UPPSALA
UNIVERSITET

Advanced Probabilistic Machine Learning

Lecture 7 – Gaussian processes part I

UPPSALA
UNIVERSITET

Andreas Lindholm
Division of Systems and Control
Department of Information Technology
Uppsala University

andreas.lindholm@it.uu.se
`www.it.uu.se/katalog/andsv164`

Summary of lecture 3–6: Idea

Main idea:

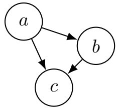
1. You express your problem as a probabilistic model (different random variables that depends on each other) using any of the graphical model types.
2. You solve for/compute/infer the distribution you are interested in (usually something like $p(\theta \mid y)$). Sometimes you can do it analytically, but often it requires approximate inference (Monte Carlo or variational inference/expectation propagation).

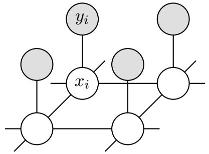
UPPSALA UNIVERSITET

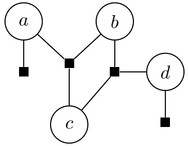
Summary of lecture 3–6: Graphical models

Three types of graphs

1. **Bayesian Networks**: represent dependencies between variables using a *Directed Acyclic Graph* (DAG)
2. **Markov Random Fields**: represents Markovian dependencies between variables using an *undirected graphs*.
3. **Factor Graphs**: represents both *variables* and *relationships* between variables (can represent both BNs and MRFs)







Three different ways to graphically illustrate how the random variables (the nodes) interact!

UPPSALA UNIVERSITET

Summary of lecture 3–6: Graphical models

Graphical models provide a general "language" for expressing probabilistic models. The point of a graphical model is to **illustrate** how the random variables in the model are assumed to interact.

A graphical model might reveal the most important information better than plain equations.

A few examples from the International Conference on Machine Learning (ICML) this year:

Robust Estimation of Tree Structured Gaussian Graphical Models

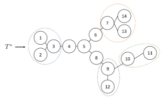
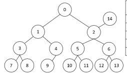


Figure 1. For this T^*, $\mathcal{T}_{T^*}$ is the set of all the trees obtained by permuting the nodes within each of the dotted regions. We prove that while T^* is unidentifiable, under our noise model, we can recover $\mathcal{T}_{T^*}$. In other words, the tree structure is recoverable up to



Node Set	Classification	i_k, pairing (If Applicable)
{0, 1, 3, 7}	Non-Star	1, {{0,1}, {3,7}}
{7, 8, 9, 10}	Non-Star	1, {{7,8}, {9,10}}
{14, 2, 10, 11}	Non-Star	2, {{14,2}, {10,11}}
{7, 9, 1, 6}	Star	N.A.
{1, 14, 10, 6}	Star	N.A.

Figure 2. Examples of classification of 4 nodes as star shape or non star shape. If they form a non star shape, the nodes are grouped in pairs of 2.

The full proof which includes arriving at this decomposition and showing that the conditional independence structure of

Learning Models from Data with Measurement Error: Tackling Underreporting

type and relationship of these errors are necessary to guarantee that the error distribution is estimable. For example, in Section 4.2 we prove for our model that the error distribution is estimable from two error-prone measurements which are independent given the true exposure. Once the error distribution is estimated, there are a number of ways one can correct for the missing ground truth observations such as imputation or sampling (Carroll et al., 2006). Using auxiliary data typically requires the least restrictive modeling assumptions and is therefore preferable when such data is available. It is unsurprising then that this approach forms the basis for much of the classic literature on measurement

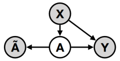


Figure 1. Proposed graphical model. Y is the outcome of interest, A is the true binary exposure, $\tilde{A}$ is the observed exposure, and X is a vector of covariates. Grey variables are observed and white variables are not.

TibGM: A Transferable and Information-Based Graphical Model Approach for Reinforcement Learning

provides further modeling flexibility, e.g. the unsupervised pretraining procedure.

In the experiments, we compare to several other state-of-the-art algorithms, like the sample-efficient deterministic off-policy algorithm, DDPG (Lillicrap et al., 2015). We also compare to proximal policy optimization (PPO, Schulman et al., 2017b) which mostly converges to nearly deterministic policies as well. Other works involved in the comparisons include algorithms with evolutionary nature like ERL (Khadka & Tumer, 2018) and GEP-PG (Colas et al., 2018), and ProMP (Rothfuss et al., 2019), whose target is to adapt well to new similar environments.

3. Methodology

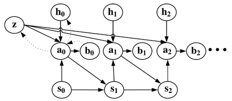
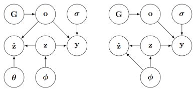


Figure 1. The graphical model depicting the proposed `TibGM` model. Actions at each time step depend on the states $\mathbf{s_t}$, as well as the disentangled latent space consisting of time-variant 'local' components $\mathbf{h_t}$ and 'global' components $\mathbf{z}$, encouraging generalisation. The variable $\mathbf{b_t}$ conveys information about the reward,

AReS and MaRS - Adversarial and MMD-Minimizing Regression for SDEs



(a) SDE-based model (b) Data-based model

Figure 1. Generative models for the two different ways to compute the derivatives of the latent states $\mathbf{z}$.

where

$$\mathbf{\Sigma} = \mathbf{C}_\phi + \mathbf{T} + \mathbf{B}\mathbf{\Omega}\mathbf{B}^T. \qquad (23)$$

Thanks to the latent state representation, the diffusion matrix

where δ represents the dirac delta.

