Chapter 3 - Solution Approach DRAFT

Weber Jakob

October 16, 2020

Contents

1 Introduction

We are now going to use the theory defined in Chapter 2 to estimate smooth, constraint functions. An overview of the different problems is given in Table 1. At first, we are using B-splines as basis functions for the estimation of the one dimensional, unknown function $y = f(x)$ for some data $\{x^{(i)}, y^{(i)}\}$ for $i = 1, \ldots, n$. Next, we use the concept of P-splines 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 1: Problem overview

The a priori knowledge can be incorporated using different types of constraints. The possible constraints are listed in Table 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: Overview of the possible constraints

To test the algorithm and the incorporation of a priori knowledge, we use 200 noisy samples, see Figure 1, from the one dimensonal function given in (1) and the monotonicity constraint.

$$f(x) = 3\sin(3\pi x) + 17x + 3 \tag{1}$$

The function is purposely chosen such that the samples violate the constraint for some x.

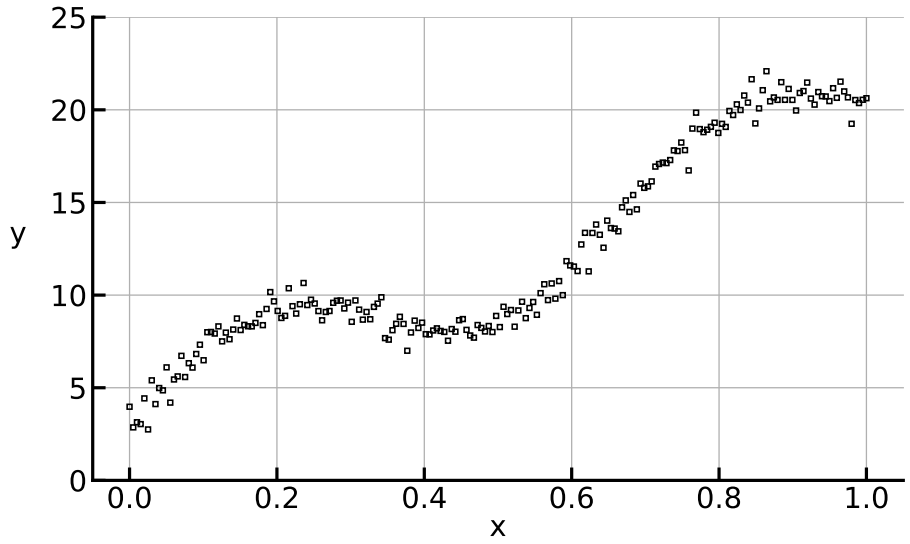

Figure 1: Noisy samples from Function 1

2 Function Estimation

2.1 1d Function Estimation

The goal is to model given data

$$\{x^{(i)}, y^{(i)}\}, \quad i = 1, \ldots, n \tag{2}$$

using B-splines as basis functions. Therefore we want to estimate the unknown function $y = f(x)$ which can be represented as a linear combination of k B-spline basis functions x_k as

$$y = f(x) = \sum_{j=1}^{k} \beta_k x_k(x) = \boldsymbol{X}\boldsymbol{\beta}, \tag{3}$$

where $\boldsymbol{X} \in \mathbb{R}^{n \times k}$ is the B-spline basis matrix 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

$$Q_1(\boldsymbol{y}, \boldsymbol{\beta}) = \|\boldsymbol{y} - f(\boldsymbol{x})\|^2 = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|^2 \tag{4}$$

The coefficients are determined by minimizing the objective function given in (4) with respect to $\boldsymbol{\beta}$, i.e.

$$\hat{\boldsymbol{\beta}}_{LS} = \arg\min_{\boldsymbol{\beta}} Q_1(\boldsymbol{y}, \boldsymbol{\beta}). \tag{5}$$

Using the least squares algorithm LS, this yields

$$\hat{\boldsymbol{\beta}}_{LS} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y}. \tag{6}$$

Figure 2 shows a B-spline model using $k = 10$ splines on an equidistant grid approximating the noisy data as well as the individual B-spline basis functions multiplied with the corresponding, estimated coefficients $\hat{\boldsymbol{\beta}}_{LS}$.

