Chapter 3 - Solution Approach DRAFT

Weber Jakob

September 15, 2020

1 One dimension

For the sake of simplicity, the following description is restricted to one dimension and B-splines as basis functions. Later, the generalization to multiple dimensions using B-splines and tensor-product splines is given.

The goal is to model given data

$$\{x_i, y_i\},\ i = 1, ..., n$$

using a priori knowledge like monotonicity (increasing or decreasing), curvature (convex or concave), unimodality (peak or valley) or multi-modality and positivity. Using B-splines as basis functions for the estimation $\hat{y} = \hat{f}(x_1) = X\hat{\beta}$ of the unknown function y, the least squares objective function is given by

$$Q(y; \beta) = \|y - \hat{y}\|^2 = \|y - X\beta\|^2$$

where $X \in \mathbb{R}^{n \times k}$ is the B-spline basis for k splines and n data points and $\beta \in \mathbb{R}^k$ are the coefficients to be estimated. The explicit solution for the least squares objective function is given by

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

The following figure shows a B-spline smooth using $k = 15$ splines on an equidistant grid for noisy data as well as the individual B-spline basis functions multiplied with the corresponding, estimated least squares coefficients $\hat{\beta}_{LS}$.

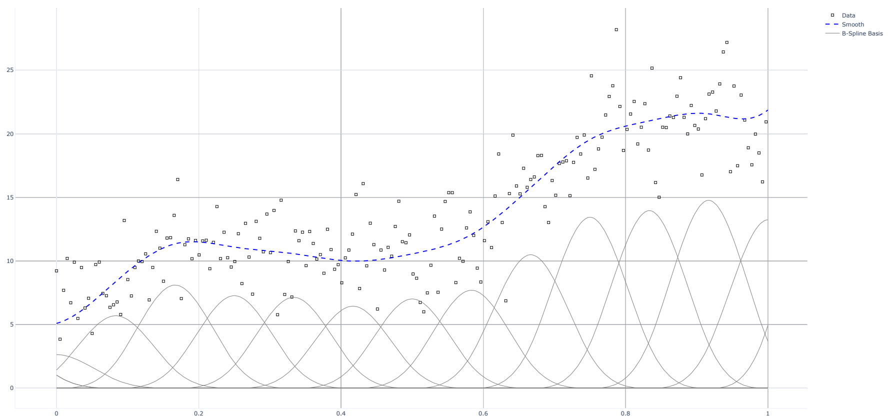

Figure 1: B-spline smooth and basis functions

The number of splines k determines the amount of smoothing. Using a small number is leading 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 might be very small but the smoothness of the estimated function may be large. This leads to large interpolation errors and wiggle function estimates. This behavior is depicted in the following figure.

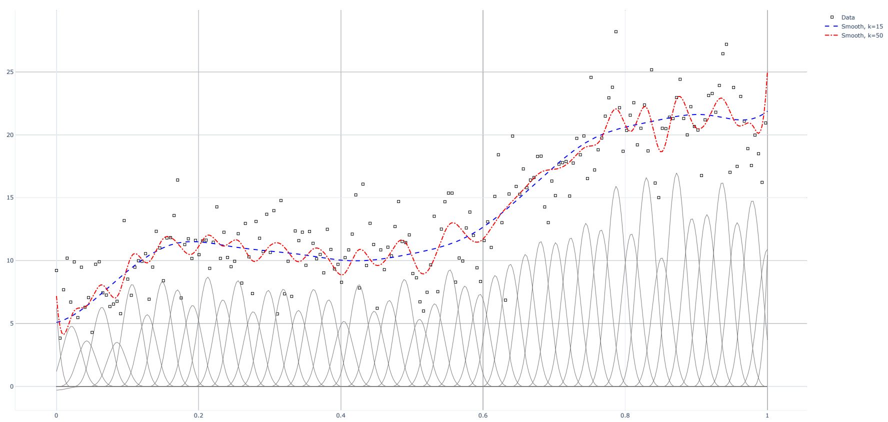

Figure 2: B-spline smooth and basis functions for largen number of splines

To prevent this, Eilers and Marx (1996) introduced the P-splines, which include a penalty based on the squared finite difference of order d of adjacent coefficients. If $d = 2$, this qualitatively corresponds to a penalized second derivative, which itself is a measure for function wiggliness.

