

# PROBABILISTIC INFERENCE AND LEARNING

## LECTURE 06

### GAUSSIAN PROCESSES

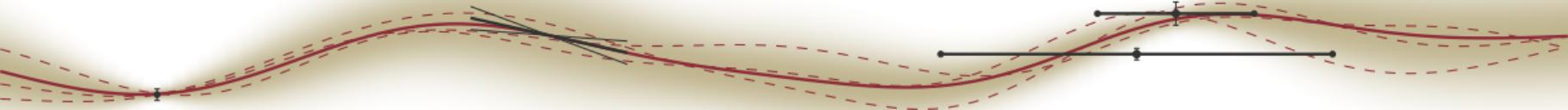
Philipp Hennig

05 November 2018

EBERHARD KARLS  
**UNIVERSITÄT**  
TÜBINGEN

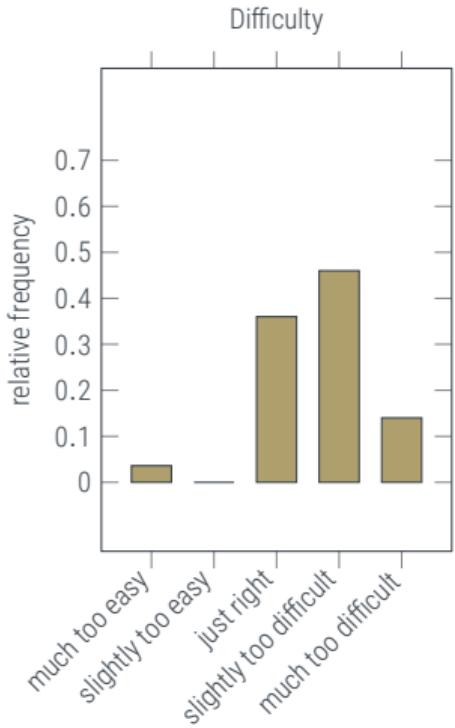
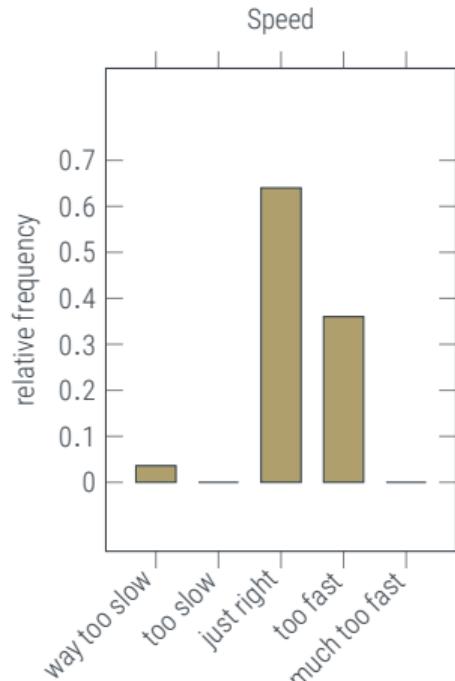
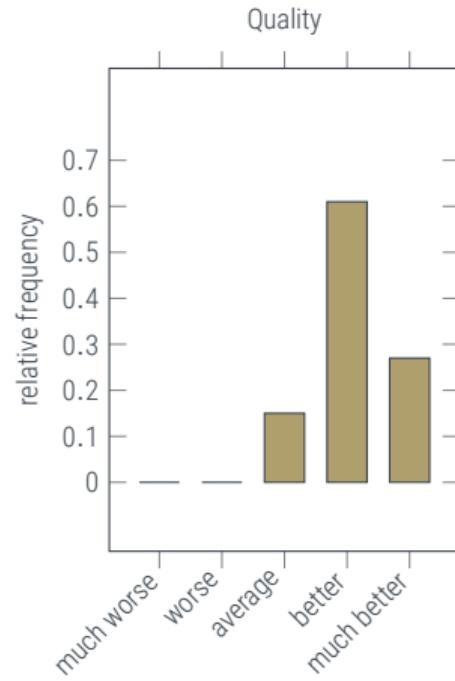


FACULTY OF SCIENCE  
DEPARTMENT OF COMPUTER SCIENCE  
CHAIR FOR THE METHODS OF MACHINE LEARNING



# Last Lecture: Debrief

Feedback dashboard





# Last Lecture: Debrief

## Detailed Feedback

### Things you did not like:

- ♦ notation for derivatives
- ♦ the colors

### Things you did not understand:

- ♦ legends on graphs missing
- ♦ "what do you mean about fitting the last layer of a deep net in a Bayesian way?"  
(Type-II ML)

### Things you enjoyed:

- ♦ connection to deep learning
- ♦ Occam factor animation
- ♦ Overview at Beginning
- ♦ comments on Tensorflow
- ♦ clarification for what terms  $K, G, \dots$  mean
- ♦ AutoDiff
- ♦ explanations on blackboard



## Overview of Lectures so far:

0. Introduction to Reasoning under Uncertainty
  - Probabilities are the mathematical formalization of uncertainty
1. Probabilistic Reasoning
  - Probabilities extend deductive to plausible reasoning. Conditional independence affects complexity
2. Probabilities over Continuous Variables
  - Probability **densities** distribute probability over continuous domains
3. Gaussian Probability Distributions
  - Gaussians map probabilistic inference to **linear algebra**
4. Gaussian Parametric Regression
  - Gaussian's can be used to infer **functions**, using linear models
5. More on Gaussian Parametric Regression
  - A parametric Gaussian regressor is a single-layer feedforward network

Today:

- It's possible to use **infinitely many features** in finite time!
- Our first Bayesian **nonparametric model**: **Gaussian processes** which are related to **kernel methods**



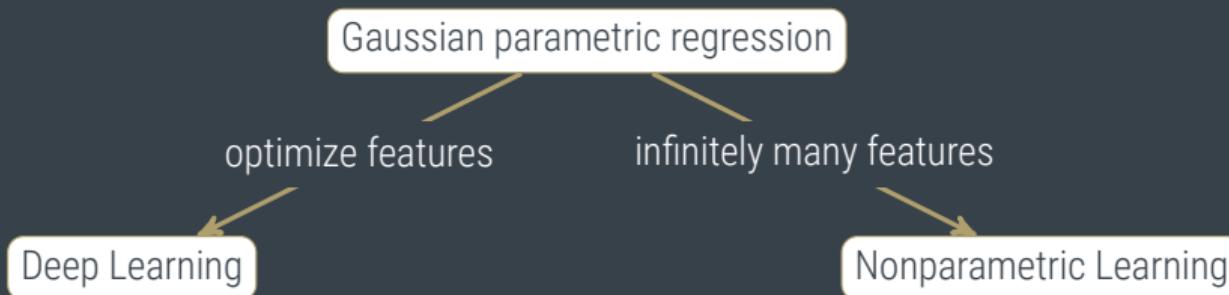
## Summary:

- Gaussian distributions can be used to **learn functions**
- Analytical inference is possible using **general linear models**

$$f(x) = \phi(x)^T w = \phi_x^T w$$

- The choice of features  $\phi : \mathbb{X} \rightarrow \mathbb{R}$  is essentially unconstrained
- with parametrized feature families, **hierarchical inference** allows **fitting** features
- Gaussian linear regressors are **single-layer neural networks** with quadratic loss

aka. least-square regression (cf. next week)



