Statistical Machine Learning

Lecture 13: Kernel Regression and Gaussian Processes

Kristian Kersting
TU Darmstadt

Summer Term 2020

Today's Objectives

- Make you understand how to use kernels for regression both from a frequentist and Bayesian point of view
- Covered Topics
  - Why kernel methods?
  - Radial basis function networks
  - What is a kernel?
  - Dual representation
  - Gaussian Process Regression

Outline

1. Kernel Methods for Regression

2. Gaussian Processes Regression

3. Bayesian Learning and Hyperparameters

4. Wrap-Up

TECHNISCHE UNIVERSITÄT DARMSTADT

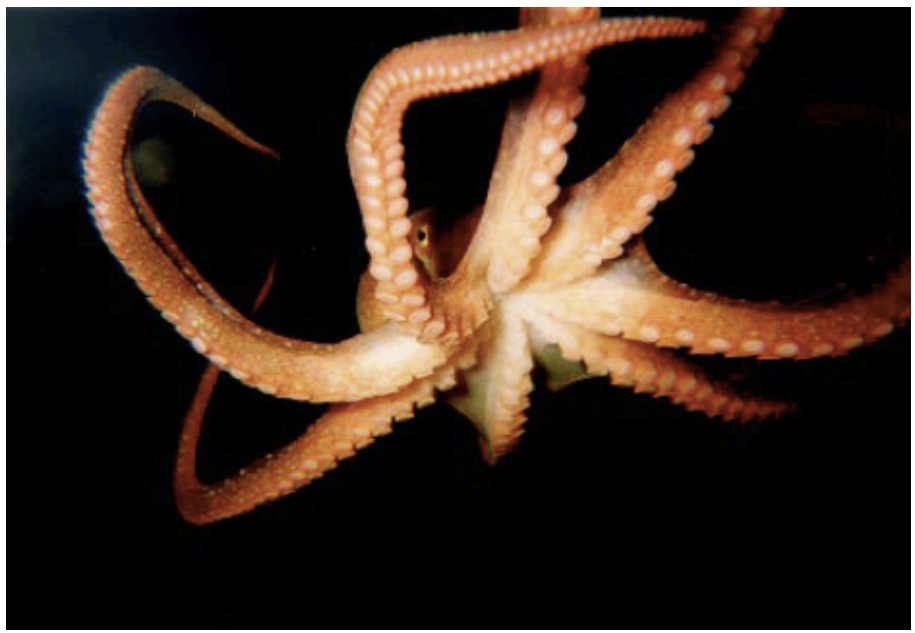
Outline

TECHNISCHE UNIVERSITÄT DARMSTADT

Why Kernels and not Neural Networks?

- Multi-Layer Perceptrons use univariate projections to "span" the space of the data (like an "octopus")

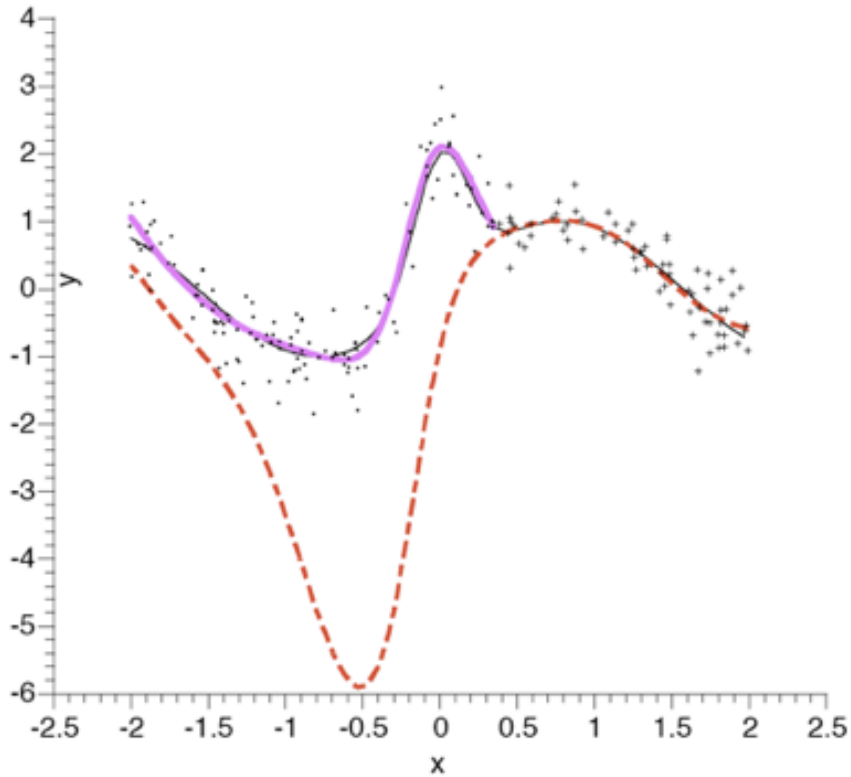
$$y = g\left(\boldsymbol{w}^\mathsf{T}\boldsymbol{x}\right)$$



TECHNISCHE UNIVERSITÄT DARMSTADT

Why Kernels and not Neural Networks?

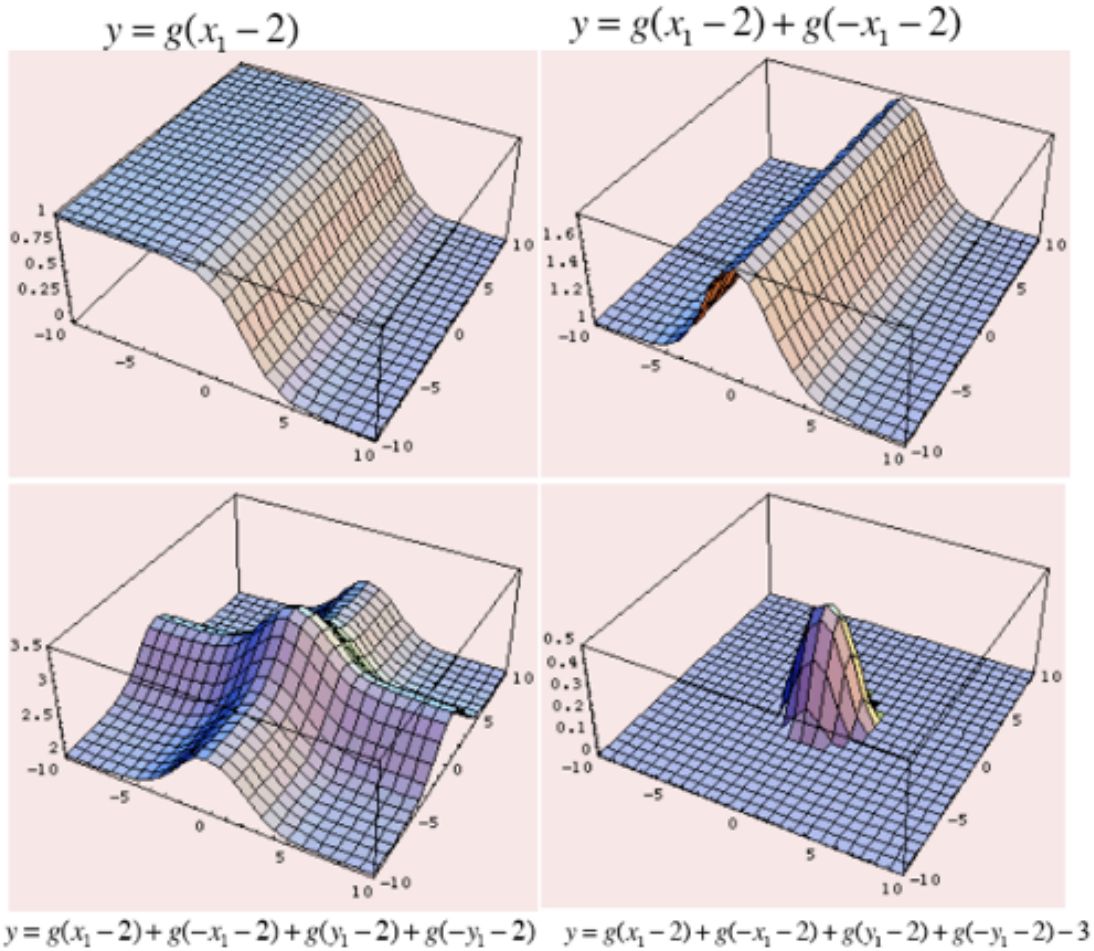
- Pros
  - Universal function approximation
  - Large range generalization (extrapolation)
  - Good for high dimensional data
- Cons
  - Hard to train
  - Danger of interference



TECHNISCHE UNIVERSITÄT DARMSTADT

Radial Basis Function Networks

- Use spatially localized kernels for learning
- Note: there are other basis functions that are not spatially localized

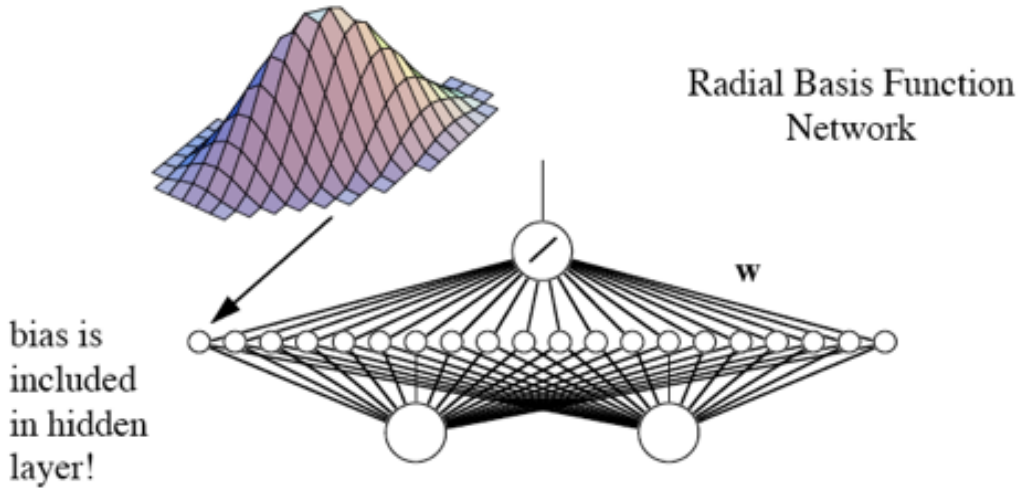


TECHNISCHE UNIVERSITÄT DARMSTADT

Radial Basis Function Networks

- For instance with Gaussian kernels

$$y = \sum_{i=1}^{K} w_i \phi(\|\mathbf{x} - \mathbf{x}_k\|) = \mathbf{w}^T \Phi(\|\mathbf{x} - \mathbf{x}_k\|)$$



$$\phi(\boldsymbol{x}, \boldsymbol{c}_k) = \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{c}_k)^\mathsf{T} \boldsymbol{D} (\boldsymbol{x} - \boldsymbol{c}_k)\right)$$

with $\boldsymbol{D}$ positive definite

TECHNISCHE UNIVERSITÄT DARMSTADT

Radial Basis Function Networks

- The "output layer" is just a linear regression
- Often needs regularization (e.g., ridge regression)

$$J = \frac{1}{2}(\boldsymbol{t} - \boldsymbol{y})^\mathsf{T}(\boldsymbol{t} - \boldsymbol{y}) = \frac{1}{2}(\boldsymbol{t} - \boldsymbol{\Phi}\boldsymbol{w})^\mathsf{T}(\boldsymbol{t} - \boldsymbol{\Phi}\boldsymbol{w})$$

$$\boldsymbol{t} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_n \end{bmatrix}, \quad \boldsymbol{\Phi} = \begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1m} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nm} \end{bmatrix}$$

$$\boldsymbol{w} = (\boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^\mathsf{T}\boldsymbol{t}$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Radial Basis Function Networks