The difference operator Δ^d 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 $d = 1$

$$D_1 = \begin{pmatrix} -1 & 1 & & & \\ -1 & 1 & & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{pmatrix} \in R^{k-1\times k}$$

and order $d = 2$

$$D_2 = \begin{pmatrix} 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \end{pmatrix} \in R^{k-2\times k}$$

Using the finite difference operator of order d, the least squares objective function is expanded to the penalized least squares objective function given by

$$Q(y;\beta) = \|y - X\beta\|^2 + \lambda_s \mathcal{J}_s(\beta; d)$$

where $\mathcal{J}_s(\beta; d) = \beta^T D_d^T D_d \beta$ and the smoothing parameter λ_s determines the amount of smoothing. The matrix D_d is called penalty matrix. The smoothing parameter λ_s plays a critical role and can be optimized using information criteria like AIC and BIC or by using cross-validation techniques.

The explicit solution for the penalized least squares coefficients is then given by

$$\hat{\beta}_{PLS} = (X^T X + \lambda_s D_d^T D_d)^{-1} X^T y.$$

For small values $\lambda_s \to 0$, the penalized least squares estimate $\hat{\beta}_{PLS}$ approaches the least squares estimate $\hat{\beta}_{LS}$, while for large values $\lambda_s >> 0$, 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.

The following figure shows the behavior of P-splines for several values of the smoothing parameter $\lambda_s = \{10^{-2}, 10^0, 10^2, 10^6\}$. As the value of λ_s gets larger, the fitted curve is more smooth and finally approaches a straight line.

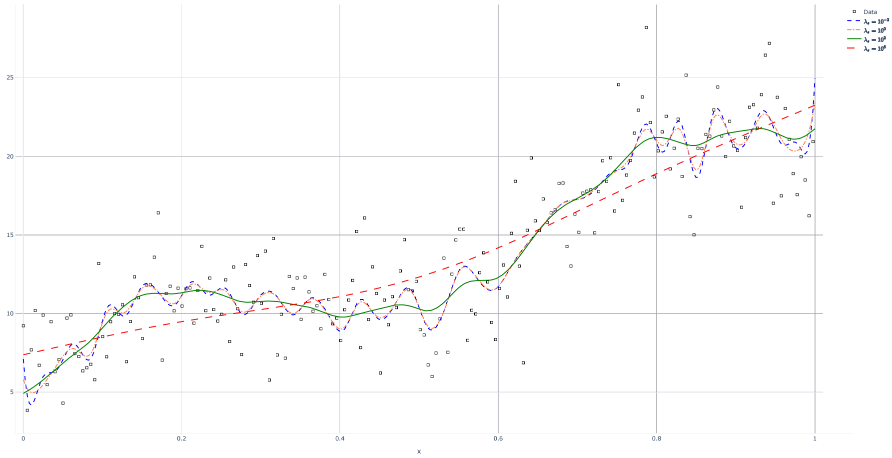

Figure 3: P-spline smooth for different λ_s

A priori knowledge can now be incorporated by an iterative approach using a sophisticated choice of the penalty matrix D_c and the use of a weight matrix V. The scheme is depicted using the user defined constraint of monotonic increasing behavior.

Monotonic increasing behavior can be obtained using the D_1 matrix as penalty matrix and a diagonal weight matrix V, where the diagonal elements v_j are given by

$$v_j = \begin{cases} 0, & \text{if } \Delta^1 \beta > 0 \\ 1, & \text{if } \Delta^1 \beta \leq 0 \end{cases}$$

.

Qualitatively, this states that the monotonic increasing penalty term is only active if adjacent coefficients β_{j-1} and β_j are non-increasing. An already increasing sequence of coefficients is not affected by this penalty term.