# Reminder: Where we left off last time

## Probabilistic Parametric Regression

$$f(x) := \phi_x^\top w \quad p(w) = \mathcal{N}(w; \mu, \Sigma) \quad p(y | f) = \mathcal{N}(y; f_x, \sigma^2 I)$$

$$p(w | y, \phi_x) = \mathcal{N}\left(w; \mu + \Sigma \phi_x (\phi_x^\top \Sigma \phi_x + \sigma^2 I)^{-1} (y - \phi_x^\top \mu), \Sigma - \Sigma \phi_x (\phi_x^\top \Sigma \phi_x + \sigma^2 I)^{-1} \phi_x^\top \Sigma\right)$$

$$p(f_x | y, \phi_x) = \mathcal{N}\left(f_x; \phi_x^\top \mu + \phi_x^\top \Sigma \phi_x (\phi_x^\top \Sigma \phi_x + \sigma^2 I)^{-1} (y - \phi_x^\top \mu), \phi_x^\top \left(\Sigma - \Sigma \phi_x (\phi_x^\top \Sigma \phi_x + \sigma^2 I)^{-1} \phi_x^\top \Sigma\right) \phi_x\right)$$



# Reminder: Where we left off last time

## Probabilistic Parametric Regression

$$\begin{aligned}\phi(x) &= \left[ e^{-\frac{1}{2}(x-8)^2} \quad e^{-\frac{1}{2}(x-7)^2} \quad e^{-\frac{1}{2}(x-6)^2} \quad \dots \right]^\top \\ &= \left[ e^{-(x-c_i)^2/(2\lambda^2)} \right]_{i=1,\dots,F}\end{aligned}$$

# What are we actually doing with those features?

let's look at that algebra again

$$\begin{aligned}
 p(f_x \mid y, \phi_x) &= \mathcal{N}(f_x; \phi_x^\top \mu + \phi_x^\top \Sigma \phi_x (\phi_x^\top \Sigma \phi_x + \sigma^2 I)^{-1} (y - \phi_x^\top \mu), \\
 &\quad \phi_x^\top \Sigma \phi_x - \phi_x^\top \Sigma \phi_x (\phi_x^\top \Sigma \phi_x + \sigma^2 I)^{-1} \phi_x^\top \Sigma \phi_x) \\
 &= \mathcal{N}(f_x; m_x + k_{xx} (k_{xx} + \sigma^2 I)^{-1} (y - m_x), \\
 &\quad k_{xx} - k_{xx} (k_{xx} + \sigma^2 I)^{-1} k_{xx})
 \end{aligned}$$

using the abstraction / encapsulation

$$m_x := \phi_x^\top \mu$$

$$m : \mathbb{X} \rightarrow \mathbb{R}$$

mean function

$$k_{ab} := \phi_a^\top \Sigma \phi_b$$

$$k : \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{R}$$

covariance function, aka. kernel



# Code

`Gaussian_Process_Regression.ipynb`

# Sometimes, more features make things **cheaper**

an example

- For simplicity, let's fix  $\Sigma = \frac{\sigma^2(c_{\max} - c_{\min})}{F} I$

$$\text{thus: } \phi(x_i)^\top \Sigma \phi(x_j) = \frac{\sigma^2(c_{\max} - c_{\min})}{F} \sum_{\ell=1}^F \phi_\ell(x_i) \phi_\ell(x_j)$$

- especially, for  $\phi_\ell(x) = \exp\left(-\frac{(x - c_\ell)^2}{2\lambda^2}\right)$

$$\begin{aligned} \phi(x_i)^\top \Sigma \phi(x_j) &= \frac{\sigma^2(c_{\max} - c_{\min})}{F} \sum_{\ell=1}^F \exp\left(-\frac{(x_i - c_\ell)^2}{2\lambda^2}\right) \exp\left(-\frac{(x_j - c_\ell)^2}{2\lambda^2}\right) \\ &= \frac{\sigma^2(c_{\max} - c_{\min})}{F} \exp\left(-\frac{(x_i - x_j)^2}{4\lambda^2}\right) \sum_{\ell=1}^F \exp\left(-\frac{(c_\ell - \frac{1}{2}(x_i + x_j))^2}{\lambda^2}\right) \end{aligned}$$

# Sometimes, more features make things cheaper

an example

$$\phi(x_i)^\top \Sigma \phi(x_j) = \frac{\sigma^2(c_{\max} - c_{\min})}{F} \exp\left(-\frac{(x_i - x_j)^2}{4\lambda^2}\right) \sum_{\ell}^F \exp\left(-\frac{(c_{\ell} - \frac{1}{2}(x_i + x_j))^2}{\lambda^2}\right)$$

now increase  $F$  so # of features in  $\delta c$  approaches  $\frac{F \cdot \delta c}{(c_{\max} - c_{\min})}$

$$\phi(x_i)^\top \Sigma \phi(x_j) \rightarrow \sigma^2 \exp\left(-\frac{(x_i - x_j)^2}{4\lambda^2}\right) \int_{c_{\min}}^{c_{\max}} \exp\left(-\frac{(c - \frac{1}{2}(x_i + x_j))^2}{\lambda^2}\right) dc$$

let  $c_{\min} \rightarrow -\infty, c_{\max} \rightarrow \infty$

$$k(x_i, x_j) := \phi(x_i)^\top \Sigma \phi(x_j) \rightarrow \sqrt{2\pi} \lambda \sigma^2 \exp\left(-\frac{(x_i - x_j)^2}{4\lambda^2}\right)$$

$$k_{x_i x_j} = \sqrt{2\pi} \lambda \sigma^2 \exp\left(-\frac{(x_i - x_j)^2}{4\lambda^2}\right)$$



# In pictures

convergence to Gaussian kernel



# In pictures

convergence to Gaussian kernel



# In pictures

convergence to Gaussian kernel

- ✚ aka. radial basis function, square( $d$ )-exponential kernel



# In pictures

convergence to Gaussian kernel

- ✚ aka. radial basis function, square( $d$ )-exponential kernel



# Code

`Gaussian_Process_Regression.ipynb`



# What just happened?

kernelization and Gaussian processes

## Definition (kernel)

$k : \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{R}$  is a (Mercer / positive definite) **kernel** if, for any finite collection  $X = [x_1, \dots, x_N]$ , the matrix  $k_{XX} \in \mathbb{R}^{N \times N}$  with  $[k_{XX}]_{ij} = k(x_i, x_j)$  is **positive semidefinite**.

```
def kernel (f) : λ (a,b) -> [[f(a[i],b[j]) for j=1:length(b)] for i=1:length(a) ]
```

actually, in python: `def kernel (f) : return lambda a,b : array([ [float64(f(a[i],b[j])) for j in range(b.size) ] for i in range(a.size) ])`

## Definition (positive definite matrix)

A symmetric matrix  $A \in \mathbb{R}^{N \times N}$  is called **positive (semi-) definite** if

$$v^T A v \geq 0 \quad \forall v \in \mathbb{R}^N.$$

Equivalently:

- All eigenvalues of the symmetric matrix  $A$  are non-negative
- $A$  is a Gram matrix – the outer product of  $N$  vectors  $[\phi_i]_{i=1,\dots,N}$

# Kernels

It is fine to think of them as inner products

Lemma:

$k$  is a Mercer kernel if it can be written as (assuming  $\mathcal{L}$  is positive set for  $\nu$ )

$$k(x, x') = \sum_{\ell \in \mathcal{L}} \phi_\ell(x) \phi_\ell(x') \quad \text{or} \quad = \int_{\mathcal{L}} \phi_\ell(x) \phi_\ell(x') d\nu(\ell)$$

Proof:  $\forall X \in \mathbb{X}^N, v \in \mathbb{R}^N : v^\top k_{XX} v = \sum_i \sum_i^N v_i \phi_\ell(x_i) \sum_j^N v_j \phi_\ell(x_j) d\nu(\ell) = \sum_i \left[ \sum_i v_i \phi_\ell(x_i) \right]^2 d\nu(\ell) \geq 0 \square$



# Gaussian processes

are programs

## Definition

Let  $\mu : \mathbb{X} \rightarrow \mathbb{R}$  be any function,  $k : \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{R}$  be a Mercer kernel. A **Gaussian process**  $p(f) = \mathcal{GP}(f; \mu, k)$  is a probability distribution over the function  $f : \mathbb{X} \rightarrow \mathbb{R}$ , such that every finite restriction to function values  $f_X := [f_{x_1}, \dots, f_{x_N}]$  is a Gaussian distribution  $p(f_X) = \mathcal{N}(f_X; \mu_X, k_{XX})$ .



