Gaussian Processes
Session 3 & 4

Arno Solin

Assistant Professor in Machine Learning
Department of Computer Science
Aalto University

OXML SUMMER SCHOOL

August 13, 2022

@arnosolin arno.solin.fi

Structure

Part III

Challenges that break the beauty

Part IV

Temporal Gaussian processes

Part V

Connections and approaches to GPs

Part VI

Recap and Q&A

Challenges that break the beauty

GPs have three challenges

Scaling to large data

A naïve 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.

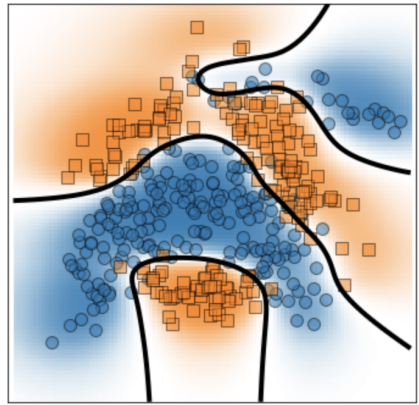
Scaling to large data

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)

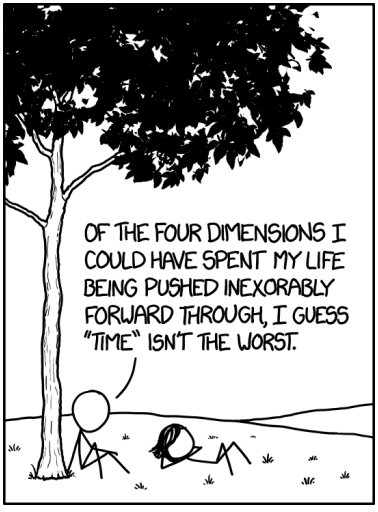


GP classification with a Bernoulli likelihood

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 block of a larger model

Temporal Gaussian processes



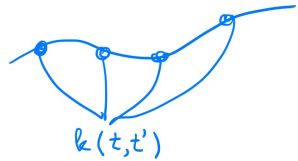
CC-NC: https://xkcd.com/1524/

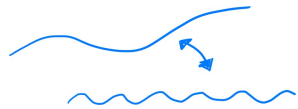
Motivation: Temporal models

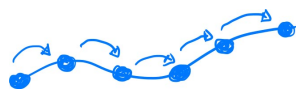
- **One-dimensional problems**
  (the data has a natural ordering)
- **Spatio-temproal models**
  (something developing over time)
- **Long / unbounded data**
  (sensor data streams, daily observations, etc.)

Tools for dealing with time-series

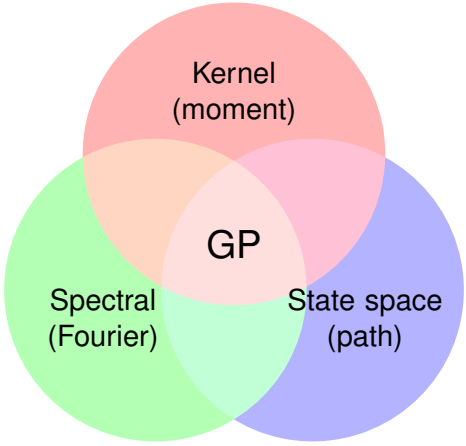
- Moment representation
  Considering the statistical properties of the input data jointly over time
- Spectral (Fourier) representation
  Analyzing the frequency-space representation of the problem/data
- State space (path) representation
  Description of sample behaviour as a dynamic system over time







Three views into (stationary) GPs



Kernel (moment) representation

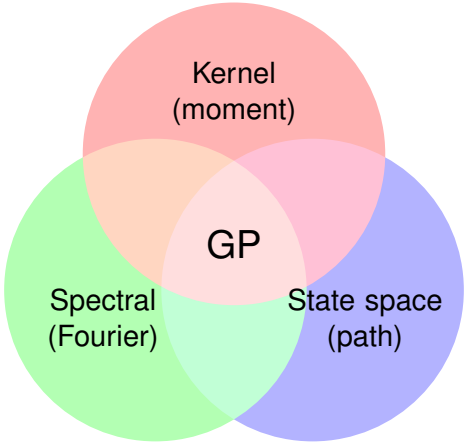
$$f(t) \sim \mathrm{GP}(\mu(t), \kappa(t, t')) \qquad \textit{GP prior}$$
$$\boldsymbol{y} \mid \boldsymbol{f} \sim \prod_i p(y_i \mid f(t_i)) \qquad \textit{likelihood}$$

- Let's focus on the GP prior only.
- A temporal Gaussian process (GP) is a random function $f(t)$, such that joint distribution of $f(t_1), \ldots, f(t_n)$ is always Gaussian.
- Mean and covariance functions have the form:

$$\mu(t) = \mathbb{E}[f(t)],$$
$$\kappa(t, t') = \mathbb{E}[(f(t) - \mu(t))(f(t') - \mu(t'))^{\mathsf{T}}].$$

- Convenient for model specification, but expanding the kernel to a covariance matrix can be problematic (the notorious $\mathcal{O}(n^3)$ scaling).

Three views into (stationary) GPs



Spectral (Fourier) representation

- The Fourier transform of a function $f(t) : \mathbb{R} \to \mathbb{R}$ is

$$\mathcal{F}[f](\mathrm{i}\,\omega) = \int_{\mathbb{R}} f(t)\, \exp(-\mathrm{i}\,\omega\, t)\, \mathrm{d}t$$

