Lecture 3: Gaussian Processes

Jens Kober

Knowledge-Based Control Systems (SC42050)

Cognitive Robotics

3mE, Delft University of Technology, The Netherlands

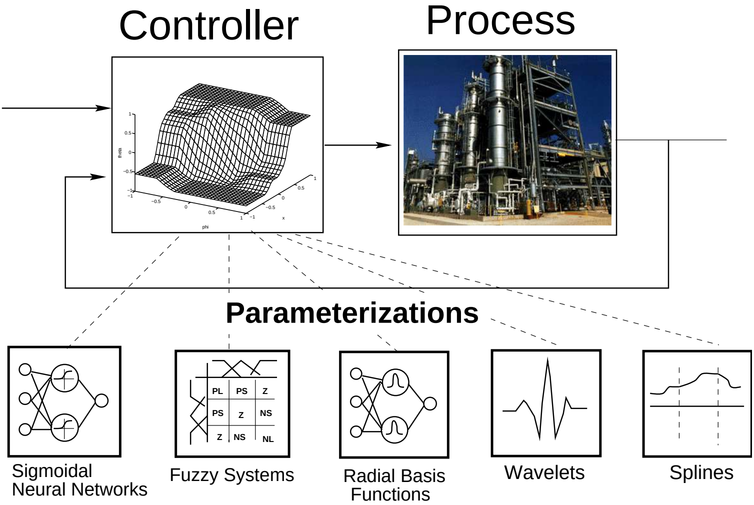
19-02-2018

Outline

1. Properties
2. Gaussian distributions
3. Inference
4. A different representation
5. Gaussian processes
6. Kernels
7. Applications

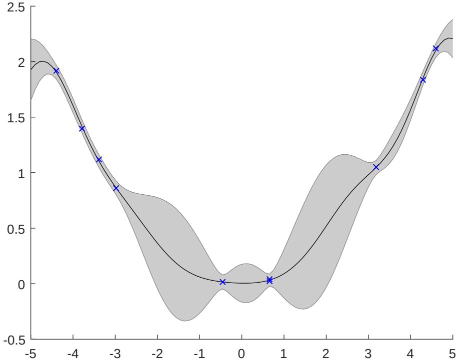
Lecture based on lectures by David MacKay and Oliver Stegle & Karsten Borgwardt

Function Approximators



Properties of Gaussian Processes

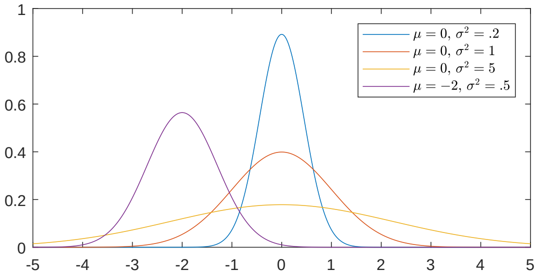
- nonlinear regression
- accurate and flexible
- predictions with error bars
- non-parametric



1D Gaussian Distribution

- Normal distribution
- Probability density function:

$$\mathcal{N}(y|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(y-\mu)^2}{\sigma^2}}$$

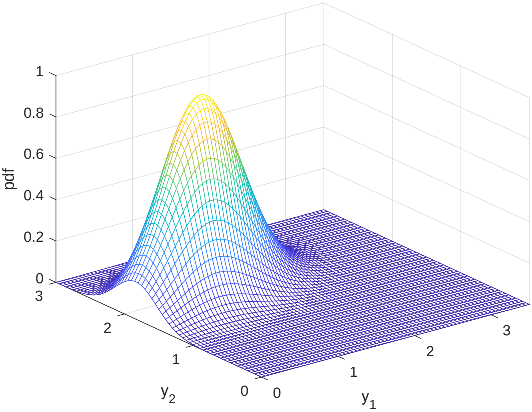


nD Gaussian Distribution

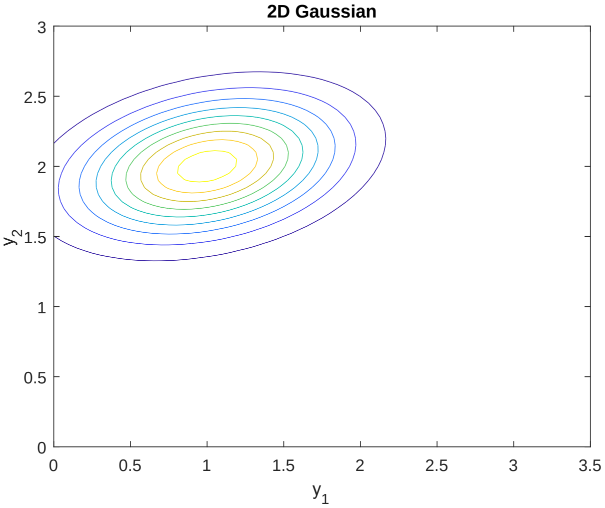
- Probability density function:

$$\mathcal{N}(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{K}) = \frac{1}{\sqrt{\det(2\pi\boldsymbol{K})}} e^{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^\top \boldsymbol{K}^{-1}(\boldsymbol{y}-\boldsymbol{\mu})}$$

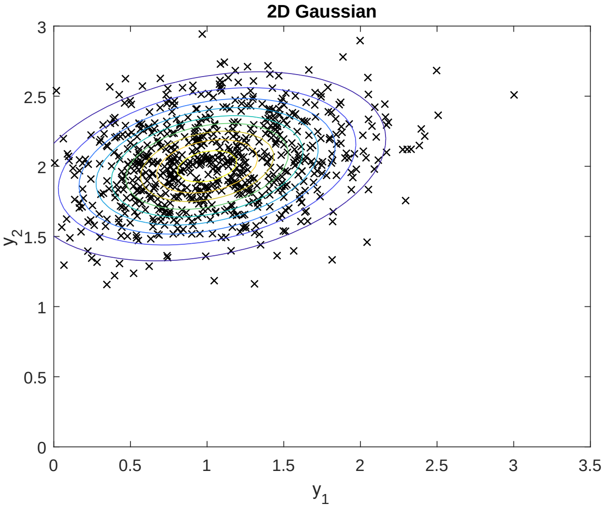
- $\boldsymbol{K}$ is the covariance matrix (or kernel matrix)



