Gaussian random fields in machine learning

Viacheslav Borovitskiy

St. Petersburg State University
St. Petersburg Department of Steklov Mathematical Institute

Winter School in Mathematics and Theoretical Computer Science
January 29 – February 3, 2021

Mini course structure

Lecture 1 Intro to Bayesian methods & Gaussian process regression, applications.
Almost no formulas, a hand-wavy exposition with lots of pictures.

Lecture 2 Predicting with Gaussian random fields and generating their sample paths.
A more rigorous intro to Gaussian process regression. Basic algorithms and their downsides. Can we do better?

Lecture 3 Efficient algorithms for sampling and conditioning.
Sampling stationary Gaussian fields. Sampling conditional Gaussian fields. Approximate conditioning of Gaussian fields. Conclusion.

Part I

Introduction to Bayesian methods
Gaussian process regression
Applications

Today's talk structure

1. Introduction
2. Bayesian inference for an unfair coin
3. Gaussian processes
4. Applications

Outline

Language and notation

"Gaussian process (GP)" and "Gaussian random field (GRF)" — interchangeably.

Sometimes I may use the Bayesian language. For instance,

$p(a)$ — density of random vector a,
$p(b)$ — density of random vector b.

GPs are indeed useful

Bayesian Optimization in AlphaGo

Yutian Chen, Aja Huang, Ziyu Wang, Ioannis Antonoglou, Julian Schrittwieser, David Silver & Nando de Freitas

DeepMind, London, UK
yutianc@google.com

They used GPs to model target function and guide decision (optimization) process.

Outline

Problem setup

The problem: estimate the unknown parameter of an unfair coin.

Let X be a random variable modeling an unfair coin.
It takes two values: 1 (for heads) and 0 (for tails)

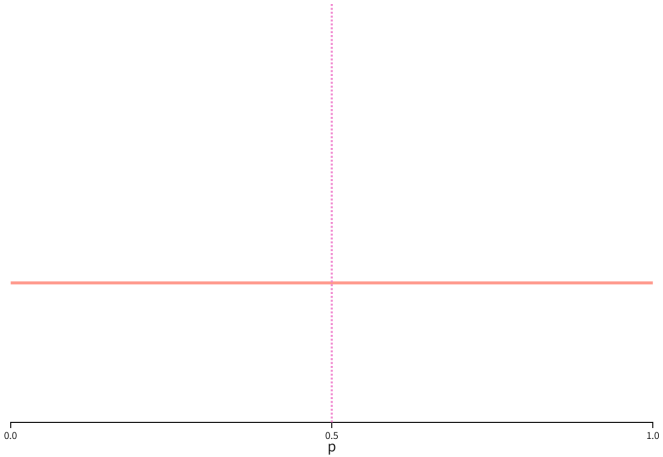
$$\mathbb{P}(X = 1) = p, \qquad \mathbb{P}(X = 0) = 1 - p.$$

We want to estimate p.

Frequentist approach	Bayesian approach
Result: a number $\hat{p}$.	Result: a distribution (density) $\hat{\rho}(p)$.
Tool: maximum likelihood estimation.	Tool: Bayes theorem.
Extra: —.	Extra: requires a prior density $\rho(p)$.
$\hat{p} = \arg\max p^{\#1}(1-p)^{\#0}$	$\hat{\rho}(p) \propto p^{\#1}(1-p)^{\#0}\rho(p)$

Example

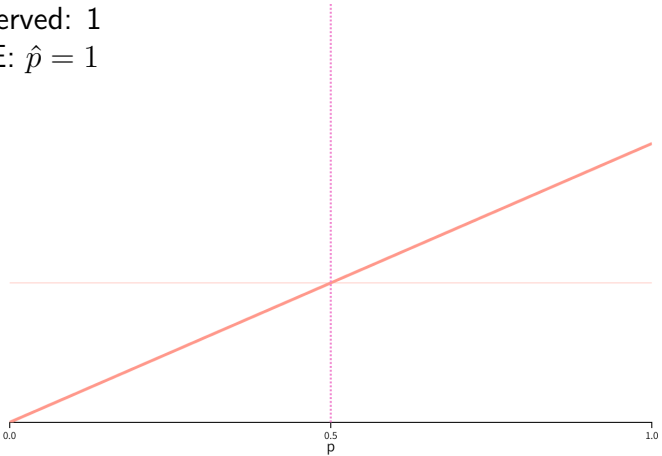
Uniform prior $\rho(p) := \mathbb{1}_{[0,1]}(p)$ and Bernoulli likelihood $\mathbb{P}(X = v \mid p) = p^v(1-p)^{1-v}$.
Assume that true p is 0.5,



Example

Uniform prior $\rho(p) := \mathbb{1}_{[0,1]}(p)$ and Bernoulli likelihood $\mathbb{P}(X = v \mid p) = p^v(1-p)^{1-v}$.
Assume that true p is 0.5,

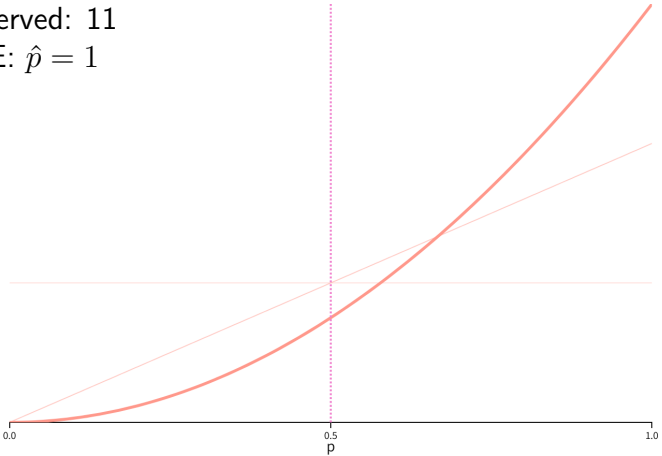
Observed: 1
MLE: $\hat{p} = 1$



Example

Uniform prior $\rho(p) := \mathbb{1}_{[0,1]}(p)$ and Bernoulli likelihood $\mathbb{P}(X = v \mid p) = p^v(1-p)^{1-v}$.
Assume that true p is 0.5,

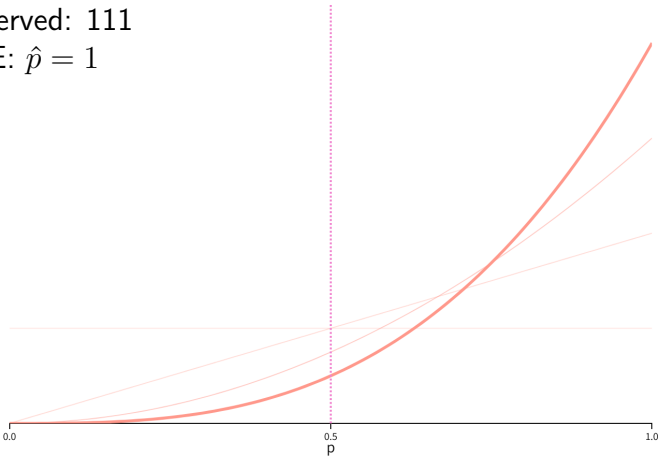
Observed: 11
MLE: $\hat{p} = 1$



Example

Uniform prior $\rho(p) := \mathbb{1}_{[0,1]}(p)$ and Bernoulli likelihood $\mathbb{P}(X = v \mid p) = p^v(1-p)^{1-v}$.
Assume that true p is 0.5,

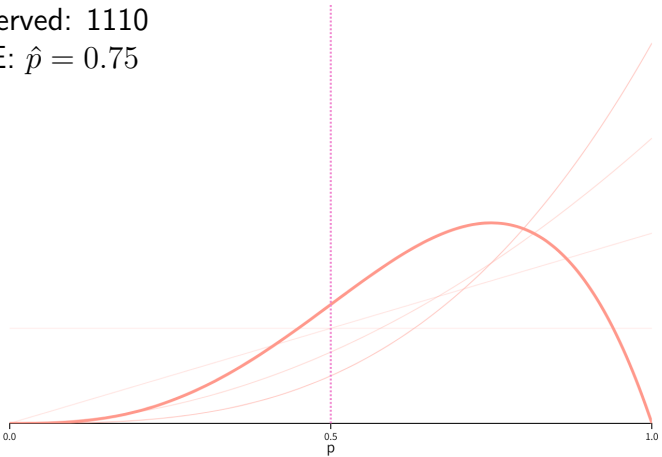
Observed: 111
MLE: $\hat{p} = 1$



Example

Uniform prior $\rho(p) := \mathbb{1}_{[0,1]}(p)$ and Bernoulli likelihood $\mathbb{P}(X = v \mid p) = p^v(1-p)^{1-v}$.
Assume that true p is 0.5,

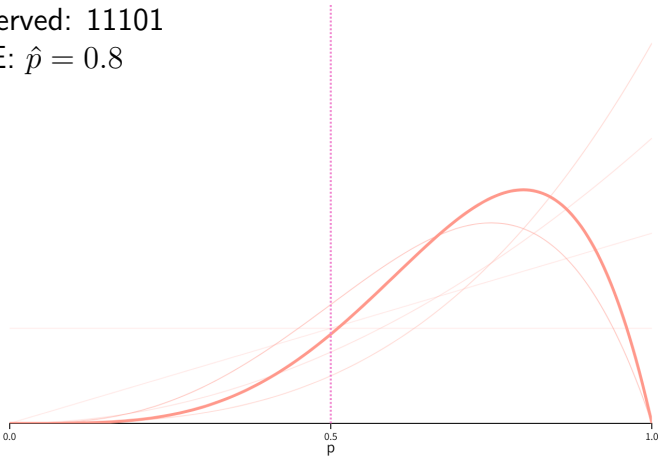
Observed: 1110
MLE: $\hat{p} = 0.75$



Example

Uniform prior $\rho(p) := \mathbb{1}_{[0,1]}(p)$ and Bernoulli likelihood $\mathbb{P}(X = v \mid p) = p^v(1-p)^{1-v}$.
Assume that true p is 0.5,

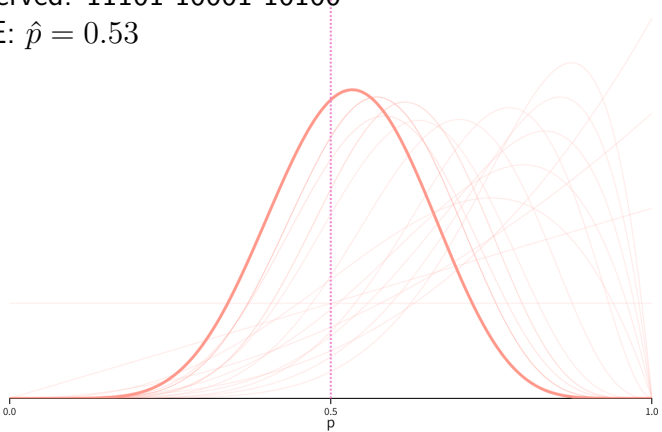
Observed: 11101
MLE: $\hat{p} = 0.8$



Example

Uniform prior $\rho(p) := \mathbb{1}_{[0,1]}(p)$ and Bernoulli likelihood $\mathbb{P}(X = v \mid p) = p^v(1-p)^{1-v}$.
Assume that true p is 0.5,

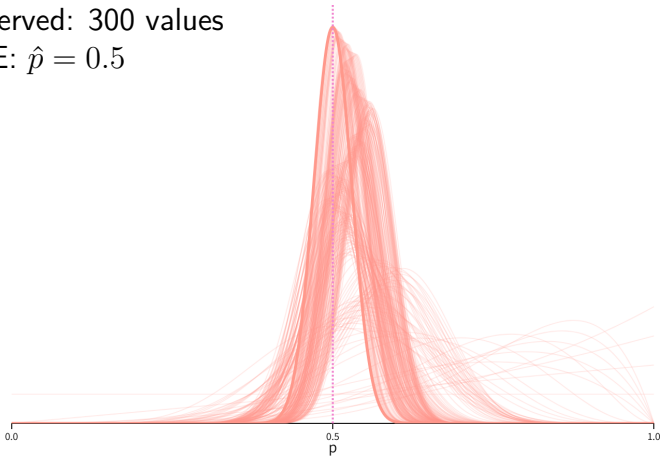
Observed: 11101 10001 10100
MLE: $\hat{p} = 0.53$



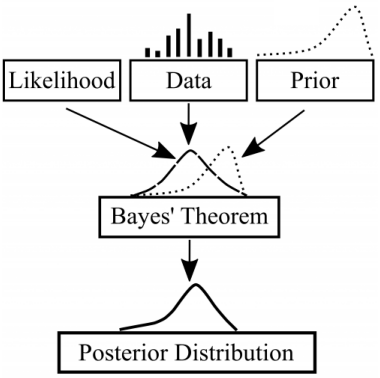
Example

Uniform prior $\rho(p) := \mathbb{1}_{[0,1]}(p)$ and Bernoulli likelihood $\mathbb{P}(X = v \mid p) = p^v(1-p)^{1-v}$.
Assume that true p is 0.5,

Observed: 300 values
MLE: $\hat{p} = 0.5$



Result



Most importantly, Bayesian approach quantifies uncertainty.

Gaussian processes (GPs) — non-parametric prior over functions.

Outline

Gaussian process regression

GP — distribution over functions.

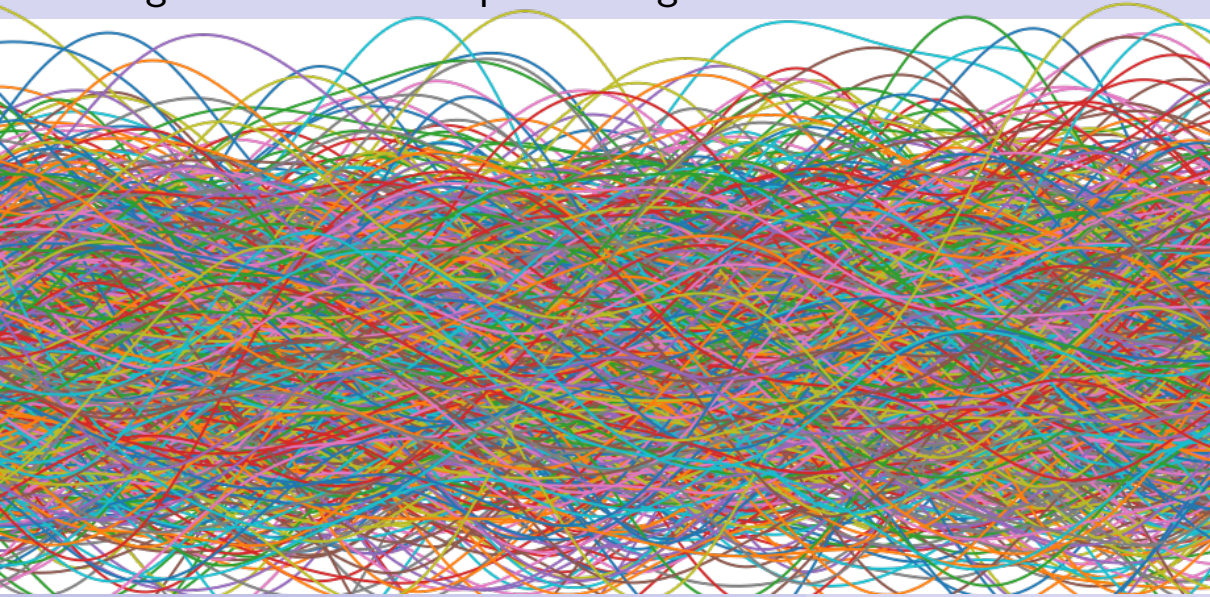
Bayesian inference for GPs:

prior:	hand-picked GP
data:	noisy evaluations of the function
likelihood:	induced by Gaussian noise assumption
posterior:	another GP

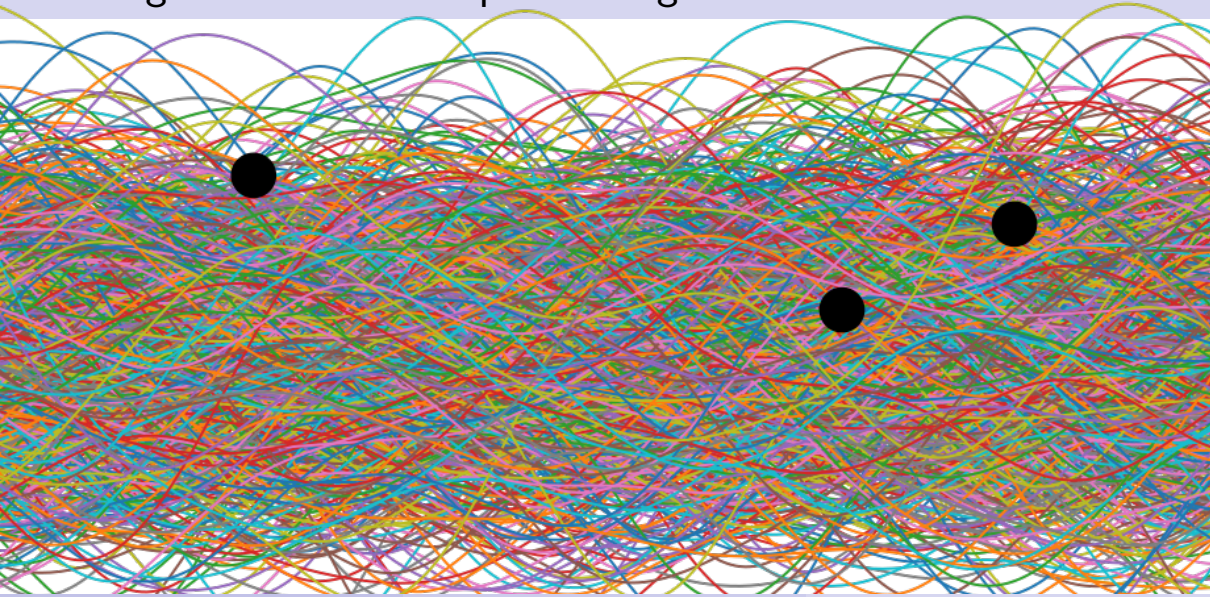
Let us explore this visually . . .