3.4. Inference

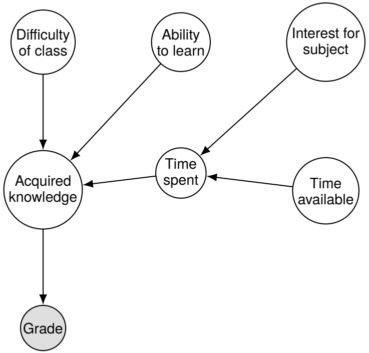
Combined with the modeling paradigms introduced in the previous sections, this yields the two generative models for the observations in Figure 1. The graphical model in Figure 1a represents the derivatives we get via the generative process described by the SDEs, in particular the nonlinear drift function $\mathbf{f}$. The model in Figure 1b represents the derivatives yielded by the generative process described by the Gaussian process. Assuming a perfect GP fit and access to the true parameters $\boldsymbol{\theta}$, intuitively these two distributions should be equal. We thus want to find parameters $\boldsymbol{\theta}$ that minimizes the difference between these two distributions.

UPPSALA UNIVERSITET

Summary of lecture 3–6: Why graphical models?

Question: How hard is this course?

Attempt to answer using a probabilistic model (expressed as a Bayesian network)



- The model is an assumption. Might or might not be true.
- Remains to specify: the mathematical expression for each arrow, as well as priors for the root nodes.
- By computing $p(\text{difficulty} \mid \text{grades})$, we will find out how hard this class is!

A probabilistic model reasons about things we have not observed (the difficulty) based on what we have observed (the grades). The graphical model is a way to visualize the model.

UPPSALA UNIVERSITET

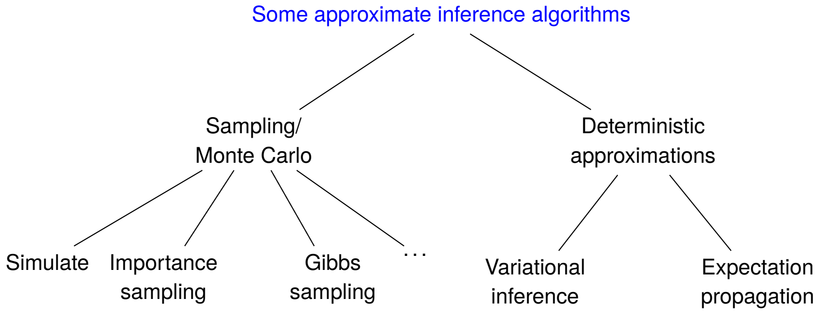
Graphical models: what we expect you to learn

- To pass this course, you do **not** have to know all technical details about graphical models by heart (sv. utantill).
- But you **do** need to know the big picture: how the different types relate to each other, what type of independence claims that can be made, etc.

UPPSALA UNIVERSITET

Summary of lecture 3–6: Approximate inference

1. The systematic tool for inference in graphical models is belief propagation (message passing).
2. If you end up with mathematical expressions you can handle: You're done!
3. If not, you have to do approximate inference.



Approximate inference: what we expect you to learn

- To pass this course, you **do not** have to be an expert in approximate inference
- But you **do** need to know the big picture and how to apply approximate inference for the mini project (or a similar problem) using available aids (course literature, problems from exercise sessions, . . .)

UPPSALA UNIVERSITET

Outlook: Probabilistic programming

In the future, you *might* not have to care about doing approximate inference yourself!
In the research field **probabilistic programming**, the goal is to let the user specify the model and *let the program automatically derive an appropriate inference scheme*.

There are many probabilistic programming languages, each with its own characteristics: **BUGS**, **STAN**, **Anglican**, **WebPPL**, **Edward**, **Pyro**, **Birch** (example $\rightarrow$), ...

$$\sigma^2 \sim \mathcal{IG}(3, 4/10)$$
$$\beta \sim \mathcal{N}(0, I\sigma^2)$$
$$y_n \sim \mathcal{N}(\mathbf{x}_n^\mathsf{T}\beta, \sigma^2)$$

```
class LinearRegressionModel < Model {
  X:Real[_,_];
  β:Random<Real[_]>;
  σ2:Random<Real>;
  y:Random<Real[_]>;
  P:Integer <- columns(X);

  σ2 ~ InverseGamma(3.0, 0.4);
  β ~ Gaussian(vector(0.0, P), identity(P)*σ2);
  y ~ Gaussian(X*β, σ2);
}

birch sample \
  -model LinearRegressionModel \
  -input datafile.dat \
  -output results.dat
```

Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, Frank Wood. **An Introduction to Probabilistic Programming**. *ArXiv:1809.10756*, 2018.
Lawrence Murray and Thomas B. Schön. **Automated learning with a probabilistic programming language: Birch**. *Annual Reviews in Control*, 46:29–43, 2018.

Today: Gaussian processes

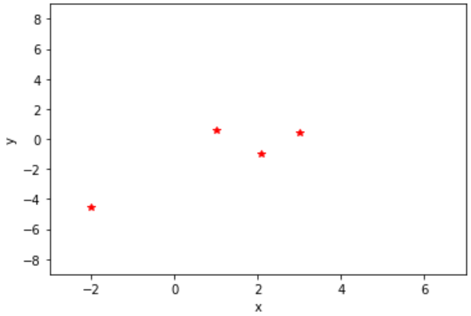
The plan for today:

1. **An introduction to Gaussian processes**
2. **Another introduction to Gaussian processes**

The plan for next lecture:
Using Gaussian processes in practice

UPPSALA UNIVERSITET

Reminder: The regression problem



Problem: Learn a model from data for how the output y depends on the input $\mathbf{x}$, say $f(\mathbf{x})$.

We will now see what it means to use the **Gaussian process** as a regression model.

