Chapter 2 & 3

Weber Jakob

November 11, 2020

Contents

Chapter 1

Fundamentals

This chapter summarizes the fundamentals of regression. Excellent overviews can be found in the textbooks **wood2017generalized**, **fahrmeir2007regression**, **friedman2001elements**. The shown fundamentals are strongly aligned with the presentation given in **fahrmeir2007regression**. Section 1.1 gives an overview of the model assumptions used throughout this work, Furthermore, Section 1.2 outlines methods to evaluate and compare different models again each other in terms of complexity and accuracy. Section 1.3 is devoted to the spline definitions, Finally, in Section 1.3 the so-called Structured Additive Regression is shown. It is basically the as

1.1 Linear Models

Given the data set $D = \{(x_1^{(i)}, \ldots, x_q^{(i)}; y^{(i)}),\ i = 1, \ldots, n\}$, we aim to model the relation between the inputs $x_1, \ldots, x_q$ and the output y with a function $f(x_1, \ldots, x_q)$. Since this relationship is not exact, there will be a random part ϵ. It is typically assumed that this part is additive and thus

$$y = f(x_1, \ldots, x_q) + \epsilon. \tag{1.1}$$

We would like to estimate the unknown function f. For this, some assumptions on the model structure are made:

1. *The unknown function f is a linear combination of the inputs*

   The function $f(x_1, \ldots, x_q)$ is modeled as a linear combination of inputs, i.e. in the form

   $$f(x_1, \ldots, x_q) = \beta_0 + \beta_1 x_1 + \cdots + \beta_q x_q, \tag{1.2}$$

   with unknown parameters $\beta_0, \ldots, \beta_q$. Note that the model (1.1) is linear in its parameters as well as in its inputs. The parameter β_0 is called intercept or bias in the machine learning community, see **bishop2006patternRecognition**. For centered data, i.e. the expected value $\mathrm{E}(x^{(i)}) = 0$, the intercept is equal to zero and can be neglected. We introduce the input vectors $\mathbf{x}^{\mathrm{T}} =$

$[1, x_1, \ldots, x_q] \in \mathbb{R}^{q+1}$ and the parameter vector $\boldsymbol{\beta}^{\mathrm{T}} = [\beta_0, \beta_1, \ldots, \beta_q] \in \mathbb{R}^{q+1}$ to obtain

$$y(x_1, \ldots, x_q) = y(\mathbf{x}^{\mathrm{T}}) = \mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}. \tag{1.3}$$

2. *Additive errors*

   An additional assumption of linear models is additivity of errors, which means that

   $$y = \mathbf{x}^{\mathrm{T}}\boldsymbol{\beta} + \epsilon. \tag{1.4}$$

   This is reasonable for many practical applications, even though it appears quite restrictive.

To estimate the unknown parameters or coefficients $\boldsymbol{\beta}$, we define the output vector $\mathbf{y}^{\mathrm{T}} = [y^{(1)}, \ldots, y^{(n)}] \in \mathbb{R}^n$ and data error vector $\boldsymbol{\epsilon}^{\mathrm{T}} = [\epsilon^{(i)}, \ldots, \epsilon^{(n)}] \in \mathbb{R}^n$ as well as the design matrix

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^{(1)} & \ldots & x_q^{(1)} \\ \vdots & & & \vdots \\ 1 & x_1^{(n)} & \ldots & x_q^{(n)} \end{bmatrix} \in \mathbb{R}^{n \times q+1} \tag{1.5}$$

and generate n equations like (1.4), which can be combined to

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}. \tag{1.6}$$

We assume that the design matrix $\mathbf{X}$ has full column rank, i.e. $\mathrm{rank}(\mathbf{X}) = q+1 = p$, implying linear independence of the columns of $\mathbf{X}$, which is necessary to obtain a unique estimator for the regression coefficients $\boldsymbol{\beta}$, see **fahrmeir2007regression**.

Another necessary requirement is that the number of data points n is larger or equal to the number of regression parameters p, which is equivalent to the statement that the linear system in (1.6) is not underdetermined.

In addition to the assumptions on the unknown function f, the necessary assumptions on the error term ϵ_i are the following **fahrmeir2007regression**:

1. *Expectation of the error*
   The errors have a mean value of zero, i.e. $\mathrm{E}(\epsilon^{(i)}) = 0$

2. *Variances and correlation structure of the errors*
   We assume constant error variance with $\mathrm{Var}(\epsilon^{(i)}) = \sigma^2$. This property is called homoscedasticity. Additionally, we assume that the errors are uncorrelated, which means $\mathrm{Cov}(\epsilon^{(i)}, \epsilon^{(j)}) = 0$ for $i \neq j$. The combination of these assumptions lead to the covariance matrix $\mathrm{Cov}(\boldsymbol{\epsilon}) = \mathrm{E}(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^{\mathrm{T}}) = \sigma^2\mathbf{I}$.

3. *Assumptions on the input and design matrix* We have to distinguish two cases where the inputs are deterministic or stochastic, In most cases, the input and the output are stochastic and hence all model assumptions are conditioned on the design matrix (1.5). That means that the

input (x gehört noch transponiert, liefert aber fehler in latex) $\mathbf{x}^{(i)}T = [1, x_1^{(i)}, \ldots, x_q^{(i)}]$ and the errors ϵ_i are not stochastically independent. For notational simplicity, we usually suppress the dependence on the design matrix.

4. *Gaussian errors*
   The errors follow at least approximately a normal distribution. Together with assumptions 1 and 2, we obtain that $\epsilon^{(i)} = \mathcal{N}(0, \sigma^2)$.

Summarizing, we have the following model assumptions:

$$\mathrm{E}(\mathbf{y}) = \mathbf{X}\boldsymbol{\beta} \tag{1.7}$$

$$\mathrm{Cov}(\mathbf{y}) = \sigma^2\mathbf{I}, \tag{1.8}$$

yielding

$$\mathbf{y} \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}). \tag{1.9}$$

A linear model with multiple input variables can therefore be interpreted as a multi-variate normal distribution with its mean vector given by $\boldsymbol{\mu} = \mathbf{X}\boldsymbol{\beta}$ and its covariance matrix given by $\sigma^2\mathbf{I}$, i.e. to specify the linear model given in (1.9), we need to estimate the regression coefficients $\boldsymbol{\beta}$ and the variance σ^2.

1.1.1 Parameter Estimation

The linear model given in (1.9) features the unknown parameters $\boldsymbol{\beta}$ and σ, which need to be estimated using the given data. In the following, the estimators $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}$ are introduced, and their statistical properties are derived. The two methods to estimate the regression parameters in the context of linear models are the method of Least Squares (LS) and the method of Maximum Likelihood (ML).

1.1.1.1 The Method of Least Squares

The unknown regression parameters $\boldsymbol{\beta} \in \mathbb{R}^p$ are estimated by minimizing the sum of squared error

$$\mathrm{LS}(\mathbf{y}, \boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 = \sum_{i=1}^{n} \epsilon_i^2 = \boldsymbol{\epsilon}^{\mathrm{T}}\boldsymbol{\epsilon}, \tag{1.10}$$

with respect to $\boldsymbol{\beta}$ **friedman2001elements**. Rewriting (1.10) leads to the least squares criterion

$$\begin{aligned} \mathrm{LS}(\mathbf{y}, \boldsymbol{\beta}) &= (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \\ &= \mathbf{y}^{\mathrm{T}}\mathbf{y} - 2\mathbf{y}^{\mathrm{T}}\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X}\boldsymbol{\beta}. \end{aligned} \tag{1.11}$$

The first-order necessary condition for optimality, cf. **luenberger1984linear**, reads as

$$\frac{\partial \mathrm{LS}(\mathbf{y}, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2\mathbf{X}^{\mathrm{T}}\mathbf{y} + 2\mathbf{X}^{\mathrm{T}}\mathbf{X}\boldsymbol{\beta} = 0. \tag{1.12}$$

The second-order condition requires the Hessian to be positive-definite, i.e.

$$\frac{\partial^2 \mathrm{LS}(\mathbf{y}, \boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^{\mathrm{T}}} = 2\mathbf{X}^{\mathrm{T}}\mathbf{X} > 0. \tag{1.13}$$

Since the design matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ is assumed to have full rank, the matrix $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ is positive definite. The least squares estimate $\hat{\boldsymbol{\beta}}_{LS}$ is hence obtained, see (1.12), by solving the so-called *normal equations*

$$\mathbf{X}^{\mathrm{T}}\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^{\mathrm{T}}\mathbf{y}. \tag{1.14}$$

Since $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ is positive definite, the *normal equations* (1.14) feature a unique solution given by the least squares estimator

$$\hat{\boldsymbol{\beta}}_{LS} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}. \tag{1.15}$$

1.1.1.2 Maximum Likelihood Estimation

Under the normality assumption, the likelihood is defined as **wood2017generalized**

$$\mathcal{L}(\boldsymbol{\beta}, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\Big(-\frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\Big). \tag{1.16}$$

The log-likelihood is then given by

$$l(\boldsymbol{\beta}, \sigma^2) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}). \tag{1.17}$$

Thus, maximizing the log-likelihood $l(\boldsymbol{\beta}, \sigma^2)$ with respect to $\boldsymbol{\beta}$ is equivalent to minimizing the least squares criterion given in (1.10). The maximum likelihood estimator $\hat{\boldsymbol{\beta}}_{ML}$ is therefore equivalent to the least squares estimator $\hat{\boldsymbol{\beta}}_{LS}$ in (1.15).

1.1.2 Estimation of the Variance σ^2

The estimation of the variance σ^2 is necessary for the construction of confidence intervals of the regression parameters and for the construction of prediction intervals. It is further used in all kinds of statistical tests as well as in model selection approaches and model assessment criteria **blobel2013statistische**.

1.1.2.1 Maximum Likelihood Estimation

The first-order necessary condition for optimality results in this case in

$$\frac{\partial l(\boldsymbol{\beta}, \sigma^2)}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = 0. \tag{1.18}$$

Substituting the maximum likelihood estimator $\hat{\boldsymbol{\beta}}_{LS}$, given in (1.15), for $\boldsymbol{\beta}$ results in the maximum likelihood estimator

$$\hat{\sigma}^2_{ML} = \frac{(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}_{LS})^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}_{LS})}{n} = \frac{\hat{\boldsymbol{\epsilon}}^{\mathrm{T}}\hat{\boldsymbol{\epsilon}}}{n}. \tag{1.19}$$

This estimator for σ^2 is rarely used since it is biased, i.e. $\mathrm{E}(\sigma^2_{ML}) \neq \sigma^2$, see **fahrmeir2007regression**.

1.1.2.2 Restricted Maximum Likelihood Estimation

The mean value of the sum of squared residuals is $\mathrm{E}(\hat{\boldsymbol{\epsilon}}^{\mathrm{T}}\hat{\boldsymbol{\epsilon}}) = (n-p)\sigma^2$. Hence, a less biased estimator $\hat{\sigma}^2$ for σ^2 is given by

$$\hat{\sigma}^2_{REML} = \frac{1}{n-q}\hat{\boldsymbol{\epsilon}}^{\mathrm{T}}\hat{\boldsymbol{\epsilon}}. \tag{1.20}$$

The restricted maximum likelihood estimator for the variance is in general less biased **fahrmeir2007regression**. Therefore, it is the commonly used estimator for the variance σ^2.

1.1.3 The Hat Matrix

Using the least squares estimator (1.15), we can estimate the mean of $\mathbf{y}$ by

$$\widehat{\mathrm{E}(\mathbf{y})} = \hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}}_{LS} \tag{1.21}$$

Substituting (1.15) results in

$$\hat{\mathbf{y}} = \mathbf{X}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y} = \mathbf{H}\mathbf{y}, \tag{1.22}$$

with the matrix $\mathbf{H} \in \mathbb{R}^{n \times n}$, which is called *hat matrix*. Using the *hat matrix*, we can express the residuals $\hat{\epsilon}^{(i)} = y^{(i)} - \hat{y}^{(i)}$ in matrix notation as

$$\hat{\boldsymbol{\epsilon}} = \mathbf{y} - \hat{\mathbf{y}} = (\mathbf{I} - \mathbf{H})\mathbf{y}. \tag{1.23}$$

The *hat matrix* $\mathbf{H}$ has the following useful properties:

- $\mathbf{H}$ is symmetric.
- $\mathbf{H}$ is idempotent, i.e. $\mathbf{H}^2 = \mathbf{H}$.
- The rank of $\mathbf{H}$ is equal to its trace.

- $\frac{1}{n} \leq h_{ii} \leq 1$, if all data points are different, i.e. $x^{(i)} \neq x^{(j)}$ for $i \neq j$. Here, h_{ii} are the diagonal elements of $\mathbf{H}$.
- The matrix $(\mathbf{I}-\mathbf{H})$ is also idempotent and symmetric, with $\text{rank}(\mathbf{I}-\mathbf{H}) = n-p$.