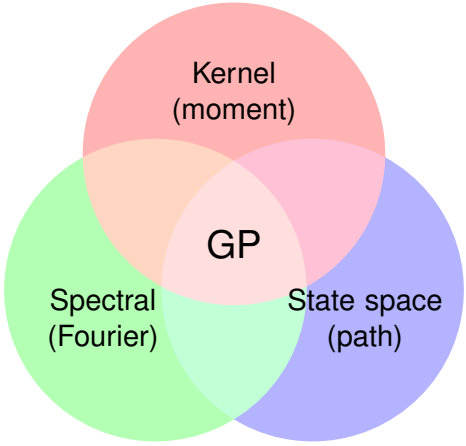
- For a stationary GP, the covariance function can be written in terms of the difference between two inputs:

$$\kappa(t, t') \triangleq \kappa(t - t')$$

- Wiener–Khinchin: If $f(t)$ is a stationary Gaussian process with covariance function $\kappa(t)$ then its spectral density is $S(\omega) = \mathcal{F}[\kappa]$.
- Spectral representation of a GP in terms of spectral density function

$$S(\omega) = \mathbb{E}[\tilde{f}(\mathrm{i}\,\omega)\, \tilde{f}^{\mathsf{T}}(-\mathrm{i}\,\omega)]$$

Three views into (stationary) GPs



State space (path) representation [1/3]

- Path or state space representation as solution to a linear time-invariant (LTI) stochastic differential equation (SDE):

$$\mathrm{d}\boldsymbol{f} = \boldsymbol{F}\,\boldsymbol{f}\,\mathrm{d}t + \boldsymbol{L}\,\mathrm{d}\boldsymbol{\beta},$$

  where $\boldsymbol{f} = (f, \mathrm{d}f/\mathrm{d}t, \ldots)$ and $\boldsymbol{\beta}(t)$ is a vector of Wiener processes.
- Equivalently, but more informally

$$\frac{\mathrm{d}\boldsymbol{f}(t)}{\mathrm{d}t} = \boldsymbol{F}\,\boldsymbol{f}(t) + \boldsymbol{L}\,\boldsymbol{w}(t),$$

  where $\boldsymbol{w}(t)$ is white noise.
- The model now consists of a drift matrix $\boldsymbol{F} \in \mathbb{R}^{m\times m}$, a diffusion matrix $\boldsymbol{L} \in \mathbb{R}^{m\times s}$, and the spectral density matrix of the white noise process $\boldsymbol{Q}_\mathrm{c} \in \mathbb{R}^{s\times s}$.
- The scalar-valued GP can be recovered by $f(t) = \boldsymbol{H}\,\boldsymbol{f}(t)$.

State space (path) representation [2/3]

- The initial state is given by a stationary state $\boldsymbol{f}(0) \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{P}_\infty)$ which fulfills

$$\boldsymbol{F}\,\boldsymbol{P}_\infty + \boldsymbol{P}_\infty\,\boldsymbol{F}^\mathsf{T} + \boldsymbol{L}\,\boldsymbol{Q}_\mathrm{c}\,\boldsymbol{L}^\mathsf{T} = \boldsymbol{0}$$

- The covariance function at the stationary state can be recovered by

$$\kappa(t, t') = \begin{cases} \boldsymbol{P}_\infty\,\exp((t'-t)\boldsymbol{F})^\mathsf{T}, & t' \geq t \\ \exp((t'-t)\boldsymbol{F})\,\boldsymbol{P}_\infty & t' < t \end{cases}$$

  where $\exp(\cdot)$ denotes the matrix exponential function.

- The spectral density function at the stationary state can be recovered by

$$S(\omega) = (\boldsymbol{F} + \mathrm{i}\,\omega\,\boldsymbol{I})^{-1}\,\boldsymbol{L}\,\boldsymbol{Q}_\mathrm{c}\,\boldsymbol{L}^\mathsf{T}\,(\boldsymbol{F} - \mathrm{i}\,\omega\,\boldsymbol{I})^{-\mathsf{T}}$$

State space (path) representation [3/3]

- Similarly as the kernel has to be evaluated into covariance matrix for computations, the SDE can be solved for discrete time points $\{t_i\}_{i=1}^n$.
- The resulting model is a discrete state space model:

$$\boldsymbol{f}_i = \boldsymbol{A}_{i-1}\,\boldsymbol{f}_{i-1} + \boldsymbol{q}_{i-1}, \quad \boldsymbol{q}_i \sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{Q}_i),$$

  where $\boldsymbol{f}_i = \boldsymbol{f}(t_i)$.
- The discrete-time model matrices are given by:

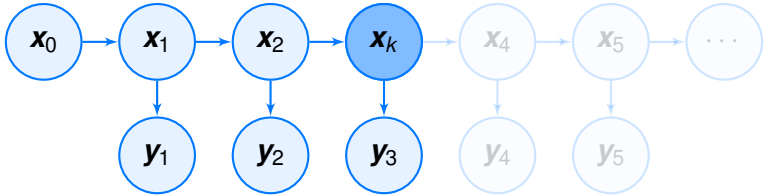
$$\boldsymbol{A}_i = \exp(\boldsymbol{F}\,\Delta t_i),$$

$$\boldsymbol{Q}_i = \int_0^{\Delta t_i} \exp(\boldsymbol{F}\,(\Delta t_i - \tau))\,\boldsymbol{L}\,\boldsymbol{Q}_\mathrm{c}\,\boldsymbol{L}^\mathsf{T}\,\exp(\boldsymbol{F}\,(\Delta t_i - \tau))^\mathsf{T}\,\mathrm{d}\tau,$$

  where $\Delta t_i = t_{i+1} - t_i$
- If the model is stationary, $\boldsymbol{Q}_i$ is given by

$$\boldsymbol{Q}_i = \boldsymbol{P}_\infty - \boldsymbol{A}_i\,\boldsymbol{P}_\infty\,\boldsymbol{A}_i^\mathsf{T}$$

Kalman filtering and smoothing

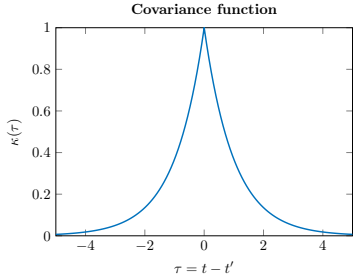


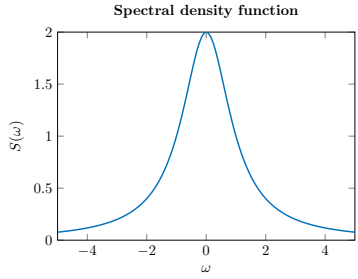
- Closed-form solution to linear-Gaussian filtering problems