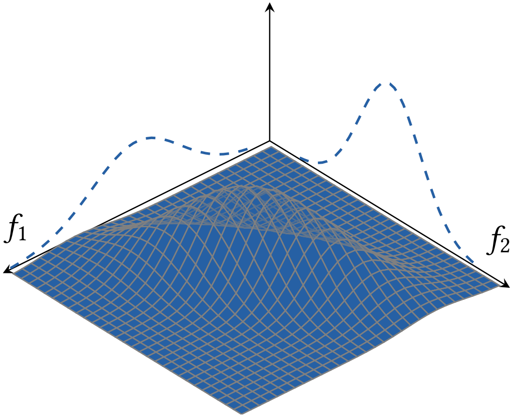
An introduction to Gaussian Processes

UPPSALA UNIVERSITET

A binary input

If $x \in \{1, 2\}$, we only have to find a model for $f(1)$ and $f(2)$.

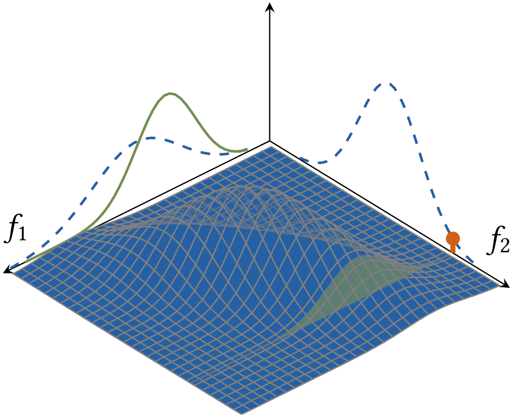
Why not a multivariate normal? (We have to estimate its parameters somehow, let's talk about that later.)



$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix} \right)$$

A binary input

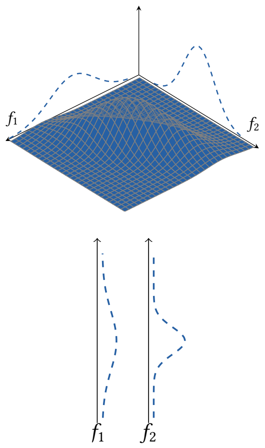
If the training data contains an observation of f_2, then our multivariate normal will automatically give us an updated prediction of f_1 as $p(f_1 \mid f_2)$ (thm 2, lecture 2)

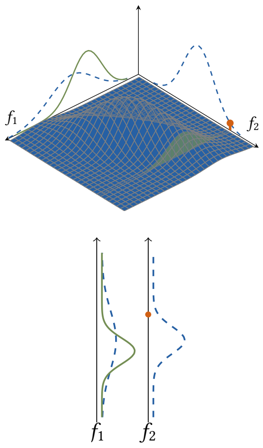


$$f_1 \mid f_2 \sim \mathcal{N}\left(\mu_1 + \frac{\sigma_{12}}{\sigma_2^2}(f_2 - \mu_2), \sigma_1^2 - \frac{\sigma_{12}\sigma_{21}}{\sigma_2^2}\right)$$

A binary input

Another way to illustrate this is to plot only the marginal distributions:





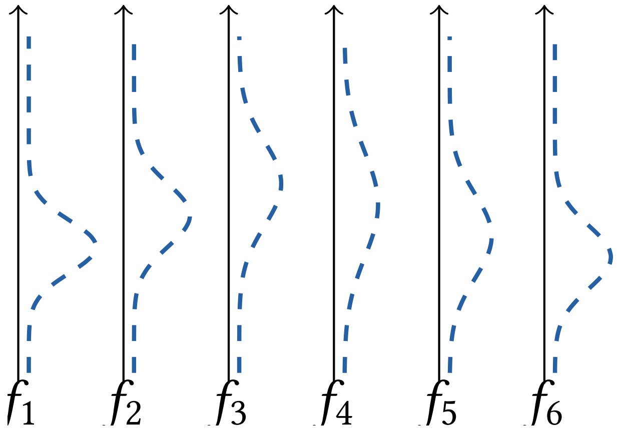
$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix} \right)$$

$$f_1 \mid f_2 \sim \mathcal{N}\left(\mu_1 + \frac{\sigma_{12}}{\sigma_2^2}(f_2 - \mu_2), \sigma_1^2 - \frac{\sigma_{12}\sigma_{21}}{\sigma_2^2} \right)$$

A discrete input

Now, if $x \in \{1, 2, 3, 4, 5, 6\}$, we can do the same thing. With $\mathbf{f} = [f_1\ f_2\ f_3\ f_4\ f_5\ f_6]^\mathsf{T}$, we assume

$$\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$



UPPSALA UNIVERSITET

A discrete input

If we block $\mathbf{f}$ in two parts $\mathbf{f} = \begin{bmatrix}\mathbf{f}_a \\ \mathbf{f}_b\end{bmatrix}$, we can write

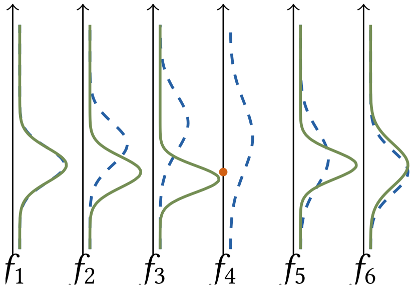
$$\begin{bmatrix}\mathbf{f}_a \\ \mathbf{f}_b\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb}\end{bmatrix}\right)$$

and get

$$\mathbf{f}_a \,|\, \mathbf{f}_b \sim \mathcal{N}\left(\boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\mathbf{f}_b - \boldsymbol{\mu}_b), \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}\right)$$

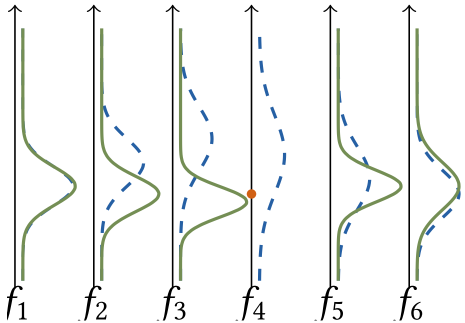
If we observe $\mathbf{f}_b$, we get an updated prediction for $\mathbf{f}_a$ as $p(\mathbf{f}_a \,|\, \mathbf{f}_b)$

For example, let $\mathbf{f}_b = f_4$.



