Kernel Methods for Regression

I. Santamaría, S. Van Vaerenbergh

GTAS, Universidad de Cantabria

3 de febrero de 2022

Maśter Universitario Oficial **Data Science**

UC UNIVERSIDAD DE CANTABRIA

UIMP Universidad Internacional Menéndez Pelayo

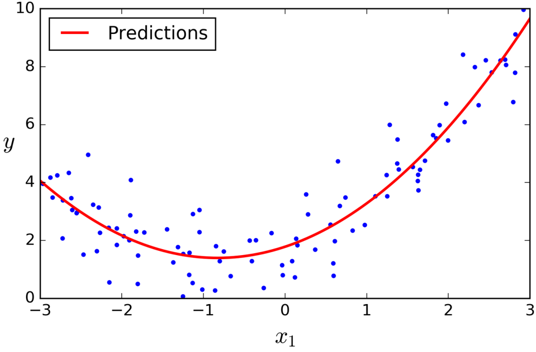
con el apoyo del

CSIC Consejo Superior de Investigaciones Científicas

Contents

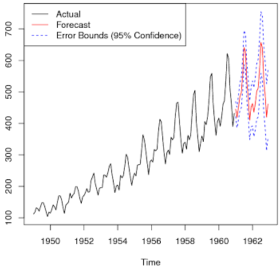
Problem formulation

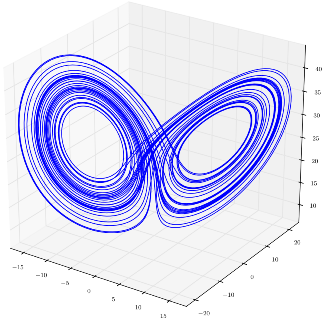
- Input: $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, $\mathbf{x}_i \in \mathcal{R}^d$
- Output: $(y_1, \ldots, y_n)$, $y_i \in \mathcal{R}$
- **Problem**: find $f(\cdot) : \mathcal{R}^d \to \mathcal{R}$ to fit (or predict) y from $\mathbf{x}$



Applications

Typical applications are **time-series prediction** or **nonlinear modeling**

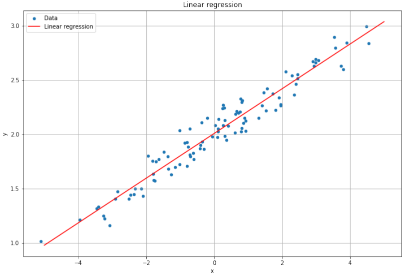




Before considering kernel methods, let us briefly review the linear case

Linear regression

- To find the hyperplane $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$ that best fits the observations



- If we redefine $\mathbf{x} = \begin{bmatrix}\mathbf{x}\\1\end{bmatrix}$ and $\mathbf{w} = \begin{bmatrix}\mathbf{w}\\b\end{bmatrix}$, the linear function becomes

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = \mathbf{x}^T\mathbf{w}$$

- Training data set $\{(\mathbf{x}_i, y_i),\ i = 1, \ldots, n\}$ with $\mathbf{x}_i \in \mathcal{R}^d$ and $y_i \in \mathcal{R}$

$$y_i = \mathbf{x}_i^T \mathbf{w} + e_i, \qquad i = 1, \ldots, n$$

- Overdetermined system ($n > d$) of linear equations

$$\underbrace{\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} x_{11} & x_{12} & & x_{1d} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nd} \end{bmatrix}}_{\mathbf{X}^T} \underbrace{\begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix}}_{\mathbf{w}} + \underbrace{\begin{bmatrix} e_1 \\ \vdots \\ e_n \end{bmatrix}}_{\mathbf{e}}$$

$$\mathbf{y} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix} \mathbf{w} + \mathbf{e} = \mathbf{X}^T \mathbf{w} + \mathbf{e}$$