The hat matrix is used in model selection techniques like cross-validation as well as in outlier detection and in diagnostic plots for linear models.

1.2 Model Selection

Linear models are widely exploited for regression problems on large data sets ($n \gg 0$), because the solution of the *normal equations* (1.14) can be computed efficiently even for large n. If these data sets also contain a large number of input variables ($q \gg 0$), the situation becomes more complicated since possible interaction effects or correlation of input variables may occur. This interaction terms limits the, otherwise perfect, interpretability of the linear model.

Therefore, we need techniques and criteria to select the *best possible model* out of the variety of different models for a given data set. Model assessment criteria, see Section 1.2.1, are used to compare different models while subset selection techniques, see Section 1.2.2, give an algorithmic approach to model generation. Further, we can influence the estimated coefficients $\boldsymbol{\beta}$ directly via regularization, see Section 1.2.3.

1.2.1 Model Assessment Criteria

One way of comparing various models, i.e models using different sets of inputs, is the use of model assessment criteria. Generally, model assessment criteria can be split in two components. The first one measures the goodness of fit, e.g. using the sum of squared errors, while the second measures the complexity of the model. Most model assessment criteria are based on the sum of expected squared prediction error (SPSE). Therefore, the derivation of the SPSE is given next.

1.2.1.1 Sum of Expected Squared Prediction Error

given independent observations $y_i,\ i = 1, \ldots, n$ and a subset of inputs $\{x_0 = 1, x_1, \ldots, x_q\}$, we want to measure the prediction quality. The specific models are defined by numbers $M \subset \{0, 1, \ldots, q\}$ of used inputs with corresponding design matrix $\mathbf{X}_M$. Moreover, $|M|$ is the caridnal number of M, i.e. the number of inputs included in the model. The least squares estimator for $\boldsymbol{\beta}$, cf. (1.15), is then given by

$$\hat{\boldsymbol{\beta}}_M = (\mathbf{X}_M^{\mathrm{T}}\mathbf{X}_M)^{-1}\mathbf{X}_M^{\mathrm{T}}\mathbf{y}.$$

The data $\mathbf{y}$ can be interpreted as random variable. We can then define an estimator $\hat{\mathbf{y}}_M$ for the vector $\boldsymbol{\mu}$ of expectations $\mu^{(i)} = \mathrm{E}(y^{(i)})$ by

$$\hat{\mathbf{y}}_M = \mathbf{X}_M\hat{\boldsymbol{\beta}}_M. \tag{1.24}$$

Moreover, it is easy to show that the following properties hold true:

(i) $\mathrm{E}(\hat{\mathbf{y}}_M) = \mathbf{X}_M(\mathbf{X}_M^{\mathrm{T}}\mathbf{X}_M)^{-1}\mathbf{X}_M^{\mathrm{T}}\mathrm{E}(\mathbf{y})$

(ii) $\mathrm{Cov}(\hat{\mathbf{y}}_M) = \sigma^2\mathbf{X}_M(\mathbf{X}_M^{\mathrm{T}}\mathbf{X}_M)^{-1}\mathbf{X}_M^{\mathrm{T}}$

(iii) $\sum_{i=1}^{n}\mathrm{Var}(\hat{y}_M^{(i)}) = \sigma^2\mathrm{tr}(\mathbf{X}_M(\mathbf{X}_M^{\mathrm{T}}\mathbf{X}_M)^{-1}\mathbf{X}_M^{\mathrm{T}}) = \sigma^2|M|$

(iv) Sum of Mean Squared Error

$$\begin{aligned}\mathrm{SMSE} &= \sum_{i=1}^{n}\mathrm{E}(\hat{y}_M^{(i)} - \mu_M^{(i)})^2 \\ &= \sum_{i=1}^{n}\mathrm{E}\Big(\big(\hat{y}_M^{(i)} - \mathrm{E}(\hat{y}_M^{(i)})\big) + \big(\mathrm{E}(\hat{y}_M^{(i)}) - \mu_M^{(i)}\big)\Big)^2 \\ &= |M|\sigma^2 + \sum_{i=1}^{n}\big(\mathrm{E}(\hat{y}_M^{(i)}) - \mu_M^{(i)}\big)^2.\end{aligned} \tag{1.25}$$

Note that the estimator (1.24) can be regarded as a prediction of future observations

$$y^{(n+i)} = \mu^{(i)} + \epsilon^{(n+i)} \tag{1.26}$$

for new input data $\{x_1^{(i)}, \ldots, x_q^{(i)}\}$. Thus, we can derive the SPSE as

$$\begin{aligned}\mathrm{SPSE} &= \sum_{i=1}^{n}\mathrm{E}(y^{(n+i)} - \hat{y}_M^{(i)})^2 \\ &= \sum_{i=1}^{n}\mathrm{E}\big((y^{(n+i)} - \mu_M^{(i)}) - (\hat{y}_M^{(i)} - \mu_M^{(i)})\big)^2 \\ &= \sum_{i=1}^{n}\mathrm{E}(y^{(n+i)} - \mu_M^{(i)})^2 + 2\mathrm{E}\big((y^{(n+i)} - \mu^{(i)})(\hat{y}_M^{(i)} - \mu_M^{(i)})\big) + \mathrm{E}(\hat{y}^{(i)} - \mu_M^{(i)})^2 \\ &= \sum_{i=1}^{n}\mathrm{E}(y^{(n+i)} - \mu_M^{(i)})^2 + \sum_{i=1}^{n}\big(\mathrm{E}(\hat{y}_M^{(i)}) - \mu_M^{(i)}\big)^2 \\ &= n\sigma^2 + \mathrm{SMSE} \\ &= n\sigma^2 + |M|\sigma^2 + \sum_{i=1}^{n}\big(\mathrm{E}(\hat{y}^{(i)}) - \mu_M^{(i)}\big)^2.\end{aligned} \tag{1.27}$$

The SPSE can be split into three parts:

1. *Irreducible Prediction Error Term*: $n\sigma^2$
   This term cannot be reduced through model selection techniques since it only contains the number of data points n and the variance σ^2.

2. *Variance Term*: $|M|\sigma^2$
   The second term contains the number of used variables $|M|$ as well as the variance σ^2. It can therefore be reduced by using a smaller number of inputs.

3. *Squared Bias Term*: $\sum_{i=1}^{n} \left(\mathrm{E}(\hat{y}^{(i)} - \mu_M^{(i)})\right)^2$
   The last term can be interpreted as bias. It can be reduced by increasing the model complexity.

The SPSE acts as an example of the bias-variance trade-off, which is characteristic for all statistical models. It states that by increasing the model complexity, the bias is reduced but instead the variance is increased. On the other hand, by decreasing model complexity, the variance of the model is reduced, but the bias is increased, see Figure 1.1 **bishop2006patternRecognition**.

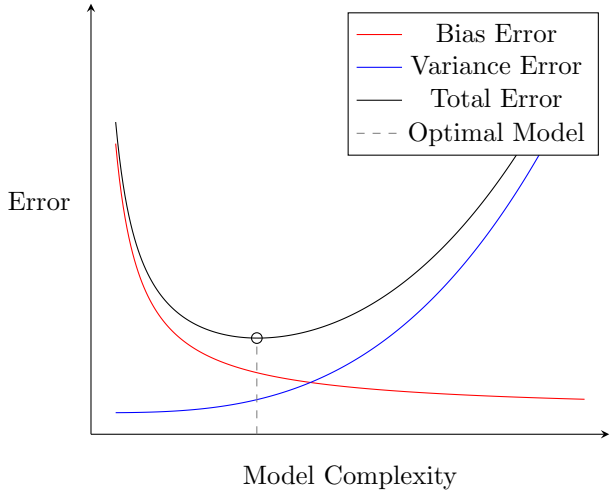

Figure 1.1: Bias-variance decomposition

In practice, the true value for the SPSE is not accessible since $\mu^{(i)}$ and σ^2 are unknown. Therefore, we need to estimate the SPSE. This can be done by using one of the following two strategies:

1. *Estimate SPSE using new and independent data*
   If new observations are available, the SPSE can be estimated by

$$\widehat{\mathrm{SPSE}} = \sum_{i=1}^{n} (y^{(n+i)} - \hat{y}_M^{(i)})^2. \tag{1.28}$$

   These new observations can also be some held-out validation data from a train-validation split of the given data.

2. *Estimate SPSE using existing data*
   When using existing data, the estimate for the SPSE is given by the squared error and an additional term depending on the estimated variance and the model complexity. The estimate is thus given by

$$\widehat{\mathrm{SPSE}} = \sum_{i=1}^{n} (y^{(i)} - \hat{y}_M^{(i)})^2 + 2|M|\hat{\sigma}^2. \tag{1.29}$$

Typically, used model assessment criteria follow the basic idea of the SPSE, see **fahrmeir2007regression**.

1.2.1.2 Corrected Coefficient of Determination

The corrected coefficient of determination R^2_{corr} is an improvement over the coefficient of determination R^2, which is defined as

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y^{(i)} - \hat{y}_M^{(i)})^2}{\sum_{i=1}^{n}(y^{(i)} - \bar{y})^2}, \tag{1.30}$$

where $\bar{y}$ is defined as the mean value of $\mathbf{y}$. The major drawback of R^2 is that it will never decrease when further inputs are included in the model, e.g. the R^2 of a model using $\{x_1, x_2, x_3\}$ is always larger or equal the R^2 of a model using $\{x_1, x_2\}$, even if the variable does not enhance the prediction quality.

The corrected coefficient of determination R^2_{corr} reduces this problem by an correction term depending on the number of parameters and is given by

$$R^2_{corr} = 1 - \frac{n-1}{n-p}(1 - R^2). \tag{1.31}$$

The corrected coefficient of determination is a standard output parameter in many statistical programs and may be used to compare even models with different number of used variables **fahrmeir2007regression**.

1.2.1.3 Mallow's Cp

Mallow's complexity parameter is based directly on the ideas specified for the estimation of the SPSE and is given by

$$\mathrm{C}_p = \frac{\sum_{i=1}^{n}(y^{(i)} - \hat{y}_M^{(i)})^2}{\hat{\sigma}^2} - n + 2|M|. \tag{1.32}$$

A lower value of Mallow's C_p corresponds to a better model fit **fahrmeir2007regression**.

1.2.1.4 Akaike Information Criterion

The AIC is among the most used model assessment criteria and defined by

$$\mathrm{AIC} = -2l(\hat{\boldsymbol{\beta}}_{ML}, \hat{\sigma}^2_{ML}) + 2(|M| + 1), \tag{1.33}$$

where $l(\hat{\boldsymbol{\beta}}_{ML}, \hat{\sigma}^2_{ML})$ is the value of the log-likelihood (1.17) at its maximum, i.e. at $\hat{\boldsymbol{\beta}}_{ML}$ and $\hat{\sigma}_{ML}$. It is worth noting that the total number of parameters is $|M| + 1$ because the variance is also counted as parameter. The log-likelihood for a linear model assuming Gaussian errors is given by, cf. (1.17),

$$-2l(\hat{\boldsymbol{\beta}}_{ML}, \hat{\sigma}^2_{ML}) = n \log(\hat{\sigma}^2_{ML}) + n. \tag{1.34}$$

Therefore, neglecting the constant n, the AIC evaluates to

$$\mathrm{AIC} = n \log(\hat{\sigma}^2_{ML}) + 2(|M| + 1). \tag{1.35}$$

A lower value of the AIC means a to a better model fit **fahrmeir2007regression**.

1.2.1.5 Bayesian Information Criteria

The BIC is similar to the AIC, but it penalizes more complex models much harder than the AIC. In its general form, it is given as

$$\mathrm{BIC} = -2l(\hat{\boldsymbol{\beta}}_{ML}, \hat{\sigma}^2_{ML}) + \log(n)(|M| + 1). \tag{1.36}$$

Again, assuming Gaussian errors for a linear model and neglecting the constant term n, the BIC evaluates to

$$\mathrm{BIC} = n \log(\hat{\sigma}^2_{ML}) + \log(n)(|M| + 1). \tag{1.37}$$

A lower value of the BIC correspond to a better model fit **fahrmeir2007regression**.