UPPSALA UNIVERSITET

A discrete input



Can we generalize this idea to continuous inputs?
That is, $x \in \mathbb{R}$ instead of $x \in \{1, 2, 3, 4, 5, 6\}$?
Yes $\rightarrow$ the Gaussian Process!

UPPSALA UNIVERSITET

A discrete input

For the case of a finite set of input values $x \in 1, 2, \dots, n$, we can use the multivariate Gaussian as a model for $f(x)$.

$$\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1n} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \dots & \sigma_n^2 \end{bmatrix} \right)$$

(For simplicity, we use prior mean $\mu = 0$. Will still be a useful model.)

How do we generalize this to continuous inputs?

The Gaussian process

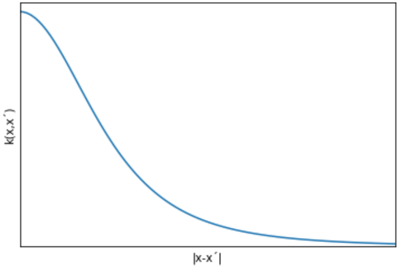
We have to introduce a *covariance function* $\kappa(x, x')$ such that

$$\begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} \kappa(x_1, x_1) & \kappa(x_1, x_2) & \dots & \kappa(x_1, x_n) \\ \kappa(x_2, x_1) & \kappa(x_2, x_2) & \dots & \kappa(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(x_n, x_1) & \kappa(x_n, x_2) & \dots & \kappa(x_n, x_n) \end{bmatrix} \right)$$

for *any* choice of $\{x_1, x_2, \dots, x_n\}$.

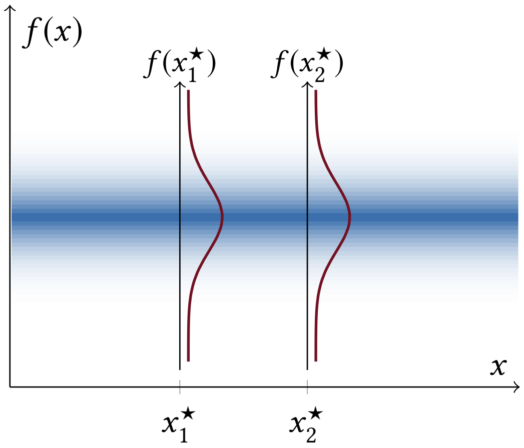
One choice of $\kappa(x, x')$, out of infinitely many, is

$$\kappa(x, x') = \left(1 + \frac{|x - x'|^2}{2\alpha\ell}\right)^{-\alpha},$$

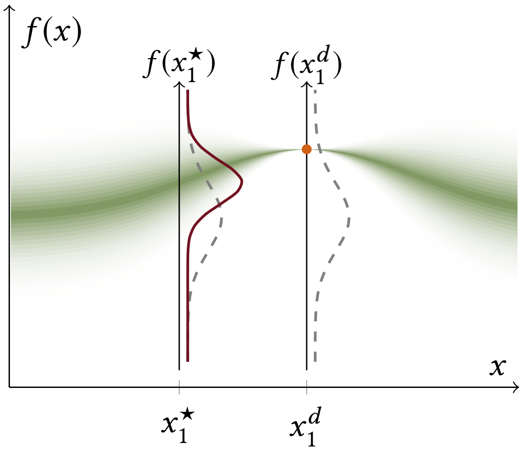


Given a $\kappa(x, x')$, everything follows as before.

The Gaussian process



The distribution for $f(x^\star)$ without any observations



The distribution for $f(x^\star)$ conditional on an observation of x_1^d

UPPSALA UNIVERSITET

The Gaussian process: Definition

A time continuous stochastic process $\{X_t; t \in T\}$ is a Gaussian process if and only if for every finite set of indices $t_1, \ldots, t_k$

$$\mathbf{X}_{t_1,\ldots,t_k} = (X_{t_1}, \ldots, X_{t_k})$$

is a multivariate Gaussian random variable.

The Gaussian process: The “core” equations

With $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$, $f(\mathbf{X}) = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix}$, $K(x_\star, x_\star) = \kappa(x_\star, x_\star)$,

