RWTH AACHEN UNIVERSITY

Advanced Machine Learning
Lecture 5

Gaussian Processes 2

07.11.2016

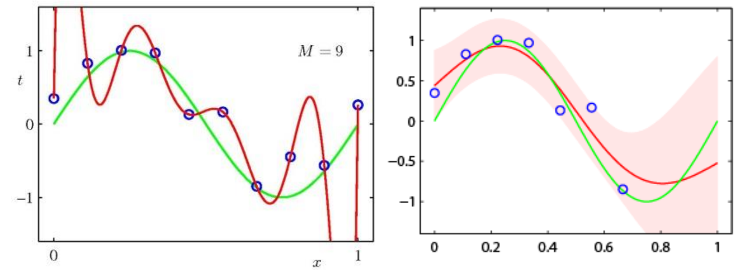
Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de/

leibe@vision.rwth-aachen.de

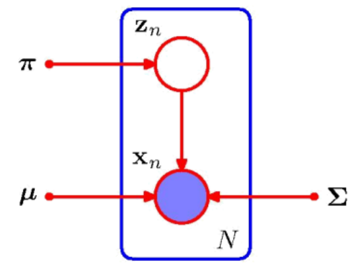
This Lecture: *Advanced Machine Learning*

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

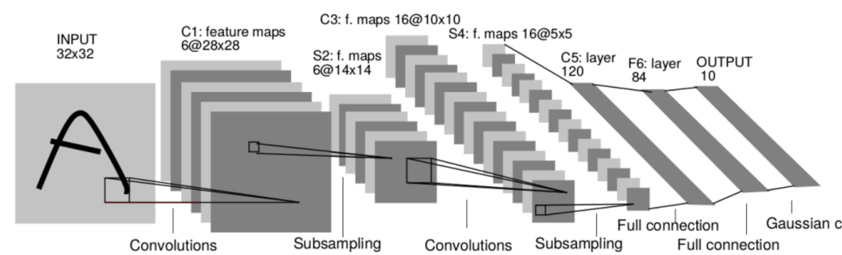
- Regression Approaches
  - Linear Regression
  - Regularization (Ridge, Lasso)
  - Kernels (Kernel Ridge Regression)
  - Gaussian Processes



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



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



Topics of This Lecture

- Kernels
  - Recap: Kernel trick
  - Constructing kernels
- Gaussian Processes
  - Recap: Definition
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Learning Gaussian Processes
  - Bayesian Model Selection
  - Model selection for Gaussian Processes
- Applications

Recap: Kernel Ridge Regression

- Dual definition
  - Instead of working with $\mathbf{w}$, substitute $\mathbf{w} = \mathbf{\Phi}^T\mathbf{a}$ into $J(\mathbf{w})$ and write the result using the kernel matrix $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^T$:

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

  - 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 $k_n(\mathbf{x}) = k(\mathbf{x}_n, \mathbf{x})$

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

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

Recap: 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*** $\phi : \mathcal{X} \to \mathcal{H}$ ***such that***

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

  - **where** $\langle . \ , \ . \rangle_{\mathcal{H}}$ ***is the inner product in*** **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 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"*.**

Slide credit: Christoph Lampert

Recap: 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!

Topics of This Lecture

- Kernels
  - Recap: Kernel trick
  - Constructing kernels
- Gaussian Processes
  - Recap: Definition
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Learning Gaussian Processes
  - Bayesian Model Selection
  - Model selection for Gaussian Processes
- Applications

Recap: 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.

Recap: 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)

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

Recap: GPs Define Prior over Functions

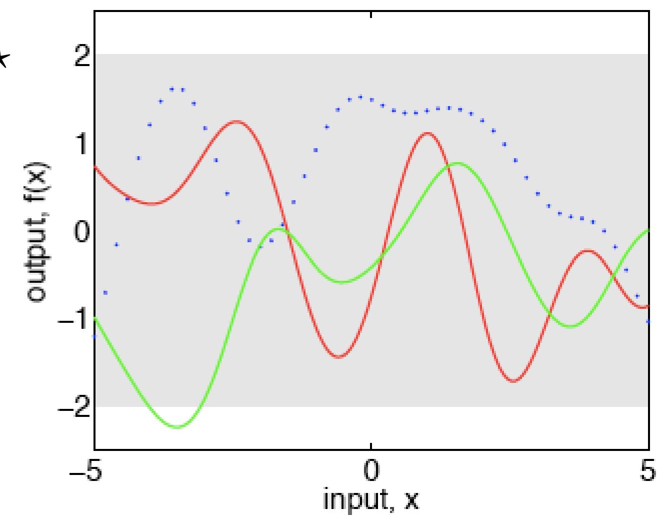
- **Distribution over functions:**
  - Specification of covariance function implies distribution over functions.
  - I.e. we can draw samples from the distribution of functions evaluated at a (finite) number of points.

  - 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

Image source: Rasmussen & Williams, 2006

Topics of This Lecture

- Kernels
  - Recap: Kernel trick
  - Constructing kernels
- Gaussian Processes
  - Recap: Definition
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Learning Gaussian Processes
  - Bayesian Model Selection
  - Model selection for Gaussian Processes
- Applications

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 $\mathbf{f}_*$ according to the prior:

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

  - $K(X, X_*)$ 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:

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

  - with:

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

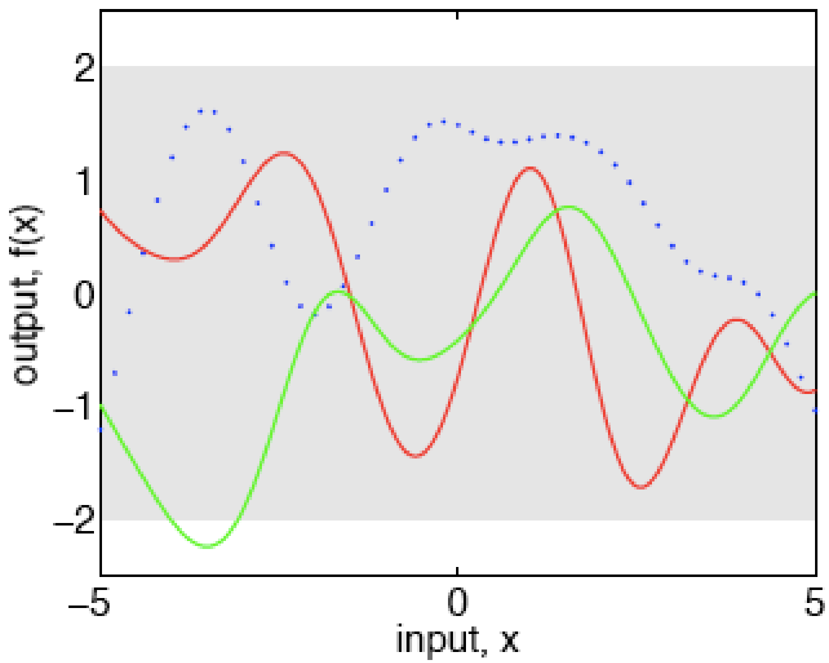
  - This uses the general property of Gaussians that

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

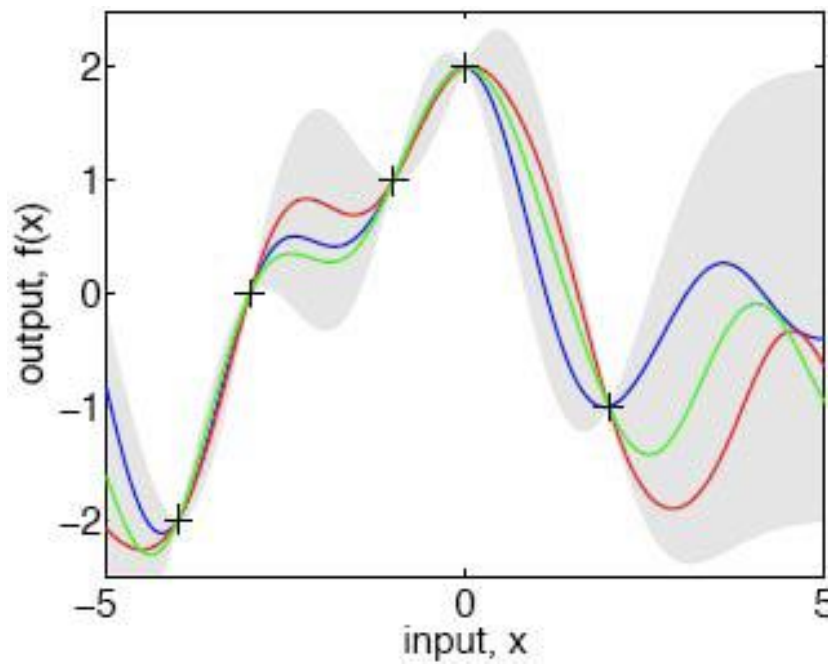
Prediction with Noise-free Observations

- **Example:**

Prior



Posterior using 5 noise-free observations



Topics of This Lecture

- Kernels
  - Recap: Kernel trick
  - Constructing kernels
- Gaussian Processes
  - Recap: Definition
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Learning Gaussian Processes
  - Bayesian Model Selection
  - Model selection for Gaussian Processes
- 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

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

  - Written in compact form:

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

- 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:

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

  - with:

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

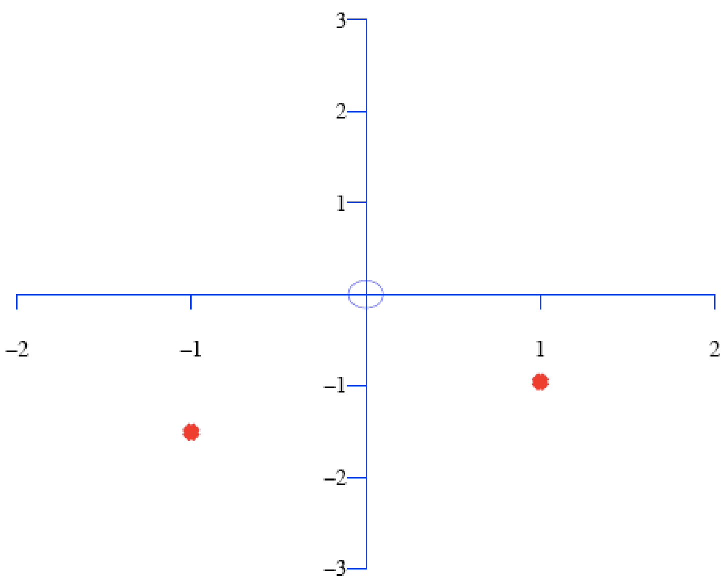
$$\mathrm{cov}[\mathbf{f}_\star] = K(X_\star, X_\star) - K(X_\star, X)\left(K(X,X) + \sigma_n^2 I\right)^{-1} K(X, X_\star)$$

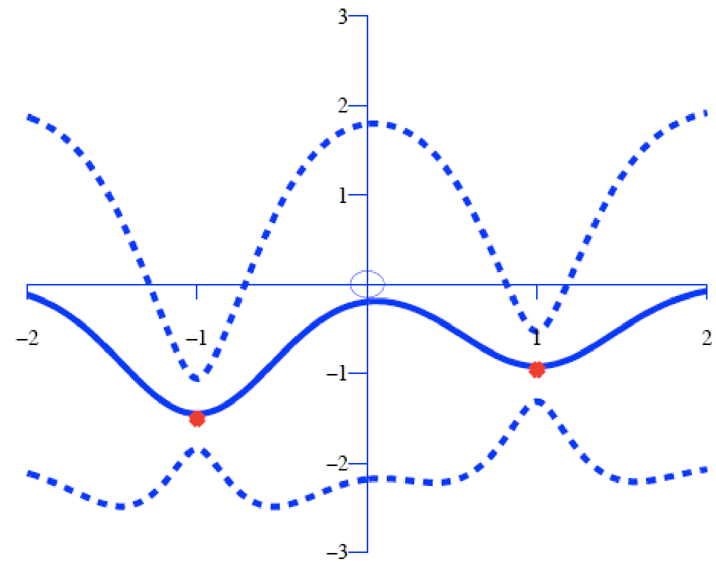
⇒ This is the key result that defines Gaussian process regression!