1.2.1.6 Cross-validation

The basic idea of cross-validations is to split the given data set into a training set to estimate the parameters and a validation set to assess the prediction quality. A special case of cross-validation is the "leave-one-out" cross-validation, where all but one data point are used for training and the model is then evaluated on this held-out data point. This seems to be quite expensive, since one needs to estimate one model per data point. However, in the context of linear models, it can be shown that the cross-validation score can be computed using one model trained on all data $\mathbf{y}$ and the *hat matrix* $\mathbf{H}_M = \mathbf{X}_M(\mathbf{X}_M^{\mathrm{T}}\mathbf{X}_M)^{-1}\mathbf{X}_M^{\mathrm{T}}$. The cross-validation score is then given by

$$\mathrm{CV} = \frac{1}{n}\sum_{i=1}^{n}\Big(\frac{y^{(i)} - \hat{y}_M^{(i)}}{1 - h_{ii,M}}\Big)^2. \tag{1.38}$$

where $h_{ii,M}$ denote the diagonal elements of the *hat matrix* and $\hat{y}_M^{(i)}$ is defined as the prediction of the model for the input $\{x_1^{(i)}, \ldots, x_q^{(i)}\}$. A lower cross-validation score corresponds to a better model fit **golub1979**.

An approximation to the cross-validation score is given by the so-called generalized cross-validation score. It is mainly used in the context of non-parametric regression or when the hat matrix $\mathbf{H}_M$ is numerically expensive to compute. In the GCV score, the diagonal elements of the hat matrix $h_{ii,M}$ are replaced by the mean of the trace of $\mathbf{H}_M$. The GCV score is then given by

$$\mathrm{GCV} = \frac{1}{n}\sum_{i=1}^{n}\Big(\frac{y^{(i)} - \hat{y}_M^{(i)}}{1 - \mathrm{trace}(\mathbf{H}_M)/n}\Big)^2. \tag{1.39}$$

The numerical advantage comes from the fact that the trace of a product of matrices is invariant to cyclical permutations, i.e.

$$\mathrm{trace}(\mathbf{H}_M) = \mathrm{trace}(\mathbf{X}_M(\mathbf{X}_M^{\mathrm{T}}\mathbf{X}_M)^{-1}\mathbf{X}_M^{\mathrm{T}}) = \mathrm{trace}(\mathbf{X}_M^{\mathrm{T}}\mathbf{X}_M(\mathbf{X}_M^{\mathrm{T}}\mathbf{X}_M)^{-1}). \tag{1.40}$$

The trace can therefore be computed from the product of two matrices of shape $p \times p$ **fahrmeir2007regression**.

1.2.2 Subset Selection Methods

To make use of the various model assessment criteria, some algorithmic approach to model selection needs to be given. The most commonly used approaches are forward, backward and stepwise selction **fahrmeir2007regression**.

In forward selection, start with a candidate model, which includes a small number of variables. In each iteration of forward selection, an additional variable is added to the candidate model. The added variable is the one with leads to the largest reduction of a predefined model assessment criteria. The algorithm stops, if no further reduction is achieved.

In backward selection, start with a candidate model, which includes all variables. In each iteration of backward selection, we eliminate the variable from the model which provides the largest reduction of a predefined model assessment criteria. The algorithm stops, if no further reduction is possible.

In step-wise selection, forward and backward selection are combined to enable the inclusion and deletion of a variable in every operation. The algorithm stops, if no further reduction is possible.

1.2.3 Regularization

Model selection can also be achieved using regularization techniques by directly influencing the parameters $\boldsymbol{\beta}$, which need to be estimated given a data set. In general, regularization restricts the parameter space by adding some penalty term depending on the complexity of the model to the least squares objective function according to (1.10). This leads to the penalized least squares (PLS) criterion

$$\mathrm{PLS}(\mathbf{y}, \boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \cdot \mathrm{pen}(\boldsymbol{\beta}), \tag{1.41}$$

where λ is the so-called smoothing parameter and $\mathrm{pen}(\boldsymbol{\beta})$ is the penalty term describing the regularization technique.

In Ridge regression, the penalty term in the penalized least squares criterion in (1.41) is given by the squared weighted L_2-norm of the parameter vector $\boldsymbol{\beta}$, i.e. $\mathrm{pen}(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_{\mathbf{K}}^2 = \boldsymbol{\beta}^{\mathrm{T}}\mathbf{K}\boldsymbol{\beta}$ with penalty matrix $\mathbf{K} \in \mathbb{R}^{p \times p}$. The closed form solution reads as

$$\hat{\boldsymbol{\beta}}_{PLS} = \arg\min_{\boldsymbol{\beta}} \big(\mathrm{PLS}(\mathbf{y}, \boldsymbol{\beta})\big) = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda\mathbf{K})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}, \tag{1.42}$$

The additional penalty term in Ridge regression leads to smaller parameter estimates $\hat{\boldsymbol{\beta}}_{PLS}$ compared to the unpenalized estimate $\hat{\boldsymbol{\beta}}_{LS}$. For large values of the smoothing parameter λ, the parameter estimates will converge towards, but never reach, zero.

Ridge regression is commonly used when the input dimension q is high, i.e. the number of parameters β_i is large, and also known as Tikhonov regularization **hoerl1970ridge**. Note that it is also possible to use a nonlinear penalty matrix $\mathbf{K}(\boldsymbol{\beta})$ resulting in

$$\mathrm{pen}(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_{\mathbf{K}(\boldsymbol{\beta})}^2 = \boldsymbol{\beta}^{\mathrm{T}}\mathbf{K}(\boldsymbol{\beta})\boldsymbol{\beta}. \tag{1.43}$$

However, the resulting penalized least squares problem has no closed form solution and must be solved by an iterative approach. We start with an initial guess $\boldsymbol{\beta}^{[0]}$ and iterate for $k = 0, 1, 2, \ldots$ the iteration

$$\boldsymbol{\beta}^{[k+1]} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda\mathbf{K}(\boldsymbol{\beta}^{[k]})\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}. \tag{1.44}$$

until $|\boldsymbol{\beta}^{[k+1]} - \boldsymbol{\beta}^{[k]}| \leq \text{Tol}$ with Tol being some given tolerance.

?? Convergence ??

1.3 Splines

A spline is a piecewise polynomial defined on a sequence of knots. This definition is quite general. Therefore, a large variety of splines exists, ranging from regression splines **eubank1990regressionsplines**, over B-splines **deBoor1978practicalGuideToSplines** to natural cubic splines and many more. We will focus on the definition of B-splines in Section 1.3.1, tensor-product B-splines as the multi-dimensional expansion of B-splines in Section 1.3.1.1, and P-splines in Section 1.3.2 **deBoor1978practicalGuideToSplines eilers1996flexible**.

1.3.1 B-Splines

We put the focus on the definition and use of B-splines, which are constructed from polynomial pieces in a recursive manner. The B-spline $B_j^l(x)$ of degree l is defined by means of the Cox-de Boor recursion formula as **fahrmeir2007regression**

$$B_j^0(x) = \begin{cases} 1 & \text{for} \quad \kappa_j \leq x < \kappa_{j+1} \\ 0 & \text{otherwise} \end{cases} \tag{1.45}$$

$$B_j^l(x) = \frac{x - \kappa_{j-l}}{\kappa_j - \kappa_{j-l}} B_{j-1}^{l-1}(x) + \frac{\kappa_{j+1} - x}{\kappa_{j+1} - \kappa_{j+1-l}} B_j^{l-1}(x) \tag{1.46}$$

given the knot sequence

$$K = \{\kappa_{1-l}, \kappa_{1-l+1}, \ldots, \kappa_{m+l-1}, \kappa_{m+l}\} \tag{1.47}$$

with the m interior knots $\kappa_1, \ldots, \kappa_m$ and the $2l$ boundary knots **fahrmeir2007regression**.

An example of B-splines of order $l = 0, 1, 2$ and 3 is given in Figure 1.2. The left chart shows the B-splines based on an equidistant sequence of knots. The B-spline B^0 is the zero function, except for $x \in [\kappa_1, \kappa_2]$ where it is equal to 1. The B-spline B^1 is the well known *hat function*, being zero except for $x \in [\kappa_1, \kappa_3]$. It consists of two linear pieces, one defined from κ_1 to κ_2 and the other from κ_2 to κ_3. Everywhere else, B^1 is equal to zero. At the joining points, the values of the linear pieces are equal. The B-spline B^2 consists of three quadratic pieces, joining at the knots κ_2 and κ_3. At these point, the values of the quadratic pieces, as well as their first derivatives are equal. Finally, the B-spline B^3 consists of 4 cubic pieces with the joining points at κ_2, κ_3 and κ_4 at which respective cubic polynomials possess equal values as well as equal first

and second derivatives. The right part of Figure 1.2 shows the same B-splines defined on a non-equidistant knot sequence. The basic properties of these B-splines are the same as for B-splines based on equidistant knots.

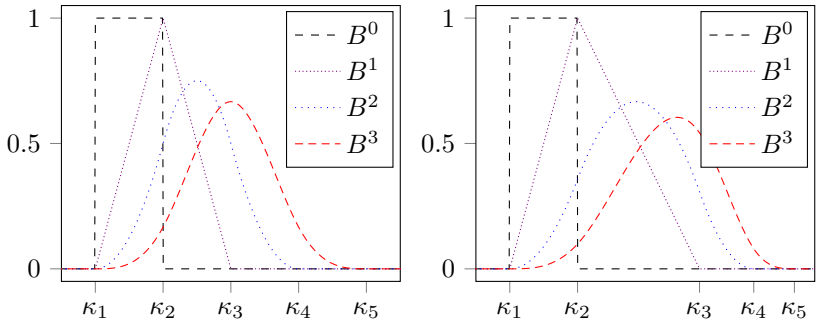

Figure 1.2: B-splines of orders $l = 0, 1, 2, 3$ for equidistant (left) and non-equidistant (right) knots

The shown locality is a very attractive feature leading to an enhanced numerical stability compared to other types of splines. Some general properties of B-splines of degree l are summarized in the following list:

- It consists of $l+1$ polynomial pieces of degree l, e.g. a cubic spline ($l = 3$) consists of 4 cubic pieces.
- The pieces join at l inner knots.
- At these knots, the derivatives up to order $l - 1$ are continuous.
- The B-spline is positive on the domain spanned by $l+2$ knots, everywhere else it is zero, e.g. for $l = 2$, a sequence of 4 knots is necessary.
- At every given x, only $l + 1$ B-splines are non-zero.

The collection of $d = m + l - 1$ B-splines of degree l over a sequence K of $m + 2l$ knots is called B-spline basis. The $2l$-knots are the boundary knots while the m knots are the interior knots. The knots can either be an equidistant sequence, which facilitates the construction and estimation of the coefficients, or a non-equidistant sequence **eilers1996flexible**. For equidistant knots, we split the domain $[a, b]$ into $m - 1$ intervals, i.e

$$h = \frac{b - a}{m - 1} \tag{1.48}$$

and obtain the sequence

$$\kappa_j = a + h(j - 1), \quad j = 1, \ldots, m. \tag{1.49}$$

Non-equidistant knot placement can be obtained using quantile-based knots, i.e. by using the $(j - 1)/(m - 1)$-quantiles for $j = 1, \ldots, m$ of the observed inputs $x^{(1)}, \ldots, x^{(n)}$ as knots. Using this approach, more knots are placed in the areas

where lots of data is present **fahrmeir2007regression**. The boundary knots are usually set to be apart from each other by at least the minimal knot distance.

A function $f(x)$ can then be represented utilizing the basis function approach by

$$f(x) = \sum_{j=1}^{d} B_j^l(x)\beta_j = \mathbf{b}^{\mathrm{T}}\boldsymbol{\beta}, \tag{1.50}$$

using the B-spline basis functions $B_j^l(x)$ of appropriate degree l and the parameter vector $\boldsymbol{\beta}^{\mathrm{T}} = [\beta_1, \ldots, \beta_d] \in \mathbb{R}^d$. The basis functions can be given in vector notation as $\mathbf{b}^{\mathrm{T}} = [B_1^l(x), \ldots, B_d^l(x)]$. Using the set of data $D = \{(x^{(i)}, y^{(i)}),\ i = 1, \ldots, n\}$, the B-spline basis for d splines of degree l is given by the matrix $\mathbf{B}$ as

$$\mathbf{B} = \begin{bmatrix} B_1^l(x^{(1)}) & \ldots & B_d^l(x^{(1)}) \\ \vdots & & \vdots \\ B_1^l(x^{(n)}) & \ldots & B_d^l(x^{(n)}) \end{bmatrix} \in \mathbb{R}^{n \times d}. \tag{1.51}$$

The n equations (1.50) can be arranged as a linear model in the form

$$\mathbf{y} = \mathbf{B}\boldsymbol{\beta}. \tag{1.52}$$

A further advantage of B-splines is that once the basis in (1.51) is given, the parameter can be estimated using the Least Squares algorithm given in Section 1.1.1.1. Therefore, the estimation is computationally efficient and easy to implement since closed-form solutions exists. Further, the advanced theoretical framework of linear models can be applied to use model selection and regularization approaches as well as to calculate e.g confidence intervals for the regression coefficients and the prediction. B-splines of appropriate order $l > 2$ produce smooths curves, i.e. first and second order derivatives are continuous, where the smoothness is mostly determined by the number of splines used. Using a low number d, the curve will be quite smooth, but possess a large data error. When using a high number of splines d, the data error will be small but the variance of the curve will be large. This is an example of the bias-variance trade-off, a classical problem of regression and machine learning and further discussed in Section 2.1.1. It is therefore necessary to introduce some kind of regularization **deBoor1978practicalGuideToSplines**.