# Another kernel

The Wiener process

$$\phi_i(x) = \theta(x - c_i) = \begin{cases} 1 & \text{if } x \geq c_i \\ 0 & \text{else} \end{cases}$$

# Another Kernel

The Wiener process

$$\phi_i(x) = \theta(x - c_i) = \begin{cases} 1 & \text{if } x \geq c_i \\ 0 & \text{else} \end{cases} \quad \Sigma = \sigma^2 \frac{c_{\max} - c_0}{F}$$

$$\Rightarrow k(x_i, x_j) = \phi(x_i)^T \Sigma \phi(x_j) = \frac{\sigma^2(c_{\max} - c_0)}{F} \sum_{\ell=1}^F \theta(\min(x_i, x_j) - c_\ell)$$

now increase  $F$  so # of features in  $\delta c$  approaches  $\frac{F \cdot \delta c}{(c_{\max} - c_0)}$

$$k(x_i, x_j) \rightarrow \sigma^2 \int_{c_0}^{c_{\max}} \theta(\min(x_i, x_j) - c) dc = \sigma^2 [1]_{c_0}^{\min(x_i, x_j)}$$

$$= \sigma^2(\min(x_i, x_j) - c_0)$$



# Graphical View

Convergence to Wiener process

$$k(x_i, x_k) = \sigma^2(\min(x_i, x_j) - c_0) \quad \text{var } f(x) = \sigma^2(x - c_0)$$



# Graphical View

Convergence to Wiener process

$$k(x_i, x_k) = \sigma^2(\min(x_i, x_j) - c_0) \quad \text{var } f(x) = \sigma^2(x - c_0)$$



# Graphical View

Convergence to Wiener process

$$k(x_i, x_k) = \sigma^2(\min(x_i, x_j) - c_0) \quad \text{var } f(x) = \sigma^2(x - c_0)$$



# Graphical View

Convergence to Wiener process

$$k(x_i, x_k) = \sigma^2(\min(x_i, x_j) - c_0) \quad \text{var } f(x) = \sigma^2(x - c_0)$$

# The Oldest Gaussian Process

Brownian Motion & Wiener process

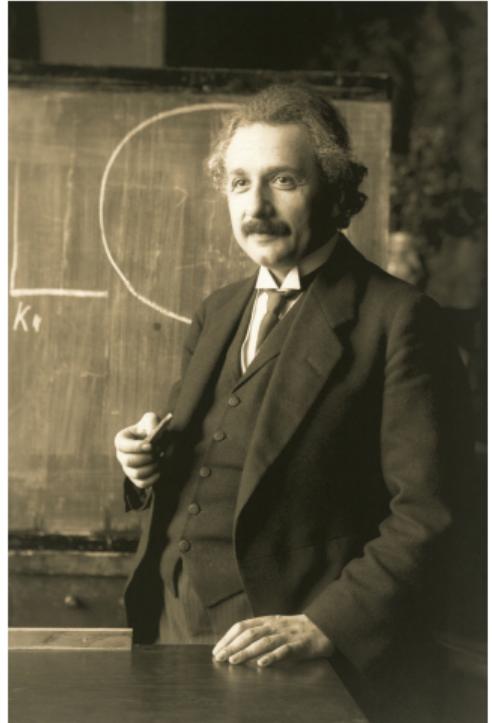


[image: F. Schmutzer, 1921, Austrian National Library]

sein. Das Problem, welches mit dem Problem der Diffusion von einem Punkte aus (unter Vernachlässigung der Wechselwirkung der diffundierenden Teilchen) übereinstimmt, ist nun mathematisch vollkommen bestimmt; seine Lösung ist:

$$f(x, t) = \frac{n}{\sqrt{4\pi D}} \frac{e^{-\frac{x^2}{4Dt}}}{\sqrt{t}}.$$

[A. Einstein, Über die von der molekularkinetischen Theorie der Wärme geforderte Bewegung von in ruhenden Flüssigkeiten suspendierten Teilchen, Ann. d. Physik, 1905]





# Just one more!

Splines

[G. Wahba, *Spline Models for Observational Data*, 1990]

remember Wiener:  $\phi_i(x) = \theta(x - c_i)$        $f(x) = \phi(x)^\top w$

consider:  $\tilde{\phi}_i(x) = \int_{-\infty}^x \theta(\tilde{x} - c_i) d\tilde{x}$     "RELU"

equivalent to:  $\tilde{f}(x) = \int_{-\infty}^x f(\tilde{x}) d\tilde{x}$

$$\implies \tilde{k}(x_i, x_j) = \tilde{\phi}(x_i)^\top \Sigma \tilde{\phi}(x_j) = \iint_{-\infty}^{x_i, x_j} k(\tilde{x}_i, \tilde{x}_j) d\tilde{x}_i d\tilde{x}_j$$

$$= \sigma^2 \left( \frac{1}{3} \min^3(x_i - x_0, x_j - x_0)^3 + \frac{1}{2} |x_i - x_j| \min^2(x_i - x_0, x_j - x_0) \right)$$



Nb: The integral of a Gaussian process is also a Gaussian process!

# Cubic Splines

from rectified linear units



$$k(x_i, x_k) = \sigma^2 \left( \frac{1}{3} \min^3(x_i - x_0, x_j - x_0)^3 + \frac{1}{2} |x_i - x_j| \min^2(x_i - x_0, x_j - x_0) \right)$$

# Cubic Splines

from rectified linear units



$$k(x_i, x_k) = \sigma^2 \left( \frac{1}{3} \min^3(x_i - x_0, x_j - x_0)^3 + \frac{1}{2} |x_i - x_j| \min^2(x_i - x_0, x_j - x_0) \right)$$

# Cubic Splines

from rectified linear units



$$k(x_i, x_k) = \sigma^2 \left( \frac{1}{3} \min^3(x_i - x_0, x_j - x_0)^3 + \frac{1}{2} |x_i - x_j| \min^2(x_i - x_0, x_j - x_0) \right)$$

# Cubic Splines

from rectified linear units



$$k(x_i, x_k) = \sigma^2 \left( \frac{1}{3} \min^3(x_i - x_0, x_j - x_0)^3 + \frac{1}{2} |x_i - x_j| \min^2(x_i - x_0, x_j - x_0) \right)$$

# Cubic Splines

from rectified linear units



$$k(x_i, x_k) = \sigma^2 \left( \frac{1}{3} \min^3(x_i - x_0, x_j - x_0)^3 + \frac{1}{2} |x_i - x_j| \min^2(x_i - x_0, x_j - x_0) \right)$$



- Sometimes it is possible to consider **infinitely many** features at once, by extending from a sum to an integral. This requires some regularity assumption about the features' locations, shape, etc.
- The resulting **nonparametric model** is known as a **Gaussian process**
- Inference in GPs is tractable (though at polynomial cost  $\mathcal{O}(N^3)$  in the number  $N$  of datapoints)
- There is no unique kernel. In fact, there are quite a few! E.g.

$$k(a, b) = \exp(-(a - b)^2)$$

Gaussian / Square Exponential / RBF kernel

$$k(a, b) = \min(a - t_0, b - t_0)$$

Wiener process

$$k(a, b) = \frac{1}{3} \min^3(a - t_0, b - t_0)$$

cubic spline kernel

$$+ \frac{1}{2} |a - b| \cdot \min^2(a - t_0, b - t_0)$$

$$k(a, b) = \frac{2}{\pi} \sin^{-1} \left( \frac{2a^\top b}{\sqrt{(1 + 2a^\top a)(1 + 2b^\top b)}} \right)$$

Neural Network kernel (Williams, 1998)



How many kernels are there?