Visual guide to Gaussian process regression

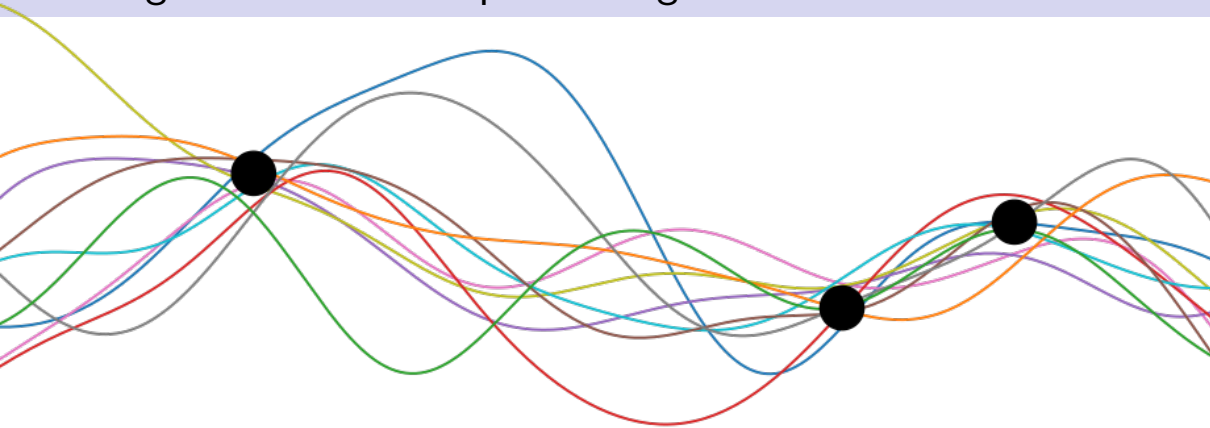
Visual guide to Gaussian process regression



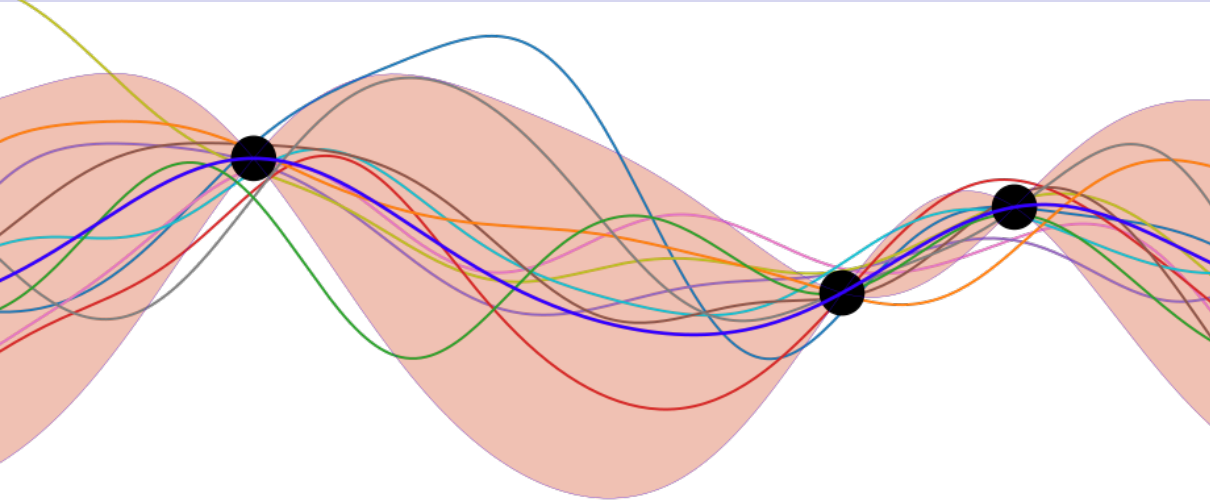
Visual guide to Gaussian process regression



Visual guide to Gaussian process regression



Visual guide to Gaussian process regression



What is a Gaussian process?

Gaussian random variable

- distribution over $\mathbb{R}$, denoted by $\mathrm{N}(\mu, \sigma^2)$,
- determined by two numbers: mean μ and variance σ^2.

Multivariate Gaussian random variable

- distribution over $\mathbb{R}^d$, denoted by $\mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$,
- determined by the mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$.

Gaussian process

- distribution over functions from X to $\mathbb{R}$, denoted by $\mathrm{GP}(m, k)$,
- determined by two functions $m : X \to \mathbb{R}$ (mean) and $k : X \times X \to \mathbb{R}$ (covariance).

Gaussian processes are appealing in practice due to their simplicity (among other stochastic processes).

Bayesian inference for GPs

Bayesian inference for GPs takes in

- a prior distribution over functions of form $GP(m, k)$,
- noisy evaluations $y_1, .., y_n$ of the unknown function of interest at $x_1, .., x_n$.

and returns the distribution over functions of form

$$GP(\hat{m}, \hat{k}).$$

Given m and k, the functions $\hat{m}$ and $\hat{k}$ can be computed in a finite time. Specifically:

$$\hat{m}(u) = m(u) + \mathbf{K}_{f(u)f(\boldsymbol{x})}\Big(\mathbf{K}_{f(\boldsymbol{x})f(\boldsymbol{x})} + \sigma^2 I\Big)^{-1}(\boldsymbol{y} - m(\boldsymbol{x}))$$

$$\hat{k}(u, v) = k(u, v) - \underbrace{\mathbf{K}_{f(u)f(\boldsymbol{x})}}_{\text{vector } 1\times n}\underbrace{\Big(\mathbf{K}_{f(\boldsymbol{x})f(\boldsymbol{x})} + \sigma^2 I\Big)^{-1}}_{\text{matrix } n\times n}\underbrace{\mathbf{K}_{f(\boldsymbol{x})f(v)}}_{\text{vector } n\times 1}.$$

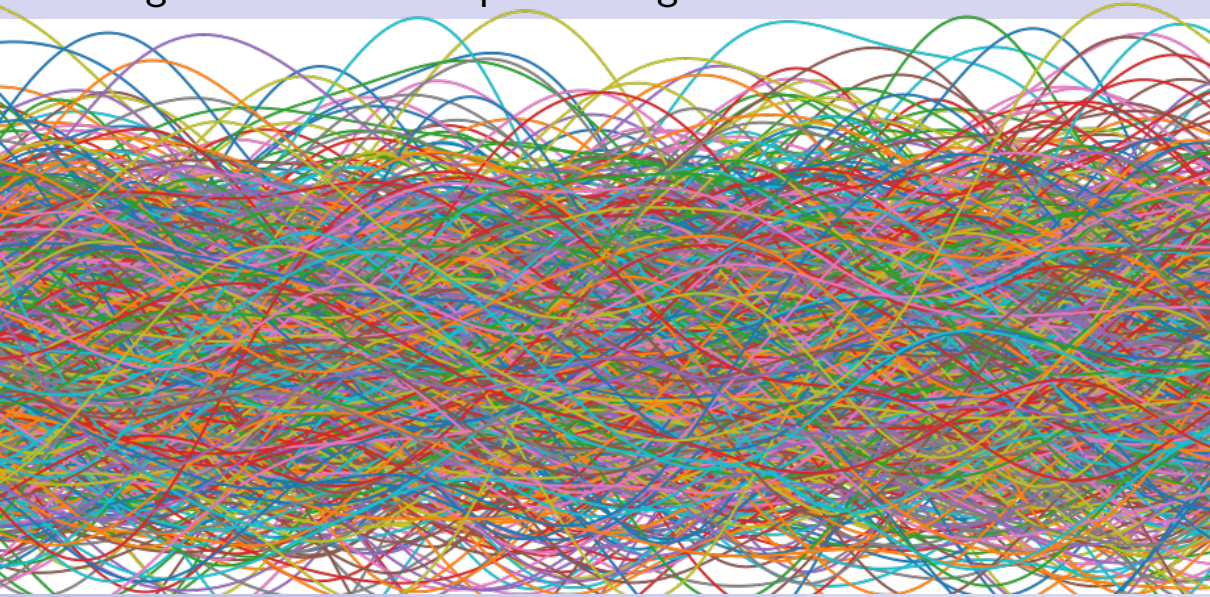
The Gaussian process regression algorithm

So how do we turn the data $(x_1, y_1), .., (x_n, y_n)$ into a reasonable stochastic model interpolating it?

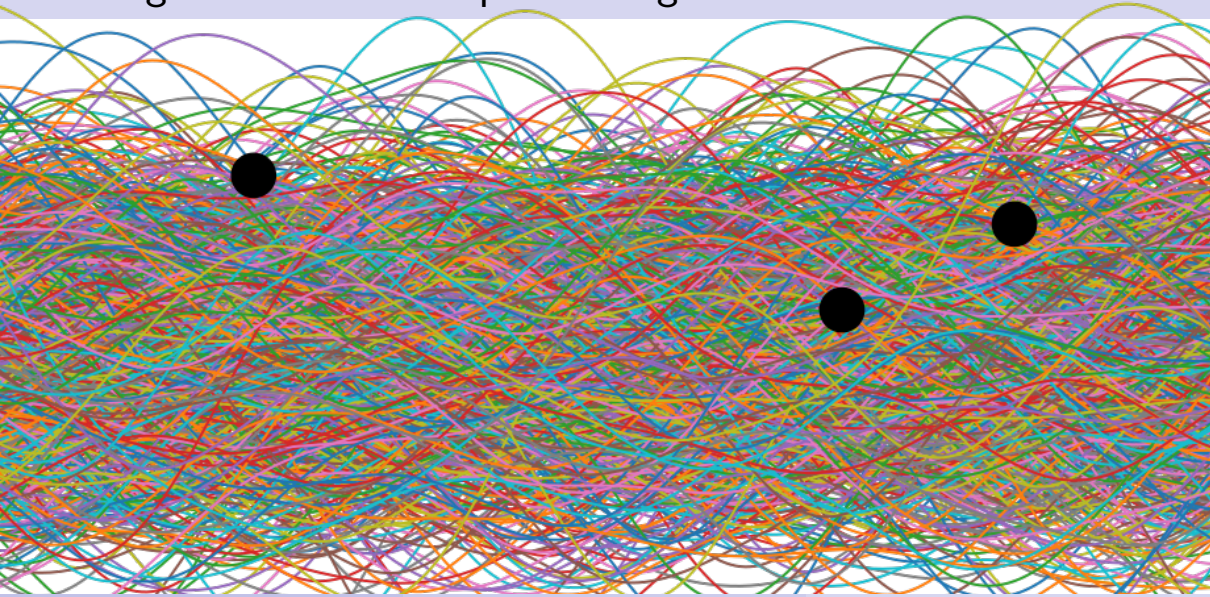
1. Come up with a parametric families $m_{\boldsymbol{\theta}}$ and $k_{\boldsymbol{\theta}}$ for prior mean and covariance functions.
2. Use maximum likelihood estimation to pick the optimal set of parameters $\boldsymbol{\theta}$ and the optimal noise value σ^2 from data $(x_1, y_1), .., (x_n, y_n)$.
3. Perform Bayesian inference with prior $GP(m_{\boldsymbol{\theta}}, k_{\boldsymbol{\theta}})$, data $(x_1, y_1), .., (x_n, y_n)$ and likelihood noise σ^2.
   As a result, obtain the posterior $\hat{m}$ and $\hat{k}$.
4. Use
   - $\mathrm{N}(\hat{m}(u), \hat{k}(u, u))$ as a stochastic prognosis at a new location u.
   - use samples of $GP(\hat{m}, \hat{k})$ as an ensemble of possible deterministic models.

Visual guide to Gaussian process regression

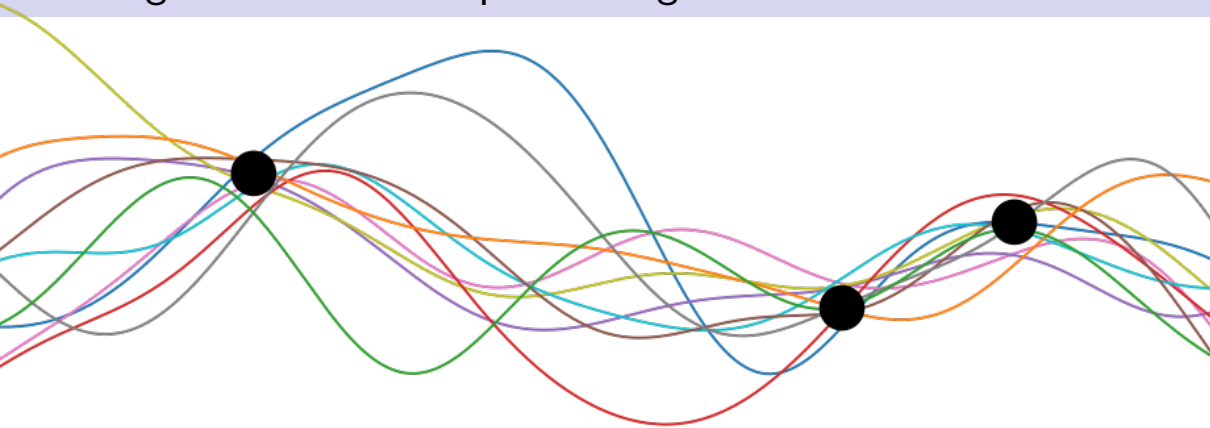
Visual guide to Gaussian process regression



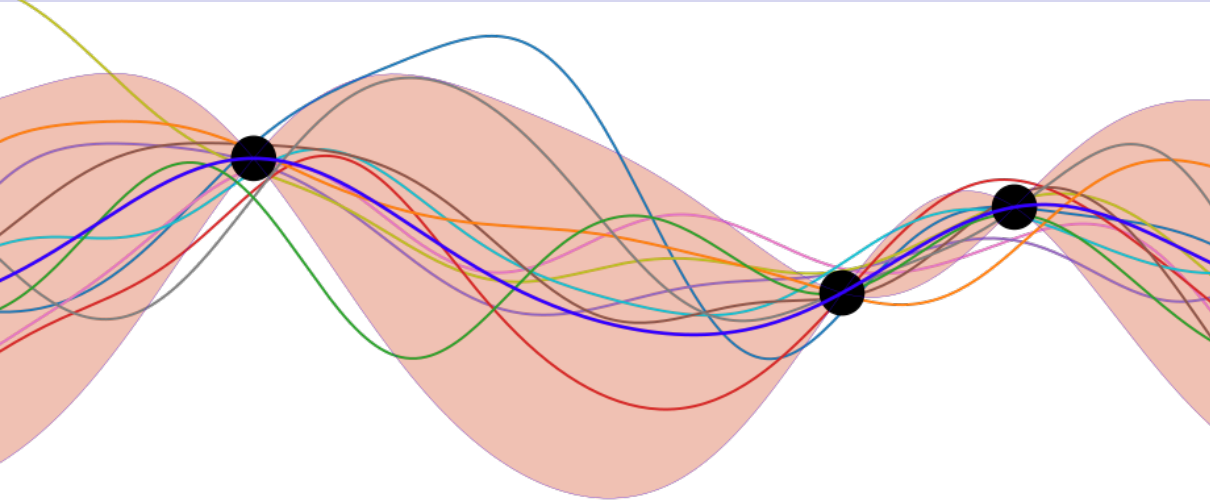
Visual guide to Gaussian process regression



Visual guide to Gaussian process regression



Visual guide to Gaussian process regression

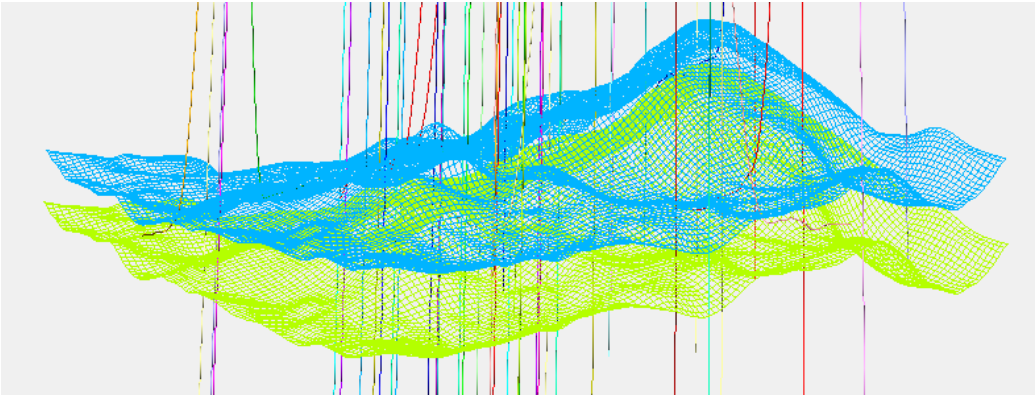


Outline

Geostatistical modeling of petroleum reservoirs

Problem: interpolate well data into the interwell space.

The data is very sparse, thus deterministic model is undesirable.

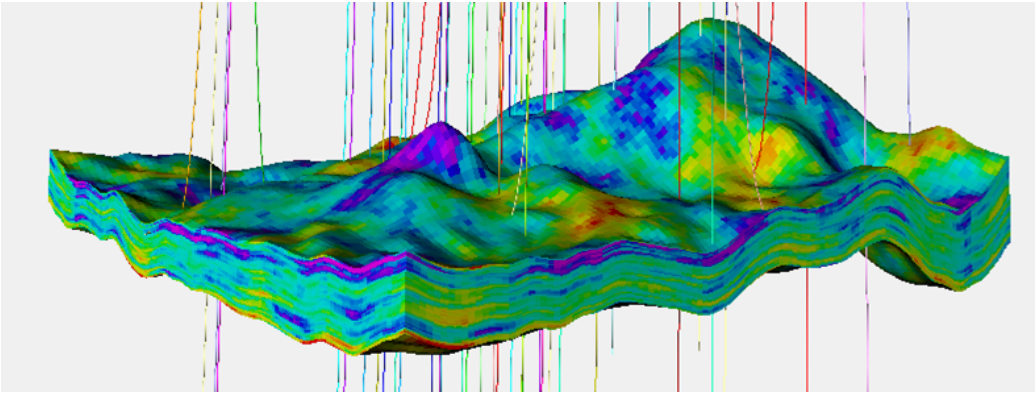


Reservoir structure, well locations.

Geostatistical modeling of petroleum reservoirs

Problem: interpolate well data into the interwell space.

The data is very sparse, thus deterministic model is undesirable.



A single sample of a Gaussian process model in the interwell space

Bayesian optimization of expensive black-box functions

Problem: minimize the target function $\phi : \mathbb{R}^d \to \mathbb{R}$.

At n'th step ϕ has already been evaluated at $x_1, .., x_n$. How do we choose x_{n+1}?

Build posterior GP f using data

$$x_1, .., x_n, \qquad \phi(x_1), .., \phi(x_n).$$

Choose

$$x_{n+1} = \arg\max_{x \in \mathbb{R}^d} \mathbb{P}(f(x) < \min_{i=1..n} \phi(x_i)). \qquad (MPI)$$

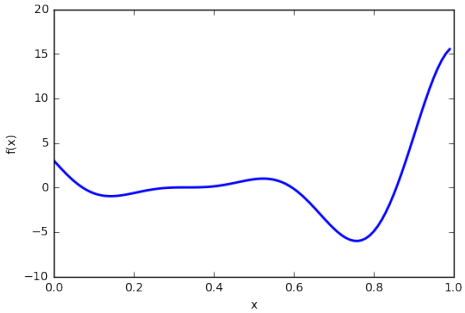
or

$$x_{n+1} = \arg\max_{x \in \mathbb{R}^d} \mathbb{E} \max(\min_{i=1..n} \phi(x_n) - f(x), 0). \qquad (EI)$$

Automatic exploration/exploitation trade-off.

