the posterior is

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) = \frac{p(\boldsymbol{y}|\boldsymbol{w}, \boldsymbol{X})p(\boldsymbol{w})}{p(\boldsymbol{y}|\boldsymbol{X})} = \mathcal{N}\left(\boldsymbol{w} \middle| \frac{1}{\sigma_y^2}\boldsymbol{A}^{-1}\boldsymbol{\Phi}\boldsymbol{y}, \boldsymbol{A}^{-1}\right),$$

  where $\boldsymbol{A} \in \mathbb{R}^{Q\times Q} = \frac{1}{\sigma_y^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Sigma}_w^{-1}$.

- Predictions are made by marginalizing the parameters using the posterior:

$$\begin{aligned} p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \int p(f_*|\boldsymbol{x}_*, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X})\mathrm{d}\boldsymbol{w} \\ &= \mathcal{N}\left(f_* \middle| \frac{1}{\sigma_y^2}\phi(\boldsymbol{x}_*)^\top \boldsymbol{A}^{-1}\boldsymbol{\Phi}\boldsymbol{y}, \phi(\boldsymbol{x}_*)^\top \boldsymbol{A}^{-1}\phi(\boldsymbol{x}_*)\right). \end{aligned}$$

From feature spaces to GPs

Math details

- We replace the definition of $\boldsymbol{A}$ and use the *matrix inversion lemma* to rewrite the expression:

$$
\begin{aligned}
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}\left(f_* \middle| \frac{1}{\sigma_y^2}\phi(\boldsymbol{x}_*)^\top \left[\frac{1}{\sigma_y^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Sigma}_w^{-1}\right]^{-1}\boldsymbol{\Phi}\boldsymbol{y}, \phi(\boldsymbol{x}_*)^\top \left[\frac{1}{\sigma_y^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Sigma}_w^{-1}\right]^{-1}\phi(\boldsymbol{x}_*)\right) \\
&= \mathcal{N}\left(f_* \middle| \phi(\boldsymbol{x}_*)^\top \boldsymbol{\Sigma}_w \boldsymbol{\Phi}(\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w\boldsymbol{\Phi} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y},\right. \\
&\quad \left. \phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*) - \phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\boldsymbol{\Phi}(\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w\boldsymbol{\Phi} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*)\right).
\end{aligned}
$$

- The **kernel trick** can now be applied:

$$
\begin{aligned}
\boldsymbol{\Phi}^\top\boldsymbol{\Sigma}_w\boldsymbol{\Phi} = \boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w^{1/2}\boldsymbol{\Sigma}_w^{1/2}\boldsymbol{\Phi} = \boldsymbol{\Psi}^\top\boldsymbol{\Psi} = k(\boldsymbol{X},\boldsymbol{X}) = \boldsymbol{K}, \\
\phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\boldsymbol{\Phi} = k(\boldsymbol{x}_*,\boldsymbol{X}) = \boldsymbol{k}_{*f}, \\
\boldsymbol{\Phi}^\top\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*) = k(\boldsymbol{X},\boldsymbol{x}_*) = \boldsymbol{k}_{f*}, \\
\phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*) = k(\boldsymbol{x}_*,\boldsymbol{x}_*) = k_{**}.
\end{aligned}
$$

- Finally, the standard GP predictive distribution is obtained:

$$
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(\boldsymbol{k}_{*f}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}, k_{**} - \boldsymbol{k}_{*f}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{k}_{f*}).
$$

From feature spaces to GPs

Math details

- We replace the definition of $\boldsymbol{A}$ and use the *matrix inversion lemma* to rewrite the expression:

$$
\begin{aligned}
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}\left(f_* \middle| \frac{1}{\sigma_y^2}\phi(\boldsymbol{x}_*)^\top \left[\frac{1}{\sigma_y^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Sigma}_w^{-1}\right]^{-1}\boldsymbol{\Phi}\boldsymbol{y}, \phi(\boldsymbol{x}_*)^\top \left[\frac{1}{\sigma_y^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Sigma}_w^{-1}\right]^{-1}\phi(\boldsymbol{x}_*)\right) \\
&= \mathcal{N}\left(f_* \middle| \phi(\boldsymbol{x}_*)^\top \boldsymbol{\Sigma}_w \boldsymbol{\Phi}(\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w\boldsymbol{\Phi} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y},\right. \\
&\quad \left. \phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*) - \phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\boldsymbol{\Phi}(\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w\boldsymbol{\Phi} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*)\right).
\end{aligned}
$$

- The **kernel trick** can now be applied:

$$
\begin{aligned}
\boldsymbol{\Phi}^\top\boldsymbol{\Sigma}_w\boldsymbol{\Phi} &= \boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w^{1/2}\boldsymbol{\Sigma}_w^{1/2}\boldsymbol{\Phi} = \boldsymbol{\Psi}^\top\boldsymbol{\Psi} = k(\boldsymbol{X}, \boldsymbol{X}) = \boldsymbol{K}, \\
\phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\boldsymbol{\Phi} &= k(\boldsymbol{x}_*, \boldsymbol{X}) = \boldsymbol{k}_{*f}, \\
\boldsymbol{\Phi}^\top\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*) &= k(\boldsymbol{X}, \boldsymbol{x}_*) = \boldsymbol{k}_{f*}, \\
\phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*) &= k(\boldsymbol{x}_*, \boldsymbol{x}_*) = k_{**}.
\end{aligned}
$$

- Finally, the standard GP predictive distribution is obtained:

$$
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(\boldsymbol{k}_{*f}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}, k_{**} - \boldsymbol{k}_{*f}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{k}_{f*}).
$$

From feature spaces to GPs

Math details

- We replace the definition of $\boldsymbol{A}$ and use the *matrix inversion lemma* to rewrite the expression:

$$
\begin{aligned}
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}\left(f_* \middle| \frac{1}{\sigma_y^2}\phi(\boldsymbol{x}_*)^\top \left[\frac{1}{\sigma_y^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Sigma}_w^{-1}\right]^{-1}\boldsymbol{\Phi}\boldsymbol{y}, \phi(\boldsymbol{x}_*)^\top \left[\frac{1}{\sigma_y^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Sigma}_w^{-1}\right]^{-1}\phi(\boldsymbol{x}_*)\right) \\
&= \mathcal{N}\left(f_* \middle| \phi(\boldsymbol{x}_*)^\top \boldsymbol{\Sigma}_w \boldsymbol{\Phi}(\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w\boldsymbol{\Phi} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y},\right. \\
&\left. \phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*) - \phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\boldsymbol{\Phi}(\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w\boldsymbol{\Phi} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*)\right).
\end{aligned}
$$

- The **kernel trick** can now be applied:

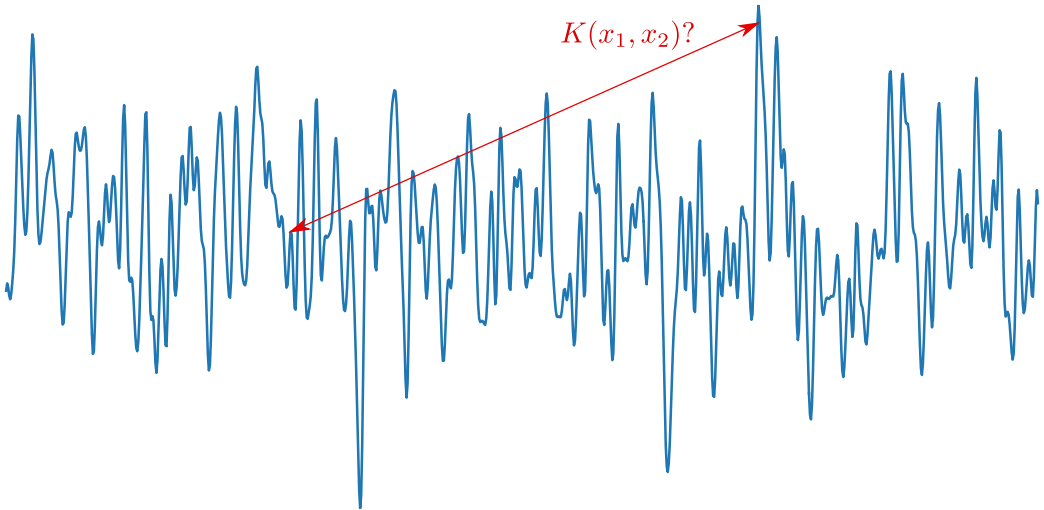
$$
\begin{aligned}
\boldsymbol{\Phi}^\top\boldsymbol{\Sigma}_w\boldsymbol{\Phi} &= \boldsymbol{\Phi}^T\boldsymbol{\Sigma}_w^{1/2}\boldsymbol{\Sigma}_w^{1/2}\boldsymbol{\Phi} = \boldsymbol{\Psi}^\top\boldsymbol{\Psi} = k(\boldsymbol{X}, \boldsymbol{X}) = \boldsymbol{K}, \\
\phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\boldsymbol{\Phi} &= k(\boldsymbol{x}_*, \boldsymbol{X}) = \boldsymbol{k}_{*f}, \\
\boldsymbol{\Phi}^\top\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*) &= k(\boldsymbol{X}, \boldsymbol{x}_*) = \boldsymbol{k}_{f*}, \\
\phi(\boldsymbol{x}_*)^\top\boldsymbol{\Sigma}_w\phi(\boldsymbol{x}_*) &= k(\boldsymbol{x}_*, \boldsymbol{x}_*) = k_{**}.
\end{aligned}
$$

- Finally, the standard GP predictive distribution is obtained:

$$
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(\boldsymbol{k}_{*f}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}, k_{**} - \boldsymbol{k}_{*f}(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{k}_{f*}).
$$

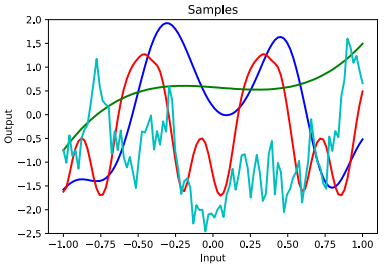
Table of Contents

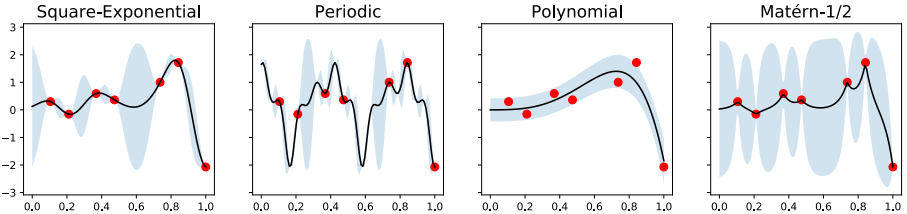
How to parameterize the kernel?



Different kernels, different predictions

- The choice of the covariance/kernel function directly affects the model behavior.
- Some of the most common options:
  - → Square-Exponential: $k_{\mathbf{SE}}(x, x') = \sigma_f^2 \exp\left(-\frac{(x-x')^2}{2l^2}\right)$
  - → Periodic: $k_{\mathbf{Per}}(x, x') = \sigma_f^2 \exp\left(-\frac{2\sin(\pi|x-x'|/p)}{l^2}\right)$
  - → Polynomial: $k_{\mathbf{Pol}}(x, x') = (x^\top x' + c)^d$
  - → Matérn-1/2: $k_{\mathbf{M}}(x, x') = \sigma_f^2 \exp\left(-\frac{|x-x'|}{2l^2}\right)$





Creating kernel functions

- Any function that generates a **positive semidefinite** covariance matrix (i.e. $\boldsymbol{a}^\top \boldsymbol{K}\boldsymbol{a} \geq 0, \forall \boldsymbol{a}$) is a valid kernel function.
- New kernel functions may be created by summing or multiplying multiple valid kernels, with optional scalings, which further improves model expressivity:

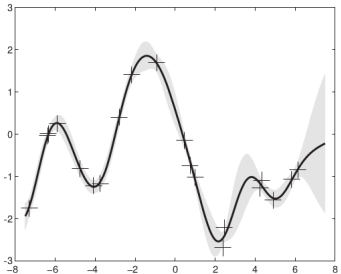
$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_m A_m k_m(\boldsymbol{x}_i, \boldsymbol{x}_j) + \prod_n B_n k_n(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

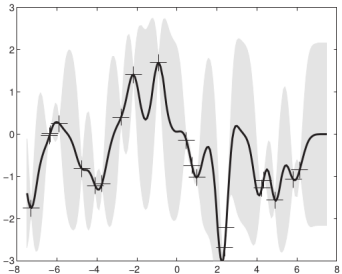
  where A_m and B_n are positive real constants.
- Mapping the inputs via an arbitrary nonlinear function $g(\cdot)$ (even if it changes the input dimension) before feeding them to the kernel function also results in valid kernels:

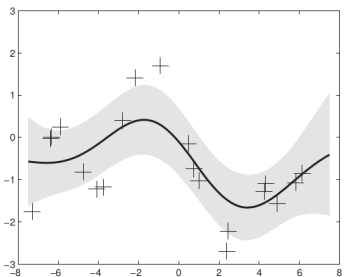
$$k_2(\boldsymbol{x}_i, \boldsymbol{x}_j) = k(g(\boldsymbol{x}_i), g(\boldsymbol{x}_j)).$$

Samples from the GP posterior

- The same kernel function and data can generate very different samples when distinct hyperparameters are used.







GP samples with larger lengthscales l (a), with smaller lengthscales l (b), and larger noise variance σ_y^2 (c).

Model selection for GPs

Hyperparameter optimization

The hyperparameter vector $\boldsymbol{\theta} = [\sigma_f^2, l_1^2, \ldots, l_D^2, \sigma_y^2]^\top$ (we have included the noise variance σ_y^2) can be optimized via maximization of the marginal log-likelihood, the so-called *model evidence*:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= \log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) \\ &= -\frac{1}{2}\underbrace{\log|\boldsymbol{K} + \sigma_y^2\boldsymbol{I}|}_{\text{model capacity}} - \frac{1}{2}\underbrace{\boldsymbol{y}^\top(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}}_{\text{data fit}} - \frac{N}{2}\log(2\pi).\end{aligned}$$

Stable implementation via Cholesky decomposition

$$\boldsymbol{K} + \sigma_y^2\boldsymbol{I} = \boldsymbol{L}\boldsymbol{L}^\top, \quad \boldsymbol{\alpha} = \boldsymbol{L}^{-1}\boldsymbol{y},$$

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_i \log L_{ii} - \frac{1}{2}\boldsymbol{\alpha}^\top\boldsymbol{\alpha} - \frac{N}{2}\log(2\pi).$$

Model selection for GPs

Hyperparameter optimization

The hyperparameter vector $\boldsymbol{\theta} = [\sigma_f^2, l_1^2, \ldots, l_D^2, \sigma_y^2]^\top$ (we have included the noise variance σ_y^2) can be optimized via maximization of the marginal log-likelihood, the so-called *model evidence*:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= \log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) \\ &= -\frac{1}{2}\underbrace{\log|\boldsymbol{K} + \sigma_y^2\boldsymbol{I}|}_{\text{model capacity}} - \frac{1}{2}\underbrace{\boldsymbol{y}^\top(\boldsymbol{K} + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}}_{\text{data fit}} - \frac{N}{2}\log(2\pi).\end{aligned}$$

Stable implementation via Cholesky decomposition

$$\begin{aligned}\boldsymbol{K} + \sigma_y^2\boldsymbol{I} = \boldsymbol{L}\boldsymbol{L}^\top, \quad \boldsymbol{\alpha} = \boldsymbol{L}^{-1}\boldsymbol{y}, \\ \mathcal{L}(\boldsymbol{\theta}) = -\sum_i \log L_{ii} - \frac{1}{2}\boldsymbol{\alpha}^\top\boldsymbol{\alpha} - \frac{N}{2}\log(2\pi).\end{aligned}$$

Model selection for GPs

Hyperparameter optimization

- The optimization follows analytical gradients:

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \theta_i} = -\frac{1}{2}\operatorname{Tr}\left((\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1}\frac{\partial(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})}{\partial \theta_i}\right) + \frac{1}{2}\boldsymbol{y}^\top(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1}\frac{\partial(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})}{\partial \theta_i}(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1}\boldsymbol{y}.$$

- It is not necessary to perform grid-search with cross-validations in the model selection phase!

Automatic Relevance Determination (ARD)

The optimized lengthscale hyperparameters $l_1^2, \ldots, l_D^2$ indicate the relevance of each input dimension, where more relevant dimensions correspond to lower values of l_d^2.

Model selection for GPs

Hyperparameter optimization

- The optimization follows analytical gradients:

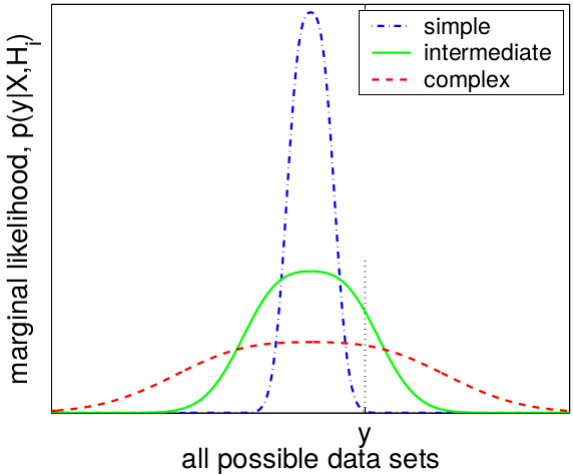
$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \theta_i} = -\frac{1}{2}\operatorname{Tr}\left((\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1}\frac{\partial(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})}{\partial \theta_i}\right) + \frac{1}{2}\boldsymbol{y}^\top(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1}\frac{\partial(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})}{\partial \theta_i}(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1}\boldsymbol{y}.$$

- It is not necessary to perform grid-search with cross-validations in the model selection phase!

Automatic Relevance Determination (ARD)

The optimized lengthscale hyperparameters $l_1^2, \ldots, l_D^2$ indicate the relevance of each input dimension, where more relevant dimensions correspond to lower values of l_d^2.

Bayesian model selection



The evidence framework privileges models with intermediate complexity.

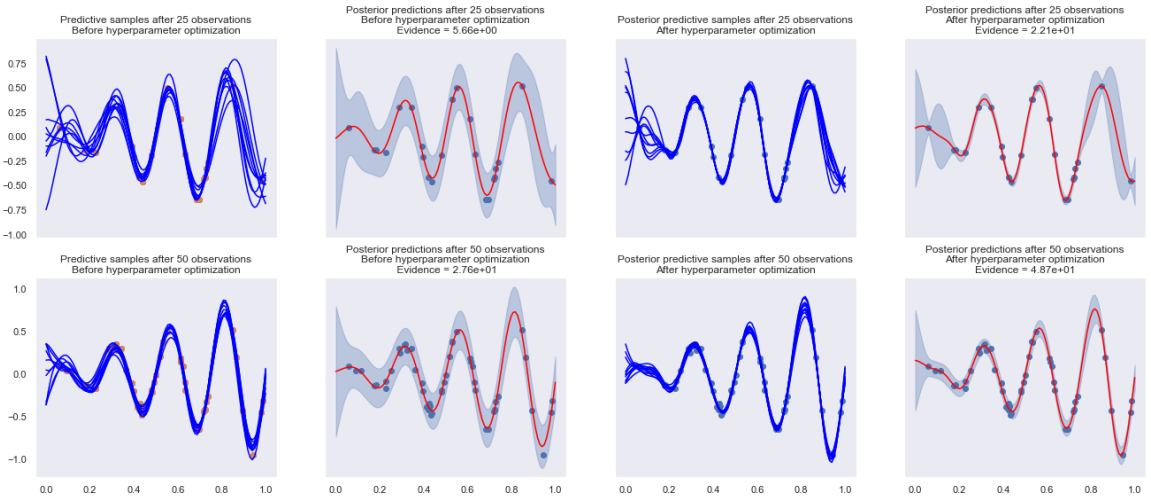
GP for nonlinear regression

Summary of the algorithm

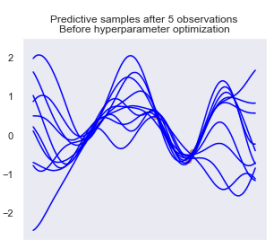
- Estimation step (model selection)
  1. Initialize the hyperparameters $\boldsymbol{\theta} = \left[\sigma_f^2, l_1^2, \cdots, l_D^2, \sigma_y^2\right]^\top$;
     $\rightarrow$ Example: $\sigma_f^2 = \mathbb{V}[\boldsymbol{y}], \quad l_d^2 = \mathbb{V}[\boldsymbol{X}_{:d}], \quad \sigma_y^2 = 0.01\sigma_f^2$.
  2. Repeat until convergence or for a maximum number of iterations:
     1. Compute the model evidence $\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})$;
     2. Compute the evidence analytical gradients $\frac{\partial \log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$;
     3. Update $\boldsymbol{\theta}$ with the computed gradients (e.g. via BFGS);
  3. Return the optimized hyperparameters $\hat{\boldsymbol{\theta}}$.
- Prediction step
  1. Given a new input $\boldsymbol{x}_*$, return the predictive distribution

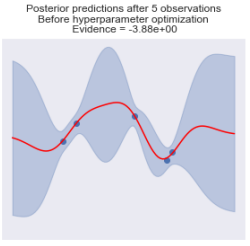
$$
\begin{aligned}
p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X}, \hat{\boldsymbol{\theta}}) &= \mathcal{N}\left(y_*|\mu_*, \sigma_*^2 + \sigma_y^2\right), \\
\mu_* &= \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{y}, \\
\sigma_*^2 &= k_{**} - \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{f*}.
\end{aligned}
$$

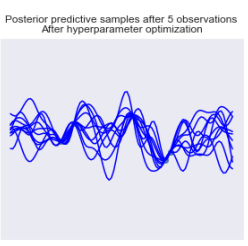
Samples from non-optimized and optimized GPs

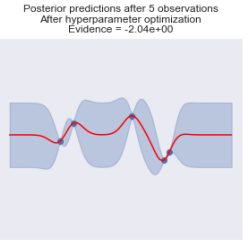


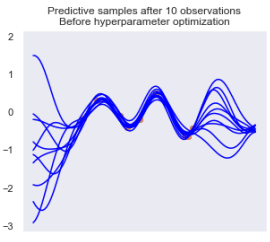
Samples from non-optimized and optimized GPs

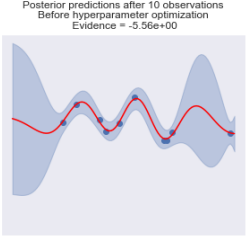


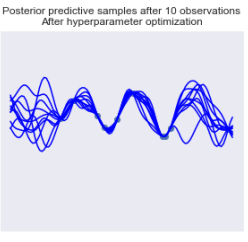












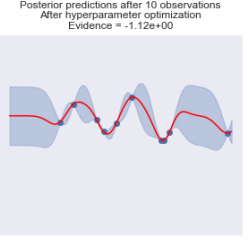


Table of Contents

Beyond Gaussian likelihood

- Gaussian likelihood is suitable only when Gaussian observations are expected:

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{f} + \boldsymbol{\epsilon}, \\ p(\boldsymbol{f}|\boldsymbol{X}) &\sim \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f), \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_y^2 \boldsymbol{I}), \\ p(\boldsymbol{y}|\boldsymbol{f}) &= \mathcal{N}(\boldsymbol{y}|\boldsymbol{f}, \sigma_y^2 \boldsymbol{I}). \end{aligned}$$

- For non-Gaussian data, we choose other likelihood $p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\phi})$ parameterized by $\boldsymbol{\phi}$.
  - Heavy-tailed regression: e.g. **Student-t** or **Laplace** likelihoods;
  - Non-negative regression: e.g. **gamma** or **exponential** likelihoods;
  - Discrete regression: e.g. **Poisson** likelihood;
  - Binary classification: e.g. **Bernoulli** likelihood;
  - Multiclass classification: e.g. **categorical** likelihood.
- **Problem**: The tractable integrals of GP models require a Gaussian likelihood.