1.3.1.1 Tensor-Product Splines

Tensor-product splines can be seen as the multi-dimensional extension of univariate B-splines. We start with a B-spline basis, cf. (1.51), for each dimension and construct the tensor-product spline from these.

We examine an example for two input dimensions x_1 and x_2. Tensor-product splines can in general be constructed for arbitrary dimensions using the approach given below. Assume that we have B-spline bases available, i.e. $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\mathbf{B} \in \mathbb{R}^{n \times d}$ using the same number of splines d for both bases and degree $l = 3$, for representing the functions $f_1(x_1)$ and $f_2(x_2)$, cf. (1.50), given by

$$f_1(x_1) = \sum_{i=1}^{d} A_i^l(x_1)\alpha_i = \mathbf{a}^{\mathrm{T}}\boldsymbol{\alpha}, \tag{1.53}$$

$$f_2(x_2) = \sum_{j=1}^{d} B_j^l(x_2)\beta_j = \mathbf{b}^{\mathrm{T}}\boldsymbol{\beta} \tag{1.54}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^d$ and $\boldsymbol{\beta} \in \mathbb{R}^d$ are the coefficient vectors and $A_i^l(x_1)$ and $B_j^l(x_2)$ are the basis functions for degree l for the respective dimension. To allow the function $f_1(x_1)$ to smoothly vary with x_2, its coefficients α_i must vary smoothly with x_2. By using the already available basis for representing smooth functions in x_2, we can write

$$\alpha_i(x_2) = \sum_{j=1}^{d} B_j^l(x_2)\beta_{ij} \tag{1.55}$$

which leads to

$$f_{1,2}(x_1, x_2) = \sum_{i=1}^{d}\sum_{j=1}^{d} A_i^l(x_1)B_j^l(x_2)\beta_{ij}. \tag{1.56}$$

We can therefore combine the individual basis matrices to generated a new basis matrix for the tensor-product spline. For any set of data $\{x_1^{(i)}, x_2^{(i)}\},\ i = 1, \ldots, n$, the relationship between the basis matrix $\mathbf{X}$ of the tensor-product spline and the marginal basis matrices $\mathbf{A}$ and $\mathbf{B}$, cf. (1.51), is given by

$$\mathbf{X} = \mathbf{A} \otimes \mathbf{B} \tag{1.57}$$

where $\otimes$ indicates the use of the row-wise Kronecker product, $\mathbf{X} \in \mathbb{R}^{n \times d^2}$ denotes the tensor-product spline basis, $\mathbf{A} \in \mathbb{R}^{n \times d}$ denotes the B-spline basis for dimension x_1 and $\mathbf{B} \in \mathbb{R}^{n \times d}$ denotes the B-spline basis for dimension x_2. The basis matrix for the tensor-product spline is therefore given by the row-wise Kronecker product of the marginal basis matrices. **wood2017generalized**

We can then model a two dimensional function $f(x_1, x_2)$ using data $\{x_1^{(i)}, x_2^{(i)}; y^{(i)}\},\ i = i, \ldots, n$ as in (1.50) as

$$f(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{X}\boldsymbol{\gamma} \tag{1.58}$$

with the tensor-product spline basis matrix $\mathbf{X} \in \mathbb{R}^{n \times d^2}$ and the coefficient vector $\boldsymbol{\gamma} \in \mathbb{R}^{d^2}$. The coefficient vector is obtained by reordering β_{ij} in (1.56).

This approach can in theory be continued for as much input dimensions as required. In practice, modeling more than two input dimensions using tensor-product splines becomes infeasible because of the exponential increase of basis functions and therefore coefficients to estimate.

1.3.1.2 Additive Regression

To circumvent this problem, we now assume the restrictive structure of additive models, given by

$$f(x_1, \ldots, x_q) = f_1(x_1) + \cdots + f_q(x_q) + \epsilon. \tag{1.59}$$

for some given data $\{x_1^{(i)}, \ldots, x_q^{(i)}; y^i\},\ i = 1, \ldots, n$.

Hence, we use one function $f_i(x_i)$ per input dimension. **fahrmeir2007regression** Using the concepts introduced above, i.e. using B-splines to estimate the function $f_i(x_i)$, we obtain for each function a linear model

$$f_i(\mathbf{x}_i) = \mathbf{X}_{s_i} \boldsymbol{\beta}_{s_i} \tag{1.60}$$

where $\mathbf{X}_{s_i} \in \mathbb{R}^{n \times d_i}$ is the B-spline basis using d_i splines for the smooth function $f_i(x_i)$ of input dimension i, $\mathbf{x}_i = (x_i^{(1)}, \ldots, x_i^{(n)})^{\mathrm{T}}$ is the data vector of input dimension i and $\boldsymbol{\beta}_{s_i} \in \mathbb{R}^{d_i}$ are the coefficients to be estimated.

The model given in (1.60) does not contain interaction terms between variables. Nevertheless, these can be easily introduced for 2 dimensions using tensor-product splines without an overflowing increase in the number of coefficients.

We can then write the additive model with interaction terms in matrix notation as

$$\mathbf{y} = \mathbf{X}_{s_1} \boldsymbol{\beta}_{s_1} + \cdots + \mathbf{X}_{s_q} \boldsymbol{\beta}_{s_q} + \sum_{j=1}^{q-1} \sum_{r>j}^{q} \mathbf{X}_{t_{j,r}} \boldsymbol{\beta}_{t_{j,r}} + \boldsymbol{\epsilon} \tag{1.61}$$

using the error term $\boldsymbol{\epsilon} \in \mathbb{R}^n$, the tensor-product spline bases $\mathbf{X}_{t_{j,r}} \in \mathbb{R}^{n \times d_j d_r}$ and the tensor-product spline basis coefficients $\boldsymbol{\beta}_{t_{j,r}}$. This can be solved using ordinary least squares.

Using the notation in (1.61) the theoretical framework of linear models can be applied to the additive regression models, since (1.61) can be formulated as large linear model as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} = \begin{pmatrix} \mathbf{X}_{s_1} & \mathbf{X}_{s_2} & \ldots & \mathbf{X}_{s_q} & \mathbf{X}_{t_{1,2}} & \ldots & \mathbf{X}_{t_{q-1,q}} \end{pmatrix} \begin{pmatrix} \boldsymbol{\beta}_{s_1} \\ \boldsymbol{\beta}_{s_2} \\ \vdots \\ \boldsymbol{\beta}_{s_q} \\ \boldsymbol{\beta}_{t_{1,2}} \\ \vdots \\ \boldsymbol{\beta}_{t_{q-1,q}} \end{pmatrix} \tag{1.62}$$

with $\mathbf{X}$ as large design matrix and $\boldsymbol{\beta}$ as combined coefficient vector. Therefore, the assumptions given in Chapter 1.1 on the error term, as well as on the model functions are used. **fahrmeir2004penalized**

1.3.2 P-Splines

P-splines use the concepts of regularization to produce smooth function estimations. In our context, a function is said to be smooth if its 2^{nd} derivative is continuous and does not vary much. They where introduced by Eilers and Marx in **eilers1996flexible** to overcome the problem stated above. Eilers and Marx simplified and generalized the idea of Osullivan in **osullivan1986penalties**, who introduced a smoothness penalty based on the integral of the squared second derivative of the estimated spline to penalized wiggly function estimates. They proposed to use equidistant knot placement and a penalty based on finite differences of order D of the coefficients of adjacent B-splines as approximation of the derivative.

The difference operator Δ^D of order D for equidistant knot placement of B-splines is defined by

$$\begin{aligned} \Delta^1\beta_j &= \beta_j - \beta_{j-1} \\ \Delta^2\beta_j &= \Delta^1(\Delta^1\beta_j) = \beta_j - 2\beta_{j-1} + \beta_{j-2} \\ &\vdots \\ \Delta^D\beta_j &= \Delta^1(...(\Delta^1\beta_j)) \end{aligned} \tag{1.63}$$

and in matrix notation for order $D = 1$

$$\mathbf{D}_1 = \begin{pmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{pmatrix} \in \mathbb{R}^{d-1\times d} \tag{1.64}$$

and order $D = 2$

$$\mathbf{D}_2 = \begin{pmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \\ & & & 1 & -2 & 1 \end{pmatrix} \in \mathbb{R}^{d-2\times d}. \tag{1.65}$$

For non-equidistant knot placement, the difference operator Δ^1 is defined according to **ferziger2008numerische** as

$$\Delta^1\beta_j = \frac{\beta_j - \beta_{j-1}}{\delta_{j,j-1}} \tag{1.66}$$

with the denominator given by

$$\delta_{j,j-1} = \arg\max_x B_j^l(x) - \arg\max_x B_{j-1}^l(x) \tag{1.67}$$

for the B-spline basis functions $B_j^l(x)$ and $B_{j-1}^l(x)$ of degree l. The second order difference operator is further defined using the first order difference operator Δ^1 as

$$\Delta^2\beta_j = \frac{\Delta^1\beta_{j+1} - \Delta^1\beta_j}{\delta_{j+1,j}}. \tag{1.68}$$

In matrix form, the first order difference operator for non-equidistant knot placement is then given as

$$\mathbf{D_1} = \begin{pmatrix} \frac{-1}{\delta_{2,1}} & \frac{1}{\delta_{2,1}} & & & \\ & \frac{-1}{\delta_{3,2}} & \frac{1}{\delta_{3,2}} & & \\ & & \ddots & \ddots & \\ & & & \frac{-1}{\delta_{k-1,k-2}} & \frac{1}{\delta_{k-1,k-2}} \end{pmatrix} \in \mathbb{R}^{d-1\times d} \tag{1.69}$$

and the second order difference operator for non-equidistant knot placement is given as

$$\mathbf{D}_2 = \begin{pmatrix} \frac{1}{\delta_{3,2}\delta_{2,1}} & \frac{-\delta_{3,1}}{\delta_{3,2}^2\delta_{2,1}} & \frac{1}{\delta_{3,2}^2} & & & \\ & \frac{1}{\delta_{4,3}\delta_{3,2}} & \frac{-\delta_{4,2}}{\delta_{4,3}^2\delta_{3,2}} & \frac{1}{\delta_{4,3}^2} & & \\ & & \ddots & \ddots & \ddots & \\ & & & \frac{1}{\delta_{k-1,k-2}\delta_{k-2,k-3}} & \frac{-\delta_{k-1,k-3}}{\delta_{k-1,k-2}^2\delta_{k-2,k-3}} & \frac{1}{\delta_{k-1,k-2}^2} \end{pmatrix} \in \mathbb{R}^{d-2\times d} \tag{1.70}$$

using the definition of $\delta_{j,j-1}$ in (1.67). This leads to the penalized least squares formulation, similar to (1.41),

$$Q_{pslines}(\mathbf{y}, \boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda_s \mathcal{J}_s(\boldsymbol{\beta}; D) \tag{1.71}$$

where $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2$ is the mean squared error on the data for the spline fit, $\mathcal{J}_s(\boldsymbol{\beta}; D) = \boldsymbol{\beta}^{\mathrm{T}}\mathbf{D}_D^{\mathrm{T}}\mathbf{D}_D\boldsymbol{\beta}$ is the smoothness penalty term given by the matrix form of the knot placement dependent difference operator Δ^D of order D and λ_s is the smoothness parameter determining the effect of the smoothness penalty. The estimated coefficients $\hat{\boldsymbol{\beta}}$ are then given by the minimization of

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} Q_{psplines}(\mathbf{y}, \boldsymbol{\beta}). \tag{1.72}$$

The penalized least squares formulation in (1.71) then yields

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda_s \mathbf{D}_D^{\mathrm{T}}\mathbf{D}_D)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}. \tag{1.73}$$

The main advantage of P-splines is their easy set up. This advantage is diminished if uneven knot placement is chosen, which is seen when comparing the matrix forms of the difference operators for the different knot placement types in e.g. (1.65) and (1.70).

A smoothness penalty term for tensor-product splines can be constructed using the Kronecker product. For this, we need to define the term *adjacent*

coefficients for two dimensional tensor-product splines. As one dimensional splines cover the input space $\{x_1\}$, two dimensional tensor-product splines cover the input space $\{x_1, x_2\}$ via the knot placement. To facilitate the description, we restrict ourselves to equidistant knot placement. **fahrmeir2007regression** An example for the definition of adjacent coefficients, also called neighborhood, is taken from **fahrmeir2007regression** and given in Figure 1.3.