# A Lot!

because the space of kernels is large

## Theorem:

Let  $\mathbb{X}, \mathbb{Y}$  be index sets and  $\phi : \mathbb{Y} \rightarrow \mathbb{X}$ . If  $k_1, k_2 : \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{R}$  are Mercer kernels, then the following functions are also Mercer kernels (up to minor regularity assumptions)

- $\alpha \cdot k_1(a, b)$  for  $\alpha \in \mathbb{R}_+$  (proof: trivial)
- $k_1(\phi(c), \phi(d))$  for  $c, d \in \mathbb{Y}$  (proof: by **Mercer's theorem**, next lecture)
- $k_1(a, b) + k_2(a, b)$  (proof: trivial)
- $k_1(a, b) \cdot k_2(a, b)$  **Schur product theorem**  
(proof involved. E.g. Bapat, 1997. Million, 2007)



# Scaled Kernels are Kernels

(this affects the effect of noise)

$$k(a, b) = 1^2 \cdot \exp\left(-\frac{(a - b)^2}{2}\right)$$



# Scaled Kernels are Kernels

(this affects the effect of noise)

$$k(a, b) = 10^2 \cdot \exp\left(-\frac{(a - b)^2}{2}\right)$$



# Kernels over Transformed Inputs are Kernels

this can be used to define length-scales

$$k(a, b) = 20 \cdot \exp\left(-\frac{(a - b)^2}{2 \cdot 5^2}\right)$$



# Kernels over Transformed Inputs are Kernels

this can be used to define length-scales

$$k(a, b) = 20 \cdot \exp\left(-\frac{(a - b)^2}{2 \cdot 0.5^2}\right)$$



# Kernels over Transformed Inputs are Kernels

nonlinear transformations (also at the heart of deep learning)

$$k(a, b) = 20 \cdot \exp\left(-\frac{(\phi(a) - \phi(b))^2}{2}\right) \quad \phi(a) = \left(\frac{a+8}{5}\right)^3$$



# Sums of Kernels are Kernels

$f \sim \mathcal{GP}(0, k_1)$  and  $g \sim \mathcal{GP}(0, k_2) \quad \Leftrightarrow \quad h = f + g \sim \mathcal{GP}(0, k_1 + k_2)$

$$k(a, b) = 20 \cdot \exp\left(-\frac{(a - b)^2}{2}\right) + \phi(a)^\top \phi(b) \quad \phi(x) := [1, x, x^2]$$



# Products of Kernels are Kernels

Schur product theorem

$$k(a, b) = 20 \cdot \exp\left(-\frac{(a - b)^2}{2}\right) \cdot \phi(a)^\top \phi(b) \quad \phi(x) := \left(\frac{x+8}{5}\right)^2$$

If you think your model has no parameters, you just haven't found them yet!

# Can We Learn the Kernel?

Bayesian Hierarchical Inference, vol. 2

$$p(f | \theta) = \mathcal{GP}(f; m_\theta, k_\theta) \quad \text{e.g. } m_\theta(x) = \phi(x)^\top \boldsymbol{\theta}, \text{ and/or } k_\theta(a, b) = \theta_1 \exp\left(-\frac{(a - b)^2}{2\theta_2^2}\right).$$

- recall from last lecture: the **evidence** in Bayes' theorem is the **marginal likelihood** for the model

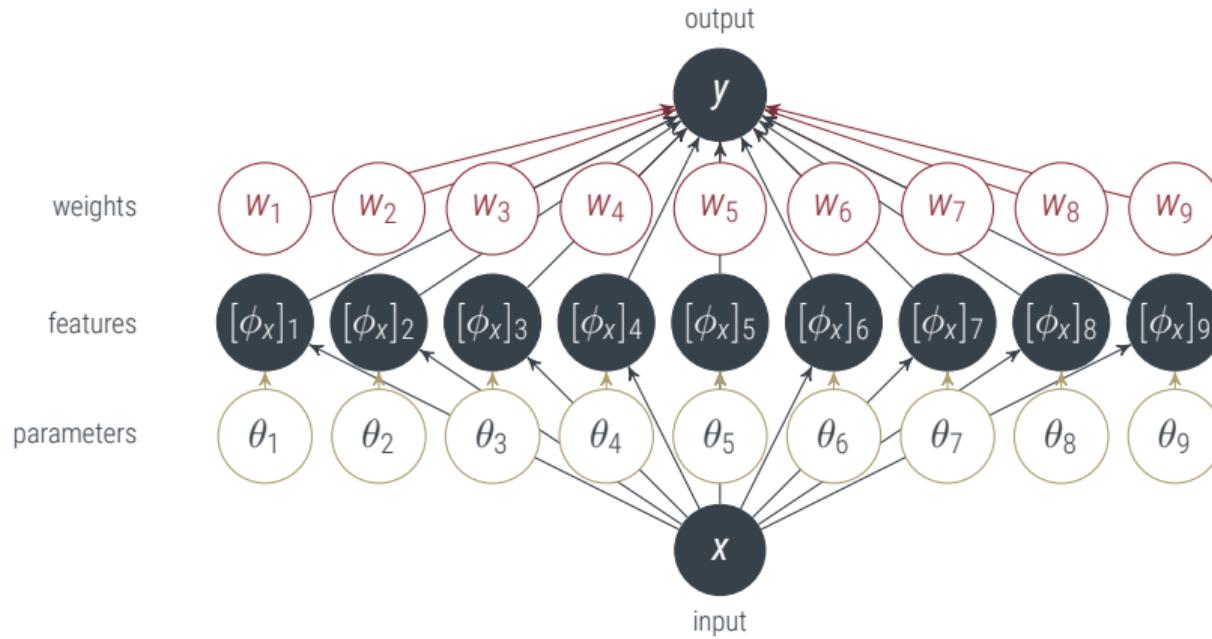
$$p(f | y, x, \boldsymbol{\theta}) = \frac{p(y | f, x, \boldsymbol{\theta}) p(f |, \boldsymbol{\theta})}{\int p(y | f, x, \boldsymbol{\theta}) p(f |, \boldsymbol{\theta}) df} = \frac{p(y | f, x, \boldsymbol{\theta}) p(f |, \boldsymbol{\theta})}{p(y | x, \boldsymbol{\theta})}$$

- For Gaussians, die evidence has **analytic form**:

$$\underbrace{\mathcal{N}(y; \phi_X^{\boldsymbol{\theta}^\top} w + b, \Lambda)}_{p(y|f, x, \boldsymbol{\theta})} \cdot \underbrace{\mathcal{N}(w, \mu, \Sigma)}_{p(f)} = \underbrace{\mathcal{N}(w; m_{\text{post}}^{\boldsymbol{\theta}}, V_{\text{post}}^{\boldsymbol{\theta}})}_{p(f|y, x, \boldsymbol{\theta})} \cdot \underbrace{\mathcal{N}(y; \phi_X^{\boldsymbol{\theta}^\top} \mu + b, \phi_X^{\boldsymbol{\theta}^\top} \Sigma \phi_X^{\boldsymbol{\theta}} + \Lambda)}_{p(y|\boldsymbol{\theta}, x)}$$

# Kernel Learning is Fitting an Entire Population of Features

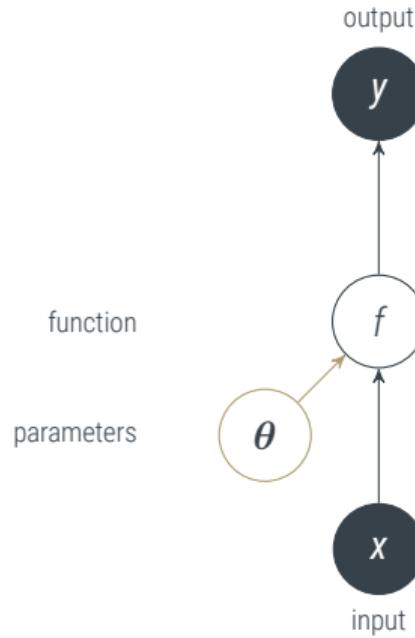
The Graphical Model View





# Kernel Learning is Fitting an Entire Population of Features

The Graphical Model View





## The Toolbox

Five principal methods for dealing with computational complexity in probabilistic inference

1. Maximum Likelihood (ML) / Maximum A-Posteriori (MAP) estimation:  
To estimate  $\theta$  in  $p(D | \theta)$  or  $p(\theta | D)$ , set  $\hat{\theta} = \arg \max_{\theta} p$ .
2. ????
3. ????
4. ????
5. ????