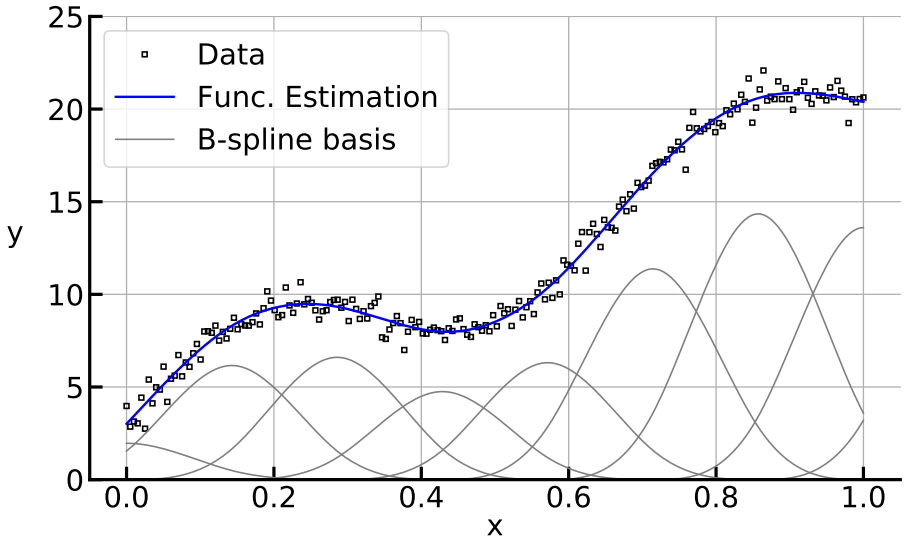

Figure 2: Approximation of noisy data by B-splines without constraints

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 bias-variance dilemma and depicted in Figure 3. **sammut2011**

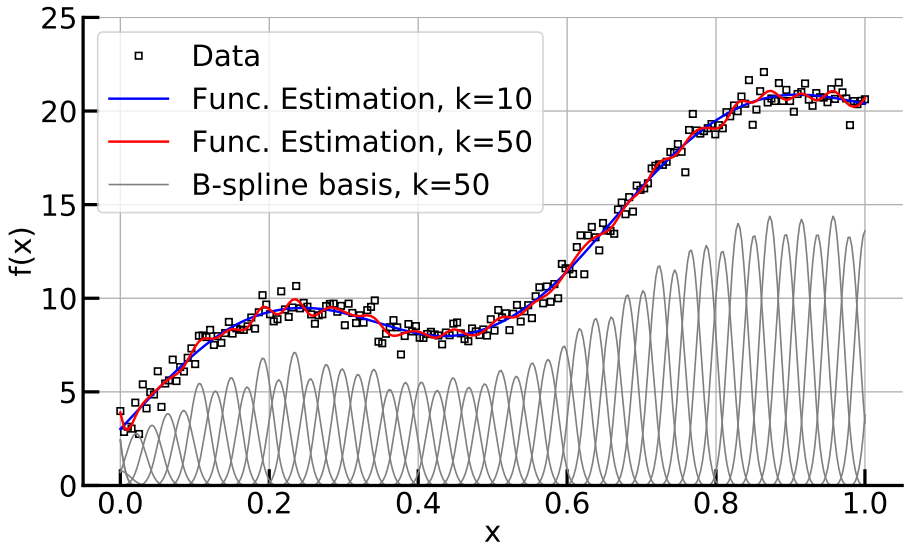

Figure 3: Approximation of noisy data by 10 and 50 B-splines without constraints

2.2 1d Smooth Function Estimation

To overcome this problem, the second derivative of the estimated function $f(x)$, i.e. $f''(x) = \sum_{j=1}^{k} \beta_k x_k''(x)$, has to be penalized to realize a smoother estimate. Eilers and Marx have introduced the so-called P-splines. **eilers1996flexible** Therefore, the objective function (4) is extended by an additional term considering the smoothness,

$$Q_2(\boldsymbol{y}, \boldsymbol{\beta}) = Q_1(\boldsymbol{y}, \boldsymbol{\beta}) + \lambda_s \mathcal{J}_s(\boldsymbol{\beta}; d) = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|^2 + \lambda_s \boldsymbol{\beta}^T \boldsymbol{D}_d^T \boldsymbol{D}_d^T \boldsymbol{\beta}, \quad (7)$$

with the smoothing parameter λ_s and an appropriate mapping matrix $\boldsymbol{D}_d$ 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 of the difference operator of order d.