$K(\mathbf{X}, x_\star) = \begin{bmatrix} \kappa(x_1, x_\star) \\ \vdots \\ \kappa(x_N, x_\star) \end{bmatrix} = K(x_\star, \mathbf{X})^\mathsf{T}$ and $K(\mathbf{X}, \mathbf{X}) = \begin{bmatrix} \kappa(x_1, x_1) & \dots & \kappa(x_1, x_N) \\ \vdots & & \vdots \\ \kappa(x_N, x_1) & \dots & \kappa(x_N, x_N) \end{bmatrix}$

we have

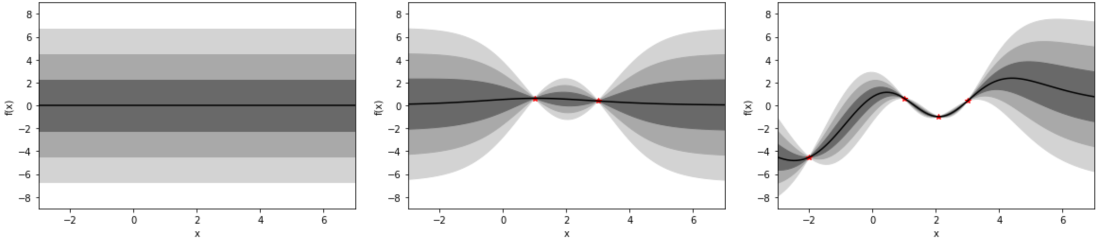
$$\begin{bmatrix} f(\mathbf{X}) \\ f(x_\star) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \begin{bmatrix} K(\mathbf{X}, \mathbf{X}) & K(x_\star, \mathbf{X}) \\ K(\mathbf{X}, x_\star) & K(x_\star, x_\star) \end{bmatrix} \right)$$

and most importantly

$$\boxed{f(x_\star) \,|\, f(\mathbf{X}) \sim \mathcal{N}\left(\mathbf{K}(x_\star, \mathbf{X}) K(\mathbf{X}, \mathbf{X})^{-1} f(\mathbf{X}), K(x_\star, x_\star) - K(x_\star, \mathbf{X}) K(\mathbf{X}, \mathbf{X})^{-1} K(\mathbf{X}, x_\star)\right)}$$

The Gaussian process as a regression model

$$f(x_\star) \,|\, f(\mathbf{X}) \sim \mathcal{N}\left(\mathbf{K}(x_\star, \mathbf{X}) K(\mathbf{X}, \mathbf{X})^{-1} f(\mathbf{X}), K(x_\star, x_\star) - K(\mathbf{X}, x_\star) K(\mathbf{X}, \mathbf{X})^{-1} K(\mathbf{X}, x_\star)\right)$$

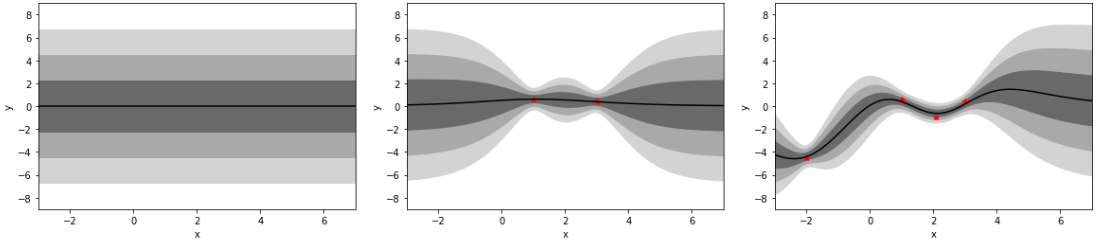


UPPSALA UNIVERSITET

The Gaussian process

But what if we don't observe $f(x)$ exactly, but observe $f(x) + \varepsilon$, with $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$?

$$f(x_\star) \,|\, \mathbf{y} \sim \mathcal{N}\left(\mathbf{K}(x_\star, \mathbf{X})(K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I})^{-1} \mathbf{y}, K(x_\star, x_\star) - K(x_\star, \mathbf{X})(K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I})^{-1} K(\mathbf{X}, x_\star)\right)$$

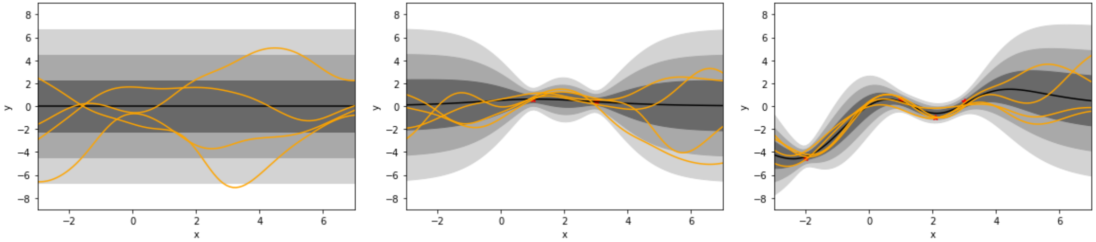


UPPSALA UNIVERSITET

Samples from the Gaussian process

We can also draw samples from $p(f(x_\star) \,|\, \mathbf{y})$

$$f(x_\star) \,|\, \mathbf{y} \sim \mathcal{N}\left(\mathbf{K}(x_\star, \mathbf{X})(K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I})^{-1}\mathbf{y}, K(x_\star, x_\star) - K(x_\star, \mathbf{X})(K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I})^{-1} K(\mathbf{X}, x_\star)\right)$$



Here, $x_\star$ is a vector with one element for each pixel on the screen $\rightarrow$ the sample looks continuous!
The Gaussian process can be understood as a *distribution over functions*.

http://www.it.uu.se/edu/course/homepage/apml/GP/

Another introduction to Gaussian Processes

UPPSALA UNIVERSITET

Bayesian linear regression again

- $y = \underbrace{\mathbf{x}^\mathsf{T}\mathbf{w}}_{f(\mathbf{x})} + \varepsilon,\ \varepsilon \sim \mathcal{N}(0, \sigma_n^2)$ (We consider σ_n^2 as a hyperparameter; not to be inferred)

  $\Rightarrow p(\mathbf{y} \,|\, \mathbf{w}) = \mathcal{N}(\mathbf{y}; \mathbf{X}\mathbf{w}, \sigma_n^2 \mathbf{I}_N)$