Disclaimer: The listed items are neither mutually exclusive nor collectively exhaustive. Some of the methods are intricately interrelated.



## The Toolbox

Five principal methods for dealing with computational complexity in probabilistic inference

1. Maximum Likelihood (ML) / Maximum A-Posteriori (MAP) estimation:

To estimate  $\theta$  in  $p(D | \theta)$  or  $p(\theta | D)$ , set  $\hat{\theta} = \arg \max_{\theta} p$ .

2. ????

3. ????

4. ????

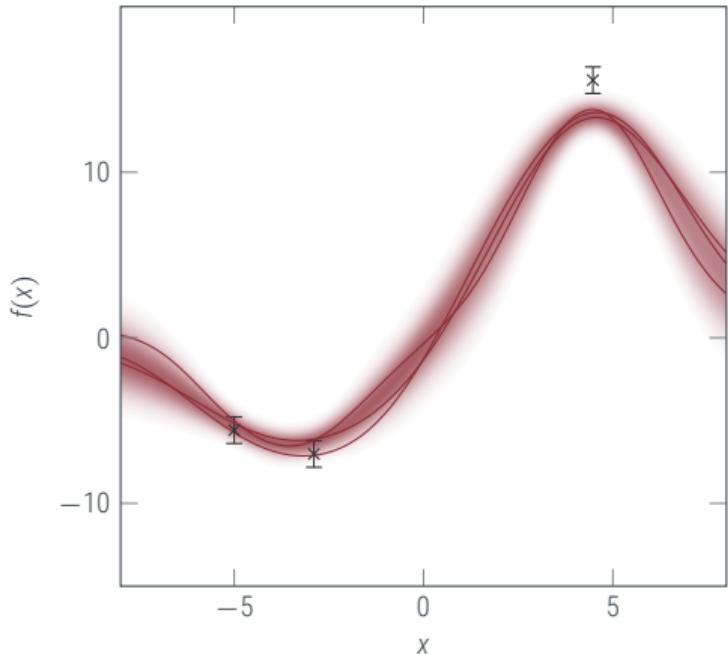
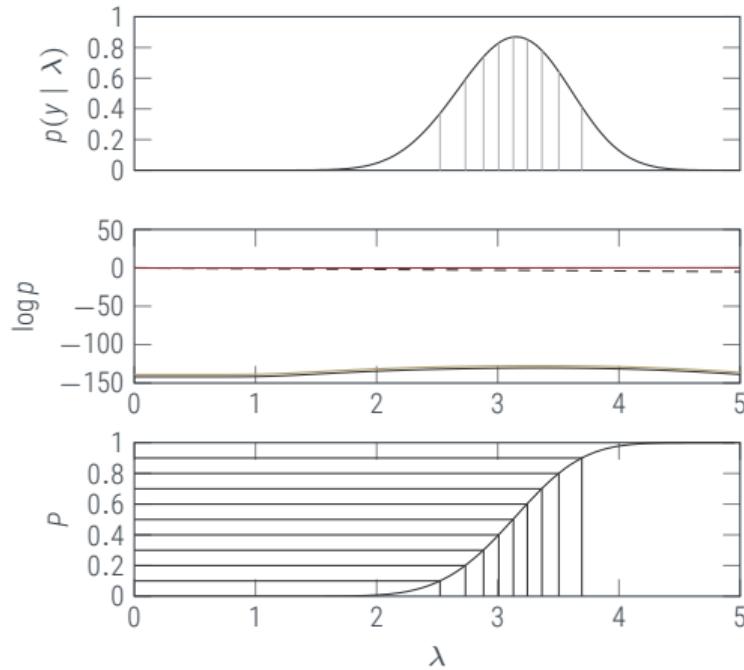
5. Numerical Quadrature:

To marginalize  $\theta$ , compute  $\int p(f | \theta) d\theta \approx \sum_i w_i \cdot p(f | \theta_i)$

Disclaimer: The listed items are neither mutually exclusive nor collectively exhaustive. Some of the methods are intricately interrelated.

# Quadrature (currently) only works well for low-dimensional spaces

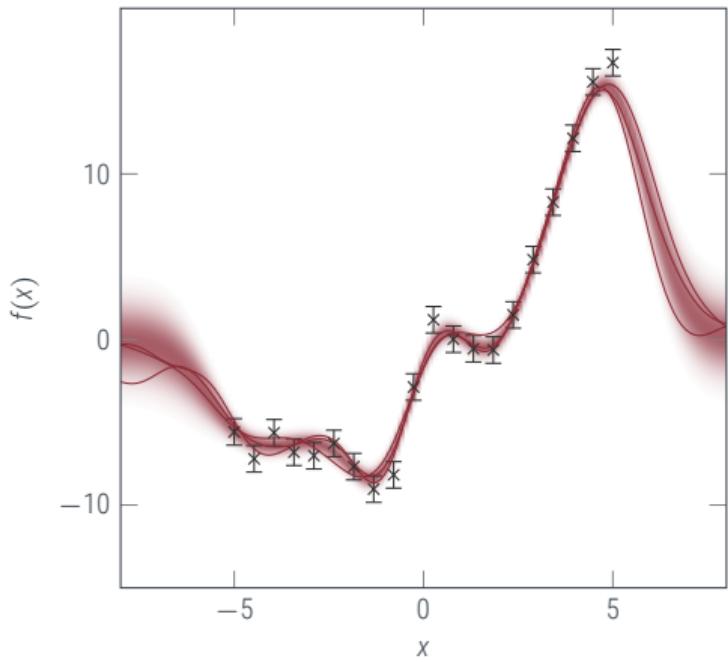
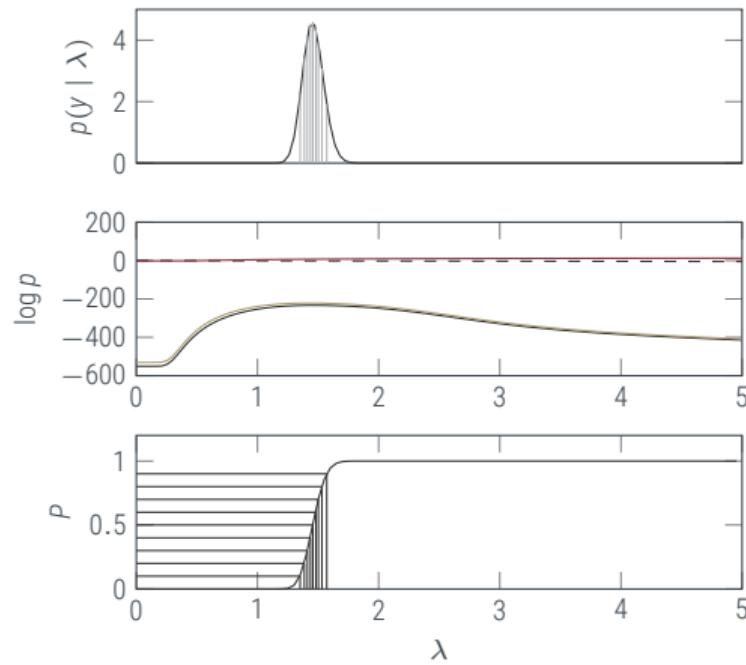
but if you *really* need to be precise, it's the best option