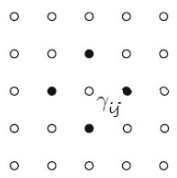

Figure 1.3: Neighborhood or adjacent coefficients for tensor-product splines

The adjacent coefficients to penalized for the coefficient $\gamma_{i,j}$, as in Chapter 1.3.2, are therefore given by the coefficients left and right, i.e. $\gamma_{i-1,j}$ and $\gamma_{i+1,j}$, and the coefficients above and below, i.e. $\gamma_{i,j-1}$ and $\gamma_{i,j+1}$ for $i, j = 1, \ldots, d$. We now want to penalize adjacent coefficient differences in each dimension. We obtain the row-wise differences of order D by applying the expanded difference matrix $\mathbf{I}^2 \otimes \mathbf{D}_D^1$ to the coefficient vector $\boldsymbol{\gamma}$, i.e.

$$\boldsymbol{\gamma}^{\mathrm{T}}(\mathbf{I}^2 \otimes \mathbf{D}_D^1)^{\mathrm{T}}(\mathbf{I}^2 \otimes \mathbf{D}_D^1)\boldsymbol{\gamma} = \sum_{j=1}^{d}\sum_{i=1}^{d}(\gamma_{i,j} - \gamma_{i-1,j})^2 \tag{1.74}$$

using the identity matrix $\mathbf{I}^2 \in \mathbb{R}^{d\times d}$ of dimension x_2 and the difference matrix $\mathbf{D}_D^1$, cf. (1.64) and (1.65) for dimension x_1. The column wise differences are obtained analogously using the identity matrix $\mathbf{I}^1 \in \mathbb{R}^{d\times d}$ of dimension x_1 and the difference matrix $\mathbf{D}_D^2$. Summing up all column-wise and row-wise differences and using the properties of the Kronecker-product yields the penalty term $\mathcal{J}_s(\boldsymbol{\gamma}; D)$ as

$$\mathcal{J}_s(\boldsymbol{\gamma}; D) = \boldsymbol{\gamma}^{\mathrm{T}}\Big[\mathbf{I}^2 \otimes \mathbf{K}_D^1 + \mathbf{K}_D^2 \otimes \mathbf{I}^1\Big]\boldsymbol{\gamma} \tag{1.75}$$

with the respective penalty matrices $\mathbf{K}_D^1 = \mathbf{D}_D^{1\mathrm{T}}\mathbf{D}_D^1$ for dimension x_1 and $\mathbf{K}_D^2 = \mathbf{D}_D^{2\mathrm{T}}\mathbf{D}_D^2$ for dimension x_2 of order D.

With the penalty term in (1.75), we can use the penalized least squares formulation, similar to (1.41) and (1.71).

Chapter 2

Solution Approach

We are now going to use the theory discussed in Chapter 1 to estimate functions using available data and a priori domain knowledge. An overview of the different problems considered in this work is given in Table 2.1. First, we are using B-splines, see Chapter 1.3.1, as basis functions for the estimation of the unknown function $y = f(x)$ for some data $\{x^{(i)}, y^{(i)}\}$ $i = 1, \ldots, n$. Next, we use the concept of P-splines, see 1.3.2, introduced by Eilers and Marx in **eilers1996flexible** to estimate smooth, one dimensional functions. Finally, we are going to incorporate a priori knowledge into the fitting process using the approach given by Hofner and apply it to one and two dimensional functions. **hofner2011monotonicity**

Problem	**Solution Approach**	**Algorithm**
1d Function Estimation	B-Splines	LS
1d Smooth Function Estimation	P-Splines	PLS
1d Constraint Function Estimation	P-Splines + Constraint Penalty	PIRLS
n-d Constraint Function Estimation	P+TP-Splines + Constraint Penalty	PIRLS

Table 2.1: Problem overview

The a priori knowledge can be incorporated using different types of constraints. The considered constraints are listed in Table 2.2.

Constraint	**Description**	**Math. Description**
Monotonicity	Functions is either increasing or decreasing.	$\lvert f'(x)\rvert \geq 0$
Curvature	Function is either convex or concave.	$\lvert f''(x)\rvert \geq 0$
Unimodality	Function has a mode/peak.	$m = \arg\max_x f(x)$ $f'(x) \geq 0 \quad \text{if } x < m$ $f'(x) \leq 0 \quad \text{if } x > m$
Boundedness	Function is bounded by the value M.	$\lvert f(x)\rvert \leq M$
Jamming	Function is jammed by the value M.	$f(x^{(M)}) \approx y^{(M)}$

Table 2.2: Overview of the considered constraints

To test the algorithm and the incorporation of a priori knowledge, we use

the one dimensional function given by

$$f(x) = 3\sin(3\pi x) + 17x + 3. \tag{2.1}$$

Figure 2.1 shows functions evaluated at 200 points. The function is partially increasing. The test function for the two dimensional case is given in Chapter 2.3.4.

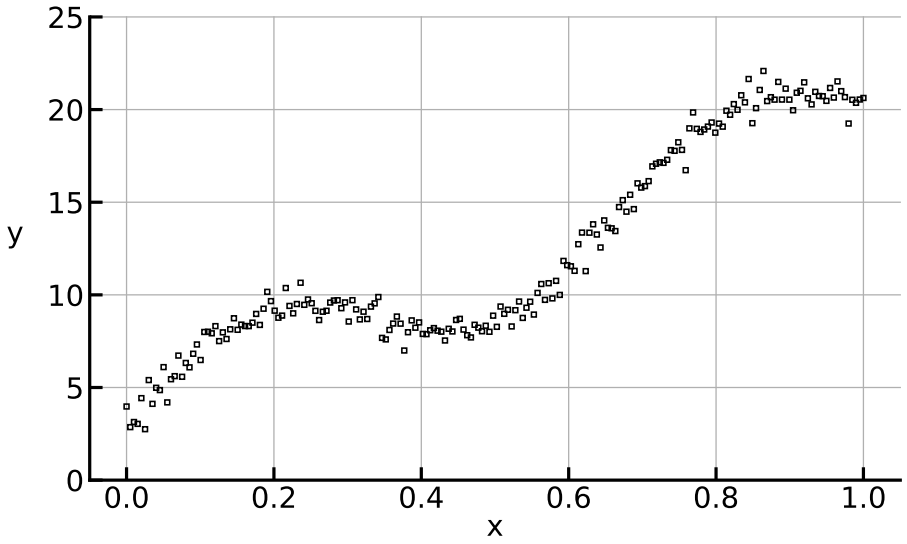

Figure 2.1: Noisy samples determined from test function (2.1)

2.1 Function Estimation

2.1.1 1d Function Estimation

The goal is to model given data

$$\{\mathbf{x}, \mathbf{y}\} = \{x^{(i)}, y^{(i)}\},\ i = 1, \ldots, n \tag{2.2}$$

using B-splines as basis functions. Therefore, we want to estimate the unknown function $\mathbf{y} = f(\mathbf{x})$, which can be represented as a linear combination of k B-spline basis functions B_j^m of degree $m = 3$, cf. (1.50), as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta}, \tag{2.3}$$

where $\mathbf{X} \in \mathbb{R}^{n \times k}$ is the B-spline basis matrix, cf. (1.51), and $\boldsymbol{\beta} \in \mathbb{R}^k$ are the coefficients to be estimated.

The least squares objective function to be minimized using the complete data is then given by

$$Q_1(\mathbf{y}, \boldsymbol{\beta}) = \|\mathbf{y} - f(\mathbf{x})\|^2 = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 \tag{2.4}$$

The coefficients are determined ive function Q_1 given in (2.4) with respect to $\boldsymbol{\beta}$, i.e.

$$\hat{\boldsymbol{\beta}}_{LS} = \arg\min_{\boldsymbol{\beta}} Q_1(\mathbf{y}, \boldsymbol{\beta}). \tag{2.5}$$

Using the least squares algorithm LS, see Chapter 1.1.1.1, the minimization problem (2.5) yields

$$\hat{\boldsymbol{\beta}}_{LS} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}. \tag{2.6}$$

Figure 2.2 shows a B-spline model using $k = 10$ splines on an equidistant grid approximating the noisy data presented in Figure 2.1, as well as the individual cubic ($m = 3$) B-spline basis functions $B_i^3(\mathbf{x})$ multiplied with the corresponding, estimated coefficients $\hat{\boldsymbol{\beta}}_{LS,j}$ $j = 1, \ldots, 10$.

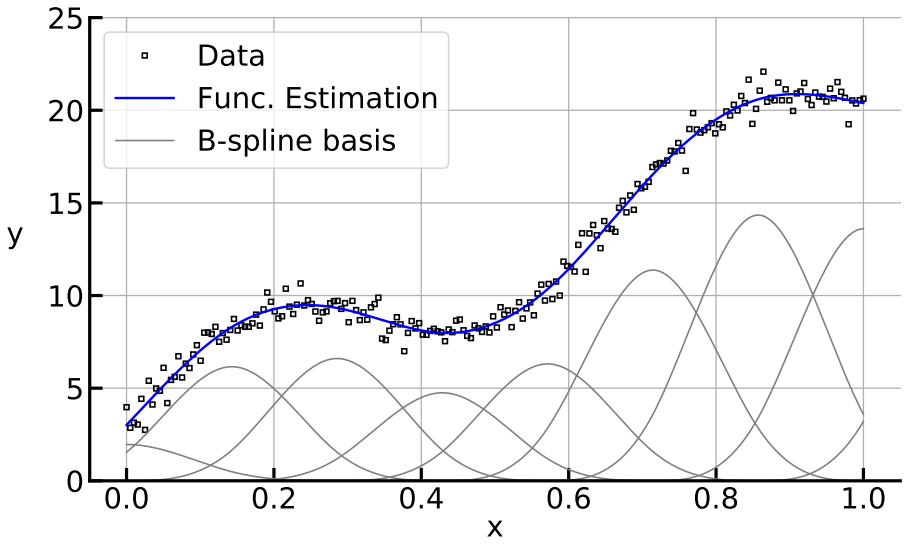

Figure 2.2: Approximation of the noisy data by B-splines without constraints

Note, the number of splines k has a strong influence on the amount of smoothing. A small number k leads to a very smooth estimate, but a large data error. On the other hand, when the number of splines is relatively large, the data error is very small but the smoothness of the estimate is poor. This behavior is an example of the well-known bias-variance dilemma and depicted in Figure 2.3. **sammut2011** Here, two B-splines models with $k = 10$ and $k = 50$ are illustrated, which are applied to the noisy data shown in Figure 2.1. To overcome this challenges, the B-splines will be extended by penalizing the second derivative of the estimation, see Chapter 2.1.2.

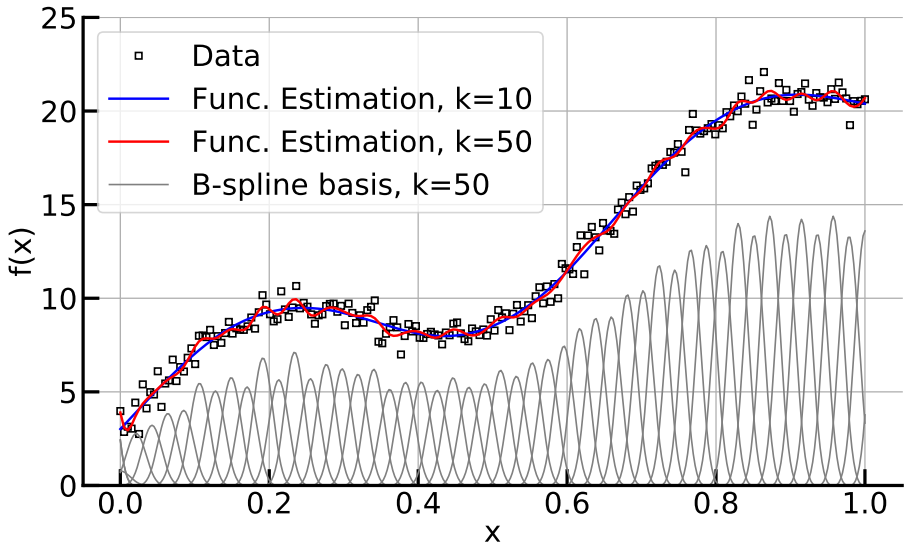

Figure 2.3: Approximation of the noisy data by 10 and 50 B-splines without constraints

2.1.2 1d Smooth Function Estimation

The second derivative of the estimated function $f(x)$, i.e. $f''(x) = \sum_{j=1}^{k} B_i''(x)\beta_k$, has to be penalized to realize a smoother estimate when using a high number of splines. Eilers and Marx have introduced the so-called P-splines. **eilers1996flexible**, see Chapter 1.3.2. Therefore, the objective function in (2.4) is extended by an additional term considering the smoothness, i.e.

$$Q_2(\mathbf{y}, \boldsymbol{\beta}) = Q_1(\mathbf{y}, \boldsymbol{\beta}) + \lambda_s \mathcal{J}_s(\boldsymbol{\beta}; d) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda_s \boldsymbol{\beta}^{\mathrm{T}} \mathbf{D}_d^{\mathrm{T}} \mathbf{D}_d \boldsymbol{\beta}, \tag{2.7}$$

with the smoothing parameter λ_s and an appropriate mapping matrix $\mathbf{D}_d$ capturing the second derivative, which itself is a measure for function wiggliness. Here, an approximation of the second derivative can be performed by the squared finite difference of order d of adjacent coefficients using the matrix form $\mathbf{D}_d$ of the difference operator of order d, see Chapter 1.3.2.

