# Advanced Machine Learning Lecture 4

## Kernels & Gaussian Processes

03.11.2016

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de/

leibe@vision.rwth-aachen.de

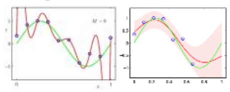
---



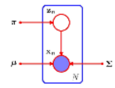
## This Lecture: *Advanced Machine Learning*

$f : \mathcal{X} \to \mathbb{R}$

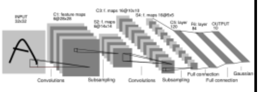
- Regression Approaches
  - Linear Regression
  - Regularization (Ridge, Lasso)
  - Kernels (Kernel Ridge Regression)
  - Gaussian Processes



- Learning with Latent Variables
  - EM and Generalizations
  - Approximate Inference



- Deep Learning
  - Neural Networks
  - CNNs, RNNs, RBMs, etc.





---



## Topics of This Lecture

- Recap: Linear Regression
- Kernels
  - Dual representations
  - Kernel Ridge Regression
  - Properties of kernels
- Gaussian Processes
  - Motivation
  - Gaussian Process definition
  - Squared exponential covariance function
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Applications



---



## Recap: Loss Functions for Regression

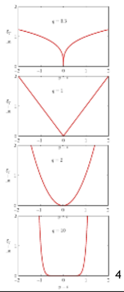
- The squared loss is not the only possible choice
  - Poor choice when conditional distribution $p(t|\mathbf{x})$ is multimodal.
- Simple generalization: Minkowski loss

$$L(t, y(\mathbf{x})) = |y(\mathbf{x}) - t|^q$$

  - Expectation

$$\mathbb{E}[L_q] = \int\int |y(\mathbf{x}) - t|^q p(\mathbf{x}, t) \mathrm{d}\mathbf{x}\mathrm{d}t$$

- Minimum of $\mathbb{E}[L_q]$ is given by
  - Conditional mean for $q = 2$,
  - Conditional median for $q = 1$,
  - Conditional mode for $q = 0$.





---

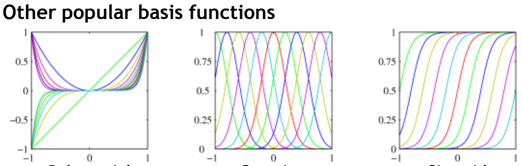


## Recap: Linear Basis Function Models

- Generally, we consider models of the following form

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

  - where $\phi_j(\mathbf{x})$ are known as *basis functions*.
  - In the simplest case, we use linear basis functions: $\phi_d(\mathbf{x}) = x_d$.
- Other popular basis functions



Polynomial — Gaussian — Sigmoid

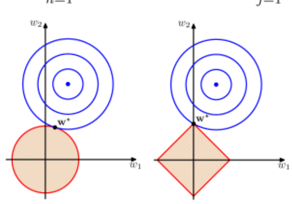


---



## Recap: Regularized Least-Squares

- Consider more general regularization functions
  - "$L_q$ norms": $\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$



- Effect: Sparsity for $q \le 1$.
  - Minimization tends to set many coefficients to zero



Image source: C.M. Bishop, 2006

## Recap: Lasso as Bayes Estimation

Advanced Machine Learning Winter'16

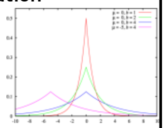
- $L_1$ regularization ("The Lasso")

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\phi(\mathbf{x}_n)\}^2 + \lambda\sum_{j=1}^{M}|w_j|$$

- Interpretation as Bayes Estimation
  - We can think of $|w_j|^q$ as the log-prior density for $w_j$.

- Prior for Lasso ($q = 1$): Laplacian distribution

$$p(\mathbf{w}) = \frac{1}{2\tau}\exp\{-|\mathbf{w}|/\tau\} \quad \text{with} \quad \tau = \frac{1}{\lambda}$$



B. Leibe 7

Image source: Wikipedia

## Topics of This Lecture

Advanced Machine Learning Winter'16

- Recap: Linear Regression
- Kernels
  - Dual representations
  - Kernel Ridge Regression
  - Properties of kernels
- Gaussian Processes
  - Motivation
  - Gaussian Process definition
  - Squared exponential covariance function
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Applications

B. Leibe 8

## Introduction to Kernel Methods

Advanced Machine Learning Winter'16

- Dual representations
  - Many linear models for regression and classification can be reformulated in terms of a dual representation, where predictions are based on linear combinations of a kernel function evaluated at training data points.
  - For models that are based on a fixed nonlinear feature space mapping $\phi(\mathbf{x})$, the kernel function is given by

$$k(\mathbf{x},\mathbf{x}') = \phi(\mathbf{x})^T\phi(\mathbf{x}')$$

  - We will see that by substituting the inner product by the kernel, we can achieve interesting extensions of many well-known algorithms…