Example

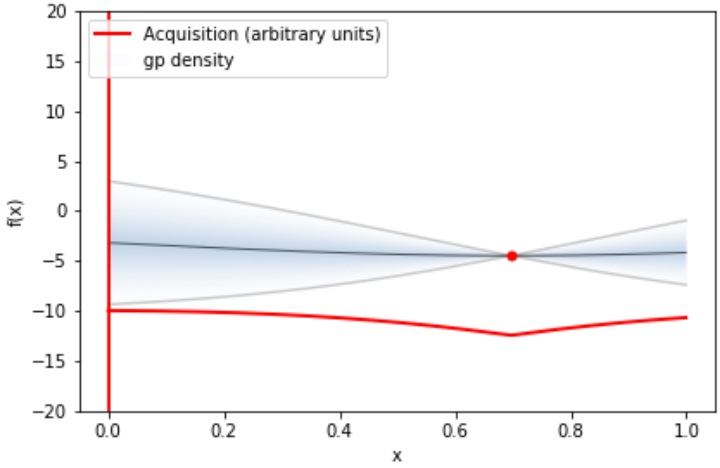
Let us minimize Forrester function $f(x) = (6x - 2)^2 \sin(12x - 4)$.



Choose some prior as $f_0 \sim \mathrm{GP}(?, ?)$.

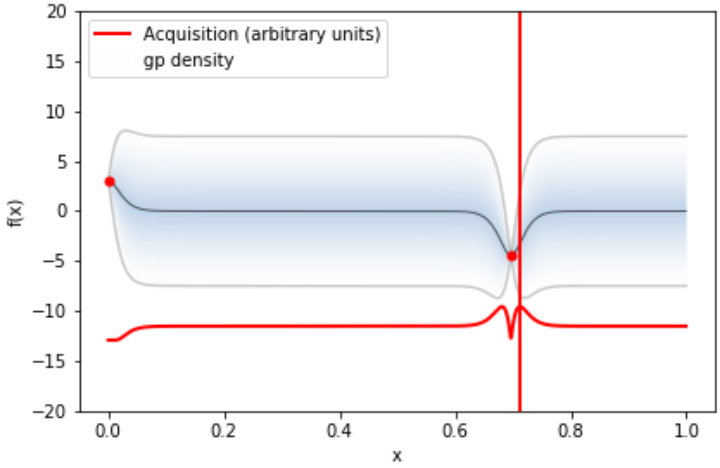
Example

Iteration 1.



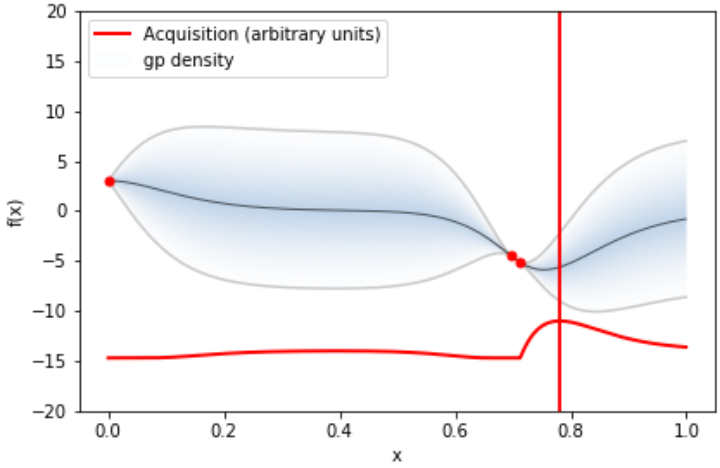
Example

Iteration 2.



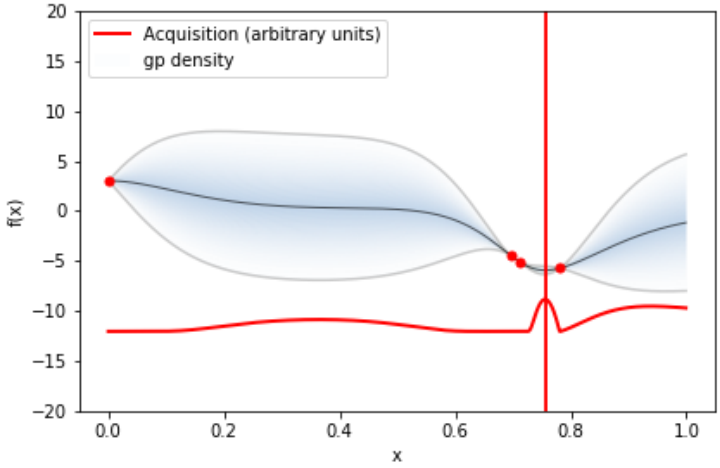
Example

Iteration 3.



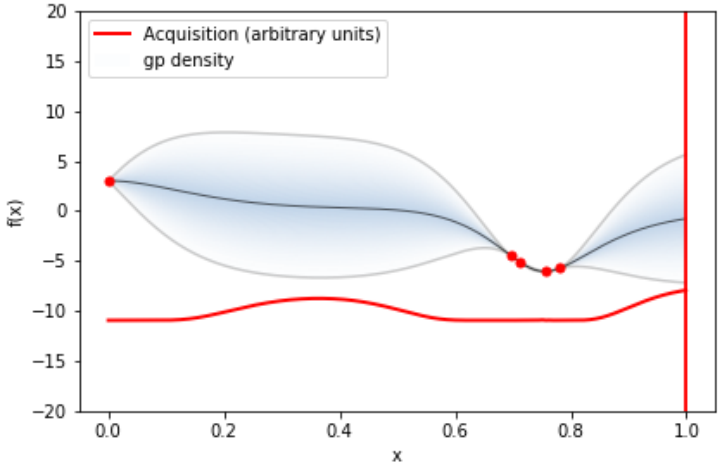
Example

Iteration 4.



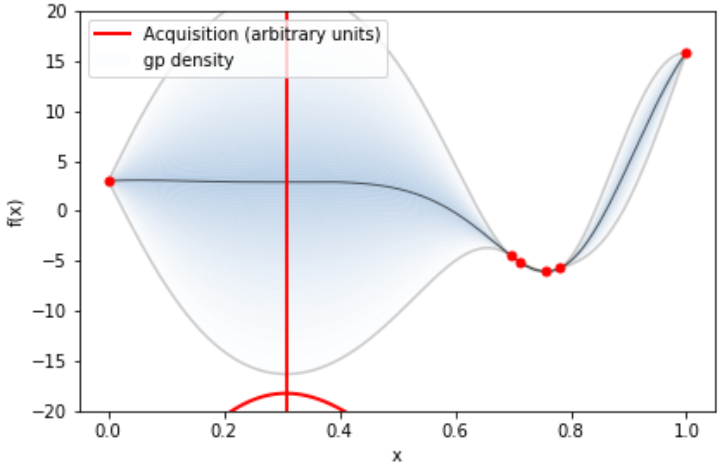
Example

Iteration 5.



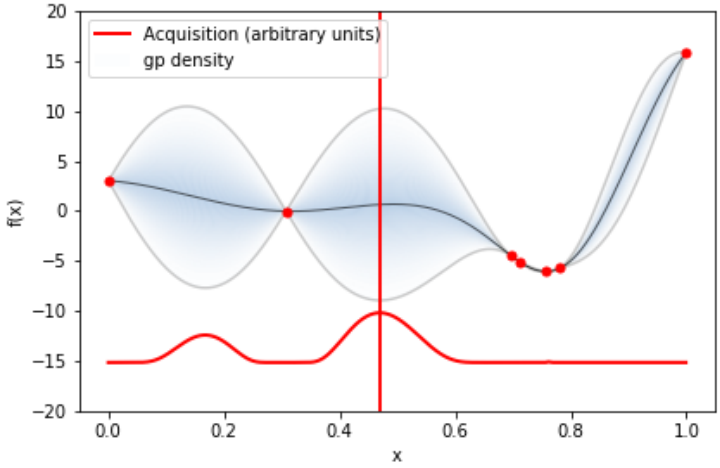
Example

Iteration 6.



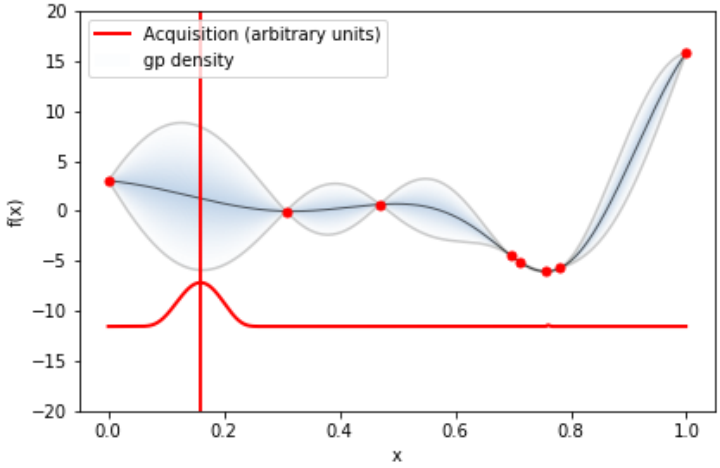
Example

Iteration 7.



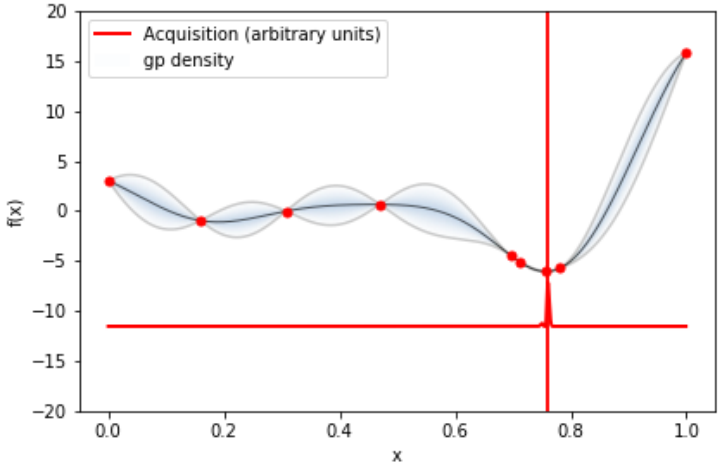
Example

Iteration 8.



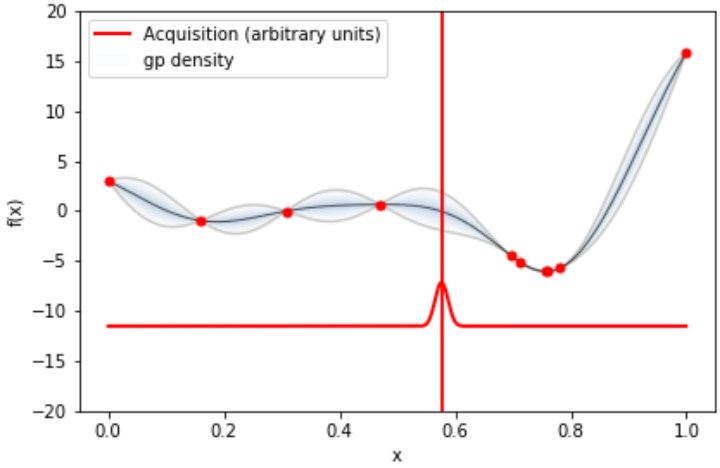
Example

Iteration 9.



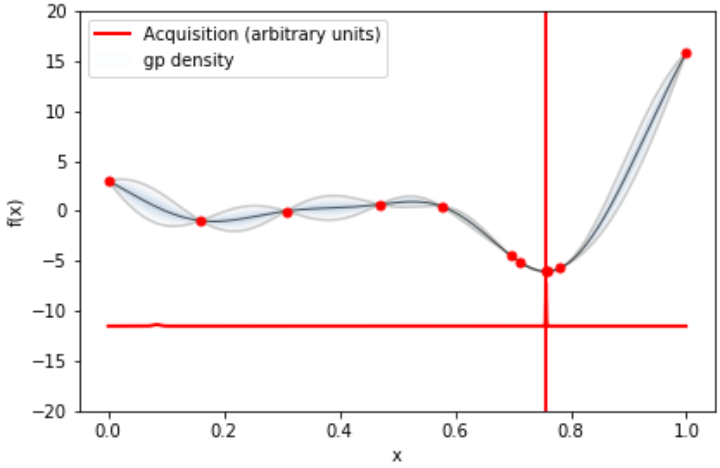
Example

Iteration 10.



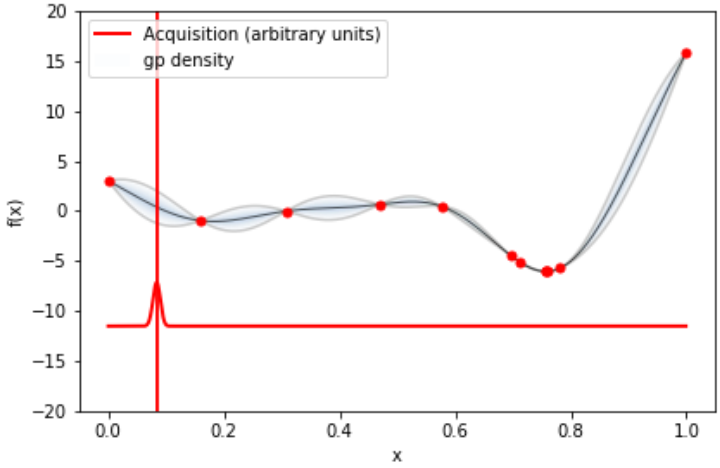
Example

Iteration 11.



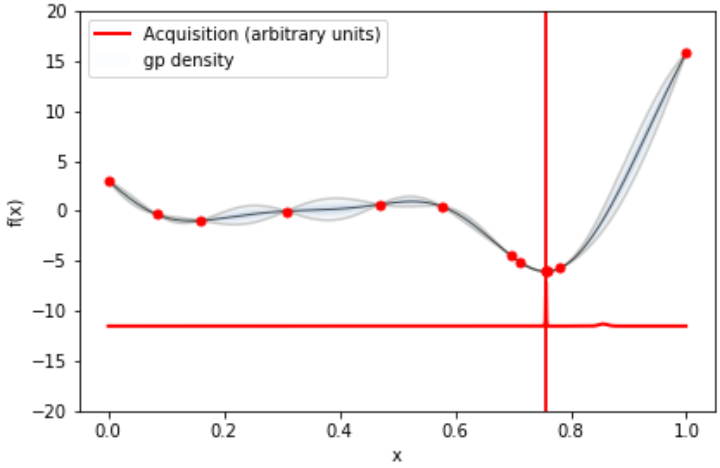
Example

Iteration 12.



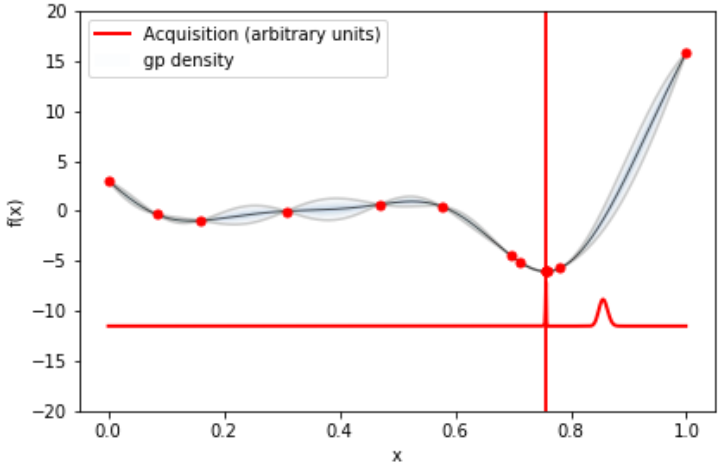
Example

Iteration 13.



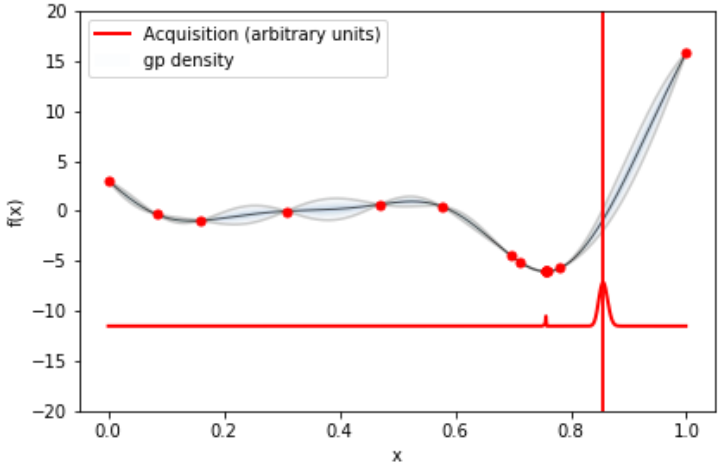
Example

Iteration 14.



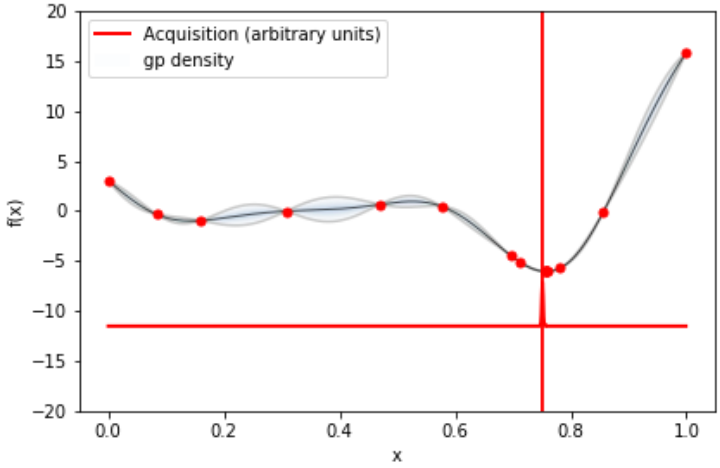
Example

Iteration 15.



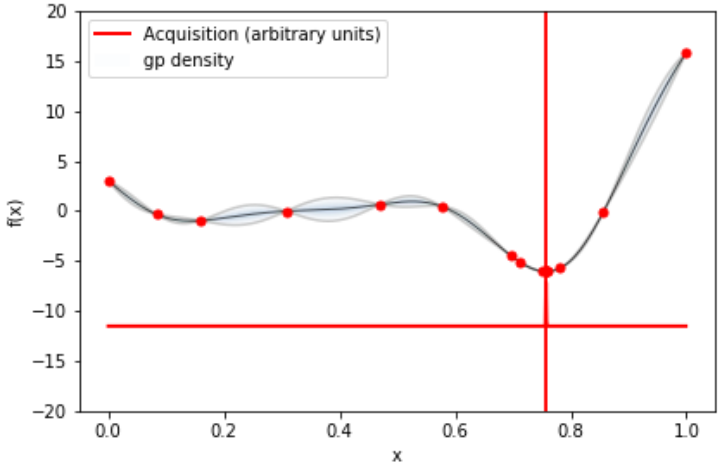
Example

Iteration 16.