- The predictive distribution is a Gaussian whose mean and variance depend on the test points X_* and on the kernel $k(\mathbf{x},\mathbf{x}')$, evaluated on the training data X.

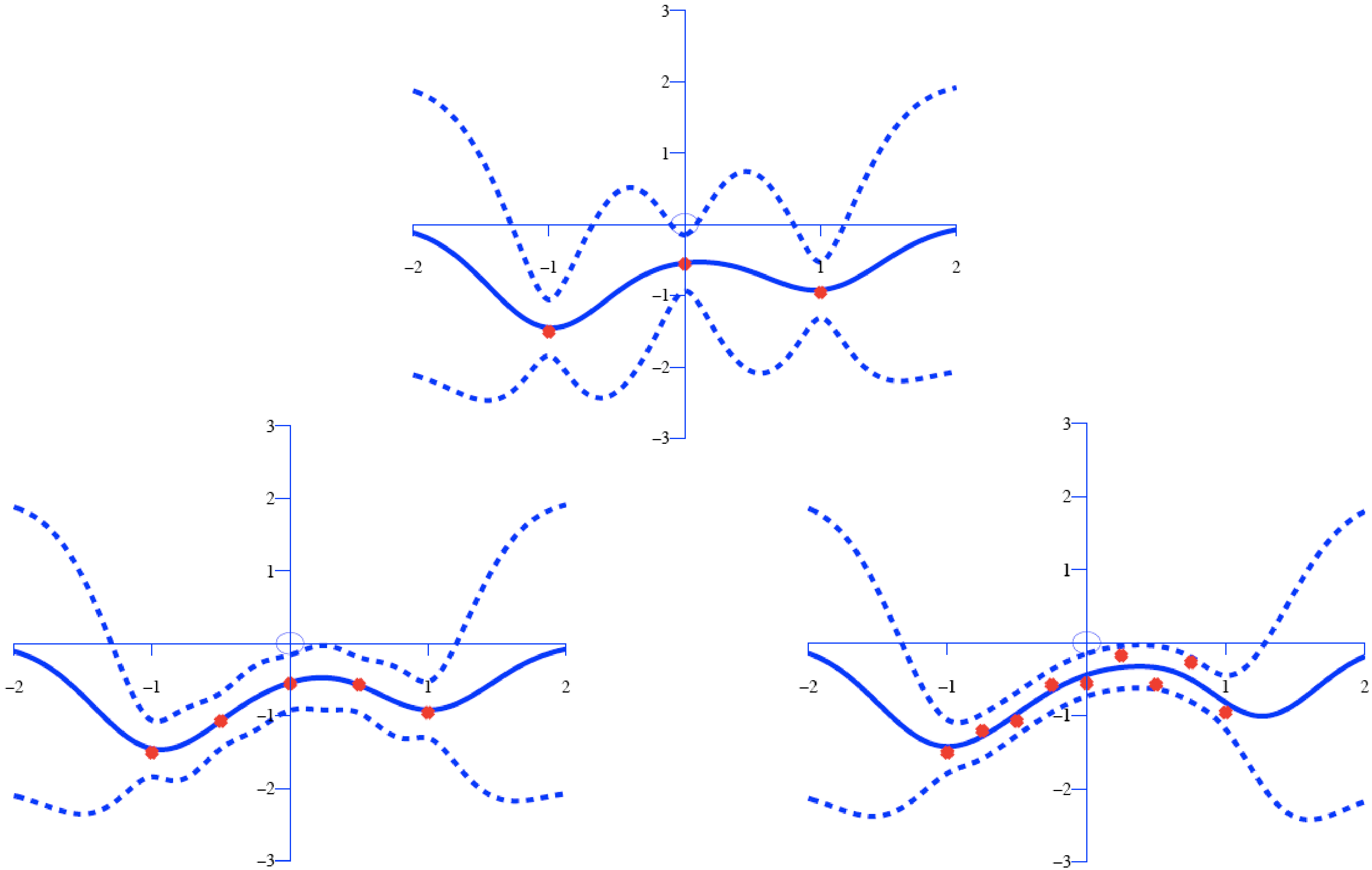
Gaussian Process Regression

- **Example**





Gaussian Process Regression



Discussion

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

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

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

- **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 (→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 $K(X,X)$):

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

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

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

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

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

$$\mathbf{S} = \frac{1}{\sigma_n^2}\mathbf{\Phi}(X)\mathbf{\Phi}(X)^T + \Sigma_p^{-1}$$

$\Rightarrow$ 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.

GP Regression Algorithm

- **Very simple algorithm!**

input: X (inputs), $\mathbf{y}$ (targets), k (covariance function), σ_n^2 (noise level), $\mathbf{x}_*$ (test input)

$L := \text{cholesky}(K + \sigma_n^2 I)$

$\boldsymbol{\alpha} := L^\top \backslash (L \backslash \mathbf{y})$

$\bar{f}_* := \mathbf{k}_*^\top \boldsymbol{\alpha}$

} predictive mean eq. (2.25)

$\mathbf{v} := L \backslash \mathbf{k}_*$

$\mathbb{V}[f_*] := k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{v}^\top \mathbf{v}$

} predictive variance eq. (2.26)

$\log p(\mathbf{y}|X) := -\frac{1}{2}\mathbf{y}^\top \boldsymbol{\alpha} - \sum_i \log L_{ii} - \frac{n}{2}\log 2\pi$ eq. (2.30)

return: $\bar{f}_*$ (mean), $\mathbb{V}[f_*]$ (variance), $\log p(\mathbf{y}|X)$ (log marginal likelihood)

- **Based on the following equations (Matrix inv. ↔ Cholesky fact.)**

$$\bar{f}_\star = \mathbf{k}_\star^T \left(K + \sigma_n^2 I\right)^{-1} \mathbf{t}$$

$$\text{cov}[f_\star] = k(\mathbf{x}_\star, \mathbf{x}_\star) - \mathbf{k}_\star^T \left(K + \sigma_n^2 I\right)^{-1} \mathbf{k}_\star$$

$$\log p(\mathbf{t}|X) = -\frac{1}{2}\mathbf{t}^T \left(K + \sigma_n^2 I\right)^{-1} \mathbf{t} - \frac{1}{2}\log|K + \sigma_n^2 I| - \frac{N}{2}\log 2\pi$$

Image source: Rasmussen & Williams, 2006

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, σ_f, σ_n)
    - Signal variance: σ_f^2
    - Range of neighbor influence (called "length scale"): l
    - Observation noise: σ_n^2

Influence of Hyperparameters

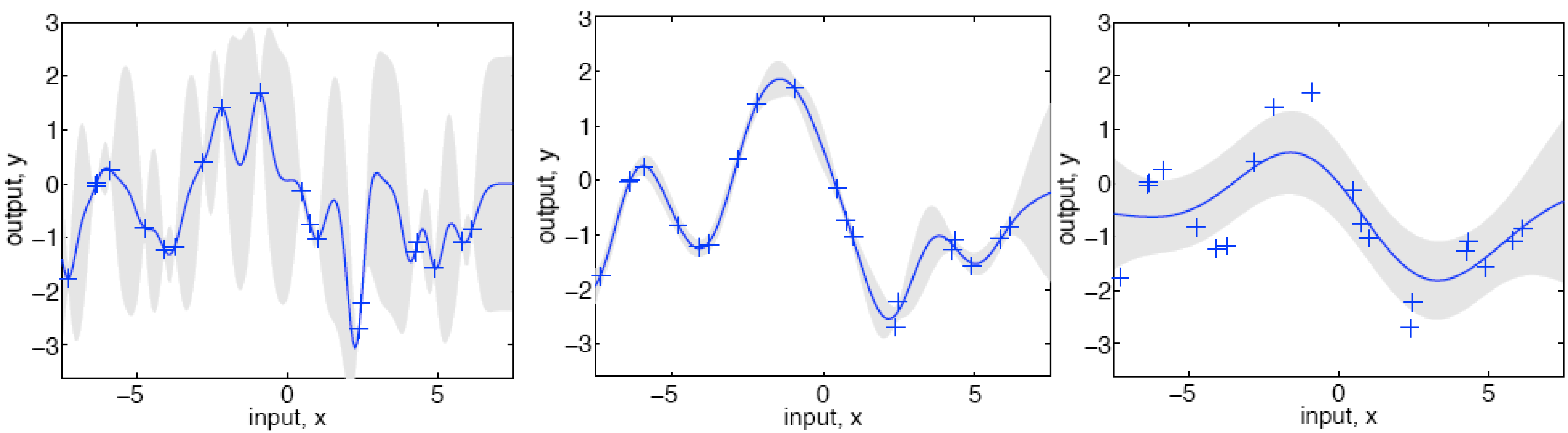
$$k_y(\mathbf{x}_p, \mathbf{x}_q) = \sigma_f^2 \exp\left\{-\frac{(\mathbf{x}_p - \mathbf{x}_q)^2}{2 \cdot l^2}\right\} + \sigma_n^2 \delta_{pq}$$

- **Examples for different settings of the length scale**

$(l, \sigma_f, \sigma_n) =$

(σ parameters set by optimizing the marginal likelihood)