B. Leibe 9

## Dual Representations: Derivation

Advanced Machine Learning Winter'16

- Consider a regularized linear regression model

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{\mathbf{w}^T\phi(\mathbf{x}_n) - t_n\}^2 + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

with the solution

$$\mathbf{w} = -\frac{1}{\lambda}\sum_{n=1}^{N}\{\mathbf{w}^T\phi(\mathbf{x}_n) - t_n\}\phi(\mathbf{x}_n)$$

  - We can write this as a linear combination of the $\phi(\mathbf{x}_n)$ with coefficients that are functions of $\mathbf{w}$:

$$\mathbf{w} = \sum_{n=1}^{N} a_n\phi(\mathbf{x}_n) = \Phi^T\mathbf{a}$$

with $\quad a_n = -\frac{1}{\lambda}\{\mathbf{w}^T\phi(\mathbf{x}_n) - t_n\}$

B. Leibe 10

## Dual Representations: Derivation

Advanced Machine Learning Winter'16

- Dual definition
  - Instead of working with $\mathbf{w}$, we can formulate the optimization for $\mathbf{a}$ by substituting $\mathbf{w} = \Phi^T\mathbf{a}$ into $J(\mathbf{w})$:

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{\mathbf{w}^T\phi(\mathbf{x}_n) - t_n\}^2 + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

$$J(\mathbf{a}) = \frac{1}{2}\mathbf{a}^T\Phi\Phi^T\Phi\Phi^T\mathbf{a} - \mathbf{a}^T\Phi\Phi^T\mathbf{t} + \frac{1}{2}\mathbf{t}^T\mathbf{t} + \frac{\lambda}{2}\mathbf{a}^T\Phi\Phi^T\mathbf{a}$$

  - Define the kernel matrix $\mathbf{K} = \Phi\Phi^T$ with elements

$$K_{nm} = \phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m)$$

  - Now, the sum-of-squares error can be written as

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

B. Leibe 11

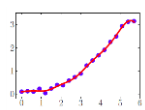
## Kernel Ridge Regression

Advanced Machine Learning Winter'16

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

  - Solving for $\mathbf{a}$, we obtain

$$\mathbf{a} = (\mathbf{K} + \lambda\mathbf{I}_N)^{-1}\mathbf{t}$$



- Prediction for a new input $\mathbf{x}$:
  - Writing $\mathbf{k}(\mathbf{x})$ for the vector with elements $k_n(\mathbf{x}) = k(\mathbf{x}_n, \mathbf{x})$

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

⇒ *The dual formulation allows the solution to be entirely expressed in terms of the kernel function $k(\mathbf{x},\mathbf{x}')$.*

⇒ The resulting form is known as Kernel Ridge Regression and allows us to perform non-linear regression.

B. Leibe 12

Image source: Christoph Lampert

## Why use $k(\mathbf{x},\mathbf{x}')$ instead of $\phi(\mathbf{x})^T\phi(\mathbf{x}')$?

1. Memory usage
   - Storing $\phi(\mathbf{x}_1),\ldots,\phi(\mathbf{x}_N)$ requires $O(NM)$ memory.
   - Storing $k(\mathbf{x}_1,\mathbf{x}_1),\ldots,k(\mathbf{x}_N,\mathbf{x}_N)$ requires $O(N^2)$ memory.
2. Speed
   - We might find an expression for $k(\mathbf{x}_i,\mathbf{x}_j)$ that is faster to evaluate than first forming $\phi(\mathbf{x})$ and then computing $\phi(\mathbf{x})^T\phi(\mathbf{x}')$.
   - Example: comparing angles ($x \in [0, 2\pi]$):

$$\begin{aligned}\langle\phi(x_i),\phi(x_j)\rangle &= \langle[\cos(x_i),\sin(x_i)],[\cos(x_j),\sin(x_j)]\rangle \\ &= \cos(x_i)\cos(x_j)+\sin(x_i)\sin(x_j) \\ k(x_i,x_j) &:= \cos(x_i-x_j)\end{aligned}$$





## Why use $k(\mathbf{x},\mathbf{x}')$ instead of $\phi(\mathbf{x})^T\phi(\mathbf{x}')$?

3. Flexibility
   - There are kernel functions $k(\mathbf{x}_i,\mathbf{x}_j)$ for which we know that a feature transformation $\phi$ exists, but we don't know what $\phi$ is.
   - This allows us to work with far more general similarity functions.
   - We can define kernels on strings, trees, graphs, …
4. Dimensionality
   - Since we no longer need to explicitly compute $\phi(\mathbf{x})$, we can work with high-dimensional (even infinite-dim.) feature spaces.