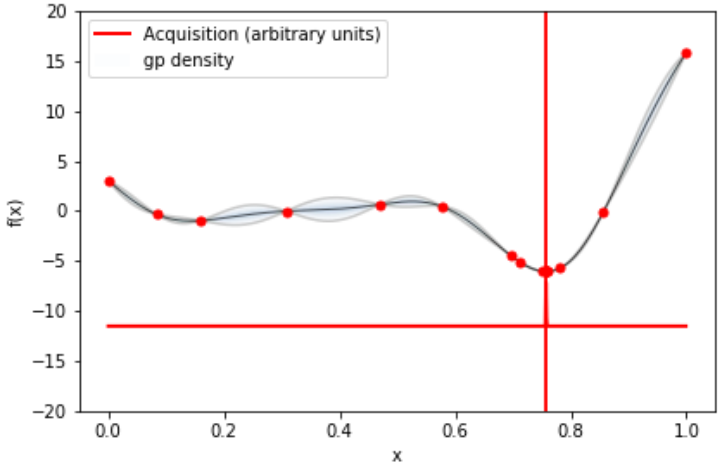
Example

Iteration 17.



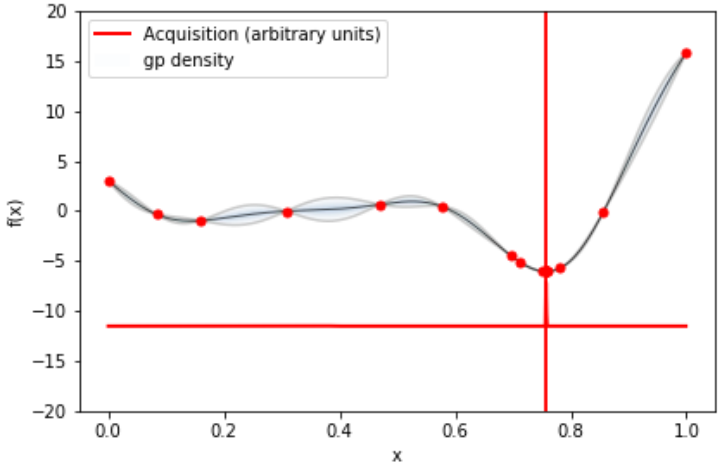
Example

Iteration 18.



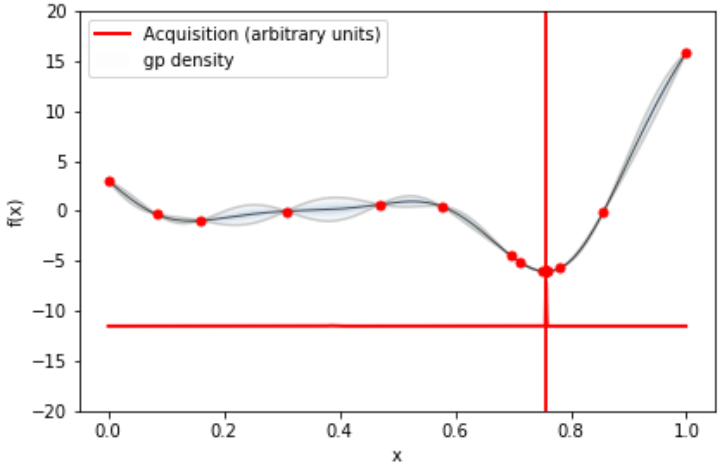
Example

Iteration 19.



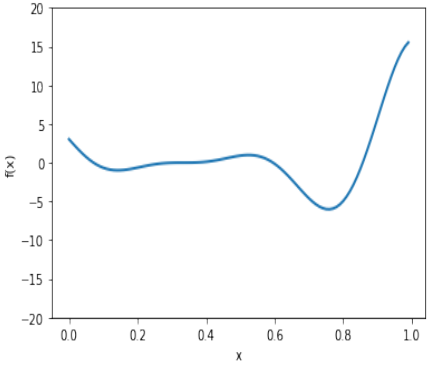
Example

Iteration 20.

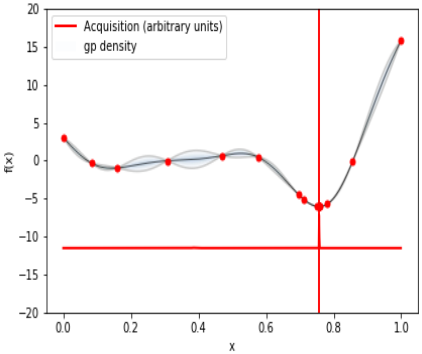


Example

Let us compare the model after 20 iterations with the target function.



(a) Target function



(b) GP model

Robotics and control

Classical control problem: physics is known, find optimal control.

Reinforcement learning problem: physics is unknown, try to learn physics from data and on the go build the optimal control.

Second approach is supposed to bring us the cheap robots, for which

- we don't indeed know the physics (it deviates too much from the "ideal"),
- learning this physics by hand is of course possible, but it increases the price.

PILCO for robotics and control

PILCO (Probabilistic Inference for Learning COntrol) — an approach that uses GPs to model the unknown physics.

PILCO: A Model-Based and Data-Efficient Approach to Policy Search

---

Marc Peter Deisenroth MARC@CS.WASHINGTON.EDU
Department of Computer Science & Engineering, University of Washington, USA

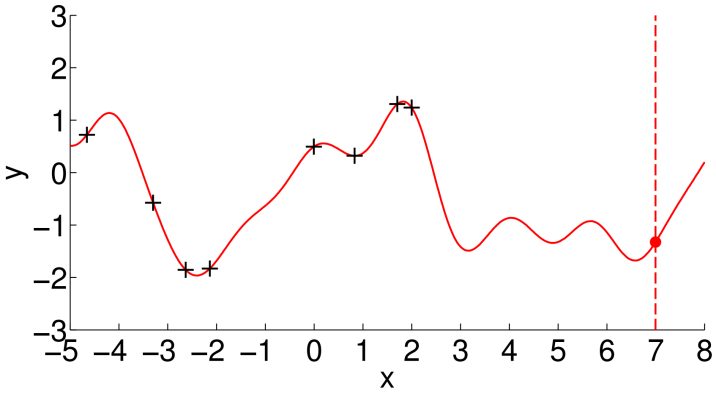
Carl Edward Rasmussen CER54@CAM.AC.UK
Department of Engineering, University of Cambridge, UK

The model can be described by $x_{t+1} = f(x_t, u_t) + w$, where

- x_t — trajectory,
- u_t — control,
- f models physics,
- $w \sim N(0, \sigma^2)$ — random noise.

PILCO for robotics and control

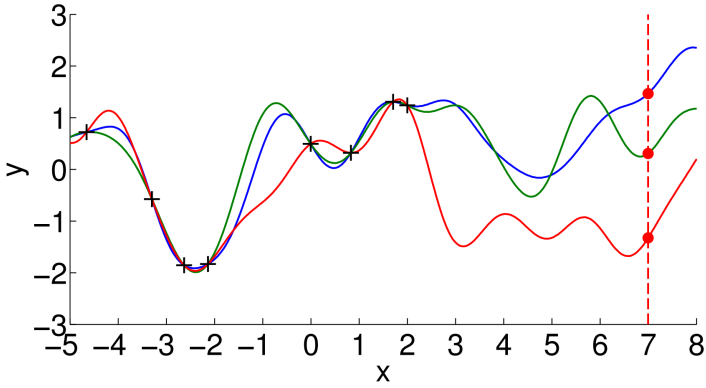
Imagine that f is modeled deterministically.



Consider a prognosis at $x = 7$.

PILCO for robotics and control

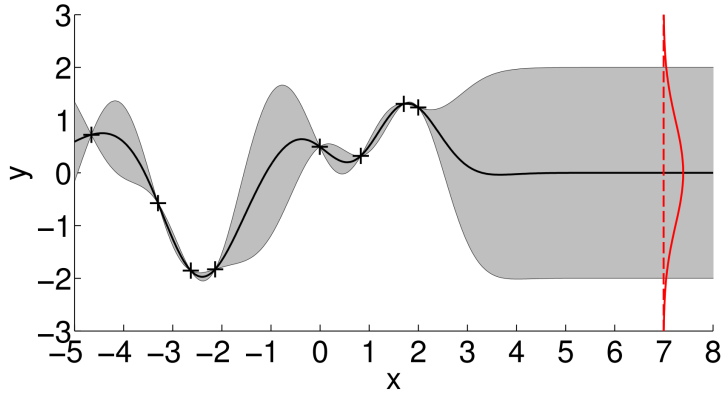
Imagine that f is modeled deterministically.



There exists a number of plausible models and thus a number of different predictions.

PILCO for robotics and control

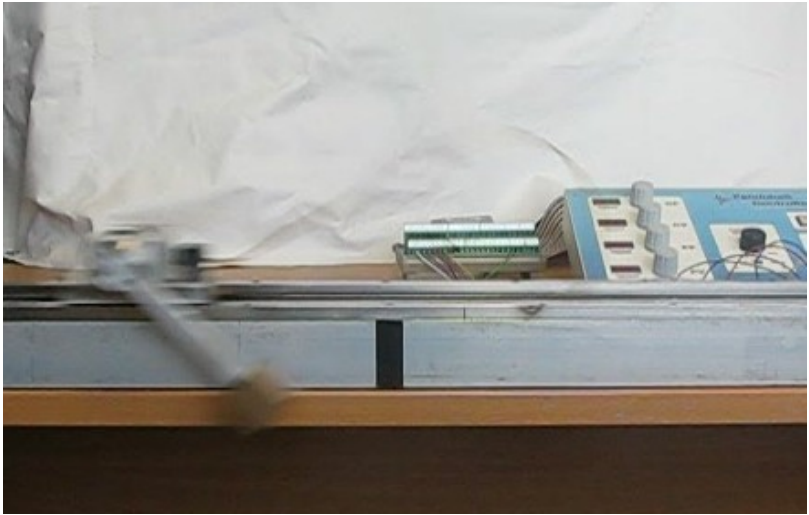
What if we model f as a GP?



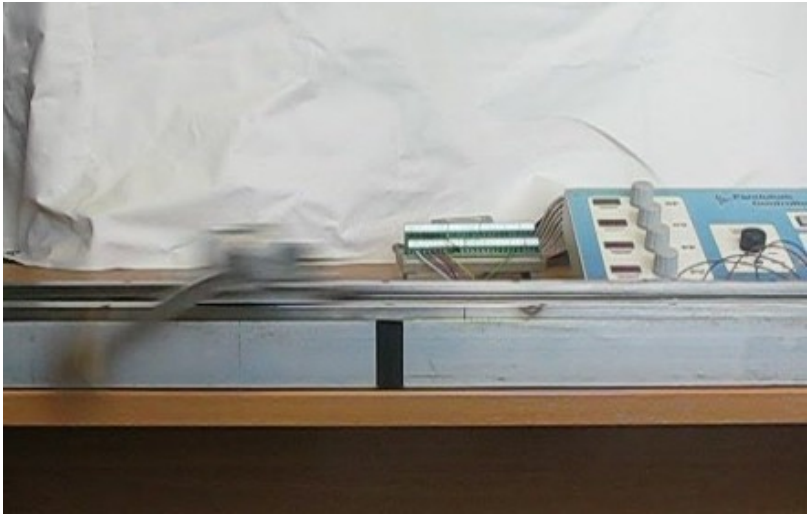
If we use GPs, we are able to use an infinite number of plausible models all at once.

Example: learning to control a pendulum

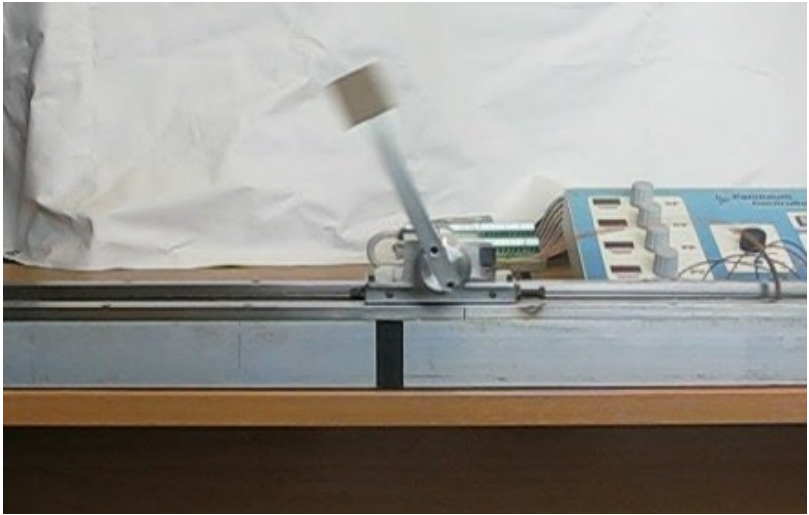
Example: learning to control a pendulum



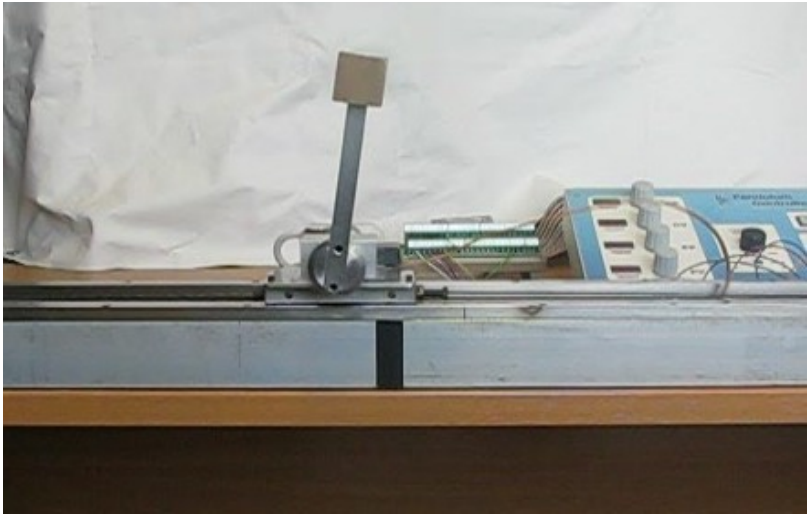
Example: learning to control a pendulum



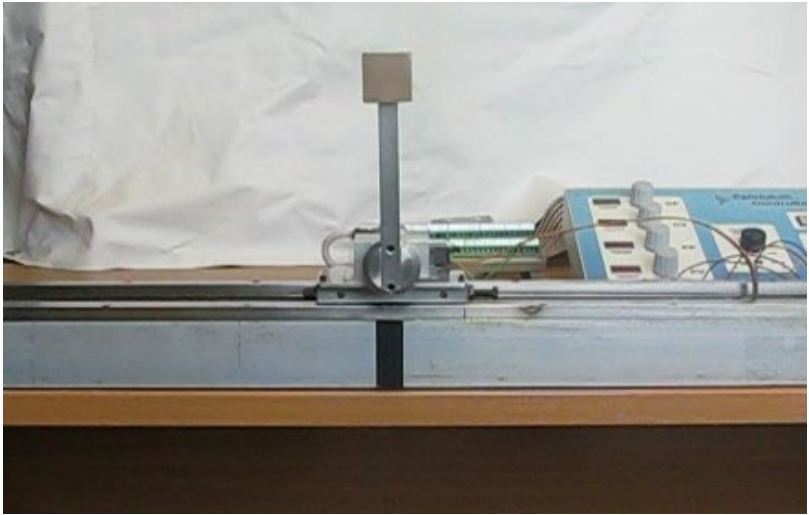
Example: learning to control a pendulum



Example: learning to control a pendulum



Example: learning to control a pendulum

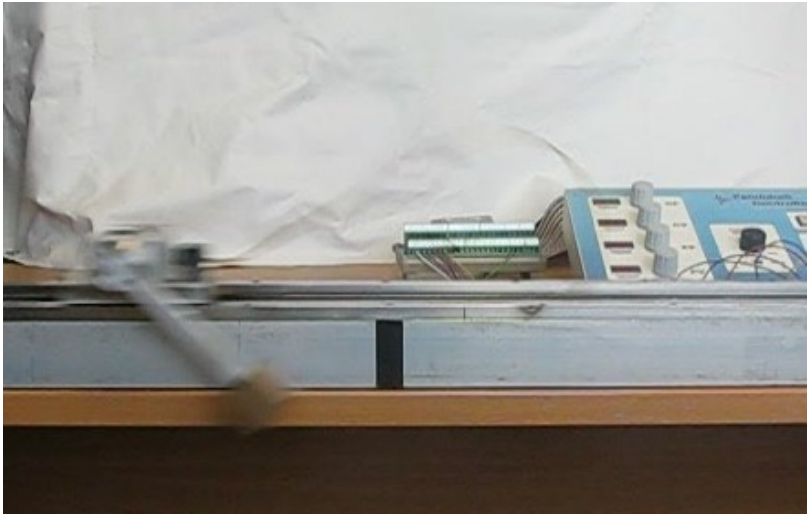


Example: learning to control a pendulum

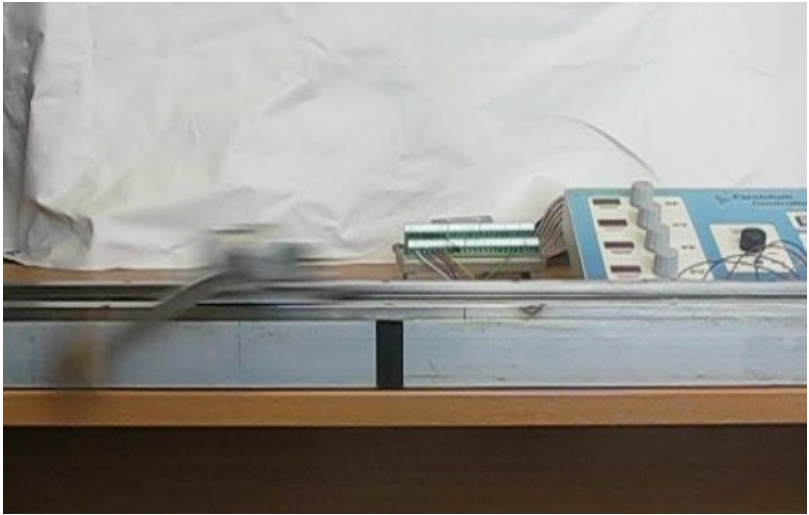
Once more...