By minimizing the objective function (2.7), i.e.

$$\hat{\boldsymbol{\beta}}_{PLS} = \arg\min_{\boldsymbol{\beta}} Q_2(\mathbf{y}, \boldsymbol{\beta}), \tag{2.8}$$

using the penalized least squares algorithm PLS, the penalized least squares estimates are given by

$$\hat{\boldsymbol{\beta}}_{PLS} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda_s \mathbf{D}_d^{\mathrm{T}} \mathbf{D}_d)^{-1} \mathbf{X}^{\mathrm{T}} \mathbf{y}. \tag{2.9}$$

In (2.9), the smoothing parameter λ_s plays a critical role and can be optimized using the information criteria specified in Chapter 1.2.1, e.g. AIC and BIC, or by using cross-validation techniques, see Chapter 1.2.1.6. **fahrmeir2007regression**

For small values $\lambda_s \to 0$, the penalized least squares estimate $\hat{\boldsymbol{\beta}}_{PLS}$ approaches the least squares estimate $\hat{\boldsymbol{\beta}}_{LS}$, cf. (2.6), while for large values $\lambda_s \gg 0$, the fitted function shows the behavior of a polynomial with $d-1$ degrees of freedom. For example, using $d = 2$ and a large smoothing parameter λ_s this configuration leads to a linear function, while using $d = 1$ would lead to a constant function. **fahrmeir2007regression**

Figure 2.4 shows the behavior of P-splines of degree $m = 3$ using $k = 50$ splines for several values of the smoothing parameter $\lambda_s = \{10^{-2}, 10^2, 10^5, 10^6\}$ and a smoothness penalty of order $d = 2$. As the value of λ_s gets larger, the fitted curve becomes more smooth and thus the 2^{nd} derivative of the curve becomes smaller due to the penalty considered in the estimation, see (2.9). For very large values of λ_s, the estimate approaches a straight line, see the yellow curve in Figure 2.4.

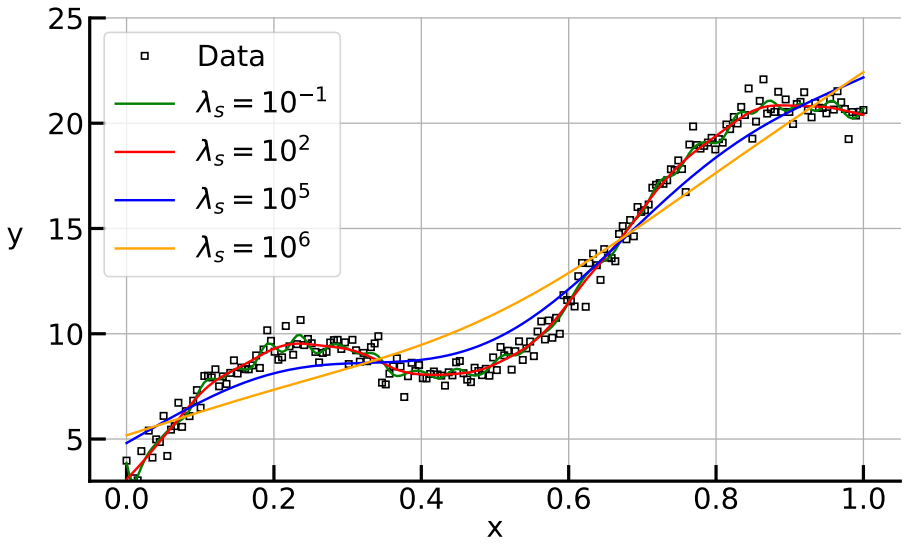

Figure 2.4: Smooth function estimation for different smoothing parameters λ_s

2.1.3 1d Constraint Function Estimation

A priori domain knowledge can now be systematically considered by the extension of the objective function (2.7) using an additional term representing the user-defined constraint, see Table 2.2. Note that this approach incorporates the a priori knowledge as soft constraints. Therefore, no guarantee can be given that the fit holds the constraint for every possible input. The constraint penalized least-squares objective function is given by

$$Q_3(\mathbf{y}, \boldsymbol{\beta}) = Q_1(\mathbf{y}, \boldsymbol{\beta}) + \lambda_s \mathcal{J}_s(\boldsymbol{\beta}; d) + \lambda_c \mathcal{J}_c(\boldsymbol{\beta}; c) \tag{2.10}$$

with the corresponding constraint parameter λ_c, which determines the influence of the user-defined constraint. Note that the parameter λ_c has to be set quite large, i.e. $\lambda_c > 10^4$, compared to λ_s to enforce the user-defined constraint.

Constraints for monotonicity, curvature, unimodality, boundedness and jamming can be modeled as

$$\mathcal{J}_c(\boldsymbol{\beta}; c) = \boldsymbol{\beta}^{\mathrm{T}} \mathbf{D}_c^{\mathrm{T}} \mathbf{V} \mathbf{D}_c \boldsymbol{\beta} \tag{2.11}$$

with the mapping matrix $\mathbf{D}_c$ and the diagonal weighting matrix $\mathbf{V} := \mathbf{V}(\boldsymbol{\beta}; c)$ capturing if the constraint c is active or inactive. The matrices $\mathbf{D}_c$ and $\mathbf{V}$ will further be defined in Chapter 2.2.

By minimizing the objective function (2.10), i.e.

$$\hat{\boldsymbol{\beta}}_{PLS,c} = \arg\min_{\boldsymbol{\beta}} Q_6(\mathbf{y}, \boldsymbol{\beta}), \tag{2.12}$$

the constraint penalized least-squares estimate can be given as

$$\hat{\boldsymbol{\beta}}_{PLS,c} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda_s \mathbf{D}_d^{\mathrm{T}} \mathbf{D}_d + \lambda_c \mathbf{D}_c^{\mathrm{T}} \mathbf{V} \mathbf{D}_c)^{-1} \mathbf{X}^{\mathrm{T}} \mathbf{y}. \tag{2.13}$$

Note, (2.13) is a nonlinear equation because the matrix $\mathbf{V}$ depends on $\boldsymbol{\beta}$. Thus, it has to be solved iteratively to calculate optimal coefficients $\hat{\boldsymbol{\beta}}_{PLS,c}$. The algorithm is shown in Figure 2.5.

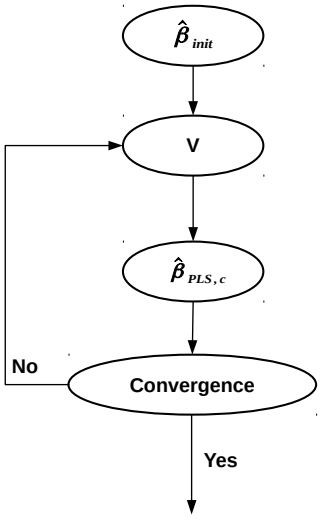

Figure 2.5: Penalized iteratively reweighted least squares algorithm

The initial estimate $\hat{\boldsymbol{\beta}}_{init}$ needed to compute the weighting matrix $\mathbf{V}$ is given by the least squares estimate $\hat{\boldsymbol{\beta}}_{LS}$. Based on the initial estimate $\hat{\boldsymbol{\beta}}_{init}$, the

weighting matrix $\mathbf{V}$ and then the constraint least-squares estimate $\hat{\boldsymbol{\beta}}_{PLS,c}$ are calculated. The algorithm is performed until no more changes in the weighting matrix $\mathbf{V}$ appear. This scheme is called the penalized iteratively reweighted least squares and is abbreviated by PIRLS. **hofner2011monotonicity**

Figure 2.6 shows an example, where the noisy data shown in Figure 2.1 is approximated by considering the monotonicity constraint. The estimate has to be monotonically increasing in contrast to the data, i.e. $f'(x) \geq 0$. The smoothing parameter λ_s was optimized using cross-validation and set to $\lambda_s = 271.9$. The constraint parameter λ_c was set to $\lambda_c = 6000$. For both function estimations, i.e. using P-splines (blue curve) vs. using constraint P-splines (red curve), the number of used splines k was set to $k = 30$.

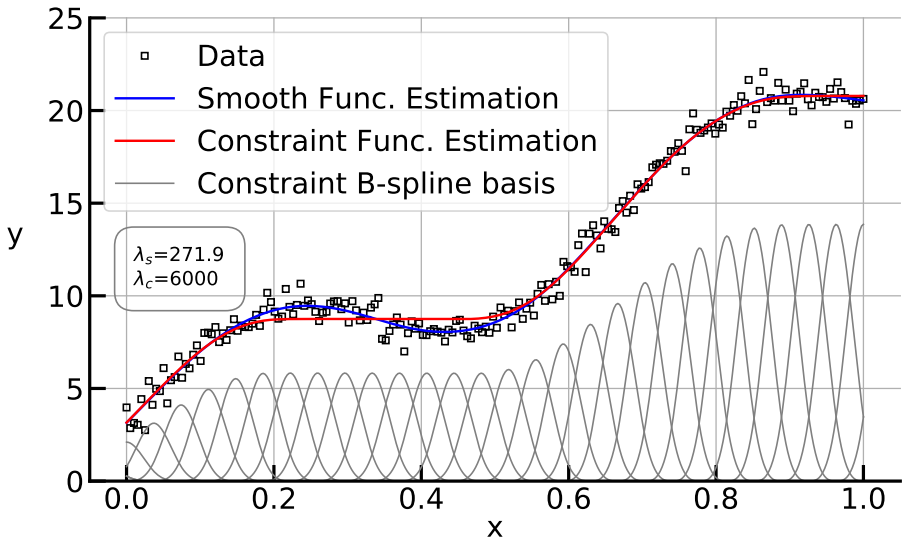

Figure 2.6: Approximation of the noisy data by P-splines and P-splines with the monotonic increasing constraint

The constraint function estimation (red curve in Figure 2.6), follows the monotonicity constraint far better that the smooth function estimation (blue curve in Figure 2.6). For $x < 0.15$ and $x > 0.6$, the two fits are nearly identical, since no constraint violation is present. Note, the entries of the weighting matrix $\mathbf{V}$ in this region are therefore 0 because the constraint is not active. For $x \in [0.15, 0.6]$ the constraint is active. The red fit produces an almost constant line in this region as an optimal solution for the competing goals of data accuracy, smoothness and constraint fidelity.

This shows, that the incorporation of a priori knowledge in the fitting process using P-splines is in principle possible using an appropriate choice of the mapping matrix $\mathbf{D}_c$ and the weighting matrix $\mathbf{V}$ as well as an iterative fitting approach using penalized iteratively reweighted least squares. These matrices are futher discussed in the following chapters.

2.2 User-defined Constraints

As stated before, a priori domain knowledge given in Table 2.2 can be introduced by the choice of the mapping matrix $\mathbf{D}_c$ and the weighting matrix $\mathbf{V}$, cf. (2.11) and (2.13). Now a description of the different matrices, which are used to enforce the a priori known domain behavior, is presented.