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

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

and in matrix notation for order $d = 1$

$$\boldsymbol{D}_1 = \begin{pmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{pmatrix} \in \mathbb{R}^{k-1 \times k}$$

and order $d = 2$

$$\boldsymbol{D}_2 = \begin{pmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \\ & & & 1 & -2 & 1 \end{pmatrix} \in \mathbb{R}^{k-2 \times k}.$$

For non-equidistant knot placement, the difference operator Δ^d needs to include some kind of weighting determining the influence of the individual splines. A possible approach can be found in **ferziger2008numerische**.

By minimizing the objective function (7), i.e.

$$\hat{\boldsymbol{\beta}}_{PLS} = \arg\min_{\boldsymbol{\beta}} Q_2(\boldsymbol{y}, \boldsymbol{\beta}), \tag{8}$$

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

$$\hat{\boldsymbol{\beta}}_{PLS} = (\boldsymbol{X}^T \boldsymbol{X} + \lambda_s \boldsymbol{D}_d^T \boldsymbol{D}_d)^{-1} \boldsymbol{X}^T \boldsymbol{y}. \tag{9}$$

The smoothing parameter λ_s plays a critical role and can be optimized using the information criteria specified in Chapter Model Selection Criteria, e.g. AIC and BIC, or by using cross-validation techniques. **fahrmeir2013regression** For small values $\lambda_s \to 0$, the penalized least squares estimate $\hat{\boldsymbol{\beta}}_{PLS}$ approaches the least squares estimate $\hat{\boldsymbol{\beta}}_{LS}$, 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 is leading to a linear function, while using $d = 1$ would lead to a constant function. **fahrmeir2013regression**

Figure 4 shows the behavior of P-splines 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 becomes smaller. For very large values of λ, the estimate approaches a straight line.

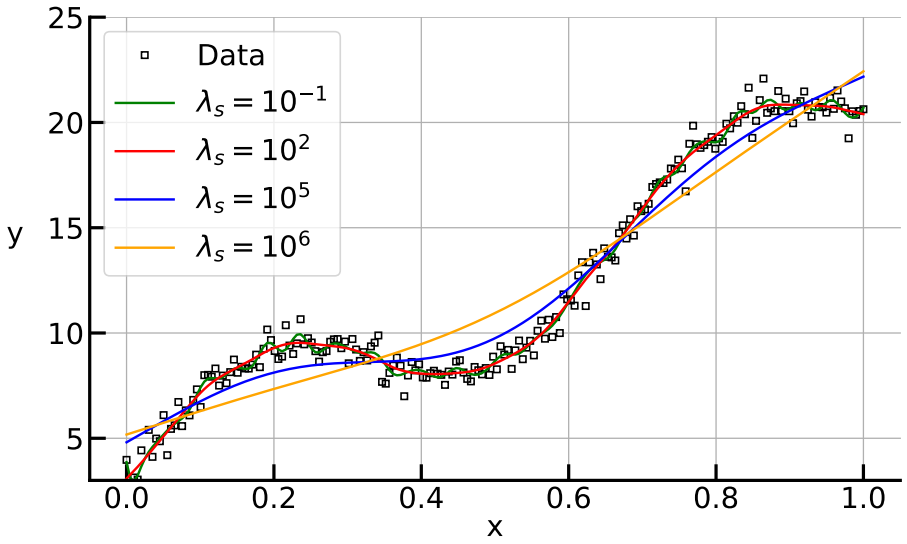

Figure 4: Smooth function estimation for different λ_s

2.3 1d Constraint Function Estimation

A priori knowledge can now be systematically incorporated by the extension of the objective function (7) using an additional term representing the user-defined constraint. 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

$$Q_3(\boldsymbol{y}, \boldsymbol{\beta}) = Q_1(\boldsymbol{y}, \boldsymbol{\beta}) + \lambda_s \mathcal{J}_s(\boldsymbol{\beta}; d) + \lambda_c \mathcal{J}_c(\boldsymbol{\beta}; c) \tag{10}$$

with the corresponding constraint parameter λ_c, which determines the influence of the 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 and boundedness can be modeled as

$$\mathcal{J}_c(\boldsymbol{\beta}; c) = \boldsymbol{\beta}^T \boldsymbol{D}_c^T \boldsymbol{V} \boldsymbol{D}_c \boldsymbol{\beta} \tag{11}$$

with the mapping matrix $\boldsymbol{D}_c$ and the diagonal weighting matrix $\boldsymbol{V} := \boldsymbol{V}(\boldsymbol{\beta}; c)$ capturing if the constraint c is active or inactive. These matrices will further be defined in Chapter 3.