Example: learning to control a pendulum

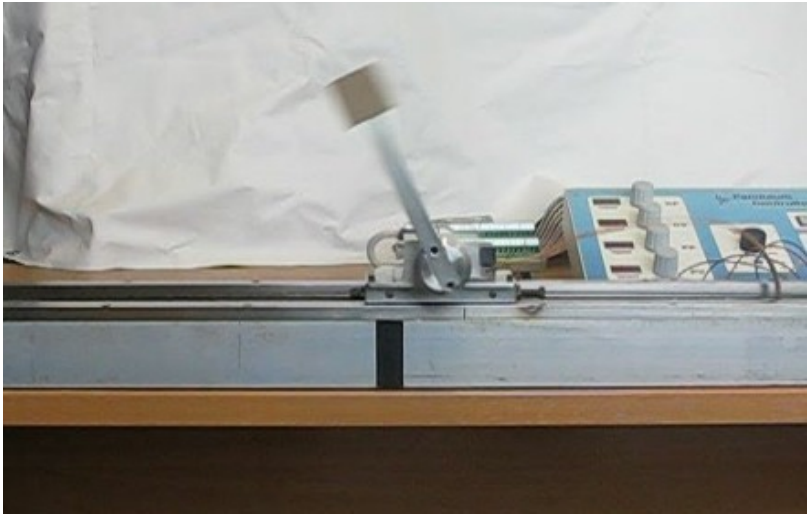
Example: learning to control a pendulum



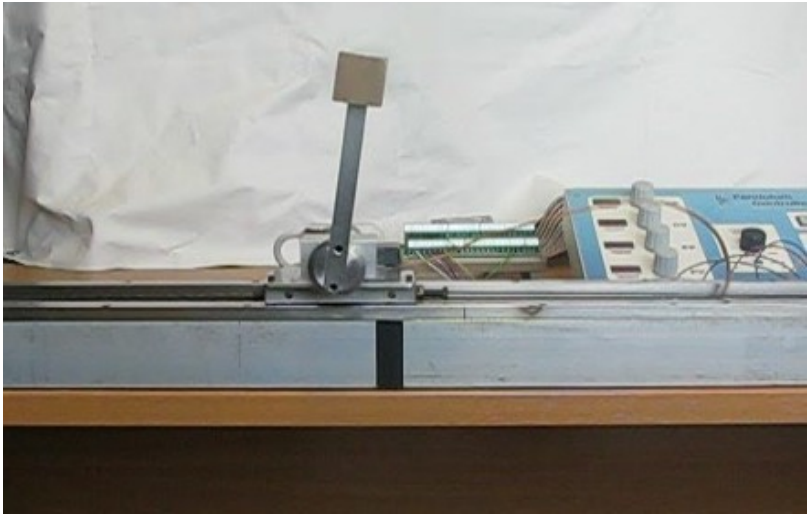
Example: learning to control a pendulum



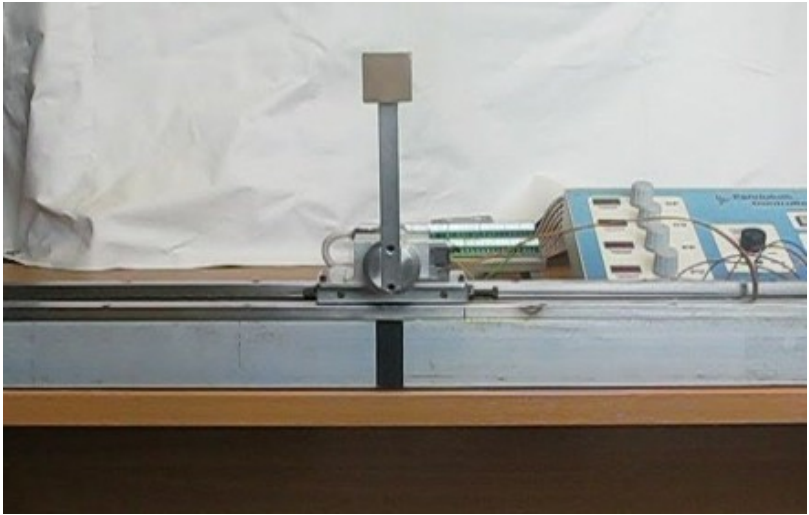
Example: learning to control a pendulum



Example: learning to control a pendulum



Example: learning to control a pendulum



Thank you for your attention!

viacheslav.borovitskiy@gmail.com

St Petersburg University

Mathematics & Computer Science department

Some figures were taken from: `http://inverseprobability.com/talks/`.

Gaussian random fields in machine learning

Viacheslav Borovitskiy

St. Petersburg State University
St. Petersburg Department of Steklov Mathematical Institute

Winter School in Mathematics and Theoretical Computer Science
January 29 – February 3, 2021

Part II

Predicting with Gaussian random fields and generating their sample paths

Today's talk structure

5. Conditional distribution of a Gaussian vector
6. Example application: Bayesian linear regression
7. Conditional Gaussian process
8. Algorithms for predicting and sampling

Outline

Foreword

In the previous talk we discussed Bayesian inference for Gaussian processes.

For discrete random variables Θ, $\mathcal{D}$ Bayes theorem states that

$$\underbrace{\mathbb{P}(\Theta = \theta \mid \mathcal{D} = d)}_{\text{Posterior}} = \frac{\overbrace{\mathbb{P}(\mathcal{D} = d \mid \Theta = \theta)}^{\text{Likelihood}} \overbrace{\mathbb{P}(\Theta = \theta)}^{\text{Prior}}}{\underbrace{\mathbb{P}(\mathcal{D} = d)}_{\text{Normalizing constant}}}.$$

For absolutely continuous random variables θ, d Bayes theorem states that

$$p(\theta \mid d) = \frac{p(d \mid \theta)p(\theta)}{p(d)} \quad \text{with} \quad p(\theta \mid d) = \frac{p(\theta, d)}{p(d)} \quad \text{and} \quad p(d \mid \theta) = \frac{p(\theta, d)}{p(\theta)},$$

where $p(\theta, d)$ is the joint density of θ and d.

Bayes theorem is about conditional distributions.
Let's find one for Gaussian random vectors first!

The problem

Consider a random vector divided in two parts

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \sim \mathrm{N}(m, \Sigma) = \mathrm{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \right).$$

Let $P = \Sigma^{-1}$ denote the precision matrix with blocks $P = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}$.

Then x has density

$$p(x_1, x_2) = |2\pi\Sigma|^{-1/2} \exp\left(-\frac{1}{2} \begin{pmatrix} x_1^\top - m_1^\top, x_2^\top - m_2^\top \end{pmatrix} \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} \begin{pmatrix} x_1 - m_1 \\ x_2 - m_2 \end{pmatrix} \right).$$

What is the distribution $p(x_1 \mid x_2)$ of x_1 given the value of x_2?

Some linear algebra

Completing the square for numbers:

$$ax^2 + bx + c = a(x - h)^2 + k \qquad \text{with} \qquad h = -\frac{b}{2a} \qquad \text{and} \qquad k = c - \frac{b^2}{4a}.$$

Completing the square for matrices ($A = A^\top$):

$$x^\top Ax + x^\top b + c = (x-h)^\top A(x-h) + k \quad \text{with} \quad h = -\frac{1}{2}A^{-1}b \quad \text{and} \quad k = c - \frac{1}{4}b^\top A^{-1}b.$$

Block matrix inversion:

$$M^{-1} = \begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} (M/D)^{-1} & -(M/D)^{-1}BD^{-1} \\ -D^{-1}C(M/D)^{-1} & D + D^{-1}C(M/D)^{-1}BD^{-1} \end{pmatrix},$$

where $M/D = A - BD^{-1}C$ is called the Schur complement.

The computation (part 1)

We have joint density

$$p(x_1, x_2) = |2\pi\Sigma|^{-1/2} \exp\left(-\frac{1}{2}\begin{pmatrix} x_1^\top - m_1^\top, x_2^\top - m_2^\top \end{pmatrix} \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} \begin{pmatrix} x_1 - m_1 \\ x_2 - m_2 \end{pmatrix}\right).$$

Then $p(x_1 \mid x_2) = \frac{p(x_2|x_1)p(x_1)}{p(x_2)} = \frac{p(x_1,x_2)}{p(x_2)}$, where $p(x_j) = \int p(x_1, x_2)\,\mathrm{d}x_j$, hence

$$\begin{aligned} p(x_1 \mid x_2) &= C(x_2) \exp\Big((x_1^\top - m_1^\top)P_{11}(x_1 - m_1) + 2(x_1^\top - m_1^\top)P_{12}(x_2 - m_2)\Big) \\ &= \hat{C}(x_2) \exp\Big(x_1^\top P_{11} x_1 + 2x_1^\top (P_{12}(x_2 - m_2) - P_{11} m_1)\Big) \end{aligned}$$

$$x^\top A x + x^\top b + c = (x-h)^\top A (x-h) + k \quad \text{with} \quad h = -\frac{1}{2}A^{-1}b \quad \text{and} \quad k = c - \frac{1}{4}b^\top A^{-1} b$$

$$= \tilde{C}(x_2) \exp\Big((x_1 - \hat{m}_1)^\top P_{11} (x_1 - \hat{m}_1)\Big)$$

where $\hat{m}_1 = -P_{11}^{-1}(P_{12}(x_2 - m_2) - P_{11} m_1)$.

The computation (part 2)

We have

$$p(x_1 \mid x_2) = \tilde{C}(x_2) \exp\Big((x_1 - \hat{m}_1)^\top P_{11}(x_1 - \hat{m}_1)\Big)$$

where $\hat{m}_1 = -P_{11}^{-1}(P_{12}(x_2 - m_2) - P_{11}m_1)$.

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} (M/D)^{-1} & -(M/D)^{-1}BD^{-1} \\ -D^{-1}C(M/D)^{-1} & D + D^{-1}C(M/D)^{-1}BD^{-1} \end{pmatrix}, M/D = A - BD^{-1}C.$$

By these formulas, $P_{11}^{-1}P_{12} = -\Sigma_{12}\Sigma_{22}^{-1}$, hence $\hat{m}_1 = \Sigma_{12}\Sigma_{22}^{-1}(x_2 - m_2) + m_1$.
Besides that, $P_{11} = (\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})^{-1}$.

The result

Since

$$p(x_1 \mid x_2) = \tilde{C}(x_2) \exp\Big((x_1 - \hat{m}_1)^\top \hat{\Sigma}^{-1} (x_1 - \hat{m}_1)\Big),$$

with

- $\hat{m}_1 = \Sigma_{12}\Sigma_{22}^{-1}(x_2 - m_2) + m_1$,
- $\hat{\Sigma} = P_{11}^{-1} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$

we have

$$x_1 \mid x_2 \sim \mathrm{N}\Big(m_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - m_2), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}\Big).$$

Note that

- the conditional distribution is again Gaussian,
- its parameters are computable through linear algebra,
- the "variance" is now lower (condtioning reduces uncertainty).

Bayesian perspective

Taking the more Bayesian perspective, we could have started, intead of the joint, with

- prior $x_1 \sim \mathrm{N}(m_1, \Sigma_{11})$
- and likelihood $p(x_2 \mid x_1) = \mathrm{N}(\Sigma_{21} m_2 + \Sigma_{11}^{-1}(x_1 - m_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12})$.

They determine the joint, so it is just another perspective on the same problem.

An important example

A priori think that $x_1 \sim \mathrm{N}(0, C)$.
Observe $x_2 = x_1 + \varepsilon$ with $\varepsilon \sim \mathrm{N}(0, \sigma_n^2 I)$ that is independent of x_1.
Because of that $p(x_2 \mid x_1) = \mathrm{N}(x_1, \sigma_n^2 I)$, hence

$$p(x_1 \mid x_2) = \mathrm{N}(C\Big(C + \sigma_n^2 I\Big)^{-1} x_2, C - C\Big(C + \sigma_n^2 I\Big)^{-1} C)$$

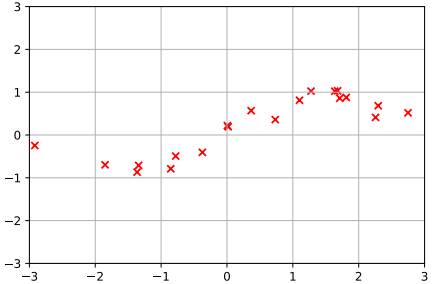
Note how formally $p(x_1 \mid x_2) = \mathrm{N}(x_2, 0)$ when $\sigma^2 = 0$ and we observe x_1 itself.

Outline

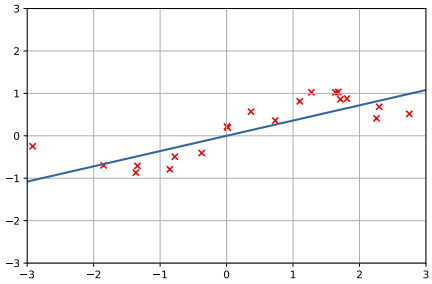
Linear regression

Given data $t_1, y_1; \ldots; t_n, y_n$ with $t_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$.
E.g. y_i — apartment price, t_i — apartment parameters (size, floor height etc.).

The standard linear regression problem is to find a linear model
$f(t) = w^\top t$ for some vector of weights $w \in \mathbb{R}^d$, such that $\sum_{i=1}^{n} (y_i - f(t_i))^2$ is minimal.



(a) Data $t_1, y_1; \ldots; t_n, y_n$



(b) Data & linear model $f(t) = w^\top t$

Bayesian linear regression

Take the model $f(t) = w^\top t$ and assume a priori $w \sim \mathrm{N}(0, I)$.
Put $y(t) = f(t) + \sigma_n^2 \varepsilon(t)$, where $\varepsilon(t) \sim \mathrm{N}(0, 1)$ is i.i.d. normal noise.

Every $f(t)$ and $y(t)$ is a Gaussian random variable, moreover

- $\mathrm{Cov}(f(t), f(t')) = t^\top t'$ and $\mathrm{Cov}(y(t), y(t')) = t^\top t' + \sigma_n^2 \mathbb{1}_{t=t'}$,
- $\mathrm{Cov}(f(t), y(t')) = \mathrm{Cov}(y(t), f(t')) = t^\top t'$.

$$\begin{pmatrix} f(t) \\ y(t_1) \\ \vdots \\ y(t_n) \end{pmatrix} \sim \mathrm{N}\left(0, \left(\begin{array}{c|ccc} t^\top t & t^\top t_1 & \dots & t^\top t_n \\ \hline t_1^\top t & t_1^\top t_1 + \sigma_n^2 & \dots & t_1^\top t_n \\ \vdots & \vdots & \ddots & \vdots \\ t_n^\top t & t_n^\top t_1 & \dots & t_n^\top t_n + \sigma_n^2 \end{array}\right)\right) = \mathrm{N}\left(0, \left(\begin{array}{c|c} \Sigma_{11} & \Sigma_{12} \\ \hline \Sigma_{21} & \Sigma_{22} \end{array}\right)\right).$$

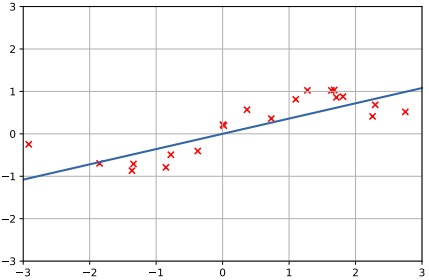
And analogously $\big(w^\top, y(t_1), \dots, y(t_n)\big)^\top \sim \mathrm{N}\left(0, \left(\begin{array}{c|c} C_{11} & C_{12} \\ \hline C_{21} & C_{22} \end{array}\right)\right)$.

Bayesian linear regression

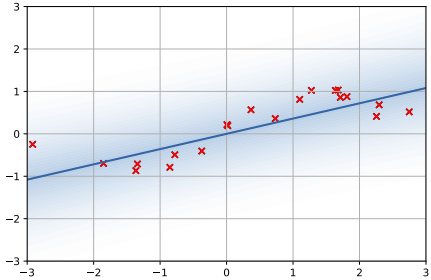
Then, denoting $\boldsymbol{y} = (y_1, \ldots, y_n)^\top$, we compute

$$p(f(t) \mid y(t_1) = y_1; \ldots; y(t_n) = y_n) = \mathrm{N}\Big(\Sigma_{12}\Sigma_{22}^{-1}\boldsymbol{y}, \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}\Big)$$

$$p(w \mid y(t_1) = y_1; \ldots; y(t_n) = y_n) = \mathrm{N}\Big(C_{12}C_{22}^{-1}\boldsymbol{y}, C_{11} - C_{12}C_{22}^{-1}C_{21}\Big)$$



(a) Linear regression



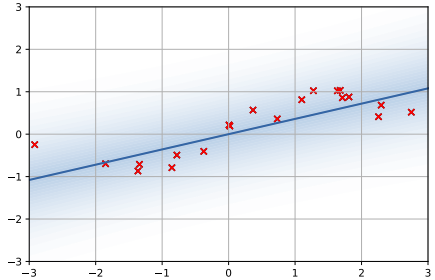
(b) Bayesian linear regression

Picking the prior and likelihood parameters

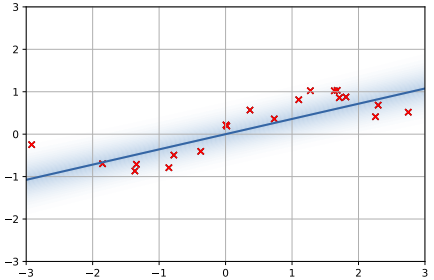
Consider the density $p(y_1, \ldots, y_n) = \mathrm{N}\left(0, \begin{pmatrix} t_1^\top t_1 + \sigma_n^2 & \ldots & t_1^\top t_n \\ \vdots & \ddots & \vdots \\ t_n^\top t_1 & \ldots & t_n^\top t_n + \sigma_n^2 \end{pmatrix}\right)$

as a function of σ_n^2 and maximize it with respect to σ_n^2.

In this context, $p(y_1, \ldots, y_n)$ is called the marginal likelihood of the data.



(a) Default likelihood noise



(b) Optimized likelihood noise

Outline

Stochastic processes, their sample paths and distributions

A stochastic process is a family $X = \{X_t\}_{t \in T}$ of random variables.

- T is some set. Depending on T different terms may be used:
  - random field, when $T \subseteq \mathbb{R}^d$,
  - random sequence, when $T \subseteq \mathbb{Z}$.
- X_t at different values of t may be dependent (and usually are).
- Alternatively, it can be defined as a random variable on some space of functions.
- Should be thought of as random function of $t \in T$.

X can be considered as a real valued function $X(\omega, t)$ of $\omega \in \Omega, t \in T$.
Then $X(\omega, \cdot)$ is called its trajectory or sample path.

The system of distributions $P^X_{t_1,\ldots,t_n}(A) = \mathbb{P}((X_{t_1}, \ldots, X_{t_n}) \in A)$ for all $n \in \mathbb{N}$, $t_1, \ldots, t_n \in T$ and $A \in \mathcal{B}_n$ is called the distribution of the random process X.

Gaussian processes

X is a Gaussian process if all its $P^X_{t_1,\dots,t_n}$ are multivariate Gaussian.
The distribution of a Gaussian process is determined by a pair of functions:

- $m(\cdot) : T \to \mathbb{R}$ — the mean function,
- $k(\cdot,\cdot) : T \times T \to \mathbb{R}$ — the covariance function (kernel),

such that

$$P^X_{t_1,\dots,t_n} = \mathrm{N}\left(\begin{pmatrix} m(t_1) \\ \vdots \\ m(t_n) \end{pmatrix} \begin{pmatrix} k(t_1,t_1) & \dots & k(t_1,t_n) \\ \vdots & \ddots & \vdots \\ k(t_n,t_1) & \dots & k(t_n,t_n) \end{pmatrix}\right)$$

A covariance matrix C should be positive semidefinite: satisfy $C^\top = C$ and $x^\top C x \geq 0$.
A valid covaraince function k should be positive semidefinitite function. That is, for any n and $t_1, \dots, t_n$, the covariance matrix as above should be positive semidefinite.