Regularized regression

- Solution to overfitting or ill-conditioned problems

$$J(\mathbf{w}) = \sum_{i=1}^{n} \mathcal{L}(e_i) + \lambda ||\mathbf{w}||^2$$

- Error penalty or **loss function** $\mathcal{L}(e_i)$

$$\begin{aligned} L_2 - \text{norm}: & \quad \mathcal{L}(e_i) = e_i^2 \\ L_1 - \text{norm}: & \quad \mathcal{L}(e_i) = |e_i| \\ \epsilon - \text{insensitive}: & \quad \mathcal{L}(e_i) = \max(0, |e_i| - \epsilon) \end{aligned}$$

- The complexity of the solution is penalized through $||\mathbf{w}||^2$
- λ is the regularization parameter
- Note: the resulting optimization problem may be nonlinear, but we find a linear regressor

Ridge regression

- Linear regression problem with L_2-norm loss function

$$J(\mathbf{w}) = \sum_{i=1}^{n} \left(y_i - \mathbf{w}^T \mathbf{x}_i \right)^2 + \lambda ||\mathbf{w}||^2$$

- The solution is

$$\mathbf{w} = \left(\mathbf{X}\mathbf{X}^T + \lambda \mathbf{I}_d \right)^{-1} \mathbf{X}\mathbf{y}$$

- If $\lambda = 0$ we recover the standard **Least Squares** solution

$$\mathbf{w}_{LS} = \left(\mathbf{X}\mathbf{X}^T \right)^{-1} \mathbf{X}\mathbf{y} = \mathbf{X}^\sharp \mathbf{y}$$

$\mathbf{X}^\sharp = (\mathbf{X}\mathbf{X}^T)^{-1} \mathbf{X}$ is the pseudoinverse or Moore-Penrose inverse of $\mathbf{X}^T$

- The Ridge Regression solution can be expressed as a linear combination of the input patterns

$$\mathbf{w} = \left(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_d\right)^{-1}\mathbf{X}\mathbf{y} = \mathbf{X}\underbrace{\left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_n\right)^{-1}\mathbf{y}}_{\boldsymbol{\alpha}}$$

$$= \mathbf{X}\boldsymbol{\alpha} = \sum_{i=1}^{n}\alpha_i\mathbf{x}_i,$$

where

$$\mathbf{X}^T\mathbf{X} = \begin{bmatrix} \langle\mathbf{x}_1,\mathbf{x}_1\rangle & \cdots & \langle\mathbf{x}_1,\mathbf{x}_n\rangle \\ \vdots & \ddots & \vdots \\ \langle\mathbf{x}_n,\mathbf{x}_1\rangle & \cdots & \langle\mathbf{x}_n,\mathbf{x}_n\rangle \end{bmatrix}$$ **is a kernel matrix !!**

- And the output for a new test pattern $\mathbf{x}$ is

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = \sum_{i=1}^{n}\alpha_i\langle\mathbf{x}_i,\mathbf{x}\rangle$$ **is a kernel expansion !!**

Kernel Ridge Regression (KRR)

- We apply the **kernel trick** and change the linear kernel $\langle \mathbf{x}_i, \mathbf{x}_j \rangle = \mathbf{x}_i^T \mathbf{x}_j$ by a nonlinear kernel (e.g, Gaussian)

$$\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle = k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \right)$$

- The solution for the coefficients is

$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}$$

- The output for a test vector $\mathbf{x}$ is

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x})$$

KRR Summary

Given $\{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$, the kernel function $k(\cdot, \mathbf{x})$, and the regularization parameter $\lambda > 0$:

1. Build the $n \times n$ kernel matrix $\mathbf{K}$ with elements $k(\mathbf{x}_i, \mathbf{x}_j)$
2. Obtain the expansion coefficients