The constraint of jamming the fit by some critical points can be incorporated using the weighted least squares approach and large weights for the critical points. **strutz2016data**

By minimizing the objective function (10), i.e.

$$\hat{\boldsymbol{\beta}}_{PLS,c} = \arg\min_{\boldsymbol{\beta}} Q_3(\boldsymbol{y}, \boldsymbol{\beta}), \tag{12}$$

the constraint penalized least-squares estimate can be given as

$$\hat{\boldsymbol{\beta}}_{PLS,c} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda_s \boldsymbol{D}_d^T \boldsymbol{D}_d + \lambda_c \boldsymbol{D}_c^T \boldsymbol{V} \boldsymbol{D}_c)^{-1} \boldsymbol{X}^T \boldsymbol{y}. \tag{13}$$

Note, (13) is a nonlinear equation because the matrix $\boldsymbol{V}$ depends on $\boldsymbol{\beta}$. Thus, it has to be solved iteratively. The algorithm is shown in Figure 5.

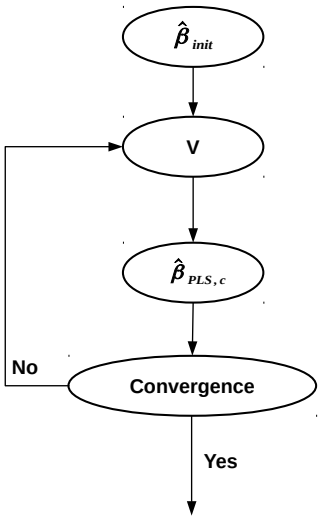

Figure 5: Penalized iteratively reweighted least squares algorithm

The initial estimate $\hat{\boldsymbol{\beta}}_{init}$ needed to compute the weighting matrix $\boldsymbol{V}$ is given by the least squares estimate $\hat{\boldsymbol{\beta}}_{LS}$. Now the calculation of the constrained least squares estimate $\hat{\boldsymbol{\beta}}_{PLS,c}$ and the calculation of the weighting matrix $\boldsymbol{V}$ is performed until no more changes in the weighting matrix $\boldsymbol{V}$ appear. This scheme is called penalized iteratively reweighted least squares and is abbreviated by PIRLS. **hofner2011monotonicity**

Figure 6 shows an example, where noisy data is approximated by considering the monotonicity constraint. The smoothing parameter was optimized using cross-validation, set to $\lambda_s = 271.9$ and used for both estimates. The constraint parameter was set to $\lambda_c = 6000$. For both function estimations, the number of used splines k was set to 30.

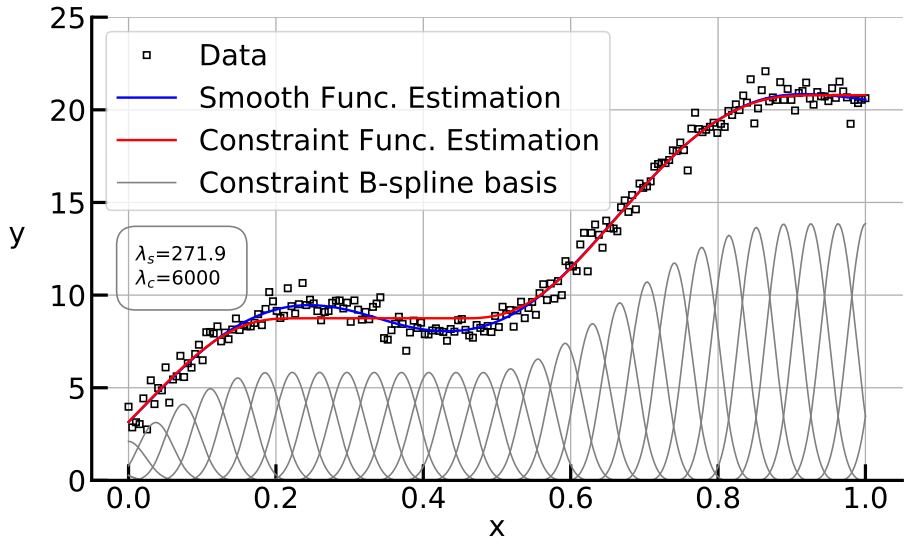

Figure 6: Monotonic constrained 1d function estimation

The red, constraint function estimation follows the monotonicity constraint far better that the blue, smooth function estimation. For $x \in [0, 0.2]$ and $x > 0.6$, the two fits are identical, since no constraint violation is present. The weighting matrix $\boldsymbol{V}$ is therefore 0 everywhere and the constraint is not active. For $x \in [0.15, 0.6]$ the constraint is active. The red fit produces a almost constant line as 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 B-splines is in principle possible using a appropriate choice of the mapping matrix $\boldsymbol{D}_c$ and the weighting matrix $\boldsymbol{V}$ as well as an iterative fitting approach using penalized iteratively reweighted least squares.