For every m and positive semidefinite k there exists a Gaussian process having them as its mean and covaraiance functions

Conditional process

Consider an l-dimensional random vector Y and some value $\boldsymbol{y} \in \mathbb{R}^l$.

Define the conditional distribution of a process X given $Y = \boldsymbol{y}$ to be the family

$$P^{X|Y=\boldsymbol{y}}_{t_1,\dots,t_n} = \mathbb{P}((X_{t_1}, \dots, X_{t_n}) \in A \mid Y = \boldsymbol{y})$$

of conditional distributions.

Consider a Gaussian process X and $Y = (X(t_1), \dots, X(t_n)) + \sigma_n^2 \boldsymbol{\varepsilon}$ with $\boldsymbol{\varepsilon} \sim \mathrm{N}(0, I)$.

Fix some $l \in \mathbb{N}$, $\tilde{t}_1, \dots, \tilde{t}_l \in T$ and denote $X(\tilde{\boldsymbol{t}}) = (X(\tilde{t}_1), \dots, X(\tilde{t}_l))$.
Denote also $m(\tilde{\boldsymbol{t}}) = \Big(m(\tilde{t}_1), \dots, m(\tilde{t}_l)\Big)$ and $m(\boldsymbol{t}) = (m(t_1), \dots, m(t_n))$.

Then, by the conditioning formula for jointly Gaussian vectors, we have

$$p(X(\tilde{\boldsymbol{t}}) \mid Y = \boldsymbol{y}) = \mathrm{N}(m(\tilde{\boldsymbol{t}}) + \mathrm{Cov}(X(\tilde{\boldsymbol{t}}), Y)\,\mathrm{Cov}(Y, Y)^{-1}(\boldsymbol{y} - m(\boldsymbol{t})),$$
$$\mathrm{Cov}(X(\tilde{\boldsymbol{t}}), X(\tilde{\boldsymbol{t}})) - \mathrm{Cov}(X(\tilde{\boldsymbol{t}}), Y)\,\mathrm{Cov}(Y, Y)^{-1}\,\mathrm{Cov}(Y, X(\tilde{\boldsymbol{t}})))$$

Conditional Gaussian process

Hence $X \mid Y = \boldsymbol{y}$ is again Gaussian with mean $\hat{m}(\cdot)$ and covariance $\hat{k}(\cdot, \cdot)$ given by

$$\hat{m}(t) = m(t) + \operatorname{Cov}(X(t), Y) \operatorname{Cov}(Y, Y)^{-1} (\boldsymbol{y} - m(\boldsymbol{t}))$$
$$\hat{k}(t, t') = \operatorname{Cov}(X(t), X(t')) - \operatorname{Cov}(X(t), Y) \operatorname{Cov}(Y, Y)^{-1} \operatorname{Cov}(Y, X(t'))$$

Note

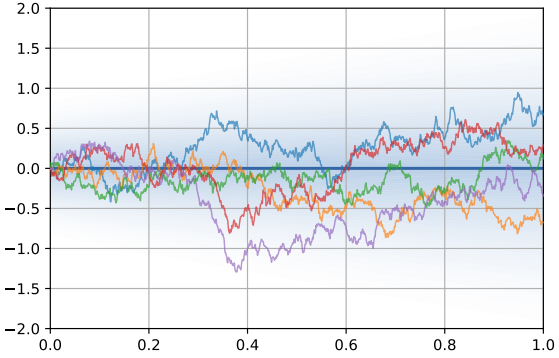
- the right hand sides are determined by $m(\cdot)$, $k(\cdot, \cdot)$ and by σ_n^2 and $\boldsymbol{y}$,
- the computation of $\hat{m}(t)$ and $\hat{k}(t, t')$ can be done by a computer,
- with $X(\boldsymbol{t}) = (X(t_1), \ldots, X(t_n))$, we have $\operatorname{Cov}(Y, Y) = \operatorname{Cov}(X(\boldsymbol{t}), X(\boldsymbol{t})) + \sigma_n^2 I$,
- the variance decreases: $\hat{k}(t, t) \leq k(t, t)$ — we gained some information.

This is exactly the "Bayesian inference for GPs" we have seen earlier!

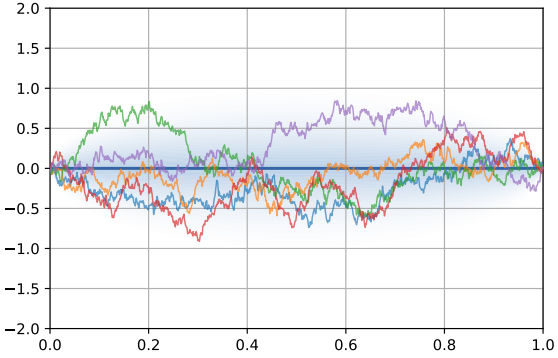
Conditional Gaussian process: an example

Brownian motion — GP with $T = [0, \infty)$, $m(t) = 0$ and $k(t, t') = \min(t, t')$.
Denote it by X.

Brownian bridge — $X \mid X(1) = 0$.



(a) Brownian motion



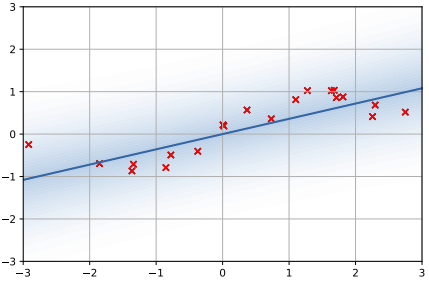
(b) Brownian bridge

Conditional Gaussian processes for the toy dataset

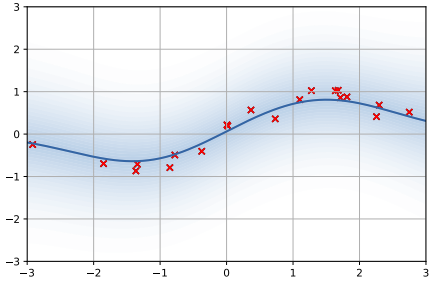
Now consider a GP X with $T = (-\infty, \infty)$, $m(t) = 0$ and $k(t, t') = \sigma^2 \exp\left(-|t-t'|^2/2l^2\right)$.
Put $Y(t_i) = X(t_i) + \sigma_n^2 \varepsilon_i$, where $\varepsilon_i \sim \mathrm{N}(0, 1)$ is i.i.d. noise.
σ^2, l and σ_n^2 are some parameters. Consider, for now, $\sigma^2 = 1$, $l = 1$, $\sigma_n^2 = 1$.

Let us solve the toy problem using the conditional GP model:

$$f(\cdot) = X \mid Y(t_1) = y_1, \ldots, Y(t_n) = y_n$$



(a) Bayesian linear regression



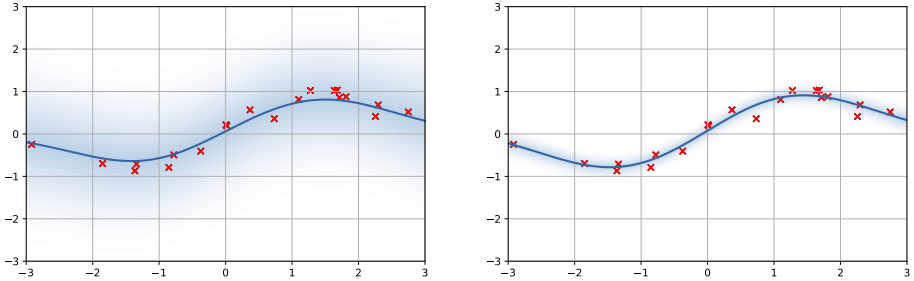
(b) Gaussian process regression

Conditional Gaussian processes: hyperparameter optimization

X from the previous slide had parameters σ^2 and l in $k(t, t') = \sigma^2 \exp\left(-|t-t'|^2/2l^2\right)$. Besides that, the likelihood was parameterized by the noise variance σ_n^2.

How can we find the optimal values of these parameters?

Consider the density of $(Y(t_1), \ldots, Y(t_n))$ as a likelihood function and maximize it.



(a) GPR with default parameters

(b) GPR with optimal parameters

The Gaussian process regression algorithm

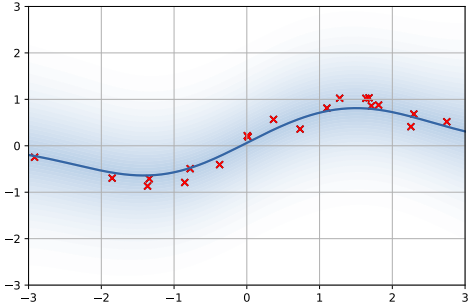
So how do we turn the data $(x_1, y_1), .., (x_n, y_n)$ into a reasonable stochastic model interpolating it?

1. Come up with a parametric families $m_{\boldsymbol{\theta}}$ and $k_{\boldsymbol{\theta}}$ for prior mean and covariance functions.
2. Use maximum likelihood estimation to pick the optimal set of parameters $\boldsymbol{\theta}$ and the optimal noise value σ^2 from data $(x_1, y_1), .., (x_n, y_n)$.
3. Perform Bayesian inference with prior $GP(m_{\boldsymbol{\theta}}, k_{\boldsymbol{\theta}})$, data $(x_1, y_1), .., (x_n, y_n)$ and likelihood noise σ^2.
   As a result, obtain the posterior $\hat{m}$ and $\hat{k}$.
4. Use
   - $\mathrm{N}(\hat{m}(u), \hat{k}(u, u))$ as a stochastic prognosis at a new location u.
   - use samples of $GP(\hat{m}, \hat{k})$ as an ensemble of possible deterministic models.

Predicting and generating sample paths

Predicting

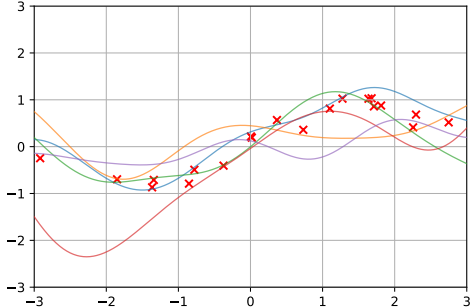
When our ultimate interest is knowing $X(t)$ for new values of t.



RBF, default parameters

Sampling

When our ultimate interest is knowing $F(X)$ for some operator F.

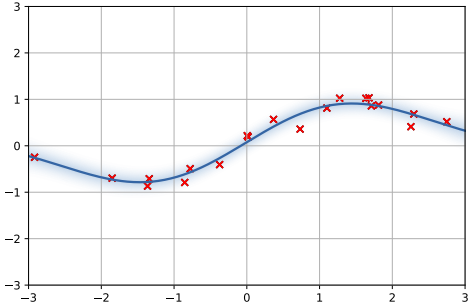


RBF, default parameters

Predicting and generating sample paths

Predicting

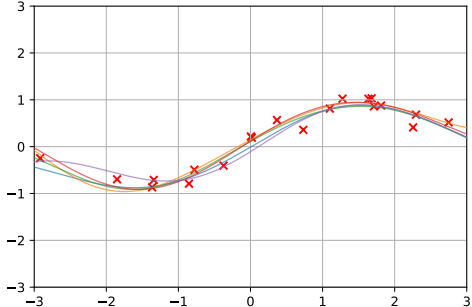
When our ultimate interest is knowing $X(t)$ for new values of t.



RBF, optimized parameters

Sampling

When our ultimate interest is knowing $F(X)$ for some operator F.

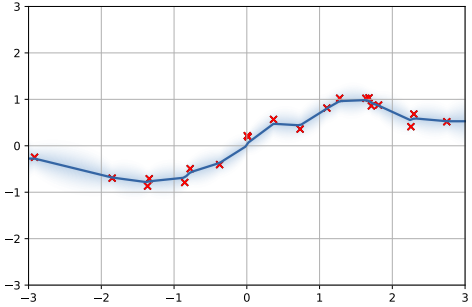


RBF, optimized parameters

Predicting and generating sample paths

Predicting

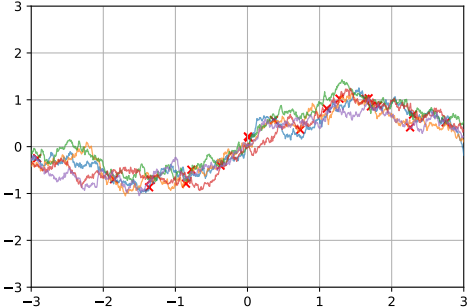
When our ultimate interest is knowing $X(t)$ for new values of t.



Brownian, optimized parameters

Sampling

When our ultimate interest is knowing $F(X)$ for some operator F.

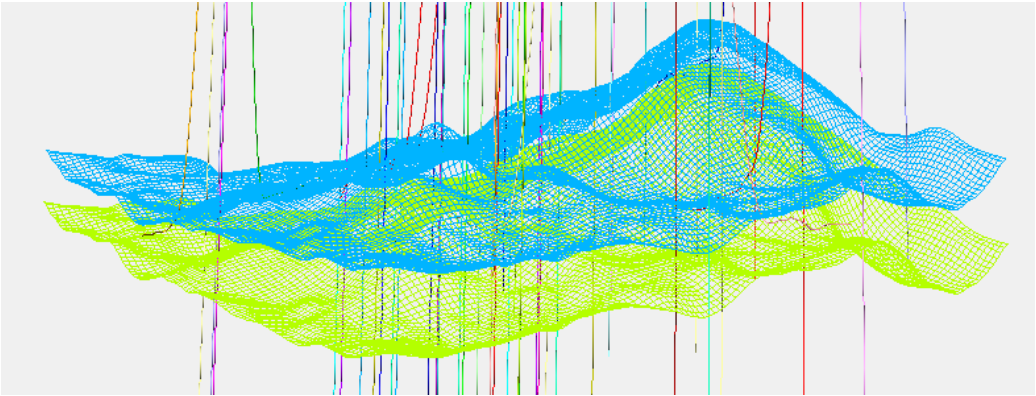


Brownian, optimized parameters

Geostatistical modeling of petroleum reservoirs

Problem: interpolate well data into the interwell space.

The data is very sparse, thus deterministic model is undesirable.

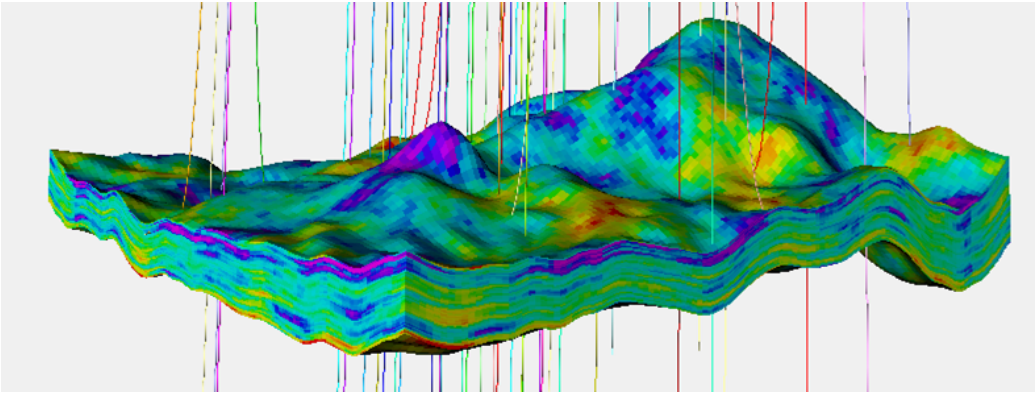


Reservoir structure, well locations.

Geostatistical modeling of petroleum reservoirs

Problem: interpolate well data into the interwell space.

The data is very sparse, thus deterministic model is undesirable.



A single sample of a Gaussian process model in the interwell space

Outline

Predicting

Recall that a Gaussian process X with mean m and covariance k conditioned on $Y = \boldsymbol{y}$ for some Gaussian vector Y has distribution

$$\hat{m}(t) = m(t) + \operatorname{Cov}(X(t), Y) \operatorname{Cov}(Y, Y)^{-1}(\boldsymbol{y} - m(\boldsymbol{t}))$$
$$\hat{k}(t, t') = \operatorname{Cov}(X(t), X(t')) - \operatorname{Cov}(X(t), Y) \operatorname{Cov}(Y, Y)^{-1} \operatorname{Cov}(Y, X(t'))$$

Take $Y = X(\boldsymbol{t}) + \sigma_n^2 \boldsymbol{\varepsilon}$ with $X(\boldsymbol{t}) = (X(t_1), \ldots, X(t_n))$ and $\boldsymbol{\varepsilon} \sim \mathrm{N}(0, I)$.
Then we have

$$\hat{m}(t) = m(t) + K_{X(t)X(\boldsymbol{t})}\Big(K_{X(\boldsymbol{t})X(\boldsymbol{t})} + \sigma_n^2 I\Big)^{-1}(\boldsymbol{y} - m(\boldsymbol{t}))$$
$$\hat{k}(t, t') = k(t, t') - K_{X(t)X(\boldsymbol{t})}\Big(K_{X(\boldsymbol{t})X(\boldsymbol{t})} + \sigma_n^2 I\Big)^{-1} K_{X(\boldsymbol{t})X(t)}$$

The time complexity of prediction is $O(n^3)$, the space complexity is $O(n^2)$.

Sampling a Gaussian vector via Cholesky decomposition

Consider a Gaussian vector $x \sim \mathrm{N}(m, \Sigma)$ of size d.

It can be represented in form

$$x = m + \Sigma^{1/2}\varepsilon \quad \text{with} \quad \varepsilon \sim \mathrm{N}(0, I).$$

$\Sigma^{1/2}$ is a matrix square root, i.e. $\Sigma^{1/2}(\Sigma^{1/2})^\top = \Sigma$.
There are many of them.

In practice $\Sigma^{1/2}$ is found through the Cholesky decomposition algorithm.

It has time complexity $O(d^3)$ and space complexity $O(d^2)$.

The naive algorithm to sample a Gaussian process

Assume we want to sample a process X with mean m and covariance k.

To do this, we discretize T into a mesh with nodes $t_1, .., t_l$.

And sample the Gaussian vector

$$\begin{pmatrix} X(t_1) \\ \dots \\ X(t_l) \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} m(t_1) \\ \vdots \\ m(t_l) \end{pmatrix} \begin{pmatrix} k(t_1, t_1) & \dots & k(t_1, t_l) \\ \vdots & \ddots & \vdots \\ k(t_l, t_1) & \dots & k(t_l, t_l) \end{pmatrix} \right)$$

This costs $O(l^3)$ time, $O(l^2)$ space and yields samples on a grid.

This complexity makes it impossible to use this algorithm in high dimensions.

Can we do better?

Yes.

To be continued...