Beyond Gaussian likelihood

- Gaussian likelihood is suitable only when Gaussian observations are expected:

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{f} + \boldsymbol{\epsilon}, \\ p(\boldsymbol{f}|\boldsymbol{X}) &\sim \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f), \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_y^2 \boldsymbol{I}), \\ p(\boldsymbol{y}|\boldsymbol{f}) &= \mathcal{N}(\boldsymbol{y}|\boldsymbol{f}, \sigma_y^2 \boldsymbol{I}). \end{aligned}$$

- For non-Gaussian data, we choose other likelihood $p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\phi})$ parameterized by $\boldsymbol{\phi}$.
  - Heavy-tailed regression: e.g. **Student-t** or **Laplace** likelihoods;
  - Non-negative regression: e.g. **gamma** or **exponential** likelihoods;
  - Discrete regression: e.g. **Poisson** likelihood;
  - Binary classification: e.g. **Bernoulli** likelihood;
  - Multiclass classification: e.g. **categorical** likelihood.
- **Problem**: The tractable integrals of GP models require a Gaussian likelihood.

Beyond Gaussian likelihood

- Gaussian likelihood is suitable only when Gaussian observations are expected:

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{f} + \boldsymbol{\epsilon}, \\ p(\boldsymbol{f}|\boldsymbol{X}) &\sim \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f), \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_y^2 \boldsymbol{I}), \\ p(\boldsymbol{y}|\boldsymbol{f}) &= \mathcal{N}(\boldsymbol{y}|\boldsymbol{f}, \sigma_y^2 \boldsymbol{I}). \end{aligned}$$

- For non-Gaussian data, we choose other likelihood $p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\phi})$ parameterized by $\boldsymbol{\phi}$.
  - Heavy-tailed regression: e.g. **Student-t** or **Laplace** likelihoods;
  - Non-negative regression: e.g. **gamma** or **exponential** likelihoods;
  - Discrete regression: e.g. **Poisson** likelihood;
  - Binary classification: e.g. **Bernoulli** likelihood;
  - Multiclass classification: e.g. **categorical** likelihood.
- **Problem**: The tractable integrals of GP models require a Gaussian likelihood.

Beyond Gaussian likelihood

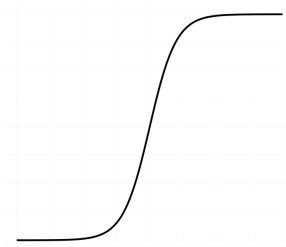
- Consider a **binary classification task** with Bernoulli observations:

$$p(y) = \mathcal{B}(y|a) = a^y(1-a^{1-y}), \quad a = p(y=1).$$

- A Bernoulli likelihood $p(y|f)$ is obtained with $a = \sigma(f) = p(y=1|f)$, where $\sigma(f) \in [0,1], \forall f \in \mathbb{R}$, is a **sigmoid function**:

$$p(y|f) = \sigma(f)^y(1-\sigma(f)^{1-y}),$$

$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{N} \sigma(f_i)^{y_i}(1-\sigma(f_i)^{1-y_i}).$$



→ $\sigma(f) = a$ is the **response (or transformation) function**.
→ $\sigma^{-1}(a) = f$ is the **link function**, which must be monotonic.
→ $f = \boldsymbol{\theta}^\top \boldsymbol{x}$ results in **logistic regression**, an example of a **generalized linear model**.

Beyond Gaussian likelihood

- Consider a **binary classification task** with Bernoulli observations:

$$p(y) = \mathcal{B}(y|a) = a^y(1-a^{1-y}), \quad a = p(y=1).$$

- A Bernoulli likelihood $p(y|f)$ is obtained with $a = \sigma(f) = p(y=1|f)$, where $\sigma(f) \in [0,1], \forall f \in \mathbb{R}$, is a **sigmoid function**:

$$p(y|f) = \sigma(f)^y(1-\sigma(f)^{1-y}),$$
$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{N} \sigma(f_i)^{y_i}(1-\sigma(f_i)^{1-y_i}).$$

→ $\sigma(f) = a$ is the **response (or transformation) function**.
→ $\sigma^{-1}(a) = f$ is the **link function**, which must be monotonic.
→ $f = \boldsymbol{\theta}^\top x$ results in **logistic regression**, an example of a **generalized linear model**.

Beyond Gaussian likelihood

- Consider a **binary classification task** with Bernoulli observations:

$$p(y) = \mathcal{B}(y|a) = a^y(1-a^{1-y}), \quad a = p(y=1).$$

- A Bernoulli likelihood $p(y|f)$ is obtained with $a = \sigma(f) = p(y=1|f)$, where $\sigma(f) \in [0,1], \forall f \in \mathbb{R}$, is a **sigmoid function**:

$$p(y|f) = \sigma(f)^y(1-\sigma(f)^{1-y}),$$
$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{N} \sigma(f_i)^{y_i}(1-\sigma(f_i)^{1-y_i}).$$

→ $\sigma(f) = a$ is the **response (or transformation) function**.
→ $\sigma^{-1}(a) = f$ is the **link function**, which must be monotonic.
→ $f = \boldsymbol{\theta}^\top \boldsymbol{x}$ results in **logistic regression**, an example of a **generalized linear model**.

Beyond Gaussian likelihood

- GP models for binary classification can be similarly obtained:

$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{N} \sigma(f_i)^{y_i}(1-\sigma(f_i)^{1-y_i}), \quad \boldsymbol{f} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_f).$$

- $\boldsymbol{f}$ is a **nuisance variable** and we are interested in predictions in the observed space:

$$\underbrace{p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})}_{\text{latent space}} = \int p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}) \mathrm{d}\boldsymbol{f},$$

$$\underbrace{p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})}_{\text{observed space}} = \int p(y_*|f_*) p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \mathrm{d}f_*.$$

- For a binary classification task with Bernoulli likelihood:

$$p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int \overbrace{\sigma(f_*)^{y_*}(1-\sigma(f_*)^{1-y_*})}^{p(y_*|f_*)} \overbrace{p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X})}^{\text{cond. Gaussian}} \overbrace{p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})}^{\text{posterior}} \mathrm{d}\boldsymbol{f} \mathrm{d}f_*,$$

$$p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int \sigma(f_*) p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}) \mathrm{d}\boldsymbol{f} \mathrm{d}f_*.$$

Beyond Gaussian likelihood

- GP models for binary classification can be similarly obtained:

$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{N} \sigma(f_i)^{y_i}(1-\sigma(f_i)^{1-y_i}), \quad \boldsymbol{f} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_f).$$

- $\boldsymbol{f}$ is a **nuisance variable** and we are interested in predictions in the observed space:

$$\underbrace{p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})}_{\text{latent space}} = \int p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}) \mathrm{d}\boldsymbol{f},$$

$$\underbrace{p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})}_{\text{observed space}} = \int p(y_*|f_*) p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \mathrm{d}f_*.$$

- For a binary classification task with Bernoulli likelihood:

$$p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int \overbrace{\sigma(f_*)^{y_*}(1-\sigma(f_*)^{1-y_*})}^{p(y_*|f_*)} \overbrace{p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X})}^{\text{cond. Gaussian}} \overbrace{p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})}^{\text{posterior}} \mathrm{d}\boldsymbol{f} \mathrm{d}f_*,$$

$$p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int \sigma(f_*) p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}) \mathrm{d}\boldsymbol{f} \mathrm{d}f_*.$$

Beyond Gaussian likelihood

- GP models for binary classification can be similarly obtained:

$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{N} \sigma(f_i)^{y_i}(1-\sigma(f_i)^{1-y_i}), \quad \boldsymbol{f} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_f).$$

- $\boldsymbol{f}$ is a **nuisance variable** and we are interested in predictions in the observed space:

$$\underbrace{p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})}_{\text{latent space}} = \int p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}) \mathrm{d}\boldsymbol{f},$$

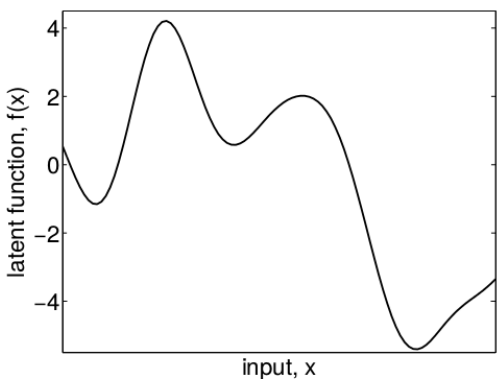
$$\underbrace{p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})}_{\text{observed space}} = \int p(y_*|f_*) p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \mathrm{d}f_*.$$

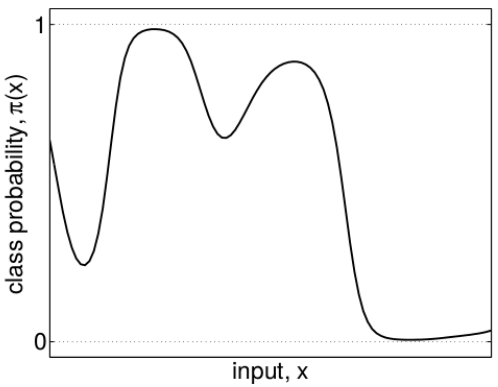
- For a binary classification task with Bernoulli likelihood:

$$p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int \overbrace{\sigma(f_*)^{y_*}(1-\sigma(f_*)^{1-y_*})}^{p(y_*|f_*)} \overbrace{p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X})}^{\text{cond. Gaussian}} \overbrace{p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})}^{\text{posterior}} \mathrm{d}\boldsymbol{f} \mathrm{d}f_*,$$

$$p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int \sigma(f_*) p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}) \mathrm{d}\boldsymbol{f} \mathrm{d}f_*.$$

Beyond Gaussian likelihood





(Rasmussen and Williams, 2006)

Beyond Gaussian likelihood

- **Problem**: The posterior $p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})$ is intractable for non-Gaussian likelihood:

$$\underbrace{p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X})}^{\text{non-Gaussian}} \overbrace{p(\boldsymbol{f}|\boldsymbol{X})}^{\text{Gaussian}}}{\underbrace{p(\boldsymbol{y}|\boldsymbol{X})}_{\text{non-Gaussian}}}.$$

- **Problem**: The marginal likelihood is also intractable for a non-Gaussian likelihood:

$$\begin{aligned}\overbrace{p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. likel.}} &= \int p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}) \mathrm{d}\boldsymbol{f} \\ &= \int \underbrace{p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X})}_{\text{non-Gaussian}} \underbrace{\mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f)}_{\text{GP prior}} \mathrm{d}\boldsymbol{f}.\end{aligned}$$

- We need **approximations** for both the posterior and the marginal likelihood.

Beyond Gaussian likelihood

- **Problem**: The posterior $p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})$ is intractable for non-Gaussian likelihood:

$$\underbrace{p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X})}^{\text{non-Gaussian}} \overbrace{p(\boldsymbol{f}|\boldsymbol{X})}^{\text{Gaussian}}}{\underbrace{p(\boldsymbol{y}|\boldsymbol{X})}_{\text{non-Gaussian}}}.$$

- **Problem**: The marginal likelihood is also intractable for a non-Gaussian likelihood:

$$\begin{aligned}\overbrace{p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. likel.}} &= \int p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}) \mathrm{d}\boldsymbol{f} \\ &= \int \underbrace{p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X})}_{\text{non-Gaussian}} \underbrace{\mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f)}_{\text{GP prior}} \mathrm{d}\boldsymbol{f}.\end{aligned}$$

- We need **approximations** for both the posterior and the marginal likelihood.

Beyond Gaussian likelihood

- **Problem**: The posterior $p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})$ is intractable for non-Gaussian likelihood:

$$\underbrace{p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X})}^{\text{non-Gaussian}} \overbrace{p(\boldsymbol{f}|\boldsymbol{X})}^{\text{Gaussian}}}{\underbrace{p(\boldsymbol{y}|\boldsymbol{X})}_{\text{non-Gaussian}}}.$$

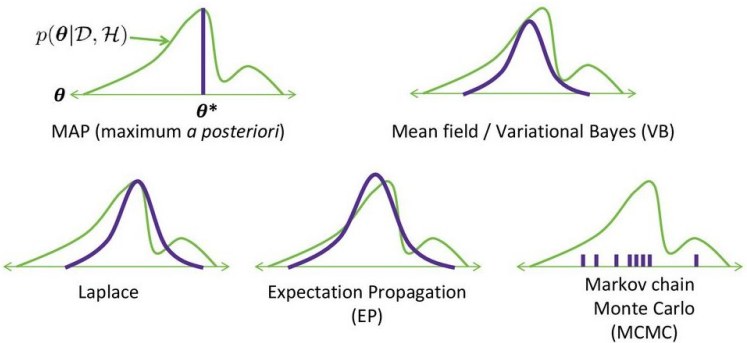
- **Problem**: The marginal likelihood is also intractable for a non-Gaussian likelihood:

$$\begin{aligned}\overbrace{p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. likel.}} &= \int p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}) \mathrm{d}\boldsymbol{f} \\ &= \int \underbrace{p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{X})}_{\text{non-Gaussian}} \underbrace{\mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f)}_{\text{GP prior}} \mathrm{d}\boldsymbol{f}.\end{aligned}$$

- We need **approximations** for both the posterior and the marginal likelihood.

Beyond Gaussian likelihood

- The approximations are usually performed by
  - → Laplace approximation;
  - → Variational inference;
  - → Expectation Propagation;
  - → Sampling methods (e.g. MCMC).



(Ben-Elazar, *et al.*, 2017)

Beyond Gaussian likelihood

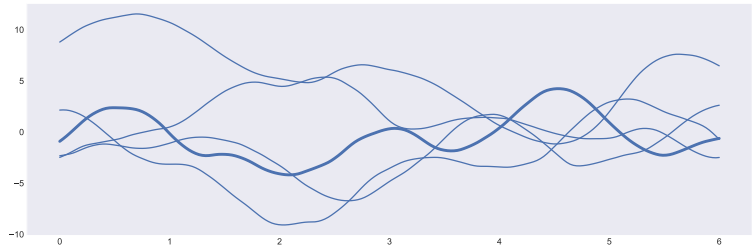
- For a **Gaussian approximate posterior** $q(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})$, the prediction becomes:

$$p(y_*|\boldsymbol{x}_*,\boldsymbol{X},\boldsymbol{y}) = \int \overbrace{p(y_*|f_*)}^{\text{non-Gaussian}}\; p(f_*|\boldsymbol{f},\boldsymbol{x}_*,\boldsymbol{X})\overbrace{p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})}^{\text{intractable}}\,\mathrm{d}\boldsymbol{f}\mathrm{d}f_*$$

$$\approx \int p(y_*|f_*)\left[\int \underbrace{p(f_*|\boldsymbol{f},\boldsymbol{x}_*,\boldsymbol{X})}_{\text{cond. Gaussian}}\;\underbrace{q(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})}_{\text{Gaussian approx.}}\;\mathrm{d}\boldsymbol{f}\right]\mathrm{d}f_*.$$

  - → The integral on $\boldsymbol{f}$ is tractable, since it involves only Gaussians.
  - → The integral on f_* is usually intractable, but since it is one-dimensional, it can be cheaply approximated by Monte Carlo, Gauss-Hermite quadrature, etc.
- Importantly, the previous frameworks also approximate the marginal likelihood.

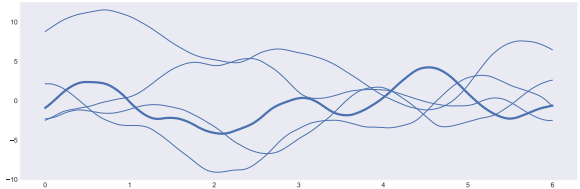
Beyond Gaussian likelihood



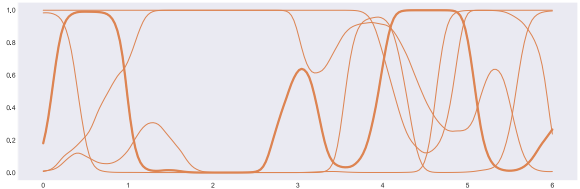
$$\boldsymbol{f} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_f)$$

$$(\boldsymbol{X}, \boldsymbol{y}), y_i \in \{0, 1\}, \forall i$$

Beyond Gaussian likelihood

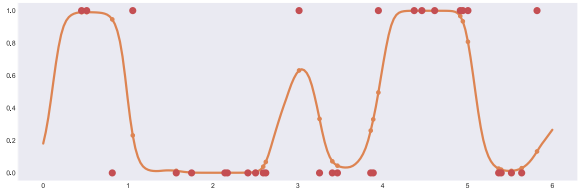


$$\boldsymbol{f} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_f)$$



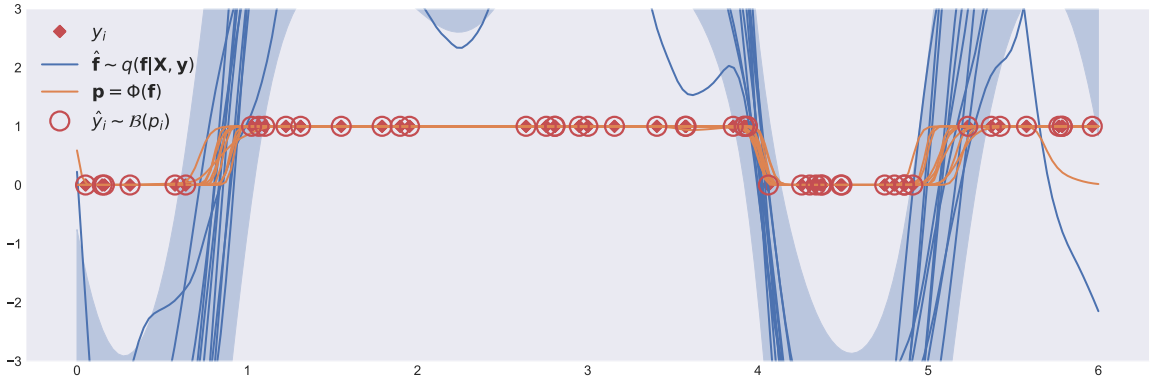
$$\boldsymbol{p} = \Phi(\boldsymbol{f}) \triangleq \int_{-\infty}^{f} \mathcal{N}(f|0,1)\mathrm{d}f$$

(Normal CDF)



$$y_i \sim \mathcal{B}(p_i), \forall i$$

Beyond Gaussian likelihood



- Gaussian approximate posterior $q(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{m}, \boldsymbol{\Sigma})$.
- Variational inference with **GPflow** (whose documentation this illustration was based on).
- Latent samples are continuous, but samples in the observed space remain discrete.

