Machine Learning

Volker Roth

Department of Mathematics & Computer Science
University of Basel

Chapter 8: Gaussian Processes

- The use of the Gaussian distribution in ML
  - Properties of the **multivariate Gaussian distribution**
  - Random variables $\rightarrow$ random vectors $\rightarrow$ **stochastic processes**
  - **Gaussian processes for regression**
  - **Model Selection**
  - **Gaussian processes for classification**
- Relation to **kernel models** (e.g. SVMs)
- Relation to **neural networks.**

Kernel Ridge Regression

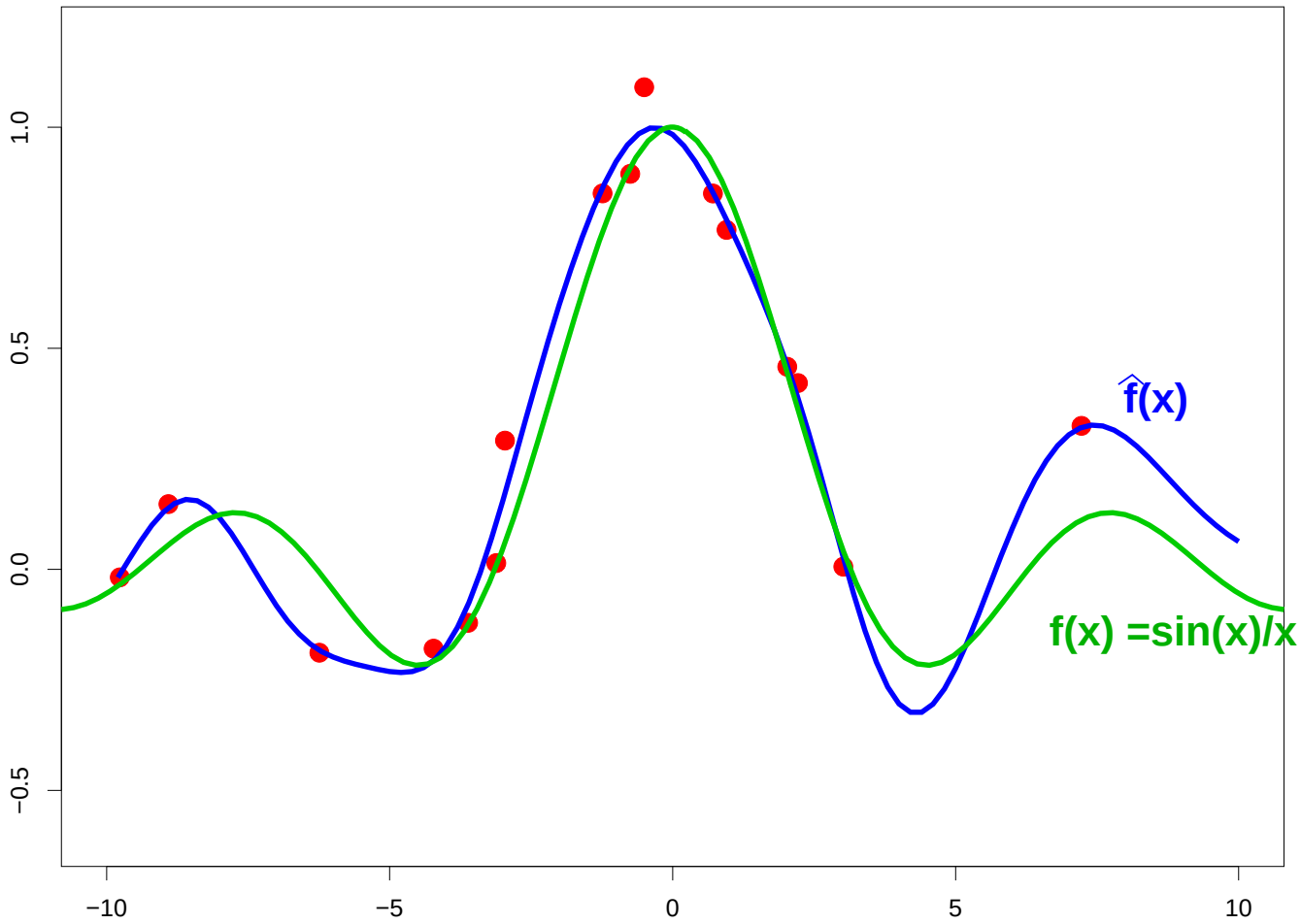
- **Kernelized ridge regression:** $\hat{\boldsymbol{w}} = (X^t X + \lambda I)^{-1} X^t \boldsymbol{y}$.
- Matrix inversion lemma: $(I + UV)^{-1} U = U(I + VU)^{-1}$
- Define new variables α_i:

$$\begin{aligned}\hat{\boldsymbol{w}} &= (X^t X + \lambda I)^{-1} X^t \boldsymbol{y} \\ &= X^t \underbrace{(XX^t + \lambda I)^{-1} \boldsymbol{y}}_{\hat{\boldsymbol{\alpha}}} = \sum_{i=1}^{n} \hat{\alpha}_i \boldsymbol{x}_i.\end{aligned}$$

- **Predictions for new $\boldsymbol{x}_*$:**

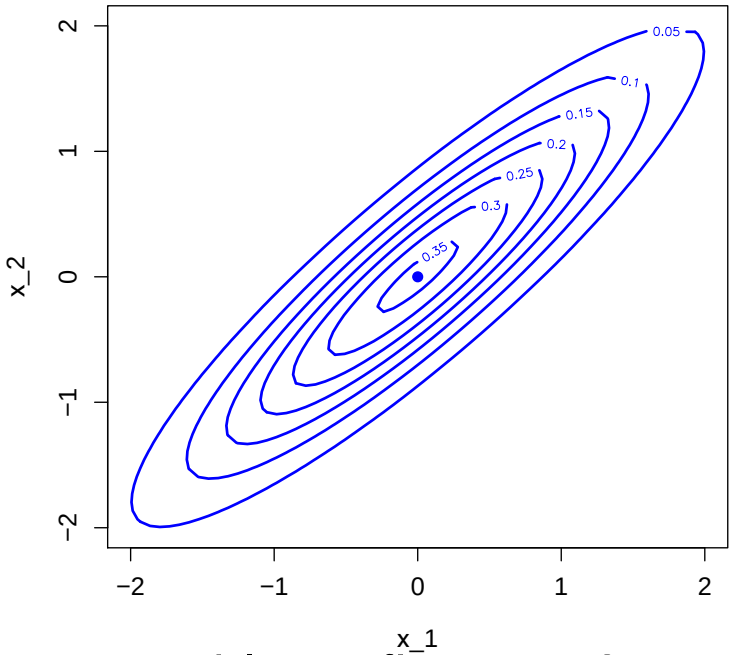
$$\hat{f}(\boldsymbol{x}_*) = \hat{\boldsymbol{w}}^t \boldsymbol{x}_* = \sum_{i=1}^{n} \hat{\alpha}_i \boldsymbol{x}_i^t \boldsymbol{x}_* = \sum_{i=1}^{n} \hat{\alpha}_i k(\boldsymbol{x}_i, \boldsymbol{x}_*).$$

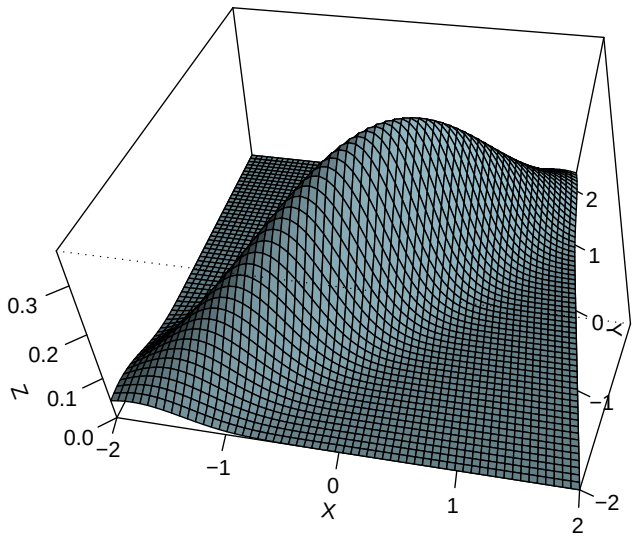
Kernel Ridge Regression



Kernel function: $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\frac{1}{2l^2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$

How can we make use of the Gaussian distribution?





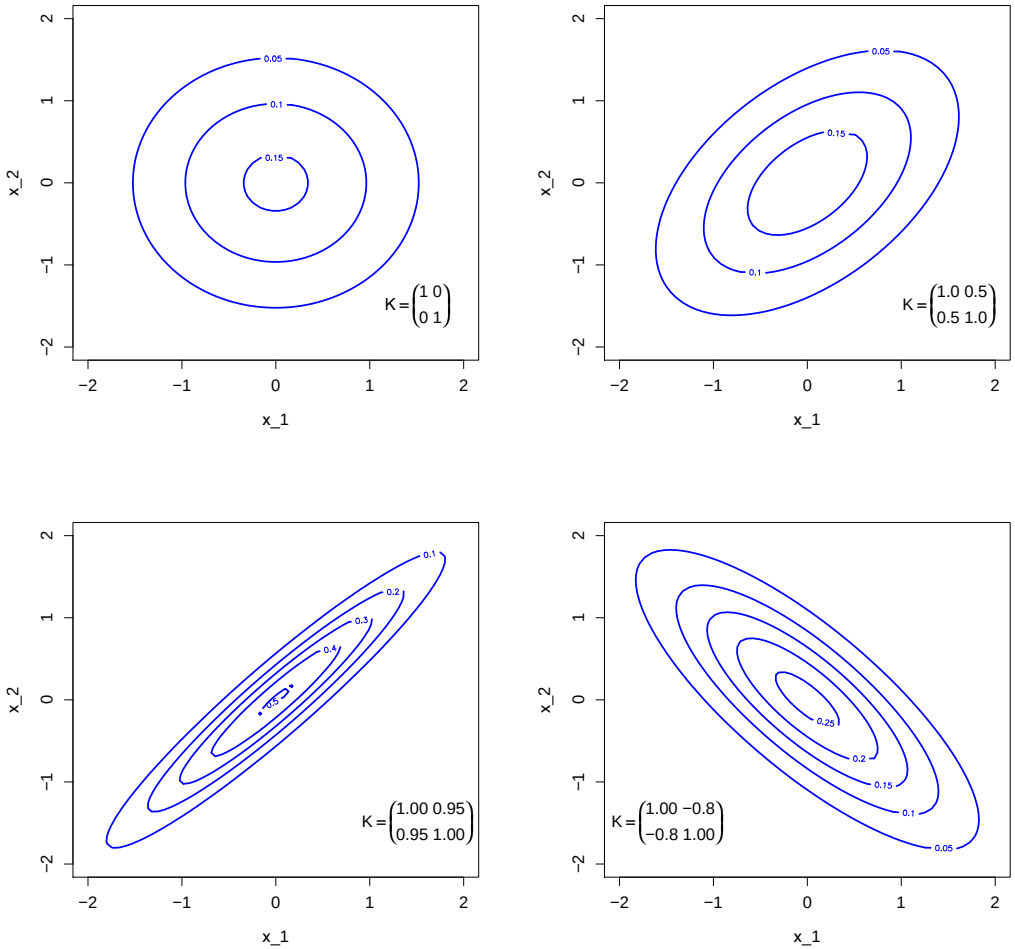
- Is it possible to fit a **nonlinear regression line** with the "boring" Gaussian distribution?
- **Yes**, but we need to introduce the concept of **Gaussian Processes**!