- *In the following, we take a closer look at the background behind kernels and at how to use them…*





## Properties of Kernels

- Definition (Positive Definite Kernel Function)
  - Let $\mathcal{X}$ be a non-empty set. A function $k: \mathcal{X}\times\mathcal{X}\to\mathbb{R}$ is called positive definite kernel function, iff
  - $k$ is symmetric, i.e. $k(x, x') = k(x', x)$ for all $x, x' \in \mathcal{X}$, and
  - for any set of points $x_1,\ldots, x_n \in \mathcal{X}$, the matrix

$$K_{ij} = (k(x_i,x_j))_{i,j}$$

is positive (semi-)definite, i.e. for all vectors $\mathbf{x}\in\mathbb{R}^n$:

$$\sum_{i,j=1}^{N}\mathbf{x}_i K_{ij}\mathbf{x}_j \geq 0$$





## Hilbert Spaces

- Definition (Hilbert Space)
  - A Hilbert Space $\mathcal{H}$ is a vector space $H$ with an *inner product* $\langle .\,,.\rangle_{\mathcal{H}}$, e.g. a mapping

$$\langle .,.\rangle_{\mathcal{H}} : H\times H\to\mathbb{R}$$

which is

  - symmetric: $\langle v, v'\rangle_{\mathcal{H}} = \langle v', v\rangle_{\mathcal{H}}$ for all $v, v' \in H$,
  - positive definite: $\langle v, v\rangle_{\mathcal{H}} \geq 0$ for all $v\in H$, where $\langle v, v\rangle_{\mathcal{H}} = 0$ only for $v = \mathbf{0}\in H$.
  - bilinear: $\langle av, v'\rangle_{\mathcal{H}} = a\langle v, v'\rangle_{\mathcal{H}}$ for $v\in H,\ a\in\mathbb{R}$; $\langle v+v', v''\rangle_{\mathcal{H}} = \langle v, v''\rangle_{\mathcal{H}} + \langle v', v''\rangle_{\mathcal{H}}$
- We can treat a Hilbert space like some $\mathbb{R}^n$, if we only use concepts like *vectors*, *angles*, *distances*.
- Note: $dim\mathcal{H} = \infty$ is possible!





## Properties of Kernels

- Theorem
  - *Let $k$: $\mathcal{X}\times\mathcal{X}\to\mathbb{R}$ be a positive definite kernel function. Then there exists a Hilbert Space $\mathcal{H}$ and a mapping $\varphi : \mathcal{X}\to\mathcal{H}$ such that*

$$k(x,x') = \langle(\phi(x),\phi(x')\rangle_{\mathcal{H}}$$

  - where $\langle .\,,.\rangle_{\mathcal{H}}$ *is the inner product in* $\mathcal{H}$.
- Translation
  - Take *any* set $\mathcal{X}$ and *any* function $k: \mathcal{X}\times\mathcal{X}\to\mathbb{R}$.
  - If $k$ is a positive definite kernel, then we can use $k$ to learn a (soft) maximum-margin classifier for the elements in $\mathcal{X}$!
- Note
  - $\mathcal{X}$ can be any set, e.g. $\mathcal{X}$ = *"all videos on YouTube"* or $\mathcal{X}$ = *"all permutations of* {1, . . . , k}*"*, or $\mathcal{X}$ = *"the internet"*.

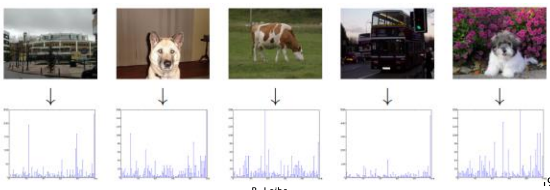




## Example: Bag of Visual Words Representation

- General framework in visual recognition
  - Create a codebook (vocabulary) of prototypical image features
  - Represent images as histograms over codebook activations
  - Compare two images by any histogram kernel, e.g. $\chi^2$ kernel

$$k_{\chi^2}(h,h') = \exp\left(-\frac{1}{\gamma}\sum_j\frac{(h_j-h'_j)^2}{h_j+h'_j}\right)$$





3

## The "Kernel Trick"

**Any algorithm that uses data only in the form of inner products can be *kernelized*.**

- How to kernelize an algorithm
  - Write the algorithm only in terms of inner products.
  - Replace all inner products by kernel function evaluations.

$\Rightarrow$ The resulting algorithm will do the same as the linear version, but in the (hidden) feature space $\mathcal{H}$.

- Caveat: working in $\mathcal{H}$ is not a guarantee for better performance. A good choice of $k$ and model selection are important!



## Outlook

- Kernels are a widely used concept in Machine Learning
  - They are the basis for Support Vector Machines from ML1.
  - We will see several other *kernelized* algorithms in this lecture…
- Examples
  - Gaussian Processes
  - Support Vector Regression
  - Kernel PCA
  - Kernel k-Means
  - …
- Let's first examine the role of kernels in probabilistic discriminative models.

$\Rightarrow$ This will lead us to Gaussian Processes.