2D Gaussian Distribution: Contour

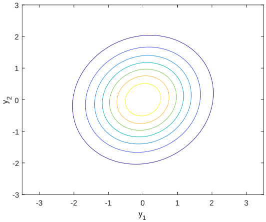


2D Gaussian Distribution: Contour & Samples

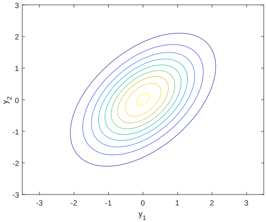


2D Gaussian Distribution: Covariance Matrices

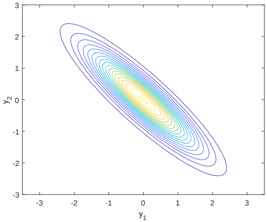
$\boldsymbol{K}$ is positive-semidefinite and symmetric



$$\boldsymbol{K} = \begin{bmatrix} 1 & .1 \\ .1 & 1 \end{bmatrix}$$

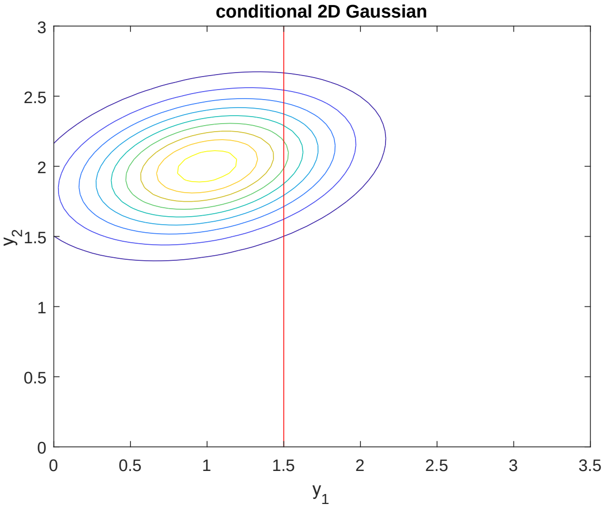


$$\boldsymbol{K} = \begin{bmatrix} 1 & .5 \\ .5 & 1 \end{bmatrix}$$

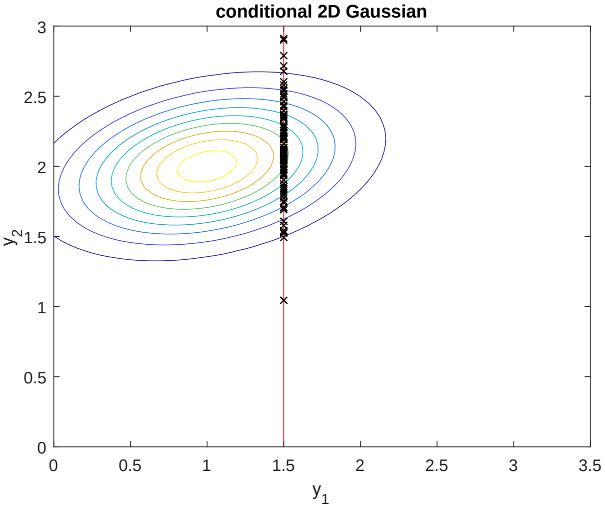


$$\boldsymbol{K} = \begin{bmatrix} 1 & -.9 \\ -.9 & 1 \end{bmatrix}$$

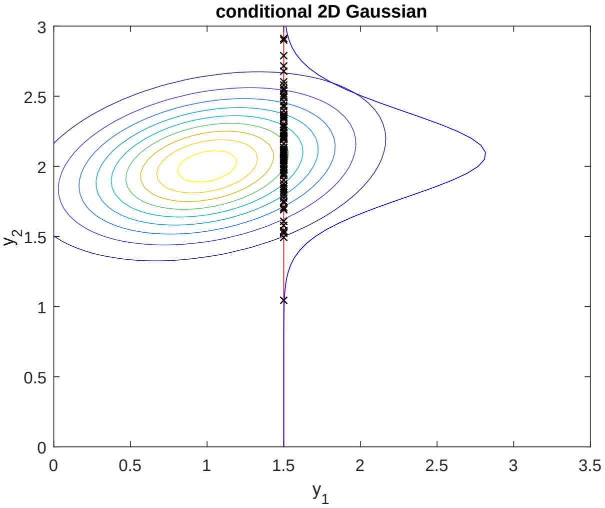
Inference



Inference



Inference



Inference

- Joint probability $p(\boldsymbol{y}_1, \boldsymbol{y}_2 | \boldsymbol{\mu}, \boldsymbol{K}) = \mathcal{N}([\boldsymbol{y}_1, \boldsymbol{y}_2] | \boldsymbol{\mu}, \boldsymbol{K})$

Inference

- Joint probability $p(\boldsymbol{y}_1, \boldsymbol{y}_2 | \boldsymbol{\mu}, \boldsymbol{K}) = \mathcal{N}([\boldsymbol{y}_1, \boldsymbol{y}_2] | \boldsymbol{\mu}, \boldsymbol{K})$
- Conditional probability $\boldsymbol{y}_1 = \boldsymbol{a}$

$$p(\boldsymbol{y}_2 | \boldsymbol{y}_1, \boldsymbol{\mu}, \boldsymbol{K}) = \frac{p(\boldsymbol{y}_1, \boldsymbol{y}_2 | \boldsymbol{\mu}, \boldsymbol{K})}{p(\boldsymbol{y}_1 | \boldsymbol{\mu}, \boldsymbol{K})} = \mathcal{N}(\boldsymbol{y}_2 | \bar{\boldsymbol{\mu}}, \bar{\boldsymbol{K}})$$

