Gaussian Processes

Arno Solin

Assistant Professor in Machine Learning
Department of Computer Science
Aalto University

PROBAI SUMMER SCHOOL

June 15, 2022

@arnosolin

arno.solin.fi

It’s all about the tools you have in your toolbox

Structure

Part I

Pragmatic introduction to Gaussian processes

Part II

Challenges that break the beauty

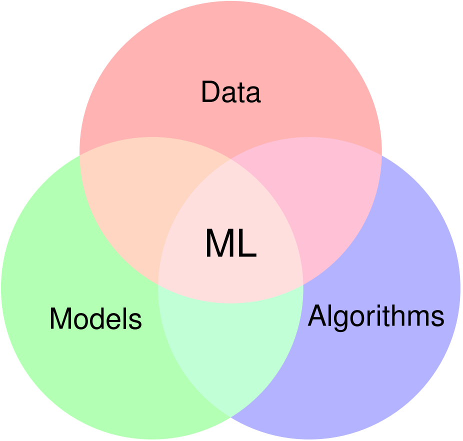
Part III

Connections and approaches to GPs

Part IV

Recap and Q&A

Gaussian processes and ML



Definitions

A random vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$ is said to have the **multivariate Gaussian distribution** if all linear combinations of $\boldsymbol{x}$ are Gaussian distributed:

$$y = a_1 x_1 + a_2 x_2 + \cdots + a_d x_d \sim \mathrm{N}(m, v)$$

for all $\boldsymbol{a} \in \mathbb{R}^d$

A **Gaussian process** (GP) is a collection of random variables over space, such that any finite subset of them have a joint Gaussian distribution.

Characterization and notation

- A Gaussian process can be considered as a distribution over functions $f : \mathcal{X} \to \mathbb{R}$ (the domain or index space $\mathcal{X}$ is typically $\mathbb{R}^d$)

$$f(\boldsymbol{x}) \sim \mathcal{GP}\big(\mu(\boldsymbol{x}), \kappa(\boldsymbol{x}, \boldsymbol{x}')\big)$$

- A Gaussian process is completely characterized by its mean function $\mu(\boldsymbol{x})$ and its covariance function $\kappa(\boldsymbol{x}, \boldsymbol{x}')$, which define

$$\mathbb{E}\left[f(\boldsymbol{x})\right] = \mu(\boldsymbol{x}) \quad \text{and} \quad \mathrm{cov}[f(\boldsymbol{x}), f(\boldsymbol{x}')] = \kappa(\boldsymbol{x}, \boldsymbol{x}')$$

Characterization and notation

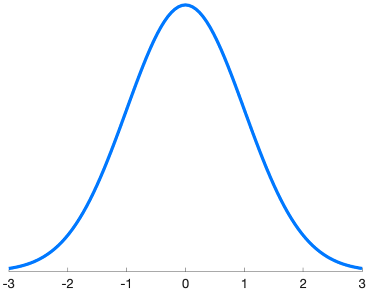
- The probability of any subset of function values $\boldsymbol{f} = f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)$ at any inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ is
$$p(\boldsymbol{f}) = \mathrm{N}(\boldsymbol{f} \mid \boldsymbol{m}, \boldsymbol{K})$$
where $\boldsymbol{m} = \mu(\boldsymbol{x}_1), \ldots, \mu(\boldsymbol{x}_n)$ and $[\boldsymbol{K}]_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$
- If $\mathcal{X} = \mathbb{R}^d$, the GP prior describes infinitely many random variable $\{f(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^d\}$, but in practice we only have to deal with a finite subset corresponding to the data set at hand, and where we want to evaluate ('test') the function
- This also gives rise to the *non-parametric* nature of GPs

Where the magic happens: The covariance function

- In the kernel representation of GPs, the covariance function $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ encodes prior beliefs of data-generating latent functions
- Typical choices are *continuity*, *differentiability* (smoothness), *periodicity*, *invariances*, *etc.*
- The RBF covariance function:

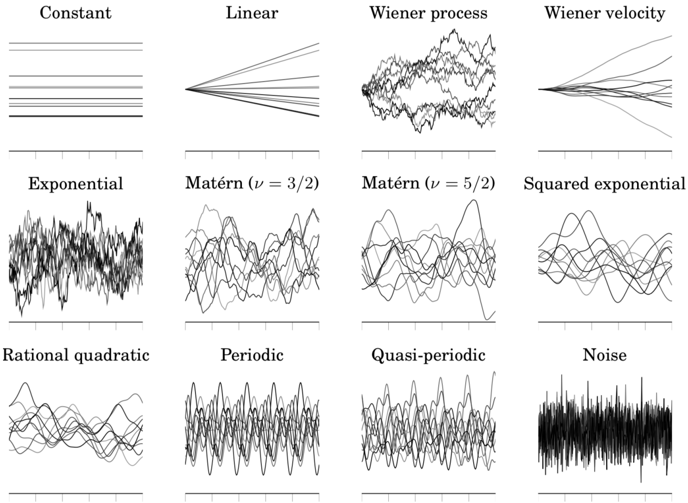
$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\ell^2}\right)$$

- The covariance functions typically have hyperparameters that are learned from data



$\|\boldsymbol{x} - \boldsymbol{x}'\|$

Examples of draws from GP priors



Anatomy of a GP model in ML

In machine learning the kernel (moment) representation is favoured

$$f(\boldsymbol{x}) \sim \mathcal{GP}(\mu(\boldsymbol{x}), \kappa(\boldsymbol{x}, \boldsymbol{x}')) \qquad \textit{GP prior}$$

$$\boldsymbol{y} \mid \boldsymbol{f} \sim \prod_i p(y_i \mid f(\boldsymbol{x}_i)) \qquad \textit{likelihood}$$

Example: GP regression

- GP regression problem with input–output training pairs $\{(x_i, y_i)\}_{i=1}^{n}$:

$$
\begin{aligned}
f(x) &\sim \mathrm{GP}(0, \kappa(x, x')), \\
y_i &= f(x_i) + \varepsilon_i, \quad \varepsilon_i \sim \mathrm{N}(0, \sigma_\mathrm{n}^2)
\end{aligned}
$$

- The posterior mean and variance for an unseen test input x_* is given by:

$$
\begin{aligned}
\mathbb{E}[f_*] &= \boldsymbol{k}_* \, (\boldsymbol{K} + \sigma_\mathrm{n}^2 \, \boldsymbol{I})^{-1} \, \boldsymbol{y}, \\
\mathbb{V}[f_*] &= \kappa(x_*, x_*) - \boldsymbol{k}_* \, (\boldsymbol{K} + \sigma_\mathrm{n}^2 \, \boldsymbol{I})^{-1} \, \boldsymbol{k}_*^\mathsf{T}
\end{aligned}
$$

- Learn hyperparamters θ by maximizing w.r.t. log marginal likelihood:

$$
\log p(\mathbf{y} \mid \theta) = -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log |\boldsymbol{K}_\theta + \sigma_\mathrm{n}^2 \, \boldsymbol{I}| - \frac{1}{2} \boldsymbol{y}^\mathsf{T} \, (\boldsymbol{K}_\theta + \sigma_\mathrm{n}^2 \, \boldsymbol{I})^{-1} \, \boldsymbol{y}
$$

- Note the inversion of the $n \times n$ matrix.

Example: GP regression [details]

- Step 1: Write the joint model

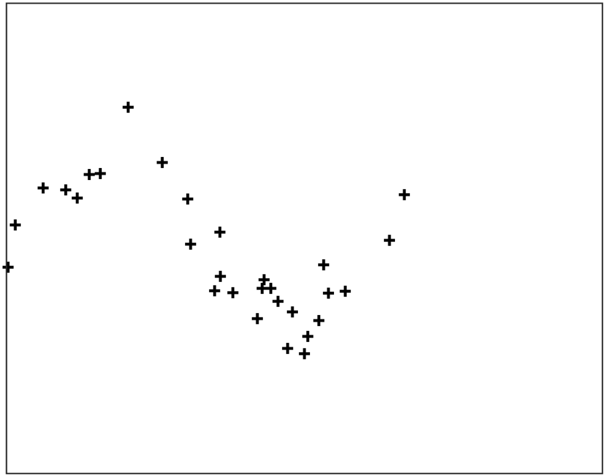
$$p(\boldsymbol{y}, \boldsymbol{f}, f_*) = p(\boldsymbol{y} \mid \boldsymbol{f})\, p(\boldsymbol{f}, f_*) = \mathrm{N}\left(\boldsymbol{y} \middle| \boldsymbol{f}, \sigma_\mathrm{n}^2 \boldsymbol{I}\right) \mathrm{N}\left(\begin{bmatrix} \boldsymbol{f} \\ f_* \end{bmatrix} \middle| \boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_{ff} & \boldsymbol{k}_{f_* f} \\ \boldsymbol{k}_{f_* f} & k_{f_* f_*} \end{bmatrix} \right)$$