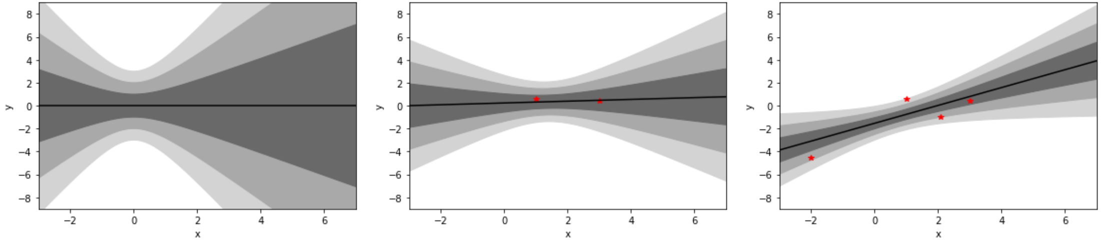
- $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \mathbf{I})$

  $\Rightarrow p(\mathbf{w} \,|\, \mathbf{y}) = \mathcal{N}\left(\mathbf{w}; \left(\mathbf{I} + \frac{1}{\sigma_n^2}\mathbf{X}^\mathsf{T}\mathbf{X}\right)^{-1}\left(\frac{1}{\sigma_n^2}\mathbf{X}^\mathsf{T}\mathbf{y}\right), \left(\mathbf{I} + \frac{1}{\sigma_n^2}\mathbf{X}^\mathsf{T}\mathbf{X}\right)^{-1}\right)$

  $\Rightarrow p(f(\mathbf{x}_\star) \,|\, \mathbf{y}) = \mathcal{N}\left(f(\mathbf{x}_\star); \mathbf{x}_\star^\mathsf{T}\left(\mathbf{I} + \frac{1}{\sigma_n^2}\mathbf{X}^\mathsf{T}\mathbf{X}\right)^{-1}\left(\frac{1}{\sigma_n^2}\mathbf{X}^\mathsf{T}\mathbf{y}\right), \mathbf{x}_\star^\mathsf{T}\left(\mathbf{I} + \frac{1}{\sigma_n^2}\mathbf{X}^\mathsf{T}\mathbf{X}\right)^{-1}\mathbf{x}_\star\right)$

$\mathbf{x}_\star$ is an arbitrary *test input*

Bayesian linear regression again



$$p(f(\mathbf{x}_\star)\,|\,\mathbf{y}) = \mathcal{N}\left(f(\mathbf{x}_\star); \underbrace{\mathbf{x}_\star^\mathsf{T}\left(\mathbf{I}+\frac{1}{\sigma_n^2}\mathbf{X}\mathbf{X}^\mathsf{T}\right)^{-1}\left(\frac{1}{\sigma_n^2}\mathbf{X}^\mathsf{T}\mathbf{y}\right)}_{\text{Posterior mean; black line}}, \underbrace{\mathbf{x}_\star^\mathsf{T}\left(\mathbf{I}+\frac{1}{\sigma_n^2}\mathbf{X}\mathbf{X}^\mathsf{T}\right)^{-1}\mathbf{x}_\star}_{\text{Posterior variance; gray areas}}\right)$$

Red dots = observed data $\{\mathbf{x}_i, y_i\}_{i=1}^N$

UPPSALA UNIVERSITET

Bayesian linear regression again

And what if we use some non-linear input transformation? Like

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix} \quad ?$$

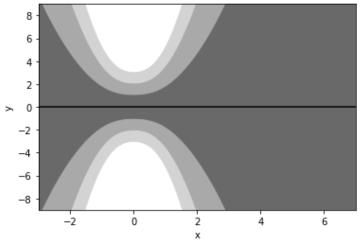
$$p(f(x_\star) \,|\, \mathbf{y}) =$$
$$\mathcal{N}\left(f(x_\star); \phi(x_\star)^\mathsf{T} \left(\mathbf{I} + \frac{1}{\sigma_n^2}\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi}\right)^{-1} \left(\frac{1}{\sigma_n^2}\mathbf{\Phi}^\mathsf{T}\mathbf{y}\right), \phi(x_\star)^\mathsf{T} \left(\mathbf{I} + \frac{1}{\sigma_n^2}\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi}\right)^{-1} \phi(x_\star)\right)$$

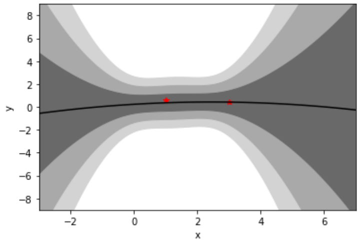
$$\mathbf{\Phi} = \begin{bmatrix} \phi(x_1)^\mathsf{T} \\ \phi(x_2)^\mathsf{T} \\ \vdots \\ \phi(x_N)^\mathsf{T} \end{bmatrix}$$

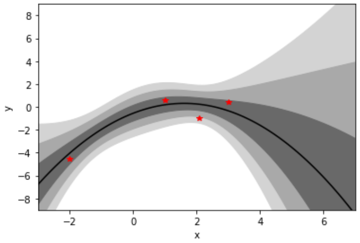
Bayesian linear regression again

With

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}$$







UPPSALA UNIVERSITET

Bayesian linear regression again

$$p(f(x_\star)\,|\,\mathbf{y}) = \mathcal{N}\left(f(x_\star); \phi(x_\star)^\mathsf{T}\left(\sigma_n^2\mathbf{I} + \mathbf{\Phi}^\mathsf{T}\mathbf{\Phi}\right)^{-1}\left(\mathbf{\Phi}^\mathsf{T}\mathbf{y}\right), \phi(x_\star)^\mathsf{T}\left(\mathbf{I} + \frac{1}{\sigma_n^2}\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi}\right)^{-1}\phi(x_\star)\right)$$