Warped GPs

- A **warping function** may be a useful preprocessing step for non-Gaussian observations.
  - → For instance, if $y > 0$, $f = \log(y)$.
- We can also consider a parameterized warping function $w(\cdot|\boldsymbol{\phi})$:

$$f = w(y|\boldsymbol{\phi}).$$

  - → $w(\cdot|\boldsymbol{\phi})$ must be monotonic, so that $y = w^{-1}(f|\boldsymbol{\phi})$ is unambiguous.
  - → If $p(\boldsymbol{f}|\boldsymbol{X})$ is a GP, we obtain a **warped GP model** (Snelson *et al.*, 2004).
- A **Bayesian warped GP** is obtained by directly modeling the inverse warping function (Lázaro-Gredilla, 2012):

$$y = g(f(\boldsymbol{x})) + \epsilon,$$

  where both $f(\cdot)$ and $g(\cdot)$ have GP priors and ϵ is an observation noise.
  - → Analogous to a 2-layer Deep GP!

Warped GPs

- A **warping function** may be a useful preprocessing step for non-Gaussian observations.
  - → For instance, if $y > 0$, $f = \log(y)$.
- We can also consider a parameterized warping function $w(\cdot|\boldsymbol{\phi})$:

$$f = w(y|\boldsymbol{\phi}).$$

  - → $w(\cdot|\boldsymbol{\phi})$ must be monotonic, so that $y = w^{-1}(f|\boldsymbol{\phi})$ is unambiguous.
  - → If $p(\boldsymbol{f}|\boldsymbol{X})$ is a GP, we obtain a **warped GP model** (Snelson *et al.*, 2004).
- A **Bayesian warped GP** is obtained by directly modeling the inverse warping function (Lázaro-Gredilla, 2012):

$$y = g(f(\boldsymbol{x})) + \epsilon,$$

where both $f(\cdot)$ and $g(\cdot)$ have GP priors and ϵ is an observation noise.

  - → Analogous to a 2-layer Deep GP!

Warped GPs

- A **warping function** may be a useful preprocessing step for non-Gaussian observations.
  - → For instance, if $y > 0$, $f = \log(y)$.
- We can also consider a parameterized warping function $w(\cdot|\boldsymbol{\phi})$:

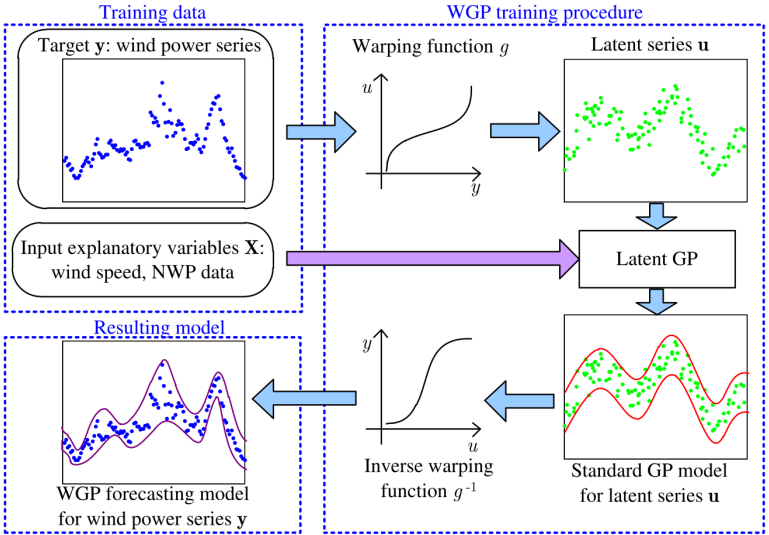
$$f = w(y|\boldsymbol{\phi}).$$

  - → $w(\cdot|\boldsymbol{\phi})$ must be monotonic, so that $y = w^{-1}(f|\boldsymbol{\phi})$ is unambiguous.
  - → If $p(\boldsymbol{f}|\boldsymbol{X})$ is a GP, we obtain a **warped GP model** (Snelson *et al.*, 2004).
- A **Bayesian warped GP** is obtained by directly modeling the inverse warping function (Lázaro-Gredilla, 2012):

$$y = g(f(\boldsymbol{x})) + \epsilon,$$

  where both $f(\cdot)$ and $g(\cdot)$ have GP priors and ϵ is an observation noise.
  - → Analogous to a 2-layer Deep GP!

Warped GPs



Kou, Peng *et al.* **"Sparse online warped Gaussian process for wind power probabilistic forecasting"**. Applied Energy, 2013.

Beyond Gaussian likelihood

- Flexible models can be obtained by considering **multiple GP priors**.
- **Heteroscedastic Gaussian likelihood**, i.e. input-dependent noise variance (Goldberg *et al.*, 1998; Lázaro-Gredilla and Titsias, 2011):

$$y = f(\boldsymbol{x}) + \epsilon,$$
$$f|\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_f), \quad \epsilon|\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \exp(g(\boldsymbol{x}))), \quad g|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_g, \boldsymbol{K}_g).$$

- **Chained GPs** with L GP priors for the likelihood parameters $\boldsymbol{\phi}$ (Saul *et al*, 2016):

$$\boldsymbol{\phi} = [\boldsymbol{g}^{(1)}, \ldots, \boldsymbol{g}^{(L)}],$$
$$p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\phi}) = p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{g}^{(1)}, \ldots, \boldsymbol{g}^{(L)}),$$
$$\boldsymbol{f}|\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_f),$$
$$\boldsymbol{g}^{(l)}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_{g^{(l)}}, \boldsymbol{K}_{g^{(l)}}), \quad 1 \leq l \leq L.$$

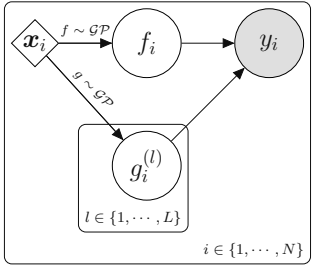
Beyond Gaussian likelihood

- Flexible models can be obtained by considering **multiple GP priors**.
- **Heteroscedastic Gaussian likelihood**, i.e. input-dependent noise variance (Goldberg *et al.*, 1998; Lázaro-Gredilla and Titsias, 2011):

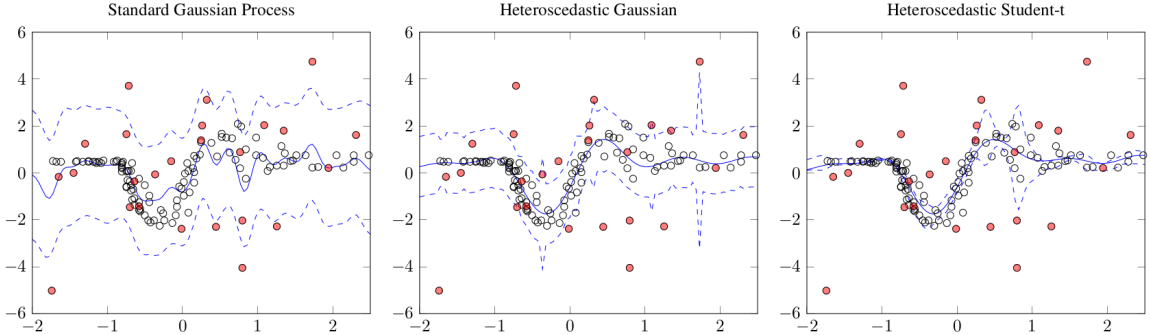
$$y = f(\boldsymbol{x}) + \epsilon,$$
$$f|\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_f), \quad \epsilon|\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \exp(g(\boldsymbol{x}))), \quad g|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_g, \boldsymbol{K}_g).$$

- **Chained GPs** with L GP priors for the likelihood parameters $\boldsymbol{\phi}$ (Saul *et al*, 2016):

$$\boldsymbol{\phi} = [\boldsymbol{g}^{(1)}, \ldots, \boldsymbol{g}^{(L)}],$$
$$p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\phi}) = p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{g}^{(1)}, \ldots, \boldsymbol{g}^{(L)}),$$
$$\boldsymbol{f}|\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}_f),$$
$$\boldsymbol{g}^{(l)}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_{g^{(l)}}, \boldsymbol{K}_{g^{(l)}}), \quad 1 \leq l \leq L.$$



Chained GPs



Saul, Alan, *et al.* **"Chained Gaussian processes"**. AISTATS, 2016.

$$
\begin{aligned}
y_i &\sim \mathrm{St}(\mu = f(\boldsymbol{x}_i), \sigma^2 = \exp(g(\boldsymbol{x}_i)), \nu), \\
\boldsymbol{f}|\boldsymbol{x} &\sim \mathcal{N}(\boldsymbol{\mu}_f, \boldsymbol{K}_f), \\
\boldsymbol{g}|\boldsymbol{x} &\sim \mathcal{N}(\boldsymbol{\mu}_g, \boldsymbol{K}_g).
\end{aligned}
$$

Beyond Gaussian likelihood

Summary

- GP models are tractable only for Gaussian likelihood.
- Other noise models are equivalent to consider **non-Gaussian likelihoods**.
- We approximate the posterior $\underbrace{p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})}_{\text{predictions}}$ and the marginal likelihood $\underbrace{p(\boldsymbol{y}|\boldsymbol{X})}_{\text{model selection}}$.
  → Usual approaches: LA, VI, EP or MCMC.
- The observations can also be **warped**, either by a parametric function or by directly considering a **second GP prior** for the **inverse warping function**.
- By considering **multiple latent functions**, we can obtain flexible **heteroscedastic** likelihoods.

Table of Contents

Sparse approximations

- We have seen how GP models are flexible and can be applied to both Gaussian and non-Gaussian observations.
- **Problem:** GP expressions involve the terms $|\boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I}|$ and $(\boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I})^{-1}$:

$$\overbrace{\log p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. likelihood}} = -\frac{1}{2}\log|\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I}| - \frac{1}{2}\boldsymbol{y}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y} - \frac{N}{2}\log(2\pi),$$

$$\underbrace{p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X})}_{\text{predictive dist.}} = \mathcal{N}\left(y_*|\boldsymbol{k}_{f*}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}, k_{**} - \boldsymbol{k}_{f*}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{k}_{f*} + \sigma_y^2\right).$$

  - $\rightarrow$ Naive computation scales with $\mathcal{O}(N^3)$.
  - $\rightarrow$ Storage scales with $\mathcal{O}(N^2)$.
- **Goal:** Mitigate (or eliminate!) the influence of N in the scaling.
- **Idea:** Maintain a smaller set of points that **summarizes the available data**.

Sparse approximations

- We have seen how GP models are flexible and can be applied to both Gaussian and non-Gaussian observations.
- **Problem:** GP expressions involve the terms $|\boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I}|$ and $(\boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I})^{-1}$:

$$\overbrace{\log p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. likelihood}} = -\frac{1}{2}\log|\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I}| - \frac{1}{2}\boldsymbol{y}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y} - \frac{N}{2}\log(2\pi),$$

$$\underbrace{p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X})}_{\text{predictive dist.}} = \mathcal{N}\left(y_*|\boldsymbol{k}_{f*}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}, k_{**} - \boldsymbol{k}_{f*}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{k}_{f*} + \sigma_y^2\right).$$

  → Naive computation scales with $\mathcal{O}(N^3)$.
  → Storage scales with $\mathcal{O}(N^2)$.

- **Goal:** Mitigate (or eliminate!) the influence of N in the scaling.
- **Idea:** Maintain a smaller set of points that **summarizes the available data**.

Sparse approximations

- We have seen how GP models are flexible and can be applied to both Gaussian and non-Gaussian observations.
- **Problem:** GP expressions involve the terms $|\boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I}|$ and $(\boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I})^{-1}$:

$$\overbrace{\log p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. likelihood}} = -\frac{1}{2}\log|\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I}| - \frac{1}{2}\boldsymbol{y}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y} - \frac{N}{2}\log(2\pi),$$

$$\underbrace{p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X})}_{\text{predictive dist.}} = \mathcal{N}\left(y_*|\boldsymbol{k}_{f*}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}, k_{**} - \boldsymbol{k}_{f*}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{k}_{f*} + \sigma_y^2\right).$$

  - → Naive computation scales with $\mathcal{O}(N^3)$.
  - → Storage scales with $\mathcal{O}(N^2)$.
- **Goal:** Mitigate (or eliminate!) the influence of N in the scaling.
- **Idea:** Maintain a smaller set of points that **summarizes the available data**.

Sparse approximations

- We have seen how GP models are flexible and can be applied to both Gaussian and non-Gaussian observations.
- **Problem:** GP expressions involve the terms $|\boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I}|$ and $(\boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I})^{-1}$:

$$\overbrace{\log p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. likelihood}} = -\frac{1}{2}\log|\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I}| - \frac{1}{2}\boldsymbol{y}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y} - \frac{N}{2}\log(2\pi),$$

$$\underbrace{p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X})}_{\text{predictive dist.}} = \mathcal{N}\left(y_*|\boldsymbol{k}_{f*}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{y}, k_{**} - \boldsymbol{k}_{f*}^\top(\boldsymbol{K}_f + \sigma_y^2\boldsymbol{I})^{-1}\boldsymbol{k}_{f*} + \sigma_y^2\right).$$

  → Naive computation scales with $\mathcal{O}(N^3)$.
  → Storage scales with $\mathcal{O}(N^2)$.
- **Goal:** Mitigate (or eliminate!) the influence of N in the scaling.
- **Idea:** Maintain a smaller set of points that **summarizes the available data**.

Sparse approximations

- Consider M **inducing variables** $\boldsymbol{u} \in \mathbb{R}^M$ from the same GP prior of $\boldsymbol{f}$.
  $\rightarrow$ $\boldsymbol{u}$ corresponds to the evaluation of $f(\cdot)$ at M **pseudo-inputs** $\boldsymbol{z}_j|_{j=1}^M \in \mathbb{R}^D$.

$$\overbrace{p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. lik.}} = \mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I}),$$
$$\underbrace{p(\boldsymbol{f}|\boldsymbol{X})}_{\text{prior}} = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f), \quad [\boldsymbol{K}_f]_{ii'} = k(\boldsymbol{x}_i, \boldsymbol{x}_{i'}),$$
$$\underbrace{p(\boldsymbol{u}|\boldsymbol{Z})}_{\text{prior}} = \mathcal{N}(\boldsymbol{u}|\boldsymbol{0}, \boldsymbol{K}_u), \quad [\boldsymbol{K}_u]_{jj'} = k(\boldsymbol{z}_j, \boldsymbol{z}_{j'}).$$

- If the knowledge from $\boldsymbol{f}$ is concentrated in $\boldsymbol{u}$, the **approximate posterior** becomes

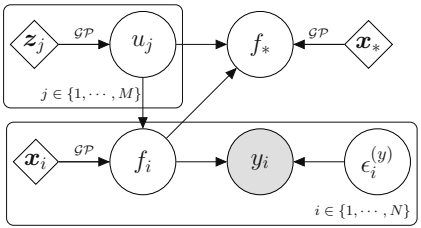
$$q(f_*) = \int \underbrace{p(f_*|\boldsymbol{u})}_{\approx p(f_*|\boldsymbol{u},\boldsymbol{f})} \underbrace{q(\boldsymbol{u})}_{\approx p(\boldsymbol{u}|\boldsymbol{y})} \, \mathrm{d}\boldsymbol{u}.$$

  $\rightarrow$ If $M = N$ and $\boldsymbol{z}_i = \boldsymbol{x}_i, \forall i$, we get $\boldsymbol{u} = \boldsymbol{f}$ and recover the original posterior exactly.

Sparse approximations

- Consider M **inducing variables** $\boldsymbol{u} \in \mathbb{R}^M$ from the same GP prior of $\boldsymbol{f}$.
  $\rightarrow$ $\boldsymbol{u}$ corresponds to the evaluation of $f(\cdot)$ at M **pseudo-inputs** $\boldsymbol{z}_j|_{j=1}^M \in \mathbb{R}^D$.

$$\overbrace{p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. lik.}} = \mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I}),$$
$$\underbrace{p(\boldsymbol{f}|\boldsymbol{X})}_{\text{prior}} = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f), \quad [\boldsymbol{K}_f]_{ii'} = k(\boldsymbol{x}_i, \boldsymbol{x}_{i'}),$$
$$\underbrace{p(\boldsymbol{u}|\boldsymbol{Z})}_{\text{prior}} = \mathcal{N}(\boldsymbol{u}|\boldsymbol{0}, \boldsymbol{K}_u), \quad [\boldsymbol{K}_u]_{jj'} = k(\boldsymbol{z}_j, \boldsymbol{z}_{j'}).$$



- If the knowledge from $\boldsymbol{f}$ is concentrated in $\boldsymbol{u}$, the **approximate posterior** becomes

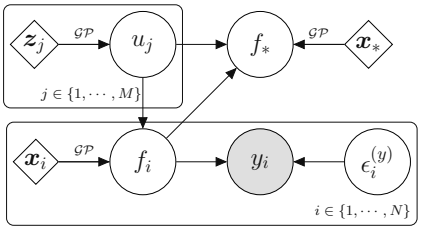
$$q(f_*) = \int \underbrace{p(f_*|\boldsymbol{u})}_{\approx p(f_*|\boldsymbol{u}, \boldsymbol{f})} \underbrace{q(\boldsymbol{u})}_{\approx p(\boldsymbol{u}|\boldsymbol{y})} \, \mathrm{d}\boldsymbol{u}.$$

  $\rightarrow$ If $M = N$ and $\boldsymbol{z}_i = \boldsymbol{x}_i, \forall i$, we get $\boldsymbol{u} = \boldsymbol{f}$ and recover the original posterior exactly.

Sparse approximations

- Consider M **inducing variables** $\boldsymbol{u} \in \mathbb{R}^M$ from the same GP prior of $\boldsymbol{f}$.
  $\rightarrow$ $\boldsymbol{u}$ corresponds to the evaluation of $f(\cdot)$ at M **pseudo-inputs** $\boldsymbol{z}_j|_{j=1}^M \in \mathbb{R}^D$.

$$\overbrace{p(\boldsymbol{y}|\boldsymbol{X})}^{\text{marg. lik.}} = \mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{K}_f + \sigma_y^2 \boldsymbol{I}),$$
$$\underbrace{p(\boldsymbol{f}|\boldsymbol{X})}_{\text{prior}} = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f), \quad [\boldsymbol{K}_f]_{ii'} = k(\boldsymbol{x}_i, \boldsymbol{x}_{i'}),$$
$$\underbrace{p(\boldsymbol{u}|\boldsymbol{Z})}_{\text{prior}} = \mathcal{N}(\boldsymbol{u}|\boldsymbol{0}, \boldsymbol{K}_u), \quad [\boldsymbol{K}_u]_{jj'} = k(\boldsymbol{z}_j, \boldsymbol{z}_{j'}).$$

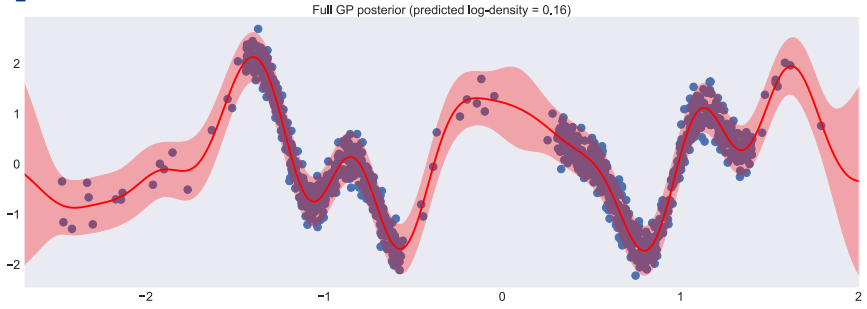


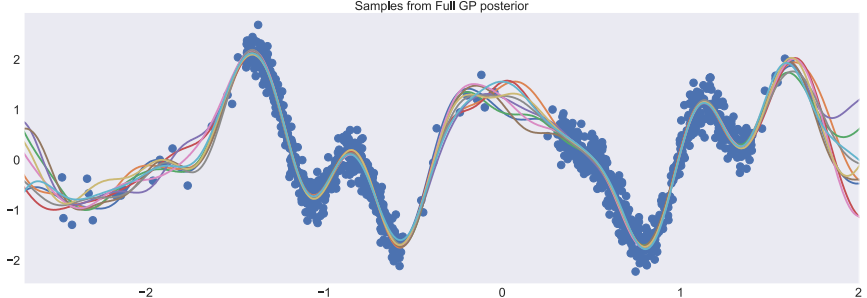
- If the knowledge from $\boldsymbol{f}$ is concentrated in $\boldsymbol{u}$, the **approximate posterior** becomes

$$q(f_*) = \int \underbrace{p(f_*|\boldsymbol{u})}_{\approx p(f_*|\boldsymbol{u},\boldsymbol{f})} \underbrace{q(\boldsymbol{u})}_{\approx p(\boldsymbol{u}|\boldsymbol{y})} \, \mathrm{d}\boldsymbol{u}.$$

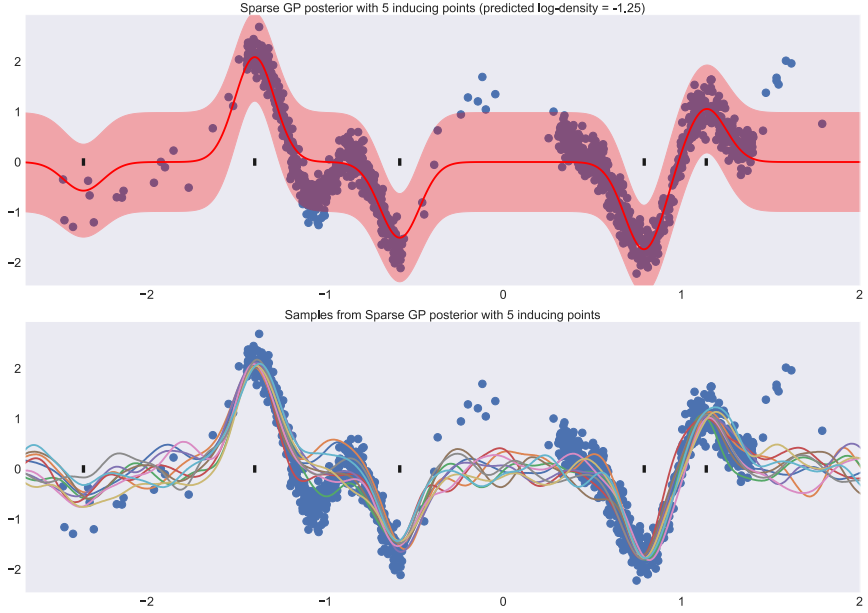
  $\rightarrow$ If $M = N$ and $\boldsymbol{z}_i = \boldsymbol{x}_i, \forall i$, we get $\boldsymbol{u} = \boldsymbol{f}$ and recover the original posterior exactly.