The 2D Gaussian distribution

2D Gaussian: $P(\boldsymbol{y}; \boldsymbol{\mu} = \mathbf{0}, \Sigma = K) = \frac{1}{\sqrt{2\pi|K|}} \exp(-\frac{1}{2}\boldsymbol{y}^t K^{-1}\boldsymbol{y})$

Covariance
(also written "co-variance")
is a measure of how much **two random variables vary together**:

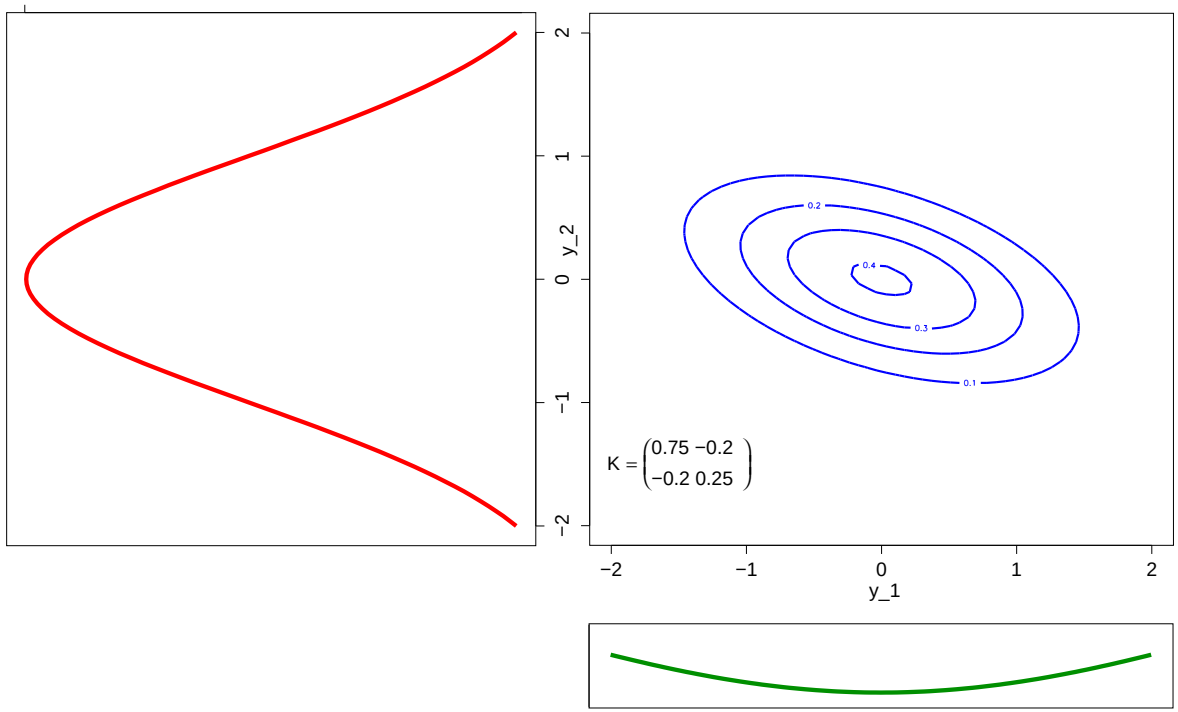
- **+1**: perfect linear coherence,
- **-1**: perfect negative linear coherence,
- **0**: no linear coherence.



Properties of the Multivariate Gaussian distribution

$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, K)$. Let $\boldsymbol{y} = \begin{pmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{pmatrix}$ and $K = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix}$.

Then $\boldsymbol{y}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, K_{11})$ and $\boldsymbol{y}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, K_{22})$.

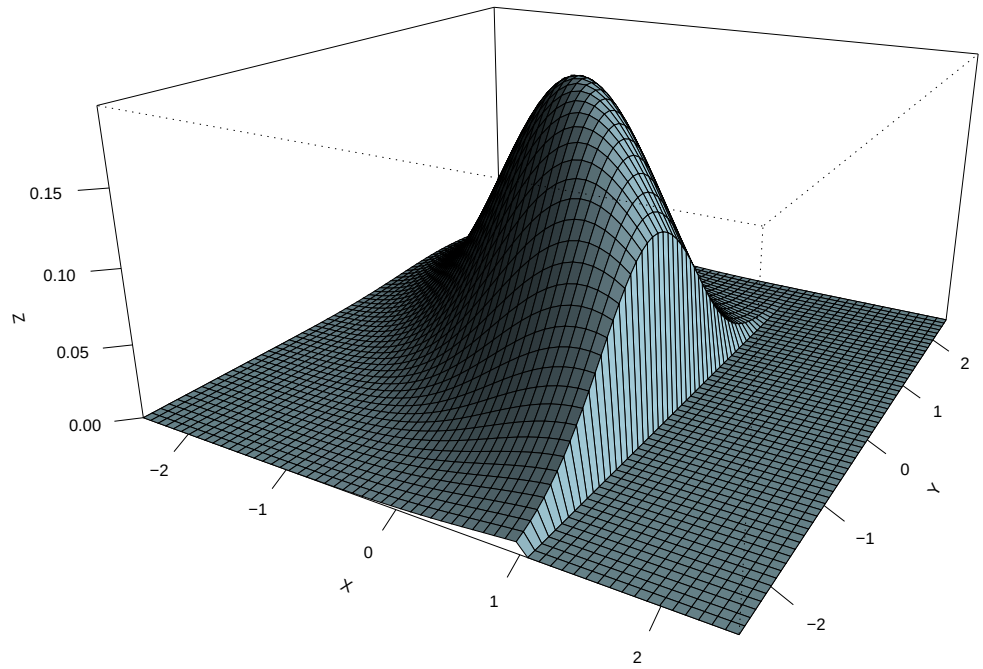


Marginals of Gaussians are again Gaussian!

Properties of the Multivariate Gaussian distribution (2)

$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, K)$. Let $\boldsymbol{y} = \begin{pmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{pmatrix}$ and $K = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix}$.

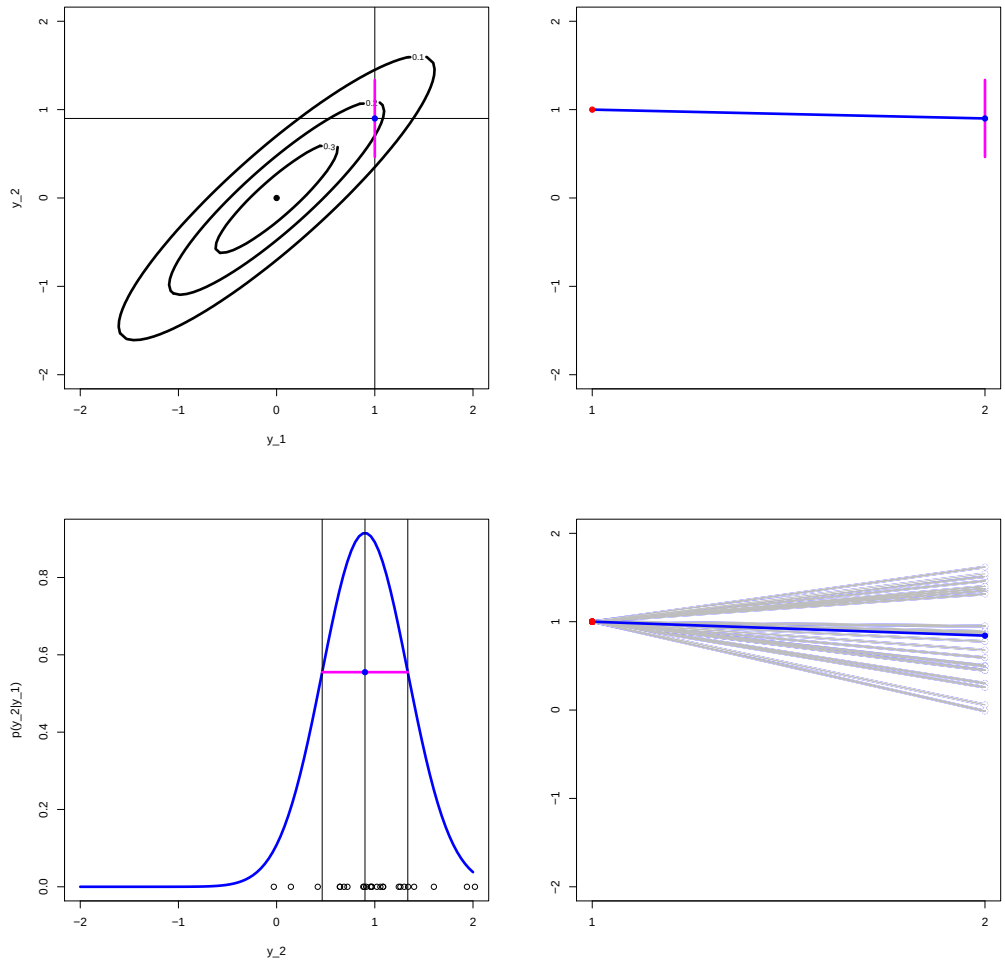
Then $\boldsymbol{y}_2 | \boldsymbol{y}_1 \sim \mathcal{N}(\boldsymbol{\mu}_2 + K_{21} K_{11}^{-1}(\boldsymbol{y}_1 - \boldsymbol{\mu}_1), K_{22} - K_{21} K_{11}^{-1} K_{12})$.