- The "input layer" can be optimized by gradient descent with respect to distance metric and centers of RBFs

$$\frac{\partial J}{\partial \boldsymbol{c}_k} = (t-y)\left(-\frac{\partial y}{\partial \boldsymbol{c}_k}\right) = -(t-y)\, w_k \left(\frac{\partial \Phi}{\partial \boldsymbol{c}_k}\right)$$

$$\frac{\partial \Phi}{\partial \boldsymbol{c}_k} = \frac{\partial}{\partial \boldsymbol{c}_k} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{c}_k)^\mathsf{T} \boldsymbol{D} (\boldsymbol{x}-\boldsymbol{c}_k)\right)$$

$$= \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{c}_k)^\mathsf{T} \boldsymbol{D} (\boldsymbol{x}-\boldsymbol{c}_k)\right)(\boldsymbol{x}-\boldsymbol{c}_k)^\mathsf{T} \boldsymbol{D}$$

$$\frac{\partial J}{\partial \boldsymbol{D}_k} = (t-y)\left(-\frac{\partial y}{\partial \boldsymbol{D}_k}\right) = -(t-y)\, w_k - \frac{\partial \Phi}{\partial \boldsymbol{D}_k}$$

$$\frac{\partial \Phi}{\partial \boldsymbol{D}_k} = \frac{\partial}{\partial \boldsymbol{D}_k} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{c}_k)^\mathsf{T} \boldsymbol{D}_k (\boldsymbol{x}-\boldsymbol{c}_k)\right)$$

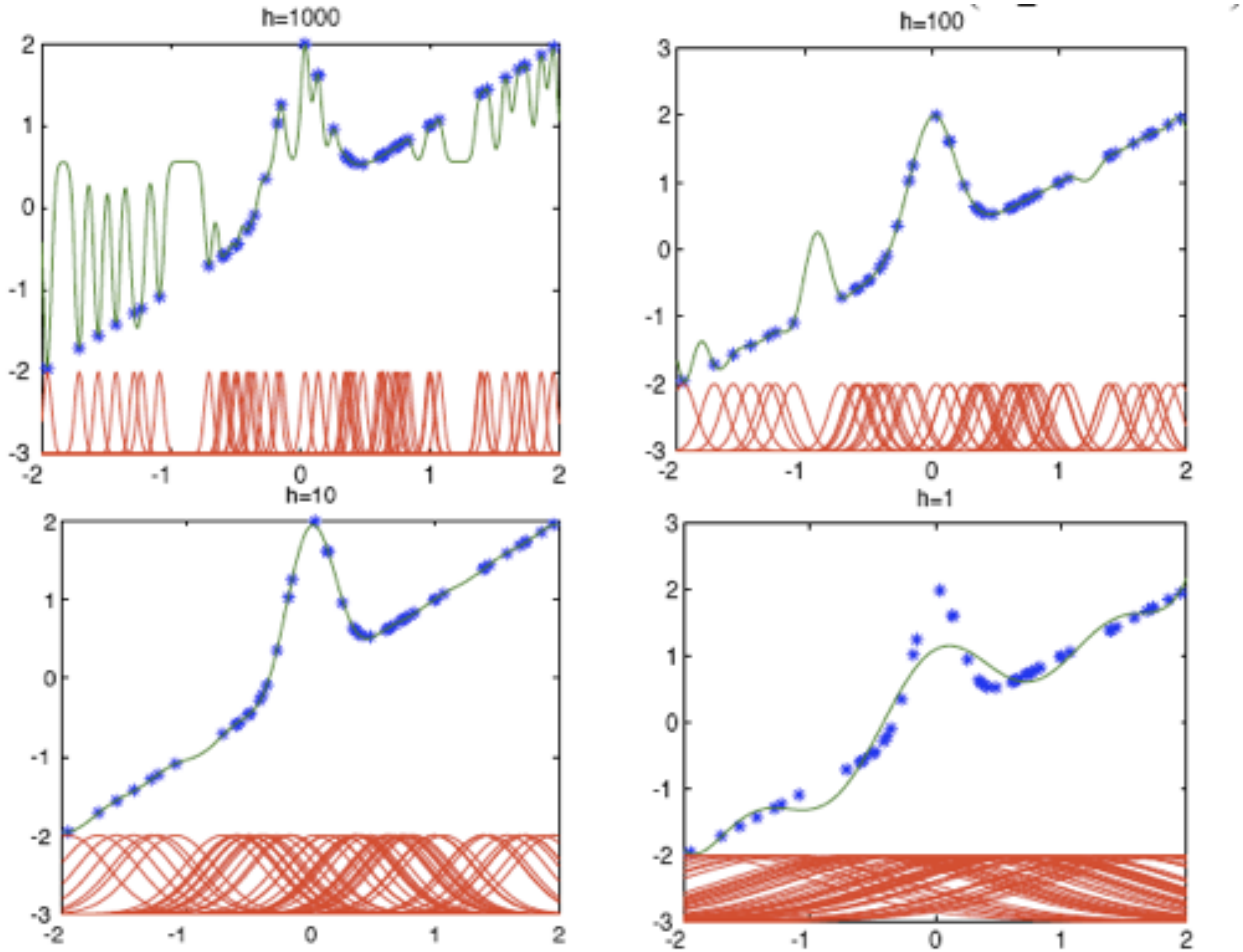
$$= \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{c}_k)^\mathsf{T} \boldsymbol{D}_k (\boldsymbol{x}-\boldsymbol{c}_k)\right)(\boldsymbol{x}-\boldsymbol{c}_k)^\mathsf{T} (\boldsymbol{x}-\boldsymbol{c}_k)$$

- Gradient descent can make $\boldsymbol{D}$ non positive definite $\Longrightarrow$ use Cholesky Decomposition
- An iterative procedure is needed to for optimization, i.e., alternate update of $\boldsymbol{w}$ and update of $\boldsymbol{c}_k$ and $\boldsymbol{D}_k$

TECHNISCHE UNIVERSITÄT DARMSTADT

Radial Basis Function Networks

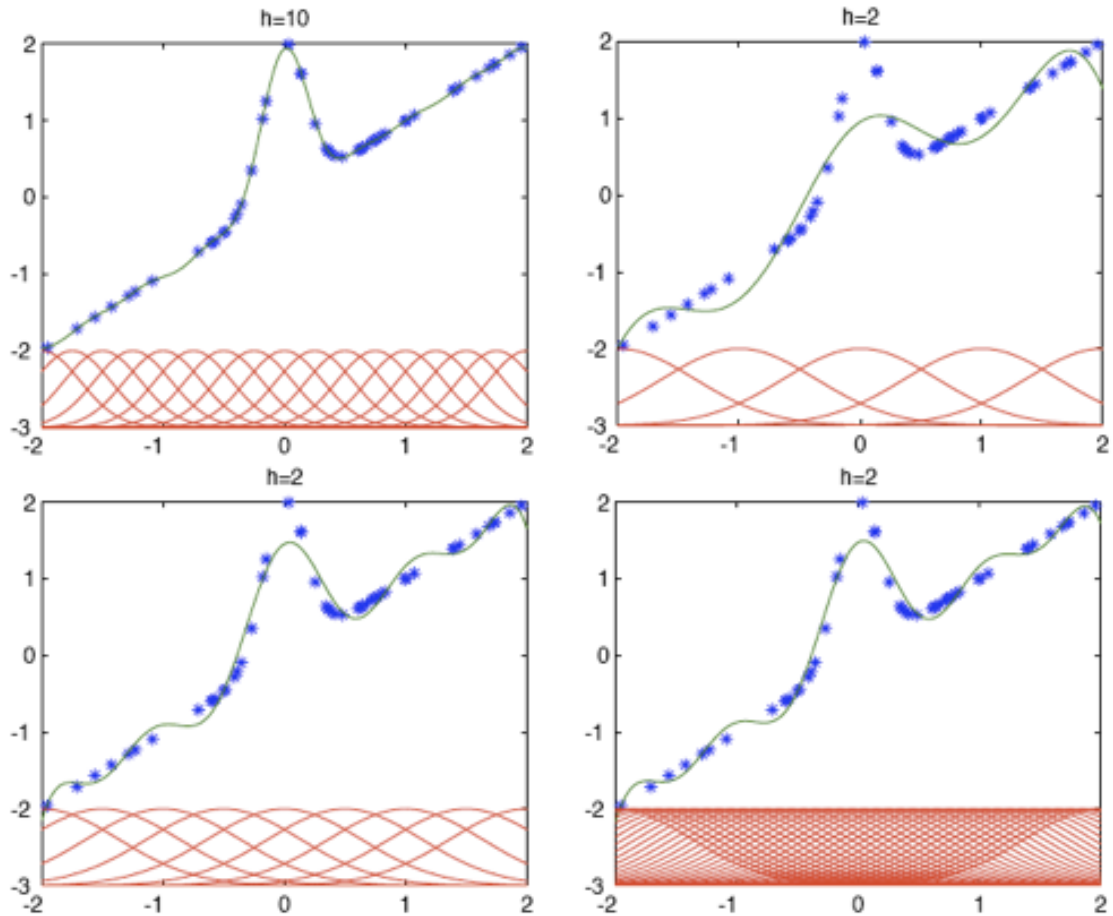
- Sensitivity to kernel width (bandwidth, dist. metric) of $\phi(x, c_k) = \exp\left(-\frac{1}{2}(x - c_k)^2 h\right)$