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

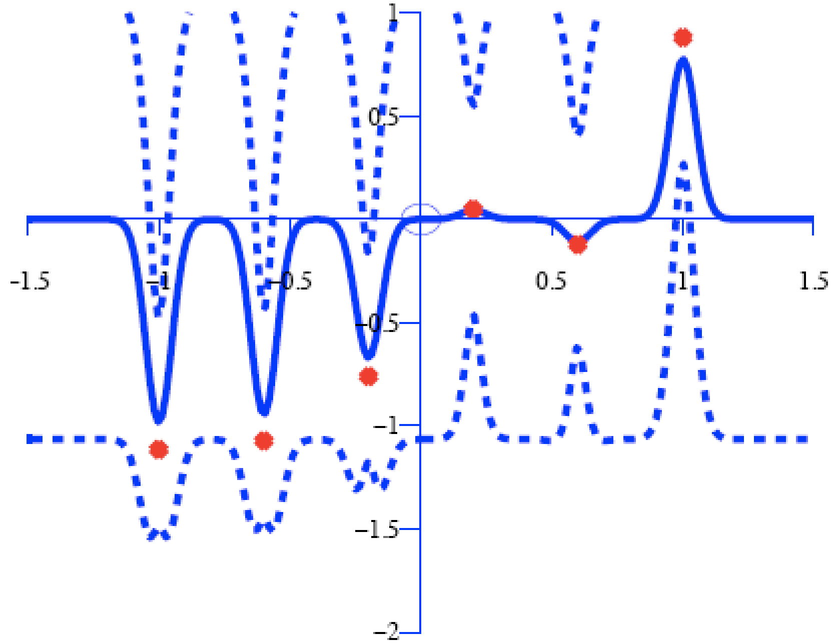


Topics of This Lecture

- Kernels
  - Recap: Kernel trick
  - Constructing kernels
- Gaussian Processes
  - Recap: Definition
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- **Learning Gaussian Processes**
  - **Bayesian Model Selection**
  - **Model selection for Gaussian Processes**
- Applications

Learning Kernel Parameters

- **Can we determine the length scale and noise levels from training data?**



Bayesian Model Selection

- Goal
  - Determine/learn different parameters of Gaussian Processes
- Hierarchy of parameters
  - Lowest level
    - $\mathbf{w}$ - e.g. parameters of a linear model.
  - Mid-level (hyperparameters)
    - θ - e.g. controlling prior distribution of $\mathbf{w}$.
  - Top level
    - Typically discrete set of model structures $\mathcal{H}_i$.
- Approach
  - Inference takes place one level at a time.

Model Selection at Lowest Level

- Posterior of the parameters $\mathbf{w}$ is given by Bayes' rule

$$p(\mathbf{w}|\mathbf{t}, X, \theta, \mathcal{H}_i) = \frac{p(\mathbf{t}|X, \mathbf{w}, \theta, \mathcal{H}_i)p(\mathbf{w}|\theta, X, \mathcal{H}_i)}{p(\mathbf{t}|X, \theta, \mathcal{H}_i)}$$

$$= \frac{p(\mathbf{t}|X, \mathbf{w}, \mathcal{H}_i)p(\mathbf{w}|\theta, \mathcal{H}_i)}{p(\mathbf{t}|X, \theta, \mathcal{H}_i)}$$

- with
  - $p(\mathbf{t}\,|\,X, \mathbf{w}, \mathcal{H}_i)$ likelihood and
  - $p(\mathbf{w}\,|\,\theta, \mathcal{H}_i)$ prior parameters $\mathbf{w}$,
  - Denominator (normalizing constant) is independent of the parameters and is called marginal likelihood.

$$p(\mathbf{t}|X, \theta, \mathcal{H}_i) = \int p(\mathbf{t}|X, \mathbf{w}, \mathcal{H}_i)p(\mathbf{w}|\theta, \mathcal{H}_i)d\mathbf{w}$$

Model Selection at Mid Level

- Posterior of parameters θ is again given by Bayes' rule

$$p(\theta|\mathbf{t}, X, \mathcal{H}_i) = \frac{p(\mathbf{t}|X, \theta, \mathcal{H}_i)p(\theta|X, \mathcal{H}_i)}{p(\mathbf{t}|X, \mathcal{H}_i)}$$

$$= \frac{p(\mathbf{t}|X, \theta, \mathcal{H}_i)p(\theta|\mathcal{H}_i)}{p(\mathbf{t}|X, \mathcal{H}_i)}$$

- where
  - The marginal likelihood of the previous level $p(\mathbf{t}|X,\theta,\mathcal{H}_i)$ plays the role of the likelihood of this level.
  - $p(\theta|\mathcal{H}_i)$ is the hyperprior (prior of the hyperparameters)
  - Denominator (normalizing constant) is given by:

$$p(\mathbf{t}|X, \mathcal{H}_i) = \int p(\mathbf{t}|X, \theta, \mathcal{H}_i)p(\theta|\mathcal{H}_i)d\theta$$

which is again a marginal likelihood (at the mid level).

Model Selection at Top Level

- At the top level, we calculate the posterior of the model

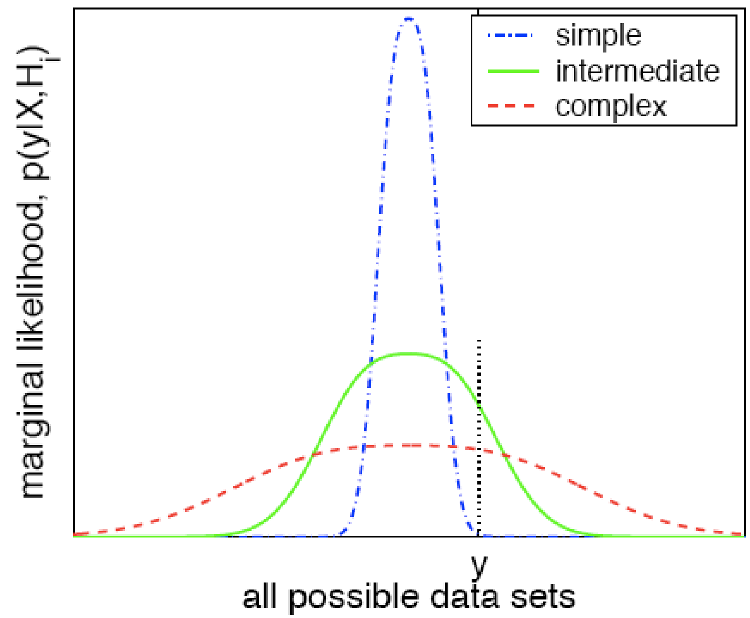
$$p(\mathcal{H}_i|\mathbf{t}, X) = \frac{p(\mathbf{t}|X, \mathcal{H}_i)p(\mathcal{H}_i)}{p(\mathbf{t}|X)}$$

- where
  - Again, the denominator of the previous level $p(\mathbf{t}\,|\,X,\mathcal{H}_i)$ plays the role of the likelihood.
  - $p(\mathcal{H}_i)$ is the prior of the model structure.
  - Denominator (normalizing constant) is given by:

$$p(\mathbf{t}|X) = \sum_i p(\mathbf{t}|X, \mathcal{H}_i)p(\mathcal{H}_i)$$

Bayesian Model Selection

- Discussion
  - Marginal likelihood is main difference to non-Bayesian methods
  - It automatically incorporates a trade-off between the model fit and the model complexity:
    - A simple model can only account for a limited range of possible sets of target values - if a simple model fits well, it obtains a high posterior.
    - A complex model can account for a large range of possible sets of target values - therefore, it can never attain a very high posterior.

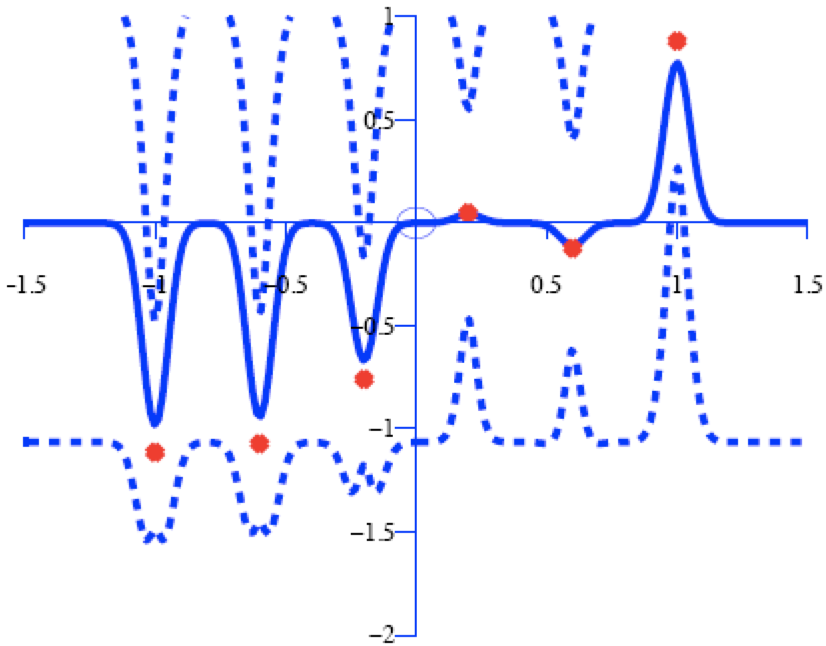


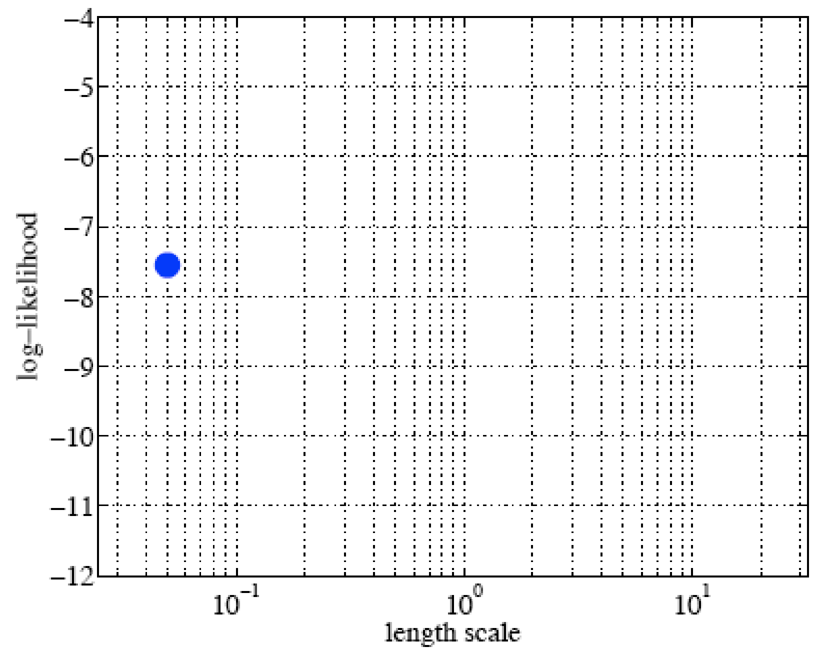
Bayesian Model Selection

- Computational issues
  - Requires the evaluation of several integrals, which may or may not be analytically tractable, depending on details of the models.
  - In general, one may have to resort to analytic approximations or MCMC methods. (→Lecture 7)

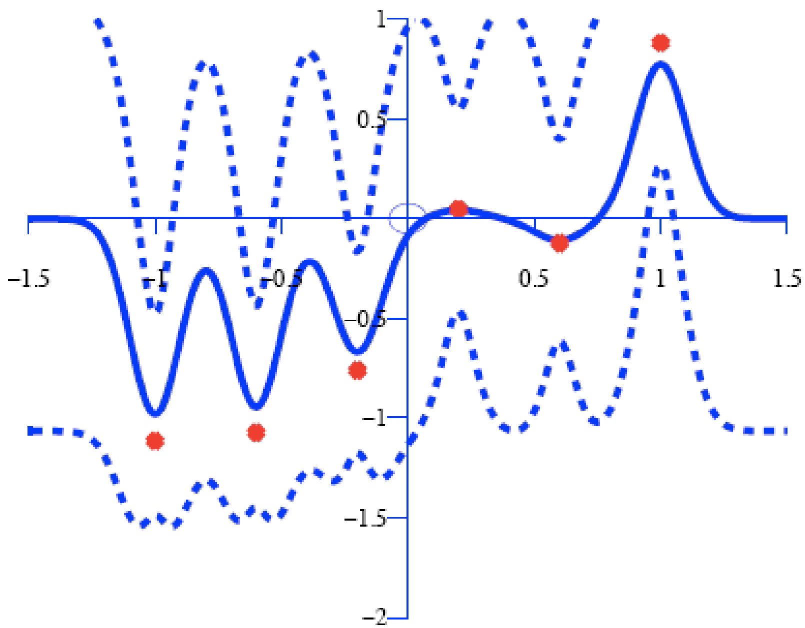
- Model selection for GP regression
  - GP regression models with Gaussian noise are an (important) exception:
    - Integrals over the parameters are analytically tractable and
    - At the same time, the models are flexible.