The penalized least squares objective function is now expanded by a term representing the user defined constraint yielding the constraint penalized least squares objective function and is given by

$$Q(y;\beta) = \|y - X\beta\|^2 + \lambda_s \mathcal{J}_s(\beta; d) + \lambda_c \mathcal{J}_c(\beta; c)$$

where $\mathcal{J}_c(\beta; c) = \beta^T D_c^T V D_c \beta$ using the user defined penalty matrix D_c and λ_c as parameter which determines the influence of the penalty. The parameter λ_c is generally set quite large, i.e. $\lambda_c > 10^4$, to enforce the user defined constraint.

Again, an explicit formula for the constraint penalized least squares estimate can be given as

$$\hat{\beta}_{PLS,c} = (X^T X + \lambda_s D_d^T D_d + \lambda_c D_c^T V D_c)^{-1} X^T y.$$

An initial estimate $\hat{\beta}_{init}$ is needed to compute the weight matrix. The unconstraint least squares estimate $\hat{\beta}_{LS}$ is a valid candidate. Now the calculation of the constraint penalized least squares estimate $\hat{\beta}_{PLS,c}$ and the calculation of the weight matrix V is iterated until no more changes in the weight matrix V appear. This scheme is called penalized iteratively-reweighted least squares and is abbreviated by *PIRLS*.

The parameter λ_c plays a similar role as the smoothing parameter λ_s, but should be set orders of magnitude higher than λ_s to enforce the user defined constraint.

The following figure shows an example of the use of the monotonicity constraint. The smoothing parameter was set to $\lambda_s = 0.1$ and the constraint parameter was set to $\lambda_c = 6000$. For both smooths, the number of used splines k was set to 30. Visual inspection shows that the constraint, red smooth follows the a priori known behavior of monotonicity far better than the blue, unconstraint smooth.

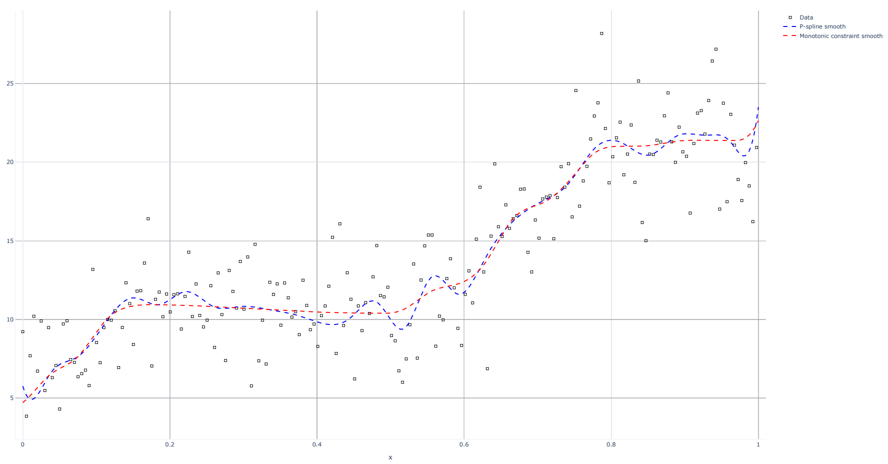

Figure 4: Constraint smooth for different λ_c

This shows, that the incorporation of a priori knowledge in the fitting process using B-splines is in principle possible using a sophisticated choice of the penalty matrix D_c and an iterative fitting approach using penalized iteratively-reweighted least squares. It is important to 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.

2 Penalty Matrices

As stated before, a priori knowledge can be introduced by the choice of the penalty matrix D_c and the weight matrix V. It now follows a description of the different penalty matrices, which are used to enforce a priori known behavior.