Full GP posterior (1083 observations)

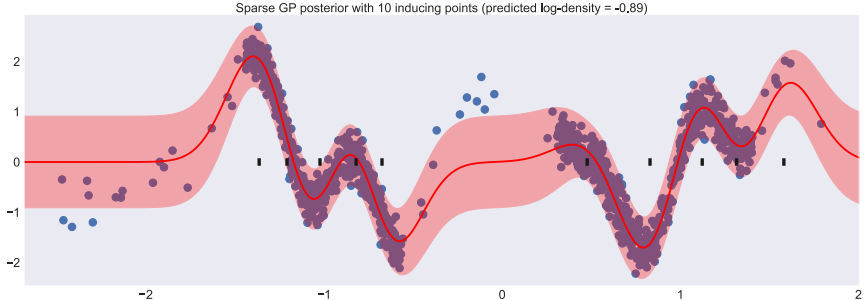


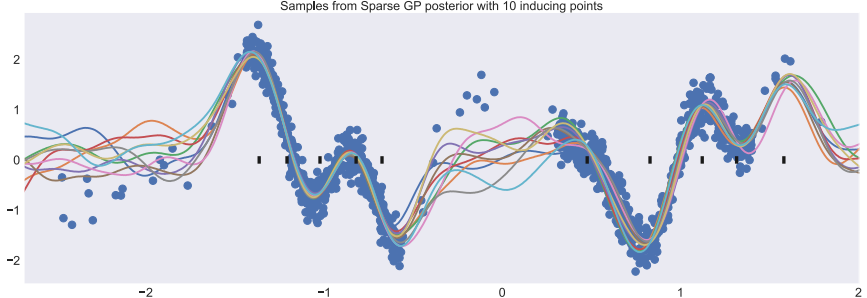


Sparse approximations (1083 observations, 5 inducing points)

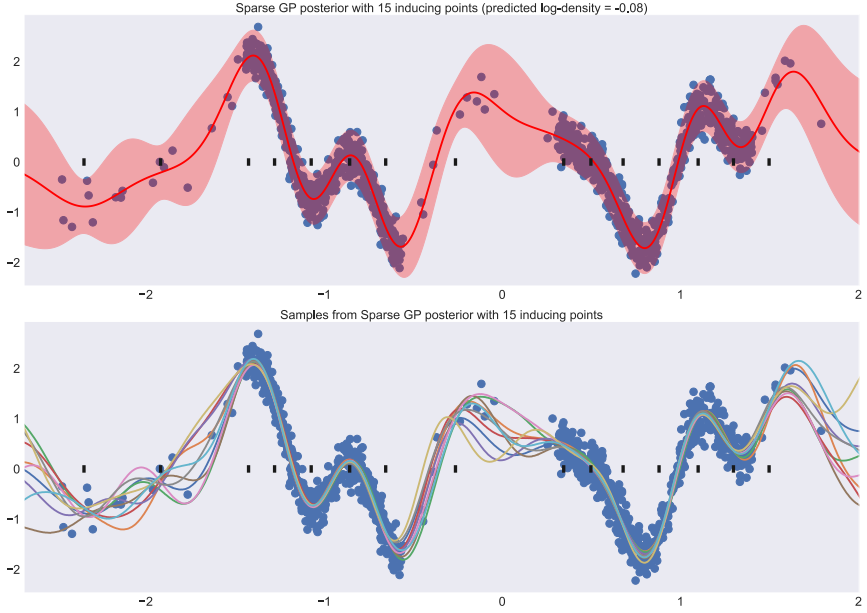


Sparse approximations (1083 observations, 10 inducing points)

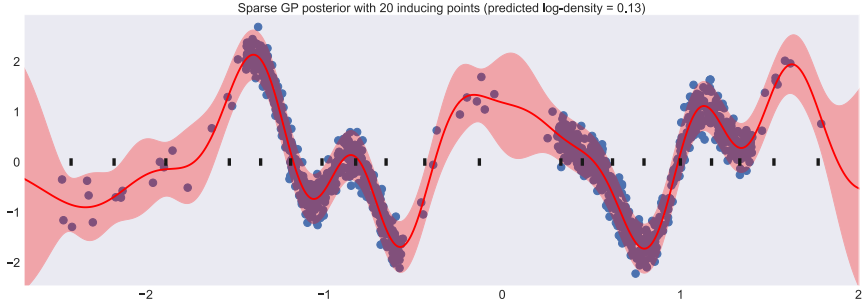


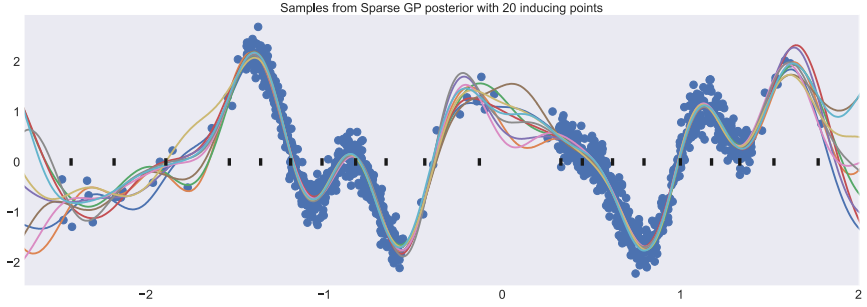


Sparse approximations (1083 observations, 15 inducing points)

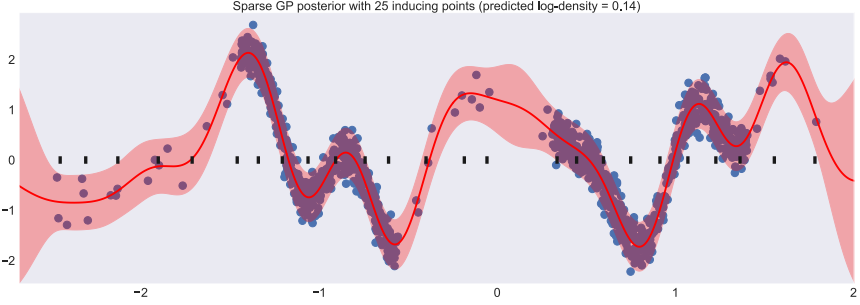


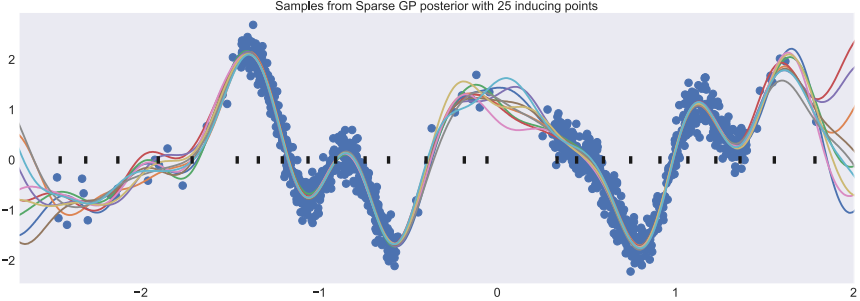
Sparse approximations (1083 observations, 20 inducing points)



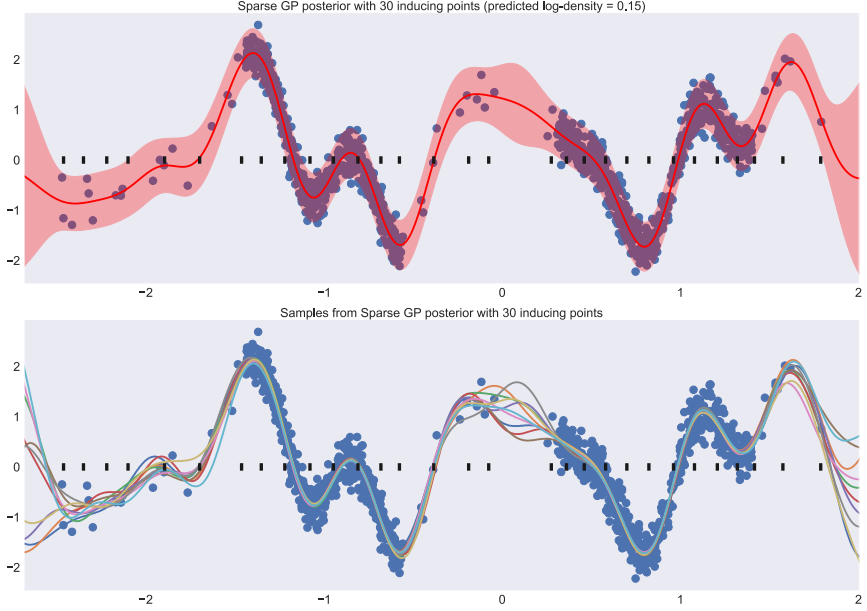


Sparse approximations (1083 observations, 25 inducing points)

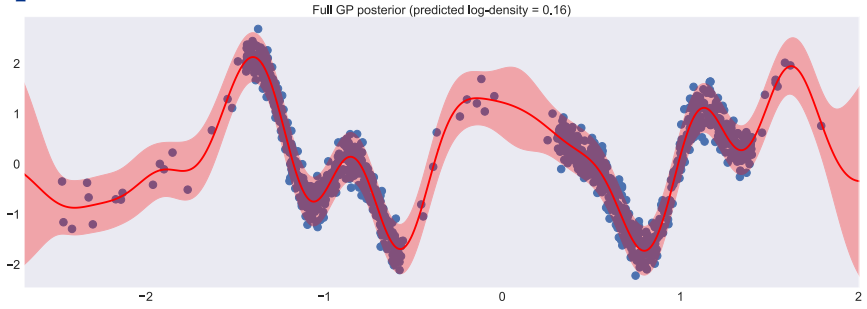


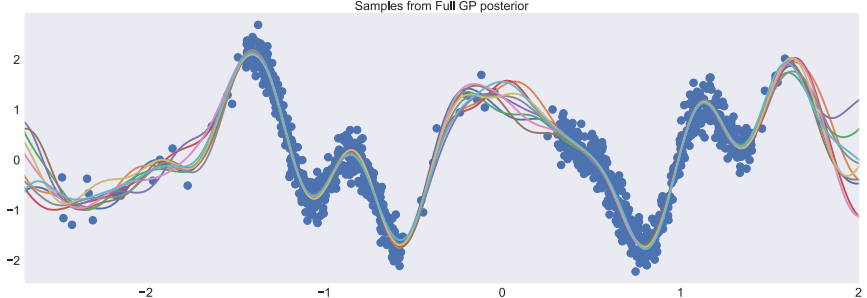


Sparse approximations (1083 observations, 30 inducing points)



Full GP posterior (1083 observations)





Sparse approximations

- Given some data, we want an approximation $q(\boldsymbol{f})$ for the true posterior $p(\boldsymbol{f}|\boldsymbol{y})$.
- The approximation quality can be quantified via the **Kullback-Leibler divergence**:

$$\begin{aligned}
\mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\boldsymbol{y})) &= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\right] \geq 0 \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\frac{p(\boldsymbol{u}|\boldsymbol{f})}{p(\boldsymbol{u}|\boldsymbol{f})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{y})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{\frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{p(\boldsymbol{y})}}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}\right] + \overbrace{\log p(\boldsymbol{y})}^{\text{evidence}} \geq 0, \\
&\implies \log p(\boldsymbol{y}) \geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right].
\end{aligned}$$

Sparse approximations

- Given some data, we want an approximation $q(\boldsymbol{f})$ for the true posterior $p(\boldsymbol{f}|\boldsymbol{y})$.
- The approximation quality can be quantified via the **Kullback-Leibler divergence**:

$$\begin{aligned}
\mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\boldsymbol{y})) &= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\right] \geq 0 \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\frac{p(\boldsymbol{u}|\boldsymbol{f})}{p(\boldsymbol{u}|\boldsymbol{f})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{y})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{\frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{p(\boldsymbol{y})}}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}\right] + \overbrace{\log p(\boldsymbol{y})}^{\text{evidence}} \geq 0, \\
&\implies \log p(\boldsymbol{y}) \geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right].
\end{aligned}$$

Sparse approximations

- Given some data, we want an approximation $q(\boldsymbol{f})$ for the true posterior $p(\boldsymbol{f}|\boldsymbol{y})$.
- The approximation quality can be quantified via the **Kullback-Leibler divergence**:

$$\begin{aligned}
\mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\boldsymbol{y})) &= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\right] \geq 0 \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\frac{p(\boldsymbol{u}|\boldsymbol{f})}{p(\boldsymbol{u}|\boldsymbol{f})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{y})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{\frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{p(\boldsymbol{y})}}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}\right] + \overbrace{\log p(\boldsymbol{y})}^{\text{evidence}} \geq 0, \\
&\implies \log p(\boldsymbol{y}) \geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right].
\end{aligned}$$

Sparse approximations

- Given some data, we want an approximation $q(\boldsymbol{f})$ for the true posterior $p(\boldsymbol{f}|\boldsymbol{y})$.
- The approximation quality can be quantified via the **Kullback-Leibler divergence**:

$$\begin{aligned}
\mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\boldsymbol{y})) &= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\right] \geq 0 \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\frac{p(\boldsymbol{u}|\boldsymbol{f})}{p(\boldsymbol{u}|\boldsymbol{f})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{y})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{\frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{p(\boldsymbol{y})}}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}\right] + \overbrace{\log p(\boldsymbol{y})}^{\text{evidence}} \geq 0, \\
&\implies \log p(\boldsymbol{y}) \geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right].
\end{aligned}$$

Sparse approximations

- Given some data, we want an approximation $q(\boldsymbol{f})$ for the true posterior $p(\boldsymbol{f}|\boldsymbol{y})$.
- The approximation quality can be quantified via the **Kullback-Leibler divergence**:

$$\begin{aligned}
\mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\boldsymbol{y})) &= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\right] \geq 0 \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\frac{p(\boldsymbol{u}|\boldsymbol{f})}{p(\boldsymbol{u}|\boldsymbol{f})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{y})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{\frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{p(\boldsymbol{y})}}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}\right] + \overbrace{\log p(\boldsymbol{y})}^{\text{evidence}} \geq 0, \\
&\implies \log p(\boldsymbol{y}) \geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right].
\end{aligned}$$

Sparse approximations

- Given some data, we want an approximation $q(\boldsymbol{f})$ for the true posterior $p(\boldsymbol{f}|\boldsymbol{y})$.
- The approximation quality can be quantified via the **Kullback-Leibler divergence**:

$$
\begin{aligned}
\mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\boldsymbol{y})) &= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\right] \geq 0 \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\frac{p(\boldsymbol{u}|\boldsymbol{f})}{p(\boldsymbol{u}|\boldsymbol{f})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{y})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{\frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{p(\boldsymbol{y})}}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}\right] + \overbrace{\log p(\boldsymbol{y})}^{\text{evidence}} \geq 0, \\
&\implies \log p(\boldsymbol{y}) \geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right].
\end{aligned}
$$

Sparse approximations

- Given some data, we want an approximation $q(\boldsymbol{f})$ for the true posterior $p(\boldsymbol{f}|\boldsymbol{y})$.
- The approximation quality can be quantified via the **Kullback-Leibler divergence**:

$$
\begin{aligned}
\mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\boldsymbol{y})) &= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\right] \geq 0 \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f})}{p(\boldsymbol{f}|\boldsymbol{y})}\frac{p(\boldsymbol{u}|\boldsymbol{f})}{p(\boldsymbol{u}|\boldsymbol{f})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{f},\boldsymbol{u}|\boldsymbol{y})}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{\frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{p(\boldsymbol{y})}}\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{q(\boldsymbol{f},\boldsymbol{u})}{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}\right] + \overbrace{\log p(\boldsymbol{y})}^{\text{evidence}} \geq 0, \\
&\implies \log p(\boldsymbol{y}) \geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right].
\end{aligned}
$$

Sparse approximations

- We replace $q(\boldsymbol{f}) = \int q(\boldsymbol{f}, \boldsymbol{u}) \mathrm{d}\boldsymbol{u}$ and (conveniently) choose $q(\boldsymbol{f}, \boldsymbol{u}) = q(\boldsymbol{u}) \underbrace{p(\boldsymbol{f}|\boldsymbol{u})}_{\text{cond. prior}}$:

$$
\begin{aligned}
\underbrace{\log p(\boldsymbol{y})}_{\text{evidence}} &\geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})\cancel{p(\boldsymbol{f}|\boldsymbol{u})}p(\boldsymbol{u})}{q(\boldsymbol{u})\cancel{p(\boldsymbol{f}|\boldsymbol{u})}}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{u})}{q(\boldsymbol{u})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}\left[\log p(\boldsymbol{y}|\boldsymbol{f})\right] + \mathbb{E}_{q(\boldsymbol{u})}\left[\log \frac{p(\boldsymbol{u})}{q(\boldsymbol{u})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}[\log p(\boldsymbol{y}|\boldsymbol{f})] - \mathrm{KL}(q(\boldsymbol{u})||p(\boldsymbol{u})) = \underbrace{\mathcal{L}}_{\text{ELBO}}.
\end{aligned}
$$

- The kernel hyperparameters and the pseudo-inputs $\boldsymbol{z}_j|_1^M$ can be optimized by maximizing the **ELBO (evidence lower bound)** $\mathcal{L}$, improving the approximation.

Sparse approximations

- We replace $q(\boldsymbol{f}) = \int q(\boldsymbol{f}, \boldsymbol{u}) \mathrm{d}\boldsymbol{u}$ and (conveniently) choose $q(\boldsymbol{f}, \boldsymbol{u}) = q(\boldsymbol{u})\ \underbrace{p(\boldsymbol{f}|\boldsymbol{u})}_{\text{cond. prior}}$:

$$
\begin{aligned}
\underbrace{\log p(\boldsymbol{y})}_{\text{evidence}} &\geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})\cancel{p(\boldsymbol{f}|\boldsymbol{u})}p(\boldsymbol{u})}{q(\boldsymbol{u})\cancel{p(\boldsymbol{f}|\boldsymbol{u})}}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{u})}{q(\boldsymbol{u})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}\left[\log p(\boldsymbol{y}|\boldsymbol{f})\right] + \mathbb{E}_{q(\boldsymbol{u})}\left[\log \frac{p(\boldsymbol{u})}{q(\boldsymbol{u})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}[\log p(\boldsymbol{y}|\boldsymbol{f})] - \mathrm{KL}(q(\boldsymbol{u})||p(\boldsymbol{u})) = \underbrace{\mathcal{L}}_{\text{ELBO}}.
\end{aligned}
$$

- The kernel hyperparameters and the pseudo-inputs $\boldsymbol{z}_j|_1^M$ can be optimized by maximizing the **ELBO (evidence lower bound)** $\mathcal{L}$, improving the approximation.

Sparse approximations

- We replace $q(\boldsymbol{f}) = \int q(\boldsymbol{f}, \boldsymbol{u}) \mathrm{d}\boldsymbol{u}$ and (conveniently) choose $q(\boldsymbol{f}, \boldsymbol{u}) = q(\boldsymbol{u})\ \underbrace{p(\boldsymbol{f}|\boldsymbol{u})}_{\text{cond. prior}}$:

$$
\begin{aligned}
\underbrace{\log p(\boldsymbol{y})}_{\text{evidence}} &\geq \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})}{q(\boldsymbol{f},\boldsymbol{u})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})\cancel{p(\boldsymbol{f}|\boldsymbol{u})}p(\boldsymbol{u})}{q(\boldsymbol{u})\cancel{p(\boldsymbol{f}|\boldsymbol{u})}}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{u})}{q(\boldsymbol{u})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}\left[\log p(\boldsymbol{y}|\boldsymbol{f})\right] + \mathbb{E}_{q(\boldsymbol{u})}\left[\log \frac{p(\boldsymbol{u})}{q(\boldsymbol{u})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}[\log p(\boldsymbol{y}|\boldsymbol{f})] - \mathrm{KL}(q(\boldsymbol{u})||p(\boldsymbol{u})) = \underbrace{\mathcal{L}}_{\text{ELBO}}\,.
\end{aligned}
$$

- The kernel hyperparameters and the pseudo-inputs $\boldsymbol{z}_j|_1^M$ can be optimized by maximizing the **ELBO (evidence lower bound)** $\mathcal{L}$, improving the approximation.