Conditionals of Gaussians are again Gaussian!

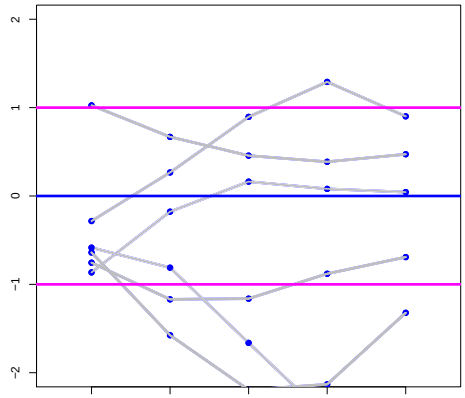
2D Gaussians: a new visualization

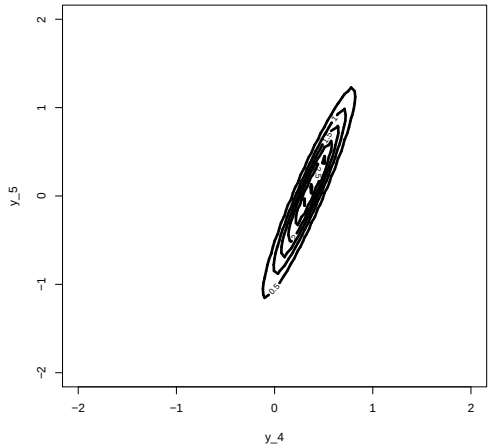
- **top left:** mean and $\pm$std.dev. of $p(y_2|y_1 = 1)$.
- **bottom left:** $p(y_2|y_1 = 1)$ and samples drawn from it.
- **top right:** x-axis: indices $(1, 2)$ of dimensions, y-axis: density in each component. Shown are $y_1 = 1$ and the conditional mean $\bar{p}(y_2|y_1 = 1)$ and std.dev.
- **bottom right:** samples drawn from above model.

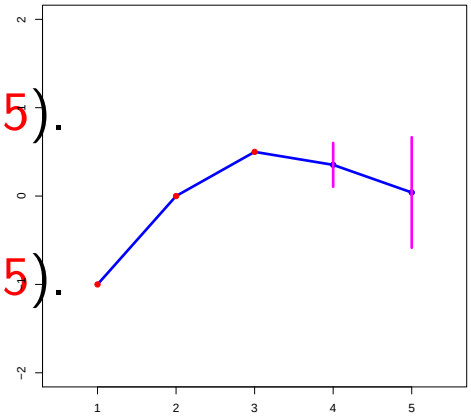


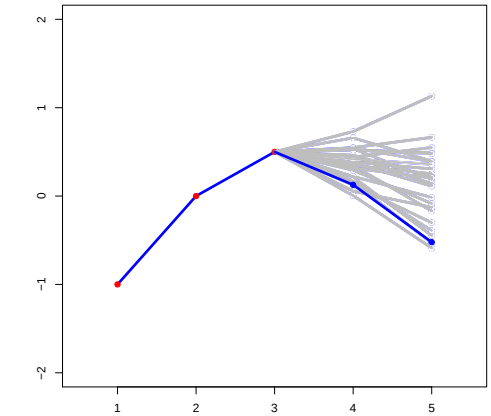
Visualizing high-dimensional Gaussians

- **top left:** 6 samples drawn from 5-dimensional Gaussian with zero mean (indicated by blue line). $\sigma = 1$ (magenta line).
- **bottom left:** Conditional mean and std.dev of $p(y_4, y_5 | y_1 = -1, y_2 = 0, y_3 = 0.5)$.
- **top right:** contour lines of $p(y_4, y_5 | y_1 = -1, y_2 = 0, y_3 = 0.5)$.
- **bottom right:** samples drawn from above model.



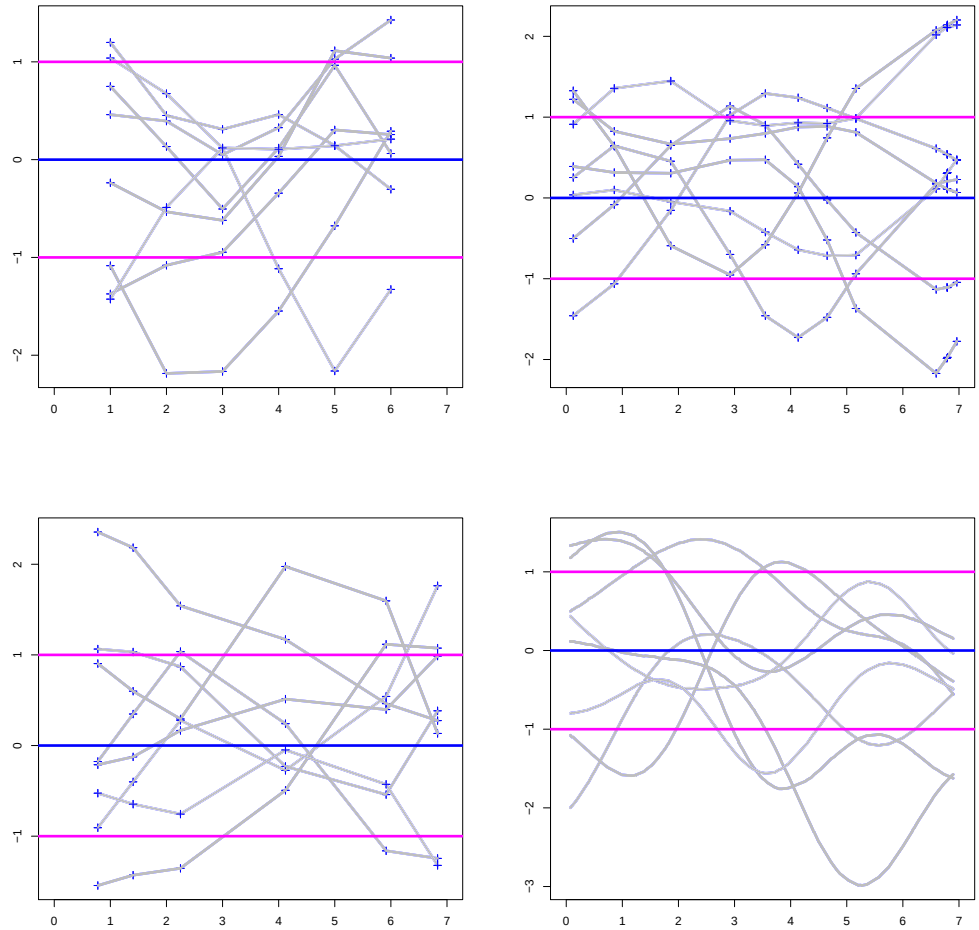




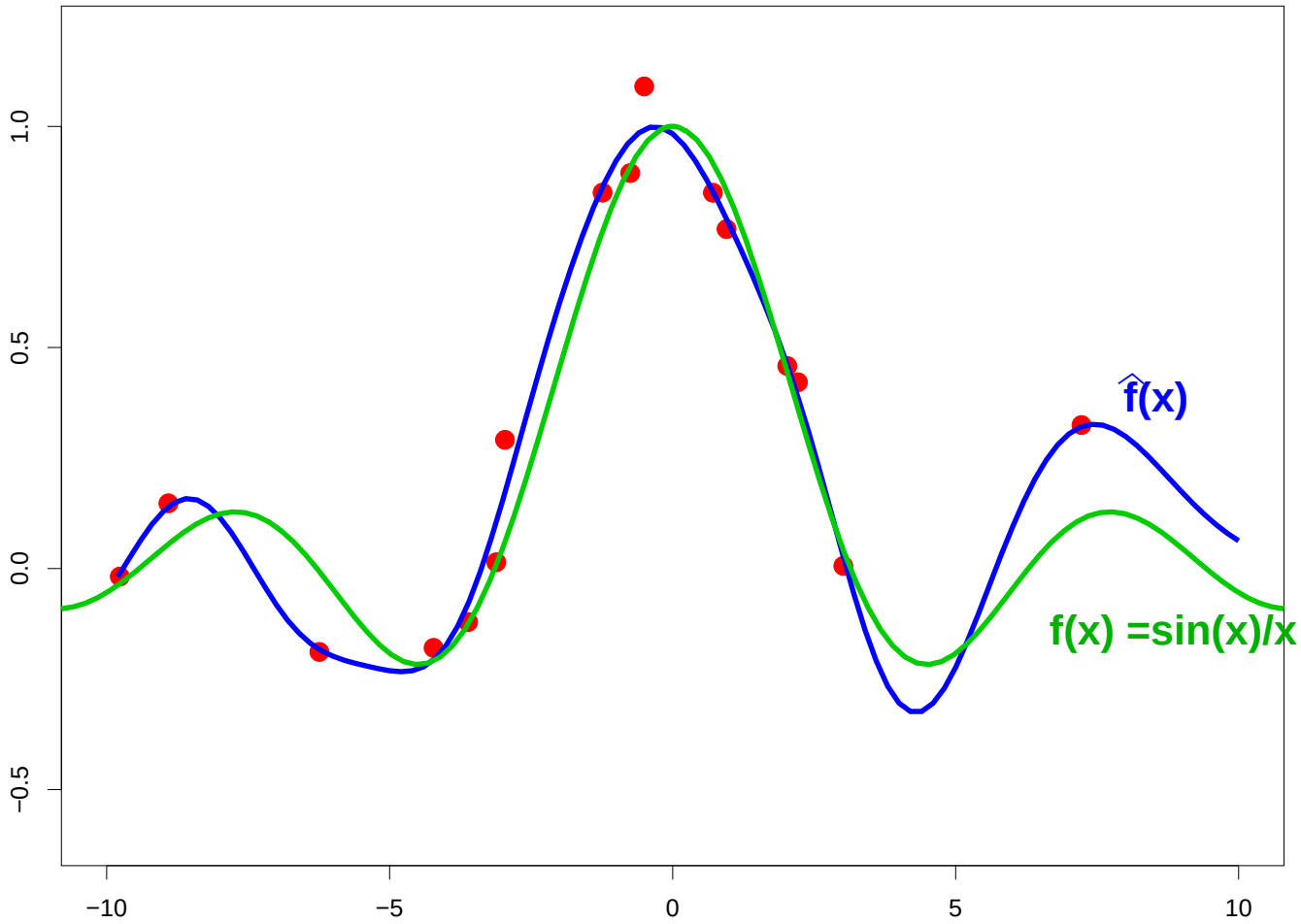


From covariance matrices to Gaussian processes

- **top left:** 8 samples, 6 dim.
  x-axis: dimension-indices.
- **bottom left:** 8 samples,
  viewed as values $y = f(\boldsymbol{x})$.
  Construction: choose 6
  input points $\boldsymbol{x}_i$ at random
  $\rightsquigarrow$ build covariance matrix K
  with **covariance function**
  $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\frac{1}{2l^2}\|\boldsymbol{x}-\boldsymbol{x}'\|^2)$
  $\rightsquigarrow$ draw $\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{0}, K)$
  $\rightsquigarrow$ plot as function of inputs.
- **top right:** same for 12 inputs
- **bottom right:** 100 inputs



This looks similar to Kernel Regression...