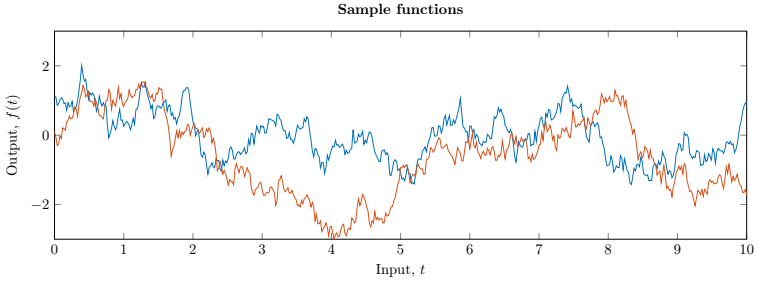
$$\begin{aligned} \boldsymbol{x}_i &= \boldsymbol{A}_{i-1}\,\boldsymbol{x}_{i-1} + \boldsymbol{q}_{i-1}, & \boldsymbol{q}_i &\sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{Q}_i), \\ \boldsymbol{y}_i &= \boldsymbol{H}\,\boldsymbol{x}_i + \boldsymbol{r}_i, & \boldsymbol{r}_i &\sim \mathrm{N}(\boldsymbol{0}, \boldsymbol{R}_i) \end{aligned}$$

- Filtering solution: $p(\boldsymbol{x}_i \mid \boldsymbol{y}_{1:i}) = \mathrm{N}(\boldsymbol{x}_i \mid \boldsymbol{m}_{i|i}, \boldsymbol{P}_{i|i})$
- Smoothing solution: $p(\boldsymbol{x}_i \mid \boldsymbol{y}_{1:T}) = \mathrm{N}(\boldsymbol{x}_i \mid \boldsymbol{m}_{i|T}, \boldsymbol{P}_{i|T})$

Three views into GPs







Example: Exponential covariance function

- Exponential covariance function (Ornstein-Uhlenbeck process):

$$\kappa(t, t') = \exp(-\lambda |t - t'|)$$

- Spectral density function:

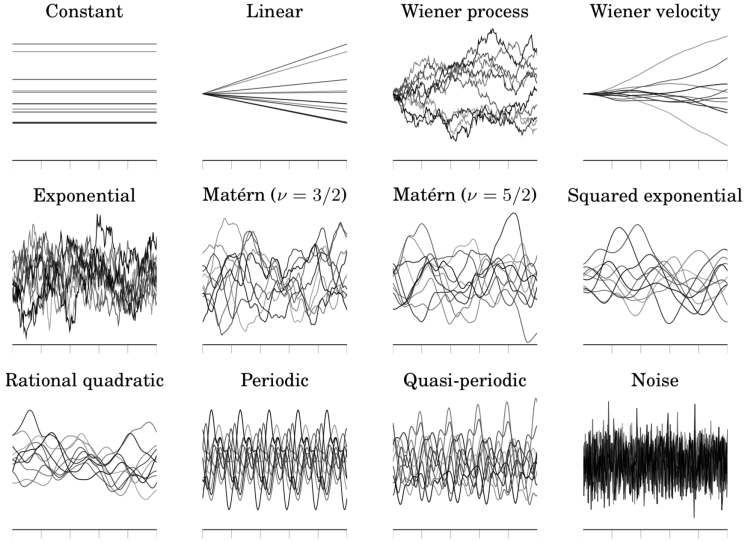
$$S(\omega) = \frac{2}{\lambda + \omega^2/\lambda}$$

- Path representation: Stochastic differential equation (SDE)

$$\frac{\mathrm{d}f(t)}{\mathrm{d}t} = -\lambda f(t) + w(t),$$

or using the notation from before:
$F = -\lambda$, $L = 1$, $Q_c = 2$, $H = 1$, and $P_\infty = 1$.

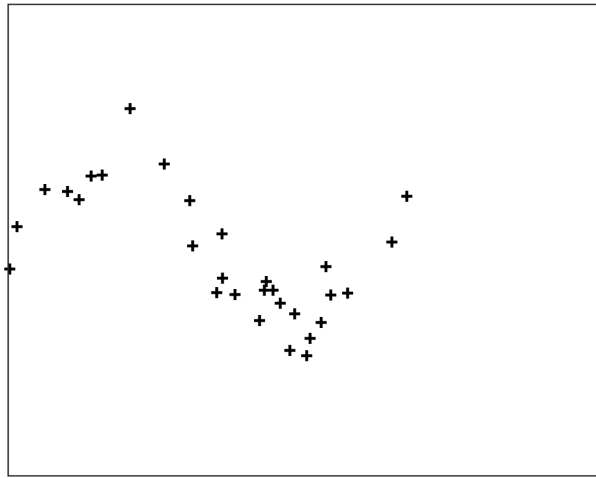
Examples of applicable GP priors



Applicable GP priors

- The covariance function needs to be Markovian (or approximated as such).
- Covers many common stationary and non-stationary models.
- Sums of kernels: $\kappa(t, t') = \kappa_1(t, t') + \kappa_2(t, t')$
  - Stacking of the state spaces
  - State dimension: $m = m_1 + m_2$
- Product of kernels: $\kappa(t, t') = \kappa_1(t, t')\,\kappa_2(t, t')$
  - Kronecker sum of the models
  - State dimension: $m = m_1\, m_2$

Example: GP regression, $\mathcal{O}(n^3)$



The input–output pairs

Example: GP regression, $\mathcal{O}(n^3)$

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

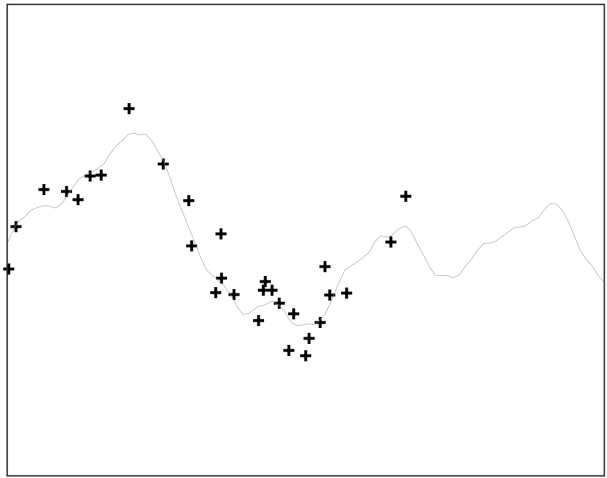
$$f(t) \sim \mathrm{GP}(0, \kappa(t, t')),$$
$$y_i = f(t_i) + \varepsilon_i, \quad \varepsilon_i \sim \mathrm{N}(0, \sigma_\mathrm{n}^2)$$