3 User-defined Constraints

As stated before, a priori knowledge can be introduced by the choice of the mapping matrix $\boldsymbol{D}_c$ and the weighting matrix $\boldsymbol{V}$. It now follows a description of the different matrices, which are used to enforce a priori known behavior.

3.1 Monotonicity Constraint

The mapping matrix $\boldsymbol{D}_c^{monoton}$ enforcing monotonic behavior follows from the first order difference operator Δ^1. The corresponding matrix for k splines is given as

$$\boldsymbol{D}_c^{monoton} = \begin{pmatrix} -1 & 1 & & \\ & -1 & 1 & \\ & & \ddots & \ddots \end{pmatrix} \in \mathbb{R}^{k-1 \times k}. \tag{14}$$

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

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1 \beta_j \geq 0 \\ 1, & \text{if } \Delta^1 \beta_j < 0. \end{cases} \tag{15}$$

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

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1 \beta_j \leq 0 \\ 1, & \text{if } \Delta^1 \beta_j > 0. \end{cases} \tag{16}$$

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**

3.2 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 $\boldsymbol{D}_c^{curvature}$ enforcing this behavior can be approximated by the second order difference operator Δ^2. The corresponding matrix for k splines is given as

$$\boldsymbol{D}_c^{curvature} = \begin{pmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \end{pmatrix} \in \mathbb{R}^{k-2 \times k}. \tag{17}$$

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

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^2 \beta_j \leq 0 \\ 1, & \text{if } \Delta^2 \beta_j > 0. \end{cases} \tag{18}$$

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

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^2 \beta_j \geq 0 \\ 1, & \text{if } \Delta^2 \beta_j < 0. \end{cases} \tag{19}$$

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

3.3 Unimodality Constraint

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 $x \geq m$. 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 mapping matrix $\boldsymbol{D}_c^{unimodal}$ enforcing unimodal behavior can be constructed using the first order difference operator Δ^1. The weighting matrix $\boldsymbol{V}$ now has a special structure.

First, we construct the B-spline basis according to the given data. We then need to find the index j_{peak} of the *peak spline*, which has the maximal value at the peak data point, see Figure 7. The index j_{peak} is now used as splitting point for the weighting matrix $\boldsymbol{V}$. 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**

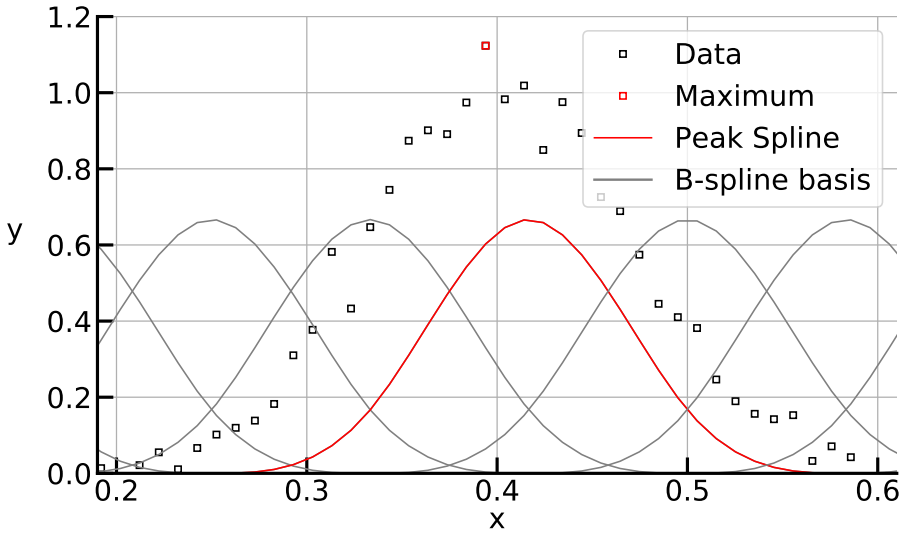

Figure 7: Identification of the peak spline

The corresponding matrix for k splines is given as

$$\boldsymbol{D}_c^{unimodal} = \begin{pmatrix} -1 & 1 & & & & & & \\ & \ddots & \ddots & & & & & \\ & & -1 & 1 & & & & \\ & & & 0 & 0 & & & \\ & & & & -1 & 1 & & \\ & & & & & \ddots & \ddots & \\ & & & & & & -1 & 1 \end{pmatrix} \in \mathbb{R}^{k-1 \times k} \tag{20}$$