Sparse approximations

- What we need to compute the ELBO $\mathcal{L}$?

$$\log p(\boldsymbol{y}) \geq \mathcal{L} = \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}[\log p(\boldsymbol{y}|\boldsymbol{f})] - \mathrm{KL}(q(\boldsymbol{u})||p(\boldsymbol{u})).$$

- Since the joint $p(\boldsymbol{f}, \boldsymbol{u})$ is Gaussian, the conditional $p(\boldsymbol{f}|\boldsymbol{u})$ is also Gaussian and given by (with omitted dependence on $\boldsymbol{X}$ and $\boldsymbol{z}$):

$$p(\boldsymbol{f}, \boldsymbol{u}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{u} \end{bmatrix} \middle| \boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_f & \boldsymbol{K}_{fu} \\ \boldsymbol{K}_{fu}^\top & \boldsymbol{K}_u \end{bmatrix}\right), \qquad [\boldsymbol{K}_{fu}]_{ij} = k(\boldsymbol{x}_i, \boldsymbol{z}_j),$$

$$p(\boldsymbol{f}|\boldsymbol{u}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{u}, \boldsymbol{K}_f - \boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{fu}^\top).$$

- How to define $q(\boldsymbol{u})$?

Sparse approximations

- What we need to compute the ELBO $\mathcal{L}$?

$$\log p(\boldsymbol{y}) \geq \mathcal{L} = \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}[\log p(\boldsymbol{y}|\boldsymbol{f})] - \mathrm{KL}(q(\boldsymbol{u})||p(\boldsymbol{u})).$$

- Since the joint $p(\boldsymbol{f}, \boldsymbol{u})$ is Gaussian, the conditional $p(\boldsymbol{f}|\boldsymbol{u})$ is also Gaussian and given by (with omitted dependence on $\boldsymbol{X}$ and $\boldsymbol{z}$):

$$p(\boldsymbol{f}, \boldsymbol{u}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{u} \end{bmatrix} \middle| \mathbf{0}, \begin{bmatrix} \boldsymbol{K}_f & \boldsymbol{K}_{fu} \\ \boldsymbol{K}_{fu}^\top & \boldsymbol{K}_u \end{bmatrix}\right), \qquad [\boldsymbol{K}_{fu}]_{ij} = k(\boldsymbol{x}_i, \boldsymbol{z}_j),$$

$$p(\boldsymbol{f}|\boldsymbol{u}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{u}, \boldsymbol{K}_f - \boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{fu}^\top).$$

- How to define $q(\boldsymbol{u})$?

Sparse approximations

- What we need to compute the ELBO $\mathcal{L}$?

$$\log p(\boldsymbol{y}) \geq \mathcal{L} = \mathbb{E}_{q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})}[\log p(\boldsymbol{y}|\boldsymbol{f})] - \mathrm{KL}(q(\boldsymbol{u})||p(\boldsymbol{u})).$$

- Since the joint $p(\boldsymbol{f}, \boldsymbol{u})$ is Gaussian, the conditional $p(\boldsymbol{f}|\boldsymbol{u})$ is also Gaussian and given by (with omitted dependence on $\boldsymbol{X}$ and $\boldsymbol{z}$):

$$p(\boldsymbol{f}, \boldsymbol{u}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{u} \end{bmatrix} \middle| \mathbf{0}, \begin{bmatrix} \boldsymbol{K}_f & \boldsymbol{K}_{fu} \\ \boldsymbol{K}_{fu}^\top & \boldsymbol{K}_u \end{bmatrix}\right), \qquad [\boldsymbol{K}_{fu}]_{ij} = k(\boldsymbol{x}_i, \boldsymbol{z}_j),$$
$$p(\boldsymbol{f}|\boldsymbol{u}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{u}, \boldsymbol{K}_f - \boldsymbol{K}_{fu}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{fu}^\top).$$

- How to define $q(\boldsymbol{u})$?

Sparse Variational GP

- The original sparse variational approach (Titsias, 2009) analytically optimizes $q(\boldsymbol{u})$:

$$
\begin{aligned}
q(\boldsymbol{u}) &= \mathcal{N}(\boldsymbol{u}|\hat{\boldsymbol{m}}, \hat{\boldsymbol{S}}), \\
\hat{\boldsymbol{m}} &= \frac{1}{\sigma_y^2}\boldsymbol{K}_u \left(\boldsymbol{K}_u + \frac{1}{\sigma_y^2}\boldsymbol{K}_{fu}^\top \boldsymbol{K}_{fu}\right)^{-1} \boldsymbol{K}_{fu}^\top \boldsymbol{y}, \\
\hat{\boldsymbol{S}} &= \boldsymbol{K}_u \left(\boldsymbol{K}_u + \frac{1}{\sigma_y^2}\boldsymbol{K}_{fu}^\top \boldsymbol{K}_{fu}\right)^{-1} \boldsymbol{K}_u.
\end{aligned}
$$

- The result is a **collapsed bound** (considering a Gaussian likelihood):

$$
\mathcal{L}_{\text{collapsed}} = \log \mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \sigma_y^2 \boldsymbol{I} + \boldsymbol{K}_{fz}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{fz}^\top) - \frac{1}{2\sigma_y^2}\operatorname{Tr}(\boldsymbol{K}_f - \boldsymbol{K}_{fz}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{fz}^\top).
$$

$\rightarrow$ Computation and storage scale with $\mathcal{O}(NM^2)$ and $\mathcal{O}(NM)$.

Sparse Variational GP

- The original sparse variational approach (Titsias, 2009) analytically optimizes $q(\boldsymbol{u})$:

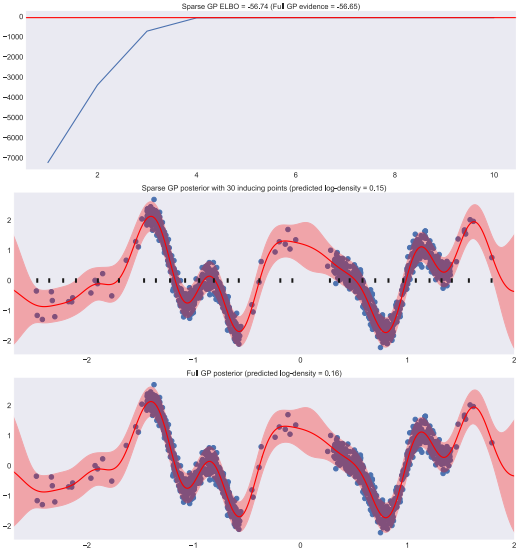
$$
\begin{aligned}
q(\boldsymbol{u}) &= \mathcal{N}(\boldsymbol{u}|\hat{\boldsymbol{m}}, \hat{\boldsymbol{S}}), \\
\hat{\boldsymbol{m}} &= \frac{1}{\sigma_y^2}\boldsymbol{K}_u \left(\boldsymbol{K}_u + \frac{1}{\sigma_y^2}\boldsymbol{K}_{fu}^\top \boldsymbol{K}_{fu}\right)^{-1} \boldsymbol{K}_{fu}^\top \boldsymbol{y}, \\
\hat{\boldsymbol{S}} &= \boldsymbol{K}_u \left(\boldsymbol{K}_u + \frac{1}{\sigma_y^2}\boldsymbol{K}_{fu}^\top \boldsymbol{K}_{fu}\right)^{-1} \boldsymbol{K}_u.
\end{aligned}
$$

- The result is a **collapsed bound** (considering a Gaussian likelihood):

$$
\mathcal{L}_{\text{collapsed}} = \log \mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \sigma_y^2 \boldsymbol{I} + \boldsymbol{K}_{fz}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{fz}^\top) - \frac{1}{2\sigma_y^2}\operatorname{Tr}(\boldsymbol{K}_f - \boldsymbol{K}_{fz}\boldsymbol{K}_u^{-1}\boldsymbol{K}_{fz}^\top).
$$

$\rightarrow$ Computation and storage scale with $\mathcal{O}(NM^2)$ and $\mathcal{O}(NM)$.

Sparse Variational GP



Observations

- The ELBO **monotonically increases**, which always **improves** the approximation.
- The gradients of the ELBO can be computed **analytically**.
- The exact gradients enable the use of **efficient optimization methods**, e.g. L-BFGS.

Stochastic Variational Inference for GPs

- SVI requires a set of **global parameters** and a **factorized ELBO**.
- SVGP explicitly parameterizes $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{m}, \boldsymbol{S})$ and obtains an **uncollapsed bound** that enables minibatch updates (Hensman *et al.*, 2013):

$$\mathcal{L}_{\text{uncollapsed}} = \sum_{i=1} \mathcal{L}_i - \text{KL}(q(\boldsymbol{u})||p(\boldsymbol{u})),$$

$$\text{where } \mathcal{L}_i = \log \mathcal{N}(y_i|\boldsymbol{k}_i^\top \boldsymbol{K}_u^{-1}\boldsymbol{m}, \sigma_y^2) - \frac{1}{2\sigma_y^2}([\boldsymbol{K}_f]_{ii} - \boldsymbol{k}_i^\top \boldsymbol{K}_u^{-1}\boldsymbol{k}_i) - \frac{1}{2}\,\text{Tr}\left(\frac{1}{\sigma_y^2}\boldsymbol{S}\boldsymbol{K}_u^{-1}\boldsymbol{k}_i\boldsymbol{k}_i^\top\right)$$

$$\text{and } \boldsymbol{k}_i^\top = [\boldsymbol{K}_{fz}]_{i:}.$$

$\rightarrow$ Given a minibatch size B, computation and storage scale with $\mathcal{O}(BM^2 + M^3)$ and $\mathcal{O}(BM + M^2)$.

Stochastic Variational Inference for GPs

- SVI requires a set of **global parameters** and a **factorized ELBO**.
- SVGP explicitly parameterizes $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{m}, \boldsymbol{S})$ and obtains an **uncollapsed bound** that enables minibatch updates (Hensman *et al.*, 2013):

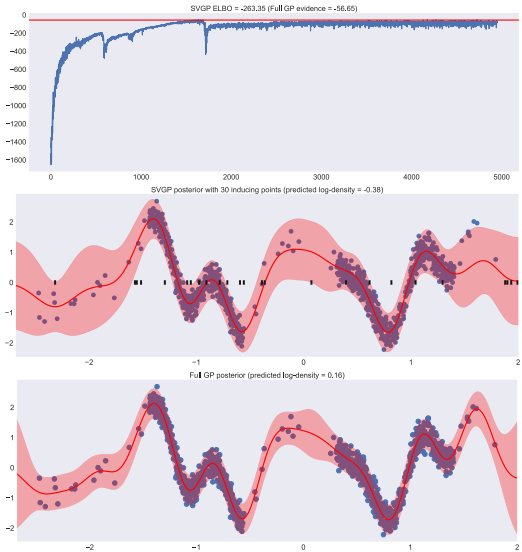
$$\mathcal{L}_{\text{uncollapsed}} = \sum_{i=1} \mathcal{L}_i - \text{KL}(q(\boldsymbol{u})||p(\boldsymbol{u})),$$

$$\text{where } \mathcal{L}_i = \log \mathcal{N}(y_i|\boldsymbol{k}_i^\top \boldsymbol{K}_u^{-1}\boldsymbol{m}, \sigma_y^2) - \frac{1}{2\sigma_y^2}([\boldsymbol{K}_f]_{ii} - \boldsymbol{k}_i^\top \boldsymbol{K}_u^{-1}\boldsymbol{k}_i) - \frac{1}{2}\,\text{Tr}\left(\frac{1}{\sigma_y^2}\boldsymbol{S}\boldsymbol{K}_u^{-1}\boldsymbol{k}_i\boldsymbol{k}_i^\top\right)$$

$$\text{and } \boldsymbol{k}_i^\top = [\boldsymbol{K}_{fz}]_{i:}.$$

$\rightarrow$ Given a minibatch size B, computation and storage scale with $\mathcal{O}(BM^2 + M^3)$ and $\mathcal{O}(BM + M^2)$.

Stochastic Variation Inference for GPs



Observations

- The ELBO **does not increase monotonically** due to the use of minibatches.
- The **noisy** gradients of the ELBO can be computed **analytically**.
- More complex **stochastic optimization methods** may be used, e.g. Adam.

Sparse approximations

- In both cases, prediction is performed by integrating out the inducing variables $\boldsymbol{u}$ using the **variational posterior** $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{m}, \boldsymbol{S})$:

$$
\begin{aligned}
q(f_*) &= \int p(f_*|\boldsymbol{u}) q(\boldsymbol{u}) \mathrm{d}\boldsymbol{u} \\
&= \int \mathcal{N}(f_* |\ \boldsymbol{k}_{*z}\boldsymbol{K}_u^{-1}\boldsymbol{u}, K_* - \boldsymbol{k}_{*z}\boldsymbol{K}_u^{-1}\boldsymbol{k}_{*z}^\top) \mathcal{N}(\boldsymbol{u}|\boldsymbol{m}, \boldsymbol{S}) \mathrm{d}\boldsymbol{u} \\
&= \mathcal{N}\left(f_* \middle| \boldsymbol{k}_{*z}\boldsymbol{K}_u^{-1}\boldsymbol{m}, K_* - \boldsymbol{k}_{*z}\boldsymbol{K}_u^{-1}(\boldsymbol{K}_u - \boldsymbol{S})\boldsymbol{K}_u^{-1}\boldsymbol{k}_{*z}^\top\right).
\end{aligned}
$$

- Recall that:
  - → In the **collapsed bound**, the moments $\hat{m}$ and $\hat{S}$ are **optimally obtained**.
  - → In the **uncollapsed bound**, the moments $\boldsymbol{m}$ and $\boldsymbol{S}$ are variational parameters **updated iteratively** using the ELBO.

Sparse approximations

- In both cases, prediction is performed by integrating out the inducing variables $\boldsymbol{u}$ using the **variational posterior** $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{m}, \boldsymbol{S})$:

$$
\begin{aligned}
q(f_*) &= \int p(f_*|\boldsymbol{u}) q(\boldsymbol{u}) \mathrm{d}\boldsymbol{u} \\
&= \int \mathcal{N}(f_* |\ \boldsymbol{k}_{*z}\boldsymbol{K}_u^{-1}\boldsymbol{u}, K_* - \boldsymbol{k}_{*z}\boldsymbol{K}_u^{-1}\boldsymbol{k}_{*z}^\top) \mathcal{N}(\boldsymbol{u}|\boldsymbol{m}, \boldsymbol{S}) \mathrm{d}\boldsymbol{u} \\
&= \mathcal{N}\left(f_* \middle| \boldsymbol{k}_{*z}\boldsymbol{K}_u^{-1}\boldsymbol{m}, K_* - \boldsymbol{k}_{*z}\boldsymbol{K}_u^{-1}(\boldsymbol{K}_u - \boldsymbol{S})\boldsymbol{K}_u^{-1}\boldsymbol{k}_{*z}^\top\right).
\end{aligned}
$$

- Recall that:
  - → In the **collapsed bound**, the moments $\hat{\boldsymbol{m}}$ and $\hat{\boldsymbol{S}}$ are **optimally obtained**.
  - → In the **uncollapsed bound**, the moments $\boldsymbol{m}$ and $\boldsymbol{S}$ are variational parameters **updated iteratively** using the ELBO.

Sparse approximations

Summary

- Sparse models enable scaling GP models to **large datasets**.
- We considered $\underbrace{\textbf{inducing variables}}_{\text{integrated out}}$ and corresponding $\underbrace{\textbf{pseudo-inputs}}_{\text{optimized}}$.
- A **variational approach** approximates the posterior of the inducing variables and obtains a lower bound to the model evidence, the **ELBO**.
- Kernel hyperparameters and variational parameters are **optimized** using the ELBO.
- Two ELBO's: **collapsed** (Titsias, 2009) and **uncollapsed** (Hensman *et al.*, 2013).
- Predictions are performed using the **variational posterior** of the inducing points.

Table of Contents

Global Bayesian optimization with GPs

- Consider the task of optimizing a function $f(\cdot)$ **whose expression is unknown**.
- We are able to evaluate $f(\cdot)$ in a **noisy** way by observing $y_i = f(\boldsymbol{x}_i) + \epsilon_i$ for any given input $\boldsymbol{x}_i$ and a random noise ϵ_i.
- Some examples:
  - $\rightarrow$ Locate the best position to extract some ore;
  - $\rightarrow$ Minimize the resource consumption in an industrial process;
  - $\rightarrow$ Configure a laboratory experiment;
  - $\rightarrow$ Maximize the access speed in a database;
  - $\rightarrow$ Select hyperparameters for a machine learning model.
- If the analytical form of $f(\cdot)$ is unknown, how will we maximize or minimize it?

Global Bayesian optimization with GPs

- Consider the task of optimizing a function $f(\cdot)$ **whose expression is unknown**.
- We are able to evaluate $f(\cdot)$ in a **noisy** way by observing $y_i = f(\boldsymbol{x}_i) + \epsilon_i$ for any given input $\boldsymbol{x}_i$ and a random noise ϵ_i.
- Some examples:
  - $\rightarrow$ Locate the best position to extract some ore;
  - $\rightarrow$ Minimize the resource consumption in an industrial process;
  - $\rightarrow$ Configure a laboratory experiment;
  - $\rightarrow$ Maximize the access speed in a database;
  - $\rightarrow$ Select hyperparameters for a machine learning model.
- If the analytical form of $f(\cdot)$ is unknown, how will we maximize or minimize it?

Global Bayesian optimization with GPs

- Consider the task of optimizing a function $f(\cdot)$ **whose expression is unknown**.
- We are able to evaluate $f(\cdot)$ in a **noisy** way by observing $y_i = f(\boldsymbol{x}_i) + \epsilon_i$ for any given input $\boldsymbol{x}_i$ and a random noise ϵ_i.
- Some examples:
  - → Locate the best position to extract some ore;
  - → Minimize the resource consumption in an industrial process;
  - → Configure a laboratory experiment;
  - → Maximize the access speed in a database;
  - → Select hyperparameters for a machine learning model.
- If the analytical form of $f(\cdot)$ is unknown, how will we maximize or minimize it?

Global Bayesian optimization with GPs

- Consider N experiments that result in N pairs $(\boldsymbol{x}_i, y_i)|_{i=1}^N$, where y_i is an observation added with Gaussian noise.
- We choose a GP prior with kernel function $k(\cdot, \cdot)$ for the function $f(\cdot)$ that we want to optimize:

$$
\begin{aligned}
p(\boldsymbol{y}|\boldsymbol{f}) &= \mathcal{N}(\boldsymbol{y}|\boldsymbol{f}, \sigma_y^2 \boldsymbol{I}), \\
p(\boldsymbol{f}|\boldsymbol{X}) &= \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}), \\
\text{where } K_{ij} &= k(\boldsymbol{x}_i, \boldsymbol{x}_j).
\end{aligned}
$$

- As we have seen before, this model is **conjugate** and all the distributions of interest are **analytical**.

Global Bayesian optimization with GPs

- Predictions in this model are well defined Gaussian distributions:

$$
\begin{aligned}
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}(f_*|\mu_*, \sigma_*^2), \\
\mu_* &= \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{y}, \\
\sigma_*^2 &= k_{**} - \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{f*}.
\end{aligned}
$$

- The predictive distribution informs the **uncertainty** related to the evaluation of the unknown function in the input $\boldsymbol{x}_*$.
- In a global minimization task, for an input search domain $\mathcal{X}$, we seek:
  - $\rightarrow$ Regions with lower predictive mean values μ_* (*exploitation*);
  - $\rightarrow$ Regions with high predictive uncertainty σ_*^2 (*exploration*).
- We assume that good solutions are **close to each other** in the input space.

Global Bayesian optimization with GPs

- Predictions in this model are well defined Gaussian distributions:

$$
\begin{aligned}
p(f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}(f_*|\mu_*, \sigma_*^2), \\
\mu_* &= \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{y}, \\
\sigma_*^2 &= k_{**} - \boldsymbol{k}_{f*}^\top (\boldsymbol{K} + \sigma_y^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{f*}.
\end{aligned}
$$

- The predictive distribution informs the **uncertainty** related to the evaluation of the unknown function in the input $\boldsymbol{x}_*$.
- In a global minimization task, for an input search domain $\mathcal{X}$, we seek:
  - $\rightarrow$ Regions with lower predictive mean values μ_* (*exploitation*);
  - $\rightarrow$ Regions with high predictive uncertainty σ_*^2 (*exploration*).
- We assume that good solutions are **close to each other** in the input space.

Global Bayesian optimization with GPs

- How do we know if a given point is located in a region of interest?
- We want to use all the information provided by past evaluations when querying the next candidate.
- **Idea**: We use an **acquisition function** to **quantify the relevance** of a given candidate.
- The acquisition function $a : \mathcal{X} \to \mathbb{R}^+$ is chosen to be easily computed.
- The next point to be evaluated is given by

$$\boldsymbol{x}_* = \arg\max a(\boldsymbol{x}).$$

- Examples of usual acquisition functions:
  - → Probability of Improvement;
  - → Expected Improvement;
  - → GP Lower Confidence Bound.