For any matrix $\mathbf{A}$, $(\mathbf{I} + \mathbf{A}^\mathsf{T}\mathbf{A})^{-1}\mathbf{A} = \mathbf{A}(\mathbf{I} + \mathbf{A}\mathbf{A}^\mathsf{T})^{-1}$. Hence,

$$\phi(x_\star)^\mathsf{T}\left(\sigma_n^2\mathbf{I} + \mathbf{\Phi}^\mathsf{T}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{y} = \phi(x_\star)^\mathsf{T}\mathbf{\Phi}^\mathsf{T}\left(\sigma_n^2\mathbf{I} + \mathbf{\Phi}\mathbf{\Phi}^\mathsf{T}\right)^{-1}\mathbf{y}$$

The matrix inversion lemma $(\mathbf{I} - \mathbf{U}\mathbf{V})^{-1} = \mathbf{I} - \mathbf{U}(\mathbf{I} + \mathbf{V}\mathbf{U})^{-1}\mathbf{V}$ gives

$$\phi(x_\star)^\mathsf{T}\left(\mathbf{I} + \frac{1}{\sigma_n^2}\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi}\right)^{-1}\phi(x_\star) = \phi(x_\star)^\mathsf{T}\phi(x_\star) - \phi(x_\star)^\mathsf{T}\mathbf{\Phi}^\mathsf{T}\left(\sigma_n^2\mathbf{I} + \mathbf{\Phi}\mathbf{\Phi}^\mathsf{T}\right)^{-1}\mathbf{\Phi}\phi(x_\star)$$

Kernels

Let $\kappa(x, x') = \phi(x)^\mathsf{T}\phi(x')$, we refer to $\kappa(\cdot, \cdot)$ as a **kernel**.

$$K(x_\star, x_\star) = \kappa(x_\star, x_\star) = \phi(x_\star)^\mathsf{T}\phi(x_\star)$$

$$K(\mathbf{X}, x_\star) = \begin{bmatrix} \kappa(x_1, x_\star) \\ \vdots \\ \kappa(x_N, x_\star) \end{bmatrix} = \begin{bmatrix} \phi(x_1)^\mathsf{T}\phi(x_\star) \\ \vdots \\ \phi(x_N)^\mathsf{T}\phi(x_\star) \end{bmatrix} = \mathbf{\Phi}\phi(x_\star) = K(x_\star, \mathbf{X})^\mathsf{T}$$

$$K(\mathbf{X}, \mathbf{X}) = \begin{bmatrix} \kappa(x_1, x_1) & \dots & \kappa(x_1, x_N) \\ \vdots & & \vdots \\ \kappa(x_N, x_1) & \dots & \kappa(x_N, x_N) \end{bmatrix} = \begin{bmatrix} \phi(x_1)^\mathsf{T}\phi(x_1) & \dots & \phi(x_1)^\mathsf{T}\phi(x_N) \\ \vdots & & \vdots \\ \phi(x_N)^\mathsf{T}\phi(x_1) & \dots & \phi(x_N)^\mathsf{T}\phi(x_N) \end{bmatrix} = \mathbf{\Phi}\mathbf{\Phi}^\mathsf{T}$$

The kernel trick

$$p(f(x_\star)\,|\,\mathbf{y}) = \mathcal{N}\Big(f(x_\star); K(x_\star, \mathbf{X})\left(\sigma_n^2\mathbf{I} + K(X,X)\right)^{-1}\mathbf{y},$$
$$K(x_\star, x_\star) - K(x_\star, \mathbf{X})\left(\sigma_n^2\mathbf{I} + K(X,X)\right)^{-1}K(\mathbf{X}, x_\star)\Big)$$

The input x only appears in $p(f(x_\star)\,|\,\mathbf{y})$ via the kernel $\kappa(x, x') = \boldsymbol{\phi}(x)^\mathsf{T}\boldsymbol{\phi}(x')$

The kernel trick: Do not compute (or even choose!) the nonlinear transformations $\boldsymbol{\phi}(x)$. Work directly with $\kappa(x, x')$ instead.

The kernel trick

The kernel trick: Do not compute (or even choose!) the nonlinear transformations $\phi(x)$. Work directly with $\kappa(x, x')$ instead.

Only requirement on $\kappa(x, x')$: $K(X, X)$ has to be positive semidefinite for all possible values on X.

One possible choice (out of infinitely many) is

$$\kappa(x, x') = \left(1 + \frac{|x - x'|^2}{2\alpha\ell}\right)^{-\alpha},$$

the rational quadratic kernel.

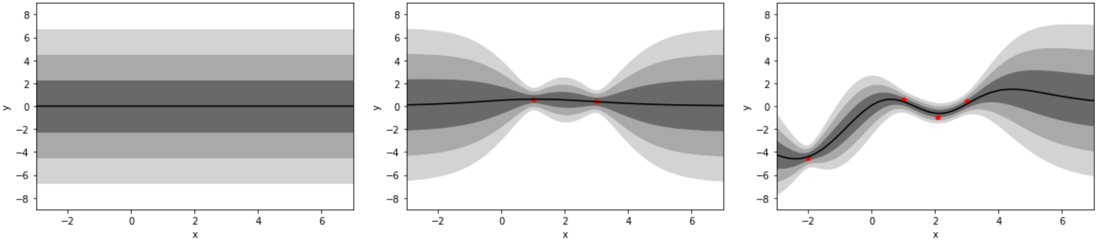
For any kernel $\kappa(\cdot, \cdot)$, a corresponding (possibly infinite) nonlinear feature transformation $\phi(\cdot)$ can be constructed, which lives in a *reproducing kernel Hilbert space*.