Gaussian Processes

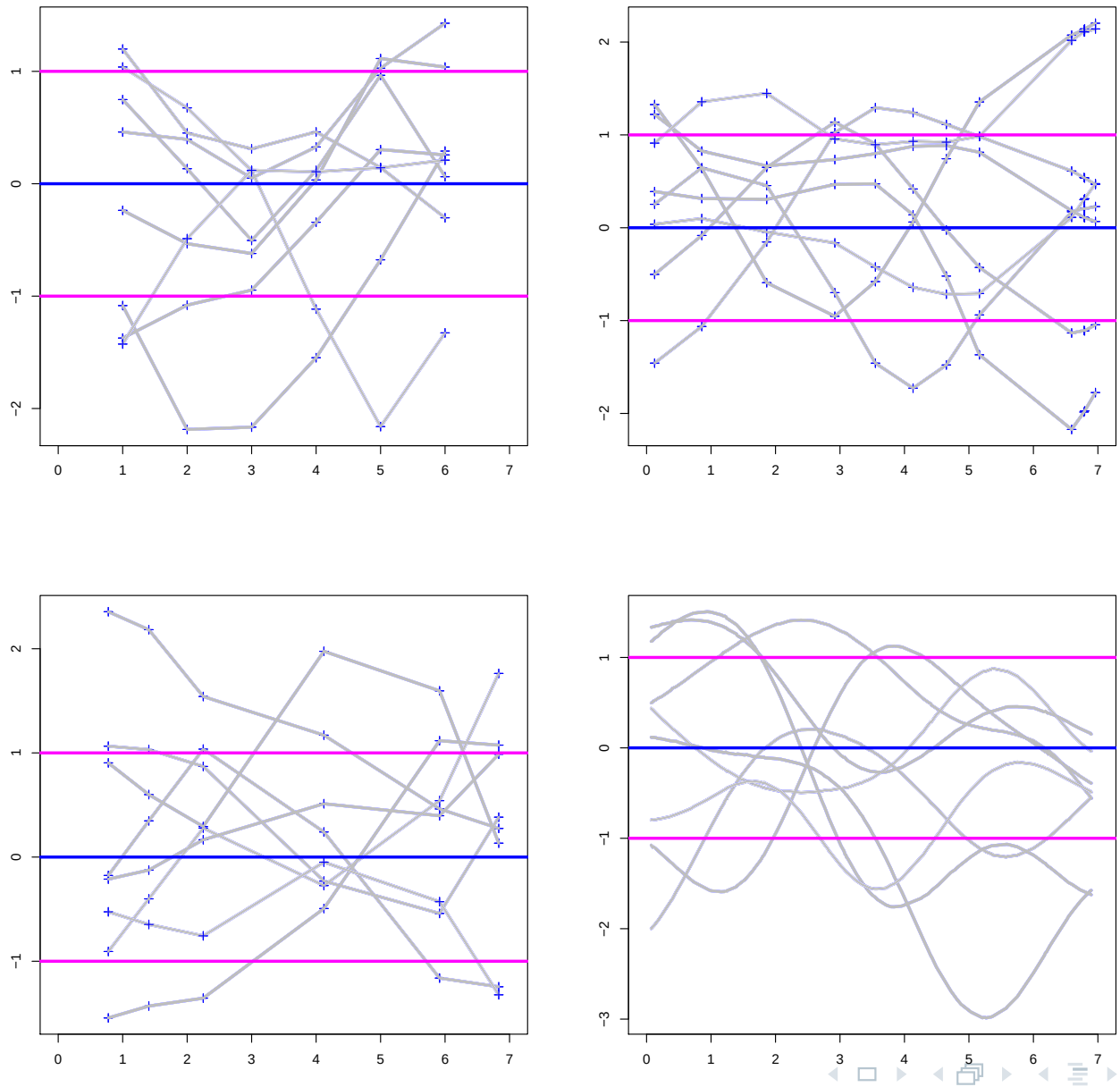
- Gaussian **Random Variable** (RV): $f \sim \mathcal{N}(\mu, \sigma^2)$.
- Gaussian **Random Vector**: Collection of n RVs, characterized by mean vector and covariance matrix: $\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$
- Gaussian **Process**: infinite Gaussian random vector, every finite subset of which is jointly Gaussian distributed
  Continuous index, e.g. time $t \rightsquigarrow$ **function** $f(t)$**.**
  Fully specified by **mean function** $m(t) = \mathbb{E}[f(t)]$
  and **covariance function** $k(t, t') = \mathbb{E}[(f(t) - m(t))(f(t') - m(t'))]$.
- In ML, we will focus on more general index sets $\boldsymbol{x} \in \mathbb{R}^d$ with **mean function** $m(\boldsymbol{x})$ and **covariance function** $k(\boldsymbol{x}, \boldsymbol{x}')$:

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

Visualizing Gaussian Processes: Sampling

- **Problem:** working with infinite vectors and covariance matrices is not very intuitive...
- **Solution:** evaluate the GP at set of n discrete times (or input vectors $\boldsymbol{x} \in \mathbb{R}^d$):
  - Choose n **input points** $\boldsymbol{x}_i$ at random $\rightsquigarrow$ matrix X
  - build **covariance matrix** $K(X, X)$ with **covariance function** $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$
  - **sample** realizations of the Gaussian random vector $\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{0}, K(X, X))$
  - plot $\boldsymbol{f}$ as **function of inputs**.

This is exactly what we have done here...



From the Prior to the Posterior

GP defines distribution over functions $\rightsquigarrow \boldsymbol{f}$ evaluated at training points X and $\boldsymbol{f}_*$ evaluated at test points X_* are jointly Gaussian:

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

Posterior $p(\boldsymbol{f}_*|X_*, X, \boldsymbol{f}(X))$: conditional of a Gaussian distribution.

Let $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, K)$. Let $\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{pmatrix}$ and $K = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix}$.

Then $\boldsymbol{x}_2|\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_2 + K_{21}K_{11}^{-1}(\boldsymbol{f}_1 - \boldsymbol{\mu}_1), K_{22} - K_{21}K_{11}^{-1}K_{12})$.

$$\boldsymbol{f}_*|X_*, X, \boldsymbol{f} \sim \mathcal{N}(\quad K(X_*,X)(K(X,X))^{-1}\boldsymbol{f}, \\ K(X_*,X_*) - K(X_*,X)(K(X,X))^{-1}K(X,X_*))$$

For only one test case:

$$f_*|\boldsymbol{x}_*, X, \boldsymbol{f} \sim \mathcal{N}(\boldsymbol{k}_*^t K^{-1}\boldsymbol{f}, k_{**} - \boldsymbol{k}_*^t K^{-1}\boldsymbol{k}_*)$$

A simple extension: noisy observations

- Assume we have access only to noisy versions of function values: $y = f(\boldsymbol{x}) + \eta,\ \eta \sim \mathcal{N}(0, \sigma^2)$ (cf. initial example of **ridge regression**).
- Noise η does not depend on data!
- Covariance of noisy observations y is sum of covariance of f and variance of noise: $cov(\boldsymbol{y}) = K(X, X) + \sigma^2 I$.

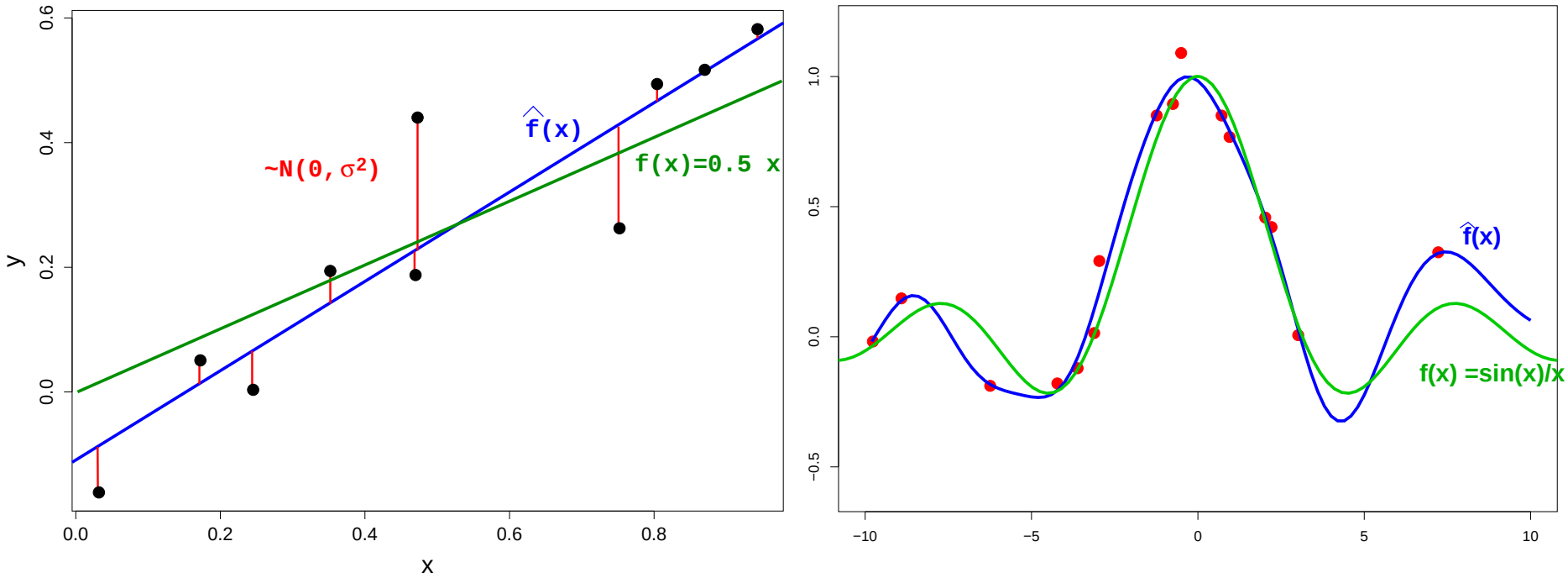
$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} K(X,X) + \sigma^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right)$$