- Step 2: Marginalize over $\boldsymbol{f}$

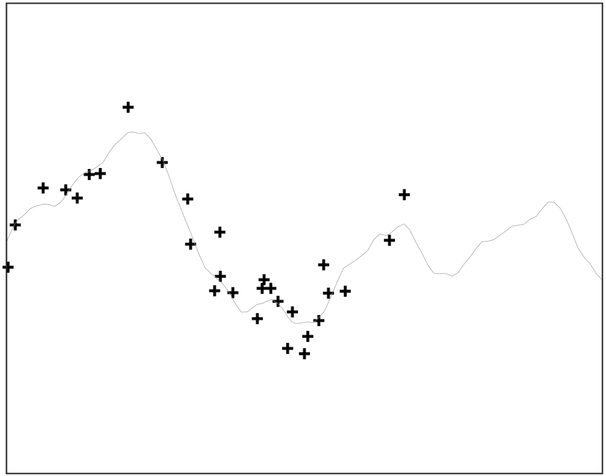
$$p(\boldsymbol{y}, f_*) = \int p(\boldsymbol{y} \mid \boldsymbol{f})\, p(\boldsymbol{f}, f_*)\, \mathrm{d}\boldsymbol{f} = \mathrm{N}\left(\begin{bmatrix} \boldsymbol{y} \\ f_* \end{bmatrix} \mid \boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_{ff} + \sigma_\mathrm{n}^2 \boldsymbol{I} & \boldsymbol{K}_{f_* f} \\ \boldsymbol{K}_{f_* f} & k_{f_* f_*} \end{bmatrix} \right)$$

- Step 3: Compute conditional distribution $p(f_* \mid \boldsymbol{y})$

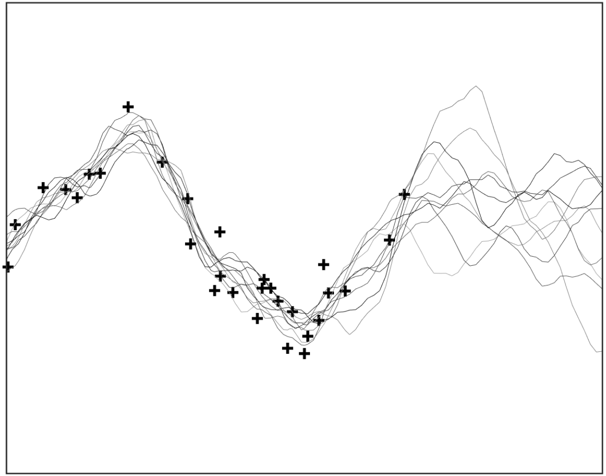
$$\begin{aligned} p(f_* \mid \boldsymbol{y}) &= \mathrm{N}\left(f_* \mid \mathbb{E}[f_*], \mathbb{V}[f_*]\right) \\ \mathbb{E}[f_*] &= \boldsymbol{k}_{f_* f} \left(\boldsymbol{K}_{ff} + \sigma_\mathrm{n}^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y} \\ \mathbb{V}[f_*] &= k_{f_* f_*} - \boldsymbol{k}_{f_* f} \left(\boldsymbol{K}_{ff} + \sigma_\mathrm{n}^2 \boldsymbol{I}\right)^{-1} \boldsymbol{k}_{f_* f}^\mathsf{T} \end{aligned}$$