## Topics of This Lecture

- Recap: Linear Regression
- Kernels
  - Dual representations
  - Kernel Ridge Regression
  - Properties of kernels
- Gaussian Processes
  - Motivation
  - Gaussian Process definition
  - Squared exponential covariance function
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Applications



## Gaussian Processes

- So far…
  - Considered linear regression models of the form

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T \phi(\mathbf{x})$$

  - where $\mathbf{w}$ is a vector of parameters
    $\phi(\mathbf{x})$ is a vector of fixed non-linear basis functions.
  - We showed that a prior distribution over $\mathbf{w}$ induced a prior distribution over functions $y(\mathbf{x},\mathbf{w})$.
  - Given a training set, we evaluated the posterior distribution over $\mathbf{w}$ $\Rightarrow$ corresponding posterior over regression functions.
  - This implies a predictive distribution $p(\mathbf{t}\,|\,\mathbf{x})$ for new inputs $\mathbf{x}$.
- Gaussian process viewpoint
  - Dispense with the parametric model and instead define a prior probability distribution over functions directly.



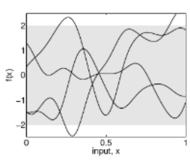
## Gaussian Process

- Gaussian distribution
  - Probability distribution over scalars / vectors.
- Gaussian process (generalization of Gaussian distrib.)
  - Describes properties of functions.
  - Function: Think of a function as a long vector where each entry specifies the function value $f(\mathbf{x}_i)$ at a particular point $\mathbf{x}_i$.
  - Issue: How to deal with infinite number of points?
    - If you ask only for properties of the function at a finite number of points…
    - Then inference in Gaussian Process gives you the same answer if you ignore the infinitely many other points.
- Definition
  - A Gaussian process (GP) is a collection of random variables any finite number of which has a joint Gaussian distribution.



## Gaussian Process

- Example prior over functions $p(f)$
  - Represents our prior belief about functions before seeing any data.
  - Although specific functions don't have mean of zero, the mean of $f(\mathbf{x})$ values for any fixed $\mathbf{x}$ is zero (here).




  - Favors smooth functions
    - I.e. functions cannot vary too rapidly
    - Smoothness is induced by the covariance function of the Gaussian Process.
  - Learning in Gaussian processes
    - Is mainly defined by finding suitable properties of the covariance function.

# Linear Regression Revisited

- Let's return to the linear regression example and re-derive the predictive distribution by working in terms of distributions over functions $y(\mathbf{x},\mathbf{w})$...
- Linear Regression Model

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T \phi(\mathbf{x})$$

  - Consider a prior distribution over $\mathbf{w}$ given by

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$$

  - For any given value of $\mathbf{w}$, the definition induces a particular function of $\mathbf{x}$.
  - The probability distribution over $\mathbf{w}$ therefore induces a probability distribution over functions $y(\mathbf{x})$.



# Linear Regression Revisited

- Linear Regression (cont'd)
  - We want to evaluate this function at specific values of $\mathbf{x}$, e.g. at the training data points $\mathbf{x}_1,\ldots,\mathbf{x}_N$.
  - We are therefore interested in the joint distribution of function values $y(\mathbf{x}_1),\ldots,y(\mathbf{x}_N)$, which we denote by the vector $\mathbf{y}$.

$$\mathbf{y} = \Phi\mathbf{w}$$

  - We know that $\mathbf{y}$ is a linear combination of Gaussian distributed variables and is therefore itself Gaussian.

⇒ Only need to find its mean and covariance.

$$\begin{aligned}\mathbb{E}[\mathbf{y}] &= \Phi\mathbb{E}[\mathbf{w}] = \mathbf{0} \\ \mathrm{cov}[\mathbf{y}] &= \mathbb{E}[\mathbf{y}\mathbf{y}^T] = \Phi\mathbb{E}[\mathbf{w}\mathbf{w}^T]\Phi^T = \frac{1}{\alpha}\Phi\Phi^T = \mathbf{K}\end{aligned}$$

  - with the kernel matrix $\mathbf{K} = \{k(\mathbf{x}_n,\mathbf{x}_m)\}_{nm}$.



# Gaussian Process

- This model is a particular example of a Gaussian Process.
  - Linear regression with a zero-mean, isotropic Gaussian prior on $\mathbf{w}$.
- General definition
  - A Gaussian Process is defined as a probability distribution over functions $y(\mathbf{x})$ such that the set of values of $y(\mathbf{x})$ evaluated at an arbitrary set of points $\mathbf{x}_1,\ldots,\mathbf{x}_N$ have a Gaussian distribution.
  - A key point about GPs is that the joint distribution over $N$ variables $y_1,\ldots,y_N$ is completely specified by the second-order statistics, namely mean and covariance.