# Quadrature (currently) only works well for low-dimensional spaces

but if you *really* need to be precise, it's the best option

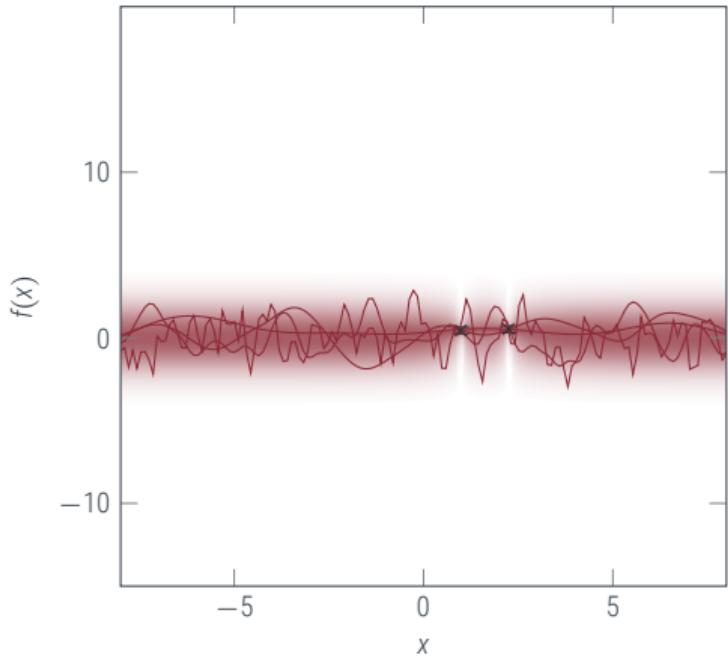
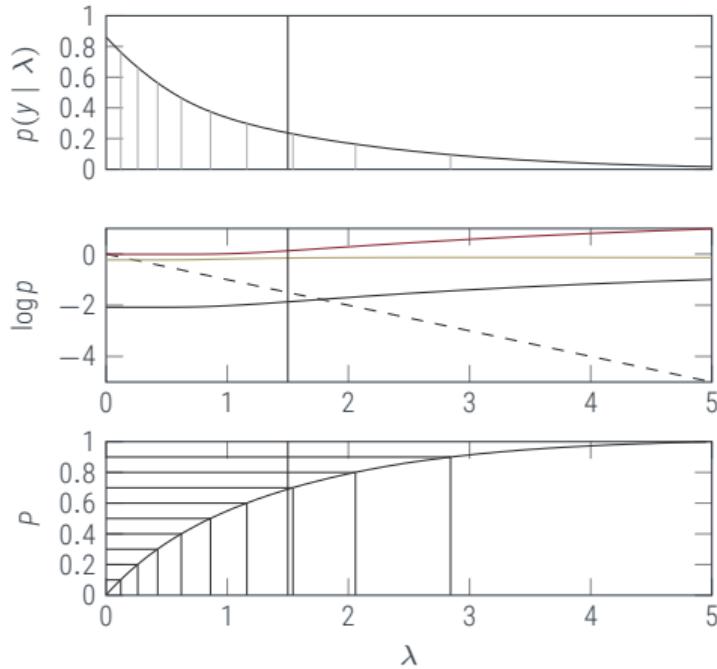


Question: Why does the posterior contract to such a "surprising" point?



# If the model is correct, the posterior is calibrated

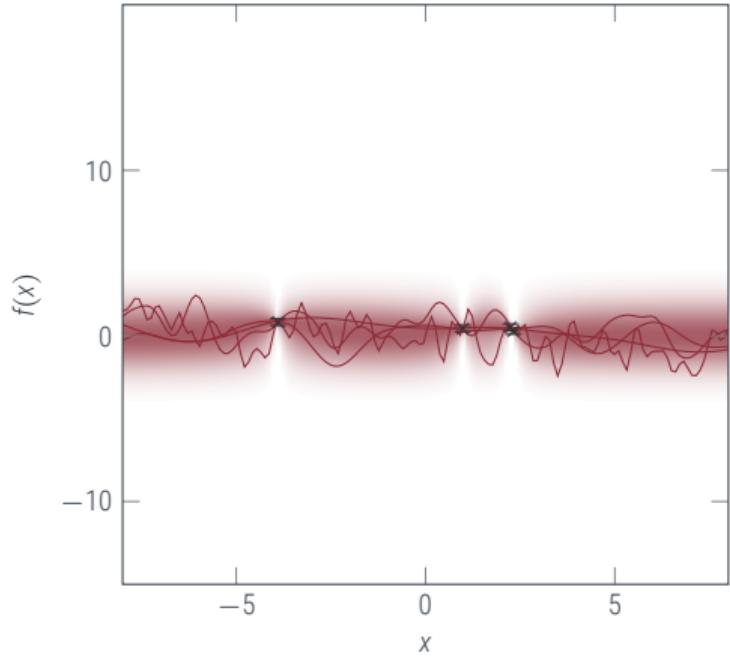
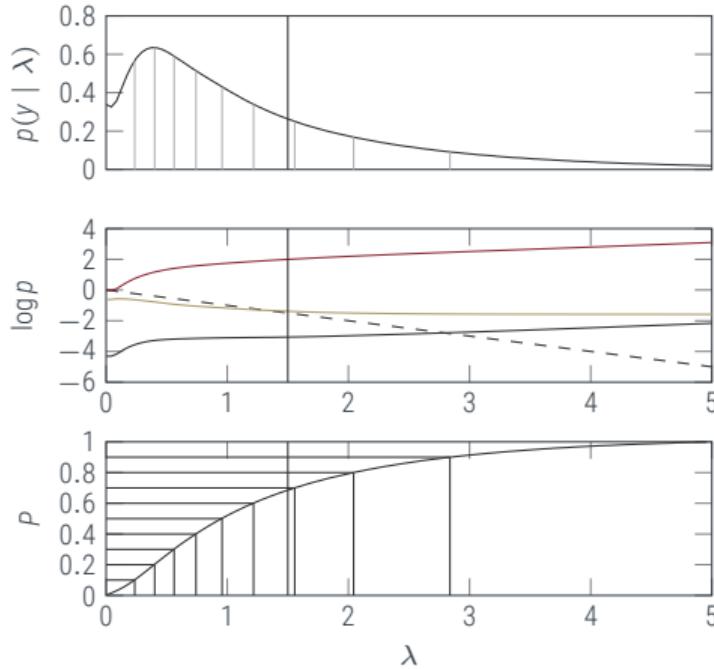
draws from ground truth model