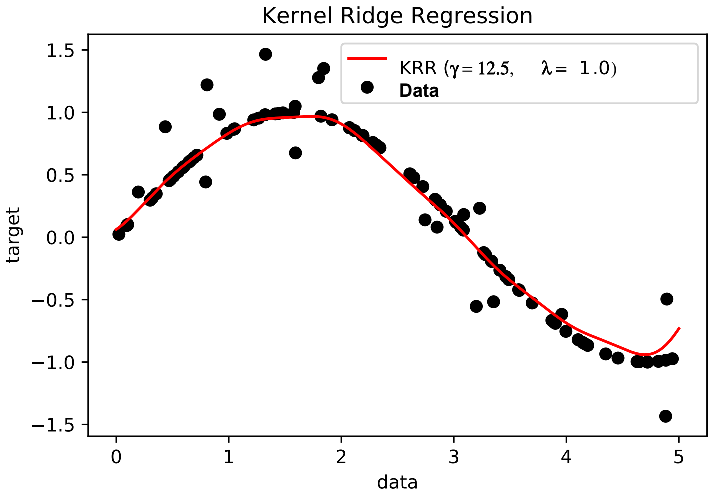
$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}$$

3. The output for a new input data $\mathbf{x}$ is (out-of-sample regression)

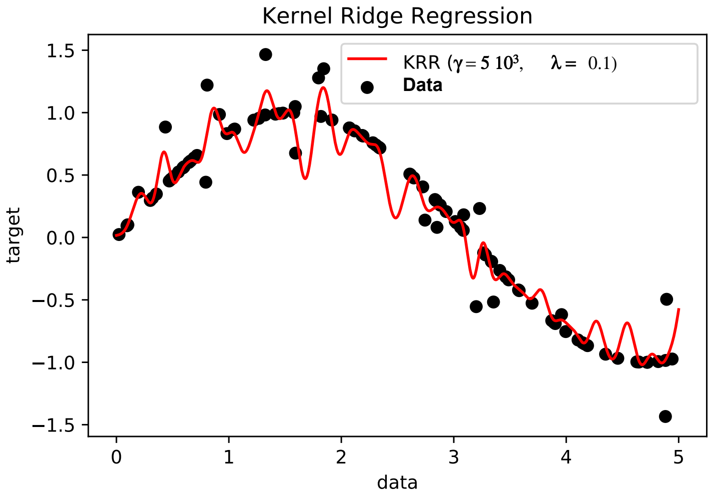
$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x})$$

- The matrix $(\mathbf{K} + \lambda \mathbf{I})$ is always invertible
- With a Gaussian kernel and $\lambda = 0$ we are interpolating the data
- The computational cost to calculate $(\mathbf{K} + \lambda \mathbf{I})^{-1}$ grows as n^3
- KRR: parameter fitting
  - λ: Regularization parameter
    - $\lambda \downarrow$ less regularization $\rightarrow$ non-smooth models, **overfitting** risk
    - $\lambda \uparrow$ more regularization $\rightarrow$ smooth models
  - γ: Gaussian kernel
    - $\gamma \downarrow$ broad (overlapped) Gaussians $\rightarrow$ smooth models
    - $\gamma \uparrow$ narrow Gaussians $\rightarrow$ non-smooth models

Example: $\gamma = 12,5; \lambda = 1$



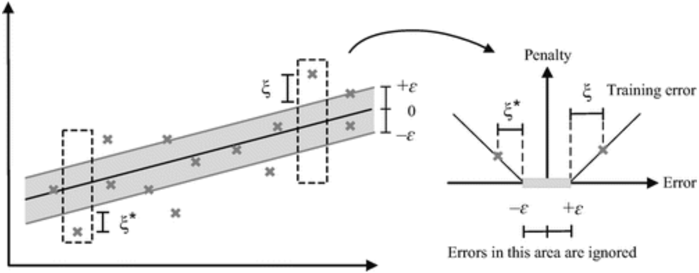
Example: $\gamma = 5 \cdot 10^3$; $\lambda = 0{,}1$