Thank you for your attention!

viacheslav.borovitskiy@gmail.com

St Petersburg University

Mathematics & Computer Science department

Some figures were taken from: `http://inverseprobability.com/talks/`.

Gaussian random fields in machine learning

Viacheslav Borovitskiy

St. Petersburg State University
St. Petersburg Department of Steklov Mathematical Institute

Winter School in Mathematics and Theoretical Computer Science
January 29 – February 3, 2021

Part III

Efficient algorithms for sampling and conditioning

Today's talk structure

9. Efficiently sampling a stationary Gaussian processes
10. Sampling from a conditional process
11. Efficient conditioning
12. Conclusion

Outline

Stationary Gaussian processes

From now on assume that $T = \mathbb{R}^d$.

A random process X is called stationary if its distribution is unaffected by shifts.
Formally, X is stationary if $P^X_{t_1,\dots,t_n} = P^X_{t_1+t,\dots,t_n+t}$ for any $n \in \mathbb{N}$ and $t; t_1, \dots, t_n \in \mathbb{R}^d$.

If X is Gaussian, than we only need

$$m(t+\tau) = m(t) \equiv \text{const} \qquad \text{and} \qquad k(t+\tau, t'+\tau) = k(t,t') = \kappa(t-t').$$

Example: GP with RBF kernel

Stationary, since $k(t,t') = \sigma^2 \exp\left(-|t-t'|^2/2l^2\right)$ depends only on $t - t'$.

Brownian motion

Not stationary, since $k(t,t') = \min(t,t')$ for instance has $k(0,0) = 0 \neq 1 = k(1,1)$.

Spectral representation of a stationary covariance function

Define the one-parameter covariance function $\kappa(\tau) = k(t, t+\tau)$.

Bochner's theorem

If κ is positive-definite, there exists a unique finite positive measure μ on $\mathbb{R}^d$ such that

$$\kappa(\tau) = \int_{\mathbb{R}^d} e^{2\pi i \tau^\top s} \, \mathrm{d}\mu(s).$$

μ is called the spectral measure. If μ has density $\rho(s)$, it is called the spectral density. The converse statement holds as well.

Example: the RBF kernel

For $\kappa(\tau) = \sigma^2 \exp\left(-\|\tau\|^2/2l^2\right)$ we have $\rho(s) = \sigma^2 (2\pi l^2)^{d/2} \exp\left(-2\pi^2 l^2 \|s\|^2\right)$.

Random measures and a stochastic integral

Consider a measure space $(S, \mathcal{A}, \mu)$, where S is a set, $\mathcal{A} \subseteq 2^S$ is a σ-algebra and μ is a finite positive measure.

A family of complex valued random variables $F = \{F_A\}_{A \in \mathcal{A}}$ that satisfies

- $\mathbb{E}\, F(A) = 0, A \in \mathcal{A}$,
- $\operatorname{Cov}(F(A_1), F(A_2)) = \mathbb{E}\Big(F(A_1)\overline{F(A_2)}\Big) = \mu(A_1 \cap A_2), A_1, A_2 \in \mathcal{A}$,
- $F(\cup_{j=1}^n A_j) = \sum_{j=1}^n F(A_j)$ a.s. for $n \in \mathbb{N}$ and non-intersecting $A_1, \ldots, A_n \in \mathcal{A}$

is called a centered random measure with uncorrelated values with intensity measure μ.

Define for a simple function $f = \sum_{j=1}^n c_j \mathbb{1}_{A_j}$ with $A_j \in \mathcal{A}$ the integral

$$\int_{\mathbb{R}^d} f \,\mathrm{d}F = \sum_{j=1}^n c_j F(A_j)$$

For simple functions $\langle f_1, f_2 \rangle_{L^2(\mathbb{R}^d, \mu)} = \operatorname{Cov}(\int_{\mathbb{R}^d} f_1 \,\mathrm{d}F, \int_{\mathbb{R}^d} f_2 \,\mathrm{d}F)$ — the isometry prop.
Hence we can extend the integral to arbitrary $f \in L^2(\mathbb{R}^d, \mu)$.

Spectral representation of a stationary Gaussian process

Let F be a Gaussian centered random measure with uncorrelated values. Denote its intensity measure by μ. Define

$$Y(t) = \int e^{2\pi i t^\top u} \, \mathrm{d}F(u), t \in \mathbb{R}^d$$

then, thanks to the isometry property of the integral, we have

$$\mathrm{Cov}(Y(t), Y(t')) = \int_{\mathbb{R}^d} e^{2\pi i (t-t')u} \, \mathrm{d}\mu(u) = K(t - t').$$

Thus, Y is a stationary Gaussian process.

The remarkable fact is that every stationary GP with continuous covariance admits such a spectral representation. In this case $F(A) = U(A) + iV(A)$, $A \in \mathcal{A}$, where

- $U(A), V(A) \sim \mathrm{N}(0, \mu(A)/2)$,
- $U(A) \perp V(A)$ and $U(A) \perp U(B)$, $V(A) \perp V(B)$, $U(A) \perp V(B)$ for $A \cap B = \varnothing$.

Sampling by means of the spectral representation

Assume that $X \sim \mathrm{GP}(0, k)$ is a stationary GP over $\mathbb{R}^d$ with known spectral measure μ.
— in practice, we usually know the spectral density in closed form.

Consider some partition $\mathbb{R}^d = \cup_{j=1}^J A_j$ and select some points $u_j \in A_j$, then write

$$X(t) = \int_{\mathbb{R}^d} e^{2\pi i t^\top u} \, \mathrm{d}F(u) = \sum_{j=1}^J \int_{A_j} e^{2\pi i t^\top u} \, \mathrm{d}F(u)$$
$$\approx \sum_{j=1}^J e^{2\pi i t^\top u_j} \int_{A_j} \mathrm{d}F(u) = \sum_{j=1}^J e^{2\pi i t^\top u_j} F(A_j).$$

To sample from the right hand side it is enough to sample random variables $F(A_j)$.
This is easy since

- $F(A_j) = U(A_j) + iV(A_j)$ and all $\{U(A_j)\}_{j=1}^J \cup \{V(A_j)\}_{j=1}^J$ are independent,
- to sample $U(A_j)$ or $V(A_j)$ it is enough to compute $\mu(A_j) = \int_{A_j} 1 \, \mathrm{d}\mu$.

Basic error analysis of the method

Let us show that $X(t) \approx \sum_{j=1}^{J} e^{2\pi i t^\top u_j} F(A_j)$ is indeed an approximation. Write

$$\mathbb{E}\left|X(t) - \sum_{j=1}^{J} e^{2\pi i t^\top u_j} F(A_j)\right|^2 = \mathbb{E}\left|\int_{\mathbb{R}^d} \left(e^{2\pi i t^\top u} - \sum_{j=1}^{J} e^{2\pi i t^\top u_j} \mathbb{1}_{A_j}(u)\right) \mathrm{d}F(u)\right|^2$$

$$= \int_{\mathbb{R}^d} \left|e^{2\pi i t^\top u} - \sum_{j=1}^{J} e^{2\pi i t^\top u_j} \mathbb{1}_{A_j}(u)\right|^2 \mathrm{d}\mu(u)$$

To estimate the right hand side we need to

- pick a reasonable partition $\mathbb{R}^d = \cup_{j=1}^{J} A_j$,
- leverage the "decay" property of μ: that $\mu(|t| > \alpha) \xrightarrow[\alpha \to \infty]{} 0$ at some rate.

Main idea

For $j = J$: make A_j large but with small $\mu(A_j)$.

For $j \neq J$: make A_j small so that $e^{2\pi i t^\top u_j}$ is close to $e^{2\pi i t^\top u}$.

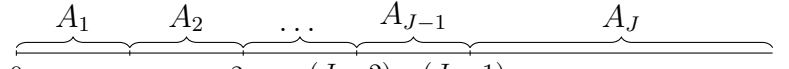
Basic error analysis of the method (continued, part 2)

Consider $d = 1$ for simplicity and assume $\mu(|t| > \alpha) = O(1/\alpha^p)$ for some $p > 0$.

Without loss of generality, assume we only need to estimate

$$\int_{\mathbb{R}_+} \left| e^{2\pi i t^\top u} - \sum_{j=1}^{J} e^{2\pi i t^\top u_j} \mathbb{1}_{A_j}(u) \right|^2 \mathrm{d}\mu(u).$$

Fix a small $\varepsilon > 0$ and partition $\mathbb{R}_+ = \cup_{j=1}^{J} A_j$ like this:



Assume additionally that $\mu(\mathbb{R}_+) \le 1$ and that $|t| \le t_{\max}$. Then

$$\int_{\mathbb{R}_+} |\ldots|^2 \mathrm{d}\mu(u) = \sum_{j=1}^{J-1} \int_{A_j} \underbrace{\left| e^{2\pi i t^\top u} - e^{2\pi i t^\top u_j} \right|^2}_{\le 4\pi^2 |t|^2 |u-u_j|^2 \le 4\pi^2 t_{\max}^2 \varepsilon^2} \mathrm{d}\mu(u) + \int_{A_J} \underbrace{\left| e^{2\pi i t^\top u} - e^{2\pi i t^\top u_J} \right|^2}_{\le 4} \mathrm{d}\mu(u)$$

Basic error analysis of the method (continued, part 3)

$$\int_{\mathbb{R}_+} |\dots|^2 \,\mathrm{d}\mu(u) = \sum_{j=1}^{J-1} \int_{A_j} \underbrace{\left| e^{2\pi i t^\top u} - e^{2\pi i t^\top u_j} \right|^2}_{\leq 4\pi^2 |t|^2 |u-u_j|^2 \leq 4\pi^2 t_{\max}^2 \varepsilon^2} \mathrm{d}\mu(u) + \int_{A_J} \underbrace{\left| e^{2\pi i t^\top u} - e^{2\pi i t^\top u_J} \right|^2}_{\leq 4} \mathrm{d}\mu(u)$$

$$\leq \sum_{j=1}^{J-1} 4\pi^2 t_{\max}^2 \varepsilon^2 + 4\mu(|u| > (J-1)\varepsilon)$$

$$\leq (J-1) 4\pi^2 t_{\max}^2 \varepsilon^2 + 4 \frac{1}{((J-1)\varepsilon)^p}$$

Taking e.g. $\varepsilon \approx (J-1)^{-3/4}$, we have

$$\int_{\mathbb{R}_+} |\dots|^2 \,\mathrm{d}\mu(u) \leq \frac{4\pi^2 t_{\max}^2}{(J-1)^{1/2}} + \frac{4}{(J-1)^{p/4}} \xrightarrow[J\to\infty]{} 0$$

Covariance approximation point of view

Denote $X_{DFF}(t) = \sum_{j=1}^{J} e^{2\pi i t^\top u_j} F(A_j)$. "DFF" is for Deterministic Fourier Features.

$X_{DFF}(t)$ is a Gaussian process with zero mean and covariance

$$k_{DFF}(t, t') = \mathrm{Cov}(X_{DFF}(t), X_{DFF}(t')) = \sum_{j=1}^{J} \mu(A_j) e^{2\pi i (t-t')^\top u_j}.$$

Bochner's theorem

If κ is positive-definite, there exists a unique finite positive measure μ on $\mathbb{R}^d$ such that

$$\kappa(\tau) = \int_{\mathbb{R}^d} e^{2\pi i \tau^\top s} \, \mathrm{d}\mu(s).$$

μ is called the spectral measure. If μ has density $\rho(s)$, it is called the spectral density. The converse statement holds as well.

Obviously, k_{DFF} is can be obtained by approximating k via the Riemannian sum.

Random Fourier Features

The covariance approximation point of view suggests an alternative method.

Consider the Monte-Carlo approximation

$$\kappa(\tau) = \int_{\mathbb{R}^d} e^{2\pi i \tau^\top s} \, \mathrm{d}\mu(s) = \mu(\mathbb{R}^d) \mathop{\mathbb{E}}_{s \sim \mu/\mu(\mathbb{R}^d)} e^{2\pi i \tau^\top s} \approx \frac{\mu(\mathbb{R}^d)}{J} \sum_{j=1}^{J} e^{2\pi i \tau^\top s_j} =: k_{RFF},$$

where $s_j \overset{\text{iid}}{\sim} \mu/\mu(\mathbb{R}^d)$ and "RFF" is for Random Fourier Features.

Two approximations

k_{RFF} corresponds to: $X_{RFF}(t) = \mu(\mathbb{R}^d)/J \sum_{j=1}^{J} w_j e^{2\pi i t^\top s_j}$, $w_j \overset{\text{iid}}{\sim} N(0,1)$,

k_{DFF} corresponds to: $X_{DFF}(t) = \sum_{j=1}^{J} (w_{j1} + i w_{j2}) e^{2\pi i t^\top u_j}$, $w_{j\cdot} \overset{\text{iid}}{\sim} N(0, \mu(A_j)/2)$.

In practice — RFF: Monte Carlo integration behaves well in high dimension.

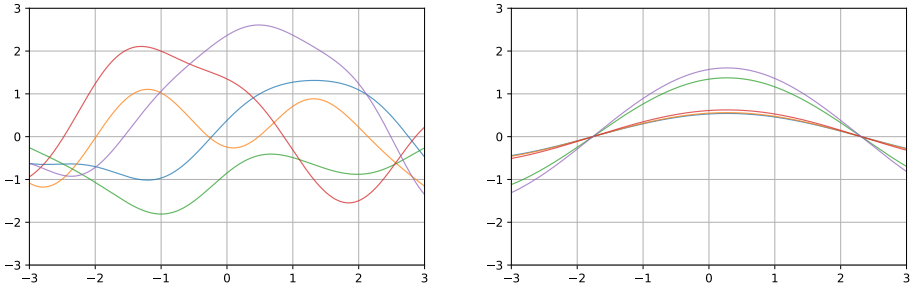
Results and example

The complexity of generating a sample path on l-sized grid with J features is

- $O(l \cdot J)$ time,
- $O(\max(l, J))$ space.

And actually, we don't need a grid! It's very useful for e.g. optimization.

Consider $X \sim \mathrm{GP}(0, k)$, where $k(t, t') = \sigma^2 \exp\left(-\|t-t'\|^2 / 2l^2\right)$ with $\sigma^2 = 1$, $l = 1$.



(a) True samples

(b) RFF samples with $J = 1$

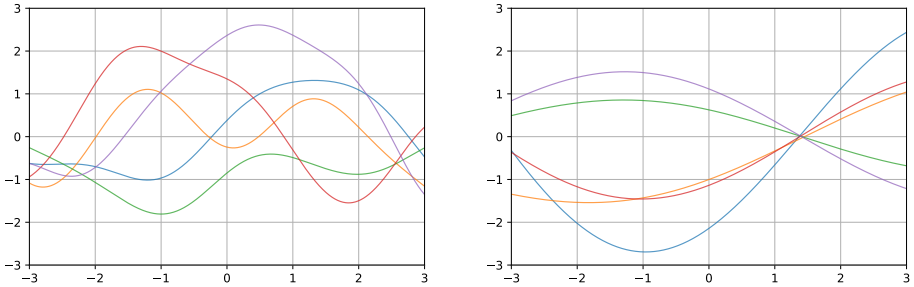
Results and example

The complexity of generating a sample path on l-sized grid with J features is

- $O(l \cdot J)$ time,
- $O(\max(l, J))$ space.

And actually, we don't need a grid! It's very useful for e.g. optimization.

Consider $X \sim \mathrm{GP}(0, k)$, where $k(t, t') = \sigma^2 \exp\left(-\|t-t'\|^2 / 2l^2\right)$ with $\sigma^2 = 1$, $l = 1$.



(a) True samples

(b) RFF samples with $J = 2$

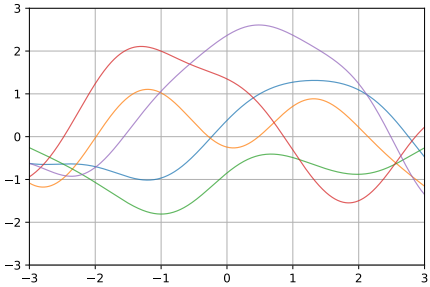
Results and example

The complexity of generating a sample path on l-sized grid with J features is

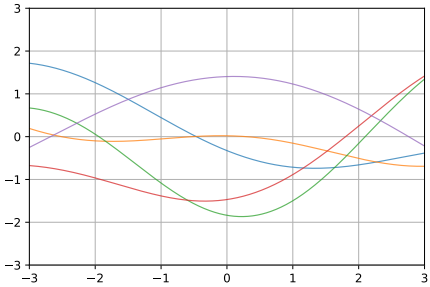
- $O(l \cdot J)$ time,
- $O(\max(l, J))$ space.

And actually, we don't need a grid! It's very useful for e.g. optimization.

Consider $X \sim \mathrm{GP}(0, k)$, where $k(t, t') = \sigma^2 \exp\left(-\|t-t'\|^2/2l^2\right)$ with $\sigma^2 = 1$, $l = 1$.



(a) True samples



(b) RFF samples with $J = 4$

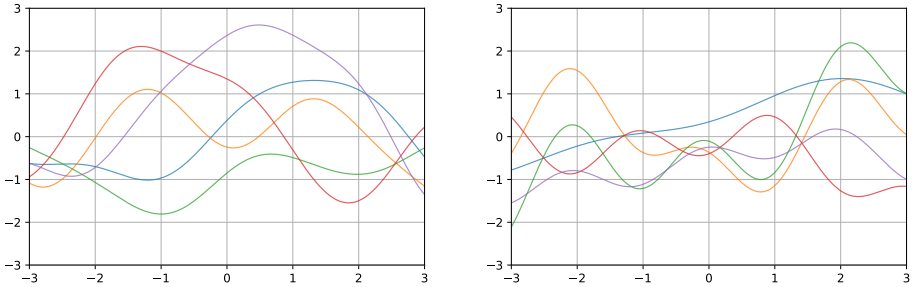
Results and example

The complexity of generating a sample path on l-sized grid with J features is

- $O(l \cdot J)$ time,
- $O(\max(l, J))$ space.

And actually, we don't need a grid! It's very useful for e.g. optimization.

Consider $X \sim \mathrm{GP}(0, k)$, where $k(t, t') = \sigma^2 \exp\left(-\|t-t'\|^2 / 2l^2\right)$ with $\sigma^2 = 1$, $l = 1$.



(a) True samples

(b) RFF samples with $J = 8$

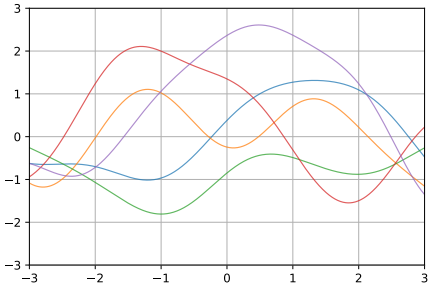
Results and example

The complexity of generating a sample path on l-sized grid with J features is

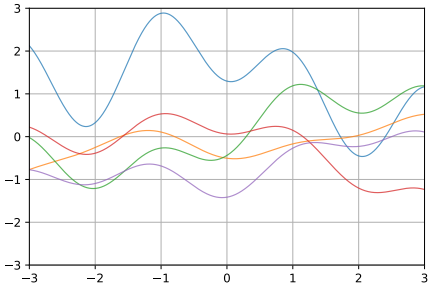
- $O(l \cdot J)$ time,
- $O(\max(l, J))$ space.