# If the model is correct, the posterior is calibrated

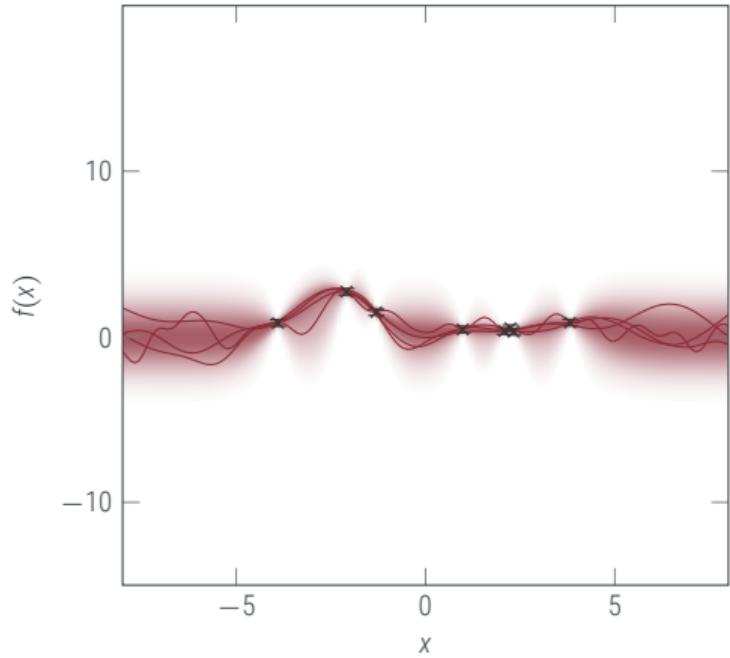
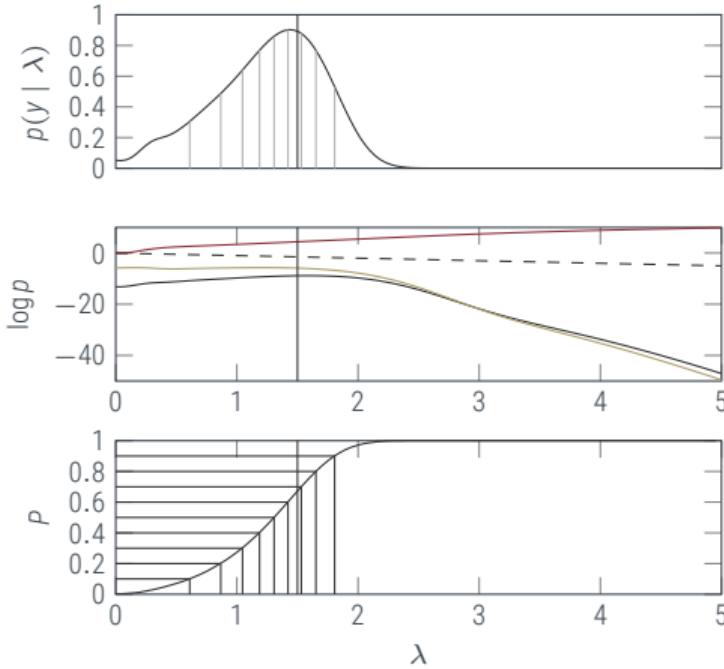
draws from ground truth model





# If the model is correct, the posterior is calibrated

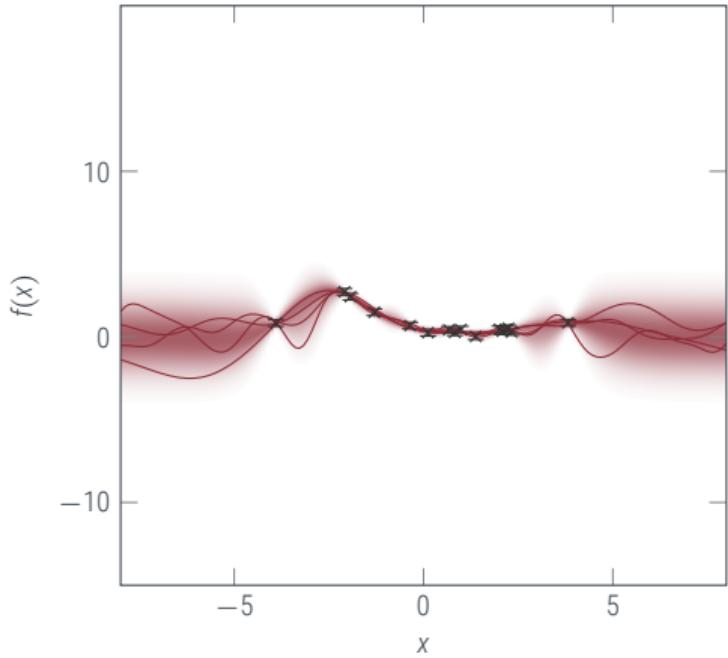
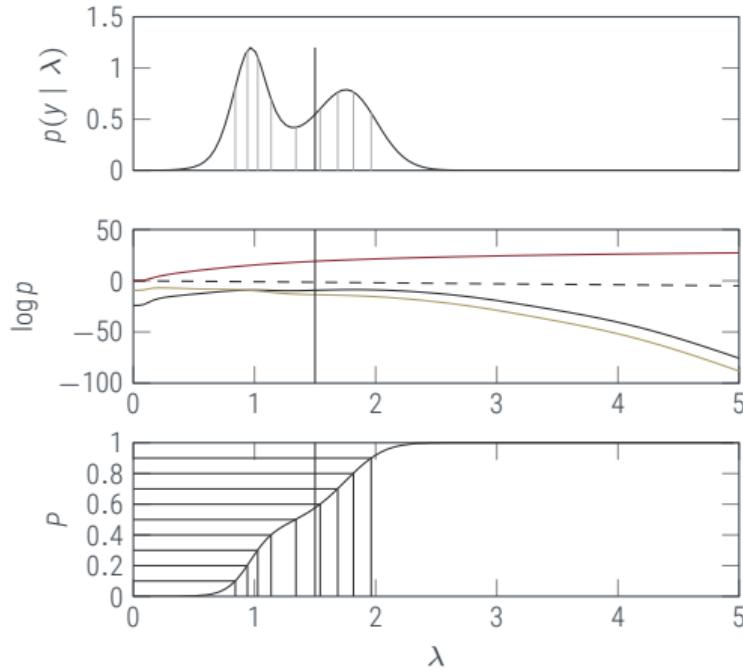
draws from ground truth model





# If the model is correct, the posterior is calibrated

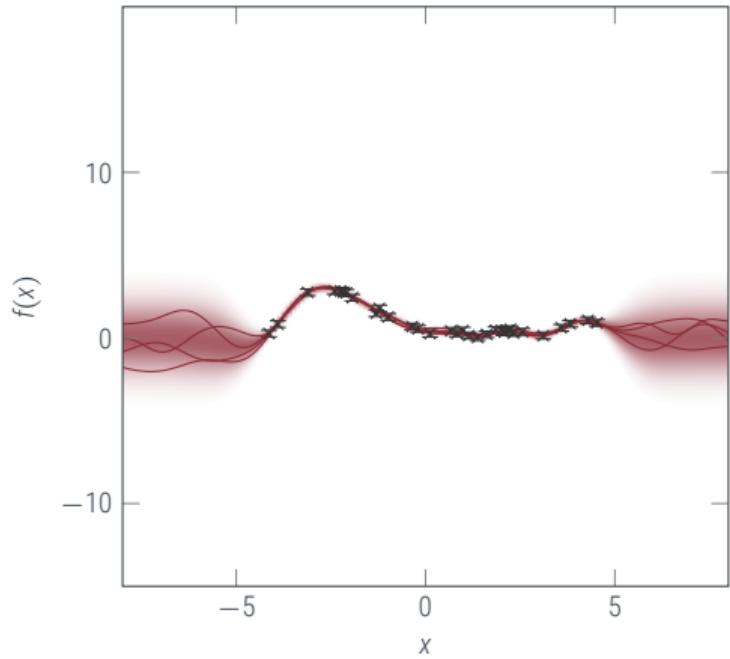
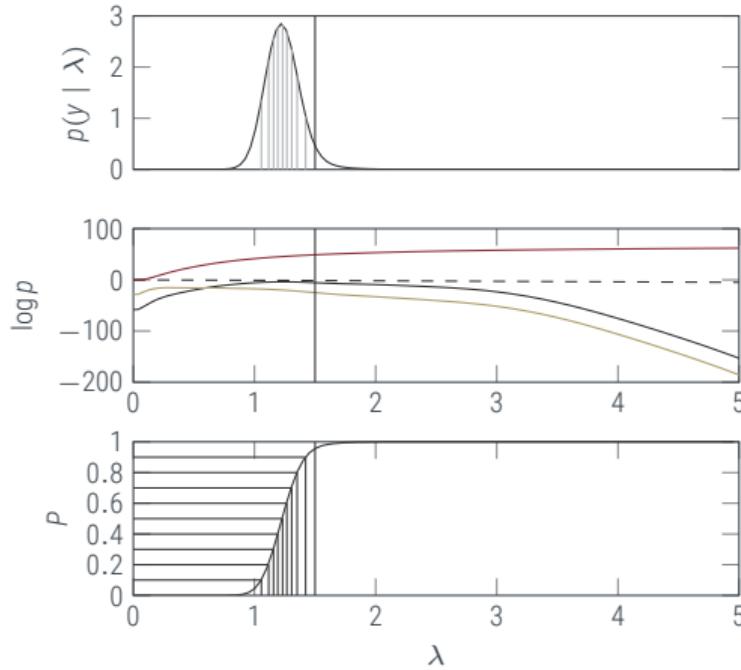
draws from ground truth model