Global Bayesian optimization with GPs

- How do we know if a given point is located in a region of interest?
- We want to use all the information provided by past evaluations when querying the next candidate.
- **Idea**: We use an **acquisition function** to **quantify the relevance** of a given candidate.
- The acquisition function $a : \mathcal{X} \to \mathbb{R}^+$ is chosen to be easily computed.
- The next point to be evaluated is given by

$$\boldsymbol{x}_* = \arg\max a(\boldsymbol{x}).$$

- Examples of usual acquisition functions:
  - → Probability of Improvement;
  - → Expected Improvement;
  - → GP Lower Confidence Bound.

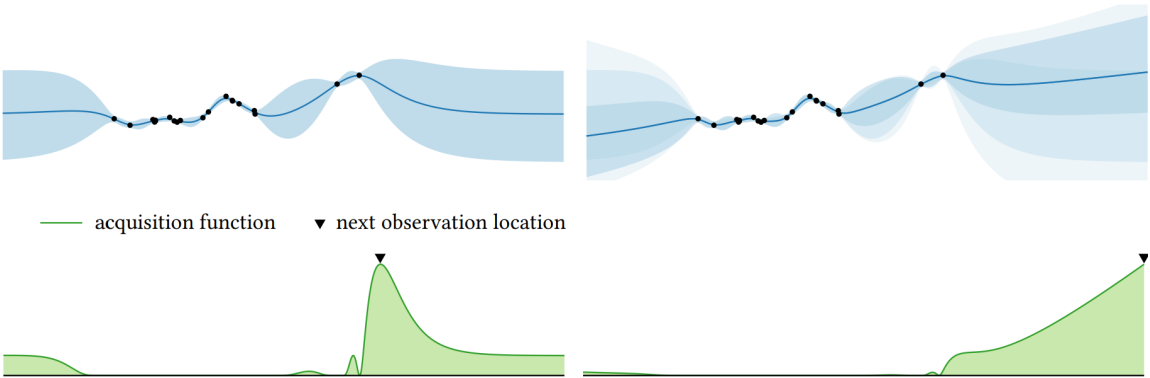
Global Bayesian optimization with GPs

- How do we know if a given point is located in a region of interest?
- We want to use all the information provided by past evaluations when querying the next candidate.
- **Idea**: We use an **acquisition function** to **quantify the relevance** of a given candidate.
- The acquisition function $a : \mathcal{X} \to \mathbb{R}^+$ is chosen to be easily computed.
- The next point to be evaluated is given by

$$\boldsymbol{x}_* = \arg\max a(\boldsymbol{x}).$$

- Examples of usual acquisition functions:
  - → Probability of Improvement;
  - → Expected Improvement;
  - → GP Lower Confidence Bound.

Illustration of Bayesian optimization steps



BO for a maximization task. Taken from Garnett (2023).

Acquisition functions - Probability of Improvement

- Aims to maximize the probability of finding points better than the best current solution $\mu^- = \min_i \mu(\boldsymbol{x}_i)$, where $\mu(\boldsymbol{x}_i)$ denotes the mean GP prediction in $\boldsymbol{x}_i$.
- Focus in *exploitation*, with balance determined by the hyperparameter $\xi \geq 0$ (usual value $\xi = 0.01$):

$$\mathrm{PI}(\boldsymbol{x}) = P(f(\boldsymbol{x}) \leq \mu^- - \xi) = \Phi\left(\frac{\mu^- - \xi - \mu(\boldsymbol{x})}{\sigma(\boldsymbol{x})}\right),$$

  where $\sigma(\boldsymbol{x}_i)$ denotes the GP predictive standard deviation in $\boldsymbol{x}_i$ and $\Phi(\cdot)$ is the cumulative distribution function of the normalized Gaussian:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp\left(-\frac{1}{2}t^2\right) \mathrm{d}t.$$

Acquisition functions - Expected Improvement

- Aims to maximize the magnitude of the expected progress when compared to the best current solution μ^-.
- Considering the hyperparameter $\xi \geq 0$ (usual value $\xi = 0.01$):

$$\text{EI}(\boldsymbol{x}) = \begin{cases} 0. & \text{if } \sigma(\boldsymbol{x}) = 0; \\ \tau(\boldsymbol{x})\Phi\left(\frac{\tau(\boldsymbol{x})}{\sigma(\boldsymbol{x})}\right) + \sigma(\boldsymbol{x})\phi\left(\frac{\tau(\boldsymbol{x})}{\sigma(\boldsymbol{x})}\right), & \text{if } \sigma(\boldsymbol{x}) > 0, \end{cases}$$

$$\text{where } \tau(\boldsymbol{x}) = \mu^- - \xi - \mu(\boldsymbol{x}).$$

Above, $\Phi(\cdot)$ and $\phi(\cdot)$ are respectively the cumulative and density functions of the normalized Gaussian.

Acquisition functions - GP Lower Confidence Bound

- Uses the GP confidence intervals to choose the next query point.
- Considering the hyperparameter $\kappa > 0$:

$$\text{LCB}(\boldsymbol{x}) = \mu(\boldsymbol{x}) - \kappa\sigma(\boldsymbol{x}),$$

where $\kappa = \sqrt{\nu\beta_t}$, $\beta_t = 2\log(t^{D/2+2}\pi^2/3\delta)$, t is the current optimization iteration, and $\nu, \delta > 0$. Usual values: $\delta = 0.1$ and $\nu = 0.2$.

Global Bayesian optimization with GPs

Summary of the algorithm

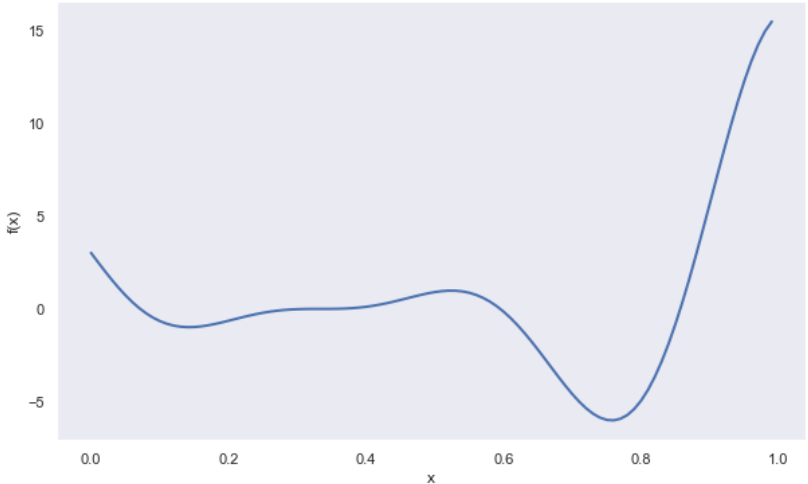
1. Define the initial dataset $\mathcal{D}_0 = (\boldsymbol{x}_i, y_i)|_{i=1}^{N_0}$;
2. Define the acquisition function $a(\boldsymbol{x})$;
3. Initialize the GP model using $\mathcal{D}_0$;
4. Repeat until convergence or until the maximum number of iterations is reached:
    1. Find the next query point by maximizing the acquisition function:

$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}} a(\boldsymbol{x}|\mathcal{D}_{t-1});$$

    2. Evaluate $y_t = f(\boldsymbol{x}_t) + \epsilon_t$;
    3. Augment the dataset $\mathcal{D}_t = \{\mathcal{D}_{t-1}, (\boldsymbol{x}_t, y_t)\}$;
    4. Update the GP model with the new dataset;
5. Return the point $\boldsymbol{x}_t$ with the best corresponding y_t.

Global Bayesian optimization with GPs

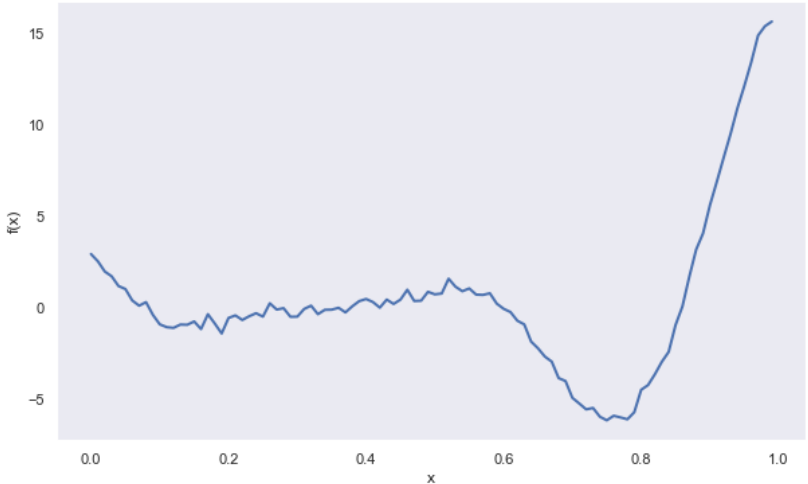
1D example



$$f(x) = (6x - 2)^2 \sin(12x - 4)$$

Global Bayesian optimization with GPs

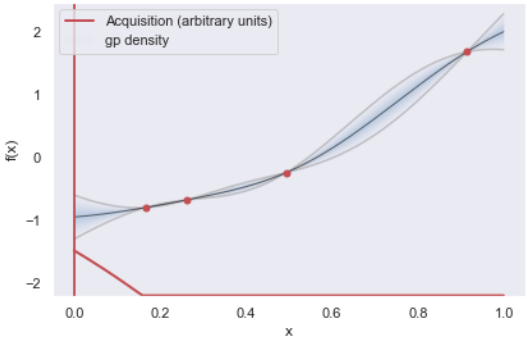
1D example



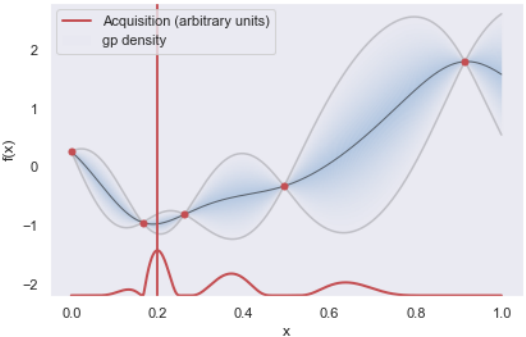
$$f(x) = (6x - 2)^2 \sin(12x - 4) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, 0.25^2)$$

Global Bayesian optimization with GPs

1D example



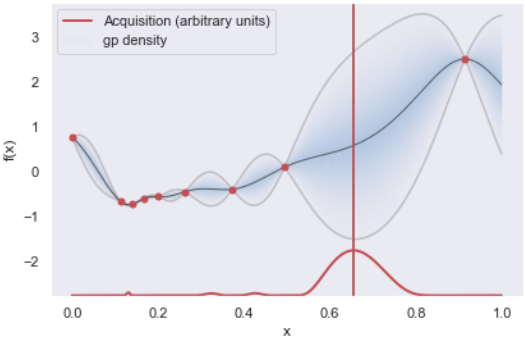
4 initial points



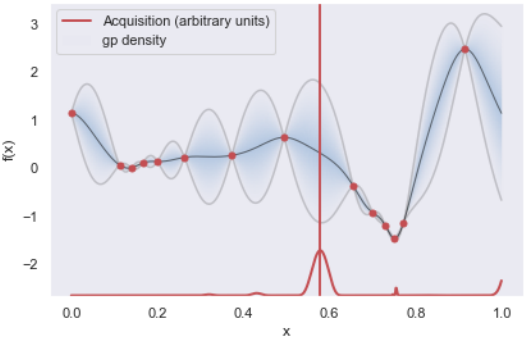
After 1 iteration

Global Bayesian optimization with GPs

1D example



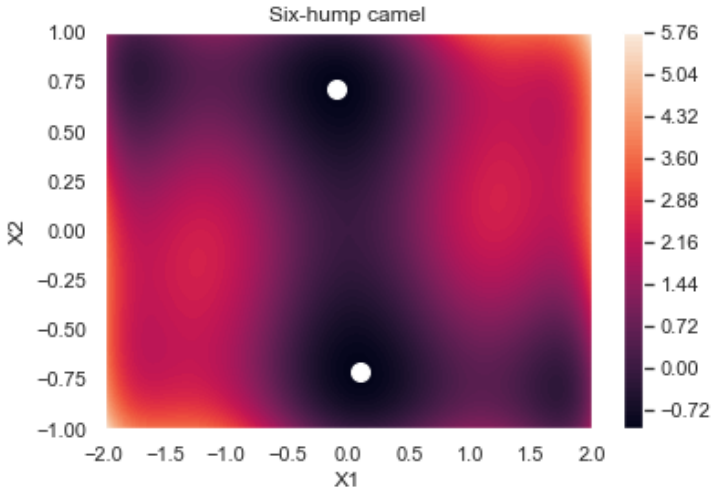
After 5 iterations



After 10 iterations

Global Bayesian optimization with GPs

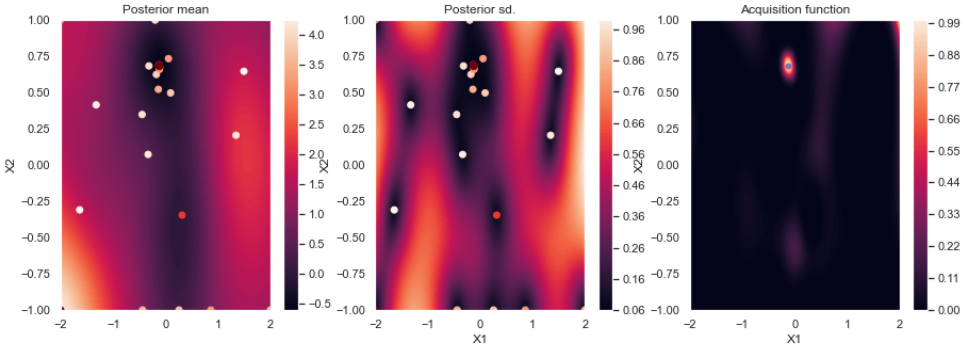
2D example



$$f(x_1, x_2) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right) x_1^2 + x_1 x_2 + (-4 + 4x_2^2)x_2^2$$

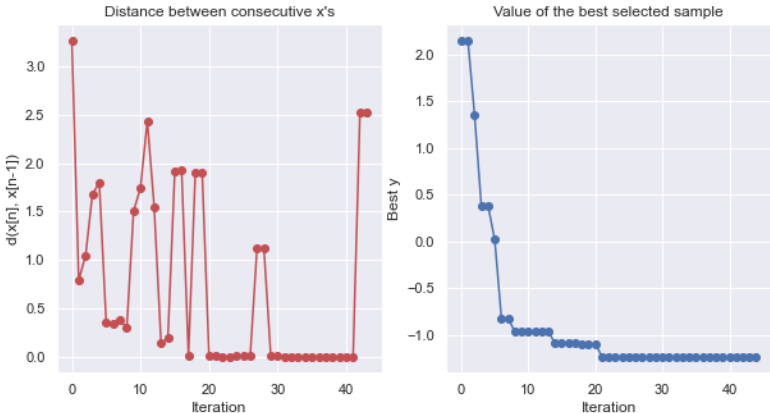
Global Bayesian optimization with GPs

2D example



Global Bayesian optimization with GPs

2D example

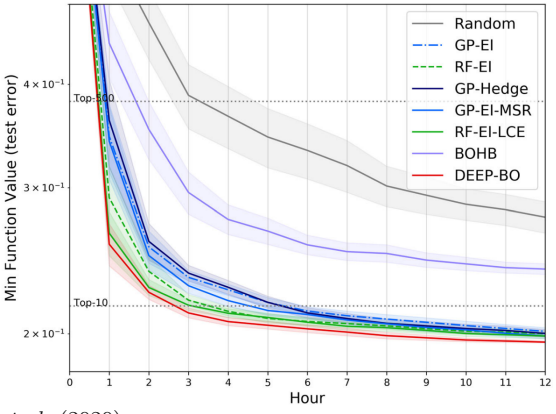


Global Bayesian optimization with GPs

Hyperparameter optimization (HPO) example

HPO of a CNN for the CIFAR-10 dataset.

Hyperparameter	Range
number of first convolution kernels	[8, 32]
number of second convolution kernels	[32, 64]
number of third convolution kernels	[64, 128]
number of forth convolution kernels	[64, 128]
number of neurons in a fully connected layer	[10, 1000]
square length of convolution kernel	[2, 3]
learning rate	[0.0001, 0.4] (log scale)
L2 regularization factor	[0.0, 1.0]
activation function type	ReLU, tanh, eLU
regularization method	None, Dropout, BatchNorm



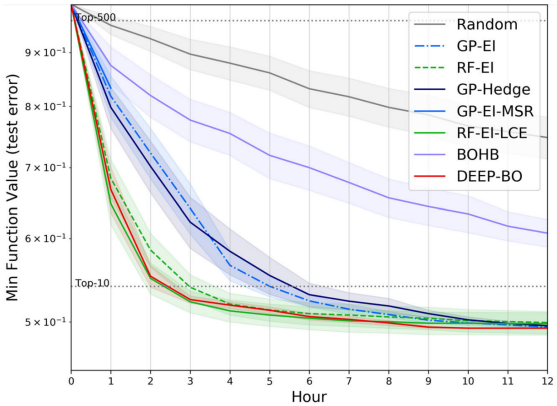
Taken from Chol *et al.* (2020).

Global Bayesian optimization with GPs

Hyperparameter optimization (HPO) example

HPO of a CNN for the CIFAR-100 dataset.

Hyperparameter	Range
number of first convolution kernels	[8, 32]
number of second convolution kernels	[32, 64]
number of third convolution kernels	[64, 128]
number of forth convolution kernels	[64, 128]
number of neurons in a fully connected layer	[10, 1000]
square length of convolution kernel	[2, 3]
learning rate	[0.0001, 0.4] (log scale)
L2 regularization factor	[0.0, 1.0]
activation function type	ReLU, tanh,sigmoid, eLU
regularization method	None, Dropout, BatchNorm



Taken from Chol *et al.* (2020).

Global Bayesian optimization with GPs

Practical aspects

- Use a kernel function less smooth than the RBF, such as the Matérn $\frac{5}{2}$ or $\frac{3}{2}$.
- Use efficient methods to optimize the acquisition function (search trees, hill-climbing, Monte Carlo, etc).
- Initialize the dataset $\mathcal{D}_0$ by using samples from the Sobol sequence or latin hypercube.
- If a priori knowledge about the task is available, choose informative initial points.
- Use a well established package: GPyOpt, GPflowOpt, Spearmint, BoTorch, Dragonfly, pybo...

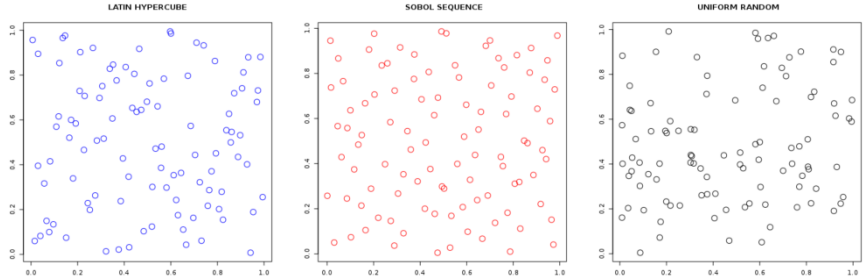


Table of Contents

Current trends on kernel design

- The properties of a zero mean GP model come from its **kernel function**:

$$
\begin{aligned}
y_i &= f_i + \epsilon_i, \quad f_i = f(\boldsymbol{x}_i), \quad i \in \{1, \ldots, N\}, \\
\boldsymbol{f} &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_f), \\
[\boldsymbol{K}_f]_{ij} &= k(\boldsymbol{x}_i, \boldsymbol{x}_j).
\end{aligned}
$$

- Functions that generate a **positive semidefinite** matrix $\boldsymbol{K}_f$ are valid kernels.
- We have seen some of the most used kernels: Squared Exponential, Periodic, Matérn, Polynomial, etc.
- How to find suitable kernel functions for a given problem?

Current trends on kernel design

- The properties of a zero mean GP model come from its **kernel function**:

$$
\begin{aligned}
y_i &= f_i + \epsilon_i, \quad f_i = f(\boldsymbol{x}_i), \quad i \in \{1, \ldots, N\}, \\
\boldsymbol{f} &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_f), \\
[\boldsymbol{K}_f]_{ij} &= k(\boldsymbol{x}_i, \boldsymbol{x}_j).
\end{aligned}
$$

- Functions that generate a **positive semidefinite** matrix $\boldsymbol{K}_f$ are valid kernels.
- We have seen some of the most used kernels: Squared Exponential, Periodic, Matérn, Polynomial, etc.
- How to find suitable kernel functions for a given problem?

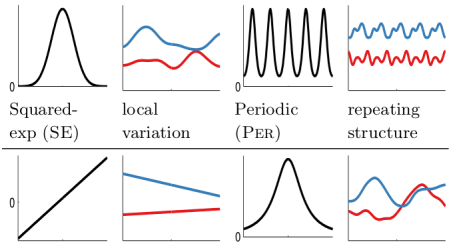
Current trends on kernel design

- The properties of a zero mean GP model come from its **kernel function**:

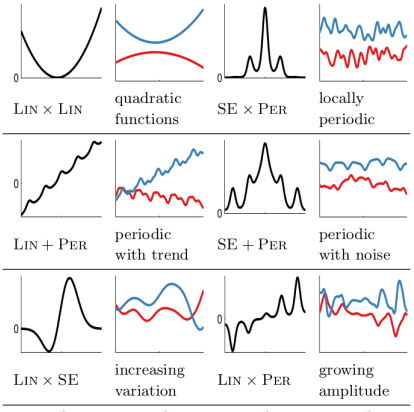
$$
\begin{aligned}
y_i &= f_i + \epsilon_i, \quad f_i = f(\boldsymbol{x}_i), \quad i \in \{1, \ldots, N\}, \\
\boldsymbol{f} &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_f), \\
[\boldsymbol{K}_f]_{ij} &= k(\boldsymbol{x}_i, \boldsymbol{x}_j).
\end{aligned}
$$

- Functions that generate a **positive semidefinite** matrix $\boldsymbol{K}_f$ are valid kernels.
- We have seen some of the most used kernels: Squared Exponential, Periodic, Matérn, Polynomial, etc.
- How to find suitable kernel functions for a given problem?