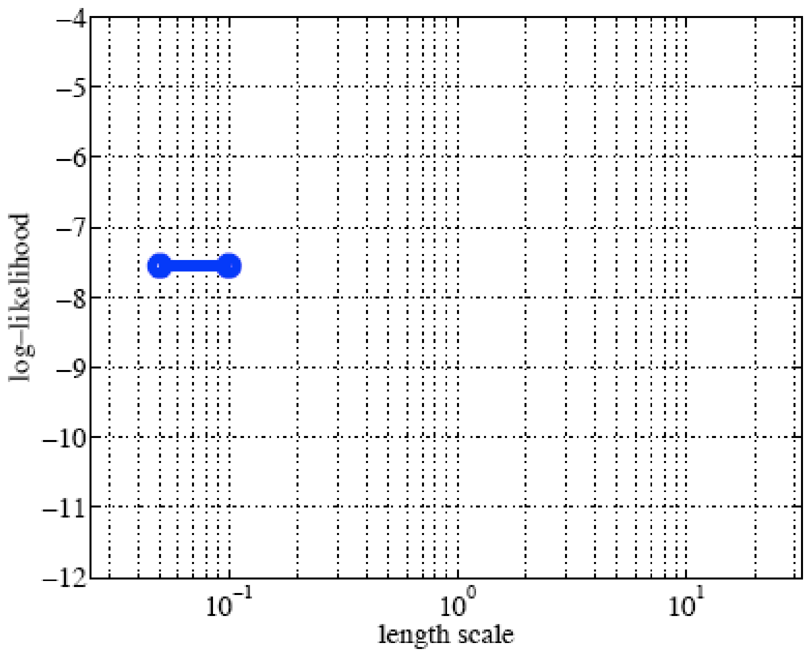
Example


1
0.5
-1.5
-1
-0.5
0.5
1
1.5
-0.5
-1
-1.5
-2

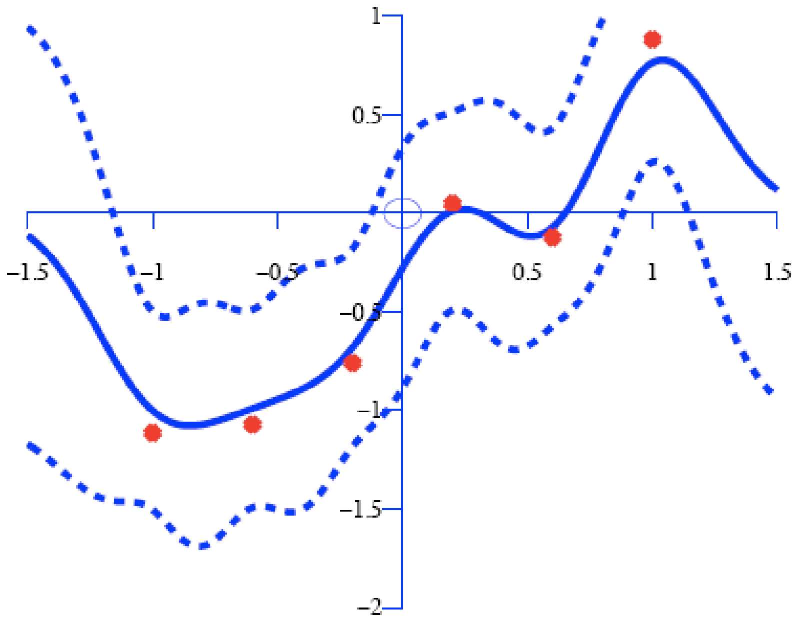

−4
−5
−6
−7
−8
−9
−10
−11
−12
log−likelihood
10^{-1}
10^{0}
10^{1}
length scale

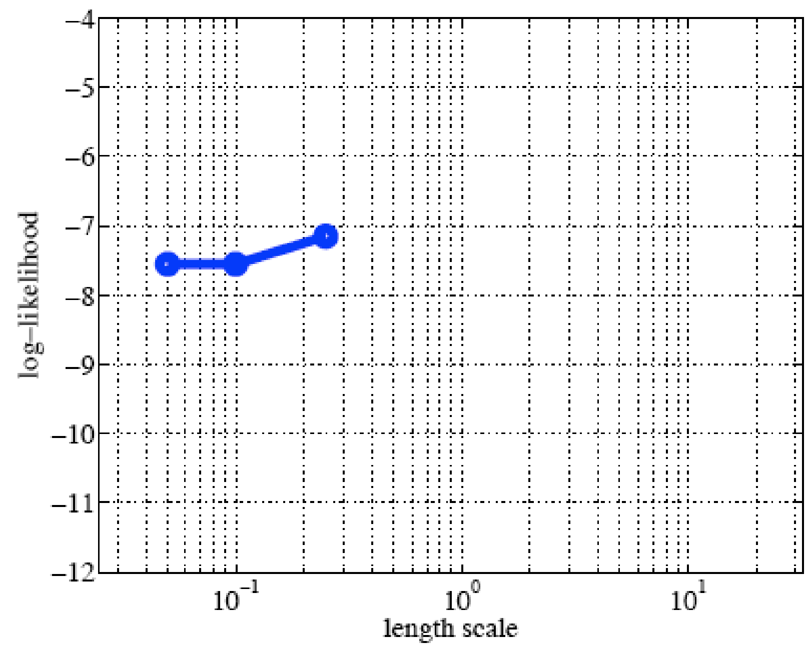
Example


1
0.5
−0.5
−1
−1.5
−2
−1.5
−1
−0.5
0.5
1
1.5

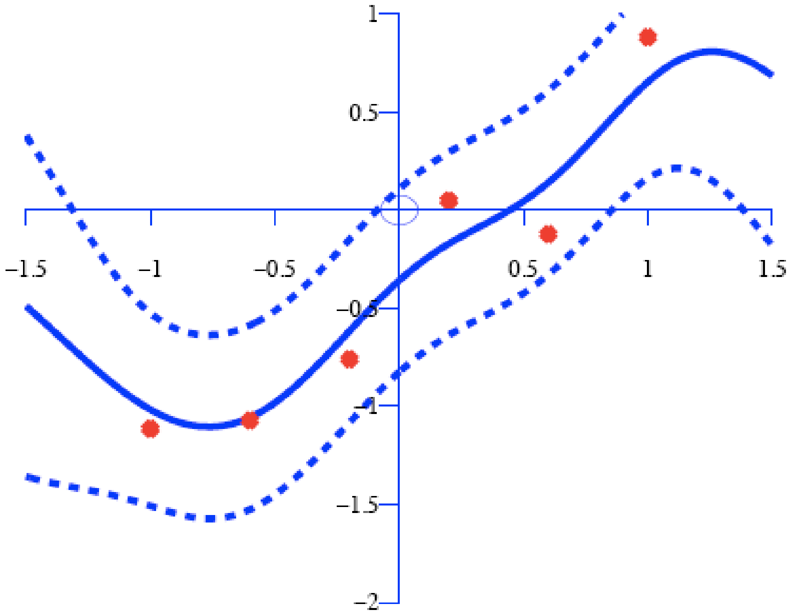

−4
−5
−6
−7
−8
−9
−10
−11
−12
log−likelihood
10^{-1}
10^{0}
10^{1}
length scale

Example

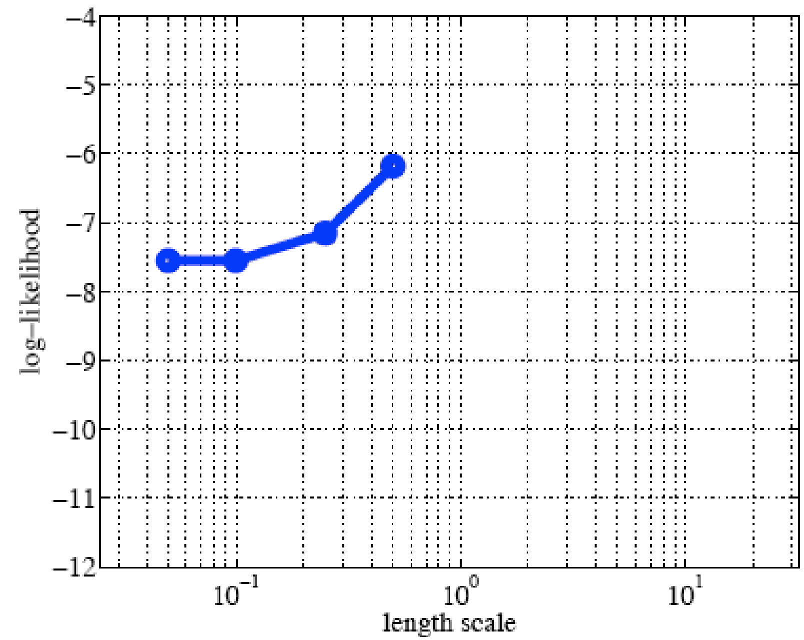

1
0.5
-1.5
-1
-0.5
0.5
1
1.5
-0.5
-1
-1.5
-2


-4
-5
-6
-7
-8
-9
-10
-11
-12
log-likelihood
10^{-1}
10^{0}
10^{1}
length scale

Example

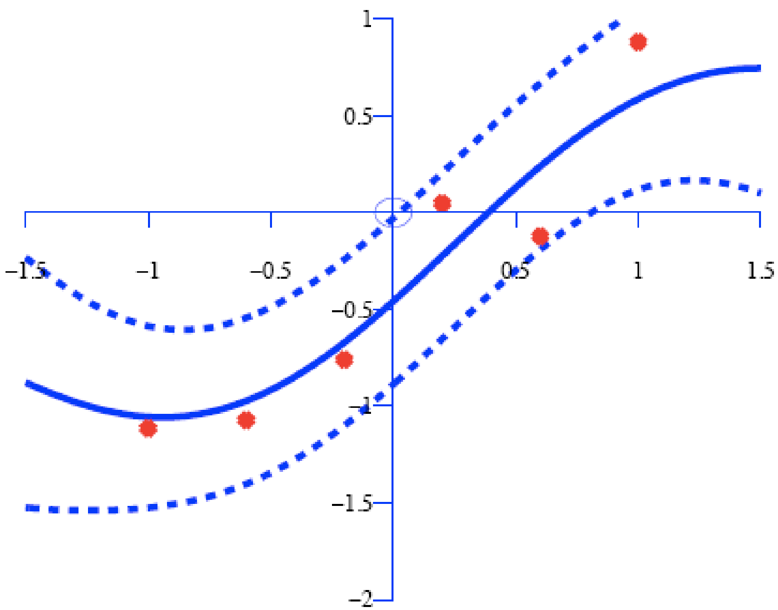


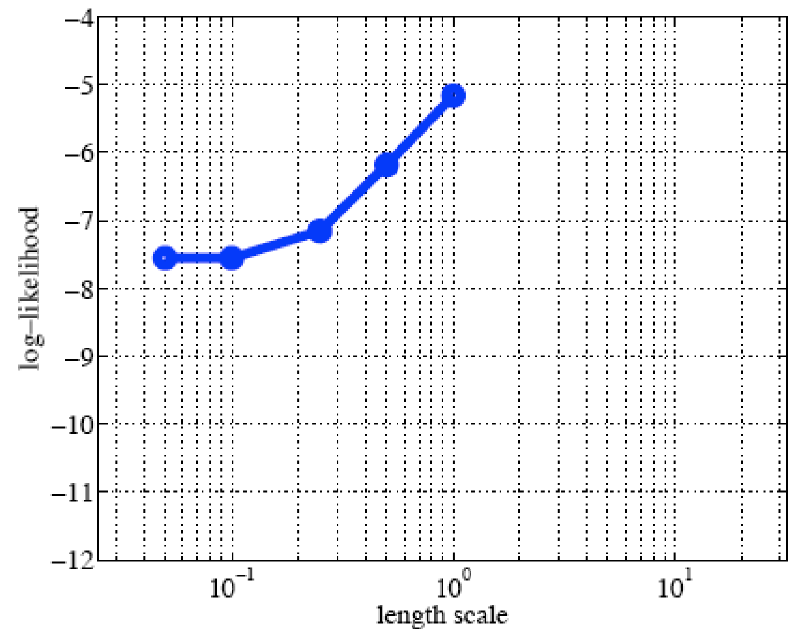
1
0.5
−0.5
−1
−1.5
−2
−1.5
−1
−0.5
0.5
1
1.5



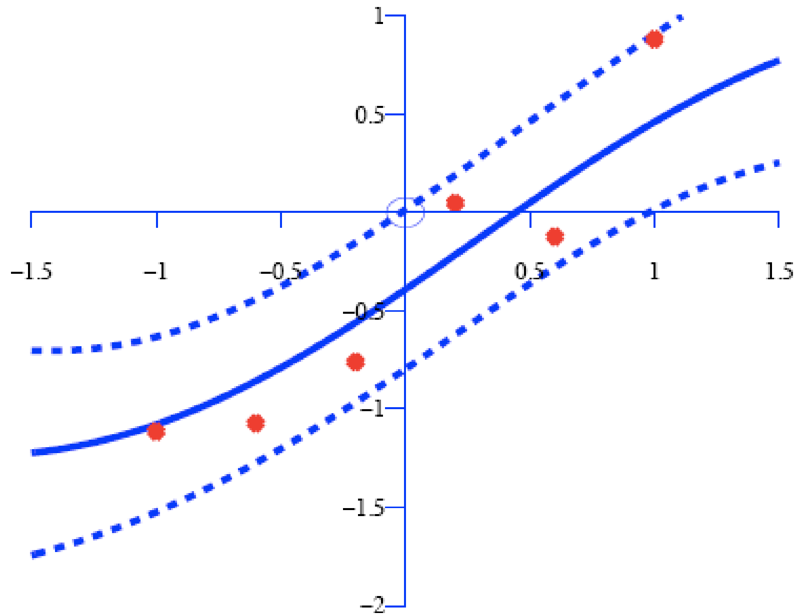
log−likelihood
length scale
−4
−5
−6
−7
−8
−9
−10
−11
−12
10^−1
10^0
10^1