Support Vector Regression (SVR)

- Instead of the L_2-norm, the SVR uses the ϵ-insensitive loss function

$$\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\left|y_i - \mathbf{w}^T\mathbf{x}_i\right|_\epsilon$$



- L_1-norm penalty for errors larger than ϵ
- Avoids overfitting by not considering small errors

- The coefficients of the expansion, α, are the solution of a Quadratic Programming (QP) problem
- The output for a test vector is

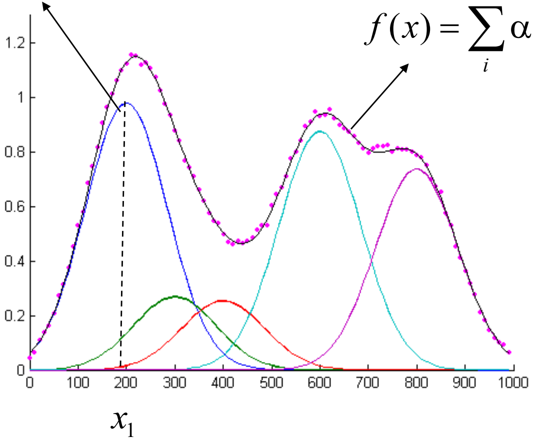
$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x})$$

- $\alpha_i \neq 0$ only for points outside the ϵ-tube $\Longrightarrow$ sparse expansion