Composition of base kernels



(Duvenaud, *et al.*, 2013)



(Duvenaud, *et al.*, 2013)

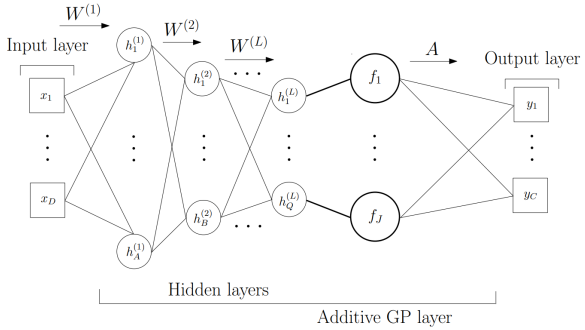
- Duvenaud, *et al.* **"Structure discovery in nonparametric regression through compositional kernel search"**, ICML, 2013.
- Malkomes, *et al.* **"Bayesian optimization for automated model selection"**. ICML AutoML Workshop, 2016.
- Kim and Teh. **"Scaling up the Automatic Statistician: Scalable structure discovery using Gaussian processes"**. AISTATS, 2018.
- Teng, *et al.* **"Scalable variational Bayesian kernel selection for sparse Gaussian process regression"**. AAAI, 2020.

Deep Kernel Learning

- **Mapping the input space** by an arbitrary function $g(\cdot)$ results in a valid kernel:

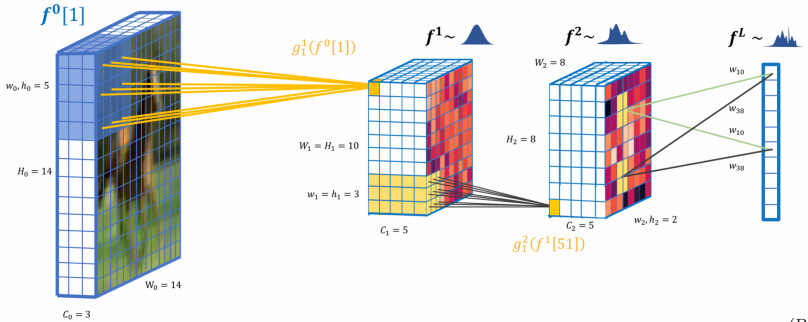
$$\tilde{k}(\boldsymbol{x}_i, \boldsymbol{x}_j) = k(g(\boldsymbol{x}_i), g(\boldsymbol{x}_j)).$$

- Calandra *et al.* (2016) model $g(\cdot)$ with a **neural network** parameterized by $\boldsymbol{\theta}$ that is **jointly optimized** following the gradients of the marginal likelihood $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{L}}{\partial \tilde{\boldsymbol{K}}} \frac{\partial \tilde{\boldsymbol{K}}}{\partial g_{\boldsymbol{\theta}}} \frac{\partial g_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$.
- Wilson *et al.* (2016) scale the approach to large datasets.



(Wilson, *et al.*, 2016)

Convolutional Gaussian Processes



(Blomqvist, *et al.*, 2018)

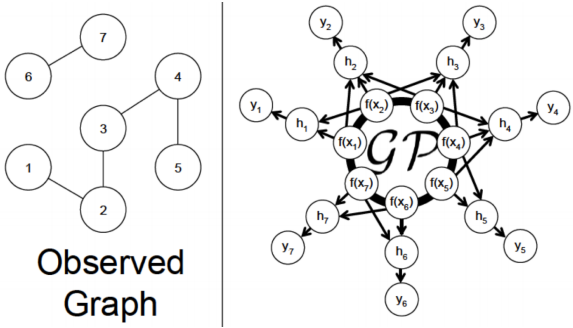
- Sum of **patch responses** $g(\boldsymbol{x}^{[p]})$ of a given image $\boldsymbol{x}$:

$$f(\boldsymbol{x}) = \sum_{p=1}^{P} g(\boldsymbol{x}^{[p]}), \quad \boldsymbol{g} \sim \mathcal{N}(\mathbf{0}, k_g(\boldsymbol{x}, \boldsymbol{x}')),$$

$$\boldsymbol{f} \sim \mathcal{N}\left(\mathbf{0}, \sum_{p=1}^{P}\sum_{p'=1}^{P} k_g(\boldsymbol{x}^{[p]}, \boldsymbol{x}^{[p']})\right).$$

- Van der Wilk, *et al.* **"Convolutional Gaussian processes"**. NeurIPS, 2017.
- Blomqvist, *et al.* **"Deep convolutional Gaussian processes"**. ECMLPKDD, 2018.
- Dutordoir, *et al.* **"Bayesian image classification with deep convolutional Gaussian processes"**. AISTATS, 2020.

Gaussian Processes over graphs



(Ng, *et al.*, 2018)

- Let $\boldsymbol{A} \in \{0,1\}^{N \times N}$ be an **adjacency matrix**, we have:

$$p(\boldsymbol{y}, \boldsymbol{h} | \boldsymbol{X}, \boldsymbol{A}) = p(\boldsymbol{h} | \boldsymbol{X}, \boldsymbol{A}) \prod_{i=1}^{N} p(y_i, h_i),$$

$$p(\boldsymbol{h} | \boldsymbol{X}, \boldsymbol{A}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{P}\boldsymbol{K}_f\boldsymbol{P}^\top), \quad p(\boldsymbol{f} | \boldsymbol{X}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_f),$$

$$\boldsymbol{P} = (\boldsymbol{I} + \boldsymbol{D})^{-1}(\boldsymbol{I} + \boldsymbol{A}),$$

where $\boldsymbol{D}$ is the **degree matrix**.

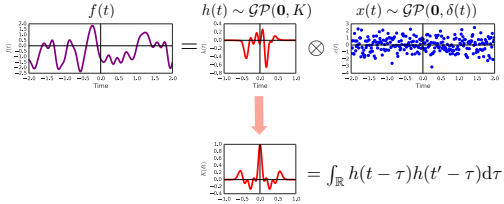
- Ng *et al.* **"Bayesian Semi-supervised Learning with Graph Gaussian Processes"**. NeurIPS, 2018.
- Walker and Glocker. **"Graph convolutional Gaussian processes"**. ICML, 2019.
- Borovitskiy, *et al.* **"Matérn Gaussian processes on graphs"**. AISTATS, 2021.

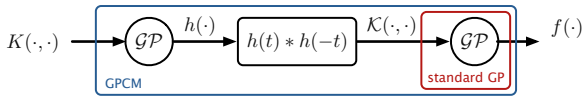
Nonparametric kernels

- **Gaussian Process Convolution Model (GPCM)**
  - **i)** draw from the filter function $h \sim \mathcal{GP}(\mathbf{0}, K)$;
  - **ii)** convolve it with a white noise process $x(t)$:

$$f(t) = \int_{\mathbb{R}} h(t-\tau)x(\tau)\mathrm{d}\tau.$$

  - → Conditionally, $f|h \sim \mathcal{GP}(\mathbf{0}, \mathcal{K})$ is a GP with a nonparametric covariance function.
  - → Closely related to linear time-invariant (LTI) systems, with $h(t)$ acting as an inpulse response.





Tobar, *et al.* **"Learning stationary time series using Gaussian processes with nonparametric kernels"**, Neurips, 2015.

Current trends on kernel design

Summary

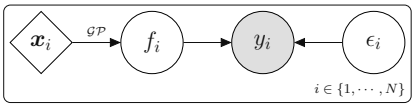
- The choice of the **kernel function** defines the properties of a GP model.
- New kernels can be designed by **manually or automatically combining** base kernels.
- **Deep kernel learning** enables flexible kernels jointly optimized with the other GP hyperparameters.
- Specific data structures can be handled with **convolutional or graph kernels**.
- It is possible to use convolutions of processes to obtain **nonparametric kernels** and work with GPs in the spectral domain.

Table of Contents

From GPLVM to Deep GPs

- So far we have considered supervised learning tasks of the form:

$$y_i = f_i + \epsilon_i, \quad f_i = f(\boldsymbol{x}_i),$$
$$p(\boldsymbol{f}|\boldsymbol{X}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f).$$



→ $y_i|_{i=1}^N$ are **noisy observations**;
→ $f_i|_{i=1}^N$ are **latent** random variables;
→ $\boldsymbol{x}_i|_{i=1}^N$ are **deterministic observed** inputs.

- For multidimensional observations $\boldsymbol{Y} \in \mathbb{R}^{N \times D_y}$ and random inputs $\boldsymbol{X}$:

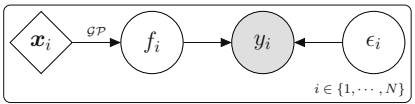
Unobserved inputs $\boldsymbol{X}$
nonlinear unsupervised projection of $\boldsymbol{Y}$

Observed inputs $\boldsymbol{X}$
supervised learning with noisy inputs

From GPLVM to Deep GPs

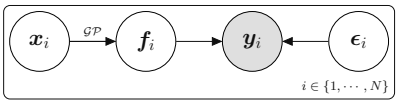
- So far we have considered supervised learning tasks of the form:

$$y_i = f_i + \epsilon_i, \quad f_i = f(\boldsymbol{x}_i),$$
$$p(\boldsymbol{f}|\boldsymbol{X}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K}_f).$$

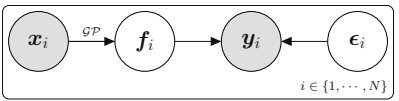


→ $y_i|_{i=1}^N$ are **noisy observations**;
→ $f_i|_{i=1}^N$ are **latent** random variables;
→ $\boldsymbol{x}_i|_{i=1}^N$ are **deterministic observed** inputs.

- For multidimensional observations $\boldsymbol{Y} \in \mathbb{R}^{N \times D_y}$ and random inputs $\boldsymbol{X}$:



Unobserved inputs $\boldsymbol{X}$
nonlinear unsupervised projection of $\boldsymbol{Y}$



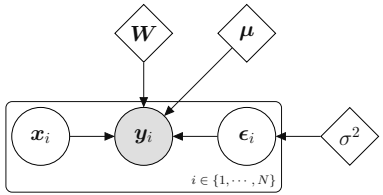
Observed inputs $\boldsymbol{X}$
supervised learning with noisy inputs

PPCA - Probabilistic PCA

- The PPCA is one of the simplest models with **continuous latent variables**:

$$\underbrace{p(\boldsymbol{X})}_{\text{prior}} = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{0}, \boldsymbol{I}), \quad p(\boldsymbol{\epsilon}_i) = \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}),$$

$$\underbrace{p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{\mu}, \sigma^2)}_{\text{likelihood}} = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{W}\boldsymbol{x}_i + \boldsymbol{\mu}, \sigma^2\boldsymbol{I}).$$



- Since the PPCA is **linear**, we have analytical marginal likelihood and posterior:

$$\underbrace{p(\boldsymbol{Y}|\boldsymbol{W}, \boldsymbol{\mu}, \sigma^2)}_{\text{marg. likelihood}} = \int \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{W}\boldsymbol{x}_i + \boldsymbol{\mu}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{x}_i|\boldsymbol{0}, \boldsymbol{I})\mathrm{d}\boldsymbol{x}_i = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{\mu}, \boldsymbol{W}\boldsymbol{W}^\top + \sigma^2\boldsymbol{I}),$$

$$\underbrace{p(\boldsymbol{X}|\boldsymbol{Y}, \hat{\boldsymbol{W}}, \hat{\boldsymbol{\mu}}, \hat{\sigma}^2)}_{\text{posterior}} = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i|\hat{\boldsymbol{M}}^{-1}\hat{\boldsymbol{W}}^\top(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}), \sigma^2\hat{\boldsymbol{M}}^{-1}), \quad \text{where } \hat{\boldsymbol{M}} = (\hat{\boldsymbol{W}}^\top\hat{\boldsymbol{W}} + \hat{\sigma}^2\boldsymbol{I})^{-1}.$$

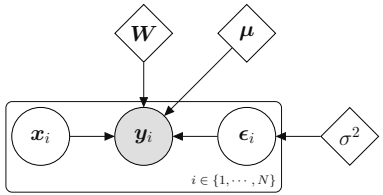
- ML estimates $\hat{\boldsymbol{\mu}}$, $\hat{\boldsymbol{W}}$ and $\hat{\sigma}^2$ can be obtained in closed form or via EM algorithm.

PPCA - Probabilistic PCA

- The PPCA is one of the simplest models with **continuous latent variables**:

$$\underbrace{p(\boldsymbol{X})}_{\text{prior}} = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{0}, \boldsymbol{I}), \quad p(\boldsymbol{\epsilon}_i) = \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}),$$

$$\underbrace{p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{\mu}, \sigma^2)}_{\text{likelihood}} = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{W}\boldsymbol{x}_i + \boldsymbol{\mu}, \sigma^2\boldsymbol{I}).$$



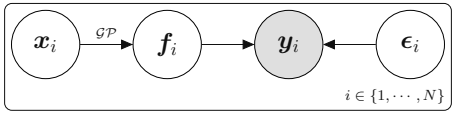
- Since the PPCA is **linear**, we have analytical marginal likelihood and posterior:

$$\underbrace{p(\boldsymbol{Y}|\boldsymbol{W}, \boldsymbol{\mu}, \sigma^2)}_{\text{marg. likelihood}} = \int \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{W}\boldsymbol{x}_i + \boldsymbol{\mu}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{x}_i|\boldsymbol{0}, \boldsymbol{I})\mathrm{d}\boldsymbol{x}_i = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i|\boldsymbol{\mu}, \boldsymbol{W}\boldsymbol{W}^\top + \sigma^2\boldsymbol{I}),$$

$$\underbrace{p(\boldsymbol{X}|\boldsymbol{Y}, \hat{\boldsymbol{W}}, \hat{\boldsymbol{\mu}}, \hat{\sigma}^2)}_{\text{posterior}} = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i|\hat{\boldsymbol{M}}^{-1}\hat{\boldsymbol{W}}^\top(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}), \sigma^2\hat{\boldsymbol{M}}^{-1}), \quad \text{where } \hat{\boldsymbol{M}} = (\hat{\boldsymbol{W}}^\top\hat{\boldsymbol{W}} + \hat{\sigma}^2\boldsymbol{I})^{-1}.$$

- ML estimates $\hat{\boldsymbol{\mu}}$, $\hat{\boldsymbol{W}}$ and $\hat{\sigma}^2$ can be obtained in closed form or via EM algorithm.

GPLVM - Gaussian Process Latent Variable Model



- Inspired by the PPCA formulation, Lawrence (2005) proposed the GPLVM:

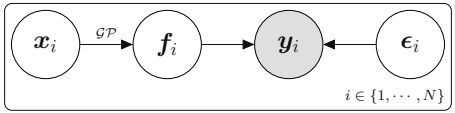
$$p(\boldsymbol{X}) = \prod_{i=1}^{N} \overbrace{\mathcal{N}(\boldsymbol{x}_i|\boldsymbol{0}, \boldsymbol{I})}^{\text{input prior}}, \quad p(\boldsymbol{F}|\boldsymbol{X}) = \prod_{d=1}^{D_y} \overbrace{\mathcal{N}(\boldsymbol{f}_{:d}|\boldsymbol{0}, \boldsymbol{K}_f)}^{\text{GP prior}},$$

$$p(\boldsymbol{Y}|\boldsymbol{F}, \boldsymbol{X}) = \prod_{d=1}^{D_y} \underbrace{\mathcal{N}(\boldsymbol{y}_{:d}|\boldsymbol{f}_{:d}, \sigma^2\boldsymbol{I})}_{\text{likelihood}} \mathcal{N}(\boldsymbol{f}_{:d}|\boldsymbol{0}, \boldsymbol{K}_f),$$

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{d=1}^{D_y} \underbrace{\mathcal{N}(\boldsymbol{y}_{:d}|\boldsymbol{0}, \boldsymbol{K}_f + \sigma^2\boldsymbol{I})}_{\text{marginal likelihood}}.$$

- Marginalization of $\boldsymbol{F}$ is analytic, but of the random input $\boldsymbol{X}$ is not.
  $\rightarrow$ $\boldsymbol{X}$ appears in a nonlinear way inside the covariance matrix $\boldsymbol{K}_f$.

GPLVM - Gaussian Process Latent Variable Model



- Inspired by the PPCA formulation, Lawrence (2005) proposed the GPLVM:

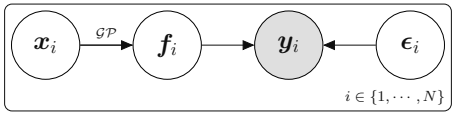
$$p(\boldsymbol{X}) = \prod_{i=1}^{N} \overbrace{\mathcal{N}(\boldsymbol{x}_i|\boldsymbol{0}, \boldsymbol{I})}^{\text{input prior}}, \quad p(\boldsymbol{F}|\boldsymbol{X}) = \prod_{d=1}^{D_y} \overbrace{\mathcal{N}(\boldsymbol{f}_{:d}|\boldsymbol{0}, \boldsymbol{K}_f)}^{\text{GP prior}},$$

$$p(\boldsymbol{Y}|\boldsymbol{F}, \boldsymbol{X}) = \prod_{d=1}^{D_y} \underbrace{\mathcal{N}(\boldsymbol{y}_{:d}|\boldsymbol{f}_{:d}, \sigma^2\boldsymbol{I})}_{\text{likelihood}} \mathcal{N}(\boldsymbol{f}_{:d}|\boldsymbol{0}, \boldsymbol{K}_f),$$

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{d=1}^{D_y} \underbrace{\mathcal{N}(\boldsymbol{y}_{:d}|\boldsymbol{0}, \boldsymbol{K}_f + \sigma^2\boldsymbol{I})}_{\text{marginal likelihood}}.$$

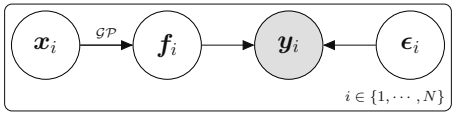
- Marginalization of $\boldsymbol{F}$ is analytic, but of the random input $\boldsymbol{X}$ is not.
  $\rightarrow$ $\boldsymbol{X}$ appears in a nonlinear way inside the covariance matrix $\boldsymbol{K}_f$.

GPLVM - Gaussian Process Latent Variable Model



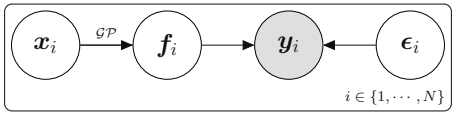
- Lawrence (2005) avoids the intractability by optimizing $\boldsymbol{X}$ in a MAP fashion.
- Titsias and Lawrence (2010) follow a variational approximation to marginalize $\boldsymbol{X}$.
  - → The posterior of $\boldsymbol{X}$ is approximated by $q(\boldsymbol{X}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{m}_i, \boldsymbol{S}_i)$.
  - → Enables tuning of the latent space dimension via ARD.
  - → More robust to overfitting.
- Contrary to standard GPs, the GPLVM is a **generative model**.
- Generation of new observations is easy, since the GP prior maps from $\boldsymbol{x}_*$ to $\boldsymbol{y}_*$.
- Latent projection given a new observation $\boldsymbol{y}_*$ is nontrivial.
  - → E.g. neural networks can be used in an autoencoder fashion (Dai *et al.*, 2016).

GPLVM - Gaussian Process Latent Variable Model



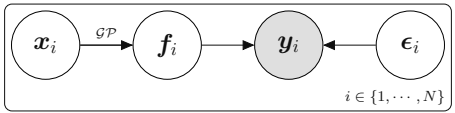
- Lawrence (2005) avoids the intractability by optimizing $\boldsymbol{X}$ in a MAP fashion.
- Titsias and Lawrence (2010) follow a variational approximation to marginalize $\boldsymbol{X}$.
  - → The posterior of $\boldsymbol{X}$ is approximated by $q(\boldsymbol{X}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{m}_i, \boldsymbol{S}_i)$.
  - → Enables tuning of the latent space dimension via ARD.
  - → More robust to overfitting.
- Contrary to standard GPs, the GPLVM is a **generative model**.
- Generation of new observations is easy, since the GP prior maps from $\boldsymbol{x}_*$ to $\boldsymbol{y}_*$.
- Latent projection given a new observation $\boldsymbol{y}_*$ is nontrivial.
  - → E.g. neural networks can be used in an autoencoder fashion (Dai *et al.*, 2016).