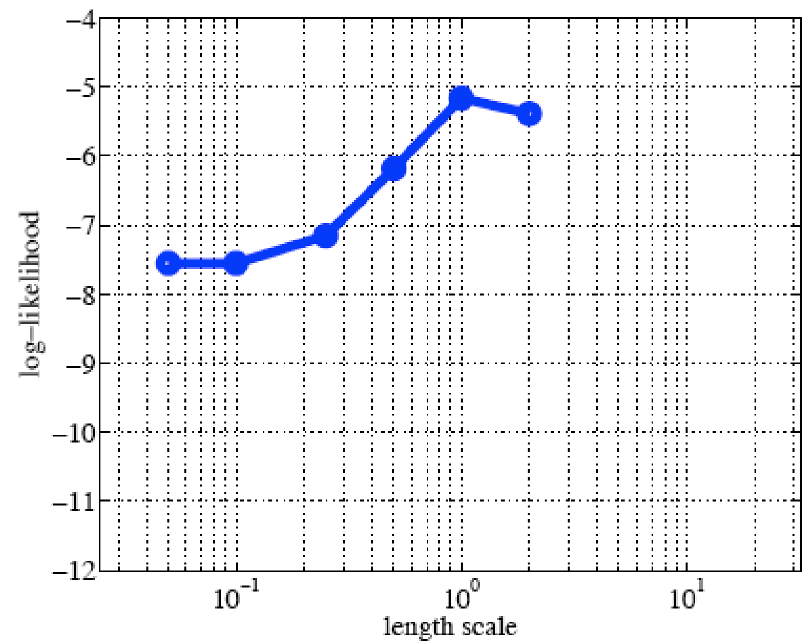
Example


1
0.5
−0.5
−1
−1.5
−2
−1.5
−1
−0.5
0.5
1
1.5

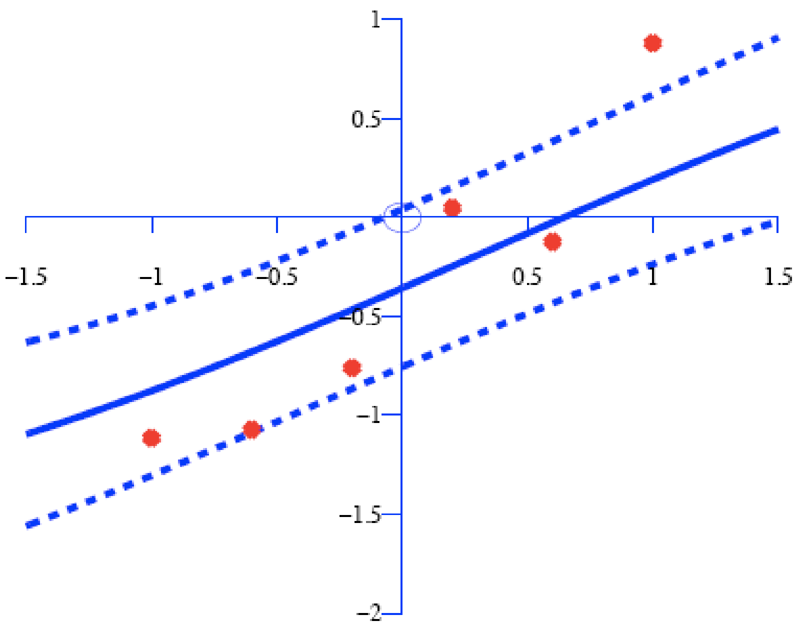

−4
−5
−6
−7
−8
−9
−10
−11
−12
log−likelihood
10^{-1}
10^{0}
10^{1}
length scale

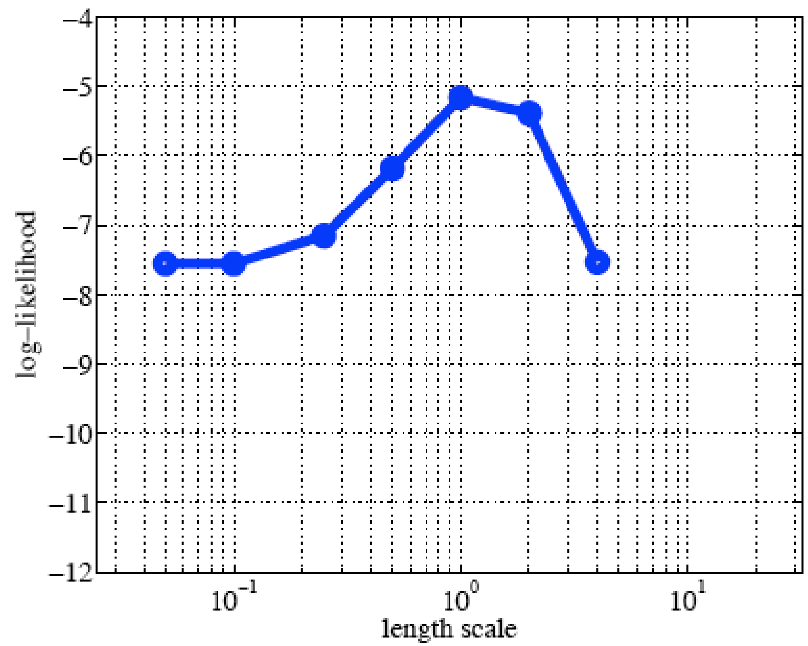
Example


1
0.5
−1.5
−1
−0.5
0.5
1
1.5
−0.5
−1
−1.5
−2

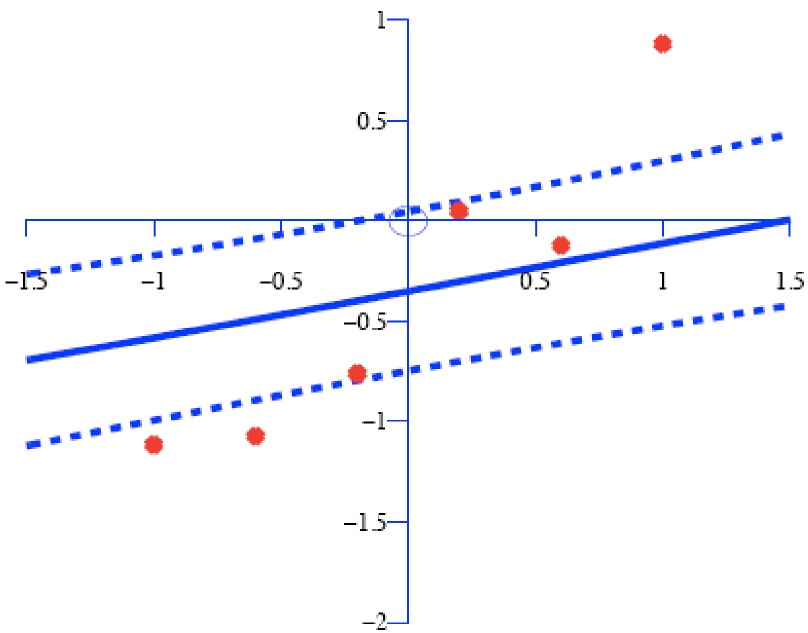

−4
−5
−6
−7
−8
−9
−10
−11
−12
log−likelihood
10^{-1}
10^{0}
10^{1}
length scale

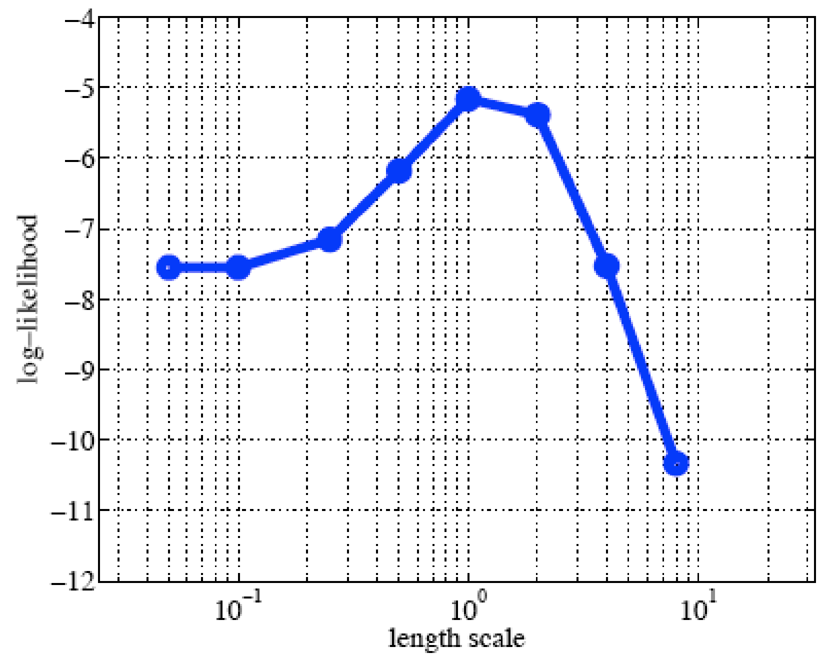
Example


1
0.5
−1.5
−1
−0.5
0.5
1
1.5
−0.5
−1
−1.5
−2


−4
−5
−6
−7
−8
−9
−10
−11
−12
log−likelihood
10^{-1}
10^{0}
10^{1}
length scale

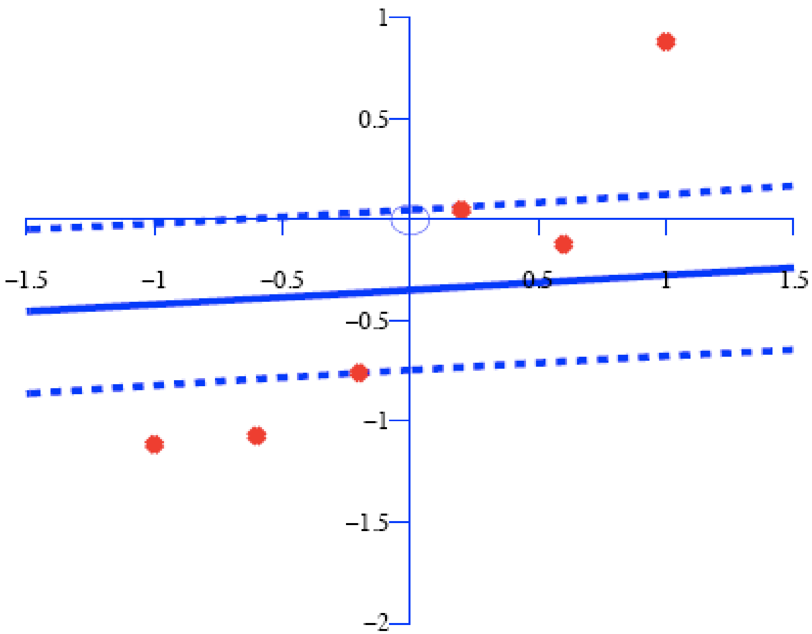
Example