- The posterior mean and variance for an unseen test input t_* is given by (see previous lectures):

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

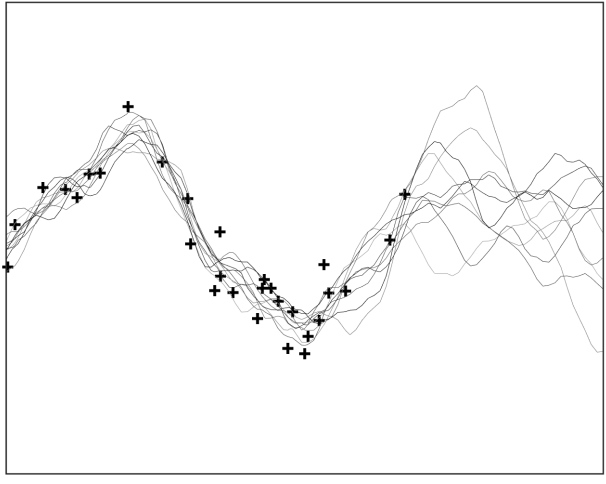
- Note the inversion of the $n \times n$ matrix.

Example: GP regression, $\mathcal{O}(n^3)$



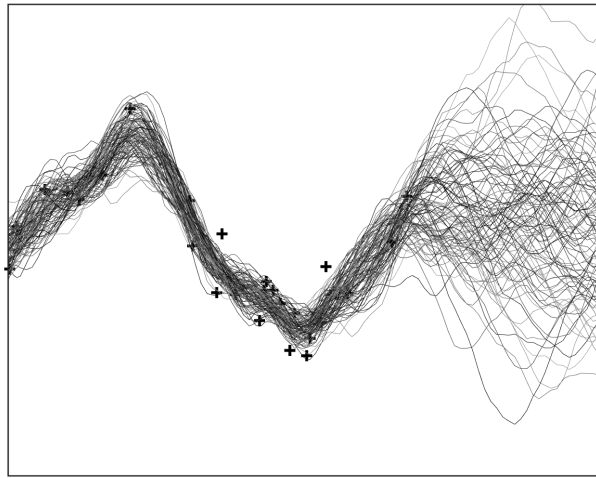
Draw from the GP posterior with a Matérn prior

Example: GP regression, $\mathcal{O}(n^3)$



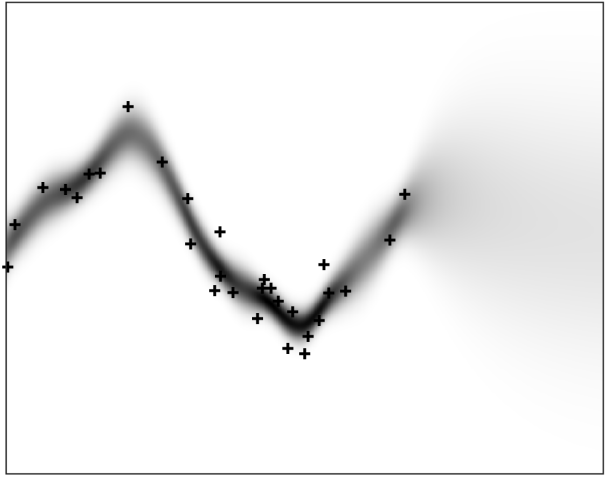
Draws from the GP posterior

Example: GP regression, $\mathcal{O}(n^3)$



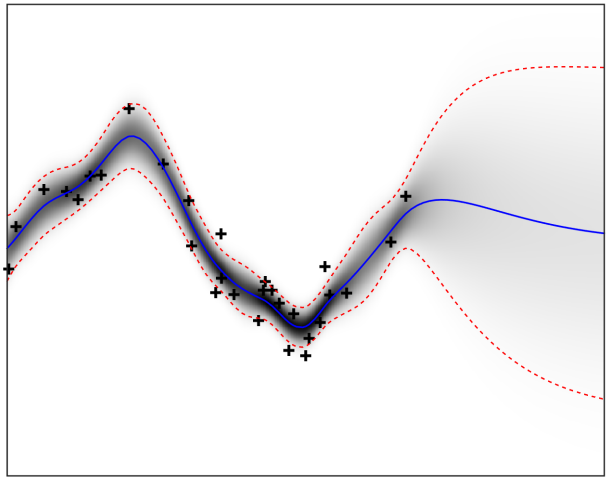
Draws from the GP posterior

Example: GP regression, $\mathcal{O}(n^3)$



The GP posterior marginals

Example: GP regression, $\mathcal{O}(n^3)$



The GP posterior marginals

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

- The sequential solution (goes under the name 'Kalman filter') considers one data point at a time, hence the linear time-scaling.
- Start from $\boldsymbol{m}_0 = \boldsymbol{0}$ and $\boldsymbol{P}_0 = \boldsymbol{P}_\infty$ and for each data point iterate the following steps.
- Kalman prediction:

$$\begin{aligned}\boldsymbol{m}_{i|i-1} &= \boldsymbol{A}_{i-1}\,\boldsymbol{m}_{i-1|i-1},\\ \boldsymbol{P}_{i|i-1} &= \boldsymbol{A}_{i-1}\,\boldsymbol{P}_{i-1|i-1}\,\boldsymbol{A}_{i-1}^{\mathsf{T}} + \boldsymbol{Q}_{i-1}.\end{aligned}$$