SVR with Gaussian kernel

$$\alpha_1 k(x,x_1) = \alpha_1 e^{-\gamma\|x-x_1\|^2}$$

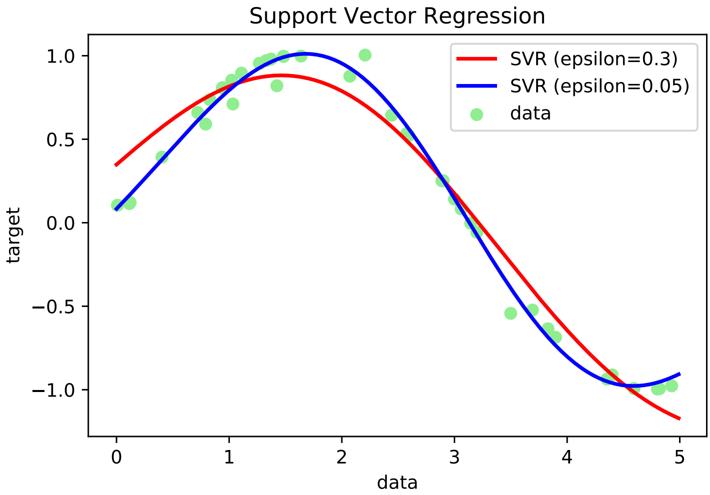
$$f(x) = \sum_i \alpha_i k(x,x_i)$$



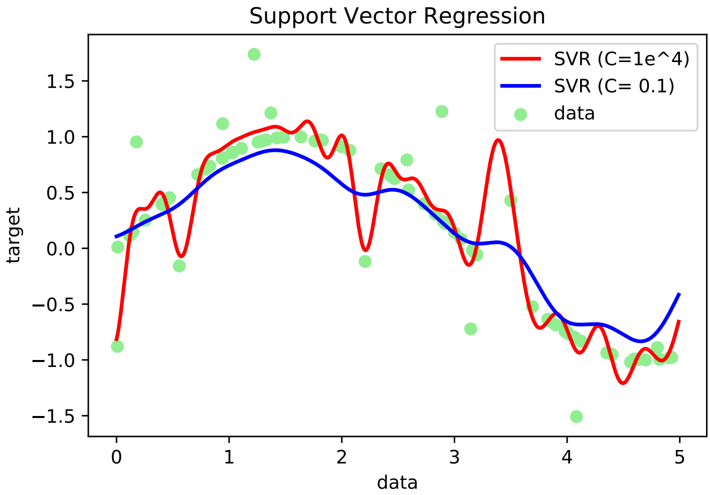
SVR: Parameter fitting

- C: Regularization parameter
  - $C \downarrow$ less penalty for errors $\rightarrow$ smooth models
  - $C \uparrow$ more penalty for errors $\rightarrow$ non-smooth models, **overfitting** risk
- ϵ: Loss function parameter
  - $\epsilon \downarrow$ small errors are penalized $\rightarrow$ non-smooth models, **overfitting** risk
  - $\epsilon \uparrow$ only large errors are penalized $\rightarrow$ smooth models
- γ: Gaussian kernel
  - $\gamma \downarrow$ broad (overlapped) Gaussians $\rightarrow$ smooth models
  - $\gamma \uparrow$ narrow Gaussians $\rightarrow$ non-smooth models

Example: $C = 10^3$, γ =0.1



Example: $\epsilon = 0.1$, $\gamma = 10$



KRR vs. SVR

- SVR and KRR minimize regularized functionals

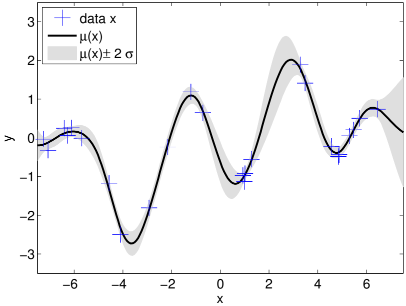
$$\text{KRR}: \quad \min_{\mathbf{w}} \lambda ||\mathbf{w}||^2 + \sum_{i=1}^{n} \left(y_i - \mathbf{w}^T \mathbf{x}_i \right)^2$$

$$\text{SVR}: \quad \min_{\mathbf{w}} \frac{1}{2} ||\mathbf{w}||^2 + C \sum_{i=1}^{n} \left| y_i - \mathbf{w}^T \mathbf{x}_i \right|_{\epsilon}$$

- To obtain the output for a test vector **x**, SVR and KRR have identical functional form: $f(\mathbf{x}) = \sum_{j=1}^{n} \alpha_j k(\mathbf{x}_j, \mathbf{x})$, but the expansion coefficients are different
  - SVR: QP problem, sparse solution, many $\alpha_i = 0$
  - KRR: Linear problem, non-sparse solution, $\alpha_i \neq 0, \forall i$

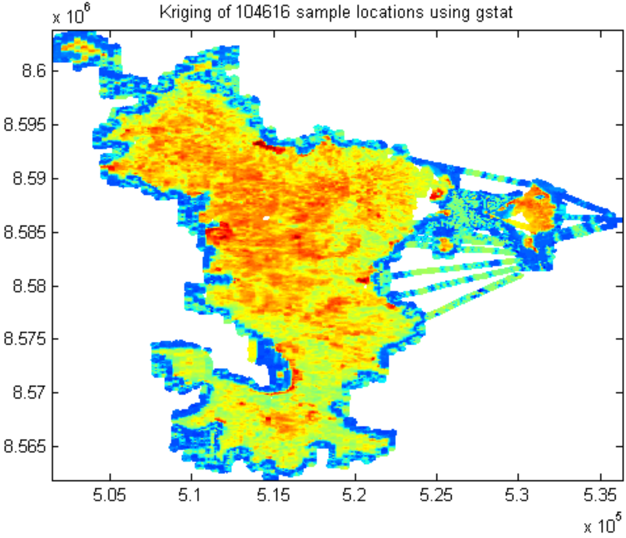
Introduction

- A limitation of SVR and KRR is that they do not provide any information about the uncertainty or confidence interval of the predictions



- **Gaussian Processes** or GPs are state-of-the-art **Bayesian** methods for regression that overcome this limitation of kernel methods