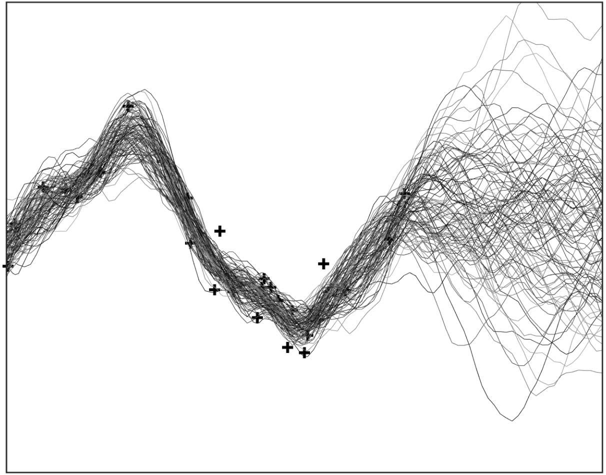
The input–output pairs



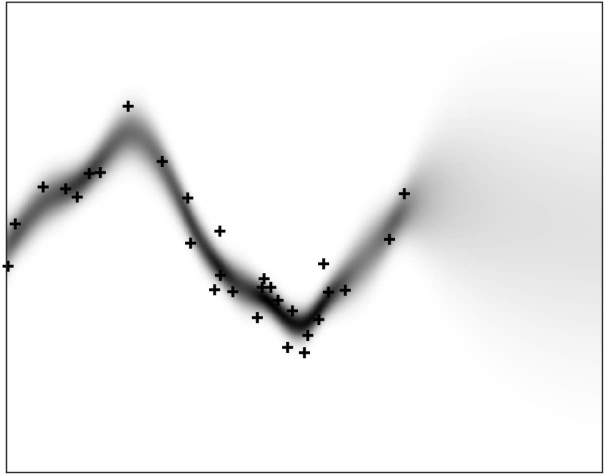
Draw from the GP posterior with a Matérn prior