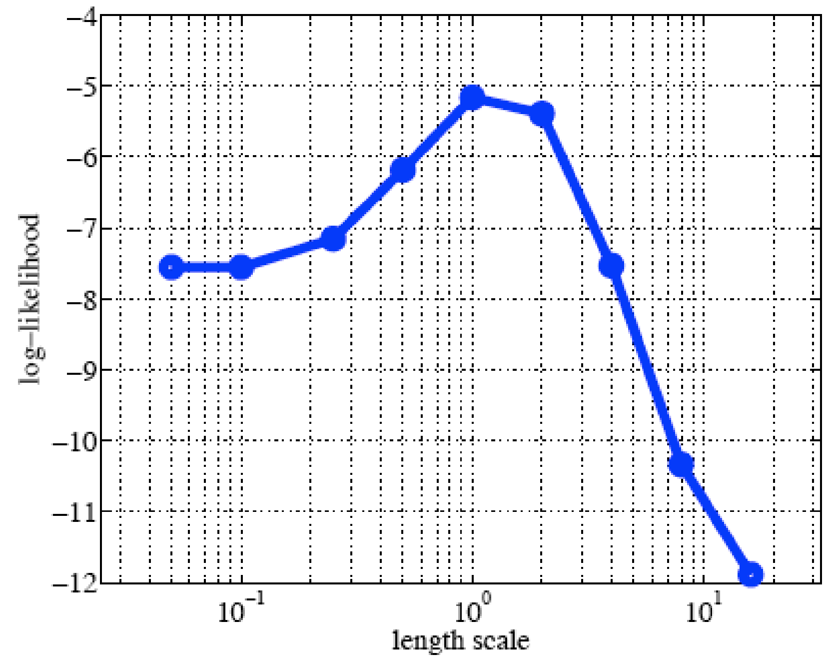

1
0.5
-1.5
-1
-0.5
0.5
1
1.5
-0.5
-1
-1.5
-2


-4
-5
-6
-7
-8
-9
-10
-11
-12
log-likelihood
10^{-1}
10^{0}
10^{1}
length scale

RWTH AACHEN UNIVERSITY

Example

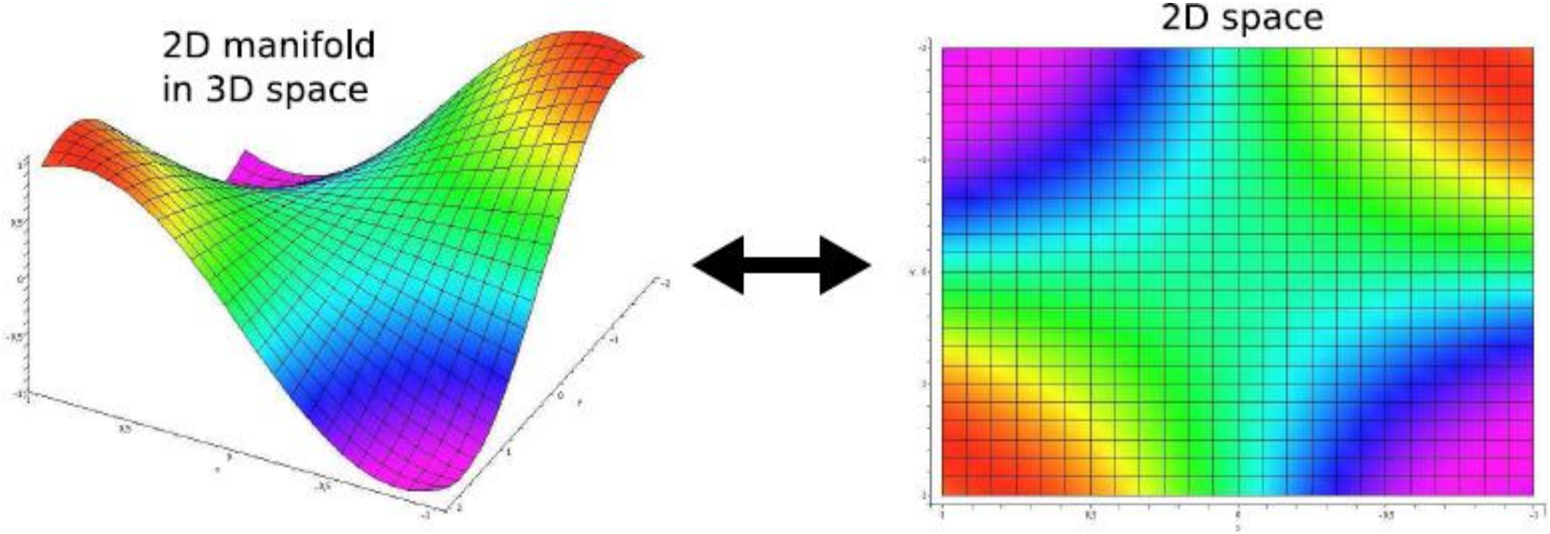


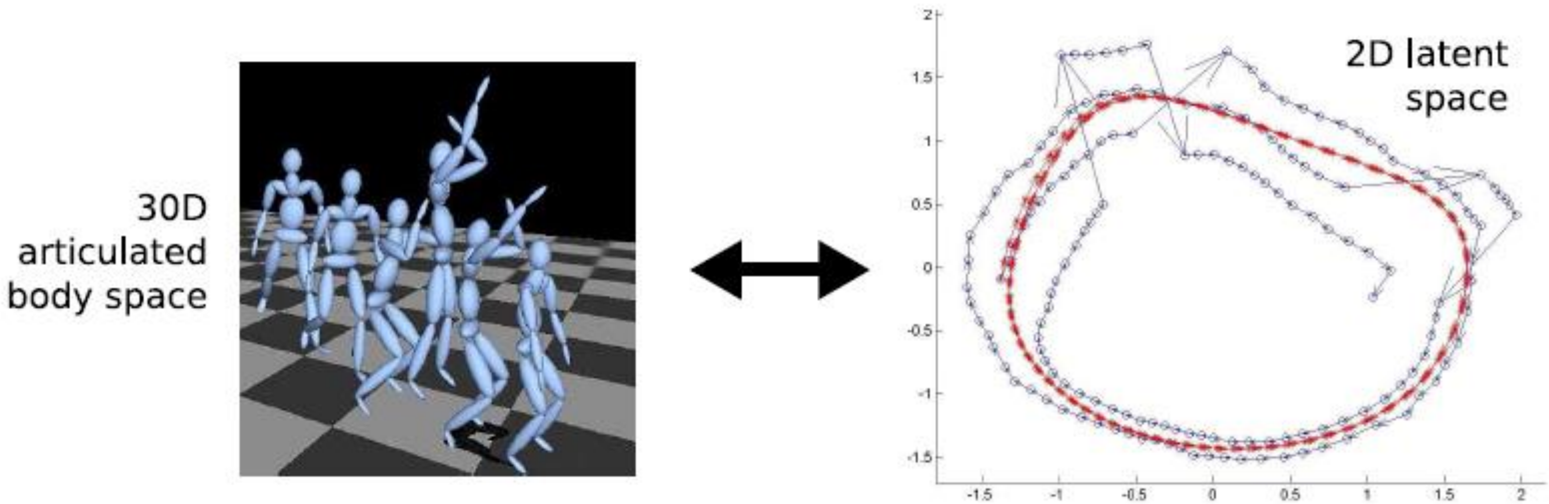


Topics of This Lecture

- Kernels
  - Recap: Kernel trick
  - Constructing kernels
- Gaussian Processes
  - Recap: Definition
  - Prediction with noise-free observations
  - Prediction with noisy observations
  - GP Regression
  - Influence of hyperparameters
- Learning Gaussian Processes
  - Bayesian Model Selection
  - Model selection for Gaussian Processes
- **Applications**