TECHNISCHE UNIVERSITÄT DARMSTADT

Radial Basis Function Networks

- Sensitivity to number of kernels and metric of
  $\phi(x, c_k) = \exp\left(-\frac{1}{2}(x - c_k)^2 h\right)$

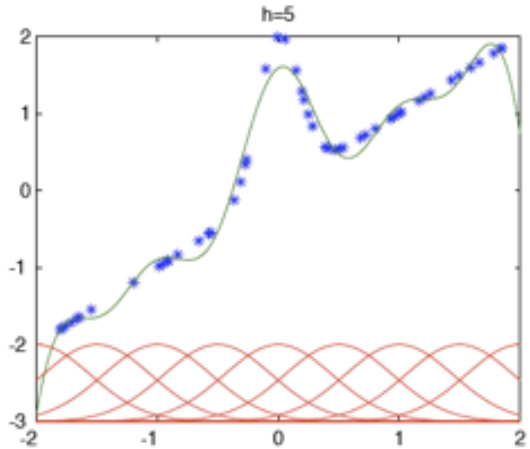


TECHNISCHE UNIVERSITÄT DARMSTADT

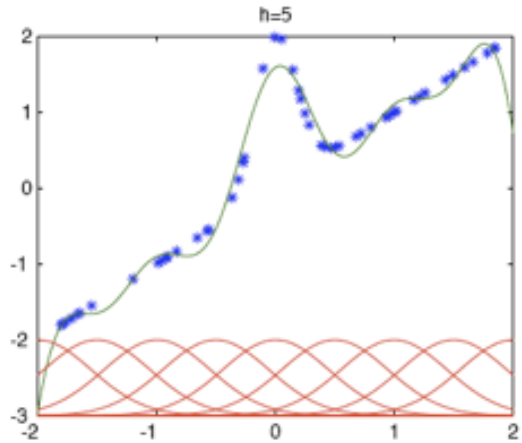
Radial Basis Function Networks

- Benefits of center and metric adaptation

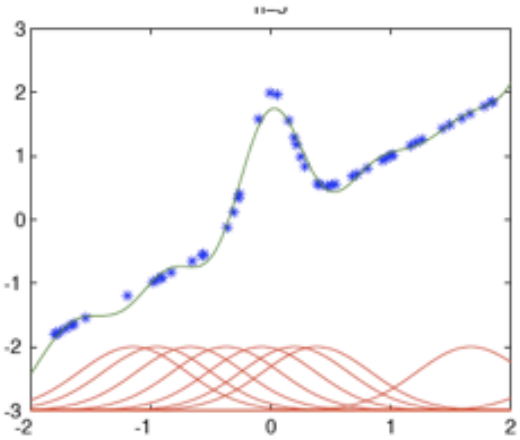
No Center Adaptation: MSE=0.022



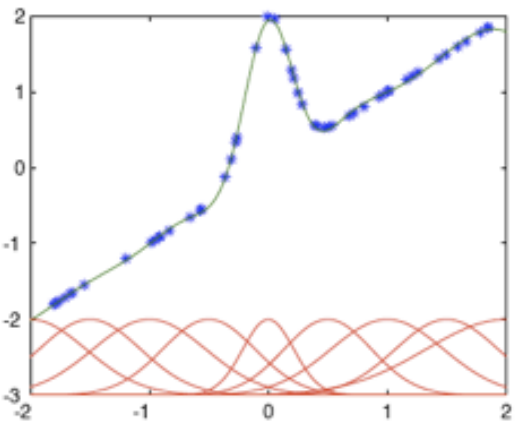
No Adaptation: MSE=0.022



With Center Adaptation: MSE=0.01



Distance Metric Adaptation: MSE=0.0001

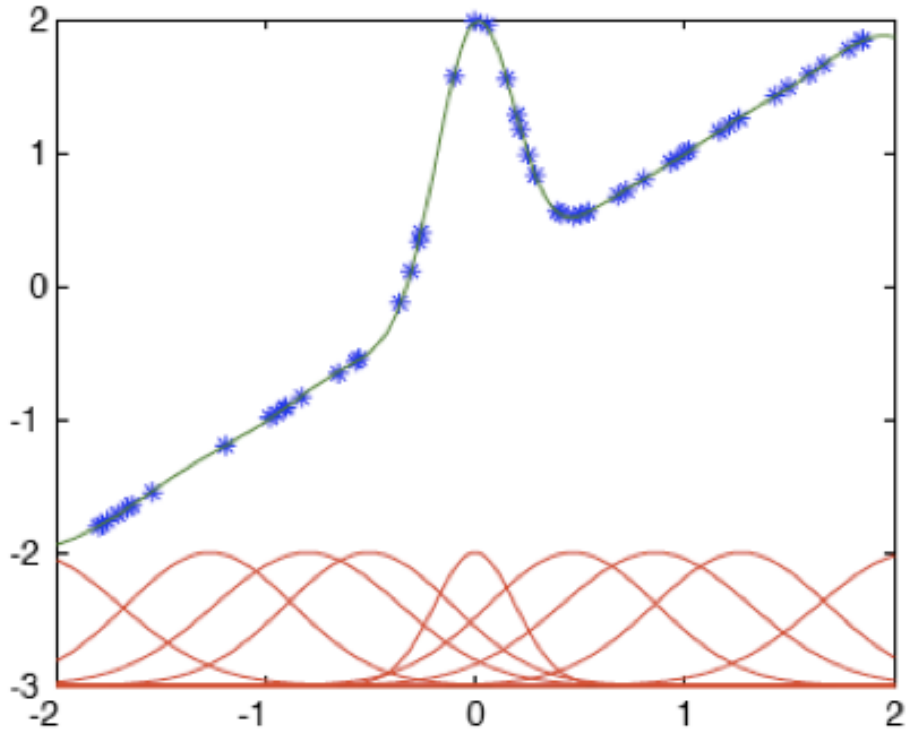


TECHNISCHE UNIVERSITÄT DARMSTADT

Radial Basis Function Networks

- All adaptations turned on
- Note: RBF tend to grow wider with a lot of overlap, and learning rates are sensitive

Distance Metric and Center Adaptation: MSE=0.00001



TECHNISCHE UNIVERSITÄT DARMSTADT

Radial Basis Function Networks - Summary

- RBFs are a powerful and efficient learning tool
- Number of RBFs and hyperparameter optimization is important and a bit difficult to tune
- Theoretical remark
  - Poggio and Girosi (1990) showed that RBF networks arise naturally from minimizing the penalized cost function

$$J = \frac{1}{2}\sum_n \left(t^n - y\left(x^n\right)\right)^2 + \frac{1}{2}\gamma \int \left|G\left(\boldsymbol{x}\right)\right|^2 \mathrm{d}\boldsymbol{x}$$

with, e.g., $G\left(\boldsymbol{x}\right) = \frac{\partial^2 y}{\partial \boldsymbol{x}^2}$, a smoothless prior

Kernel Methods in General

- What is a kernel?
  - Most intuitive approach for a fixed nonlinear feature space: an inner product of feature vectors

$$k\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \phi\left(\boldsymbol{x}\right)^{\mathsf{T}} \phi\left(\boldsymbol{x}'\right)$$

- A kernel is symmetric

$$k\left(\boldsymbol{x}, \boldsymbol{x}'\right) = k\left(\boldsymbol{x}', \boldsymbol{x}\right)$$

- Examples
  - Stationary kernels: $k\left(\boldsymbol{x}, \boldsymbol{x}'\right) = k\left(\boldsymbol{x} - \boldsymbol{x}'\right)$
  - Linear kernel: $k\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{x}'$
  - Homogeneous kernels: $k\left(\boldsymbol{x}, \boldsymbol{x}'\right) = k\left(\|\boldsymbol{x} - \boldsymbol{x}'\|\right)$

Dual Representation of Linear Regression

- The dual representation gives natural rise to the kernel functions

$$J(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(\boldsymbol{w}^\mathsf{T}\phi(\boldsymbol{x}_n) - t_n\right)^2 + \frac{\lambda}{2}\boldsymbol{w}^\mathsf{T}\boldsymbol{w}, \quad \text{where } \lambda \geq 0$$

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = \sum_{n=1}^{N}\left(\boldsymbol{w}^\mathsf{T}\phi(\boldsymbol{x}_n) - t_n\right)\phi(\boldsymbol{x}_n) + \lambda\boldsymbol{w} = 0$$

$$\boldsymbol{w} = -\frac{1}{\lambda}\sum_{n=1}^{N}\left(\boldsymbol{w}^\mathsf{T}\phi(\boldsymbol{x}_n) - t_n\right)\phi(\boldsymbol{x}_n) = \sum_{n=1}^{N} a_n\phi(\boldsymbol{x}_n) = \Phi^\mathsf{T}\boldsymbol{a}$$

where $\Phi = [\phi(x_1)^\mathsf{T} \ldots \phi(x_N)^\mathsf{T}] \in \mathbb{R}^{N\times D}$

- Thus, $\boldsymbol{w}$ is a linear combination of $\phi(\boldsymbol{x}_n)$
- The dual representation focuses on solving for $\boldsymbol{a}$, and not $\boldsymbol{w}$

TECHNISCHE UNIVERSITÄT DARMSTADT

Dual Representation of Linear Regression

- Insert the dual representation into the cost function

$$J(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(\boldsymbol{w}^\intercal\phi(\boldsymbol{x}_n) - t_n\right)^2 + \frac{\lambda}{2}\boldsymbol{w}^\intercal\boldsymbol{w}$$

$$J(\boldsymbol{a}) = \frac{1}{2}\sum_{n=1}^{N}\left(\boldsymbol{a}^\intercal\Phi\phi(\boldsymbol{x}_n) - t_n\right)^2 + \frac{\lambda}{2}\boldsymbol{a}^\intercal\Phi\Phi^T\boldsymbol{a}$$

$$= \frac{1}{2}\sum_{n=1}^{N}\boldsymbol{a}^\intercal\Phi\phi(\boldsymbol{x}_n)\,\phi(\boldsymbol{x}_n)^\intercal\,\boldsymbol{a} + \frac{1}{2}\sum_{n=1}^{N}t_n^2 - \sum_{n=1}^{N}\boldsymbol{a}^\intercal\Phi\phi(\boldsymbol{x}_n)\,t_n + \frac{\lambda}{2}\boldsymbol{a}^\intercal\Phi\Phi^T\boldsymbol{a}$$

$$= \frac{1}{2}\boldsymbol{a}^\intercal\underbrace{\Phi\Phi^\intercal}_{\boldsymbol{K}}\Phi\Phi^\intercal\boldsymbol{a} + \frac{1}{2}\boldsymbol{t}^t\boldsymbol{t} - \boldsymbol{a}^\intercal\Phi\Phi^\intercal\boldsymbol{t} + \frac{\lambda}{2}\boldsymbol{a}^\intercal\Phi\Phi^T\boldsymbol{a}$$

$$= \frac{1}{2}\boldsymbol{a}^\intercal\boldsymbol{K}\boldsymbol{K}\boldsymbol{a} + \frac{1}{2}\boldsymbol{t}^t\boldsymbol{t} - \boldsymbol{a}^\intercal\boldsymbol{K}\boldsymbol{t} + \frac{\lambda}{2}\boldsymbol{a}^\intercal\boldsymbol{K}\boldsymbol{a}$$