GPs are known in geostatistics as **kriging**



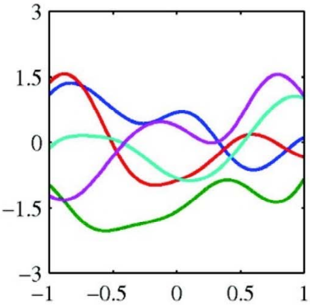
Bayes Theorem

- **Prior distribution**: What *a priori* knowledge do we have about the function we want to estimate $f(\mathbf{x})$?
- **Likelihood**: What information about $f(\mathbf{x})$ do the observations provide? $\rightarrow$ Noise distribution
- **Bayes Theorem**: How to combine the prior with the likelihood to yield a **posterior distribution** for $f(\mathbf{x})$

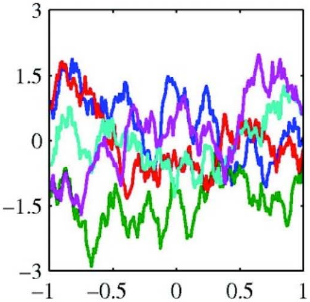
$$\text{Posterior} \propto \text{Prior} \times \text{Likelihood}$$

- GPs: the prior distribution and the noise distribution (likelihood) are both Gaussian $\Longrightarrow$ the posterior is also Gaussian

Prior: a zero-mean Gaussian with covariance matrix **K** (kernel matrix): $f(\mathbf{x}) \sim \mathcal{GP}(0, \mathbf{K})$

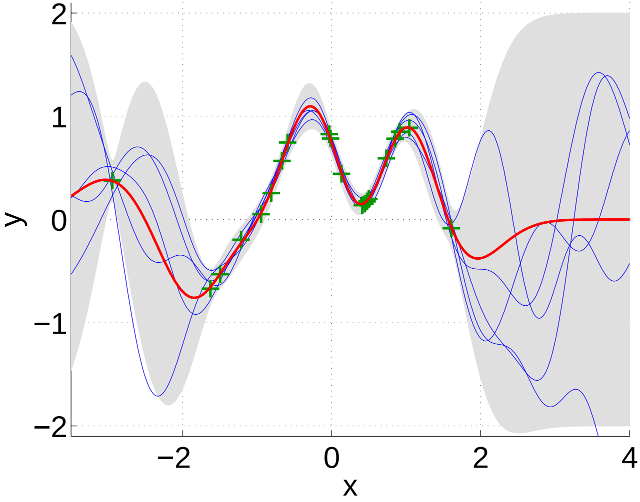


$$k(x_1, x_2) = \exp\left(\frac{-(x_1 - x_2)^2}{2\sigma^2}\right)$$



$$k(x_1, x_2) = \exp\left(\frac{-|x_1 - x_2|}{\sigma^2}\right)$$

Likelihood: a zero-mean Gaussian with variance σ_e^2:
$e_i \sim \mathcal{N}(0, \sigma_e^2)$



Posterior: Given a new test point $\mathbf{x}$, the posterior distribution for the latent function (GP output) is a Gaussian: $\mathcal{N}(f(\mathbf{x}), \sigma^2)$

- Mean

$$f(\mathbf{x}) = \mathbf{k}^T[\mathbf{K} + \sigma_e^2\mathbf{I}]^{-1}\mathbf{y},$$

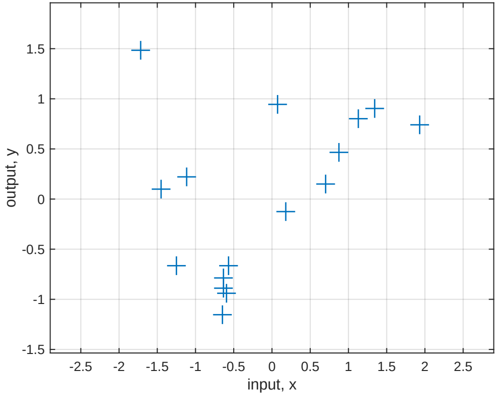
  same expression as a KRR with regularization parameter $\lambda = \sigma_e^2$!!