# Gaussian Process

- A Gaussian process is completely defined by
  - Mean function $m(\mathbf{x})$ and

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$$

  - Covariance function $k(\mathbf{x},\mathbf{x}')$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x})(f(\mathbf{x}') - m(\mathbf{x}'))]$$

  - We write the Gaussian process (GP)

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$$

Slide adapted from Bernt Schiele



# Gaussian Process

- Property
  - Defined as a collection of random variables, which implies consistency.
  - Consistency means
    - If the GP specifies e.g. $(y_1,y_2) \sim \mathcal{N}(\mu,\Sigma)$ $\quad \Sigma = \begin{bmatrix}\Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22}\end{bmatrix}$
    - Then it must also specify $y_1 \sim \mathcal{N}(\mu_1,\Sigma_{11})$
  - I.e. examination of a larger set of variables does not change the distribution of a smaller set.

Slide credit: Bernt Schiele



# Gaussian Process: Example

- Example:
  - Bayesian linear regression model: $f(\mathbf{x}) = \phi(\mathbf{x})^T\mathbf{w}$
  - With Gaussian prior: $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \Sigma_p)$

⇒ Mean:

$$\mathbb{E}[f(\mathbf{x})] = \phi(\mathbf{x})^T\mathbb{E}[\mathbf{w}] = 0$$

⇒ Covariance:

$$\begin{aligned}\mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] &= \phi(\mathbf{x})^T\mathbb{E}[\mathbf{w}\mathbf{w}^T]\phi(\mathbf{x}') \\ &= \phi(\mathbf{x})^T\Sigma_p\phi(\mathbf{x}') \\ &= \tilde{\phi}(\mathbf{x})^T\tilde{\phi}(\mathbf{x}') \quad \text{where} \quad \tilde{\phi}(\mathbf{x}) = \Sigma_p^{\frac{1}{2}}\phi(\mathbf{x})\end{aligned}$$

Slide credit: Bernt Schiele

## Gaussian Process: Squared Exponential

- Typical covariance function
  - Squared exponential (SE)
    - Covariance function specifies the covariance between pairs of random variables

$$\mathrm{cov}[f(\mathbf{x}_p), f(\mathbf{x}_q)] = k(\mathbf{x}_p, \mathbf{x}_q) = \exp\left\{-\frac{1}{2}|\mathbf{x}_p - \mathbf{x}_q|^2\right\}$$

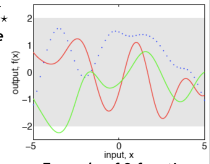
- Remarks
  - Covariance between the outputs is written as a function between the inputs.
  - The squared exponential covariance function corresponds to a Bayesian linear regression model with an infinite number of basis functions.
  - For any positive definite covariance function $k(.,.)$, there exists a (possibly infinite) expansion in terms of basis functions.





## Gaussian Process: Prior over Functions

- Distribution over functions:
  - Specification of covariance function implies distribution over functions.
  - I.e. we can draw samples from the distribution of functions evaluated at a (finite) number of points.
  - Procedure
    - We choose a number of input points $X_\star$
    - We write the corresponding covariance matrix (e.g. using SE) element-wise: $K(X_\star, X_\star)$
    - Then we generate a random Gaussian vector with this covariance matrix: $f_\star \sim \mathcal{N}(\mathbf{0}, K(X_\star, X_\star))$




Example of 3 functions sampled





## Topics of This Lecture







## Prediction with Noise-free Observations

- Assume our observations are noise-free:

$$\{(\mathbf{x}_n, f_n) \mid n = 1, \ldots, N\}$$

- Joint distribution of the training outputs $\mathbf{f}$ and test outputs $\mathbf{f}_\star$ according to the prior:

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

  - $K(X, X_\star)$ contains covariances for all pairs of training and test points.

- To get the posterior (after including the observations)
  - We need to restrict the above prior to contain only those functions which agree with the observed values.
  - Think of generating functions from the prior and rejecting those that disagree with the observations (obviously prohibitive).





## Prediction with Noise-free Observations

- Calculation of posterior: simple in GP framework
  - Corresponds to conditioning the joint Gaussian prior distribution on the observations:

$$\mathbf{f}_\star | X_\star, X, \mathbf{f} \sim \mathcal{N}(\bar{\mathbf{f}}_\star, \mathrm{cov}[\mathbf{f}_\star]) \qquad \bar{\mathbf{f}}_\star = \mathbb{E}[\mathbf{f}_\star | X, X_\star, \mathbf{t}]$$

  - with:

$$\begin{aligned}\bar{\mathbf{f}}_\star &= K(X_\star, X)K(X,X)^{-1}\mathbf{f}\\ \mathrm{cov}[\mathbf{f}_\star] &= K(X_\star, X_\star) - K(X_\star, X)K(X,X)^{-1}K(X, X_\star)\end{aligned}$$

  - This uses the general property of Gaussians that

$$\boldsymbol{\mu} = \begin{bmatrix}\boldsymbol{\mu}_a\\ \boldsymbol{\mu}_b\end{bmatrix}, \; \boldsymbol{\Sigma} = \begin{bmatrix}\boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab}\\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb}\end{bmatrix} \Rightarrow \begin{aligned}\boldsymbol{\mu}_{a|b} &= \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\mathbf{x}_b - \boldsymbol{\mu}_b)\\ \boldsymbol{\Sigma}_{a|b} &= \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}\end{aligned}$$