- Kalman update:

$$\begin{aligned}\boldsymbol{v}_i &= y_i - \boldsymbol{H}\,\boldsymbol{m}_{i|i-1},\\ \boldsymbol{S}_i &= \boldsymbol{H}_i\,\boldsymbol{P}_{i|i-1}\,\boldsymbol{H}^{\mathsf{T}} + \sigma_{\mathrm{n}}^2,\\ \boldsymbol{K}_i &= \boldsymbol{P}_{i|i-1}\,\boldsymbol{H}^{\mathsf{T}}\,\boldsymbol{S}_i^{-1},\\ \boldsymbol{m}_{i|i} &= \boldsymbol{m}_{i|i-1} + \boldsymbol{K}_i\,\boldsymbol{v}_i,\\ \boldsymbol{P}_{i|i} &= \boldsymbol{P}_{i|i-1} - \boldsymbol{K}_i\,\boldsymbol{S}_i\,\boldsymbol{K}_i^{\mathsf{T}}.\end{aligned}$$

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

- To condition all time-marginals on all data, run a backward sweep (Rauch–Tung–Striebel smoother):

$$\begin{aligned}
\boldsymbol{m}_{i+1|i} &= \boldsymbol{A}_i\, \boldsymbol{m}_{i|i}, \\
\boldsymbol{P}_{i+1|i} &= \boldsymbol{A}_i\, \boldsymbol{P}_{i|i}\, \boldsymbol{A}_i^\mathsf{T} + \boldsymbol{Q}_i, \\
\boldsymbol{G}_i &= \boldsymbol{P}_{i|i}\, \boldsymbol{A}_i^\mathsf{T}\, \boldsymbol{P}_{i+1|i}^{-1}, \\
\boldsymbol{m}_{i|n} &= \boldsymbol{m}_{i|i} + \boldsymbol{G}_i\, (\boldsymbol{m}_{i+1|n} - \boldsymbol{m}_{i+1|i}), \\
\boldsymbol{P}_{i|n} &= \boldsymbol{P}_{i|i} + \boldsymbol{G}_i\, (\boldsymbol{P}_{i+1|n} - \boldsymbol{P}_{i+1|i})\, \boldsymbol{G}_i^\mathsf{T},
\end{aligned}$$

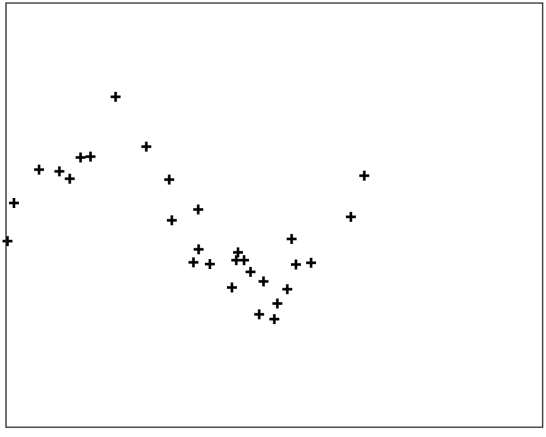
- The marginal mean and variance can be recovered by:

$$\mathbb{E}[f_i] = \boldsymbol{H}\, \boldsymbol{m}_{i|n} \quad \text{and} \quad \mathbb{V}[f_i] = \boldsymbol{H}\, \boldsymbol{P}_{i|n}\, \boldsymbol{H}^\mathsf{T}$$

- The log marginal likelihood evaluated as a by-product of the Kalman update:

$$\log p(\boldsymbol{y}) = -\frac{1}{2} \sum_{i=1}^{n} \log |2\pi\, \boldsymbol{S}_i| + \boldsymbol{v}_i^\mathsf{T}\, \boldsymbol{S}_i^{-1}\, \boldsymbol{v}_i$$

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



The state space representation enables efficient inference through Kalman filtering

Example: Births in the US

- Number of births in the US
- Daily data between 1969–1988 ($n = 7305$)
- GP regression with a prior covariance function:

$$\kappa(t, t') = \kappa_{\text{Mat.}}^{\nu=5/2}(t, t') + \kappa_{\text{Mat.}}^{\nu=3/2}(t, t') + \kappa_{\text{Per.}}^{\text{year}}(t, t')\, \kappa_{\text{Mat.}}^{\nu=3/2}(t, t') + \kappa_{\text{Per.}}^{\text{week}}(t, t')\, \kappa_{\text{Mat.}}^{\nu=3/2}(t, t')$$

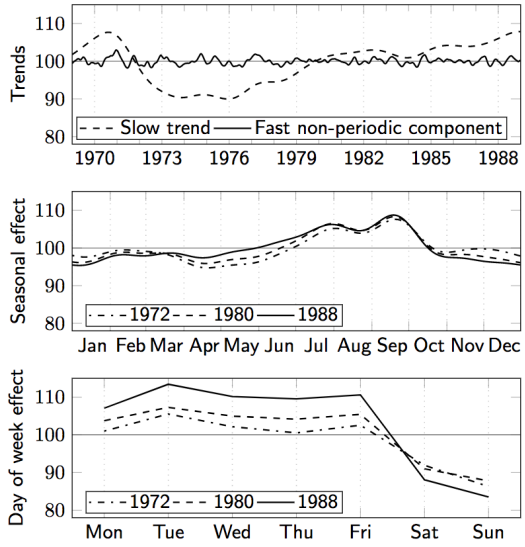
- Learn hyperparameters by optimizing the marginal likelihood

- Number of birth
- Daily data betw
- GP regression

$\kappa(t, t')$

$\kappa_{\text{Mat.}}^{\nu=3/2}(t, t')$

- Learn hyperpar od



Explaining changes in number of births in the US

Example: Aircraft accidents

- Commercial aircraft accidents 1919–2017
- Log-Gaussian Cox process (Poisson likelihood) by ADF/EP
- Daily binning, $n = 35{,}959$
- GP prior with a covariance function:

$$\kappa(t,t') = \kappa_{\text{Mat.}}^{\nu=3/2}(t,t') + \kappa_{\text{Per.}}^{\text{year}}(t,t')\,\kappa_{\text{Mat.}}^{\nu=3/2}(t,t') + \kappa_{\text{Per.}}^{\text{week}}(t,t')\,\kappa_{\text{Mat.}}^{\nu=3/2}(t,t')$$