Inference

- Joint probability $p(\boldsymbol{y}_1, \boldsymbol{y}_2 | \boldsymbol{\mu}, \boldsymbol{K}) = \mathcal{N}([\boldsymbol{y}_1, \boldsymbol{y}_2] | \boldsymbol{\mu}, \boldsymbol{K})$
- Conditional probability $\boldsymbol{y}_1 = \boldsymbol{a}$

$$p(\boldsymbol{y}_2 | \boldsymbol{y}_1, \boldsymbol{\mu}, \boldsymbol{K}) = \frac{p(\boldsymbol{y}_1, \boldsymbol{y}_2 | \boldsymbol{\mu}, \boldsymbol{K})}{p(\boldsymbol{y}_1 | \boldsymbol{\mu}, \boldsymbol{K})} = \mathcal{N}(\boldsymbol{y}_2 | \bar{\boldsymbol{\mu}}, \bar{\boldsymbol{K}})$$

- Partitioning:

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{bmatrix} \qquad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} \qquad \boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_{11} & \boldsymbol{K}_{12} \\ \boldsymbol{K}_{21} & \boldsymbol{K}_{22} \end{bmatrix}$$

Inference

- Joint probability $p(\boldsymbol{y}_1, \boldsymbol{y}_2 | \boldsymbol{\mu}, \boldsymbol{K}) = \mathcal{N}([\boldsymbol{y}_1, \boldsymbol{y}_2] | \boldsymbol{\mu}, \boldsymbol{K})$
- Conditional probability $\boldsymbol{y}_1 = \boldsymbol{a}$

$$p(\boldsymbol{y}_2 | \boldsymbol{y}_1, \boldsymbol{\mu}, \boldsymbol{K}) = \frac{p(\boldsymbol{y}_1, \boldsymbol{y}_2 | \boldsymbol{\mu}, \boldsymbol{K})}{p(\boldsymbol{y}_1 | \boldsymbol{\mu}, \boldsymbol{K})} = \mathcal{N}(\boldsymbol{y}_2 | \bar{\boldsymbol{\mu}}, \bar{\boldsymbol{K}})$$

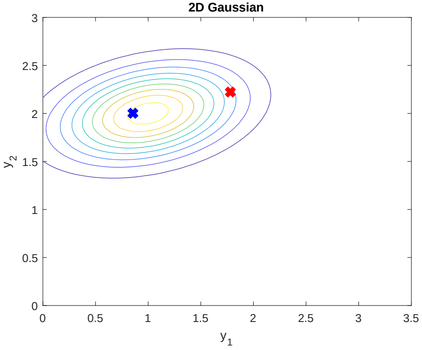
- Partitioning:

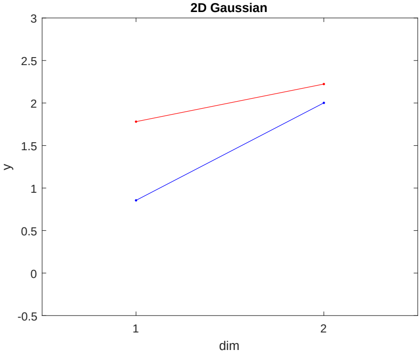
$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \end{bmatrix} \qquad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} \qquad \boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_{11} & \boldsymbol{K}_{12} \\ \boldsymbol{K}_{21} & \boldsymbol{K}_{22} \end{bmatrix}$$

- New mean and covariance

$$\bar{\boldsymbol{\mu}} = \boldsymbol{\mu}_2 + \boldsymbol{K}_{21} \boldsymbol{K}_{11}^{-1} (\boldsymbol{a} - \boldsymbol{\mu}_1) \qquad \bar{\boldsymbol{K}} = \boldsymbol{K}_{22} - \boldsymbol{K}_{21} \boldsymbol{K}_{11}^{-1} \boldsymbol{K}_{12}$$

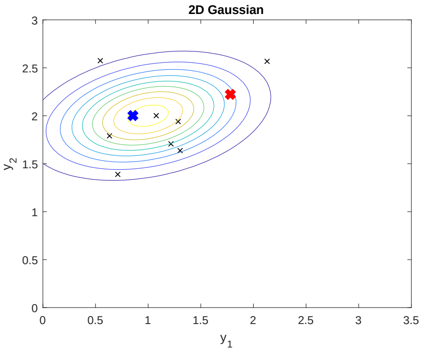
A Different Representation

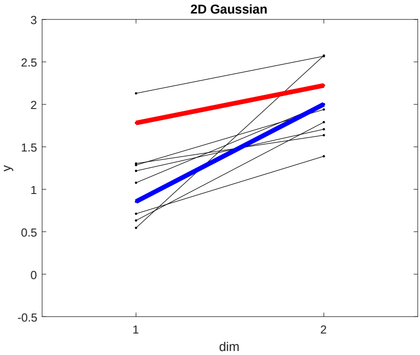




$y = \begin{bmatrix} 0.86 & 2.00 \end{bmatrix}$ $y = \begin{bmatrix} 1.78 & 2.22 \end{bmatrix}$