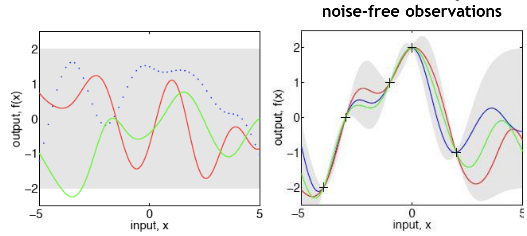




## Prediction with Noise-free Observations

- Example:

Prior — Posterior using 5 noise-free observations



## Topics of This Lecture

- Recap: Linear Regression
- Kernels
  - Dual representations
  - Kernel Ridge Regression
  - Properties of kernels
- Gaussian Processes
  - Motivation
  - Gaussian Process definition
  - Squared exponential covariance function
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Applications



## Prediction with Noisy Observations

- Typically, we assume noise in the observations

$$t = f(\mathbf{x}) + \epsilon \qquad \epsilon \sim \mathcal{N}(0, \sigma_n^2)$$

- The prior on the noisy observations becomes

$$\text{cov}[y_p, y_q] = k(\mathbf{x}_p, \mathbf{x}_q) + \sigma_n^2 \delta_{pq}$$

  - Written in compact form:

$$\text{cov}[\mathbf{y}] = K(X, X) + \sigma_n^2 I$$

- Joint distribution of the observed values and the test locations under the prior is then:

$$\begin{bmatrix} \mathbf{t} \\ \mathbf{f}_\star \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X)+\sigma_n^2 I & K(X,X_\star) \\ K(X_\star,X) & K(X_\star,X_\star) \end{bmatrix}\right)$$



## Prediction with Noisy Observations

- Calculation of posterior:
  - Corresponds to conditioning the joint Gaussian prior distribution on the observations:

$$\mathbf{f}_\star | X_\star, X, \mathbf{t} \sim \mathcal{N}(\bar{\mathbf{f}}_\star, \text{cov}[\mathbf{f}_\star]) \qquad \bar{\mathbf{f}}_\star \;=\; \mathbb{E}[\mathbf{f}_\star | X, X_\star, \mathbf{t}]$$

  - with:

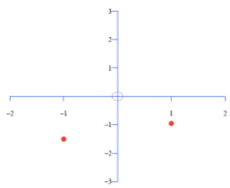
$$\begin{aligned} \bar{\mathbf{f}}_\star &= K(X_\star, X)\left(K(X,X)+\sigma_n^2 I\right)^{-1}\mathbf{t} \\ \text{cov}[\mathbf{f}_\star] &= K(X_\star,X_\star) - K(X_\star,X)\left(K(X,X)+\sigma_n^2 I\right)^{-1} K(X,X_\star) \end{aligned}$$

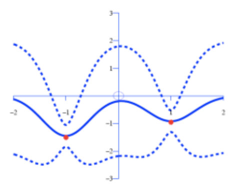
$\Rightarrow$ This is the key result that defines Gaussian process regression!
  - The predictive distribution is a Gaussian whose mean and variance depend on the test points $X_\star$ and on the kernel $k(\mathbf{x},\mathbf{x}')$, evaluated on the training data $X$.



## Gaussian Process Regression

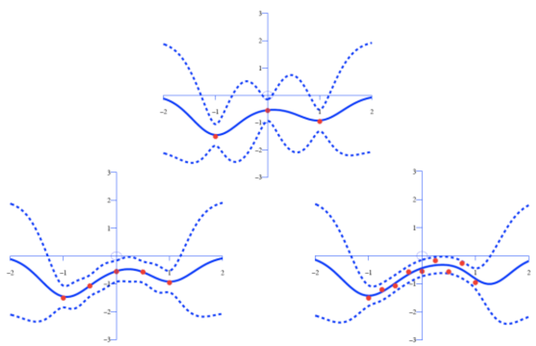
- Example







## Gaussian Process Regression





## Discussion

- Key result: $\mathbf{f}_\star | X_\star, X, \mathbf{t} \sim \mathcal{N}(\bar{\mathbf{f}}_\star, \text{cov}[\mathbf{f}_\star])$ with

$$\begin{aligned} \bar{\mathbf{f}}_\star &= K(X_\star, X)\left(K(X,X)+\sigma_n^2 I\right)^{-1}\mathbf{t} \\ \text{cov}[\mathbf{f}_\star] &= K(X_\star,X_\star) - K(X_\star,X)\left(K(X,X)+\sigma_n^2 I\right)^{-1} K(X,X_\star) \end{aligned}$$