The weights v_j to incorporate the peak constraint have the following structure:

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1\beta_j \geq 0 \\ 1, & \text{if } \Delta^1\beta_j < 0. \end{cases}, \quad \text{if } j < j_{peak} \tag{21}$$

and

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1\beta_j \leq 0 \\ 1, & \text{if } \Delta^1\beta_j > 0. \end{cases}, \quad \text{if } j > j_{peak} \tag{22}$$

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

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1\beta_j \leq 0 \\ 1, & \text{if } \Delta^1\beta_j > 0 \end{cases}, \quad \text{if } j < j_{valley} \tag{23}$$

and

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } \Delta^1\beta_j \geq 0 \\ 1, & \text{if } \Delta^1\beta_j < 0. \end{cases}, \quad \text{if } j > j_{valley}. \tag{24}$$

3.4 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. 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 a weighting matrix $\boldsymbol{V} \in \mathbb{R}^{n \times n}$, with individual weights v_j specified as follows:

$$v_j(\boldsymbol{\beta}) = \begin{cases} 0, & \text{if } f(x^{(j)}) \geq M \\ 1, & \text{if } f(x^{(j)}) < M \end{cases} \tag{25}$$

The constrained,penalized least squares objective function is then of the form

$$Q_4(\boldsymbol{y}, \boldsymbol{\beta}) = Q_1(\boldsymbol{y}, \boldsymbol{\beta}) + \lambda_s \mathcal{J}_s(\boldsymbol{\beta}; d) + \lambda_{bounded} \mathcal{J}_{bounded}(\boldsymbol{\beta}) \tag{26}$$

where

$$\mathcal{J}_{bounded} = \boldsymbol{\beta}^T \boldsymbol{X}^T \boldsymbol{V} \boldsymbol{X} \boldsymbol{\beta} \tag{27}$$

is the penalty term specifying boundedness from below by M and $\lambda_{bounded}$ is the constraint parameter, which is set multiple orders of magnitude higher than the smoothness parameter λ_s to enforce the constraint. The matrix $\boldsymbol{X}$ is the B-spline basis.

Using different values of M allows us to bound from below from any number M. Switching the comparison operators in (25) enables us to bound functions from above.

3.5 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 weighted least squares approach. **strutz2016data** We can therefore use a priori knowledge that the estimated function should be in the proximity of $y^{(jamm)}$ for $x = x^{(jamm)}$.

The constraint, weighted objective function is then of the form

$$Q_5(\boldsymbol{y}, \boldsymbol{\beta}) = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_W^2 \tag{28}$$

using the matrix norm $\|.\|_W$ induced by the weighting matrix $\boldsymbol{W} \in \mathbb{R}^{n \times n}$. The weights w_i for $i = 1, \ldots, n$ are equal to 1, except for the point p for which the weight w_p is set to some high value, e.g. $w_p = 1000$.

By minimizing the objective function in (28), i.e.

$$\hat{\boldsymbol{\beta}}_{WLS} = \arg\min_{\boldsymbol{\beta}} Q_5(\boldsymbol{y}, \boldsymbol{\beta}), \tag{29}$$

we obtain the weighted least squares estimate

$$\hat{\boldsymbol{\beta}}_{WLS} = (\boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{y}. \tag{30}$$

4 n-d Constraint Function Estimation

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

$$y = f(x_1, ..., x_p) = \sum_{j=1}^{p} s_j(x_j) + \sum_{j=1}^{p-1} \sum_{l>j}^{p} t_{j,l}(x_j, x_l) \tag{31}$$

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

$$n_{total} = p + \frac{p(p-1)}{2}. \tag{32}$$

The constrained penalized least squares objective function for additive models can now be written similar to (10) as

$$Q_6(\boldsymbol{y}, \boldsymbol{\beta}) = Q_1(\boldsymbol{y}, \boldsymbol{\beta}) + \boldsymbol{\lambda}_s^T \boldsymbol{\mathcal{J}}_s(\boldsymbol{\beta}; \boldsymbol{d}) + \boldsymbol{\lambda}_c^T \boldsymbol{\mathcal{J}}_c(\boldsymbol{\beta}; \boldsymbol{c}). \tag{33}$$

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 estimate, respectively.