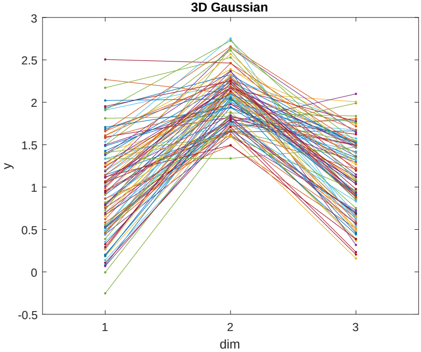
A Different Representation



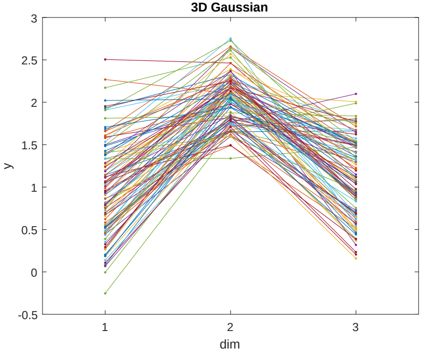


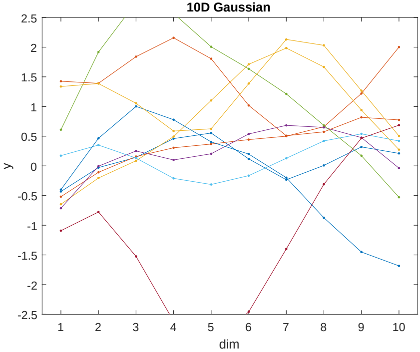
$y = \begin{bmatrix} 0.86 & 2.00 \end{bmatrix}$ $y = \begin{bmatrix} 1.78 & 2.22 \end{bmatrix}$