$$\begin{aligned} \boldsymbol{f}_* | X_*, X, \boldsymbol{y} \sim \mathcal{N}(& K(X_*, X)(K(X,X) + \sigma^2 I)^{-1} \boldsymbol{y}, \\ & K(X_*, X_*) - K(X_*, X)(K(X,X) + \sigma^2 I)^{-1} K(X, X_*)) \end{aligned}$$

$$f_* | \boldsymbol{x}_*, X, \boldsymbol{y} \sim \mathcal{N}(\boldsymbol{k}_*^t (K + \sigma^2 I)^{-1} \boldsymbol{y}, k_{**} - \boldsymbol{k}_*^t (K + \sigma^2 I)^{-1} \boldsymbol{k}_*)$$

$\Rightarrow$ Posterior mean is solution of **kernel ridge regression**!

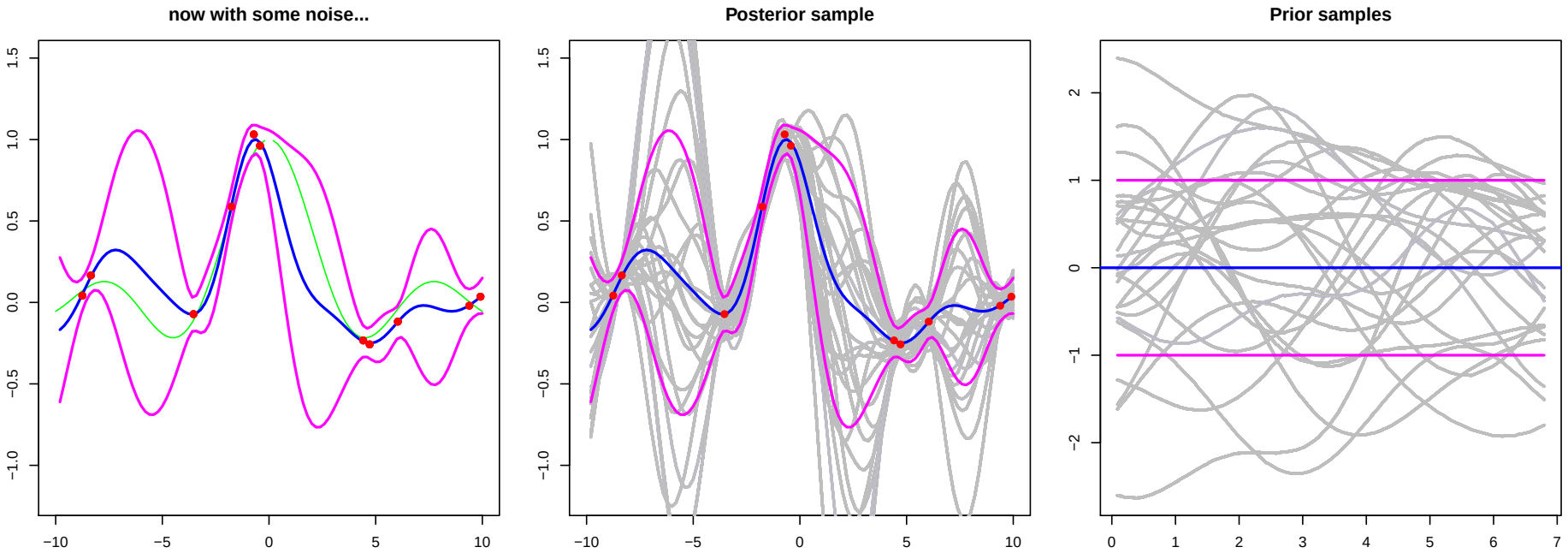
Noisy observations: examples



Noisy observations: $y = f(\boldsymbol{x}) + \eta,\ \eta \sim \mathcal{N}(0, \sigma^2)$

Mean predictions: $\hat{\boldsymbol{f}}_* = K_*(K + \sigma^2 I)^{-1}\boldsymbol{y}$.

Gaussian processes for regression



- **Left:** 11 training points generated as $y = \sin(x)/x + \nu$, $\nu \sim \mathcal{N}(0, 0.01)$
  Covariance $k(\boldsymbol{x}_p, \boldsymbol{x}_q) = \exp(-\frac{1}{2l^2}\|\boldsymbol{x}_p - \boldsymbol{x}_q\|^2) + \sigma^2\delta_{p,q}$.
  100 test points uniformly chosen from $[-10, 10] \rightsquigarrow$ matrix X_*.
  Mean prediction $E[\boldsymbol{f}_*|X_*, X, \boldsymbol{y}]$ and $\pm$std.dev.
- **Middle:** samples drawn from posterior $\boldsymbol{f}_*|X_*, X, \boldsymbol{y}$.
- **Right:** samples drawn from prior $\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{0}, K(X, X))$.

Covariance Functions

- A GP specifies a distribution over functions $f(\boldsymbol{x})$, characterized by mean function $m(\boldsymbol{x})$ and covariance function $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$.
- Finite subset evaluated at n inputs $\rightsquigarrow$ Gaussian distribution:

  $$\boldsymbol{f}(X) = (f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n))^t \sim \mathcal{N}(\boldsymbol{\mu}, K),$$

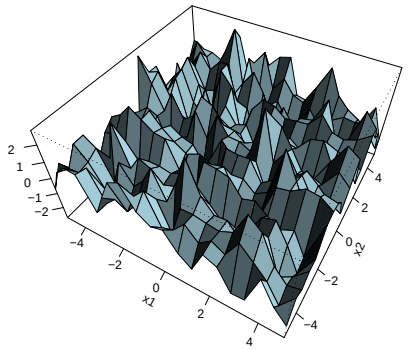
  where K is the covariance matrix with entries $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$.
- Covariance matrices are **symmetric positive semi-definite:**

  $$K_{ij} = K_{ji} \quad \text{and} \quad \boldsymbol{x}^t K \boldsymbol{x} \geq 0, \ \forall \boldsymbol{x}.$$

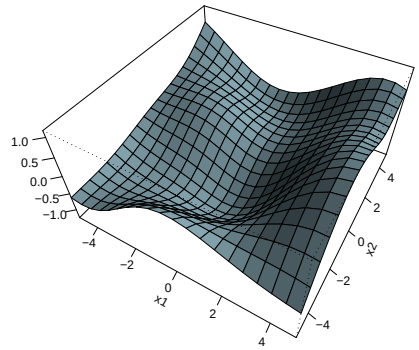
- We already know that **Mercer kernels** have this property
  $\rightsquigarrow$ all Mercer kernels define proper covariance functions in GPs.
- Kernels frequently have **additional parameters.**
- The **noise variance** in the observation model
  $y = f(\boldsymbol{x}) + \eta, \eta \sim \mathcal{N}(0, \sigma^2)$ is another parameter.
- How should we choose these parameters? $\rightsquigarrow$ **model selection.**

Model Selection

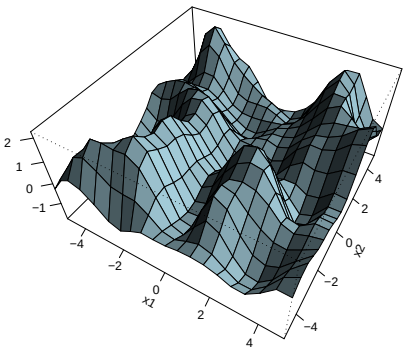
- **top left:** sample function from prior $\boldsymbol{f} \sim \mathcal{N}(\mathbf{0}, K(X, X))$ with **covariance function** $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\frac{1}{2l^2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2)$. Length scale $l = 10^{-0.5}$ small $\rightsquigarrow$ highly varying function.
- **bottom left:** same for $l = 10^0$ $\rightsquigarrow$ smoother function
- **top right:** same for $l = 10^{0.5}$ $\rightsquigarrow$ even smoother...
- **bottom right:** almost linear function for $l = 10^1$.



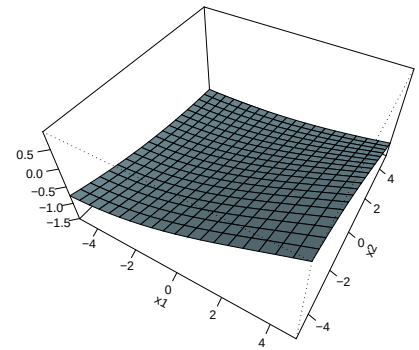
length scale: 10^−0.5, sample no. 1



length scale: 10^0.5, sample no. 1



length scale: 10^0, sample no. 1



length scale: 10^1, sample no. 1

Model Selection (2)

- How to select the parameters?
- One possibility: maximize **marginal likelihood:**

$$p(\boldsymbol{y}|X) = \int p(\boldsymbol{y}|\boldsymbol{f}, X)p(\boldsymbol{f}|X)\, d\boldsymbol{f}.$$

- We do not need to integrate: we know that

$$\boldsymbol{f}|X \sim \mathcal{N}(\boldsymbol{0}, K) \quad \text{and} \quad \boldsymbol{y} = \boldsymbol{f} + \eta,\ \eta \sim \mathcal{N}(0, \sigma^2).$$

  Since η does not depend on X, the variances simply add:

$$\boldsymbol{y}|X \sim \mathcal{N}(\boldsymbol{0}, K + \sigma^2 I).$$

- Possible strategy:
  Select parameters on a grid and choose maximum.
- Or: Compute derivatives of marginal likelihood and use gradient descent.