- $\boldsymbol{K} = \Phi\Phi^\intercal$ is the Gram Matrix, and $K_{ij} = \phi(\boldsymbol{x}_i)^\intercal\,\phi(\boldsymbol{x}_j) = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$

TECHNISCHE UNIVERSITÄT DARMSTADT

Dual Representation of Linear Regression

- Solve the dual problem for $\boldsymbol{a}$

$$J(\boldsymbol{a}) = \frac{1}{2}\boldsymbol{a}^\intercal \boldsymbol{K}\boldsymbol{K}\boldsymbol{a} + \frac{1}{2}\boldsymbol{t}^t\boldsymbol{t} - \boldsymbol{a}^\intercal \boldsymbol{K}\boldsymbol{t} + \frac{\lambda}{2}\boldsymbol{a}^\intercal \boldsymbol{K}\boldsymbol{a}$$

$$\frac{\partial J(\boldsymbol{a})}{\partial \boldsymbol{a}} = \boldsymbol{K}\boldsymbol{K}\boldsymbol{a} - \boldsymbol{K}\boldsymbol{t} + \lambda\boldsymbol{K}\boldsymbol{a} = \boldsymbol{K}\left(\boldsymbol{K}\boldsymbol{a} - \boldsymbol{t} + \lambda\boldsymbol{a}\right) = 0$$

$$\boldsymbol{a} = \left(\boldsymbol{K} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{t}$$

- Side note: since by definition of a kernel matrix, $\boldsymbol{K}$ is Positive Semi-Definite, $\boldsymbol{K}^{-1}$ exists

TECHNISCHE UNIVERSITÄT DARMSTADT

Dual Representation of Linear Regression

- Compute the prediction as

$$y(\boldsymbol{x}) = \boldsymbol{w}^\mathsf{T}\phi(\boldsymbol{x}) = \boldsymbol{a}^\mathsf{T}\Phi\phi(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^\mathsf{T}(\boldsymbol{K} + \lambda\boldsymbol{I})^{-1}\boldsymbol{t}$$

  where $\boldsymbol{k}(\boldsymbol{x}) = [k(\boldsymbol{x}, \boldsymbol{x}_1) \dots k(\boldsymbol{x}, \boldsymbol{x}_N)]^\mathsf{T}$

- All computations can be expressed in terms of the kernel function k

Pros and Cons of the Dual Representation

- **Cons**
  - Need to invert a $N \times N$ matrix
- **Pros**
  - Can work entirely in feature space with the help of kernels
  - Can even consider infinite feature spaces, as the kernel function does only have the inner product of feature vectors, which is a scalar, even for infinite feature spaces
  - Many novel algorithms can be derived from the dual representation
  - Many old problems of RBFs (how many kernels, which metric, which centers) can be solved in a principled way

Some Useful Kernels

- Polynomial kernels
  - E.g., 2nd order: $k(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{x}^\intercal \boldsymbol{z})^2$
  - N-th order with offset: $k(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{x}^\intercal \boldsymbol{z} + c)^N$
- Gaussian Kernel (also called Radial Basis Function - RBF)

$$k\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \exp\left(-\frac{1}{2\sigma^2}\left\|\boldsymbol{x} - \boldsymbol{x}'\right\|^2\right)$$

  - Arises from a feature space with an **INFINITE** number of radial basis functions

$$\int_{-\infty}^{+\infty} \exp\left(-\frac{1}{2\sigma^2}\left\|\boldsymbol{x} - \boldsymbol{x}'\right\|^2\right) \exp\left(-\frac{1}{2\sigma^2}\left\|\tilde{\boldsymbol{x}} - \boldsymbol{x}'\right\|^2\right) \mathrm{d}\tilde{\boldsymbol{x}}$$
$$\propto \exp\left(-\frac{1}{2\sigma^2}\left\|\boldsymbol{x} - \boldsymbol{x}'\right\|^2\right)$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Outline

TECHNISCHE UNIVERSITÄT DARMSTADT

Dual Representation of Linear Regression

- Classical linear (ridge/regularized) regression

$$
\begin{aligned}
y(\boldsymbol{x}) &= \boldsymbol{w}^\mathsf{T}\phi(\boldsymbol{x}) \\
\boldsymbol{w} &= \left(\boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\Phi} + \lambda \boldsymbol{I}\right)^{-1}\boldsymbol{\Phi}^\mathsf{T}\boldsymbol{t} \\
\boldsymbol{t} &= \boldsymbol{y} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \lambda)
\end{aligned}
$$

- Dual representation of linear regression

$$
\begin{aligned}
y(\boldsymbol{x}) &= \boldsymbol{a}^\mathsf{T}\boldsymbol{k}(\boldsymbol{x}) \\
\boldsymbol{a} &= \left(\boldsymbol{K} + \lambda \boldsymbol{I}\right)^{-1}\boldsymbol{t} \\
\boldsymbol{k}(\boldsymbol{x}) &= \begin{bmatrix} k(\boldsymbol{x}, \boldsymbol{x}_1) & \ldots & k(\boldsymbol{x}, \boldsymbol{x}_n) \end{bmatrix}^\mathsf{T} \\
k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) &= \phi(\boldsymbol{x}_i)^\mathsf{T}\phi\left(\boldsymbol{x}_j\right) \\
\boldsymbol{t} &= \boldsymbol{y} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \lambda)
\end{aligned}
$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Bayesian Linear Regression Revisited

- Regression model
$$y(\boldsymbol{x}) = \boldsymbol{w}^\mathsf{T}\phi(\boldsymbol{x})$$
- Parameter Distribution
$$p(\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{w} \,\middle|\, \mathbf{0}, \alpha^{-1}\boldsymbol{I}\right)$$
  - Thus, for any $\boldsymbol{w}$, one particular function of $\boldsymbol{x}$ is defined
  - The distribution over $\boldsymbol{w}$ thus induces a distribution over functions
- Goal: evaluate the function at some values of $\boldsymbol{x}$, e.g., the training set $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$
$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{w}$$
and predict the joint probability $p(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$

Bayesian Linear Regression Revisited

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{w} \qquad p(\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{w} \,\middle|\, \mathbf{0}, \alpha^{-1}\boldsymbol{I}\right)$$

- y is a linear combination of Gaussian random variables, and thus Gaussian itself
- To obtain the joint distribution of all $\boldsymbol{y}$, we only need the mean and covariance

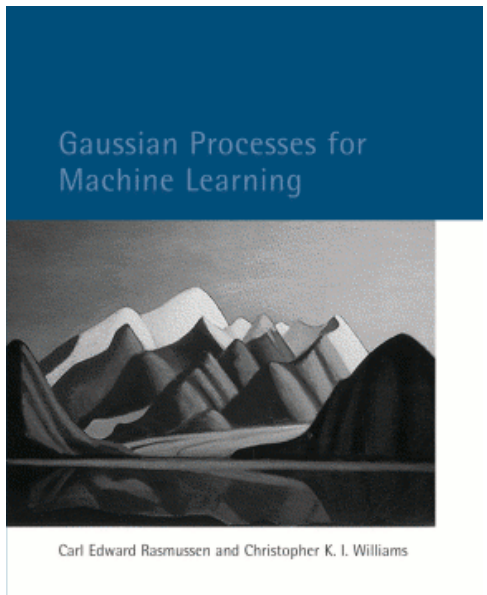
$$\mathbb{E}\{\boldsymbol{y}\} = \mathbb{E}\{\boldsymbol{\Phi}\boldsymbol{w}\} = \boldsymbol{\Phi}\mathbb{E}\{\boldsymbol{w}\} = \mathbf{0}$$

$$\operatorname{cov}\{\boldsymbol{y}\} = \mathbb{E}\{\boldsymbol{y}\boldsymbol{y}^\mathsf{T}\} = \boldsymbol{\Phi}\mathbb{E}\{\boldsymbol{w}\boldsymbol{w}^\mathsf{T}\}\boldsymbol{\Phi}^\mathsf{T} = \frac{1}{\alpha}\boldsymbol{\Phi}\boldsymbol{\Phi}^\mathsf{T} = \boldsymbol{K}$$

$$\text{where } K_{ij} = k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \frac{1}{\alpha}\phi\left(\boldsymbol{x}_i\right)^\mathsf{T}\phi\left(\boldsymbol{x}_j\right)$$

Gaussian Processes

- A Gaussian Process (GP) is a probability distribution over functions $y(\boldsymbol{x})$, such that any finite set of function values $y(\boldsymbol{x})$ evaluated at inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ is jointly Gaussian distributed
- A Gaussian Process over n variables $y_1, \ldots, y_n$ is completely specified by the 2nd order statistics, i.e., mean and covariance
- Rasmussen and Williams, 2006, *Gaussian Processes for Machine Learning* (`http://www.gaussianprocess.org/gpml/`)
- Good introduction to GPs by Carl Rasmussen: `http://videolectures.net/mlss09uk_rasmussen_gp/`

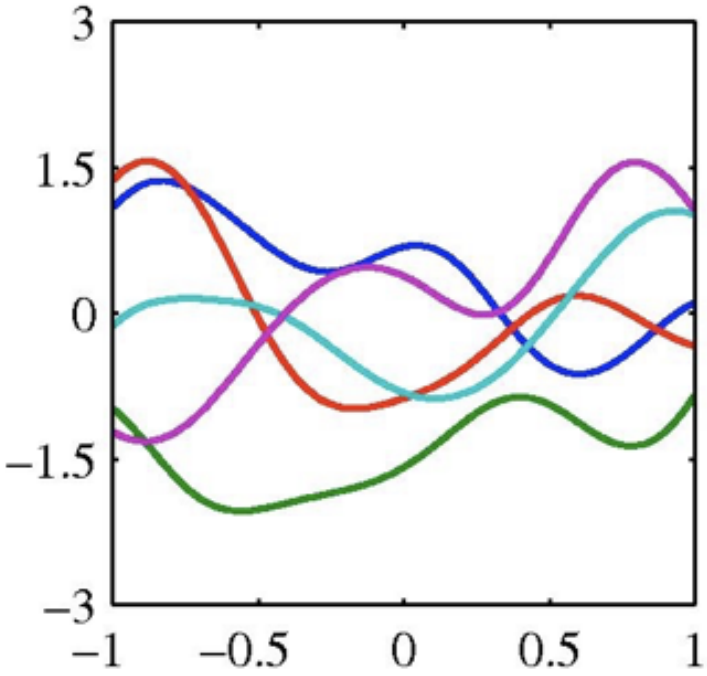


Gaussian Processes

- A GP is fully specified by a mean function and a covariance function (kernel)
  - Prior mean function: expected function before observing any data
  - Covariance function: encodes some structural assumptions (e.g., smoothness) (e.g., multivariate Gaussian kernel)
- Most applications assume the prior mean of $\boldsymbol{y}$ to be zero
  - Corresponds to a mean-zero prior of $\boldsymbol{w}$
- Thus, a GP is completely defined by