2.1 Monotonicity

The penalty matrix enforcing monotonic behavior is given by the use of the first order difference operator Δ^1. In matrix form for k splines, it is given as

$$D_{monoton} = \begin{pmatrix} -1 & 1 & & \\ & -1 & 1 & \\ & & \ddots & \ddots \end{pmatrix} \in \mathbb{R}^{k-1 \times k}.$$

The difference between monotonic increasing and decreasing behavior is controlled by the weight matrix V. For increasing behavior, the weight matrix V is given by the weights v_j according to

$$v_j = \begin{cases} 0, & \text{if } \Delta^1 \beta_j > 0 \\ 1, & \text{if } \Delta^1 \beta_j \leq 0. \end{cases}$$

For decreasing behavior, the weight matrix V is given by the weights v_j according to

$$v_j = \begin{cases} 0, & \text{if } \Delta^1 \beta_j < 0 \\ 1, & \text{if } \Delta^1 \beta_j \geq 0. \end{cases}$$

This states, that the penalty term is only applied if adjacent coefficients β_{j-1} and β_j are increasing or decreasing, respectively.

2.2 Curvature

In the simplest case, the curvature of a function can either be convex or concave. The penalty matrix enforcing this behavior is given by the use of the second order difference operator Δ^2. In matrix form for k splines, it is given as

$$D_{curvature} = \begin{pmatrix} 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \end{pmatrix} \in R^{k-2 \times k}.$$

The difference between concave and convex curvature is controlled by the weight matrix V. For concave behavior, the weight matrix V is given by the weights v_j according to

$$v_j = \begin{cases} 0, & \text{if } \Delta^2 \beta_j < 0 \\ 1, & \text{if } \Delta^2 \beta_j \geq 0. \end{cases}$$

For convex curvature, the weight matrix V is given by the weights v_j according to

$$v_j = \begin{cases} 0, & \text{if } \Delta^2 \beta_j > 0 \\ 1, & \text{if } \Delta^2 \beta_j \leq 0. \end{cases}$$

2.3 Unimodality

The penalty matrix enforcing unimodal behavior can be constructed using the first order difference operator Δ^1. The weight matrix V now has a special structure. We assume that there is a peak in the data and therefore want to constrain the fit to include a peak. We then need to find the index i_{peak} of the spline, which has the maximal value around this peak. The index i_{peak} is now used as splitting point for the weight matrix V. All coefficients i for $i < i_{peak}$ are constraint to be monotonic increasing, while all coefficients i for $i > i_{peak}$ are constraint to be monotonic decreasing. The coefficient at position i_{peak} stays unconstraint. The unimodal penalty matrix has the form

$$D_{unimodal} = \begin{pmatrix} -1 & 1 & & & & \\ & \ddots & \ddots & & & \\ & & 0 & 0 & & \\ & & & -1 & 1 & \\ & & & & \ddots & \ddots \end{pmatrix} \in R^{k-1 \times k}$$

The weights v_j have the following structure:

$$v_j = \begin{cases} \begin{cases} 0, & \text{if } \Delta^1 \beta_j > 0 \\ 1, & \text{if } \Delta^1 \beta_j \leq 0. \end{cases} & , \quad \text{if } j < i_{peak} \\ \begin{cases} 0, & \text{if } \Delta^1 \beta_j < 0 \\ 1, & \text{if } \Delta^1 \beta_j \geq 0. \end{cases} & , \quad \text{if } j > i_{peak} \end{cases}$$

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 spline index of the valley i_{valley}, then constraining all splines for $i < i_{valley}$ to be monotonic decreasing and all splines for $i > i_{valley}$ to be monotonic increasing. The coefficient at position i_{valley} stays unconstraint. The weights v_j for the weight matrix are the given by

$$v_j = \begin{cases} \begin{cases} 0, & \text{if } \Delta^1 \beta_j < 0 \\ 1, & \text{if } \Delta^1 \beta_j \geq 0. \end{cases} & , \quad \text{if } j < i_{valley} \\ \begin{cases} 0, & \text{if } \Delta^1 \beta_j > 0 \\ 1, & \text{if } \Delta^1 \beta_j \leq 0. \end{cases} & , \quad \text{if } j > i_{valley} \end{cases}$$

2.4 Multi-modality

The penalty and weight matrices for the multi-modality constraint can be constructed using the scheme of unimodal constraints for each mode. It is important to find the right peak or valley points, which can be difficult with noisy data.