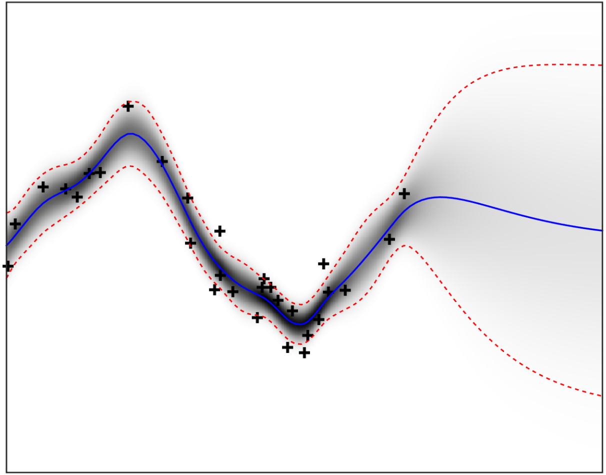
Draws from the GP posterior



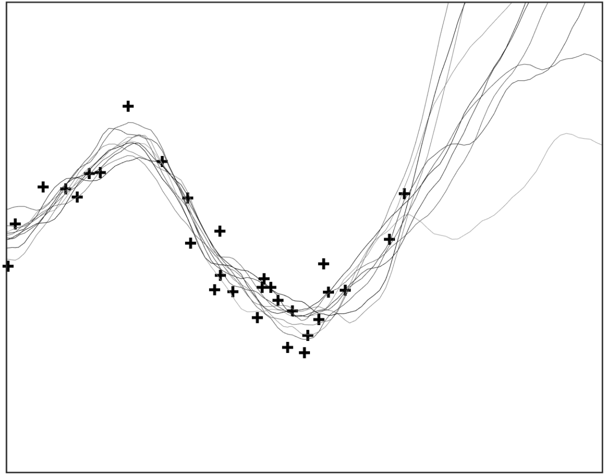
Draws from the GP posterior



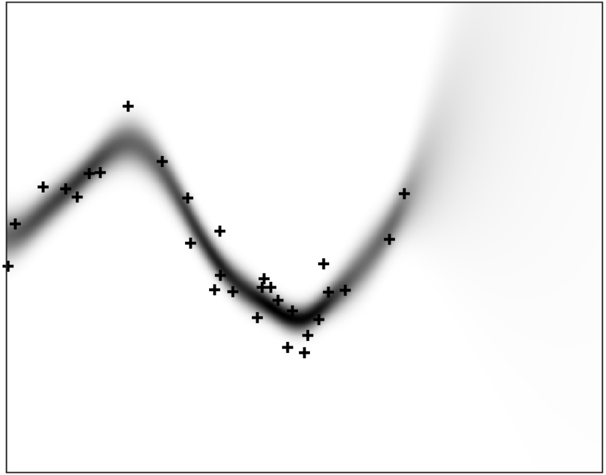
The GP posterior marginals



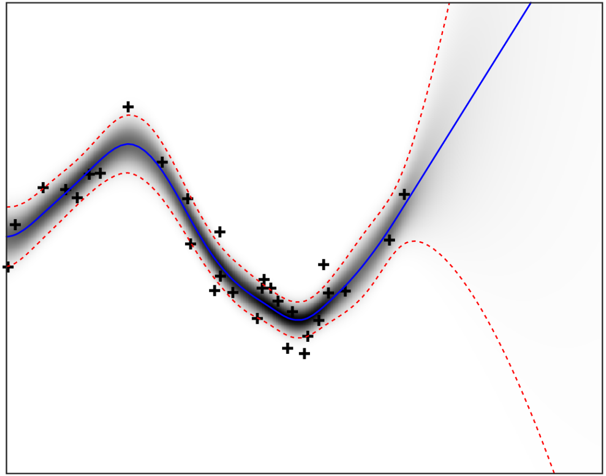
The stationary prior is mean-reverting



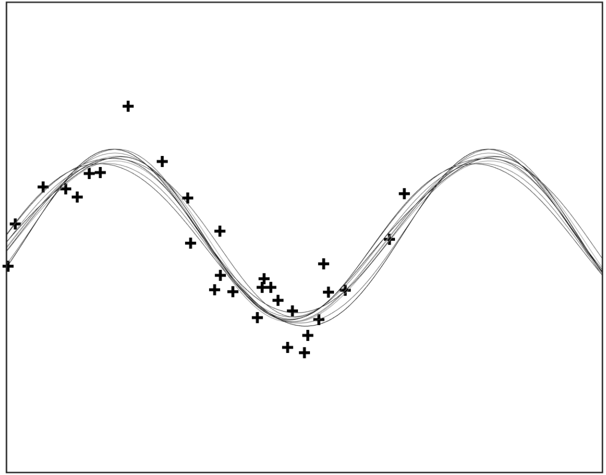
with a non-stationary prior



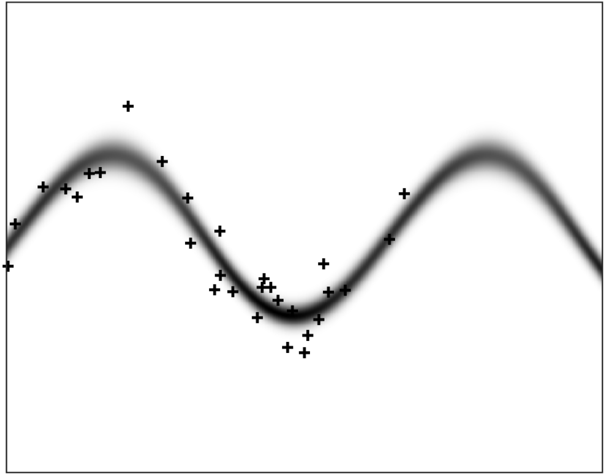
with a non-stationary prior



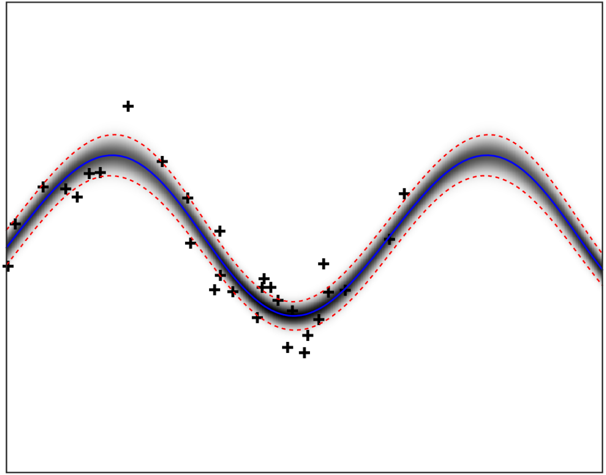
with a non-stationary prior



with a periodic prior



with a periodic prior



with a periodic prior

Challenges that break the beauty

GPs have three challenges

Scaling to large data

A nav̈e solution to dealing with the expanded Gram (covariance) matrix requires $\mathcal{O}(n^3)$ compute and $\mathcal{O}(n^2)$ memory. Infeasible for $n > 10{,}000$.

Dealing with non-conjugate likelihoods

For a Gaussian observation model the GP posterior is available in closed-form. For non-conjugate likelihood models one has to resort to approximate inference methods.