Higher-Dimensional Gaussians

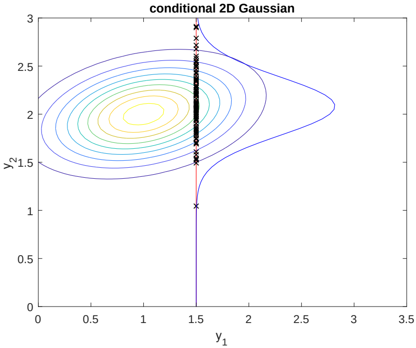


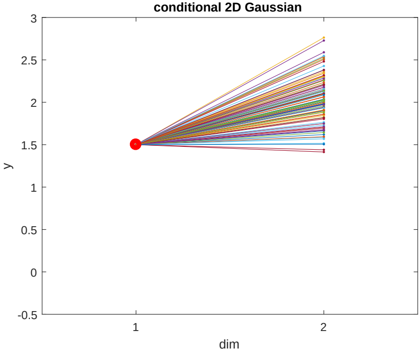
Higher-Dimensional Gaussians



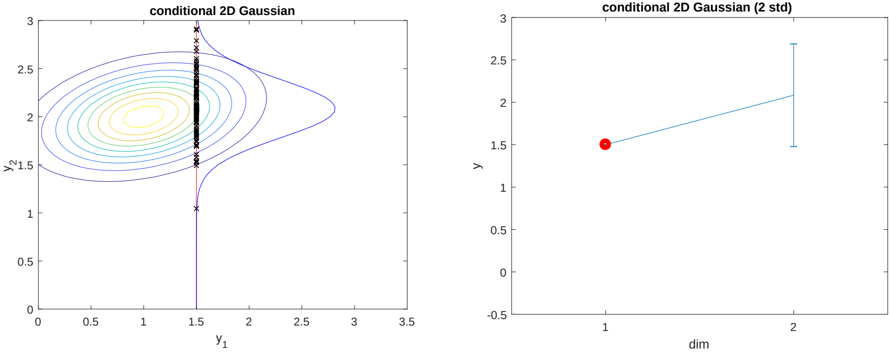


2D Conditional

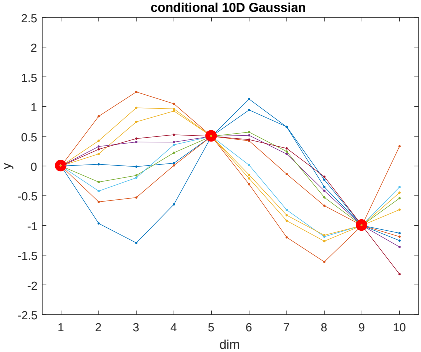


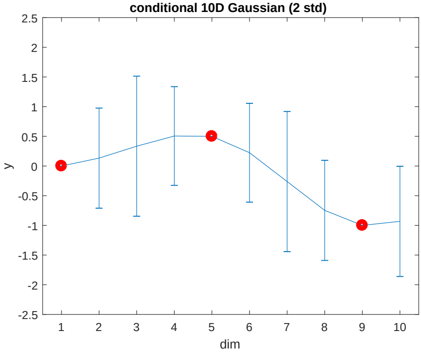


2D Conditional

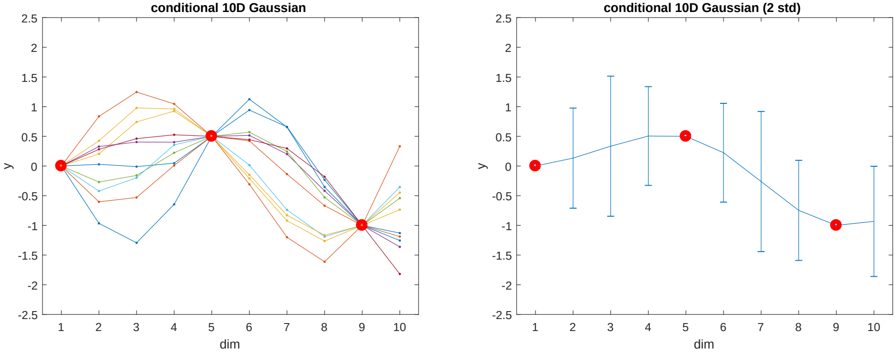


10D Conditional



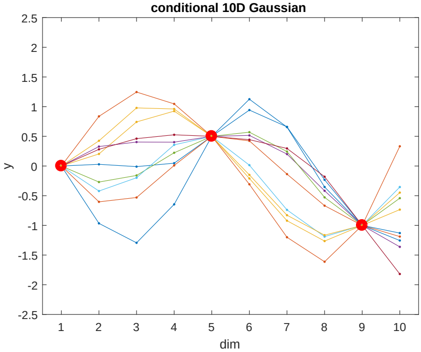


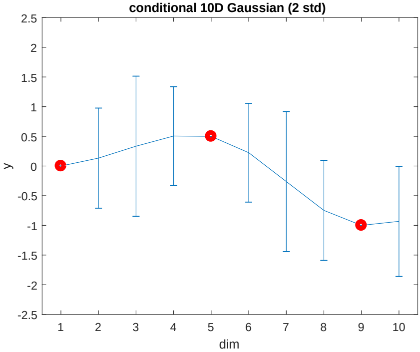
10D Conditional



- Looks like nonlinear regression!

10D Conditional





- Looks like nonlinear regression!
- Where did $\boldsymbol{K}$ come from?

10D Conditional: $\boldsymbol{K}$

$$
\boldsymbol{K} = \begin{bmatrix}
1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 & 0.00 & 0.00 \\
0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 & 0.00 \\
0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 \\
0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 \\
0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 \\
0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 \\
0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 \\
0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 \\
0.00 & 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 \\
0.00 & 0.00 & 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00
\end{bmatrix}
$$

10D Conditional: $\boldsymbol{K}$

$$\boldsymbol{K} = \begin{bmatrix} 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 & 0.00 & 0.00 \\ 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 & 0.00 \\ 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 \\ 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 \\ 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 \\ 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 \\ 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 \\ 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 \\ 0.00 & 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 \\ 0.00 & 0.00 & 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 \end{bmatrix}$$

- Diagonal structure

10D Conditional: $\boldsymbol{K}$

$$\boldsymbol{K} = \begin{bmatrix} 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 & 0.00 & 0.00 \\ 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 & 0.00 \\ 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 \\ 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 \\ 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 \\ 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 \\ 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 \\ 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 \\ 0.00 & 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 \\ 0.00 & 0.00 & 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 \end{bmatrix}$$

- Diagonal structure
- Generated by a covariance function (kernel function)

$$\boldsymbol{K}_{i,j} = k(x_i, x_j; \boldsymbol{\theta}_K)$$

10D Conditional: $\boldsymbol{K}$

$$\boldsymbol{K} = \begin{bmatrix} 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 & 0.00 & 0.00 \\ 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 & 0.00 \\ 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 & 0.00 \\ 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 & 0.01 \\ 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 & 0.04 \\ 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 & 0.14 \\ 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 & 0.32 \\ 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 & 0.61 \\ 0.00 & 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 & 0.88 \\ 0.00 & 0.00 & 0.00 & 0.01 & 0.04 & 0.14 & 0.32 & 0.61 & 0.88 & 1.00 \end{bmatrix}$$

- Diagonal structure
- Generated by a covariance function (kernel function)

$$\boldsymbol{K}_{i,j} = k(x_i, x_j; \boldsymbol{\theta}_K)$$

- Here:

$$k_{\mathrm{SE}}(x_i, x_j; \sigma_f, l) = \sigma_f^2 e^{-\frac{1}{2l^2}(x_i - x_j)^2}$$

squared exponential kernel (hyperparameters: l horizontal lengthscale, σ_f vertical lengthscale)

Definition

Gaussian Process

A Gaussian process is a collection of random variables with the property that the joint distribution of any finite subset is a Gaussian.

Function Space View

- Kernel function and hyperparameters = prior belief on function properties (smoothness, lenghtscales, etc.)

Function Space View

- Kernel function and hyperparameters = prior belief on function properties (smoothness, lenghtscales, etc.)
- Construct a joint Gaussian for an arbitrary selection of input locations $\boldsymbol{X}$

Function Space View

- Kernel function and hyperparameters = prior belief on function properties (smoothness, lenghtscales, etc.)
- Construct a joint Gaussian for an arbitrary selection of input locations $\boldsymbol{X}$
- Prior on infinite function $f(x)$

$$p(f(x)) = \mathrm{GP}\left(f(x)|k\right)$$

Function Space View

- Kernel function and hyperparameters = prior belief on function properties (smoothness, lenghtscales, etc.)
- Construct a joint Gaussian for an arbitrary selection of input locations $\boldsymbol{X}$
- Prior on infinite function $f(x)$

$$p(f(x)) = \mathrm{GP}\left(f(x)|k\right)$$

- Prior on function values $\boldsymbol{f} = (f_1, \ldots, f_N)$

$$p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{\theta}_K) = \mathcal{N}\left(\boldsymbol{f}|\mathbf{0}, \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}}(\boldsymbol{\theta}_K)\right)$$

Noise-free Observations

- Noise-free training data $\mathcal{D} = \{\boldsymbol{x}_n, f_n\}_{n=1}^{N}$

Noise-free Observations

- Noise-free training data $\mathcal{D} = \{\boldsymbol{x}_n, f_n\}_{n=1}^{N}$
- Want to predict $\boldsymbol{f}^*$ at test points $\boldsymbol{X}^*$

Noise-free Observations

- Noise-free training data $\mathcal{D} = \{\boldsymbol{x}_n, f_n\}_{n=1}^N$
- Want to predict $\boldsymbol{f}^*$ at test points $\boldsymbol{X}^*$
- Joint distribution of $\boldsymbol{f}$ and $\boldsymbol{f}^*$:

$$p([\boldsymbol{f}, \boldsymbol{f}^*] | \boldsymbol{X}, \boldsymbol{X}^*, \boldsymbol{\theta}_K) = \mathcal{N}\left([\boldsymbol{f}, \boldsymbol{f}^*] \middle| \mathbf{0}, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}^*} \\ \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}^*} \end{bmatrix}\right)$$