2.5 Penalty 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 penalty matrices to two dimensions and tensor-product splines, we again use the concept of the Kronecker product.

We want to penalized adjacent coefficient differences, but this time, in two dimensions. Therefore, an appropriate spatial neighbourhood needs to be defined. An example for such neighbourhood for the coefficient β_{jk} 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 a penalty matrix of order d for each dimension and denote them by D_d^1 for dimension 1 and D_d^2 for dimension. Using the Kronecker product, we generate the expanded difference matrix $D_{d,exp}^1 = I_{d_2} \otimes D_d^1$ for I_{d_2} as identity matrix of dimensions $d_2 \times d_2$ and $D_{d,exp}^2 = D_d^2 \otimes I_{d_1}$ for I_{d_1} as identity matrix of dimensions $d_1 \times d_1$.

Row-wise differences of order d and column-wise differences of order d are now obtained by applying the expanded difference matrix $D_{d,exp}^1$ and $D_{d,exp}^2$ to the coefficient vector β, respectively.

Using these concepts, in principle every possible pair of one dimensional 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 then has the form

$$\mathcal{J}_c(\beta; d) = \beta^T {D_{c,exp}^1}^T V_1 D_{c,exp}^1 \beta + \beta^T {D_{c,exp}^2}^T V_2 D_{c,exp}^2 \beta$$

with $D_{c,exp}^1 = I_{d_2} \otimes D_{unimodal}^1$ and $D_{c,exp}^2 = D_{unimodal}^2 \otimes I_{d_1}$ as individual penalty terms and V_1 and V_2 as weight matrices. The same constraint penalized least squares objective function can now be used to estimate the coefficients $\beta_{PLS,c}$.

3 Other constraints

3.1 Positivity

For certain physical systems, it is known a priori that the measured quantity cannot be smaller than zero. 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 use user-defined constraints for positivity.

The user defined constraint for positivity again uses a weight matrix V_{pos}, with individual weights v_j specified as follows:

$$v_j = \begin{cases} 0, & \text{if } \hat{y} = X\hat{\beta} > 0 \\ 1, & \text{if } \hat{y} = X\hat{\beta} \leq 0 \end{cases}$$

The constraint penalized least squares objective function is then of the form

$$Q(y; \beta) = \|y - X\beta\|^2 + \lambda_s \mathcal{J}_s(\beta; d) + \lambda_{pos} \mathcal{J}_{pos}(\beta)$$

where $\mathcal{J}_{pos} = \beta^T X^T V_{pos} X \beta$ is the penalty term specifying positivity and λ_{pos} is the constraint parameter, which is set multiple orders of magnitude higher than the smoothness parameter λ_s to enforce the constraint.

Using this approach, negativity and special threshold value constraints, e.g. the output is constraint to be larger than 2, can also be enforced.

4 Extension to Multiple Dimensions

The extension from one input to multiple inputs uses the concept of additive models given in *Chapter Additve Models.* Given input data $\{x_{i1}, \dots, x_{ip}, y_i\}$

for $i = 1, \ldots, n$ and p as the number of inptus, the combined model using all available B-splines and tensor-product splines is given as

$$\hat{y} = f(x_1, ..., x_p) = \sum_{i=1}^{p} s_i(x_i) + \sum_{i=1}^{p-1} \sum_{j>i}^{p} t_{i,j}(x_i, x_j)$$

where $s_i(x_i)$ is the B-spline smooth given by $s_i(x_i) = X_i \beta_i$ and $t_{i,j}(x_i, x_j)$ is the tensor-product smooth given by $t_{i,j}(x_i, x_j) = X_{i,j} \beta_{i,j}$. The total number of smooths is then given by $n_{smooths} = p + \frac{p(p-1)}{2}$. Assuming the use of k splines for the B-spline smooth and k^2 splines for the tensor-product smooth, the total number of coefficients to estimate is given by $k_{smooths} = p * k + \frac{p(p-1)}{2} k^2$. Since all B-spline smooths and tensor-product spline smooths follow a linear model structure, we can combine them into one large model given by