- Variance

$$\sigma^2 = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}^T[\mathbf{K} + \sigma_e^2\mathbf{I}]^{-1}\mathbf{k}$$

  where $\mathbf{k} = \left[k(\mathbf{x}_1, \mathbf{x}), \ldots, k(\mathbf{x}_n, \mathbf{x})\right]^T$, and $\mathbf{K}$ is the kernel matrix with elements $k(\mathbf{x}_i, \mathbf{x}_j)$

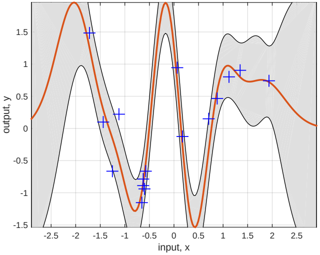
Example



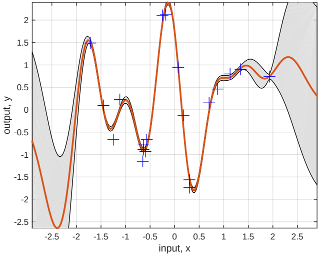
- GP with Gaussian kernel
- Hyperparameters: kernel size σ^2 and noise variance σ_e^2

Fixed kernel size ($\sigma^2 = 0.2$)

$\sigma_e^2 = 0.2$

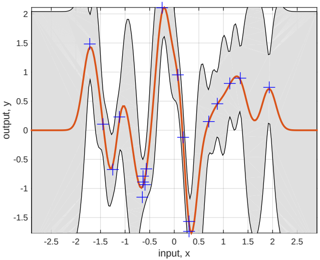


$\sigma_e^2 = 0.02$

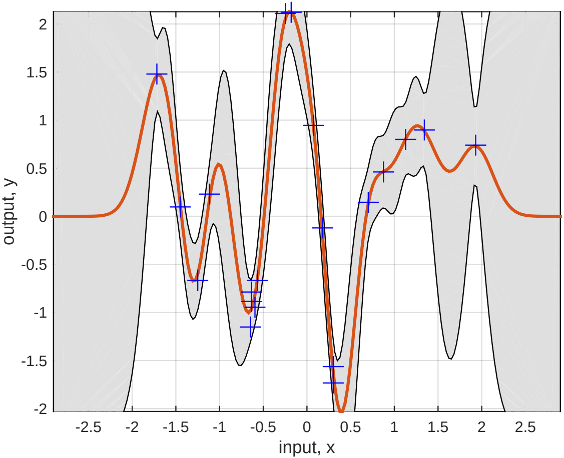


Fixed noise variance ($\sigma_e^2 = 0.2$)

$\sigma^2 = 0.02$ $\sigma^2 = 1$



Typically we use the Maximum Likelihood estimates of the hyperparameters: $\hat{\sigma}^2 \approx 0.04$ y $\sigma_e^2 \approx 0.135$



Software

Matlab:

- GPML software package: http://www.gaussianprocess.org/gpml/code/matlab/doc/

Python:

- GPy: http://sheffieldml.github.io/GPy/
- GPs via TensorFlow: https://github.com/GPflow/GPflow

scikit-learn includes a simple version

Conclusions

- KRR
  - Regularized LS in the feature space
  - Loss function: L_2-norm $\rightarrow$ Non-sparse solution
  - Inversion of the regularized kernel matrix
  - Hyperparameter estimation: Cross-validation
- SVR
  - Based on the SRM principle
  - Loss function: ϵ-insensitive $\rightarrow$ Sparse solution
  - QP problem
  - Hyperparameter estimation: Cross-validation
- GPs
  - Bayesian approximation
  - Provides confidence intervals
  - Mean value of the posterior = KRR
  - Hyperparameter estimation: Maximum Likelihood