Model Selection (3)

- **Example problem:** $y = \sin(x)/x + \eta,\ \eta \sim \mathcal{N}(0, 0.01)$.
- **Log marg. likeli.** $= \log \mathcal{N}(\mathbf{0}, K + \sigma^2 I) =$

$$\underbrace{-\frac{1}{2}\boldsymbol{y}^t (K + \sigma^2 I)^{-1} \boldsymbol{y}}_{\text{data fit}} \underbrace{- \frac{1}{2} \log |K + \sigma^2 I|}_{\text{complexity penalty}} \underbrace{- \frac{n}{2} \log(2\pi)}_{\text{norm. constant}} .$$

- 2d-Example with Gaussian RBF:

$$(K + \sigma^2 I) = \begin{pmatrix} 1 + \sigma^2 & a \\ a & 1 + \sigma^2 \end{pmatrix} \Rightarrow |K + \sigma^2 I| = (1 + \sigma^2)^2 - a^2 > 0$$

Note that $a \to 0$ if length scale $l \to 0$

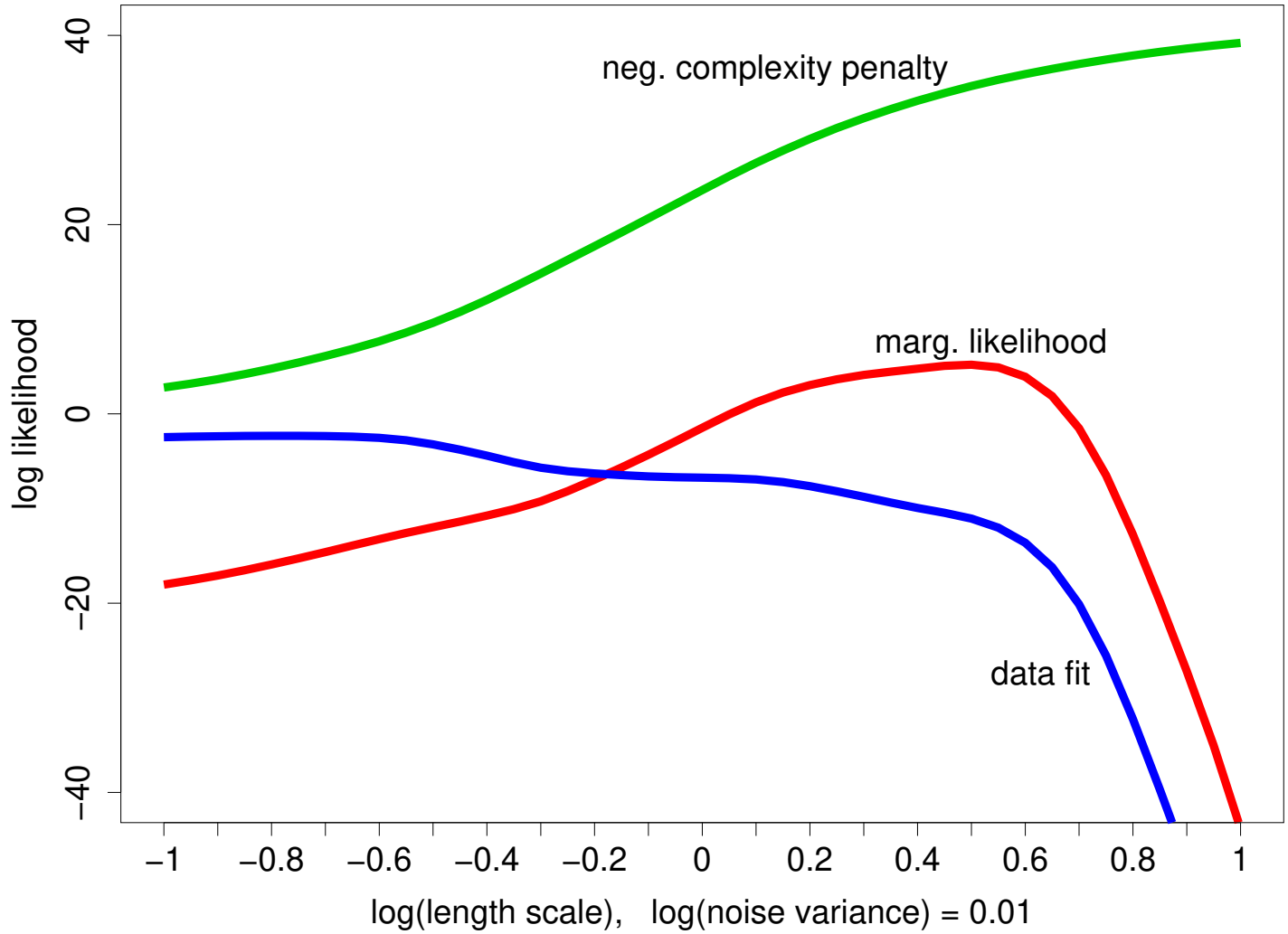
$\rightsquigarrow$ complexity penalty has high values for small length scales.

Matrix inverse includes a dominating factor $|K + \sigma^2 I|^{-1}$

$\rightsquigarrow$ data fit term also high for small l.

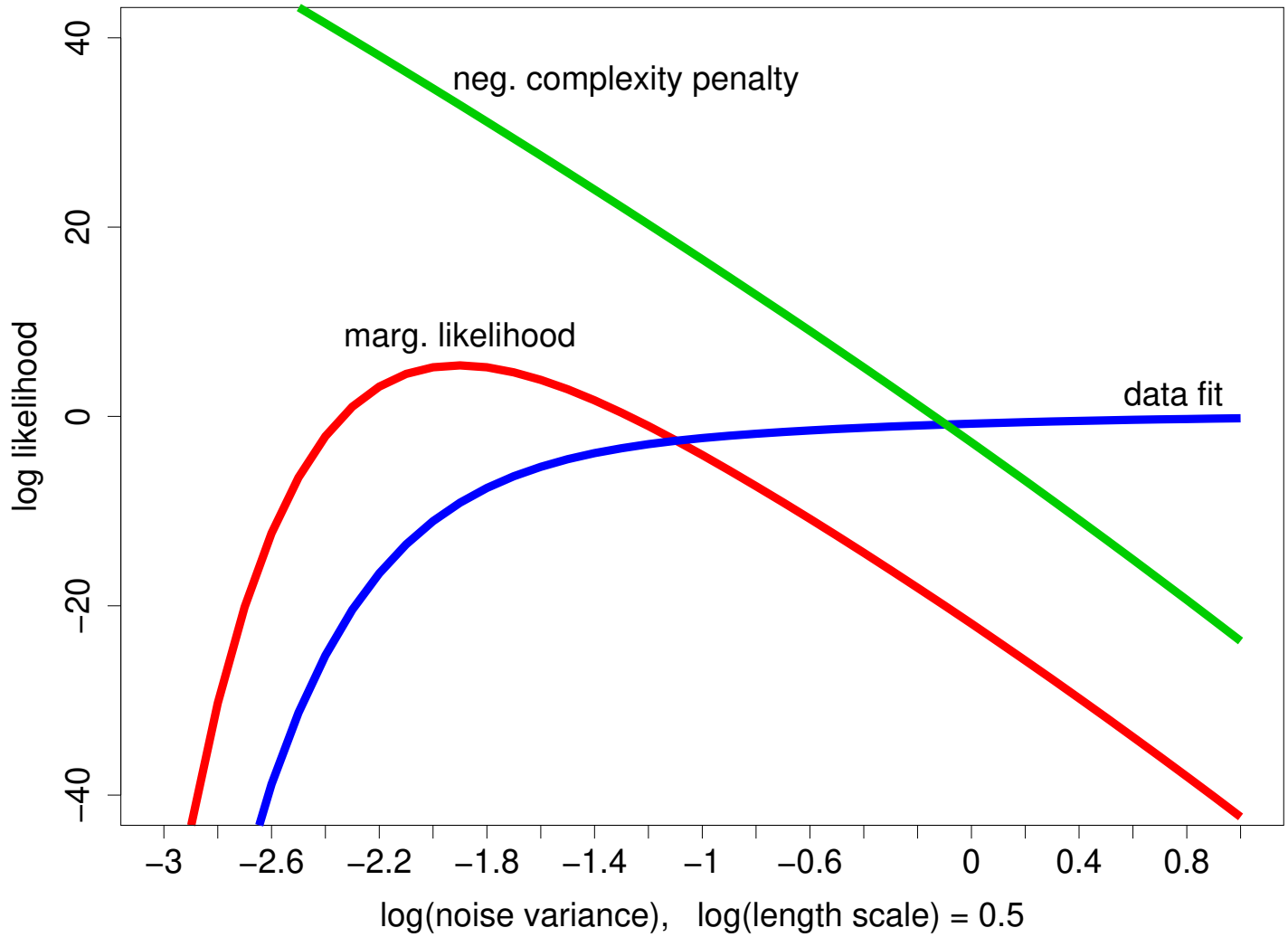
Model Selection (4)

Fixing $\sigma^2 = 0.01$ and **varying length scale** l:



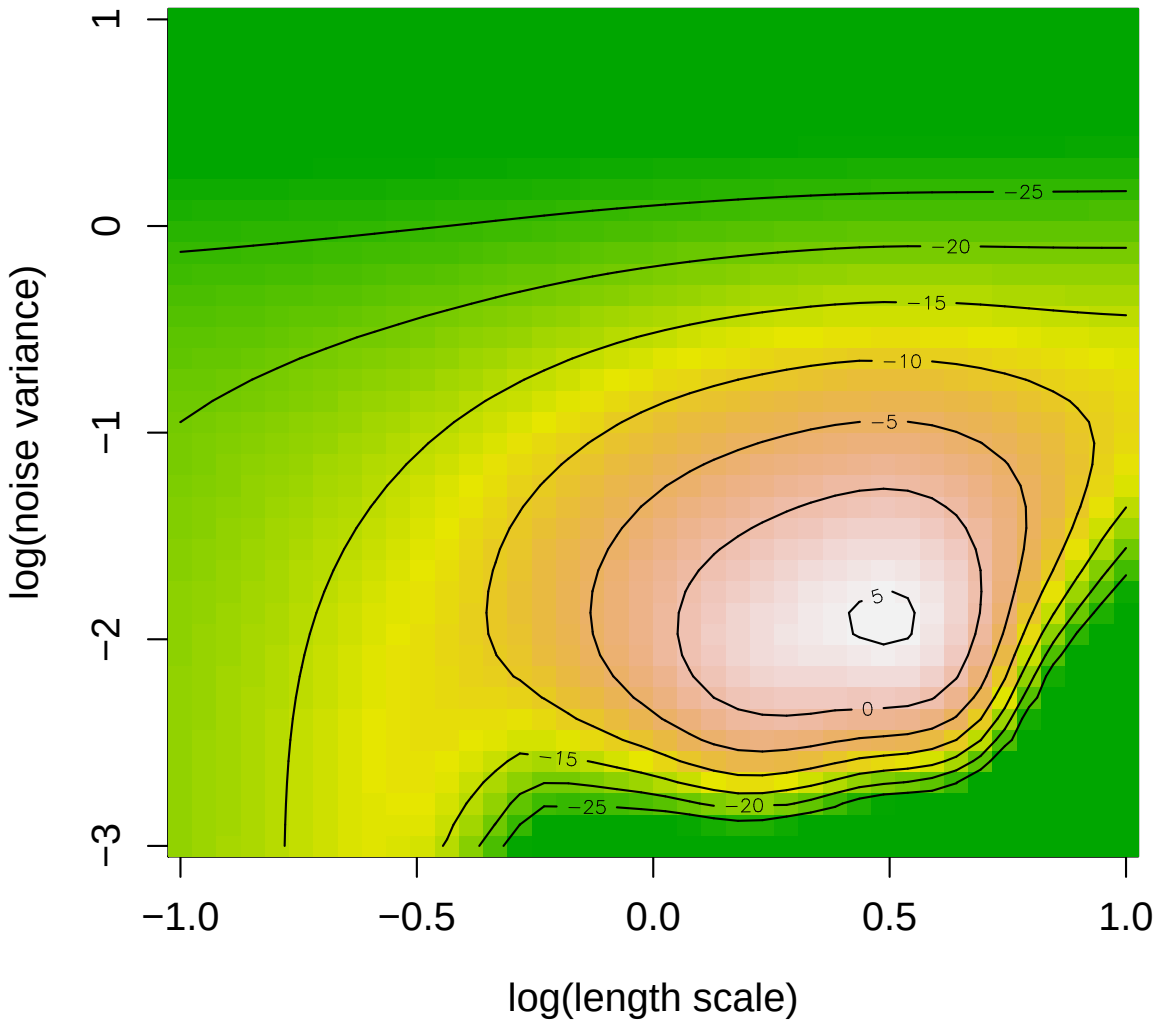
Model Selection (5)

Fixing length scale $l = 0.5$ and **varying the noise level** σ^2:

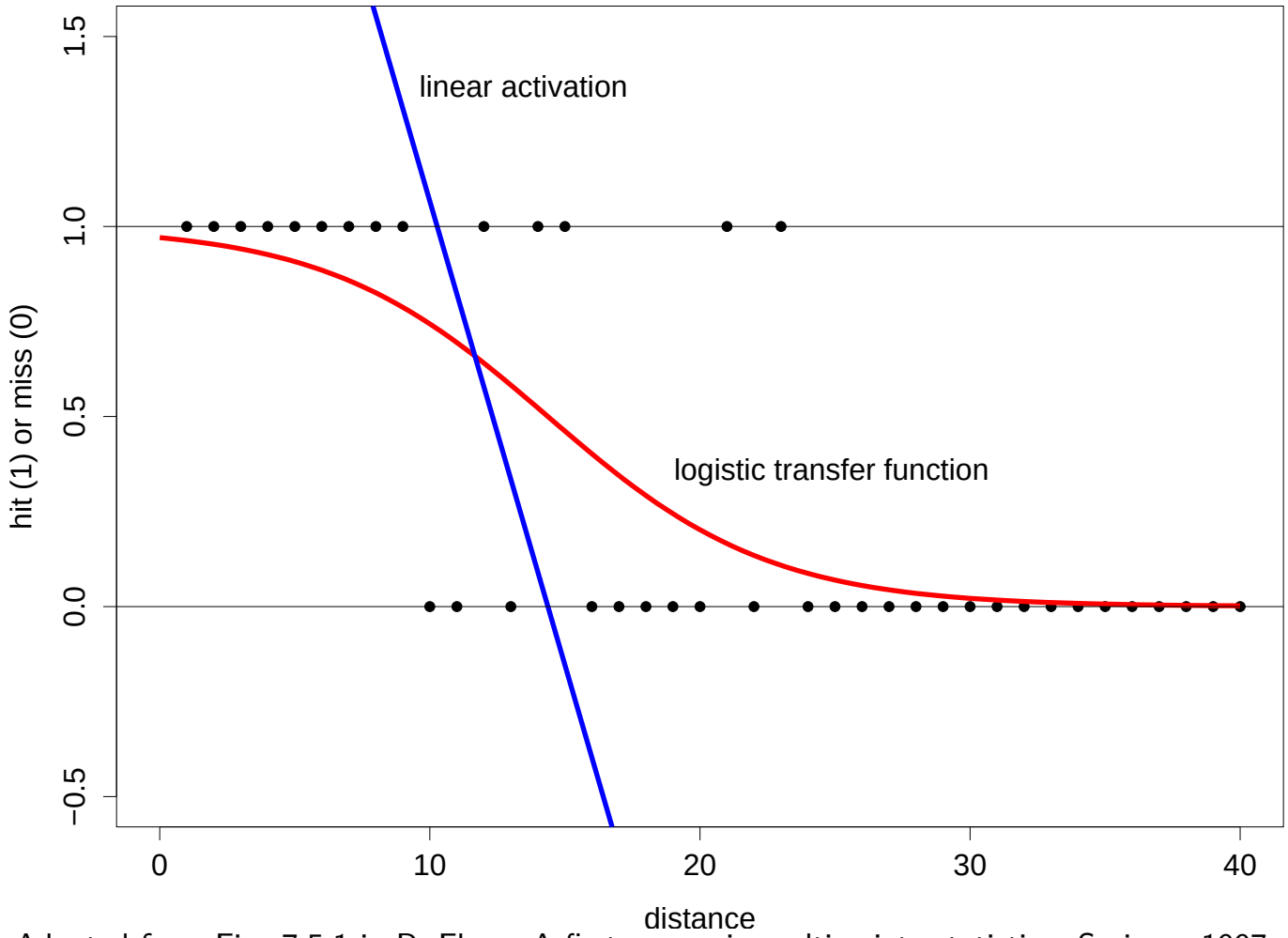


Model Selection (6)

Varying both σ^2 and l:



Classification: Basket Ball Example



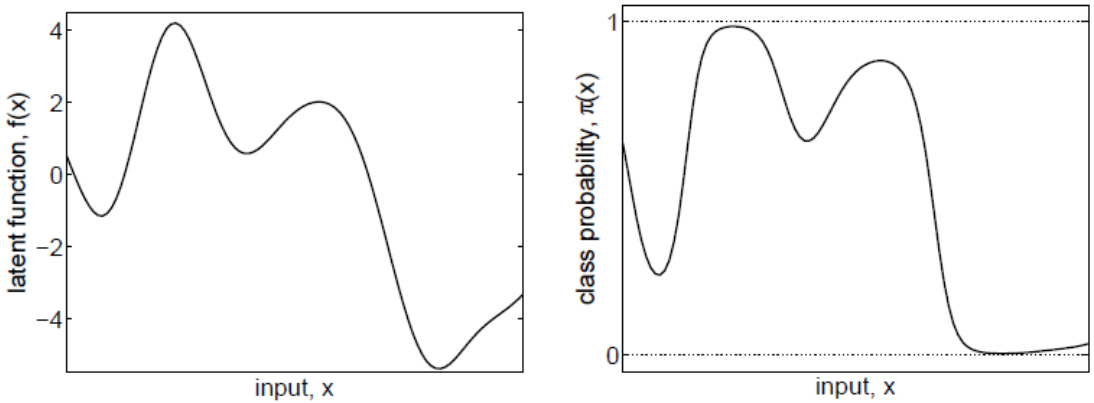
Adapted from Fig. 7.5.1 in B. Flury: A first course in multivariate statistics. Springer 1997.

Classical Logistic Regression

- Targets $y \in \{0, 1\}$
  $\rightsquigarrow$ Bernoulli RV with "success probability" $\pi(\boldsymbol{x}) = P(1|\boldsymbol{x})$.
- Likelihood: $P(\boldsymbol{y}|X, f) = \prod_{i=1}^{n} \left(\pi_f(\boldsymbol{x}_i)\right)^{y_i} \left(1 - \pi_f(\boldsymbol{x}_i)\right)^{1-y_i}$
- **Linear logistic regression:** unbounded $f(\boldsymbol{x}) = \boldsymbol{w}^t \boldsymbol{x}$ ("**activation**")
  Bounded estimates: pass $f(\boldsymbol{x})$ through **logistic transfer function** $\sigma(f(\boldsymbol{x})) = \frac{e^{f(\boldsymbol{x})}}{1+e^{f(\boldsymbol{x})}} = \frac{1}{1+e^{-f(\boldsymbol{x})}}$ and set $\pi_f(\boldsymbol{x}) = \sigma(f(\boldsymbol{x}))$.
- Use gradient-based methods for finding $\hat{\boldsymbol{w}}$ that maximizes the log posterior.
- Kernel trick: expand $\boldsymbol{w} = X^t \boldsymbol{\alpha}$, substitute dot products by kernel function $k(\boldsymbol{x}, \boldsymbol{x}') \rightsquigarrow$ **kernel logistic regression**.

GP Classification

- Place GP prior over "latent" function $\boldsymbol{f}(\boldsymbol{x}) \sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'))$.
- "Squash" it through logistic function $\rightsquigarrow$ prior on $\pi(\boldsymbol{x}) = \sigma(f(\boldsymbol{x}))$.



(Rasmussen & Williams, 2006)

- **Problem:** Bernoulli likelihood $\rightsquigarrow$ predictive distribution $p(y_* = 1|X, \boldsymbol{y}, \boldsymbol{x}_*)$ cannot be calculated analytically.
- Possible **solution:** use **Laplace approximation**.