Application: Non-Linear Dimensionality Reduction

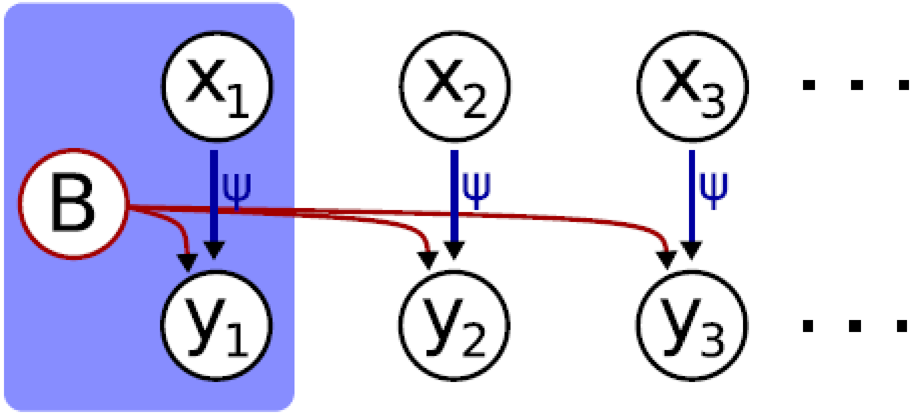




Slide credit: Andreas Geiger

Gaussian Process Latent Variable Model

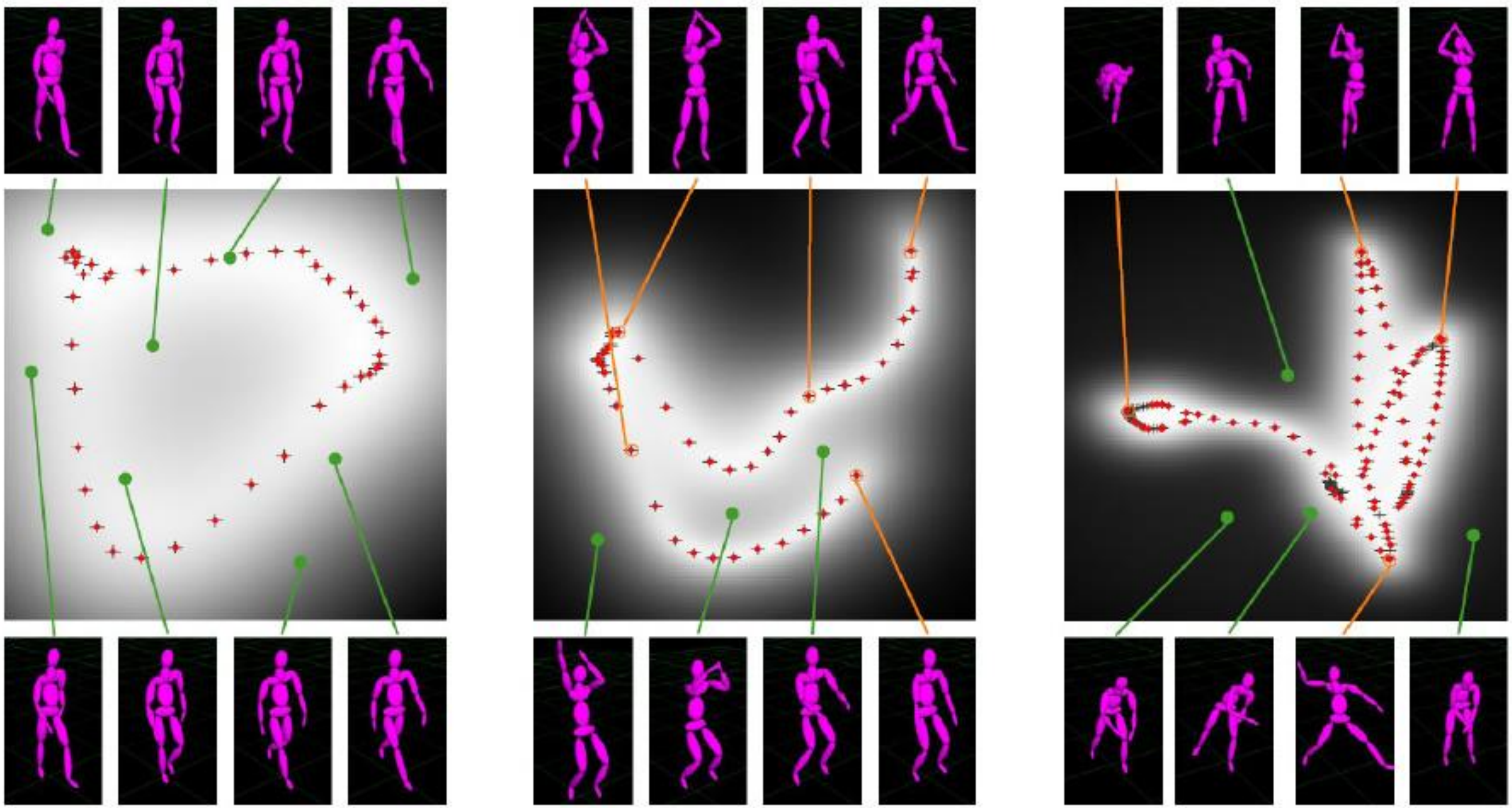
- At each time step t, we express our observations $\mathbf{y}$ as a combination of basis functions ψ of latent variables $\mathbf{x}$.



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

- This is modeled as a Gaussian process…

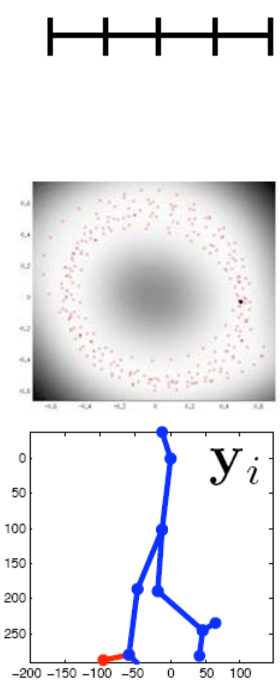
Example: Style-based Inverse Kinematics



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].



Time (frame #)

Latent space

Configuration

$$\mathbf{T} = [t_i \in \mathbb{R}]$$

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

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

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

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

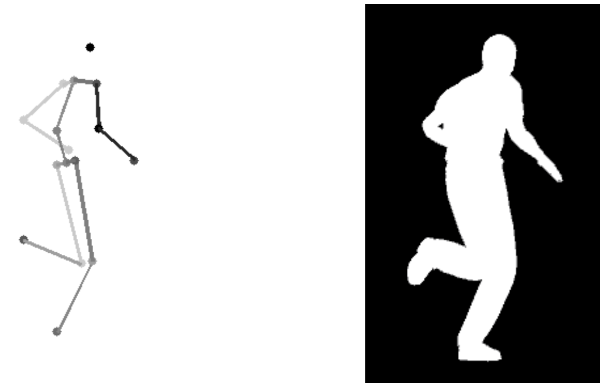
Training

[Andriluka, Roth, Schiele, CVPR'08]

Application: Mapping b/w Pose and Appearance

- Appearance prediction
  - Regression problem
  - High-dimensional data on both sides

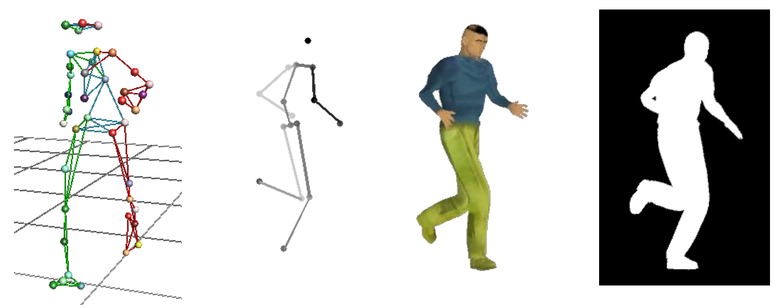
  ⇒ Low-dim. representation needed for learning!



- 3D joint locations
- 60-dim.

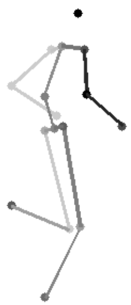
- segm. image
- ~2500-dim.

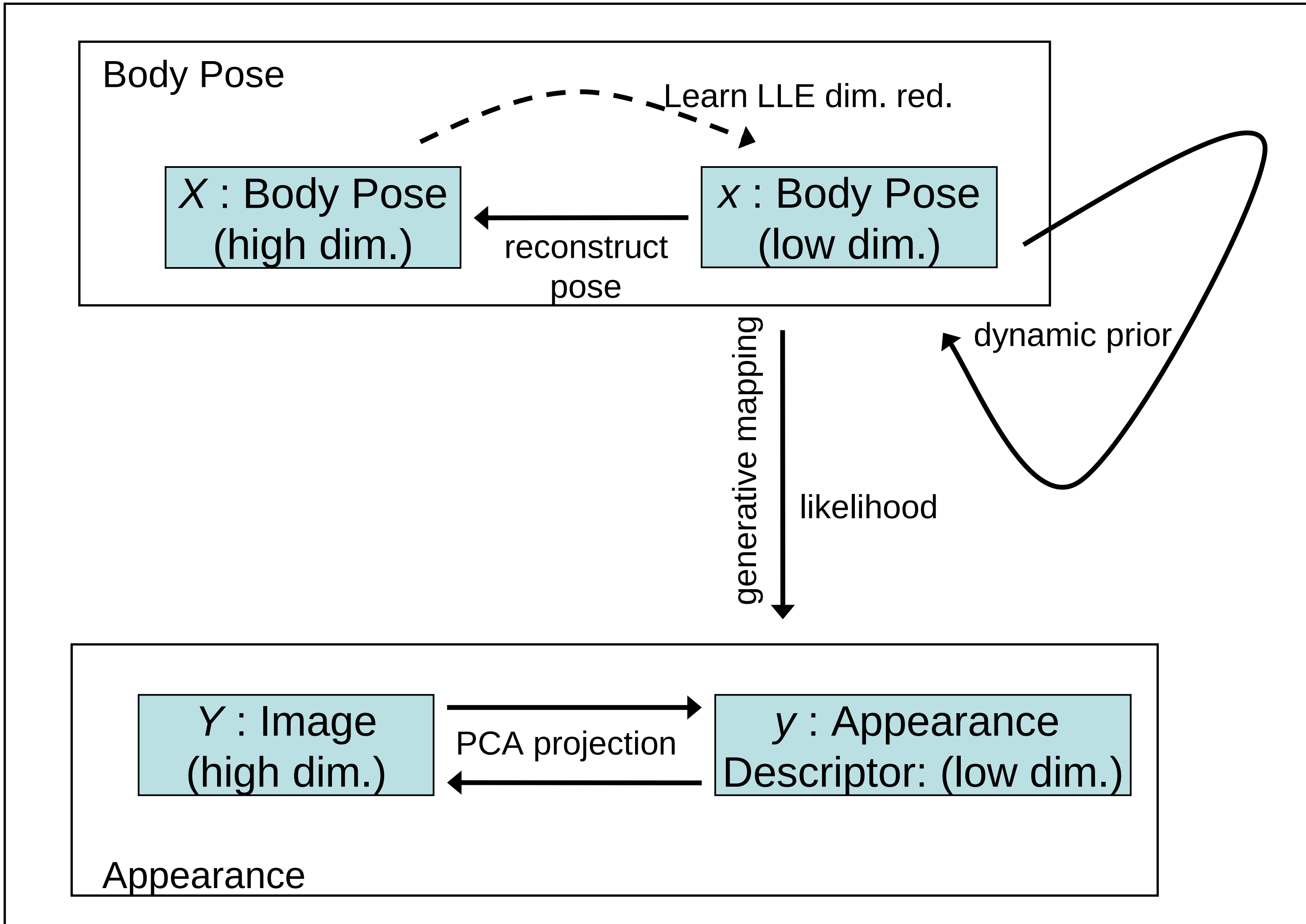
- Training with Motion-capture data possible
  - Synthesized silhouettes for training
  - Background subtraction for test