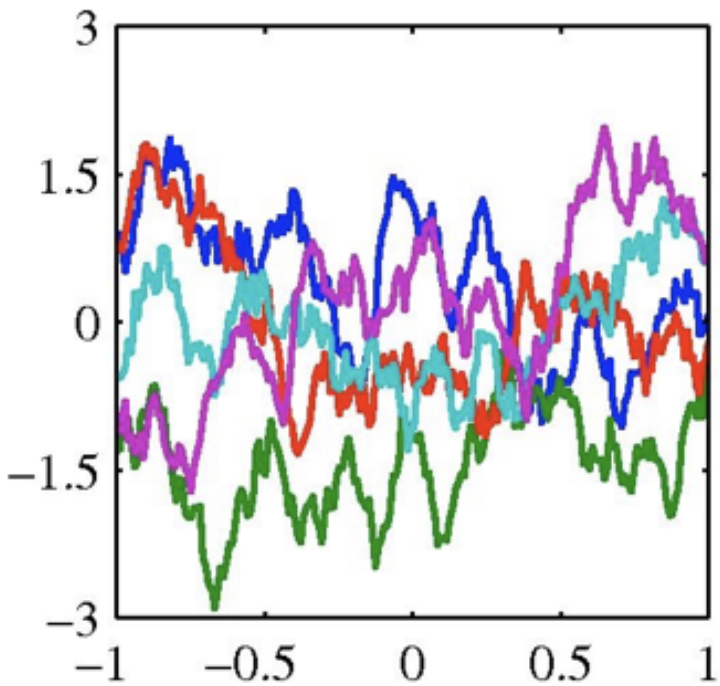
$$\mathbb{E}\{\boldsymbol{y}\} = \mathbb{E}\{\boldsymbol{\Phi}\boldsymbol{w}\} = \boldsymbol{\Phi}\mathbb{E}\{\boldsymbol{w}\} = \mathbf{0}$$
$$\mathbb{E}\left\{y\left(\boldsymbol{x}_i\right) y\left(\boldsymbol{x}_j\right)\right\} = k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$$

GPs - Different Covariance Functions



$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \exp\left(-\frac{1}{2\sigma^2}\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2\right)$$

Gaussian/RBF Kernel



$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \exp\left(-\theta\left\|\mathbf{x}_i - \mathbf{x}_j\right\|\right)$$

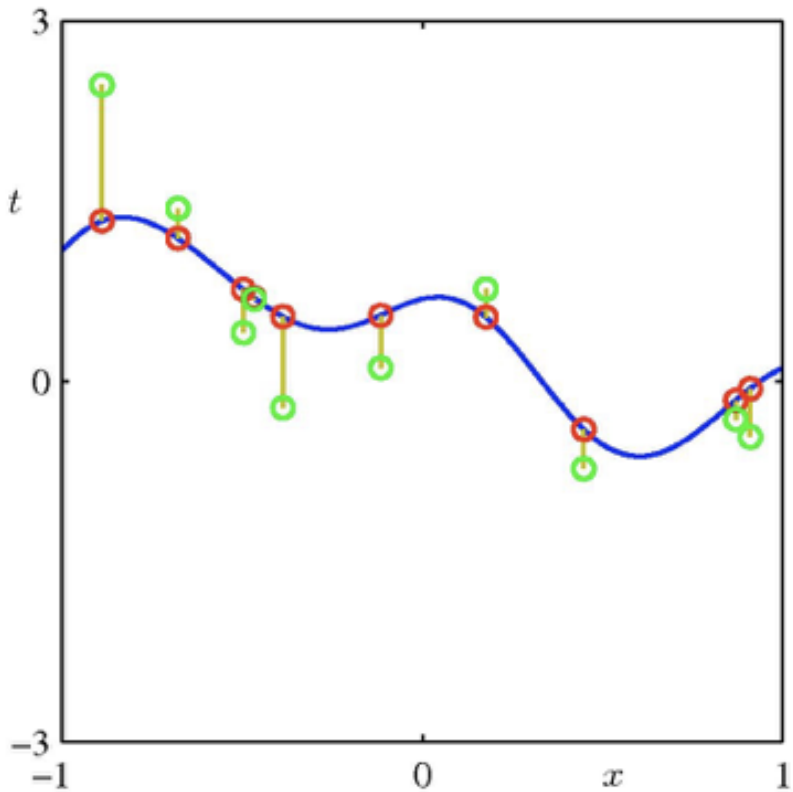
Ornstein-Uhlenbeck Process (Brownian Motion)

TECHNISCHE UNIVERSITÄT DARMSTADT

GPs for Regression

- Generative model: $t_n = y(x_n) + \epsilon$
- Noise model: $p(t_n \mid y_n) = \mathcal{N}\left(t_n \middle| y_n, \beta^{-1}\right)$
- Prior distribution over function values: $p(y) = \mathcal{N}\left(y \middle| 0, K\right)$
- The kernel function that determines K is typically chosen to express the property that, for similar points x_n and x_m the corresponding values $y(x_n)$ and $y(x_m)$ will be more strongly correlated than for dissimilar points. The definition of similarity depends on the application

GPs for Regression - Sampling Example



- Illustration of the sampling of data points t_n from a GP. The blue curve shows a sample function y from the GP posterior over functions. The red points show the values of y_n obtained by evaluating the function at a set of input values x_n. The corresponding values of t_n, shown in green, are obtained by adding independent Gaussian noise to each of the y_n

TECHNISCHE UNIVERSITÄT DARMSTADT

Inferring Functions with GPs

- Prior over functions (GP): $p(y)$
- Likelihood (measurement/noise model): $p(t \mid y)$
- Posterior over functions via Bayes theorem

$$p(y|t) = \frac{p(t|y)p(y)}{p(t)}$$

GPs Regression - Prediction for New Data Points

- Training set

$$\boldsymbol{t}_n = (t_1, \ldots, t_n)^\mathsf{T} \text{ with corresponding } \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$$

- **Predict** t_{n+1} for x_{n+1}
- Approach: evaluate the predictive distribution

$$p\left(t_{n+1} \,\middle|\, x_{n+1}, t_{1:n}, x_{1:n}\right)$$

- For the derivation, remember that GPs assume that $p\left(t_1, t_2, \ldots, t_n, t_{n+1}\right)$ is jointly Gaussian
- Therefore, the conditional distribution $p\left(t_{n+1} \,\middle|\, x_{n+1}, t_{1:n}, x_{1:n}\right)$ is also Gaussian distributed

TECHNISCHE UNIVERSITÄT DARMSTADT

Gaussian Conditioning

- Assume $\boldsymbol{x}$ is Gaussian distributed and it can be partitioned in two disjoint subsets $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$. We can rewrite the distribution in terms of the mean and covariance matrices of $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$

$$p(\boldsymbol{x}) = \mathcal{N}\left(\boldsymbol{x} \,\middle|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$$

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_a \\ \boldsymbol{x}_b \end{bmatrix}, \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{bmatrix}, \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{bmatrix}$$

- The conditional distribution is also Gaussian

$$p(\boldsymbol{x}_a \mid \boldsymbol{x}_b) = \mathcal{N}\left(\boldsymbol{x}_a \,\middle|\, \boldsymbol{\mu}_{a|b}, \boldsymbol{\Sigma}_{a|b}\right)$$

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\boldsymbol{x}_b - \boldsymbol{\mu}_b)$$

$$\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Gaussian Conditioning and Marginalization

- With Gaussian distributions the following holds

$$
\begin{aligned}
p(\boldsymbol{x}, \boldsymbol{y}) &\sim \mathcal{N} \\
&\Downarrow \\
p(\boldsymbol{x} \mid \boldsymbol{y}) &\sim \mathcal{N} \\
p(\boldsymbol{y} \mid \boldsymbol{x}) &\sim \mathcal{N} \\
p(\boldsymbol{x}) &\sim \mathcal{N} \\
p(\boldsymbol{y}) &\sim \mathcal{N}
\end{aligned}
$$

GPs Regression - Prediction for New Data Points

$$p\left(\boldsymbol{t}_{n+1}\right) = \mathcal{N}\left(\boldsymbol{t}_{n+1} \,\middle|\, \mathbf{0}, \boldsymbol{C}_{n+1}\right)$$

$$\boldsymbol{C}_{n+1} = \begin{pmatrix} \boldsymbol{C}_n & \boldsymbol{k} \\ \boldsymbol{k} & c \end{pmatrix}$$

where

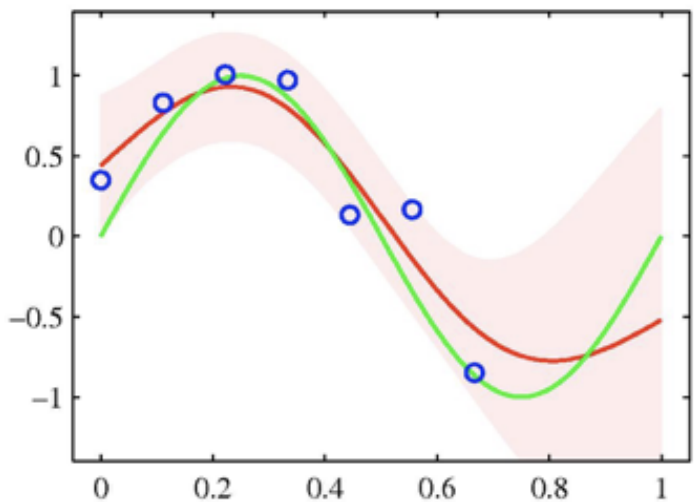
$$\boldsymbol{k} = \begin{bmatrix} k\left(\boldsymbol{x}_1, \boldsymbol{x}_{n+1}\right) & \ldots & k\left(\boldsymbol{x}_n, \boldsymbol{x}_{n+1}\right) \end{bmatrix}^{\mathsf{T}}$$

$$c = k\left(\boldsymbol{x}_{n+1}, \boldsymbol{x}_{n+1}\right) + \beta^{-1}$$

GPs Regression - Prediction for New Data Points

- Prediction Equations

$$m(\boldsymbol{x}_{n+1}) = \boldsymbol{k}^\top \boldsymbol{C}_N^{-1} \boldsymbol{t}$$
$$\sigma^2(\boldsymbol{x}_{n+1}) = c - \boldsymbol{k}^\top \boldsymbol{C}_N^{-1} \boldsymbol{k}$$