The objective function (33) is then optimized, i.e.

$$\hat{\boldsymbol{\beta}}_{PLS,c,nd} = \arg\min_{\boldsymbol{\beta}} Q_5(\boldsymbol{y}, \boldsymbol{\beta}), \tag{34}$$

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

We now need to specify the three parts of the objective function in (33).

4.1 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

$$k_{total} = pk + \frac{p(p-1)}{2}k^2. \tag{35}$$

Since all B-spline and tensor-product spline models follow a linear model structure, we can combine them into one large model given by

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} \tag{36}$$

where the matrix $\boldsymbol{X} \in \mathbb{R}^{n \times k_{total}}$ is given by a horizontal concatenation of the individual bases and the combined coefficient vector $\boldsymbol{\beta} \in \mathbb{R}^{k_{total}}$ is given by a vertical concatenation of the individual coefficient vectors. The model has then the following form

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{X}_{s_1} \dots \boldsymbol{X}_{s_p} & \boldsymbol{X}_{t_{1,2}} \dots \boldsymbol{X}_{t_{p-1,p}} \end{pmatrix} \begin{pmatrix} \boldsymbol{\beta}_{s_1} \\ \vdots \\ \boldsymbol{\beta}_{s_p} \\ \boldsymbol{\beta}_{t_{1,2}} \\ \vdots \\ \boldsymbol{\beta}_{t_{p-1,p}} \end{pmatrix}. \tag{37}$$

The data term $Q_1(\boldsymbol{y}, \boldsymbol{\beta})$ in the constrained penalized least squares objective function given in (33) can now be evaluated using arbitrary input dimensions.

4.2 Smoothness Term

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

$$\boldsymbol{\mathcal{J}}_s(\boldsymbol{\beta}; \boldsymbol{d}) = \begin{pmatrix} \mathcal{J}_{s_1}(\boldsymbol{\beta}_{s_1}; d_{s_1}) \\ \vdots \\ \mathcal{J}_{s_p}(\boldsymbol{\beta}_{s_p}; d_{s_p}) \\ \mathcal{J}_{t_{1,2}}(\boldsymbol{\beta}_{t_{1,2}}; d_{t_{1,2}}) \\ \vdots \\ \mathcal{J}_{t_{p-1,p}}(\boldsymbol{\beta}_{t_{p-1,p}}; d_{t_{p-1,p}}) \end{pmatrix} \tag{38}$$

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

4.3 Constraint Term

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

$$\boldsymbol{\mathcal{J}}_c(\boldsymbol{\beta}; \boldsymbol{c}) = \begin{pmatrix} \mathcal{J}_{s_1}(\boldsymbol{\beta}_{s_1}; c_{s_1}) \\ \vdots \\ \mathcal{J}_{s_p}(\boldsymbol{\beta}_{s_p}; c_{s_p}) \\ \mathcal{J}_{t_{1,2}}(\boldsymbol{\beta}_{t_{1,2}}; c_{t_{1,2}}) \\ \vdots \\ \mathcal{J}_{t_{p-1,p}}(\boldsymbol{\beta}_{t_{p-1,p}}; c_{t_{p-1,p}}) \end{pmatrix} \tag{39}$$

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

4.4 Mapping Matrices for Tensor-Product Splines

The tensor-product spline basis is given by the Kronecker product of two B-spline bases, as depicted in Chapter *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.

We want to penalize adjacent coefficient differences to enforce smoothness, but this time, in two dimensions. Therefore, an appropriate spatial neighbourhood needs to be defined. An example for such neighbourhood for the coefficient $\beta_{j,k}$ is given by the coefficients left and right, i.e. $\beta_{j-1,k}$ and $\beta_{j+1,k}$, and the coefficients above and below, i.e. $\beta_{j,k-1}$ and $\beta_{j,k+1}$.