$$\hat{y} = X\beta$$

where the combined basis matrix $X \in \mathbb{R}^{n \times k_{smooths}}$ is given by a horizontal concatenation of the individual bases and the combined coefficient vector $\beta \in \mathbb{R}^{k_{smooths}}$ is given by a vertical concatenation of the individual coefficient vectors. The combined model $\hat{y}_{comb}$ now has the following form

$$\hat{y}_{comb} = X\beta = \begin{pmatrix} X_1 \ldots X_p \ X_{1,2} \ldots X_{p-1,p} \end{pmatrix} \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_p \\ \beta_{12} \\ \vdots \\ \beta_{p-1,p} \end{pmatrix}.$$

The data term in the constraint penalized least squares objective function given above can now be evaluated using arbitrary input dimensions. The remaining question is now how the smoothness penalty term and the constraint penalty term are constructed. For both, the concept of block diagonal matrices is applied. The smoothness penalty term $\mathcal{J}_s(\beta, \mathbf{d})$ is now given as

$$\mathcal{J}_s(\beta; \mathbf{d}) = \text{blockdiag}(\mathcal{J}_1, \ldots, \mathcal{J}_p, J_{1,2}, \ldots, \mathcal{J}_{p-1,p})$$

with $\mathcal{J}_i := \mathcal{J}_i(\beta_i; d_i) = \beta_i^T D_{d_i}^T D_{d_i} \beta_i$ determining the smoothness penalty term for each individual smooth i.

The constraint penalty term $\mathcal{J}_c(\beta; c)$ is then given as

$$\mathcal{J}_c(\beta; c) = \text{blockdiag}(\mathcal{J}_{1,c}, \ldots, \mathcal{J}_{p,c}, \mathcal{J}_{1,2,c} \ldots, \mathcal{J}_{p-1,p,c})$$

with $\mathcal{J}_{i,c} := \mathcal{J}_{i,c}(\beta_i; c_i) = \beta_i^T D_{c_i}^T V_{c_i} D_{c_i} \beta_i$ determining the constraint penalty term for each individual smooth i with individual weight matrix V_{c_i}.

The constraint penalized least squares objective function can now be written as

$$Q(y; \beta) = \|y - X\beta\|^2 + \boldsymbol{\lambda}_s \mathcal{J}_s(\beta; \boldsymbol{d}) + \boldsymbol{\lambda}_c \mathcal{J}_c(\beta; \boldsymbol{c})$$

this time with $\boldsymbol{\lambda}_s, \boldsymbol{\lambda}_c \in \mathbb{R}^{n_{smooths}}$ defined as vectors with one value of smoothness and constraint parameter for each smooth, respectively. Using the penalized iteratively-reweighted least squares algorithm, we then obtain the estimated coefficients using the explicit solution

$$\hat{\beta}_{PLS,c} = (X^T X + \boldsymbol{\lambda}_s \mathcal{J}_s + \boldsymbol{\lambda}_c \mathcal{J}_c)^{-1} X^T y.$$

As example, we take a look at the function

$$f(x_1, x_2) = 1.2 * \exp\big(-\frac{(x_1 - 0.5)^2}{0.05} - \frac{(x_2 - 0.5)^2}{0.05}\big) + x_1 * x_2$$

for $x_1 \in [0, 1]$ and $x_2 \in [0, 1]$ and random gaussian noise with $\sigma_{noise} = 0.01$. We therefore expect a peak in the data. Using this knowledge, we create a model using the following smooths:

- B-spline smooth $s_1(x_1)$ using $k_{x_1} = 20$, $c =$ peak, $\lambda_s = 1$ and $\lambda_c = 1000$
- B-spline smooth $s_2(x_2)$ using $k_{x_2} = 20$, $c =$ peak, $\lambda_s = 1$ and $\lambda_c = 1000$
- Tensor-product smooth $t_{1,2}(x_1, x_2)$ using $k_{x_1} = k_{x_2} = 10$, $c =$ smoothness, $\lambda_s = 1$, $\lambda_c = 1000$

The fit for this model is shown in the following figure. The model fits the data quite well and holds the specified constraints.