GPLVM - Gaussian Process Latent Variable Model



- Lawrence (2005) avoids the intractability by optimizing $\boldsymbol{X}$ in a MAP fashion.
- Titsias and Lawrence (2010) follow a variational approximation to marginalize $\boldsymbol{X}$.
  - → The posterior of $\boldsymbol{X}$ is approximated by $q(\boldsymbol{X}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{m}_i, \boldsymbol{S}_i)$.
  - → Enables tuning of the latent space dimension via ARD.
  - → More robust to overfitting.
- Contrary to standard GPs, the GPLVM is a **generative model**.
- Generation of new observations is easy, since the GP prior maps from $\boldsymbol{x}_*$ to $\boldsymbol{y}_*$.
- Latent projection given a new observation $\boldsymbol{y}_*$ is nontrivial.
  - → E.g. neural networks can be used in an autoencoder fashion (Dai *et al.*, 2016).

GPLVM - Gaussian Process Latent Variable Model



- Lawrence (2005) avoids the intractability by optimizing $\boldsymbol{X}$ in a MAP fashion.
- Titsias and Lawrence (2010) follow a variational approximation to marginalize $\boldsymbol{X}$.
  - → The posterior of $\boldsymbol{X}$ is approximated by $q(\boldsymbol{X}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{m}_i, \boldsymbol{S}_i)$.
  - → Enables tuning of the latent space dimension via ARD.
  - → More robust to overfitting.
- Contrary to standard GPs, the GPLVM is a **generative model**.
- Generation of new observations is easy, since the GP prior maps from $\boldsymbol{x}_*$ to $\boldsymbol{y}_*$.
- Latent projection given a new observation $\boldsymbol{y}_*$ is nontrivial.
  - → E.g. neural networks can be used in an autoencoder fashion (Dai *et al.*, 2016).

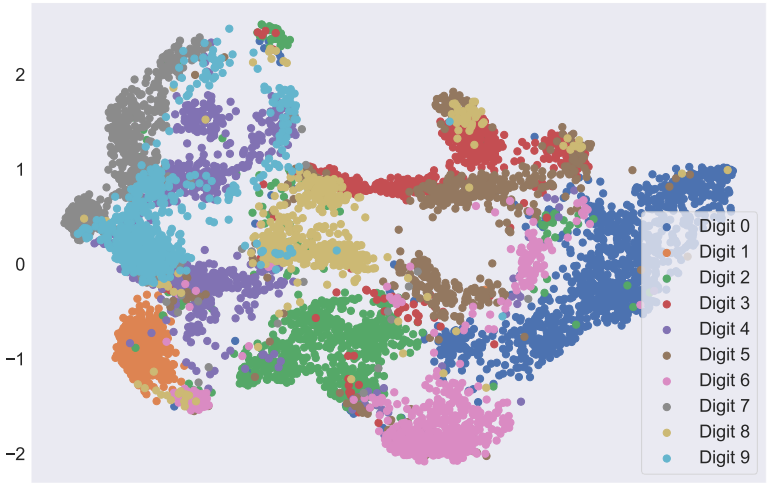
USPS digits - 7291 training samples

256 → 2 dimension projection with PCA

$$p(\boldsymbol{x}_i|\boldsymbol{y}_i, \hat{\boldsymbol{W}}, \hat{\boldsymbol{\mu}}, \hat{\sigma}^2)$$

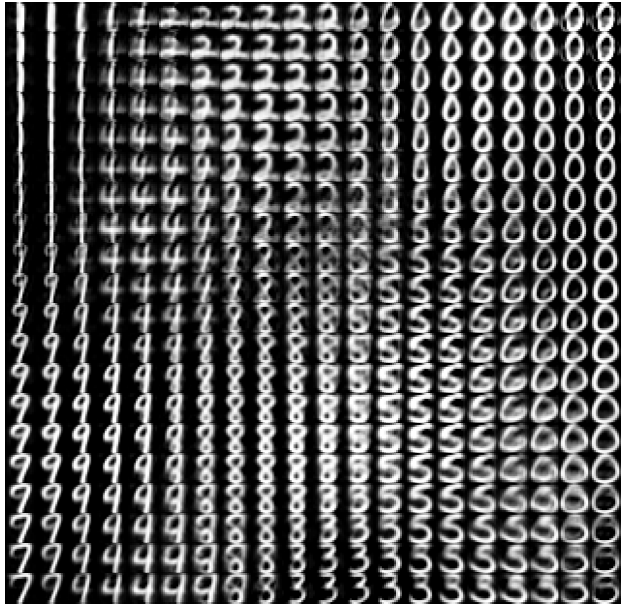
USPS digits - 7291 training samples

256 → 2 dimension projection with GPLVM (50 inducing points)

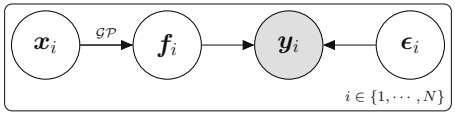


$$p(\boldsymbol{x}_i|\boldsymbol{y}_i) \approx q(\boldsymbol{x}_i|\boldsymbol{y}_i) = \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{m}_i, \boldsymbol{S}_i).$$

USPS digits - GPLVM samples

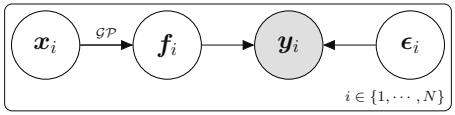


GPLVM - Gaussian Process Latent Variable Model



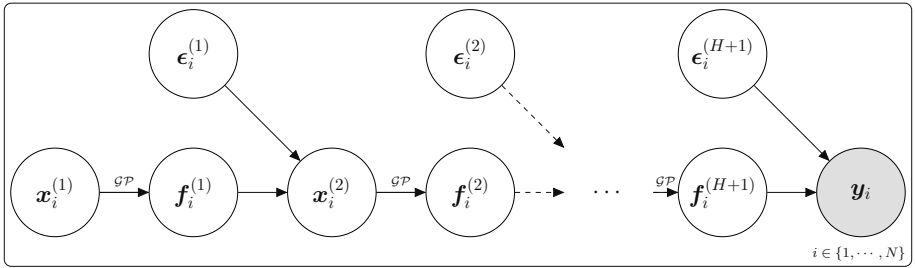
- Instead of a simple prior $p(\boldsymbol{X}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{0}, \boldsymbol{I})$, we could have different choices:
  - → **Discriminative priors** for classification (Urtasun and Darrell, 2007);
  - → **Dynamical priors** with temporal dependence (Wang *et al.*, 2007; Damianou *et al.*, 2011; Frigola *et al.*, 2014; Mattos *et al.*, 2016);
  - → Mixture of **visible and missing** data (Damianou and Lawrence, 2015).
  - → **Conditional** variables (Dutordoir *et al.*, 2016);
  - → **Other(s) GP prior(s)** (Lawrence and Moore, 2007; Damianou and Lawrence, 2013; Mattos *et al.*, 2016; Bui *et al.*, 2016; Salimbeni and Deisenroth, 2017).

GPLVM - Gaussian Process Latent Variable Model



- Instead of a simple prior $p(\boldsymbol{X}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{0}, \boldsymbol{I})$, we could have different choices:
  - → **Discriminative priors** for classification (Urtasun and Darrell, 2007);
  - → **Dynamical priors** with temporal dependence (Wang *et al.*, 2007; Damianou *et al.*, 2011; Frigola *et al.*, 2014; Mattos *et al.*, 2016);
  - → Mixture of **visible and missing** data (Damianou and Lawrence, 2015).
  - → **Conditional** variables (Dutordoir *et al.*, 2016);
  - → **Other(s) GP prior(s)** (Lawrence and Moore, 2007; Damianou and Lawrence, 2013; Mattos *et al.*, 2016; Bui *et al.*, 2016; Salimbeni and Deisenroth, 2017).

Deep Gaussian Processes

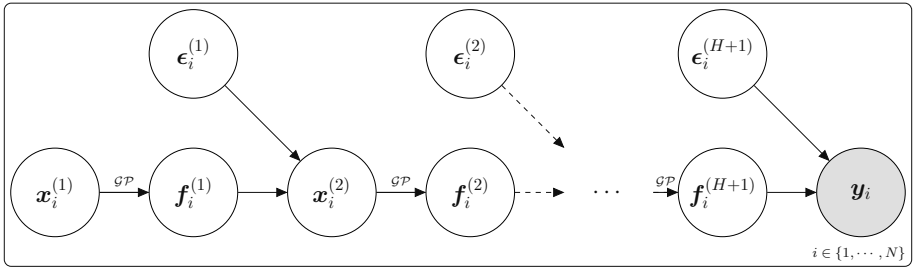


- A **hierarchy** of H GP priors gives rise to a **deep** GP model with H hidden layers:

$$\boldsymbol{y}_i = f^{(H+1)}\left(\boldsymbol{x}_i^{(H+1)}\right) + \boldsymbol{\epsilon}_i^{(H+1)}, \qquad \boldsymbol{f}_{:d}^{(H+1)} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{K}_f^{(H+1)}\right)$$
$$\boldsymbol{x}_i^{(h+1)} = f^{(h)}\left(\boldsymbol{x}_i^{(h)}\right) + \boldsymbol{\epsilon}_i^{(h)}, \qquad \boldsymbol{f}_{:d}^{(h)} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{K}_f^{(h)}\right), \qquad 1 \leq h \leq H.$$

- The intractabilities require approximate inference and inducing point approaches.

Deep Gaussian Processes

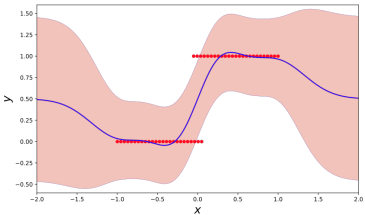


- A **hierarchy** of H GP priors gives rise to a **deep** GP model with H hidden layers:

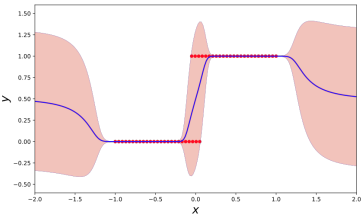
$$\boldsymbol{y}_i = f^{(H+1)}\left(\boldsymbol{x}_i^{(H+1)}\right) + \boldsymbol{\epsilon}_i^{(H+1)}, \qquad \boldsymbol{f}_{:d}^{(H+1)} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{K}_f^{(H+1)}\right)$$
$$\boldsymbol{x}_i^{(h+1)} = f^{(h)}\left(\boldsymbol{x}_i^{(h)}\right) + \boldsymbol{\epsilon}_i^{(h)}, \qquad \boldsymbol{f}_{:d}^{(h)} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{K}_f^{(h)}\right), \qquad 1 \leq h \leq H.$$

- The intractabilities require approximate inference and inducing point approaches.

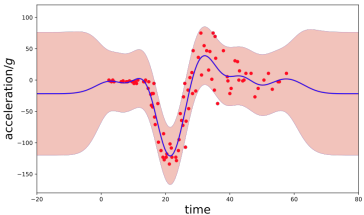
Deep Gaussian Processes



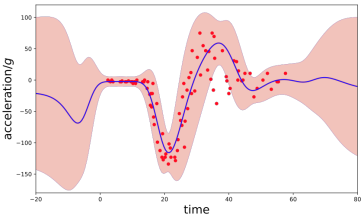
(a) Shallow GP.



(b) Deep GP.



(c) Shallow GP.



(d) Deep GP.

(Lawrence, 2020)

Deep Gaussian Processes

- But what are the advantages of deep GPs?
  - → The resulting model **cannot be explained** by a single GP.
  - → Multiple layers **alleviate the choice** of the kernel function.
  - → Enables **hierarchical** feature learning.
  - → Deep GPs are **related to deep neural networks** with infinite hidden units.
- Supported by modern frameworks, such as **GPyTorch** and **GPflux**.
- Research on deep GPs has a wide range of topics:
  - → Variational Auto-encoded Deep Gaussian Processes (Dai *et al.*, 2016);
  - → Recurrent Gaussian Processes (Mattos *et al.*, 2016);
  - → Doubly Stochastic Variational Inference for Deep GPs (Salimbeni and Deisenroth, 2017);
  - → Deep Convolutional GPs (Blomqvist *et al.*, 2018);
  - → Deep GPs with Importance-Weighted Variational Inference (Salimbeni *et al.*, 2019);
  - → Stochastic Deep Gaussian Processes over Graphs (Li *et al.*, 2020);
  - → Global Inducing Point Variational Posteriors for BNNs and DGPs (Ober and Aitchison, 2021).

Deep Gaussian Processes

- But what are the advantages of deep GPs?
  - → The resulting model **cannot be explained** by a single GP.
  - → Multiple layers **alleviate the choice** of the kernel function.
  - → Enables **hierarchical** feature learning.
  - → Deep GPs are **related to deep neural networks** with infinite hidden units.
- Supported by modern frameworks, such as **GPyTorch** and **GPflux**.
- Research on deep GPs has a wide range of topics:
  - → Variational Auto-encoded Deep Gaussian Processes (Dai *et al.*, 2016);
  - → Recurrent Gaussian Processes (Mattos *et al.*, 2016);
  - → Doubly Stochastic Variational Inference for Deep GPs (Salimbeni and Deisenroth, 2017);
  - → Deep Convolutional GPs (Blomqvist *et al.*, 2018);
  - → Deep GPs with Importance-Weighted Variational Inference (Salimbeni *et al.*, 2019);
  - → Stochastic Deep Gaussian Processes over Graphs (Li *et al.*, 2020);
  - → Global Inducing Point Variational Posteriors for BNNs and DGPs (Ober and Aitchison, 2021).

Deep Gaussian Processes

- But what are the advantages of deep GPs?
  - → The resulting model **cannot be explained** by a single GP.
  - → Multiple layers **alleviate the choice** of the kernel function.
  - → Enables **hierarchical** feature learning.
  - → Deep GPs are **related to deep neural networks** with infinite hidden units.
- Supported by modern frameworks, such as **GPyTorch** and **GPflux**.
- Research on deep GPs has a wide range of topics:
  - → Variational Auto-encoded Deep Gaussian Processes (Dai *et al.*, 2016);
  - → Recurrent Gaussian Processes (Mattos *et al.*, 2016);
  - → Doubly Stochastic Variational Inference for Deep GPs (Salimbeni and Deisenroth, 2017);
  - → Deep Convolutional GPs (Blomqvist *et al.*, 2018);
  - → Deep GPs with Importance-Weighted Variational Inference (Salimbeni *et al.*, 2019);
  - → Stochastic Deep Gaussian Processes over Graphs (Li *et al.*, 2020);
  - → Global Inducing Point Variational Posteriors for BNNs and DGPs (Ober and Aitchison, 2021).

From GPLVM to Deep GPs

Summary

- GPLVM enables **unsupervised learning** with GP priors in a **generative setting**.
- GP learning with **missing input data** uses the same GPLVM inference tools.
- A **hierarchy** of GPLVM blocks results in a **deep GP model**.
- The multilayer **composition of processes** cannot be represented by a single GP.
- Bayesian inference with GPLVMs and deep GPs is hard, but some **open source modern frameworks** are available.

Table of Contents

Concluding Remarks

- GPs are **Bayesian nonparametric** models, meaning that they:
  - **i)** naturally combine existing biases and the knowledge extracted from data;
  - **ii)** provide **probabilistic estimates**;
  - **iii)** the more data they see, the better they become.
- GPs can be applied to a plethora of settings: denoising, regression, interpolation, extrapolation, classification, optimization, unsupervised learning, hierarchical modelling, etc.
- **Hyperparameter optimization** can be performed via marginal likelihood, and jointly with sparse approximations, if they are required.
- The GP community is very active, come on over and join us!

Additional GP topics

Gaussian Process Summer Schools - GPSS

Superb presentations and workshops on several GP related subjects: `http://gpss.cc/`.

- More thorough discussion on sparse and scalable GPs
  - → Bauer, van der Wilk and Rasmussen. **"Understanding probabilistic sparse Gaussian process approximations"**. NeurIPS, 2016.
  - → Bui, Yan and Turner. **"A Unifying Framework for Gaussian Process Pseudo-Point Approximations using Power Expectation Propagation"**. JMLR, 2017.
  - → Liu, *et al.* **"When Gaussian process meets big data: A review of scalable GPs"**. IEEE TNNLS, 2020.
  - → Leibfried, *et al.* **"A tutorial on sparse Gaussian processes and variational inference"**. arXiv preprint, 2021.
- Bayesian optimization
  - → Frazier, **"Bayesian Optimization"**. INFORMS Tutorials, 2018.
  - → Shahriari, Swersky, Wang, Adams and de Freitas, **"Taking the Human Out of the Loop: A Review of Bayesian Optimization"**. Proc. of the IEEE, 2016.
  - → Snoek, Larochelle and Adams, **"Practical Bayesian Optimization of Machine Learning Algorithms"**. NeurIPS, 2012.

Additional GP topics

Gaussian Process Summer Schools - GPSS

Superb presentations and workshops on several GP related subjects: `http://gpss.cc/`.

- More thorough discussion on sparse and scalable GPs
  - → Bauer, van der Wilk and Rasmussen. **"Understanding probabilistic sparse Gaussian process approximations"**. NeurIPS, 2016.
  - → Bui, Yan and Turner. **"A Unifying Framework for Gaussian Process Pseudo-Point Approximations using Power Expectation Propagation"**. JMLR, 2017.
  - → Liu, *et al.* **"When Gaussian process meets big data: A review of scalable GPs"**. IEEE TNNLS, 2020.
  - → Leibfried, *et al.* **"A tutorial on sparse Gaussian processes and variational inference"**. arXiv preprint, 2021.
- Bayesian optimization
  - → Frazier, **"Bayesian Optimization"**. INFORMS Tutorials, 2018.
  - → Shahriari, Swersky, Wang, Adams and de Freitas, **"Taking the Human Out of the Loop: A Review of Bayesian Optimization"**. Proc. of the IEEE, 2016.
  - → Snoek, Larochelle and Adams, **"Practical Bayesian Optimization of Machine Learning Algorithms"**. NeurIPS, 2012.

Additional GP topics

Gaussian Process Summer Schools - GPSS

Superb presentations and workshops on several GP related subjects: `http://gpss.cc/`.

- More thorough discussion on sparse and scalable GPs
  - → Bauer, van der Wilk and Rasmussen. **"Understanding probabilistic sparse Gaussian process approximations"**. NeurIPS, 2016.
  - → Bui, Yan and Turner. **"A Unifying Framework for Gaussian Process Pseudo-Point Approximations using Power Expectation Propagation"**. JMLR, 2017.
  - → Liu, *et al.* **"When Gaussian process meets big data: A review of scalable GPs"**. IEEE TNNLS, 2020.
  - → Leibfried, *et al.* **"A tutorial on sparse Gaussian processes and variational inference"**. arXiv preprint, 2021.
- Bayesian optimization
  - → Frazier, **"Bayesian Optimization"**. INFORMS Tutorials, 2018.
  - → Shahriari, Swersky, Wang, Adams and de Freitas, **"Taking the Human Out of the Loop: A Review of Bayesian Optimization"**. Proc. of the IEEE, 2016.
  - → Snoek, Larochelle and Adams, **"Practical Bayesian Optimization of Machine Learning Algorithms"**. NeurIPS, 2012.

Additional GP topics

- Deep GPs and beyond
  - → Salimbeni and Deisenroth. **"Doubly stochastic variational inference for deep Gaussian processes"**. NeurIPS, 2017.
  - → Salimbeni, *et al.* **"Deep Gaussian processes with importance-weighted variational inference"**. ICML, 2019.
  - → Bui, *et al.* **"Deep Gaussian processes for regression using approximate expectation propagation"**. ICML, 2016.
  - → Dunlop, *et al.* **"How deep are deep Gaussian processes?"**. JMLR, 2018.
  - → Aitchison, Yang and Ober. **"Deep kernel processes"**. ICML, 2021.
- GPs, deep learning and Neural Tangent Kernel (NTK)
  - → Matthews, *et al.* **"Gaussian process behaviour in wide deep neural networks"**. ICLR, 2018.
  - → Lee, *et al.* **"Deep neural networks as Gaussian processes"**. ICLR, 2018.
  - → Jacot, Gabriel and Hongler. **"Neural tangent kernel: Convergence and generalization in neural networks"**. NeurIPS, 2018.
  - → Dutordoir, *et al.* **"Deep Neural Networks as Point Estimates for Deep Gaussian Processes"**. arXiv preprint, 2021.

Additional GP topics

- Deep GPs and beyond
  - → Salimbeni and Deisenroth. **"Doubly stochastic variational inference for deep Gaussian processes"**. NeurIPS, 2017.
  - → Salimbeni, *et al.* **"Deep Gaussian processes with importance-weighted variational inference"**. ICML, 2019.
  - → Bui, *et al.* **"Deep Gaussian processes for regression using approximate expectation propagation"**. ICML, 2016.
  - → Dunlop, *et al.* **"How deep are deep Gaussian processes?"**. JMLR, 2018.
  - → Aitchison, Yang and Ober. **"Deep kernel processes"**. ICML, 2021.
- GPs, deep learning and Neural Tangent Kernel (NTK)
  - → Matthews, *et al.* **"Gaussian process behaviour in wide deep neural networks"**. ICLR, 2018.
  - → Lee, *et al.* **"Deep neural networks as Gaussian processes"**. ICLR, 2018.
  - → Jacot, Gabriel and Hongler. **"Neural tangent kernel: Convergence and generalization in neural networks"**. NeurIPS, 2018.
  - → Dutordoir, *et al.* **"Deep Neural Networks as Point Estimates for Deep Gaussian Processes"**. arXiv preprint, 2021.

Questions?

César Lincoln C. Mattos
@cesarlincoln
cesarlincoln@dc.ufc.br