Let us now define the mapping matrices $\boldsymbol{D}_{d_1}$ and $\boldsymbol{D}_{d_2}$ of orders d_1 and d_2 for each dimension, respectively. Using the Kronecker product, we generate

the expanded mapping matrix $\boldsymbol{D}_{d_1,exp} = \boldsymbol{I}_{d_2} \otimes \boldsymbol{D}_{d_1}$ with the identity matrix $\boldsymbol{I}_{d_2} \in \mathbb{R}^{d_2 \times d_2}$ and $\boldsymbol{D}_{d_2,exp} = \boldsymbol{D}_{d_2} \otimes \boldsymbol{I}_{d_1}$ with the identity matrix $\boldsymbol{I}_{d_1} \in \mathbb{R}^{d_1 \times d_1}$.

Row-wise mappings of order d_1 and column-wise mappings of order d_2 are now obtained by applying the expanded difference matrix $\boldsymbol{D}_{d_1,exp}$ and $\boldsymbol{D}_{d_2,exp}$ to the coefficient vector $\boldsymbol{\beta}$, respectively.

Using these concepts, in principle every possible pair of one dimensional user-defined constraints can now be constructed, e.g. unimodality in two dimensions would be obtained using the unimodal penalty matrix depicted above for each dimension.

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

$$\mathcal{J}_c(\boldsymbol{\beta}; c_1, c_2) = \boldsymbol{\beta}^T \boldsymbol{D}_{c_1,exp}^T \boldsymbol{V}_1 \boldsymbol{D}_{c_1,exp} \boldsymbol{\beta} + \boldsymbol{\beta}^T \boldsymbol{D}_{c_2,exp}^T \boldsymbol{V}_2 \boldsymbol{D}_{c_2,exp} \boldsymbol{\beta} \tag{40}$$

with $\boldsymbol{D}_{c_1,exp} = \boldsymbol{I}_{d_2} \otimes \boldsymbol{D}_{c_1}$ and $\boldsymbol{D}_{c_2,exp} = \boldsymbol{D}_{c_2} \otimes \boldsymbol{I}_{d_1}$ as individual mapping matrices and the weighting matrices $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$ according to the given constraints.

The constrained penalized least squares objective function in (33) can now be used to estimate the coefficients $\boldsymbol{\beta}_{PLS,c,nd}$. **fahrmeir2013regression**

4.5 2-d Example

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

$$f(x_1, x_2) = 2 \exp\Big(- \frac{(x_1 - 0.25)^2}{0.08} \Big) + x_2^2 + \eta \tag{41}$$

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 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 8. The model fits the data quite well and holds the specified constraints for the individual dimensions.

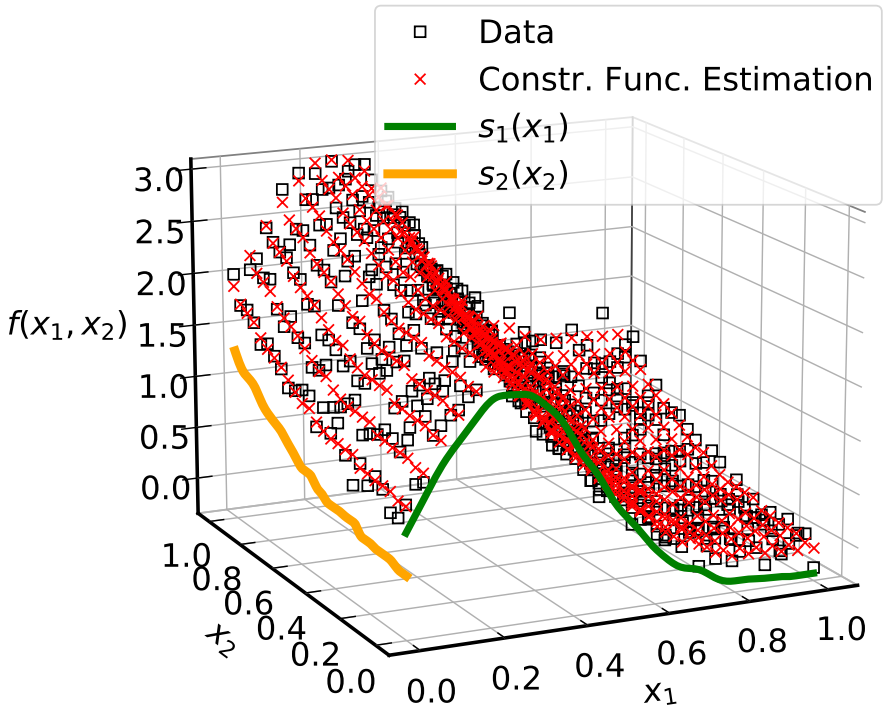

Figure 8: 2-d test function for n-d constrained function estimation