2.2.1 Monotonicity Constraint

The mapping matrix $\mathbf{D}_{monoton}$ enforcing monotonic behavior is given by the first order difference operator Δ^1 for equidistant knot placement, cf. 1.64. The corresponding matrix for k splines is given as

$$\mathbf{D}_{monoton} = \begin{pmatrix} -1 & 1 & & \\ & -1 & 1 & \\ & & \ddots & \ddots \end{pmatrix} \in \mathbb{R}^{k-1 \times k}. \tag{2.14}$$

The difference between monotonic increasing and decreasing behavior is controlled by the weighting matrix $\mathbf{V}$. For increasing behavior, the weighting matrix $\mathbf{V}$ is given by the weights v_j according to

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1 \beta_j \geq 0 \\ 1, & \text{if } \Delta^1 \beta_j < 0 \end{cases} \quad \text{for } j = 2, \ldots, k-1. \tag{2.15}$$

For decreasing behavior, the weighting matrix $\mathbf{V}$ is given by the weights v_j according to

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1 \beta_j \leq 0 \\ 1, & \text{if } \Delta^1 \beta_j > 0 \end{cases} \quad \text{for } j = 2, \ldots, k-1. \tag{2.16}$$

This states, that the penalty term $\mathcal{J}_c(\boldsymbol{\beta}; c)$ only contributes if adjacent coefficients β_{j-1} and β_j are increasing or decreasing, respectively. **hofner2011monotonicity eilers2005unimodal**

2.2.2 Curvature Constraint

In the simplest case, the curvature of the function $f(x)$ can either be convex, i.e. $f''(x) \geq 0$, or concave, i.e. $f''(x) \leq 0$. The mapping matrix $\mathbf{D}_{curvature}$ enforcing this behavior can be approximated by the second order difference operator Δ^2 for equidistant knot placement, cf. (1.65). The corresponding matrix for k splines is given as

$$\mathbf{D}_{curvature} = \begin{pmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \end{pmatrix} \in \mathbb{R}^{k-2 \times k}. \tag{2.17}$$

The difference between concave and convex curvature is controlled by the weighting matrix $\mathbf{V}$. For concave curvature, the weighting matrix $\mathbf{V}$ is given by the weights v_j according to

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^2\beta_j \leq 0 \\ 1, & \text{if } \Delta^2\beta_j > 0 \end{cases} \quad \text{for } j = 1, \ldots, k-2. \tag{2.18}$$

For convex curvature, the weighting matrix $\mathbf{V}$ is given by the weights v_j according to

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^2\beta_j \geq 0 \\ 1, & \text{if } \Delta^2\beta_j < 0 \end{cases} \quad \text{for } j = 1, \ldots, k-2. \tag{2.19}$$

Therefore, the penalty term $\mathcal{J}_c(\boldsymbol{\beta}; c)$ in (2.10) or (2.13) only contributes if the second order difference of adjacent coefficients $\boldsymbol{\beta}$ is either positive or negative, respectively. **eilers2005unimodal**

2.2.3 Unimodality Constraint

We assume that there is a peak in the data $\{x^{(i)}, y^{(i)}\}$ and therefore want to constrain the fit to include a peak. The peak constraint is given by the unimodal mapping matrix $D_{unimodal}$ and the peak weighting matrix V. A function $f(x)$ is said to be unimodal if for some value m, it is monotonically increasing for $x \leq m$ and monotonically decreasing for $x \geq m$.

The mapping matrix $\mathbf{D}_{unimodal}$ enforcing unimodal behavior can be constructed using the first order difference operator Δ^1 for equidistant knot placement, cf. (1.64), and is given for k splines as

$$\mathbf{D}_{unimodal} = \begin{pmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{pmatrix} \in \mathbb{R}^{k-1 \times k} \tag{2.20}$$

The weighting matrix $\mathbf{V}$ now has a special structure. First, we construct the B-spline basis using the given data as in Chapter 1.3.1. We then need to find the index j_{peak} of the *peak spline*, which has the maximal value at the peak data point $\max\{f(x^{(i)}) \ \forall \ i\}$, see Figure 2.7. The index j_{peak} is now used as splitting point for the weighting matrix $\mathbf{V}$. All coefficients β_j for $j < j_{peak}$ are constrained to be monotonic increasing, i.e. $\Delta^1\beta_j \geq 0$ for $j = 1, \ldots, j_{peak} - 1$, while all coefficients β_j for $j > j_{peak}$ are constrained to be monotonic decreasing, i.e. $\Delta^1\beta_j \leq 0$ for $j = j_{peak}+1, \ldots, k$. The coefficient $\beta_{j_{peak}}$ stays unconstrained. **eilers2005unimodal**

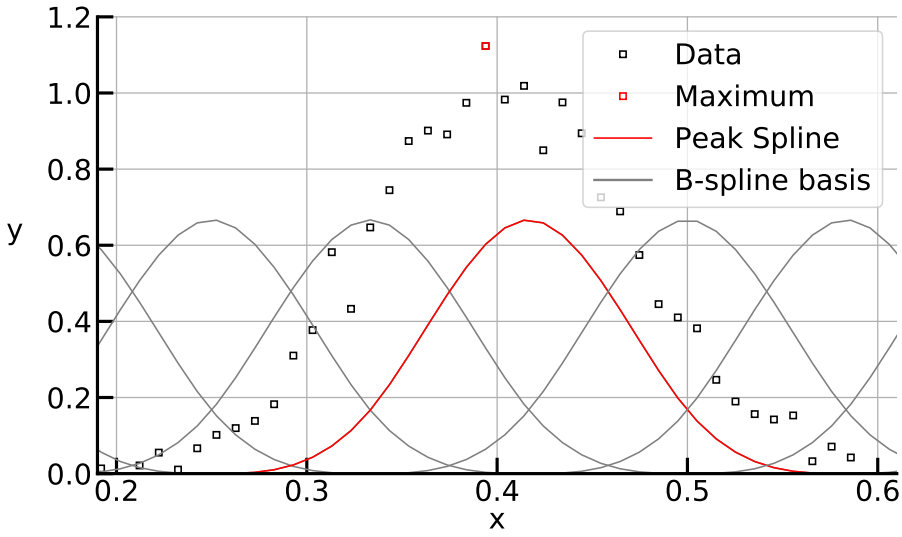

Figure 2.7: Identification of the peak spline based on data

The weights v_j to incorporate the peak constraint have the following structure, i.e.

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1\beta_j \geq 0 \\ 1, & \text{if } \Delta^1\beta_j < 0 \end{cases}, \quad \text{for } j = 2, \ldots, j_{peak} - 1 \tag{2.21}$$

and

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1\beta_j \leq 0 \\ 1, & \text{if } \Delta^1\beta_j > 0 \end{cases}, \quad \text{for } j = j_{peak} + 1, \ldots, k. \tag{2.22}$$

The weight $v_{j_{peak}}$ for the *peak spline* is given by $v_{j_{peak}}(\boldsymbol{\beta}) = 0$.

When assuming a valley in the data, the same approach as above can easily be used by multiplying the data with -1 or by always doing the inverse operation, i.e. finding the index j_{valley} of the *valley spline*, then constraining all splines for $j < j_{valley}$ to be monotonic decreasing, i.e. $\Delta^1\beta_j \leq 0$ for $j = 1, \ldots, j_{valley} - 1$, and all splines for $j > j_{valley}$ to be monotonic increasing, i.e. $\Delta^1\beta_j \geq 0$ for $j = j_{valley} + 1, \ldots, k$. The coefficient $\beta_{j_{valley}}$ stays unconstrained.

The weights v_j to consider a valley constraint are given by

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1\beta_j \leq 0 \\ 1, & \text{if } \Delta^1\beta_j > 0 \end{cases}, \quad \text{for } j = 2, \ldots, j_{valley} - 1 \tag{2.23}$$

and

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1\beta_j \geq 0 \\ 1, & \text{if } \Delta^1\beta_j < 0. \end{cases}, \quad \text{for } j = j_{valley} + 1, \ldots, k. \tag{2.24}$$

The weight $v_{j_{valley}}$ for the *valley spline* is given by $v_{j_{valley}}(\boldsymbol{\beta}) = 0$.

2.2.4 Boundedness Constraint

For certain physical systems, it is known a priori that the measured quantity cannot be smaller than zero, i.e. $f(x) \geq 0$. Using data-driven modeling on noisy data can lead to predictions in the interpolation and extrapolation regime, which may not hold this constraint due to uncertainties captured by the data. It is therefore appropriate to apply the user-defined constraint of boundedness from below.

The user-defined constraint for boundedness from below by $M = 0$ uses as mapping matrix $\mathbf{D}_c$ the B-spline basis matrix $\mathbf{X} \in \mathbb{R}^{n \times k}$. The weighting matrix $\mathbf{V} \in \mathbb{R}^{n \times n}$, with individual weights v_j, is specified as follows:

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } f(x^{(j)}) \geq M \\ 1, & \text{if } f(x^{(j)}) < M \end{cases} \quad \text{for } j = 1, \ldots, n. \tag{2.25}$$

Using different values of M allows us to bound from below from any number M. Switching the comparison operators in (2.25) enables us to bound functions from above.

2.2.5 Jamming Constraint

Jamming the function $f(x)$ by some point $p = \{x^{(jamm)}, y^{(jamm)}\}$ means that the estimated function $f(x^{(jamm)}) \approx y^{(jamm)}$. This can be incorporated using the B-spline basis matrix $\mathbf{X} \in \mathbb{R}^{n \times k}$ as mapping matrix $\mathbf{D}_c$ and a weighting matrix $\mathbf{V} \in \mathbb{R}^{n \times n}$ given by

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } x^{(j)} \neq x^{(jamm)} \\ 1, & \text{if } x^{(j)} = x^{(jamm)} \end{cases} \quad \text{for } j = 1, \ldots, n. \tag{2.26}$$

2.2.6 Penalty Term for Tensor-Product Splines

To extend the framework of mapping matrices to two dimensions and tensor-product splines, we again use the concept of Kronecker products given in Chapter 1.3.1.1. In principle, every possible pair of one dimensional user-defined constraints can be constructed using the approach in Chapter 1.3.1.1, e.g. unimodality in two dimensions would be obtained using the unimodal mapping matrix depicted above for each dimension. We then also need to include the constraint specific weight matrices $\mathbf{V}$.

The penalty term for the constraint given by c_1 for dimension 1 and c_2 for dimension 2 then has the form

$$\mathcal{J}_c(\boldsymbol{\beta}; c) = \boldsymbol{\beta}^{\mathrm{T}} \left[\mathbf{I}^2 \otimes \mathbf{K}_{c_1} + \mathbf{K}_{c_2} \otimes \mathbf{I}^1 \right] \boldsymbol{\beta} \tag{2.27}$$

with the respective penalty matrices $\mathbf{K}_{c_1} = \mathbf{D}_{c_1}^{\mathrm{T}} \mathbf{V}_1 \mathbf{D}_{c_1}$ for dimension x_1 and $\mathbf{K}_{c_2} = \mathbf{D}_{c_2}^{\mathrm{T}} \mathbf{V}_2 \mathbf{D}_{c_2}$ for dimension x_2 using the weighting matrices $\mathbf{V}_1$ and $\mathbf{V}_2$, the mapping matrices $\mathbf{D}_{c_1}$ and $\mathbf{D}_{c_2}$ and the identity matrices $\mathbf{I}^1$ and $\mathbf{I}^2$ for the respective dimension.

2.3 n-d Constraint Function Estimation

The extension from one input to multiple input dimensions uses the concept of additive models given in Chapter 1.3.1.2. Given input data $\{x_1^{(i)}, \ldots, x_q^{(i)}; y^{(i)}\}$, $i = 1, \ldots, n$ and q as the number of inputs, the combined model using all available B-splines and tensor-product splines is given, cf. (1.61), as

$$y = f(x_1, ..., x_q) = \sum_{j=1}^{q} s_j(x_j) + \sum_{j=1}^{q-1} \sum_{r>j}^{q} t_{j,r}(x_j, x_r) \tag{2.28}$$

where $s_j(x_j)$ is the B-spline estimate given by $s_j(x_j) = \mathbf{X}_{s_j} \boldsymbol{\beta}_{s_j}$ and $t_{j,r}(x_j, x_r)$ is the tensor-product estimate is given by $t_{j,r}(x_j, x_r) = \mathbf{X}_{t_{j,r}} \boldsymbol{\beta}_{t_{j,r}}$. The number of individual estimates is given by

$$n_{total} = q + \frac{q(q-1)}{2}. \tag{2.29}$$

The constrained penalized least squares objective function for additive models can now be written similar to (2.10) as

$$Q_6(\mathbf{y}, \boldsymbol{\beta}) = Q_1(\mathbf{y}, \boldsymbol{\beta}) + \boldsymbol{\lambda}_s^{\mathrm{T}} \mathcal{J}_s(\boldsymbol{\beta}; \mathbf{d}) + \boldsymbol{\lambda}_c^{\mathrm{T}} \mathcal{J}_c(\boldsymbol{\beta}; \mathbf{c}). \tag{2.30}$$

with $\boldsymbol{\lambda}_s \in \mathbb{R}^{n_{total}}$ and $\boldsymbol{\lambda}_c \in \mathbb{R}^{n_{total}}$ defined as vectors with one value of smoothness and constraint parameter for each individual estimate, respectively.

We now need to specify the three parts of the objective function in (2.30).

2.3.1 Data Term

Assuming the use of k splines for the B-spline estimates and k^2 splines for the tensor-product estimates, the total number of coefficients to be determined is given by

$$k_{total} = qk + \frac{q(q-1)}{2} k^2. \tag{2.31}$$

Since all B-spline and tensor-product spline models follow a linear model structure, see Chapter 1.3.1 and 1.3.1.1, we can combine them into one large model, cf. (1.62), given by

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} \tag{2.32}$$

where the matrix $\mathbf{X} \in \mathbb{R}^{n \times k_{total}}$ is given in (1.62) as horizontal concatenation of the individual bases and the combined coefficient vector $\boldsymbol{\beta} \in \mathbb{R}^{k_{total}}$ is given in (1.62) by a vertical concatenation of the individual coefficient vectors.