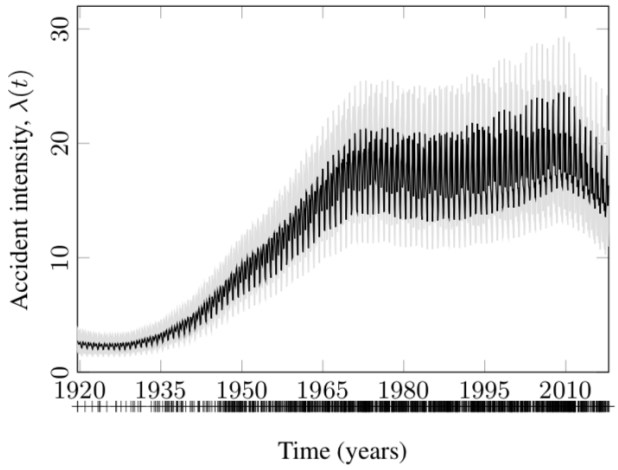
- Learn hyperparameters by optimizing the marginal likelihood

Example: Aircraft accidents

- Commercia
- Log-Gauss
- Daily binnir
- GP prior wi

κ /2(t, t′)

- Learn hype

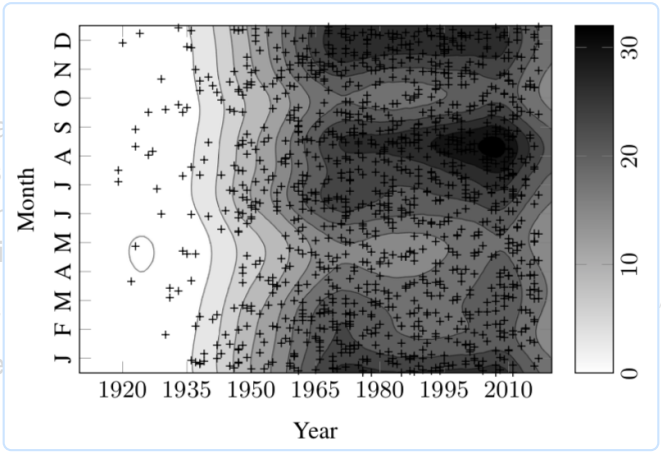


Example: Aircraft accidents

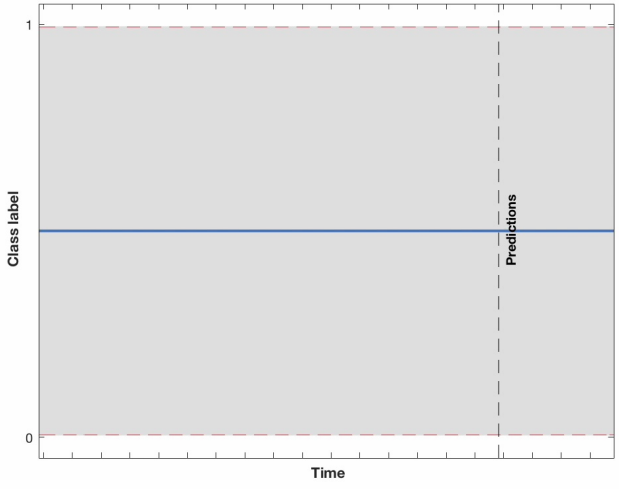
- Commercia
- Log-Gauss
- Daily binnir
- GP prior wi

κ $^{/2}(t, t')$

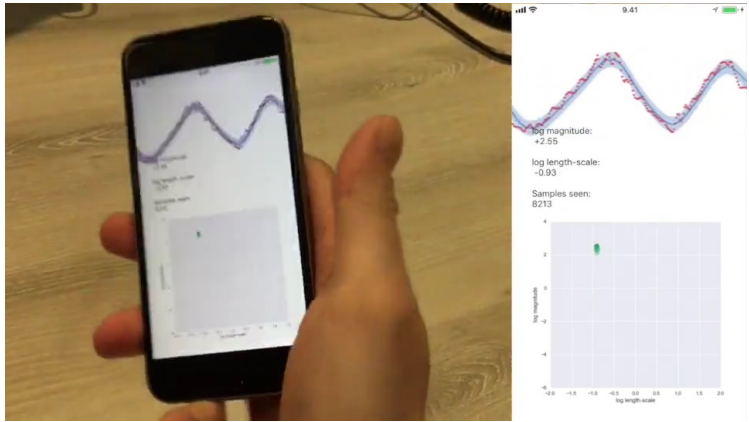
- Learn hype



What if the data really is infinite?



On-line inference by infinite time-horizon GPs



https://youtu.be/myCvUT3XGPc

Spatio-temporal GPs

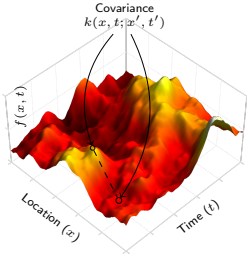
$$f(\boldsymbol{x}) \sim \mathrm{GP}(0, \kappa(\boldsymbol{x}, \boldsymbol{x}'))$$
$$\boldsymbol{y} \mid \boldsymbol{f} \sim \prod_i p(y_i \mid f(\boldsymbol{x}_i))$$

$$f(\boldsymbol{r}, t) \sim \mathrm{GP}(0, \kappa(\boldsymbol{r}, t; \boldsymbol{r}', t'))$$
$$\boldsymbol{y} \mid \boldsymbol{f} \sim \prod_i p(y_i \mid f(\boldsymbol{r}_i, t_i))$$

Spatio-temporal Gaussian processes

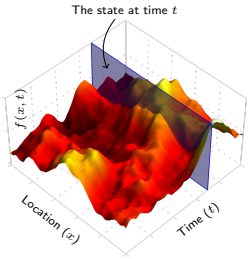
GPs under the kernel formalism

$$f(\boldsymbol{x}, t) \sim \mathrm{GP}(0, k(\boldsymbol{x}, t; \boldsymbol{x}', t'))$$
$$y_i = f(\boldsymbol{x}_i, t_i) + \varepsilon_i$$