GP Classification using Laplace's approximation

- Prior $\boldsymbol{f}|X \sim \mathcal{N}(\boldsymbol{0}, K)$. Bernoulli likelihood:

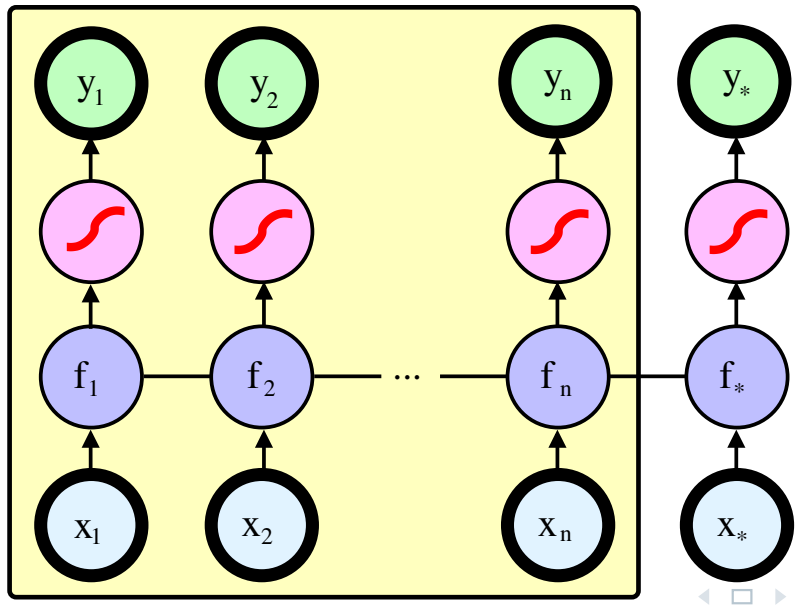
$$p(\boldsymbol{y}|X, \boldsymbol{f}) = \prod_{i=1}^{n} \left(\sigma(f(\boldsymbol{x}_i))\right)^{y_i} \left(1 - \sigma(f(\boldsymbol{x}_i))\right)^{1-y_i}.$$

- Gaussian approximation of posterior:

$$p(\boldsymbol{f}|X, \boldsymbol{y}) \approx \mathcal{N}(\hat{\boldsymbol{f}}, H^{-1}).$$

- Predictions: compute

$$p(y_* = 1|\boldsymbol{y}, \boldsymbol{x}_*, X) = \int \sigma(f_*) p\underbrace{(f_* \mid \boldsymbol{y},\ \boldsymbol{x}_*,\ X)}_{\text{latent function at } \boldsymbol{x}_*} df_* = \mathbb{E}_{p(f_*|\boldsymbol{y},\boldsymbol{x}_*,X)}(\sigma)$$



GP Classification using Laplace's approximation

- First **predict latent function** at test case $\boldsymbol{x}_*$:

$$p(f_*|\boldsymbol{y}, \boldsymbol{x}_*, X) = \int \underbrace{p(f_*|\boldsymbol{f}, \boldsymbol{x}_*, X)}_{\text{Gaussian}} \quad \underbrace{p(\boldsymbol{f}|X, \boldsymbol{y})d\boldsymbol{f}}_{\text{approx. Gaussian } \mathcal{N}(\hat{\boldsymbol{f}}, H^{-1})}$$

$$\approx \mathcal{N}(\mu_*, \sigma_*), \text{ with}$$

$$\mu_* = \boldsymbol{k}_*^t K^{-1} \hat{\boldsymbol{f}},$$

$$\sigma_* = k_{**} - \boldsymbol{k}_*^t \tilde{K}^{-1} \boldsymbol{k}_*$$

- Then use **Monte Carlo approximation**

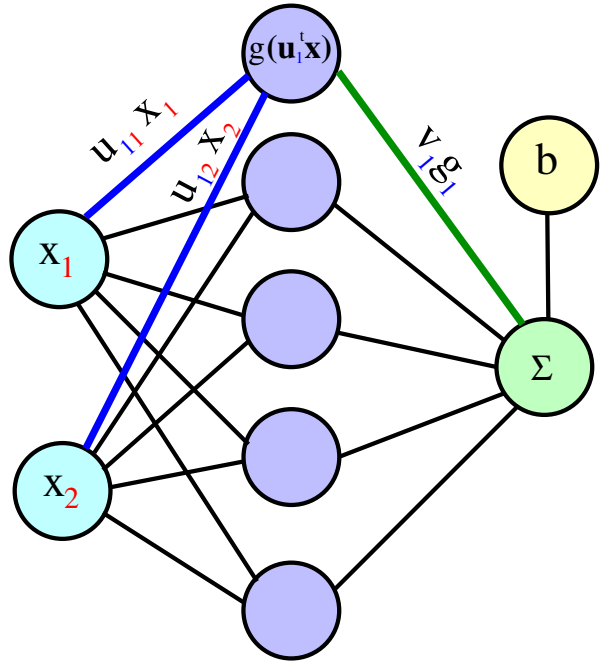
$$p(y_*|\boldsymbol{y}, \boldsymbol{x}_*, X) = \mathbb{E}_{p(f_*|\boldsymbol{y}, \boldsymbol{x}_*, X)}(\sigma) \approx \frac{1}{S} \sum_{s=1}^{S} \sigma(f_*^s(\boldsymbol{x}_*)),$$

where f_*^s are samples from the (approximated) distribution over latent function values.

GPs and Neural networks

Consider a neural network with one hidden layer of n_H units:

$$f(\boldsymbol{x}) = b + \sum_{j=1}^{n_H} v_j g(\boldsymbol{x}; \boldsymbol{u}_j).$$



- **Bayesian treatment:** **All weights are considered as RVs** for which we define **prior distributions:** zero-mean Gaussian priors for b and v, and for components of the weight vector $\boldsymbol{u}_j$ at the j-th hidden unit.
- What is the asymptotic distribution of $f(\boldsymbol{x})$, as $n_H \to \infty$?

GPs and Neural networks

NN-GP kernel (Neal 1996, Williams 1998)

A MLP with one hidden layer of infinite width, which has Gaussian priors on all parameters, converges to a Gaussian process with a well-defined covariance function (extensions to deep MLPs, CNNs etc. exist).

- Network with n_H hidden units implements function $f(\boldsymbol{x}) = b + \sum_{j=1}^{n_H} v_j g(\boldsymbol{x}; \boldsymbol{u}_j), \quad g(\boldsymbol{x}; \boldsymbol{u}_j) = \varphi(u_{j0} + \boldsymbol{x}^t \boldsymbol{u}_j)$.
- Priors: $b \sim N(0, \sigma_b^2), v_j \sim N(0, \sigma_v^2), u_{ji} \sim N(0, \sigma_{u_{ji}}^2)$
- Let $\boldsymbol{\theta} = \{b, v_j, \boldsymbol{u}_j\}$ be all the parameters.

Mean of network output:

$$\mathbb{E}_{\boldsymbol{\theta}}[f(\boldsymbol{x})] = \overbrace{\mathbb{E}_{\boldsymbol{\theta}}[b]}^{=0} + \sum_{j=1}^{n_H} \mathbb{E}_{\boldsymbol{\theta}}[v_j g(\boldsymbol{x}; \boldsymbol{u}_j)]$$

$$\overset{(v \text{ indep. of } u)}{=} \sum_{j=1}^{n_H} \underbrace{\mathbb{E}_{\boldsymbol{\theta}}[v_j]}_{=0} \mathbb{E}_{\boldsymbol{u}}[g(\boldsymbol{x}; \boldsymbol{u}_j)] = 0.$$

GPs and Neural networks: Covariance

Covariance when the function is applied to two inputs:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\theta}}[f(\boldsymbol{x})f(\boldsymbol{x}')] =& \mathbb{E}_{\boldsymbol{\theta}}[(b + \sum_{j=1}^{n_H} v_j g(\boldsymbol{x}))(b + \sum_{j=1}^{n_H} v_j g(\boldsymbol{x}'))] \\ =& \sigma_b^2 + \sum_{j=1}^{n_H} \mathbb{E}_{\boldsymbol{\theta}}[v_j^2]\ \mathbb{E}_{\boldsymbol{u}}[g(\boldsymbol{x};\boldsymbol{u}_j)g(\boldsymbol{x}';\boldsymbol{u}_j)] \\ =& \quad \sigma_b^2 + \sigma_v^2 n_H\, \mathbb{E}_{\boldsymbol{u}}[g(\boldsymbol{x};\boldsymbol{u})g(\boldsymbol{x}';\boldsymbol{u})],\end{aligned}$$

because all of the hidden units are identically distributed.

- Let $n_H \to \infty$. Scale magnitude of output by defining $\sigma_v^2 = \frac{\omega}{n_H}$.
- Input to the output neuron is an **infinite sum over i.i.d. RVs.** $\rightsquigarrow$ **central limit theorem** $\rightsquigarrow$ for a single input $\boldsymbol{x}$: Gaussian distribution $f(\boldsymbol{x}) \sim N(0, \sigma_b^2 + \omega \mathbb{E}_{\boldsymbol{u}}[g(\boldsymbol{x})^2])$
- Collection of n inputs $\boldsymbol{x}_i \rightsquigarrow$ joint distribution: multivariate zero-mean Gaussian with covariance $\sigma_b^2 + \omega \mathbb{E}_{\boldsymbol{u}}[g(\boldsymbol{x})g(\boldsymbol{x}')] := k_{\text{NN-GP}}(\boldsymbol{x}, \boldsymbol{x}')$.
- For some activations φ, NN-GP kernel can be computed analytically.

Summary

- **GPs: fully probabilistic models**
  $\rightsquigarrow$ posterior $p(\boldsymbol{f}_*|X, \boldsymbol{y}, \boldsymbol{x}_*)$.
- Uniquely defined by specifying **covariance function**.
- **Mathematically simple:**
  we only need to calculate **conditionals of Gaussians**!
- **Connections:**
  regression: $\text{MAP}(\text{GP}_r) = \text{kernel ridge reg.}$
  $\text{GP}_c \approx$ probabilistic version of SVM.

  Networks of infinite width can be interpreted as
  Gaussian Processes with the NN-GP kernel.
  It is also possible to derive kernels from networks **after training:**
  Neural Tangent Kernel.