- Observations
  - The mean can be written in linear form

$$\bar{f}(\mathbf{x}_\star) = k(\mathbf{x}_\star, X)\underbrace{[K(X,X)+\sigma_n^2 I]^{-1}\mathbf{t}}_{\boldsymbol{\alpha}} = \sum_{n=1}^{N} \alpha_n k(\mathbf{x}_\star, \mathbf{x}_n).$$

    - This form is commonly encountered in the kernel literature ($\rightarrow$SVM)
  - The variance is the difference between two terms

$$V(\mathbf{x}_\star) = \underbrace{k(\mathbf{x}_\star, \mathbf{x}_\star)}_{\text{Prior variance}} - \underbrace{k(\mathbf{x}_\star, X)[K(X,X)+\sigma_n^2 I]^{-1} k(X, \mathbf{x}_\star)}_{\text{Explanation of data } X}$$

## Computational Complexity

- Computational complexity
  - Central operation in using GPs involves inverting a matrix of size $N \times N$ (the kernel matrix $K(X,X)$):

$$\mathbf{f}_\star = K(X_\star, X)\boxed{\left(K(X,X)+\sigma_n^2 I\right)^{-1}}\mathbf{t}$$
$$\mathrm{cov}[\mathbf{f}_\star] = K(X_\star, X_\star) - K(X_\star, X)\boxed{\left(K(X,X)+\sigma_n^2 I\right)^{-1}}K(X, X_\star)$$

⇒ Effort in $\mathcal{O}(N^3)$ for $N$ data points!

  - Compare this with the basis function model (→Lecture 3)

$$p(f_\star|\mathbf{x}_\star, X, \mathbf{t}) \sim \mathcal{N}\left(\frac{1}{\sigma_n^2}\phi(\mathbf{x}_\star)^T\mathbf{S}^{-1}\mathbf{\Phi}(X)\mathbf{t}, \phi(\mathbf{x}_\star)^T\mathbf{S}^{-1}\phi(\mathbf{x}_\star)\right)$$
$$\mathbf{S} = \frac{1}{\sigma_n^2}\mathbf{\Phi}(X)\mathbf{\Phi}(X)^T + \Sigma_p^{-1}$$

⇒ Effort in $\mathcal{O}(M^3)$ for $M$ basis functions.



## Computational Complexity

- Complexity of GP model
  - Training effort: $\mathcal{O}(N^3)$ through matrix inversion
  - Test effort: $\mathcal{O}(N^2)$ through vector-matrix multiplication
- Complexity of basis function model
  - Training effort: $\mathcal{O}(M^3)$
  - Test effort: $\mathcal{O}(M^2)$
- Discussion
  - If the number of basis functions $M$ is smaller than the number of data points $N$, then the basis function model is more efficient.
  - However, advantage of GP viewpoint is that we can consider covariance functions that can only be expressed by an infinite number of basis functions.
  - Still, exact GP methods become infeasible for large training sets.



## Topics of This Lecture

- Recap: Linear Regression
- Kernels
  - Dual representations
  - Kernel Ridge Regression
  - Properties of kernels
- Gaussian Processes
  - Motivation
  - Gaussian Process definition
  - Squared exponential covariance function
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Applications



## Influence of Hyperparameters

- Most covariance functions have some free parameters.
  - Example:

$$k_y(\mathbf{x}_p, \mathbf{x}_q) = \sigma_f^2 \exp\left\{-\frac{(\mathbf{x}_p - \mathbf{x}_q)^2}{2 \cdot l^2}\right\} + \sigma_n^2\delta_{pq}$$

  - Parameters: $(l, \sigma_f, \sigma_n)$
    - Signal variance: $\sigma_f^2$
    - Range of neighbor influence (called "length scale"): $l$
    - Observation noise: $\sigma_n^2$

Slide credit: Bernt Schiele



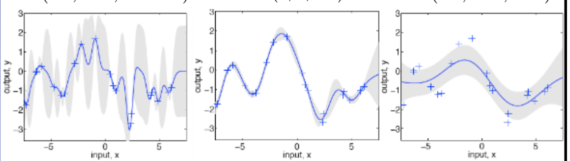
## Influence of Hyperparameters

$$k_y(\mathbf{x}_p, \mathbf{x}_q) = \sigma_f^2 \exp\left\{-\frac{(\mathbf{x}_p - \mathbf{x}_q)^2}{2 \cdot l^2}\right\} + \sigma_n^2\delta_{pq}$$

- Examples for different settings of the length scale

$(l, \sigma_f, \sigma_n) =$ ($\sigma$ parameters set by optimizing the marginal likelihood)