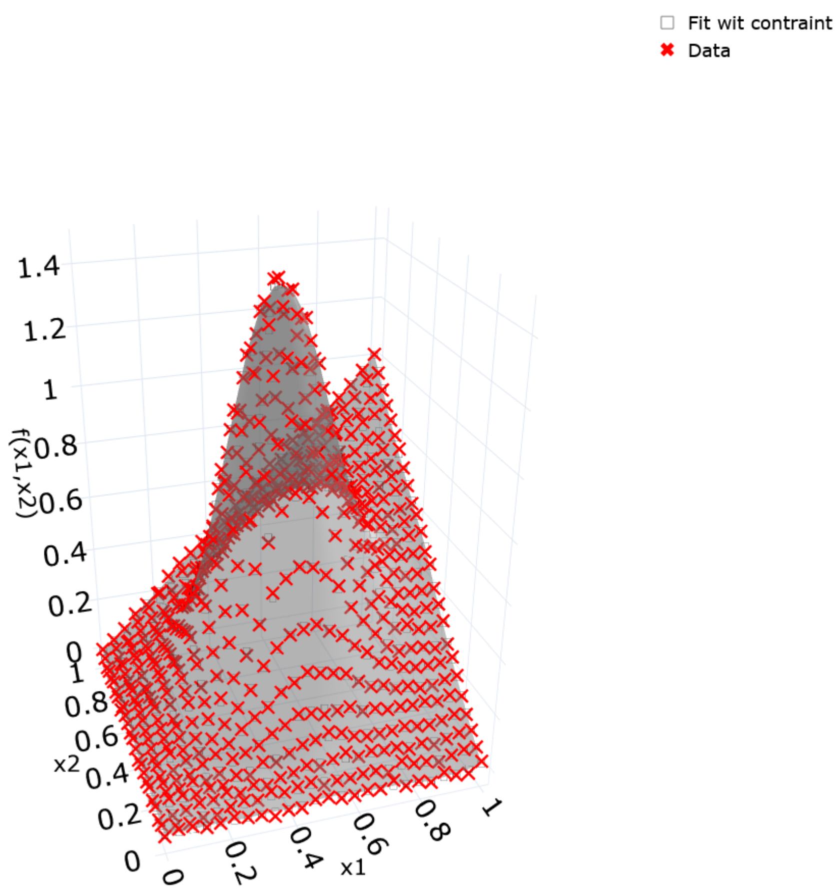

Figure 5: Test function and fit

The following figures show the individual smooths. The peak behaviour of smooth $s_1(x_1)$ and $s_2(x_2)$ is clearly visible.

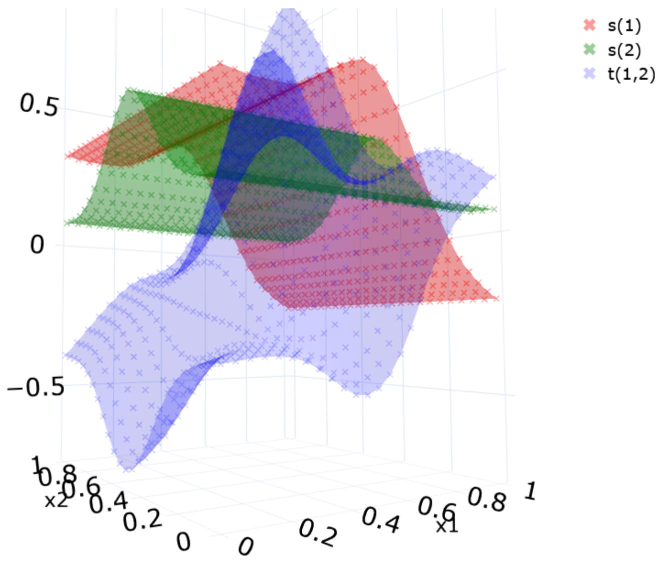

Figure 6: Parition of the fit into individual parts