- Example of Sinusoidal Data Set (green: true function; blue: noisy data; red: GPR predictive mean; shaded: $\pm 2\sigma$)

GPs Regression - Notes

- Interpretation as RBFs

$$m(\boldsymbol{x}_{n+1}) = \boldsymbol{k}^\intercal \boldsymbol{C}_N^{-1} \boldsymbol{t} = \sum_{n=1}^{N} a_n k(\boldsymbol{x}_n, \boldsymbol{x}_{n+1})$$

- Computational Complexity
  - For building the model: $O(N^3)$
  - For prediction of one function value: $O(N^2)$ (for the variance)
- Key advantage of GPR: non-parametric and probabilistic

TECHNISCHE UNIVERSITÄT DARMSTADT

GPs Regression - Notes

- Naive methods can deal with $\sim 10.000 - 20.000$ data points
- Advanced methods (e.g., Sparse GPs) for more than 50.000 data points
- **IMPORTANT**: Hyperparameter optimization (parameters of the kernel / covariance function). E.g., for squared-exponential kernel

$$k\left(x_i, x_j\right) = \sigma_f^2 \exp\left(-\frac{1}{2l^2}\left(x_i - x_j\right)^2\right) + \sigma_n^2 \delta_{ij}$$

where σ_f^2 is the signal variance, l is the length-scale and σ_n^2 is the noise variance

GPs - Summary

- GPs are a Bayesian approach to regression with possibly infinite feature spaces
- Resulting prediction equations are very straightforward and obtained in closed-form because of the Gaussian properties
- Hyperparameter optimization more complex and expensive
- While GP for Regression is computationally very expensive, it is one of the most principled approaches to statistical learning for regression

TECHNISCHE UNIVERSITÄT DARMSTADT

Outline

Bayesian Learning - Pros

- Bayesian methods are a superset of many learning methods
- Regularization is a natural consequence
- No need for splitting into training and test sets
- Confidence intervals and error bars can be obtained
- Regularization can be obtained automatically
- Model comparison
- Active learning (determine where to sample next)
- Automatic relevance detection (which inputs are important)
- Black-box learning approaches
- Theoretically among the most powerful method

Bayesian Learning - Cons

- Requires to choose prior distributions, mostly based on analytical convenience rather than real knowledge about the problem
- Computationally intractable
  - Posterior probabilities involve the computation of an integral

$$p(\theta \mid x) = \frac{p(x|\theta)\, p(\theta)}{p(x)} = \frac{p(x|\theta)\, p(\theta)}{\int p(x, \theta)\, \mathrm{d}\theta} = \frac{p(x|\theta)\, p(\theta)}{\int p(x \mid \theta)\, p(\theta)\, \mathrm{d}\theta}$$

  - On the contrary, in non-Bayesian statistics we estimate parameters with maximum likelihood estimation. MLE is usually easier because it involves finding the maximum of the likelihood function, for which you can still use gradient descent, if there is no analytical solution

TECHNISCHE UNIVERSITÄT DARMSTADT

Bayesian Learning - Key Issues

- All parameters are treated probabilistically, i.e., avoid “point estimates”
- The probabilistic treatment allows integrating out all unknown parameters
- The problem of infinite regress?

Bayesian Learning - Key Issues

- Quantities of most interest in Bayesian approaches
  - Model Evidence

$$p(D) = \int p(D, \theta) \, \mathrm{d}\theta = \int p(D \mid \theta) \, p(\theta) \, \mathrm{d}\theta$$

  - Posterior of parameters

$$p(\theta \mid D) = \frac{p(D \mid \theta) \, p(\theta)}{p(D)}$$

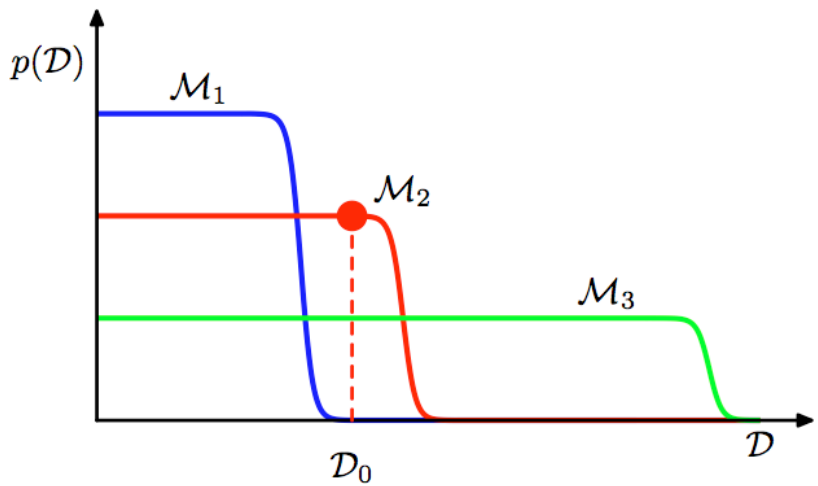
  - Predictive distribution

$$p(x \mid D) = \int p\left(x, \theta \,\middle|\, \theta\right) \mathrm{d}\theta$$

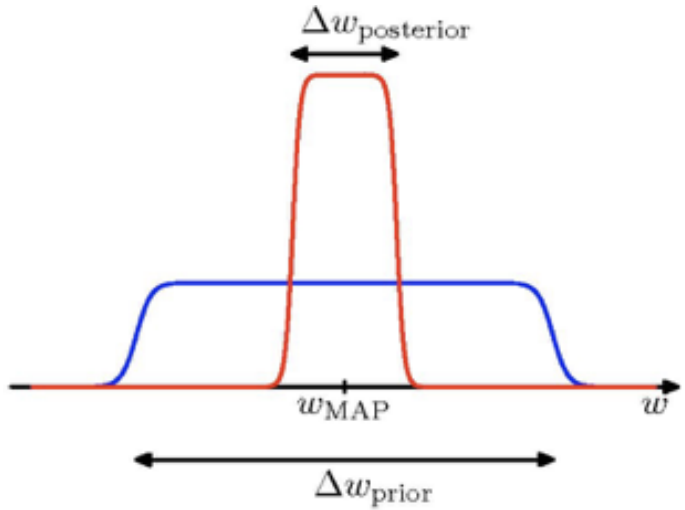
The Philosophy of Bayesian Model Selection

- Due to probability measure, models can be compared using the evidence
  - Complex models have lower probability density over large range of data sets
  - Simple models have high probability density over a small range of data sets
  - Thus, there should be a compromise in terms of complexity and confidence in the model

$$p\left(M_i \middle| D\right) = \frac{p\left(D \middle| M_i\right) p\left(M_i\right)}{\sum_j p\left(D \middle| M_j\right) p\left(M_j\right)}$$



Why the Evidence Achieves Regularization



- Approximate evidence in a one parameter scenario

$$p(D) = \int p(D \mid w)\, p(w)\, \mathrm{d}w \approx p\left(D \,\middle|\, w_{\text{MAP}}\right) \frac{\Delta w_{\text{posterior}}}{\Delta w_{\text{prior}}}$$

$$\ln p(D) \approx \ln p\left(D \,\middle|\, w_{\text{MAP}}\right) + \ln\left(\frac{\Delta w_{\text{posterior}}}{\Delta w_{\text{prior}}}\right)$$

- Note that the 2nd term penalizes the model complexity according to how finely the posterior is tuned

TECHNISCHE UNIVERSITÄT DARMSTADT

Why the Evidence Achieves Regularization

- For M parameters

$$\ln p\left(D\right) \approx \ln p\left(D \,\middle|\, w_{\mathrm{MAP}}\right) + M \ln\left(\frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}}\right)$$

- The penalty increases with the number of parameters

Bayesian Learning

- Parameters are modeled by probability distributions
- The conditional distribution of a new data point x given the training data D can be written as the marginalized joint distribution

$$p(x \mid D) = \int p(x, \boldsymbol{w} \mid D)\, \mathrm{d}\boldsymbol{w} = \int p(x \mid D, \boldsymbol{w})\, p(\boldsymbol{w} \mid D)\, \mathrm{d}\boldsymbol{w}$$
$$= \int p(x \mid \boldsymbol{w})\, p(\boldsymbol{w} \mid D)\, \mathrm{d}\boldsymbol{w}$$

- Hence the Bayesian approach performs a weighted average over all values of $\boldsymbol{w}$

Bayesian Learning

- Connection to maximum likelihood estimation

$$
\begin{aligned}
p(x \mid D) &= \int p(x \mid \boldsymbol{w}, D)\, p(\boldsymbol{w} \mid D)\, \mathrm{d}\boldsymbol{w} \\
&\approx p(x \mid \hat{\boldsymbol{w}}, D) \underbrace{\int p(\boldsymbol{w} \mid D)\, \mathrm{d}\mathbf{w}}_{=1} \\
&= p(x \mid \hat{\boldsymbol{w}}, D)
\end{aligned}
$$

- The approximation usually holds for sufficiently many training data points

Bayesian Learning

- How to perform Bayesian Updates

$$p(\boldsymbol{w} \mid D) = \frac{p(D \mid \boldsymbol{w})\, p(\boldsymbol{w})}{p(D)} = \frac{p(\boldsymbol{w})}{p(D)} \prod_{n=1}^{N} p\left(\boldsymbol{x}^n \,\middle|\, D\right)$$

$$p(D) = \int p(\boldsymbol{w}') \prod_{n=1}^{N} p\left(\boldsymbol{x}^n \,\middle|\, \boldsymbol{w}'\right) \mathrm{d}\boldsymbol{w}'$$

- To obtain predictions, one has to evaluate the integrals

$$p(D) = \int p(\boldsymbol{w}') \prod_{n=1}^{N} p\left(\boldsymbol{x}^n \,\middle|\, \boldsymbol{w}'\right) \mathrm{d}\boldsymbol{w}'$$

$$p(x \mid D) = \int p(x \mid \boldsymbol{w})\, p(\boldsymbol{w} \mid D)\, \mathrm{d}\boldsymbol{w}$$