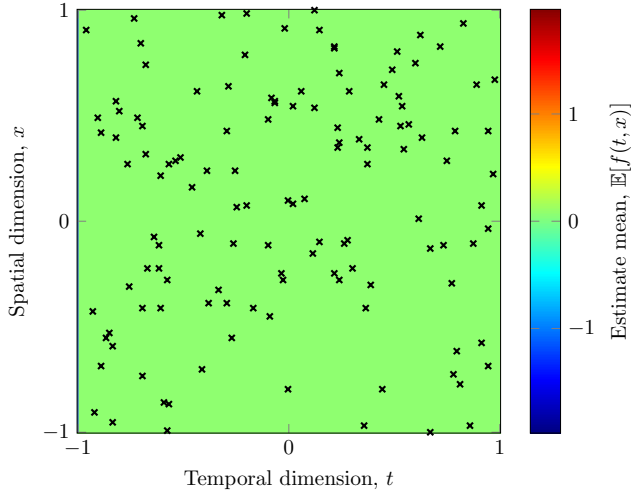


Stochastic partial differential equation formalism

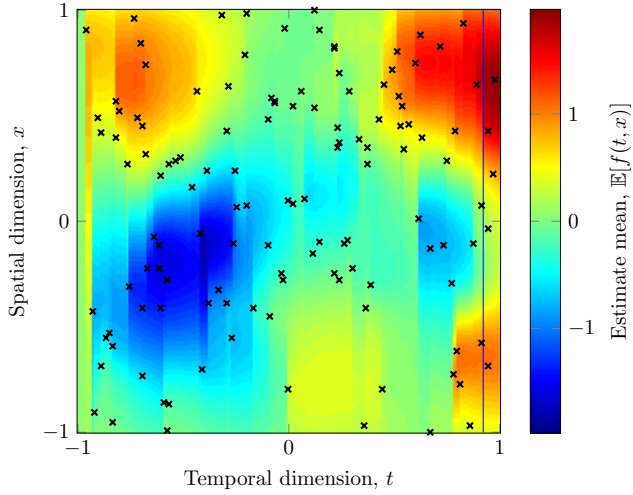
$$\frac{\partial \boldsymbol{f}(\boldsymbol{x}, t)}{\partial t} = \mathcal{F}\, \boldsymbol{f}(\boldsymbol{x}, t) + \mathcal{L}\, w(\boldsymbol{x}, t)$$
$$y_i = \mathcal{H}_i\, \boldsymbol{f}(\boldsymbol{x}, t) + \varepsilon_i$$



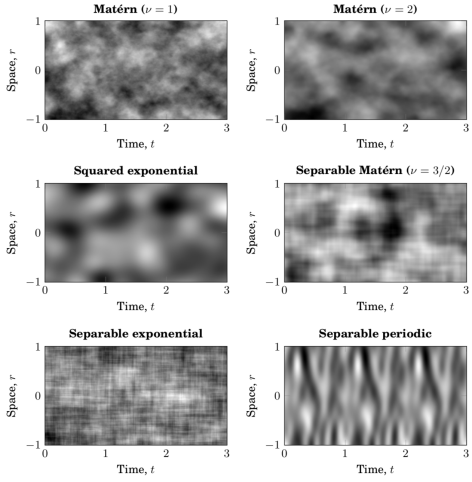
Spatio-temporal GP regression



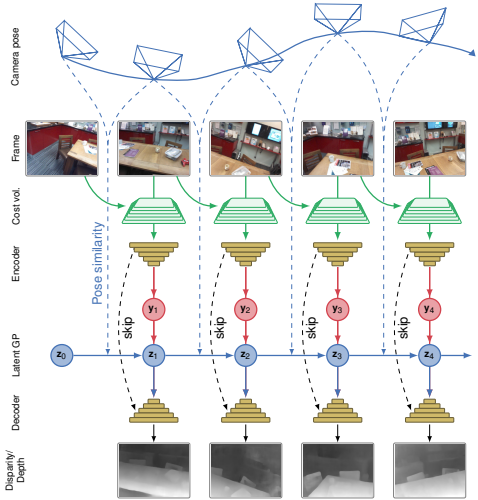
Spatio-temporal GP regression



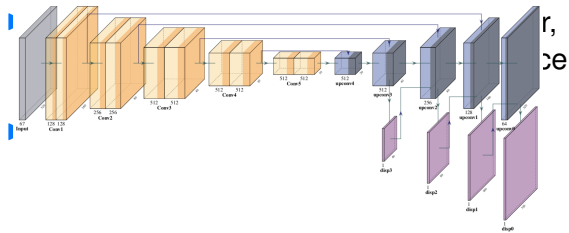
Spatio-temporal GP priors



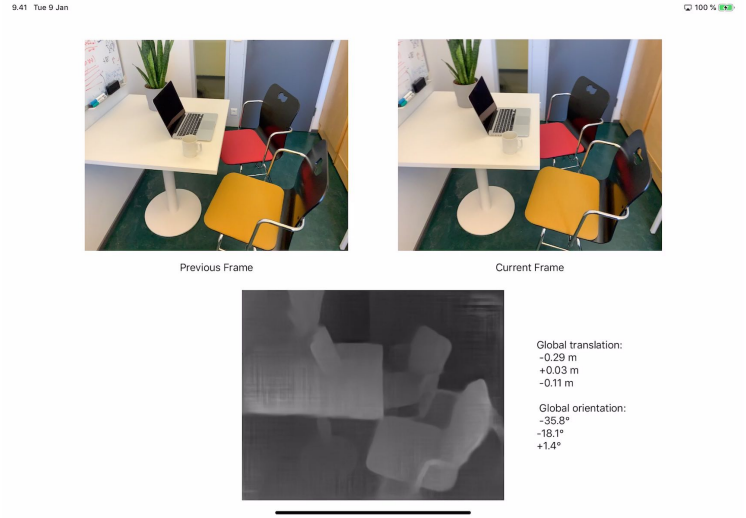
Priors in larger models



- Inputs: Frame pairs and relative camera poses
- State-of-the-art in CV: Encoder–decoder network for depth estimation