$= (0.3, 1.08, 0.00005)$ $= (1, 1, 0.1)$ $= (3.0, 1.16, 0.89)$



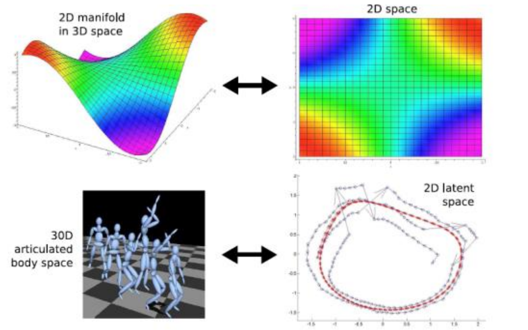

Slide credit: Bernt Schiele
Image source: Rasmussen & Williams, 2006



## Topics of This Lecture

- Recap: Linear Regression
- Kernels
  - Dual representations
  - Kernel Ridge Regression
  - Properties of kernels
- Gaussian Processes
  - Motivation
  - Gaussian Process definition
  - Squared exponential covariance function
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Applications

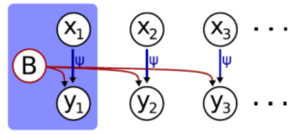
## Application: Non-Linear Dimensionality Reduction






## Gaussian Process Latent Variable Model

- At each time step $t$, we express our observations $\mathbf{y}$ as a combination of basis functions $\psi$ of latent variables $\mathbf{x}$.

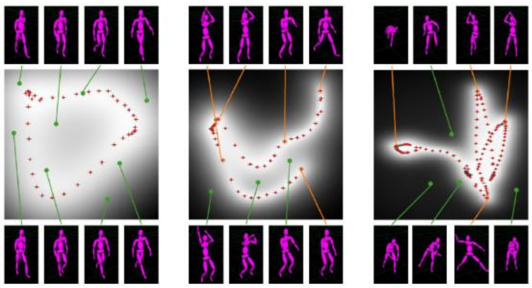



$$\mathbf{y}_t = \sum_j b_j \psi_j(\mathbf{x}_t) + \delta_t$$

- This is modeled as a Gaussian process...



## Example: Style-based Inverse Kinematics

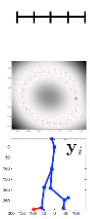


Learned GPLVMs using a *walk*, a *jump shot* and a *baseball pitch*



## Application: Modeling Body Dynamics

- Task: estimate full body pose in $m$ video frames.
  - High-dimensional $\mathbf{Y}$.
  - Model body dynamics using hierarchical Gaussian process latent variable model (hGPLVM) [Lawrence & Moore, ICML 2007].



Training

Time (frame #) (T) $\mathbf{T} = [t_i \in \mathbb{R}]$

$$p(\mathbf{Z}|\mathbf{T},\theta) = \prod_{i=1}^{q} \mathcal{N}(\mathbf{Z}_{:,i}|\mathbf{0}, \mathbf{K_T})$$

Latent space (Z) $\mathbf{Z} = [\mathbf{z}_i \in \mathbb{R}^q]$

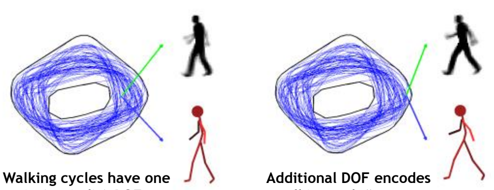
$$p(\mathbf{Y}|\mathbf{Z},\theta) = \prod_{i=1}^{D} \mathcal{N}(\mathbf{Y}_{:,i}|\mathbf{0}, \mathbf{K_z})$$

Configuration (Y) $\mathbf{Y} = [\mathbf{y}_i \in \mathbb{R}^D]$



## Articulated Motion in Latent Space (different work)

- Gaussian Process regression from latent space to
  - Pose [→ = p(Pose|z) to recover original pose from latent space]
  - Silhouette [→ = p(Silhouette|z) to do inference on silhouettes]

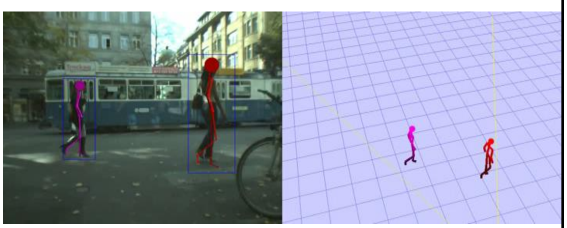


Walking cycles have one main (periodic) DOF

Additional DOF encodes „walking style“



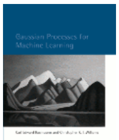
## Results



454 frames (~35 sec)
23 Pedestrians
20 detected by multi-body tracker

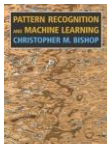
## References and Further Reading

- Kernels and Gaussian Processes are (shortly) described in Chapters 6.1 and 6.4 of Bishop's book.



Christopher M. Bishop
Pattern Recognition and Machine Learning
Springer, 2006

Carl E. Rasmussen, Christopher K.I. Williams
Gaussian Processes for Machine Learning
MIT Press, 2006




- A better introduction can be found in Chapters 1 and 2 of the book by Rasmussen & Williams (also available online: http://www.gaussianprocess.org/gpml/)