- Generally this is a very complex computation
- Analytical solutions exist only if the posterior has the same parametric form as the prior (**conjugate priors**, reproducing densities)

Example - Bayesian Density Estimation with a Gaussian

- Determine the mean of the Gaussian by Bayesian Learning

$$\textbf{Prior: } p_0(\mu) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{(\mu-\mu_0)^2}{2\sigma_0^2}\right)$$

$$\textbf{Gaussian model: } p(x \mid \mu) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

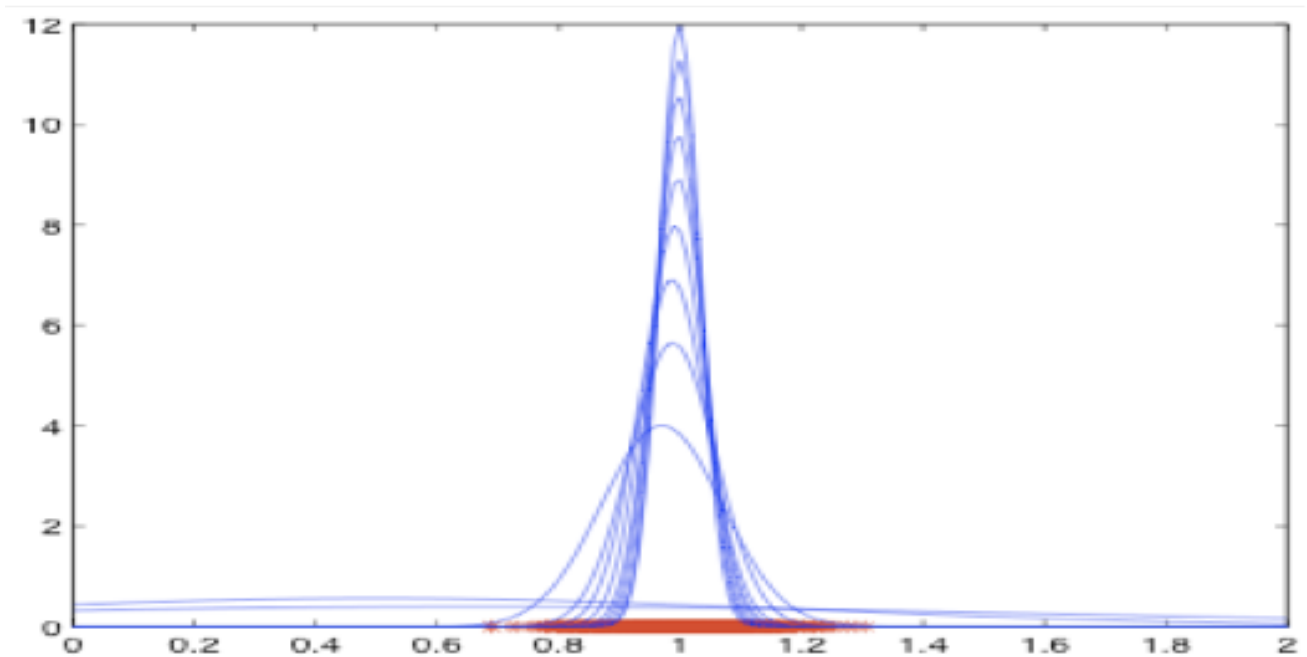
$$p_N(\mu \mid \boldsymbol{X}) = \frac{p_0(\mu)}{p(\boldsymbol{X})} \prod_{n=1}^{N} p(x^n \mid \mu)$$

- We get a Gaussian posterior with parameters

$$\mu_N = \frac{N\sigma_0^2}{N\sigma_0^2+\sigma^2}\bar{x} + \frac{\sigma_0^2}{N\sigma_0^2+\sigma^2}\mu_0, \quad \frac{1}{\sigma_N^2} = \frac{N}{\sigma^2} + \frac{1}{\sigma_0^2}, \quad \bar{x} = \frac{1}{N}\sum_{i=1}^{N} x^n$$

Example - Bayesian Density Estimation with a Gaussian

- Evolution of the posterior probability of the mean (blue) with increasing number of data points (red)



Bayesian Learning

- Assume probability distribution over network weights and some prior
- Need to interpret outputs probabilistically
- **Bayesian Learning Procedure**
  - Start with prior distribution $p(w)$ and choose appropriate parameters (usually broad distribution to reflect uncertainty)
  - Observe data, compute posterior of parameters with Bayes rule
  - Continue updating if more data comes in, replacing the prior with the posterior
  - In order to make a prediction, the expectation given the posterior distribution has to be found (might be very complex)

Gaussian Priors

- As usual, Gaussian priors are most convenient to deal with (for real numbers), although they may be unrealistic

$$p_0(w) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{(w-w_0)^2}{2\sigma_0^2}\right)$$

- More generally ...

$$p(\boldsymbol{w}) = \frac{1}{Z_w(\alpha)} \exp\left(-\alpha E_w\right)$$

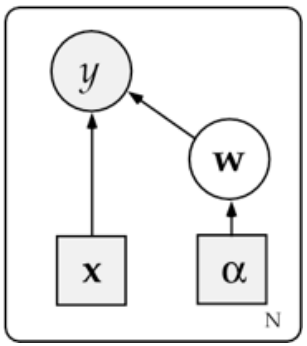
where

$$Z_w(\alpha) = \int \exp\left(-\alpha E_w\right) \mathrm{d}\boldsymbol{w} = \left(\frac{2\pi}{\alpha}\right)^{W/2}$$

$$E_w = \frac{1}{2}\|\boldsymbol{w}-\boldsymbol{w}_0\|^2 = \frac{1}{2}\sum_{i=1}^{w}(w_i - w_{0,i})^2, \quad E_w = \frac{1}{2}\|\boldsymbol{w}\|^2 = \frac{1}{2}\sum_{i=1}^{W} w_i^2$$

α is a hyperparameter

Example - Logistic Regression



$$p_N(\boldsymbol{w} \mid D) = \frac{p_0(\boldsymbol{w})}{p(D)} \prod_{n=1}^{N} p(t^n \mid \boldsymbol{w})$$

$$\textbf{Prior: } p(\boldsymbol{w}) = \exp\left(-\frac{\alpha}{2}\|\boldsymbol{w}\|^2\right) / \left(\frac{2\pi}{\alpha}\right)^{W/2}$$

$$\textbf{Likelihood: } p\left(t^n \,\middle|\, \boldsymbol{w}\right) = y(\boldsymbol{x}, \boldsymbol{w}) = \frac{1}{1 + \exp\left(-\boldsymbol{w}^\mathsf{T}\boldsymbol{x}\right)}$$

$$p(D) = \int y(\boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{w})\, \mathrm{d}\boldsymbol{w}$$

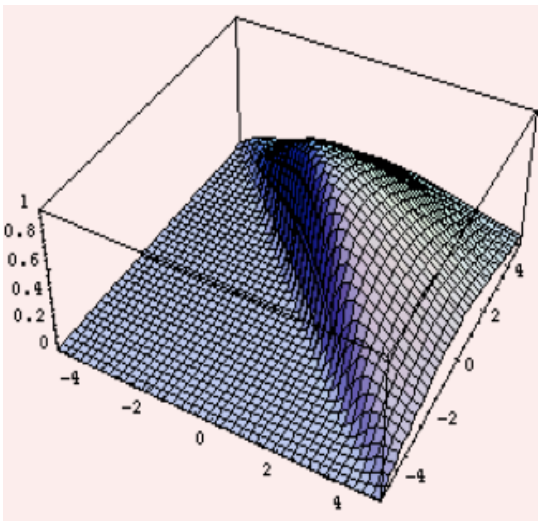
- Note: $p(D)$ is difficult to estimate, but we do not need it as long as we do not attempt model comparison, since it is only a scaling factor

Example - Logistic Regression

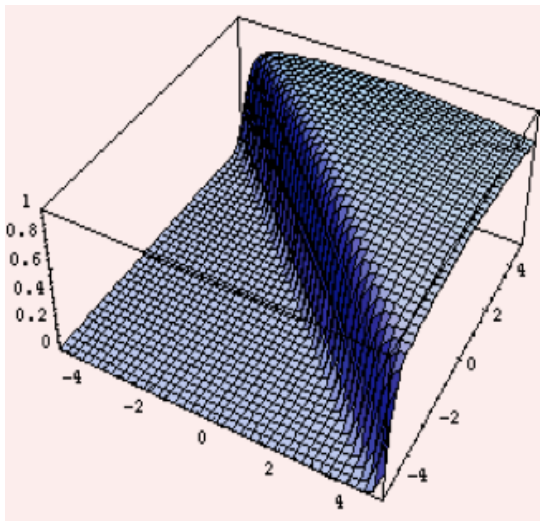
- Consider the data set

$$\boldsymbol{X} = \begin{bmatrix} 5 & 5 \\ -5 & -5 \\ 0 & 1 \\ -1 & 0 \end{bmatrix}, \quad \boldsymbol{T} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

- Use only two data points



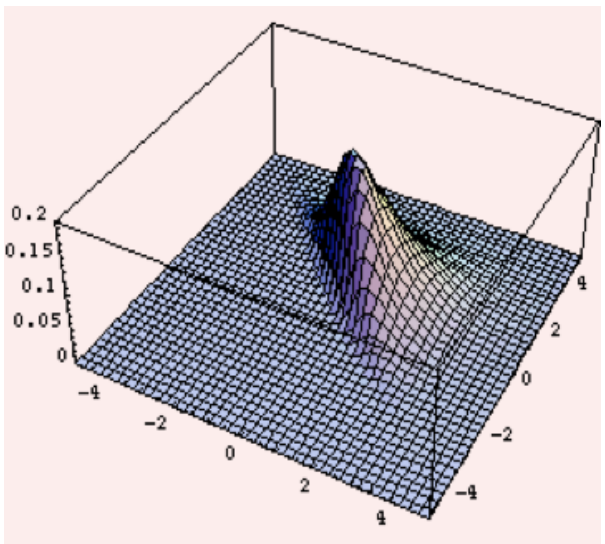
$\alpha = 0.1$



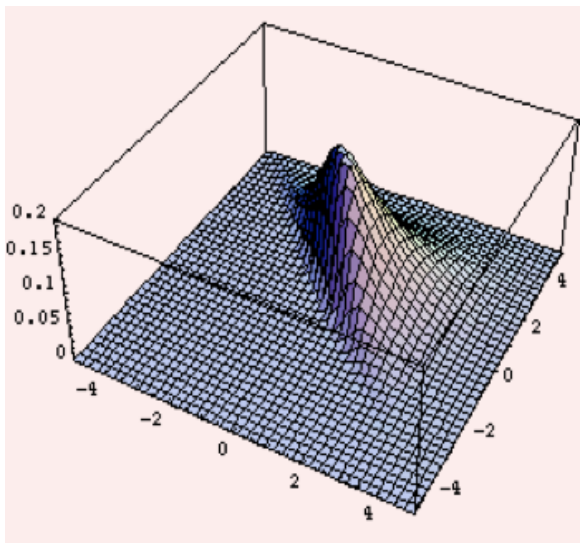
$\alpha = 0.01$

Example - Logistic Regression

- Use all data points



$\alpha = 0.1$



$\alpha = 0.01$

Gaussian Noise Models

- Make Gaussian Assumption for Likelihood

$$p(D \mid \boldsymbol{w}) = \frac{1}{Z_D(\beta)} \exp(-\beta E_D)$$

$$\text{where } Z_D(\beta) = \int \exp(-\beta E_D)$$

- For instance, for regression assume

$$p\left(t \,\middle|\, \boldsymbol{x}, \boldsymbol{w}\right) \propto \exp\left(-\frac{1}{2}\beta\left(y(\boldsymbol{x}, \boldsymbol{w}) - t\right)^2\right)$$