Representational power

Gaussian processes are ideal for problems where it is easy to specify *meaningful* priors. For applications such as image classification this is hard.

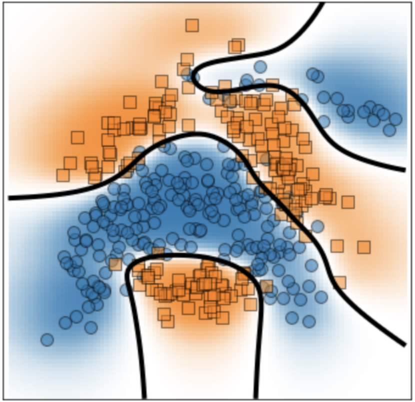
Scaling to large data

The naïve $\mathcal{O}(n^3)$ computational bottleneck ($\mathcal{O}(n^2)$ memory) can be tackled by

- Exploiting structure in the data
  (data on grid, inputs are in 1D, ...)
- Exploiting structure in the GP prior
  (GP prior is stationary, separable over input dimensions, ...)
- Solving the linear system approximately
  (conjugate-gradient solvers)
- Split problem into smaller chunks
  (local experts, subset of data, ...)
- Approximate the problem
  (Nyström, low-rank, inducing points, ...)
- Approximate the problem solution
  (SVGP = sparse (and stochastic) variational methods)

Dealing with non-conjugate likelihood models

- **MCMC (sampling) methods**
  (accurate but generally heavy)
- **Laplace approximation (LA)**
  (fast and simple)
- **Expectation propagation (EP)**
  (efficient but tricky)
- **Variational methods (VB/VI)**
  (popular but not problem-free)



GP classification with a Bernoulli likelihood

Representational power

- GPs can be seen as shallow, but infinitely wide models (see also deep GPs)
- Thus as such they are not ideal for problems where the data resides on some low-dimensional manifold in a high-dimensional space
- Instead, they can play a role as a building black of a larger model

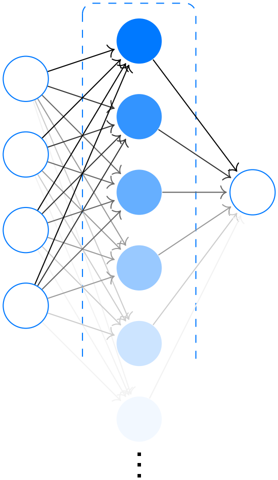
Connections and approaches to GPs

Connection to Neural Networks

- Radford Neal showed in the '90s that a random (untrained) single-layer feedforward network converges to a GP in the limit of infinite width.
- Let $\sigma(\cdot)$ be some non-linear (activation) function, and $\boldsymbol{w}$ and b be the network weights and biases.
- The associated kernel for the infinite-width network:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \int p(\boldsymbol{w})\, p(b)\, \sigma(\boldsymbol{w}^\mathsf{T} \boldsymbol{x} + b)\, \sigma(\boldsymbol{w}^\mathsf{T} \boldsymbol{x}' + b)\, \mathrm{d}\boldsymbol{w}\, \mathrm{d}b$$

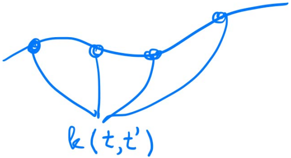
- The link can help analyze and understand NNs



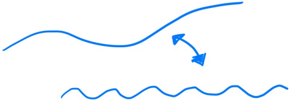
Connection to signal processing / SDEs

Alternative representations of GPs:

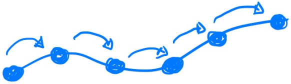
- **Moment representation**
  Considering the statistical properties of the input data jointly over time



- **Spectral (Fourier) representation**
  Analyzing the frequency-space representation of the problem/data

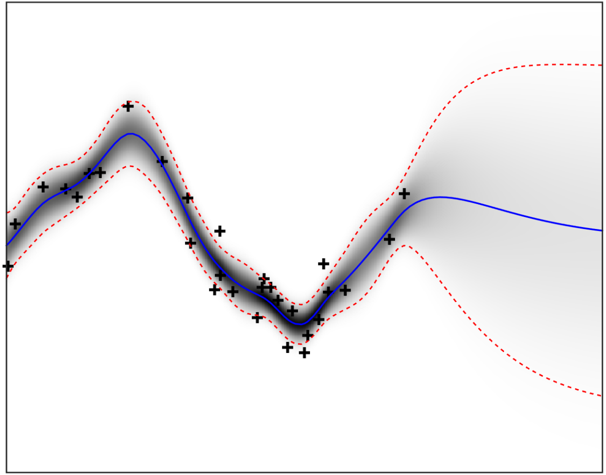


- **State space (path) representation**
  Description of sample behaviour as a dynamic system over time