And actually, we don't need a grid! It's very useful for e.g. optimization.

Consider $X \sim \mathrm{GP}(0, k)$, where $k(t, t') = \sigma^2 \exp\left(-\|t-t'\|^2/2l^2\right)$ with $\sigma^2 = 1$, $l = 1$.



(a) True samples



(b) RFF samples with $J = 16$

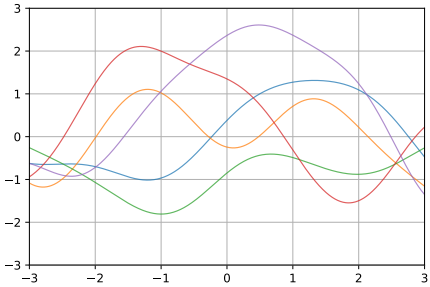
Results and example

The complexity of generating a sample path on l-sized grid with J features is

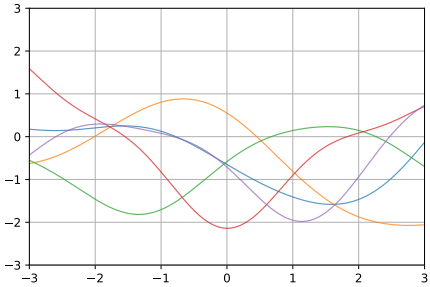
- $O(l \cdot J)$ time,
- $O(\max(l, J))$ space.

And actually, we don't need a grid! It's very useful for e.g. optimization.

Consider $X \sim \mathrm{GP}(0, k)$, where $k(t, t') = \sigma^2 \exp\left(-\|t-t'\|^2/2l^2\right)$ with $\sigma^2 = 1$, $l = 1$.



(a) True samples



(b) RFF samples with $J = 32$

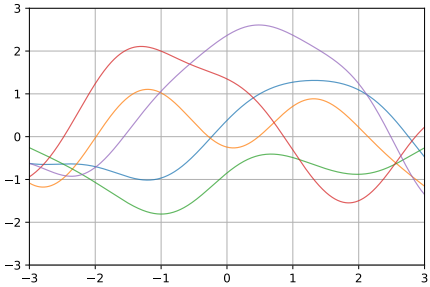
Results and example

The complexity of generating a sample path on l-sized grid with J features is

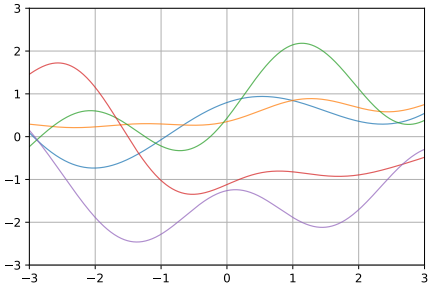
- $O(l \cdot J)$ time,
- $O(\max(l, J))$ space.

And actually, we don't need a grid! It's very useful for e.g. optimization.

Consider $X \sim \mathrm{GP}(0, k)$, where $k(t, t') = \sigma^2 \exp\left(-\|t-t'\|^2/2l^2\right)$ with $\sigma^2 = 1$, $l = 1$.



(a) True samples



(b) RFF samples with $J = 64$

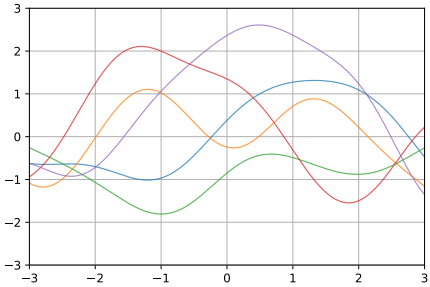
Results and example

The complexity of generating a sample path on l-sized grid with J features is

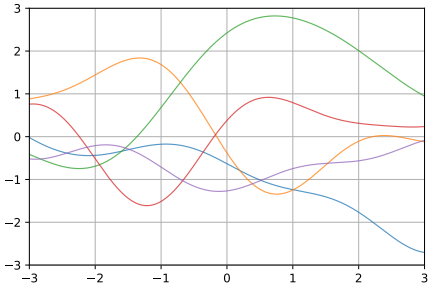
- $O(l \cdot J)$ time,
- $O(\max(l, J))$ space.

And actually, we don't need a grid! It's very useful for e.g. optimization.

Consider $X \sim \mathrm{GP}(0, k)$, where $k(t, t') = \sigma^2 \exp\left(-\|t-t'\|^2/2l^2\right)$ with $\sigma^2 = 1$, $l = 1$.



(a) True samples



(b) RFF samples with $J = 128$

Outline

Sampling from a conditional process

In practice, the unconditional process X (the prior) is usually stationary.
But the conditional process (the posterior) is not!

Recall the conditioning formulas

$$\hat{m}(t) = m(t) + K_{X(t)X(\boldsymbol{t})}\Big(K_{X(\boldsymbol{t})X(\boldsymbol{t})} + \sigma_n^2 I\Big)^{-1}(y - m(\boldsymbol{t}))$$

$$\hat{k}(t,t') = k(t,t') - K_{X(t)X(\boldsymbol{t})}\Big(K_{X(\boldsymbol{t})X(\boldsymbol{t})} + \sigma_n^2 I\Big)^{-1}K_{X(\boldsymbol{t})X(t)}$$

E.g. when $\sigma_n^2 = 0$, we have $\hat{k}(t_j, t_j) = 0$ where t_j are data locations.

RFF and DFF approximate GP (almost) with a Bayesian linear regression model.

k_{RFF} corresponds to: $\quad X_{RFF}(t) = \mu(\mathbb{R}^d)/J \sum_{j=1}^{J} w_j e^{2\pi i t^\top s_j}$ $\quad$ with $w_j \overset{\text{iid}}{\sim} N(0,1)$.

Can condition the vector of weights w and then sample with $O(J^3)$.
Turns out it is not very fast and not very accurate. Let's explore an alternative.

An alternative way of conditioning a Gaussian vector

Consider a random vector divided in two parts (assume zero mean for simplicity)

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \sim \mathrm{N}(0, \Sigma) = \mathrm{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \right).$$

Let us find the best linear estimator of x_1 given x_2.
Formally, find a matrix A that minimizes $\mathbb{E}\|x_1 - Ax_2\|^2$.

Omitting the computations, we get $A = \Sigma_{12}\Sigma_{22}^{-1}$.
Also we get that $x_1 - \Sigma_{12}\Sigma_{22}^{-1}x_2 \perp x_2$.

Because of that, we have

$$x_1 = \underbrace{\Sigma_{12}\Sigma_{22}^{-1}x_2}_{\text{function of } x_2} + \underbrace{(x_1 - \Sigma_{12}\Sigma_{22}^{-1}x_2)}_{\text{independent of } x_2}.$$

An alternative way of conditioning a Gaussian vector

We have

$$x_1 = \underbrace{\Sigma_{12}\Sigma_{22}^{-1}x_2}_{\text{function of } x_2} + \underbrace{(x_1 - \Sigma_{12}\Sigma_{22}^{-1}x_2)}_{\text{independent of } x_2}.$$

Lemma

Assume that a and b are two random vectors. If we have almost surely

$$a = f(b) + c$$

for some deterministic f and for some c independent of b, then

$$p(a \mid b = \beta) = p(f(\beta) + c).$$

Applying this lemma, we get that

$$\Sigma_{12}\Sigma_{22}^{-1}\boldsymbol{y} + (x_1 - \Sigma_{12}\Sigma_{22}^{-1}x_2) \sim p(x_1 \mid x_2 = \boldsymbol{y}).$$

An alternative way of conditioning a Gaussian vector (continued)

We have

$$\Sigma_{12}\Sigma_{22}^{-1}\boldsymbol{y} + (x_1 - \Sigma_{12}\Sigma_{22}^{-1}x_2) \sim p(x_1 \mid x_2 = \boldsymbol{y}).$$

This allows transforming prior samples to posterior samples!

I.e. to sample from $p(x_1 \mid x_2 = \boldsymbol{y})$:

1. sample $(\hat{x}_1, \hat{x}_2)^\top \sim p(x_1, x_2)$,
2. return $\Sigma_{12}\Sigma_{22}^{-1}\boldsymbol{y} + (\hat{x}_1 - \Sigma_{12}\Sigma_{22}^{-1}\hat{x}_2)$.

This trick for sampling from conditioned Gaussian was rediscovered many times.
I call it the Matheron's formula after a French geostatistician Georges Matheron.

Sampling from a conditional process

When lifted from Gaussian vectors to GPs, Matheron's formula states that

$$X_c(t) = X(t) + \operatorname{Cov}(X(t), Y) \operatorname{Cov}(Y, Y)^{-1}(\boldsymbol{y} - Y)$$

has the distribution $X \mid Y = \boldsymbol{y}$. With this, we can

1. sample from the unconditional process X e.g. with RFF,
   — costs $O(l \cdot J)$ time
   — costs $O(\min(l, J))$ space
   — for l-sized grid and J approximating terms
2. update this sample to get a sample from the conditional process.
   — costs $O(n^3 + ln)$ time
   — costs $O(n^2)$ space
   — for n-dimensional data and an l-sized grid

And actually we don't need grids!

Sampling from a conditional process

Interactive demo `https://sml-group.cc/blog/2020-gp-sampling/`.

Outline

The problem

To predict or sample from a conditional Gaussian process we need to solve $n \times n$ linear system incurring $O(n^3)$ time cost and $O(n^2)$ space cost.

Recall the conditioning formulas

$$\hat{m}(t) = m(t) + K_{X(t)X(\boldsymbol{t})}\left(K_{X(\boldsymbol{t})X(\boldsymbol{t})} + \sigma_n^2 I\right)^{-1}(\boldsymbol{y} - m(\boldsymbol{t}))$$

$$\hat{k}(t, t') = k(t, t') - K_{X(t)X(\boldsymbol{t})}\left(K_{X(\boldsymbol{t})X(\boldsymbol{t})} + \sigma_n^2 I\right)^{-1} K_{X(\boldsymbol{t})X(t)}$$

Can we do better than this?

The main idea

Denote $X_c \sim \mathrm{GP}(\hat{m}, \hat{k})$ a Gaussian process with conditional distribution.

The main idea

Consider some parametric family of Gaussian processes (or rather their distributions)

$$\{G_\gamma\}_{\gamma \in \Gamma}$$

such that

- G_γ is simpler to predict with than X_c,
- $\mathrm{d}(G_\gamma, X_c)$ for some distance d can be made small,
- $\mathrm{d}(G_\gamma, X_c)$ can be computed and differentiated efficiently.

Find $\hat{\gamma} = \arg\min \mathrm{d}(G_\gamma, X_c)$ and use $G_{\hat{\gamma}}$ instead of X_c.

A family G_γ

The simplest family that we can consider is

$$G_\gamma := X \mid X(\boldsymbol{z}) + \sigma_z^2 \varepsilon(\boldsymbol{z}) = \boldsymbol{u},$$

where, for some $s \ll n$,

- $\boldsymbol{z} = (z_1, \ldots, z_s)^\top$ are pseudo-locations,
- $\boldsymbol{u} = (u_1, \ldots, u_s)^\top$ are pseudo-observations,
- σ_z^2 is pseudo-observation noise,
- $\gamma = (\boldsymbol{z}, \boldsymbol{u}, \sigma_z^2)$.

Here, as before $X(\boldsymbol{z}) = (X(z_1), \ldots, X(z_s))^\top$ and $\varepsilon(\boldsymbol{z}) = (\varepsilon(z_1), \ldots, \varepsilon(z_s))^\top$.

We seek to find pseudo-data of smaller size that can be used instead of the actual data.

A more expressive family G_γ

We can make pseudo-observations random. For any $k \in \mathbb{N}$ and any $\tilde{\boldsymbol{t}} = \left(\tilde{t}_1, \ldots, \tilde{t}_k\right)^\top$

$$G_\gamma(\tilde{\boldsymbol{t}}) := \int_{\mathbb{R}^s} p\Big(X(\tilde{\boldsymbol{t}}) \mid X(\boldsymbol{z}) = \boldsymbol{u}\Big) q(\boldsymbol{u}) \, \mathrm{d}\boldsymbol{u},$$

where, for some $s \ll n$,

- $\boldsymbol{z} = (z_1, \ldots, z_s)^\top$ are pseudo-locations,
- $q(\boldsymbol{u}) = \mathrm{N}(m_{\boldsymbol{u}}, \Sigma_{\boldsymbol{u}})$ are random pseudo-observations,
- $\gamma = (\boldsymbol{z}, m_{\boldsymbol{u}}, \Sigma_{\boldsymbol{u}})$.

Here, as before

- $G_\gamma(\tilde{\boldsymbol{t}}) = \left(G_\gamma(\tilde{t}_1), \ldots, G_\gamma(\tilde{t}_k)\right)^\top$
- $X(\boldsymbol{z}) = (X(z_1), \ldots, X(z_s))^\top$,
- $X(\tilde{\boldsymbol{t}}) = \left(X(\tilde{t}_1), \ldots, X(\tilde{t}_k)\right)^\top$.

The distance: KL-divergence

Consider two densities $p_1(x)$, $p_2(x)$. Then

$$D_{KL}(p_1(x) \parallel p_2(x)) \stackrel{def}{=} \int p_1(x) \log \frac{p_1(x)}{p_2(x)} dx$$

Properties

- It is non-negative: $D_{KL}(p_1(x) \parallel p_2(x)) \geq 0$.
- It is non-degenerate: $D_{KL}(p_1(x) \parallel p_2(x)) = 0$ implies $p_1(x) = p_2(x)$.

It is not symmetric: $D_{KL}(p_1(x) \parallel p_2(x)) \neq D_{KL}(p_2(x) \parallel p_1(x))$!

When $p_1(x) = 0$, the density $p_2(x)$ may be arbitrary.
When $p_2(x) = 0$ we should have $p_1(x) = 0$.

KL-divergence between X_c and G_γ

Take G_γ corresponding to random pseudo-observations.
Recall that $\boldsymbol{t}$ denotes data locations and $\boldsymbol{z}$ denotes pseudo-locations.
Take $k \in \mathbb{N}$ and $\tilde{\boldsymbol{t}} = \left(\tilde{t}_1, \ldots, \tilde{t}_k\right)^\top$, then consider

$$D_{KL}(G_\gamma(\tilde{\boldsymbol{t}} \oplus \boldsymbol{t} \oplus \boldsymbol{z}) \parallel X_c(\tilde{\boldsymbol{t}} \oplus \boldsymbol{t} \oplus \boldsymbol{z})),$$

where $\oplus$ denotes vector concatenation.

A simple computation (Matthews et al 2016 AISTATS) gives

$$D_{KL}(G_\gamma(\tilde{\boldsymbol{t}} \oplus \boldsymbol{t} \oplus \boldsymbol{z}) \parallel X_c(\tilde{\boldsymbol{t}} \oplus \boldsymbol{t} \oplus \boldsymbol{z})) = D_{KL}(G_\gamma(\boldsymbol{t} \oplus \boldsymbol{z}) \parallel X_c(\boldsymbol{t} \oplus \boldsymbol{z})),$$

i.e. this KL-divergence doesn't depend on $\tilde{\boldsymbol{t}}$ or $X(\tilde{\boldsymbol{t}})$.

Minimizing the specific KL-divergence between a pair of Gaussian vectors implies the minimization of KL-divergences between all pairs of GP's marginal distributions!

KL-divergence between X_c and G_γ (continued)

One can show that evaluating and differentiating $D_{KL}(G_\gamma(\boldsymbol{t} \oplus \boldsymbol{z}) \parallel X_c(\boldsymbol{t} \oplus \boldsymbol{z}))$ costs
— $O(s^2 \cdot n)$ time ($O(s^3)$ with additional approximation),
— $O(s \cdot n)$ space ($O(s^2)$ with additional approximation).

Thus the problem of finding the optimal $\gamma = (\boldsymbol{z}, m_{\boldsymbol{u}}, \Sigma_{\boldsymbol{u}})$ can be efficiently solved by gradient descent.

Sampling and predicting from the approximate conditional

It's not hard to write out the the explicit mean and covariance functions of G_γ. We have $G_\gamma \sim \mathrm{GP}(\tilde{m}, \tilde{k})$ with

$$\begin{aligned}\tilde{m}(t) &= m(t) + K_{X(t)X(\boldsymbol{z})} K^{-1}_{X(\boldsymbol{z})X(\boldsymbol{z})} (m_{\boldsymbol{u}} - m(\boldsymbol{z})) \\ \tilde{k}(t,t') &= k(t,t') - K_{X(t)X(\boldsymbol{z})} K^{-1}_{X(\boldsymbol{z})X(\boldsymbol{z})} K_{X(\boldsymbol{z})X(t)} \\ &\quad + K_{X(t)X(\boldsymbol{z})} K^{-1}_{X(\boldsymbol{z})X(\boldsymbol{z})} \Sigma_{\boldsymbol{u}} K^{-1}_{X(\boldsymbol{z})X(\boldsymbol{z})} K_{X(\boldsymbol{z})X(t)}\end{aligned}$$

Predictions with these formulas cost $O(s^3)$ time and $O(s^2)$ space.

To sample we can

1. sample $\hat{\boldsymbol{u}} \sim \mathrm{N}(m_{\boldsymbol{u}}, \Sigma_{\boldsymbol{u}})$ — costs $O(s^3)$ time and $O(s^2)$ space,
2. sample $p(X \mid X(\boldsymbol{z}) = \hat{\boldsymbol{u}})$ with RFF + Matheron's formula — costs $O(l \cdot J + s^3 + s \cdot l)$ time and $O(\min(l, J) + s^2)$ space.

Outline

Summary

We have learned

- what GPs are,
- what are their applications in ML,
- how to predict with GPs and how to sample them,
- how to do this efficiently but approximately (in some scenarios).

GPs are state of the art models for

- small data,
- uncertainty quantification problems.

There is a number of Python (and other language) libraries, e.g.

- NumPy-based Python library `https://sheffieldml.github.io/GPy/`,
- TensorFlow-based Python library `github.com/GPflow/GPflow`,
- PyTorch-based Python library `gpytorch.ai`.

More on the modern methods and problems

- What if we want to do classification instead of regression?
  Keywords: non-Gaussian likelihoods.
- Additional applications
  E.g. Gaussian Process Latent Variable Model — dimensional reduction with GPs.
- More complex GP-based models.
  E.g. Deep Gaussian Processes, Convolutional GPs.
- Theoretical questions.
  E.g. Bayesian neural network convergence to GPs.

Thank you for your attention!

viacheslav.borovitskiy@gmail.com

St Petersburg University

Mathematics & Computer Science department

Some figures were taken from: `http://inverseprobability.com/talks/`.