Gaussian Noise Models

- Posterior Distributions of Weights
  - Since all distributions are Gaussian, the posterior must be Gaussian and can be written as

$$p(\boldsymbol{w} \mid D) = \frac{1}{Z_S} \exp\left(-\beta E_D - \alpha E_W\right) = \frac{1}{Z_S} \exp\left(-S(\boldsymbol{w})\right)$$

$$\text{where } Z_D(\alpha, \beta) = \int \exp\left(-\beta E_D - \alpha E_W\right) \mathrm{d}\boldsymbol{w}$$

- What is the parameter vector maximizing the posterior? This can be achieved by minimizing

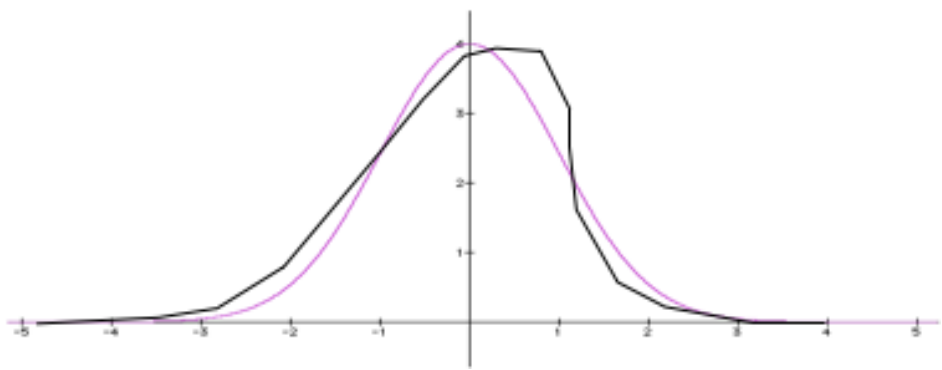
$$S(\boldsymbol{w}) = \frac{\beta}{2} \sum_{n=1}^{N} \left(y(\boldsymbol{x}^n; \boldsymbol{w}) - t^n\right)^2 + \frac{\alpha}{2} \sum_{i=1}^{W} w_i^2$$

Gaussian Approximation of Posterior Distribution

- Assume that there is an **analytically intractable** distribution
  - E.g., after obtaining the posterior of the parameters, the likelihood of the model may be desirable

$$p(y \mid D) = \int p(y \mid \boldsymbol{w})\, p(\boldsymbol{w} \mid D)\, \mathrm{d}\boldsymbol{w}$$

  - E.g., the posterior is required in Gaussian form
- Way out: e.g., approximate the intractable distribution with a Gaussian



Laplace Approximation

- Assume the generic probability distribution

$$p(z) = \frac{1}{Z} f(z), \text{ with } Z = \int f(z)\,\mathrm{d}z$$

- Goal: Approximate $p(z)$ with a Gaussian distribution, centered at the mode z_0

$$\left.\frac{\mathrm{d}f(z)}{\mathrm{d}z}\right|_{z=z_0}$$

- We get a 2nd order Taylor series expansion

$$\ln f(z) \approx \ln f(z_0) - \frac{1}{2}\left(-\left.\frac{\mathrm{d}^2}{\mathrm{d}z^2}\ln f(z)\right|_{z=z_0}\right)(z-z_0)^2$$

Laplace Approximation

- Taking the exp we get

$$f(z) \approx f(z_0) \exp\left(-\frac{1}{2}A(z-z_0)^2\right)$$

$$q(z) \approx \left(\frac{A}{2\pi}\right)^{1/2} \exp\left(-\frac{1}{2}A(z-z_0)^2\right)$$

- For the multivariate case

$$\ln f(\boldsymbol{z}) \approx \ln f(\boldsymbol{z_0}) - \frac{1}{2}(\boldsymbol{z}-\boldsymbol{z}_0)^\intercal \boldsymbol{A}(\boldsymbol{z}-\boldsymbol{z}_0)$$

$$\boldsymbol{A} = -\nabla\nabla \ln f(\boldsymbol{z})\Big|_{z=z_0}$$

$$f(\boldsymbol{z}) \approx f(\boldsymbol{z}_0) \exp\left(-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{z}_0)^\intercal \boldsymbol{A}(\boldsymbol{z}-\boldsymbol{z}_0)\right)$$

$$q(z) \approx \left(\frac{|\boldsymbol{A}|}{(2\pi)^M}\right)^{1/2} \exp\left(-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{z}_0)^\intercal \boldsymbol{A}(\boldsymbol{z}-\boldsymbol{z}_0)\right) = \mathcal{N}\left(\boldsymbol{z}\,\middle|\,\boldsymbol{z}_0, \boldsymbol{A}^{-1}\right)$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Laplace Approximation

- Illustration for approximation of logistic regression

INTELLIGENTE AUTONOME SYSTEME

Statistical Machine Learning

Lecture 13: Kernel Regression and Gaussian Processes

Jan Peters
TU Darmstadt

Summer Semester 2019

Dealing with Hyperparameters

- Augment Framework to also model the hyperparameters probabilistically

$$p(\boldsymbol{w} \mid D) = \int\int p\left(\boldsymbol{w}, \alpha, \beta \,\middle|\, D\right) \mathrm{d}\alpha \mathrm{d}\beta$$
$$= \int\int p\left(\boldsymbol{w} \,\middle|\, \alpha, \beta, D\right) p\left(\alpha, \beta \,\middle|\, D\right) \mathrm{d}\alpha \mathrm{d}\beta$$

- Assuming a sharp peak of the distributions of the hyperparameters

$$p(\boldsymbol{w} \mid D) \approx p\left(\boldsymbol{w} \,\middle|\, D, \alpha_{\mathrm{MP}}, \beta_{\mathrm{MP}}\right) \int\int p\left(\alpha, \beta \,\middle|\, D\right) \mathrm{d}\alpha \mathrm{d}\beta = p\left(\boldsymbol{w} \,\middle|\, D, \alpha_{\mathrm{MP}}, \beta_{\mathrm{MP}}\right)$$

- These assumptions offer the possibility to first find the hyperparameters that maximize the posterior, and then perform the remaining calculations with these optimized hyperparameters

- Note that there are also other methods to obtain the hyperparameters

Hyperparameters in Gaussian Processes

- What are the hyperparameters in GPs?
  - E.g, exponential-quadratic kernel

$$k(\boldsymbol{x}_n, \boldsymbol{x}_m) = \theta_0 \exp\left(-\frac{\theta_1}{2}\|\boldsymbol{x}_n - \boldsymbol{x}_m\|^2\right) + \theta_2 + \theta_3 \boldsymbol{x}_n^\intercal \boldsymbol{x}_m$$

- Approach: optimize the evidence w.r.t. the hyperparameters

$$p(\boldsymbol{t}) = \int p(\boldsymbol{t} \mid \boldsymbol{y})\, p(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y} = \mathcal{N}\left(\boldsymbol{t} \,\middle|\, \boldsymbol{0}, \boldsymbol{C}\right)$$

$$\text{with } C(\boldsymbol{x}_n, \boldsymbol{x}_m) = k(\boldsymbol{x}_n, \boldsymbol{x}_m) + \beta^{-1}\delta_{nm}$$

- E.g., by gradient descent

$$\frac{\partial}{\partial \theta_i} \log p(\boldsymbol{t} \mid \theta) = -\frac{1}{2}\mathrm{Tr}\left(\boldsymbol{C}_n^{-1}\frac{\partial \boldsymbol{C}_n}{\partial \theta_i}\right) + \frac{1}{2}\boldsymbol{t}^\intercal \boldsymbol{C}_n^{-1}\frac{\partial \boldsymbol{C}_n}{\partial \theta_i}\boldsymbol{C}_n^{-1}\boldsymbol{t}$$

Hyperparameters - Summary

- Bayesian learning offers an automatic way of regularization and meta-parameter tuning
- The evidence framework for model selection offers a principled tool to compare different learning systems
- Most of the time, Bayesian learning is analytically intractable
- Approximation methods exist to deal with the intractable components (Bayesian “hacking”)

TECHNISCHE UNIVERSITÄT DARMSTADT

Outline

TECHNISCHE UNIVERSITÄT DARMSTADT

4. Wrap-Up

You know now:

- What RBF Networks are
- What Kernels are, how to construct them and why they are beneficial
- How to derive the dual formulation of linear regression, and what are its pros and cons
- What GPs are, and the assumptions behind them
- With GPs we can predict the value for a new point in closed form, because of the Gaussian conditionals
- Doing regression with GPs we get a mean value and a variance (uncertainty) of the estimate
- Generally methods with kernels do not scale well with data
- The ideas behind Bayesian Learning, its pros and cons

Self-Test Questions

- Why kernel methods for regression?
- How do you get from radial basis functions to kernels?
- What is the role of the two pseudo-inverses in kernel regression?
- Why are kernel regression methods very computationally expensive?
- Why is kernel regression the dual to linear regression?
- What is the major advantage of GPs over Kernel Ridge Regression?
- Why are GPs a Bayesian approach?
- What principle allowed deriving GPs from a Bayesian regression point of view?
- How to get the hyperparameters in a Bayesian setup?

TECHNISCHE UNIVERSITÄT DARMSTADT

Extra Material & Homework

- Extra material
  - Goertler, et al., "A Visual Exploration of Gaussian Processes", Distill, 2019 (https://distill.pub/2019/visual-exploration-gaussian-processes/)
- Reading Assignment for next lecture
  - Bishop 8