S. Särkkä and A. Solin. *Applied Stochastic Differential Equations*.

Example: Exact GP regression in $\mathcal{O}(n)$



The state space representation enables efficient inference through Kalman filtering

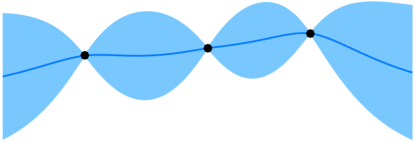
Connection to physics

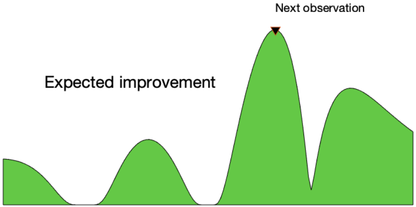
- First-principle models often written in terms of differential equations (ODEs, SDEs, PDEs, SPDEs)
- GPs used as structured priors ('latent forces') and for quantifying uncertainty
- GPs are preserved under linear operations (operating with linear operators)

Maxwell's equations induce a GP model for magnetic field variation

Connection to Bayesian optimization

- Sometimes the objective function in an optimization problem is expensive to evaluate
- In Bayesian optimization, a GP prior is used for cleverly guide where to observe the objective function next





R. Garnett. *Bayesian Optimization Book*. `https://bayesoptbook.com/`

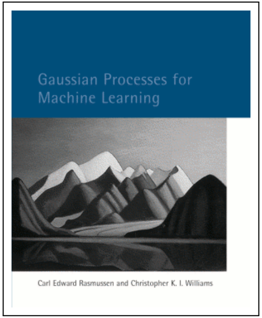
Recap and Q&A

Recap

A shallow but infinitely wide introduction to GPs

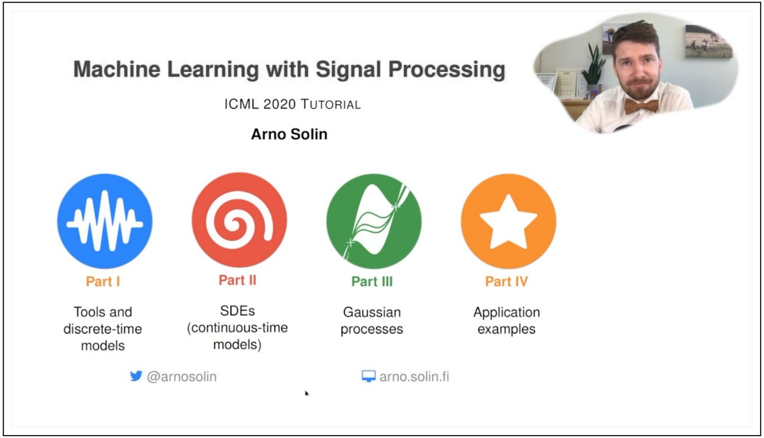
- Gaussian processes provide a plug-and-play framework for probabilistic inference and learning
- Give an explicit way of injecting prior knowledge into a problem
- Provide meaningful uncertainty estimates and means for quantifying uncertainty

Old but gold: The GP book



Carl Edward Rasmussen and Christopher K.I. Williams
Gaussian Processes for Machine Learning
The MIT Press, 2006. `http://gaussianprocess.org/gpml/`

Tutorial on Machine Learning with Signal Processing



https://youtu.be/vTRD03_yReI

Software packages

There are several software packages for working with GP models.
No package contains *everything*

- **GPflow**: `https://www.gpflow.org/`
- **GPyTorch**: `https://gpytorch.ai/`
- **GPy**: `https://sheffieldml.github.io/GPy/`

- **GPML**: `http://gaussianprocess.org/gpml/code`
- **GPstuff**: `https://research.cs.aalto.fi/pml/software/gpstuff/`

Gaussian Process Summer School

The next GPSS will be held in Sheffield, UK,
September 12–15, 2022

`http://gpss.cc/`