The data term $Q_1(\mathbf{y}, \boldsymbol{\beta})$ in the constrained penalized least squares objective function given in (2.30) can now be evaluated using arbitrary input dimensions.

2.3.2 Smoothness Term

The combined smoothness penalty term $\mathcal{J}_s(\boldsymbol{\beta}; \mathbf{d}) \in \mathbb{R}^{n_{total}}$ is then given as

$$\mathcal{J}_s(\boldsymbol{\beta}; \mathbf{d}) = \begin{pmatrix} \mathcal{J}_{s_1}(\boldsymbol{\beta}_{s_1}; d_{s_1}) \\ \vdots \\ \mathcal{J}_{s_q}(\boldsymbol{\beta}_{s_q}; d_{s_q}) \\ \mathcal{J}_{t_{1,2}}(\boldsymbol{\beta}_{t_{1,2}}; d_{t_{1,2}}) \\ \vdots \\ \mathcal{J}_{t_{q-1,q}}(\boldsymbol{\beta}_{t_{q-1,q}}; d_{t_{q-1,q}}) \end{pmatrix} \tag{2.33}$$

with $\mathcal{J}_e(\boldsymbol{\beta}_e; d_e) = \boldsymbol{\beta}_e^{\mathrm{T}} \mathbf{D}_{d_e}^{\mathrm{T}} \mathbf{D}_{d_e} \boldsymbol{\beta}_e$ determining the smoothness penalty term using the coefficients $\boldsymbol{\beta}_e$ and mapping matrix $\mathbf{D}_{d_e}$, see Chapter 1.3.2 and Chapter 1.3.1.1, for each estimate $e \in \{s_1, \ldots, s_q, t_{1,2}, \ldots, t_{q-1,q}\}$. The vector $\mathbf{d} \in \mathbb{R}^{n_{total}}$ consists of the orders d_e determining the mapping matrix $\mathbf{D}_{d_e}$ of the smoothness constraint for each individual estimate e.

2.3.3 Constraint Term

The combined constraint penalty term $\mathcal{J}_c(\boldsymbol{\beta}; \mathbf{c}) \in \mathbb{R}^{n_{total}}$ is then given as

$$\mathcal{J}_c(\boldsymbol{\beta}; \mathbf{c}) = \begin{pmatrix} \mathcal{J}_{s_1}(\boldsymbol{\beta}_{s_1}; c_{s_1}) \\ \vdots \\ \mathcal{J}_{s_q}(\boldsymbol{\beta}_{s_q}; c_{s_q}) \\ \mathcal{J}_{t_{1,2}}(\boldsymbol{\beta}_{t_{1,2}}; c_{t_{1,2}}) \\ \vdots \\ \mathcal{J}_{t_{q-1,q}}(\boldsymbol{\beta}_{t_{q-1,q}}; c_{t_{q-1,q}}) \end{pmatrix} \tag{2.34}$$

with $\mathcal{J}_e(\boldsymbol{\beta}_e; c_e) = \boldsymbol{\beta}_e^{\mathrm{T}} \mathbf{D}_{c_e}^{\mathrm{T}} \mathbf{V}_{c_e} \mathbf{D}_{c_e} \boldsymbol{\beta}_e$ determining the constraint penalty term using the coefficients $\boldsymbol{\beta}_e$, the mapping matrix $\mathbf{D}_{c_e}$ and the weighting matrix $\mathbf{V}_e$ for each estimate $e \in \{s_1, \ldots, s_q, t_{1,2}, \ldots, t_{q-1,q}\}$, see Chapter (2.2). The vector $\mathbf{c} \in \mathbb{R}^{n_{total}}$ consists of the constraint type c_e, e.g. monoton increasing, determining the mapping matrix $\mathbf{D}_{c_e}$ for each individual estimate e.

The objective function (2.30) is then optimized, i.e.

$$\hat{\boldsymbol{\beta}}_{PLS,c,nd} = \arg\min_{\boldsymbol{\beta}} Q_6(\mathbf{y}, \boldsymbol{\beta}), \tag{2.35}$$

using the penalized iteratively reweighted least squares algorithm, cf. (2.13), to obtain the coefficients $\hat{\boldsymbol{\beta}}_{PLS,c,nd}$ as

$$\hat{\boldsymbol{\beta}}_{PLS,c,nd} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \mathbf{K}_s + \mathbf{K}_c)^{-1} \mathbf{X}^{\mathrm{T}} \mathbf{y}. \tag{2.36}$$

In (2.36), $\mathbf{X} \in \mathbb{R}^{n \times k_{total}}$ is the combined basis matrix, cf. (1.62), $\mathbf{K}_s \in \mathbb{R}^{k_{total} \times k_{total}}$ is the combined smoothness matrix given as

$$\mathbf{K}_s = \begin{bmatrix} \lambda_{s_1} \mathbf{D}_{d_{s_1}}^{\mathrm{T}} \mathbf{D}_{d_{s_1}} & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \ddots & & & & & \\ 0 & & \lambda_{s_q} \mathbf{D}_{d_{s_q}}^{\mathrm{T}} \mathbf{D}_{d_{s_q}} & 0 & & & \\ 0 & & 0 & \lambda_{s_{1,2}} \mathbf{D}_{d_{t_{1,2}}}^{\mathrm{T}} \mathbf{D}_{d_{t_{1,2}}} & 0 & \cdots & 0 \\ & & & 0 & \ddots & & \\ 0 & & & 0 & & \lambda_{s_{q-1,q}} \mathbf{D}_{d_{t_{q-1,q}}}^{\mathrm{T}} \mathbf{D}_{d_{t_{q-1,q}}} & 0 \\ 0 & & & 0 & & 0 & \end{bmatrix} \tag{2.37}$$

and $\mathbf{K}_c \in \mathbb{R}^{k_{total} \times k_{total}}$ is the combined constraint matrix as

$$\mathbf{K}_c = \begin{pmatrix} \lambda_{c_1}\mathbf{D}_{c_{s_1}}^{\mathrm{T}}\mathbf{V}_{c_{s_1}}\mathbf{D}_{c_{s_1}} & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \ddots & & & & & \\ 0 & & \lambda_{c_q}\mathbf{D}_{c_{s_q}}^{\mathrm{T}}\mathbf{V}_{c_{s_q}}\mathbf{D}_{c_{s_q}} & 0 & & & \\ 0 & & 0 & \lambda_{c_{1,2}}\mathbf{D}_{c_{t_{1,2}}}^{\mathrm{T}}\mathbf{V}_{c_{t_{1,2}}}\mathbf{D}_{c_{t_{1,2}}} & 0 & & \\ & & & 0 & \ddots & & \\ 0 & & & & & \lambda_{c_{q-1,q}}\mathbf{D}_{c_{t_{q-1,q}}}^{\mathrm{T}}\mathbf{V}_{c_{t_{q-1,q}}}\mathbf{D}_{c_{t_{q-1,q}}} \end{pmatrix}. \tag{2.38}$$

2.3.4 2-d Example

As example for the n-d constraint function estimation, we take a look at the function

$$f(x_1, x_2) = 2\exp\Big(-\frac{(x_1 - 0.25)^2}{0.08}\Big) + x_2^2 + \eta \tag{2.39}$$

for $x_1 \in [0,1]$ and $x_2 \in [0,1]$ and random Gaussian noise η with $\sigma_{noise} = 0.1$. Therefore we expect a peak in dimension x_1 as well as increasing behavior for dimension x_2, see Figure 2.8. The user-defined constraints are therefore $c_1 =$ unimodal and $c_2 =$ monotonic increasing Using this knowledge, we create a model with the following characteristics:

- B-spline smooth $s_1(x_1)$: $k_{x_1} = 50$, $c = c_1$, $\lambda_s = 1$ and $\lambda_c = 6000$
- B-spline smooth $s_2(x_2)$: $k_{x_2} = 50$, $c = c_2$, $\lambda_s = 1$ and $\lambda_c = 6000$

The fit for this model as well as the individual estimates $s_1(x_1)$ and $s_2(x_2)$ are shown in Figure 2.8. The model fits the data quite well and holds the specified constraints for the individual dimensions.

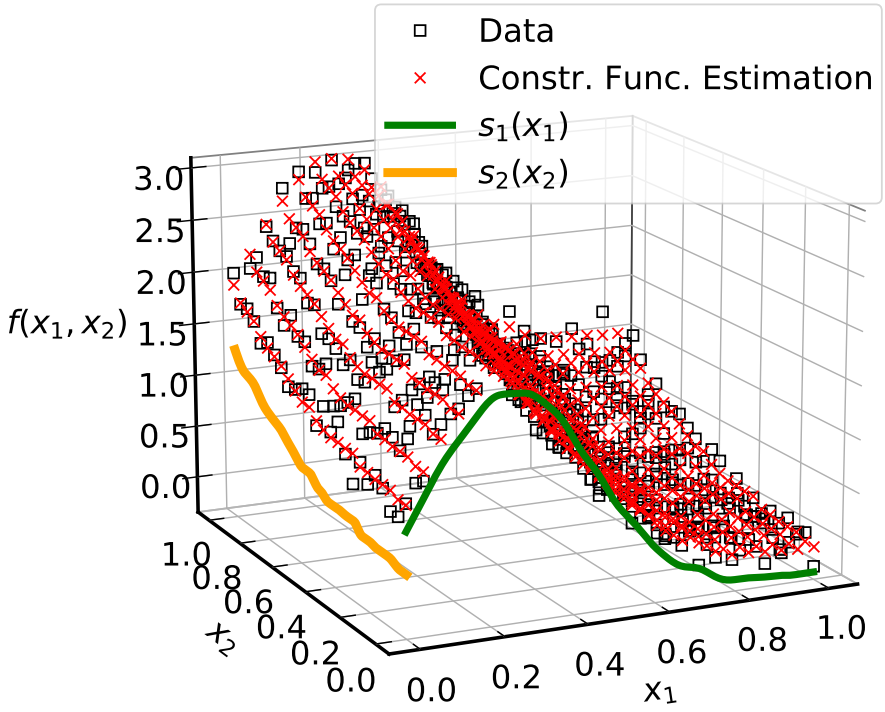

Figure 2.8: 2-d test function for n-d constrained function estimation

Bibliography

bishop2006patternRecognition Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

blobel2013statistische Volker Blobel and Erich Lohrmann. *Statistische und numerische Methoden der Datenanalyse*. Springer-Verlag, 2013.

deBoor1978practicalGuideToSplines Carl De Boor et al. *A practical guide to splines*. Vol. 27. springer-verlag New York, 1978.

eilers1996flexible Paul HC Eilers and Brian D Marx. "Flexible smoothing with B-splines and penalties". In: *Statistical science* (1996), pp. 89–102.

eilers2005unimodal Paul HC Eilers. "Unimodal smoothing". In: *Journal of Chemometrics: A Journal of the Chemometrics Society* 19.5-7 (2005), pp. 317–328.

eubank1990regressionsplines Randall L Eubank and Clifford H Spiegelman. "Testing the goodness of fit of a linear model via nonparametric regression techniques". In: *Journal of the American Statistical Association* 85.410 (1990), pp. 387–392.

fahrmeir2004penalized Ludwig Fahrmeir, Thomas Kneib, and Stefan Lang. "Penalized structured additive regression for space-time data: a Bayesian perspective". In: *Statistica Sinica* (2004), pp. 731–761.

fahrmeir2007regression Ludwig Fahrmeir et al. *Regression*. Springer, 2007.

ferziger2008numerische Joel H Ferziger and Milovan Peric. *Numerische Strömungsmechanik*. Springer-Verlag, 2008.

friedman2001elements Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. 10. Springer series in statistics New York, 2001.

golub1979 Gene H Golub, Michael Heath, and Grace Wahba. "Generalized cross-validation as a method for choosing a good ridge parameter". In: *Technometrics* 21.2 (1979), pp. 215–223.

hoerl1970ridge Arthur E Hoerl and Robert W Kennard. "Ridge regression: Biased estimation for nonorthogonal problems". In: *Technometrics* 12.1 (1970), pp. 55–67.

hofner2011monotonicity Benjamin Hofner, Jörg Müller, and Torsten Hothorn. "Monotonicity-constrained species distribution models". In: *Ecology* 92.10 (2011), pp. 1895–1901.

luenberger1984linear David G Luenberger, Yinyu Ye, et al. *Linear and nonlinear programming*. Vol. 2. Springer, 1984.

osullivan1986penalties Finbarr O'Sullivan. "A statistical perspective on ill-posed inverse problems". In: *Statistical science* (1986), pp. 502–518.

sammut2011 Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

wood2017generalized Simon N Wood. *Generalized additive models: an introduction with R*. CRC press, 2017.