Gaussian process regression

$$p(f(x_\star) \,|\, \mathbf{y}) = \mathcal{N}\Big(f(x_\star); K(x_\star, \mathbf{X}) \left(\sigma_n^2 \mathbf{I} + K(X, X)\right)^{-1} \mathbf{y},$$

$$K(x_\star, x_\star) - K(x_\star, \mathbf{X}) \left(\sigma_n^2 \mathbf{I} + K(X, X)\right)^{-1} K(\mathbf{X}, x_\star)\Big)$$

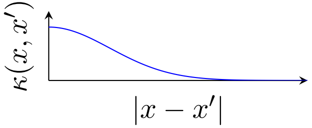
$$\kappa(x, x') = \left(1 + \frac{|x - x'|^2}{2\alpha\ell}\right)^{-\alpha},$$



Kernel = covariance function

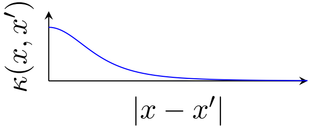
Squared exponential/RBF

$$\kappa(x,x') = \sigma^2 \exp(-\tfrac{1}{2\ell^2}(x-x')^2)$$



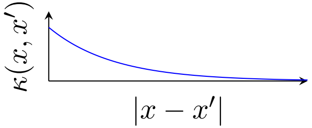
Rational quadratic

$$\kappa(x,x') = \sigma^2 \left(1 + \tfrac{|x-x'|^2}{2\alpha\ell}\right)^{-\alpha}$$



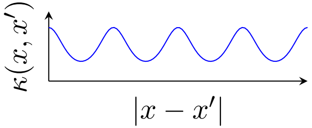
Matérn 1

$$\kappa(x,x') = \sigma^2 \exp(-\tfrac{1}{\ell^2}|x-x'|)$$



Periodic kernel

$$\kappa(x,x') = \sigma^2 \exp(-\tfrac{2}{\ell^2}\sin^2(\pi\tfrac{|x-x'|}{p}))$$

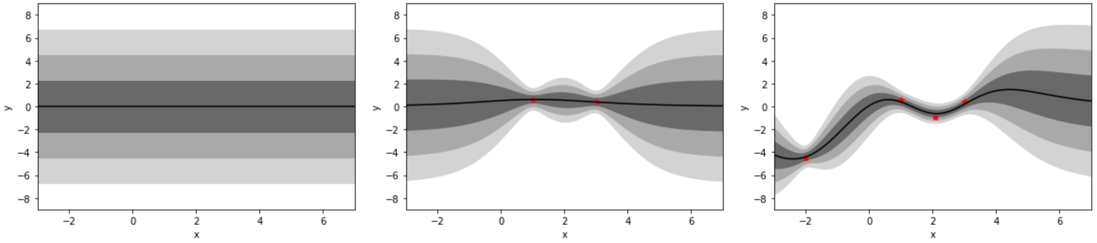


The Gaussian process model: summary

A Bayesian/probabilistic non-parametric model for regression

- Bayesian/probabilistic: The predictions $f(x_\star) \,|\, \mathbf{y}$ are not points, but distributions.
- Nonparametric: The predictions $f(x_\star) \,|\, \mathbf{y}$ depends on all observed data, and not just a fixed set of parameters

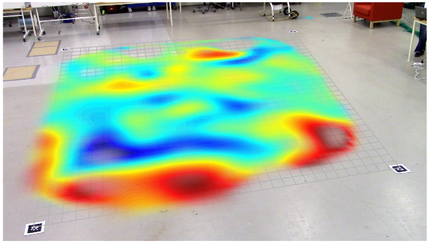
$$f(x_\star) \,|\, \mathbf{y} \sim \mathcal{N}\big(\underbrace{K(x_\star, \mathbf{X})(K(\mathbf{X},\mathbf{X}) + \sigma_n^2 \mathbf{I})^{-1}\mathbf{y}}_{\text{Predictive posterior mean}}, \underbrace{K(x_\star, x_\star) - K(x_\star, \mathbf{X})(K(\mathbf{X},\mathbf{X}) + \sigma_n^2 \mathbf{I})^{-1} K(\mathbf{X}, x_\star)}_{\text{Predictive posterior variance}} \big)$$



Most kernels contain hyperparameters $\sigma, \ell, \ldots$. How to choose them in practice? $\rightarrow$ next lecture

Research example – Building magnetic field maps using GP

Problem: Build a map of the indoor magnetic field using Gaussian processes.



`https://www.youtube.com/watch?v=enlMiUqPVJo`

Color = posterior mean (strength of magnetic field; the quantity of interest)

Opacity = posterior variance (more transparent means more variance)

[1] Niklas Wahlström, Manon Kok, Thomas B. Schön and Fredrik Gustafsson, **Modeling magnetic fields using Gaussian processes** *The 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, 2013.
[2] Arno Solin, Manon Kok, Niklas Wahlström, Thomas B. Schön and Simo Särkkä, **Modeling and interpolation of the ambient magnetic field by Gaussian processes** *IEEE Transactions on Robotics*, Volume: 34 Issue: 4, 2018

A few concepts to summarize lecture 7

- Gaussian processes as a generalization of the multidimensional Gaussian distribution
- Gaussian processes as a generalization of Bayesian linear regression
- Covariance functions/kernels