(dependence of $\boldsymbol{K}$ on $\boldsymbol{\theta}_K$ dropped for brevity)

Noise-free Observations

- Noise-free training data $\mathcal{D} = \{\boldsymbol{x}_n, f_n\}_{n=1}^{N}$
- Want to predict $\boldsymbol{f}^*$ at test points $\boldsymbol{X}^*$
- Joint distribution of $\boldsymbol{f}$ and $\boldsymbol{f}^*$:

$$p([\boldsymbol{f}, \boldsymbol{f}^*] | \boldsymbol{X}, \boldsymbol{X}^*, \boldsymbol{\theta}_K) = \mathcal{N}\left([\boldsymbol{f}, \boldsymbol{f}^*] \middle| \boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}^*} \\ \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}^*} \end{bmatrix}\right)$$

  (dependence of $\boldsymbol{K}$ on $\boldsymbol{\theta}_K$ dropped for brevity)
- Real data is rarely noise-free

Inference

- Noisy training data $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$

Inference

- Noisy training data $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$
- Joint distribution of latent function values $\boldsymbol{f}$ and $\boldsymbol{f}^*$ given $\boldsymbol{y}$:

$$p([\boldsymbol{f}, \boldsymbol{f}^*] | \boldsymbol{X}, \boldsymbol{X}^*, \boldsymbol{y}, \boldsymbol{\theta}_K, \sigma^2) \propto \overbrace{\mathcal{N}\left([\boldsymbol{f}, \boldsymbol{f}^*] \middle| \mathbf{0}, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}^*} \\ \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}^*} \end{bmatrix}\right)}^{\text{Prior}} \times \underbrace{\prod_{n=1}^{N} \mathcal{N}\left(y_n | f_n, \sigma^2\right)}_{\text{Likelihood}}$$

Inference

- Noisy training data $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$
- Joint distribution of latent function values $\boldsymbol{f}$ and $\boldsymbol{f}^*$ given $\boldsymbol{y}$:

$$p([\boldsymbol{f}, \boldsymbol{f}^*] | \boldsymbol{X}, \boldsymbol{X}^*, \boldsymbol{y}, \boldsymbol{\theta}_K, \sigma^2) \propto \overbrace{\mathcal{N}\left([\boldsymbol{f}, \boldsymbol{f}^*] \middle| \boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}^*} \\ \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}^*} \end{bmatrix}\right)}^{\text{Prior}} \times \underbrace{\prod_{n=1}^{N} \mathcal{N}\left(y_n | f_n, \sigma^2\right)}_{\text{Likelihood}}$$

- Integrating out $\boldsymbol{f}$ yields

$$p([\boldsymbol{y}, \boldsymbol{f}^*] | \boldsymbol{X}, \boldsymbol{X}^*, \boldsymbol{y}, \boldsymbol{\theta}_K, \sigma^2) \propto \mathcal{N}\left([\boldsymbol{y}, \boldsymbol{f}^*] \middle| \boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma^2 \boldsymbol{I} & \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}^*} \\ \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}^*} \end{bmatrix}\right)$$

Predictions

- Turn into conditional distribution

Predictions

- Turn into conditional distribution
- Joint distribution

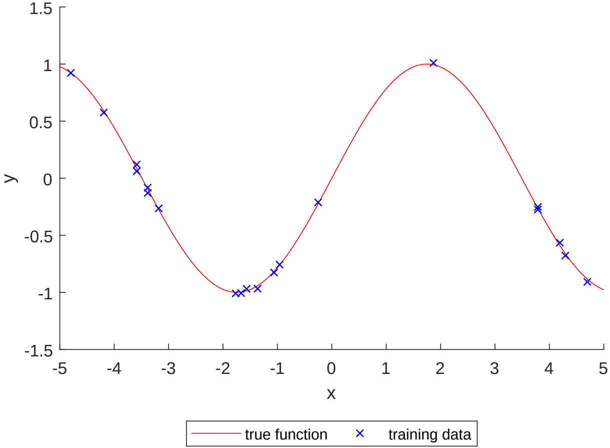
$$p([\boldsymbol{y}, \boldsymbol{f}^*] | \boldsymbol{X}, \boldsymbol{X}^*, \boldsymbol{y}, \boldsymbol{\theta}_K, \sigma^2) \propto \mathcal{N}\left([\boldsymbol{y}, \boldsymbol{f}^*] \middle| \boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma^2 \boldsymbol{I} & \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}^*} \\ \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}^*} \end{bmatrix}\right)$$

- Gaussian predictive distribution for $\boldsymbol{f}^*$
  $p(\boldsymbol{f}^* | \boldsymbol{X}, \boldsymbol{X}^*, \boldsymbol{y}, \boldsymbol{\theta}_K, \sigma^2) = \mathcal{N}(\boldsymbol{f}^* | \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*)$ with

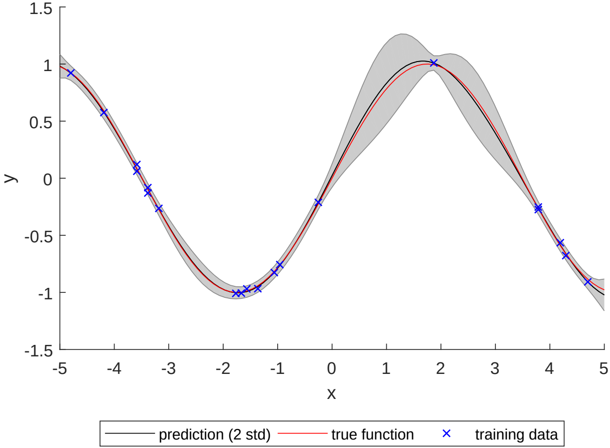
$$\boldsymbol{\mu}^* = \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}} \left(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y}$$
$$\boldsymbol{\Sigma}^* = \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}^*} + \sigma^2 \boldsymbol{I} - \boldsymbol{K}_{\boldsymbol{X}^*,\boldsymbol{X}} \left(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}^*}$$