[Jaeggli, Koller-Meier, Van Gool, ACCV'07]

Learning a Generative Mapping

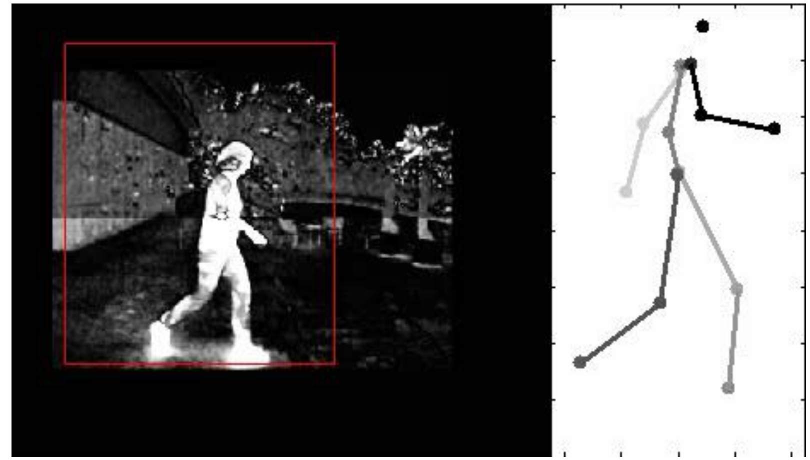


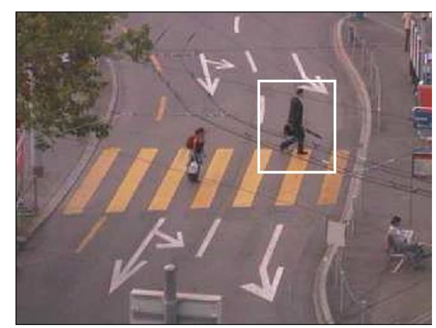


[Jaeggli, Koller-Meier, Van Gool, ACCV'07]

Experimental Results

- Difficulties
  - Changing viewpoints
  - Low resolution (50 px)
  - Compression artifacts
  - Disturbing objects



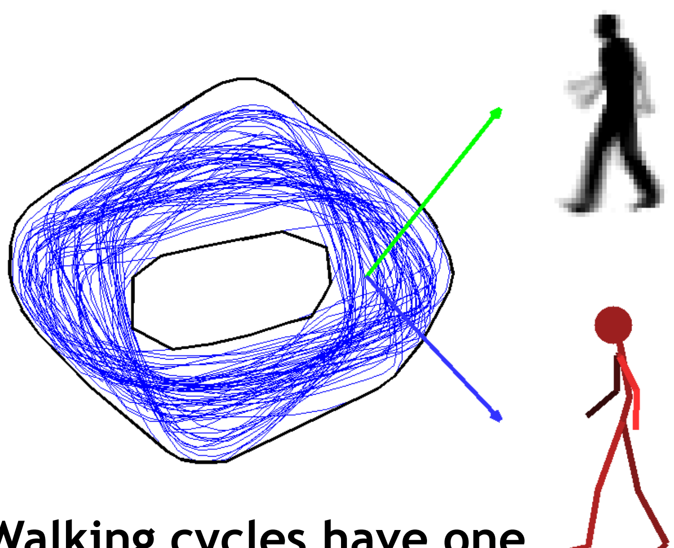


Original video

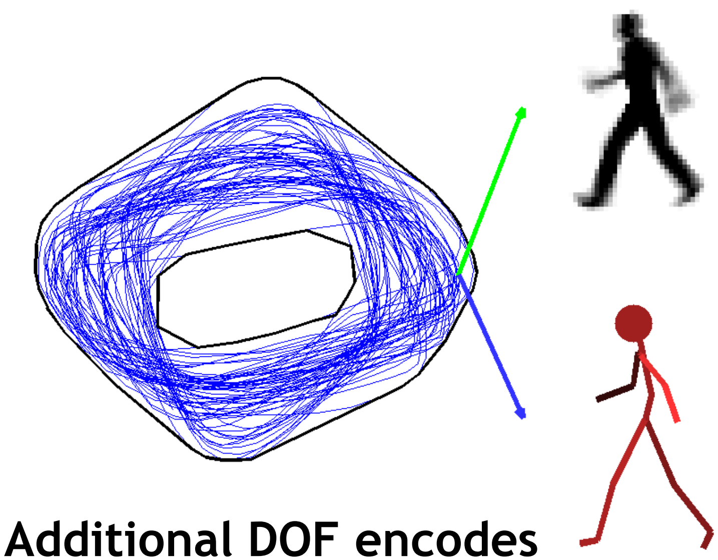
[Jaeggli, Koller-Meier, Van Gool, ACCV'07]

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]



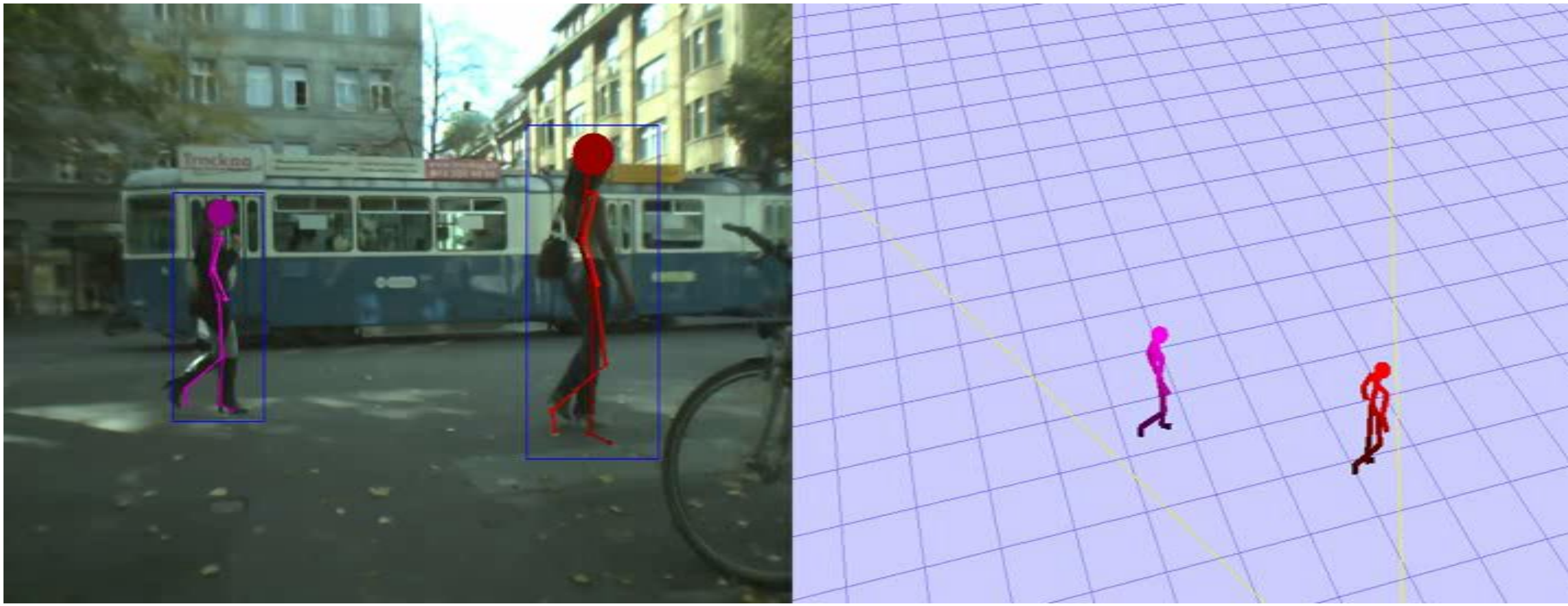
Walking cycles have one main (periodic) DOF



Additional DOF encodes „walking style“

[Gammeter, Ess, Leibe, Schindler, Van Gool, ECCV’08]

Results

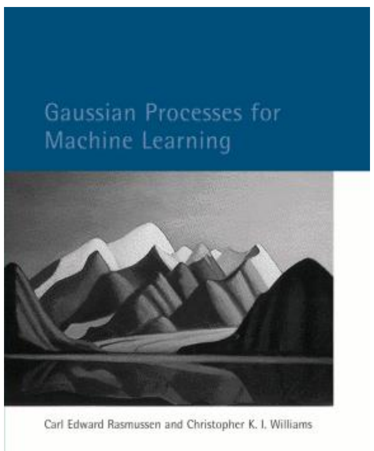


454 frames (~35 sec)
23 Pedestrians
20 detected by multi-body tracker

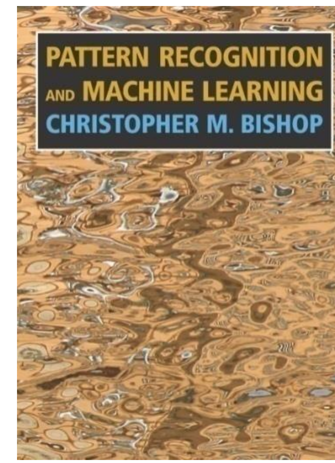
[Gammeter, Ess, Leibe, Schindler, Van Gool, ECCV'08]

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 3 and 5 of the book by Rasmussen & Williams (also available online: http://www.gaussianprocess.org/gpml/)