# If the model is correct, the posterior is calibrated

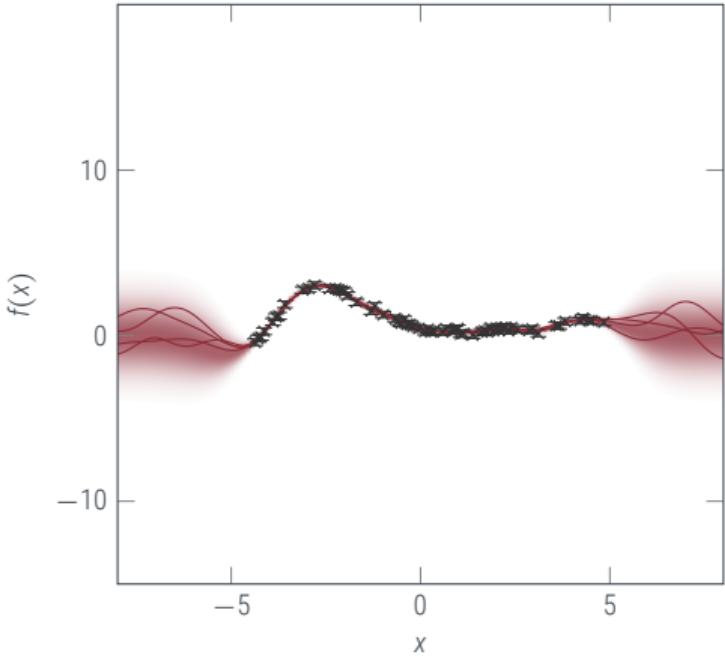
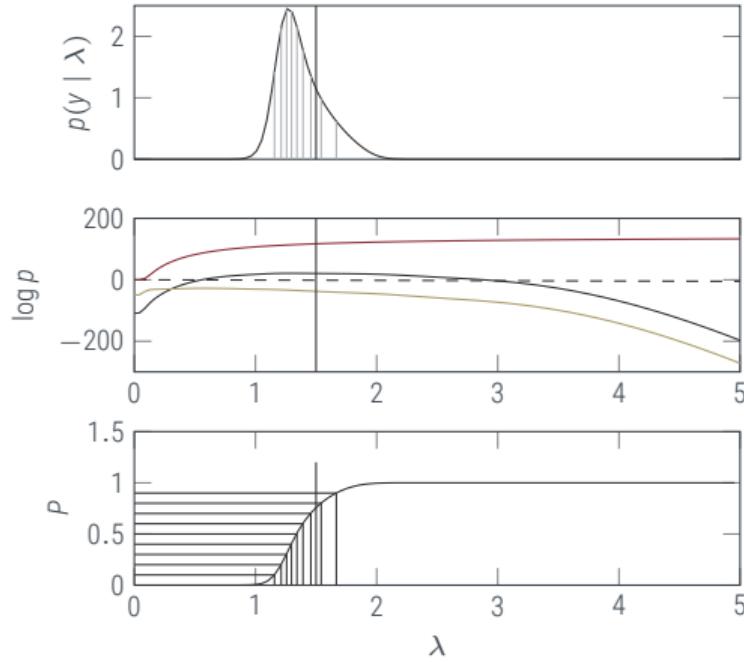
draws from ground truth model





# If the model is correct, the posterior is calibrated

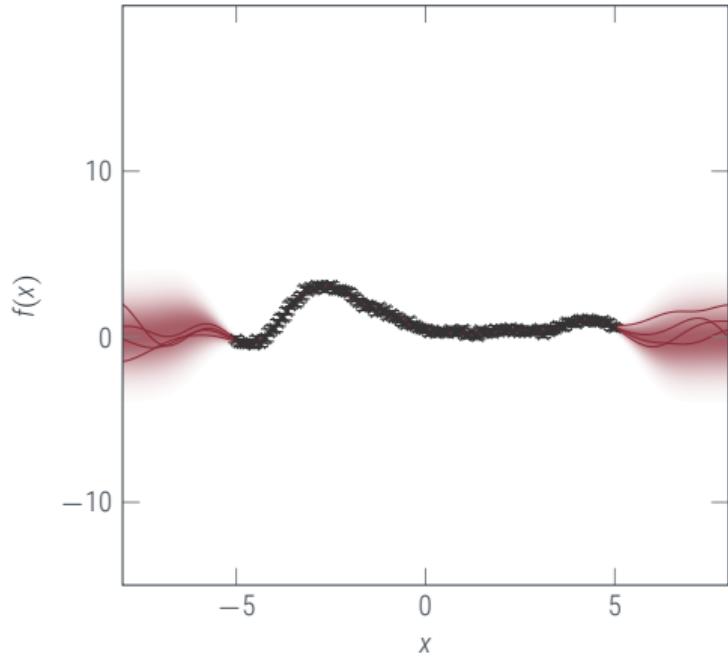
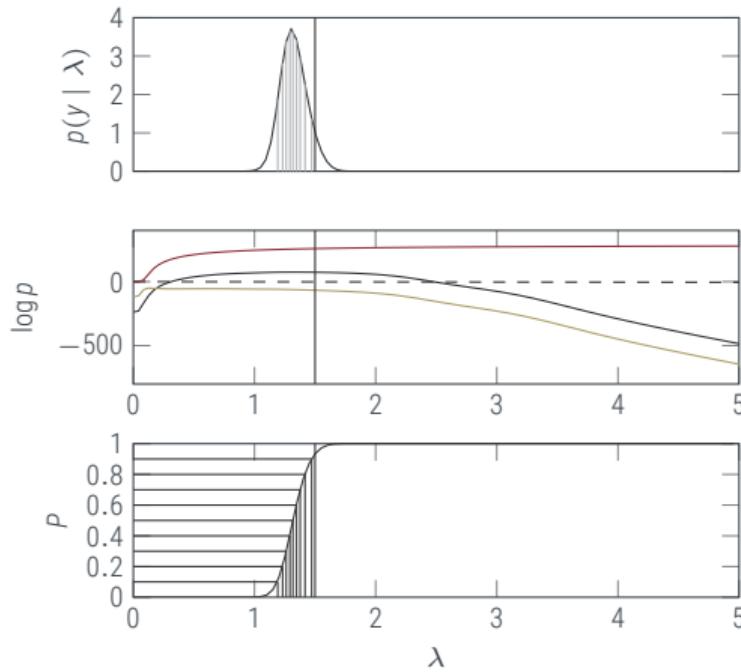
draws from ground truth model





# If the model is correct, the posterior is calibrated

draws from ground truth model





## Summary: Gaussian Processes

- Sometimes it is possible to consider **infinitely many** features at once, by extending from a sum to an integral.
- The resulting **nonparametric model** is known as a **Gaussian process**
- Inference in GPs is tractable (though at polynomial cost  $\mathcal{O}(N^3)$  in the number  $N$  of datapoints)
- There is an entire semi-ring of kernels: If  $k_1, k_2$  are kernels,  $\phi$  any function, then so are
  - $\alpha \cdot k_1(a, b)$  for  $\alpha \in \mathbb{R}_+$
  - $k_1(\phi(c), \phi(d))$  for  $c, d \in \mathbb{Y}$
  - $k_1(a, b) + k_2(a, b)$
  - $k_1(a, b) \cdot k_2(a, b)$
- Kernels can be **learned** just like features, e.g. by type-II MAP/ML or numerical integration

Gaussian processes are an extremely important basic model for supervised regression. They have been used in many practical applications, but also provide a theoretical foundation for more complicated models, e.g. in classification, unsupervised learning, deep learning. More later.

Next lecture: Some **theory** for Gaussian processes. If you don't like pretty pictures, wait for it!