- Matrix inverse (training) is expensive

Example

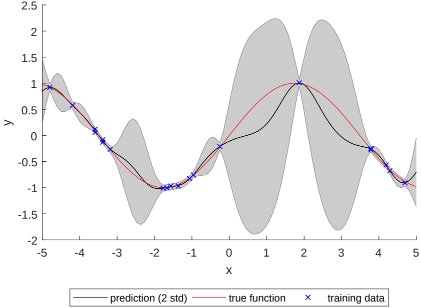


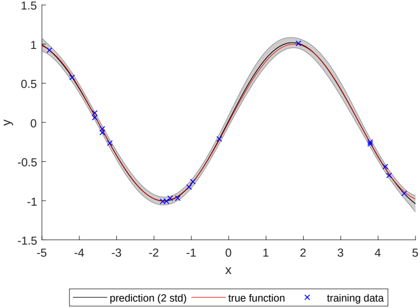
Example


1.5
1
0.5
0
-0.5
-1
-1.5
y
-5
-4
-3
-2
-1
0
1
2
3
4
5
x
prediction (2 std)
true function
training data

Kernel Parameters

horizontal lengthscale (too narrow, too wide)





$\rightarrow$ optimize kernel parameters as part of training

Kernels

- Squared exponential

$$k_{\text{SE}}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}(x_{d,i} - x_{d,j})^2\, l_d^{-2}\right)$$

Kernels

- Squared exponential

$$k_{\mathrm{SE}}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}\left(x_{d,i} - x_{d,j}\right)^2 l_d^{-2}\right)$$

- Linear

$$k_{\mathrm{lin}}(x_i, x_j) = \sigma_o^2 + x_i x_j$$

Kernels

- Squared exponential

$$k_{\mathrm{SE}}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}(x_{d,i} - x_{d,j})^2\, l_d^{-2}\right)$$

- Linear

$$k_{\mathrm{lin}}(x_i, x_j) = \sigma_o^2 + x_i x_j$$

- Brownian motion (Wiener process)

$$k_{\mathrm{Brownian}}(x_i, x_j) = \min\left(x_i, x_j\right)$$

Kernels

- Squared exponential

$$k_{\mathrm{SE}}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}(x_{d,i} - x_{d,j})^2\, l_d^{-2}\right)$$

- Linear

$$k_{\mathrm{lin}}(x_i, x_j) = \sigma_o^2 + x_i x_j$$

- Brownian motion (Wiener process)

$$k_{\mathrm{Brownian}}(x_i, x_j) = \min(x_i, x_j)$$

- Periodic

$$k_{\mathrm{periodic}}(x_i, x_j) = \exp\left(-\frac{2\sin^2\left(\frac{x_i - x_j}{2}\right)}{\lambda^2}\right)$$

Kernels

- Squared exponential

$$k_{\mathrm{SE}}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}(x_{d,i} - x_{d,j})^2\, l_d^{-2}\right)$$

- Linear

$$k_{\mathrm{lin}}(x_i, x_j) = \sigma_o^2 + x_i x_j$$

- Brownian motion (Wiener process)

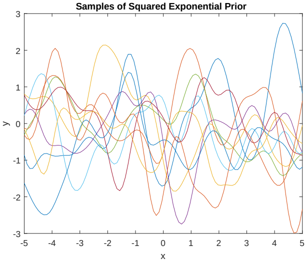
$$k_{\mathrm{Brownian}}(x_i, x_j) = \min(x_i, x_j)$$

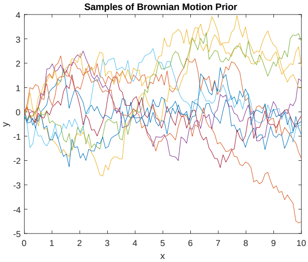
- Periodic

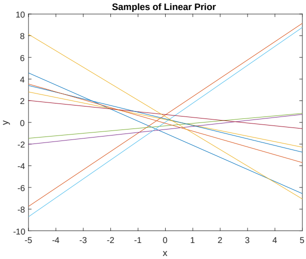
$$k_{\mathrm{periodic}}(x_i, x_j) = \exp\left(-\frac{2\sin^2\left(\frac{x_i - x_j}{2}\right)}{\lambda^2}\right)$$

- Many more
(need to correspond to an inner product $k(x_i, x_j) = \boldsymbol{\phi}(x_i)^{\mathsf{T}}\boldsymbol{\phi}(x_j)$)