Online inference on an iPad



https://youtu.be/iellGrlNW7k

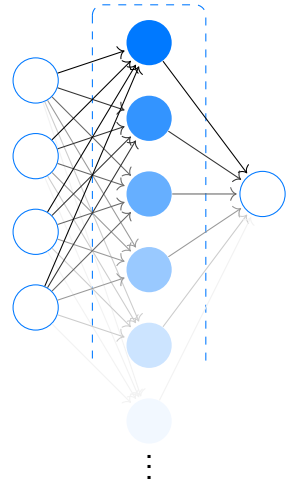
Connections and approaches to GPs

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:

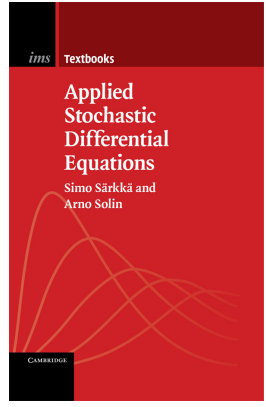
$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \int p(\boldsymbol{w})\, p(b)\, \sigma(\boldsymbol{w}^\mathsf{T}\boldsymbol{x}+b)\, \sigma(\boldsymbol{w}^\mathsf{T}\boldsymbol{x}'+b)\, \mathrm{d}\boldsymbol{w}\, \mathrm{d}b$$

- The link can help analyze and understand NNs



Connection to more general SDEs

- More general stochastic differential equation (‘non-Gaussian processes’) models are widely used across fields (e.g., Black–Scholes)
- GPs can be used as driving ‘latent forces’ that inject structured noise into the model
- These models do not generally have Gaussian marginals and the set of tools for simulating from and ‘solving’ the SDE become diverse



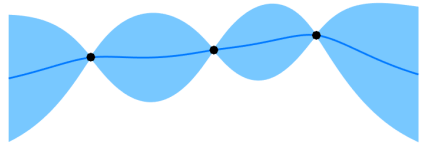
Connection to physics / first principles

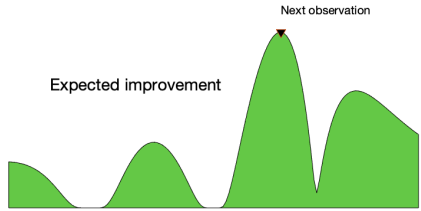
- First-principle models often written in terms of differential equations (ODEs, SDEs, PDEs, SPDEs)
- GPs used as structured priors ('latent forces') and for quantifying uncertainty
- GPs are preserved under linear operations (operating with linear operators)

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

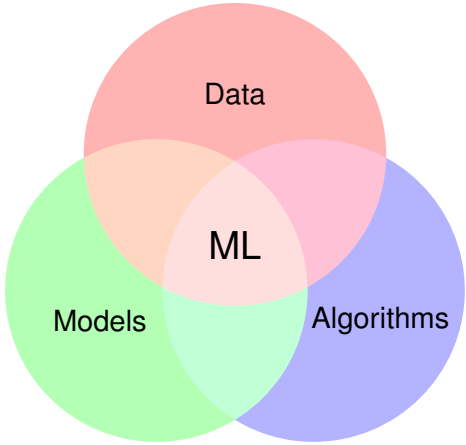




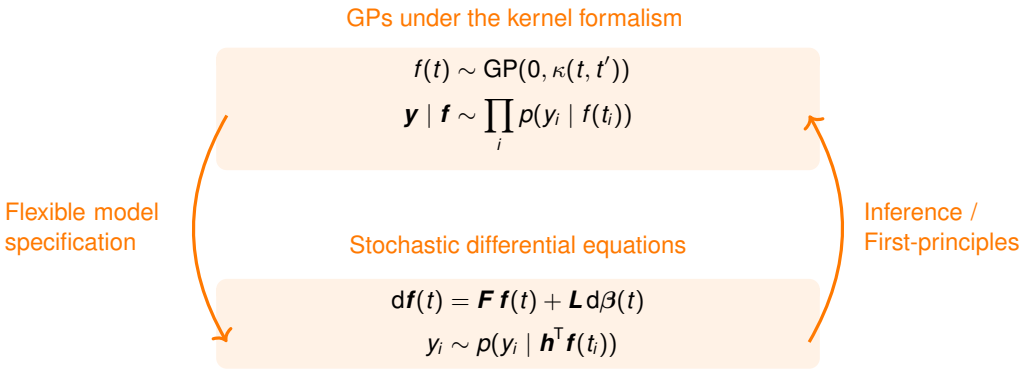
R. Garnett. *Bayesian Optimization Book*. `https://bayesoptbook.com/`

Recap and Q&A

Gaussian processes and ML



Gaussian processes ❤ SDEs



Different representations of GPs

- Gaussian processes have different representations:
  • Covariance function • Spectral density • State space
- Temporal (single-input) Gaussian processes
  $\iff$ stochastic differential equations (SDEs)
- Conversions between the representations can make model building easier
- (Exact) inference of the latent functions, can be done in $\mathcal{O}(n)$ time and memory complexity by Kalman filtering

Summary

- 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

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/`

Bibliography

Most examples and methods presented on this lecture are presented in greater detail in the following works:

- Särkkä, S., Solin, A., and Hartikainen, J. (2013). Spatio-temporal learning via infinite-dimensional Bayesian filtering and smoothing. *IEEE Signal Processing Magazine*, 30(4):51–61.
- Särkkä, S. (2013). *Bayesian Filtering and Smoothing*. Cambridge University Press. Cambridge, UK.
- Solin, A. (2016). *Stochastic Differential Equation Methods for Spatio-Temporal Gaussian Process Regression*. Doctoral dissertation, Aalto University.
- Solin, A., Hensman, J., and Turner, R.E. (2018). Infinite-horizon Gaussian processes. *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3490–3499. Montréal, Canada.
- Särkkä, S., and Solin, A. (2019). *Applied Stochastic Differential Equations*. Cambridge University Press. Cambridge, UK.