Examples of priors







Constructing Kernels

- Sum

$$k_{\text{sum}}(x_i, x_j) = k_1(x_i, x_j) + k_2(x_i, x_j)$$

addition of independent processes

Constructing Kernels

- Sum

$$k_{\text{sum}}(x_i, x_j) = k_1(x_i, x_j) + k_2(x_i, x_j)$$

  addition of independent processes

- Product

$$k_{\text{prod}}(x_i, x_j) = k_1(x_i, x_j)k_2(x_i, x_j)$$

  product of independent processes

Constructing Kernels

- Sum

$$k_{\text{sum}}(x_i, x_j) = k_1(x_i, x_j) + k_2(x_i, x_j)$$

  addition of independent processes

- Product

$$k_{\text{prod}}(x_i, x_j) = k_1(x_i, x_j)k_2(x_i, x_j)$$

  product of independent processes

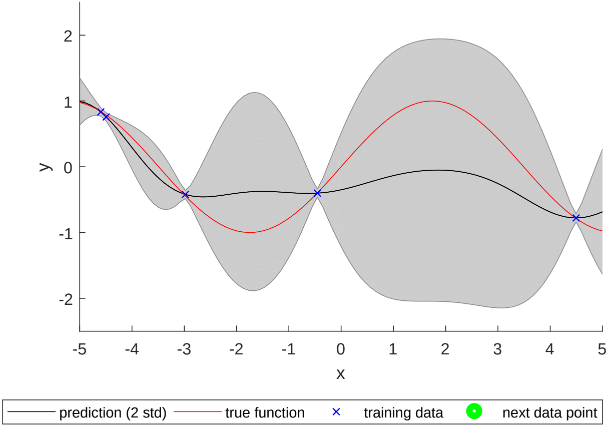
- Convolution

$$k_{\text{conv}}(x_i, x_j) = \int h(x_i, z_i)k(z_i, z_j)h(x_j, z_j)\, dz_i\, dz_j$$

  blurring with kernel h

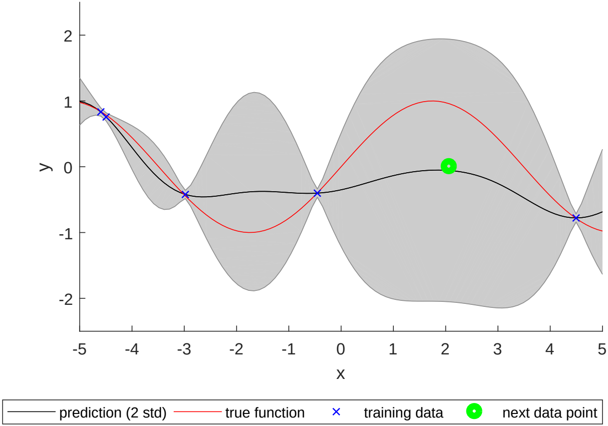
Active Learning

select next training point (here the one with highest uncertainty)



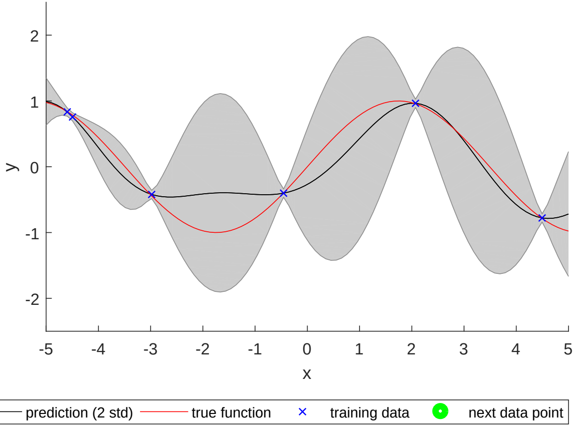
Active Learning

select next training point (here the one with highest uncertainty)



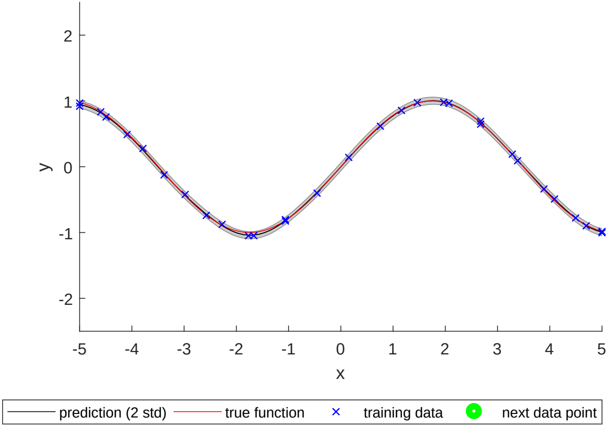
Active Learning

select next training point (here the one with highest uncertainty)

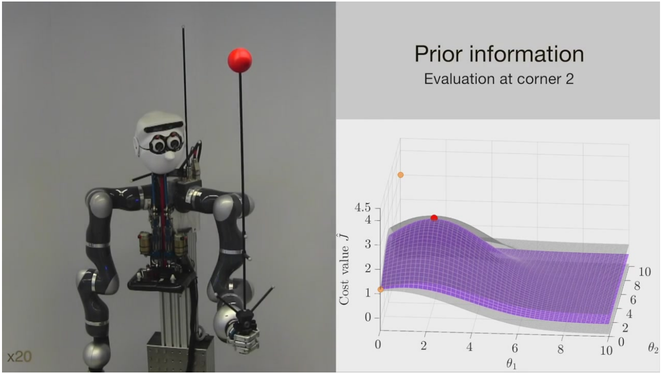


Active Learning

select next training point (here the one with highest uncertainty)



Bayesian Optimization[1]



https://youtu.be/d0ux-bXFCU4

[1] A. Marco, P. Hennig, J. Bohg, S. Schaal, and S. Trimpe (May 2016). "Automatic LQR Tuning Based on Gaussian Process Global Optimization". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*

Summary

- Easy to use
  - model with infinite number of parameters

Summary

- Easy to use
  - model with infinite number of parameters
- State-of-the-art performance

Summary

- Easy to use
  - model with infinite number of parameters
- State-of-the-art performance
- Many standard regression methods are special cases of GPs
  - Radial basis functions
  - Splines
  - Linear (ridge) regression
  - Feed-forward neural networks with one hidden layer

Summary

- Easy to use
  - model with infinite number of parameters
- State-of-the-art performance
- Many standard regression methods are special cases of GPs
  - Radial basis functions
  - Splines
  - Linear (ridge) regression
  - Feed-forward neural networks with one hidden layer
- Problems
  - Ill-conditioned
  - N^3 complexity is bad news for $N > 1000 \rightarrow$ approximate/sparse methods