Static Function Approximation using interpretable, data-driven methods

DIPLOMARBEIT

Ausgeführt zum Zwecke der Erlangung des akademischen Grades eines

Diplom-Ingenieurs (Dipl.-Ing.)

unter der Leitung von

Dipl.-Phys. Dr. Claas Abert
Dipl.-Ing. Dr.techn. Stephan Strommer

eingereicht an der

Technischen Universität Wien
Fakultät für Elektrotechnik und Informationstechnik
Institut für Automatisierungs- und Regelungstechnik

von
<Jakob Weber>
Matrikelnummer <a01326729
Rahlgasse 6/26-27
1060 Vienna
Austria

Vienna, 23. September 2020

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Introduction

This is the introduction chapter. It is going to contain the following points:

- Background Motivation: Like in the expose, data-driven approaches are necessary because of complexe phenomena
- Interpretable AI: Shift from black-box models to interpretable models
- Related work in XAI
- outline

2 Fundamentals

2.1 Linear Models

2.1.1 Definition and Model Assumptions

Given the set of data points $\{x_{i1}, ..., x_{iq}; y_i\}$ for $i = 1, ..., n$, we aim to model the relation between the set of inputs or predictor variables $\{x_1, ..., x_q\}$ and the output y with a function $f(x_1, ..., x_q)$ and an additional noise term ϵ. Thus we obtain the model formulation as

$$y_i = f(x_{i1}, ..., x_{iq}) + \epsilon_i \tag{2.1}$$

.

The goal is now the estimation of the unknown function f. For this, several assumptions on the model structure are needed:

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

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

   $$f(x_1, ..., x_q) = \beta_0 + \beta_1 x_1 + ... + \beta_q x_q, \tag{2.2}$$

   with unknown parameters $\beta_0, ..., \beta_q$, which need to be estimated. The model is therefore linear in its parameters as well as in its inputs. [1] The parameter β_0 is called intercept or bias in the machine learning community. For centered data, i.e. $\mathbb{E}(x_i) = 0$, the intercept is equal to zero and can be neglected.

   Commonly, the linear model is represented in vector notation given by

   $$y_i = f(x_{i1}, ..., x_{iq}) = x_i' \beta + \epsilon_i$$

   where $x' = (1, x_{i1}, \ldots, x_{iq})$ and $\beta = (\beta_0, \ldots, \beta_q)'$.

2. *Additive errors*

   The assumptions of additive errors leads to the following model structure

   $$y = x'\beta + \epsilon \tag{2.3}$$

   This assumptions comes out to be quite restrictive, although it is reasonable for many practical applications.

To estimate the unknown parameters β, we define the vectors $y = (y_1, ..., y_n)'$ and $\epsilon = (\epsilon_1, ..., \epsilon_n)'$ as well as the design matrix X,

$$X = \begin{pmatrix} 1 & x_{11} & \dots & x_{1q} \\ \vdots & & & \vdots \\ 1 & x_{n1} & \dots & x_{nq} \end{pmatrix} \in \mathbb{R}^{n \times q+1} \tag{2.4}$$

and generate n equations like Eq.2.3 which can be combined as

$$y = X\beta + \epsilon. \tag{2.5}$$

We assume that the design matrix X has full column rank, i.e. $rk(X) = q+1 = p$, implying linear independence of the columns of X, which is necessary to obtain a unique estimator for the regression coefficients β. [2]

Another necessary requirement is that the number of data points n is larger or equal to the number of regression coefficients p, which is equal to the statement that the linear system $y = X\beta$ is not under-determined.

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

1. *Expectation of the error*
   The errors have a mean of zero, i.e. $\mathbb{E}[\epsilon_i] = 0$

2. *Variances and correlation structure of the errors*
   We assume constant error variance with $\mathbb{V}ar[\epsilon_i] = \sigma^2$ (homoscedasticity). Additionally, we assume that the errors are uncorrelated, which means $\mathbb{C}ov(\epsilon_i, \epsilon_j) = 0$ for $i \neq j$. These assumptions combined lead to the covariance matrix $\mathbb{C}ov(\epsilon) = \mathbb{E}(\epsilon\epsilon') = \sigma^2 I$.

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

It follows from the model assumptions that

$$\mathbb{E}[y_i] = \mathbb{E}[x_i'\beta + \epsilon_i] = x_i'\beta \tag{2.6}$$

$$\mathbb{V}[y_i] = \mathbb{V}[x_i'\beta + \epsilon_i] = \mathbb{E}\big[(x_i'\beta + \epsilon_i - \mathbb{E}[y_i])^2\big] = \mathbb{V}[\epsilon_i] = \sigma^2 \tag{2.7}$$

$$\mathbb{C}ov(y_i, y_j) = \mathbb{C}ov(\epsilon_i, \epsilon_j) = 0, \tag{2.8}$$

for the mean and variance of y_i, and the covariance between y_i and y_j. With the additionally assumed Gaussian errors, we have

$$y \sim \mathcal{N}(X\beta, \sigma^2 I). \tag{2.9}$$

A linear model can therefore be interpreted as a normal distribution $\mathcal{N}(\mu, \sigma^2)$ with its mean given by $\mu = X\beta$ and its variance given by $\sigma^2 = \sigma^2 I$. To specify the linear model given through Eq.2.9, we need to estimate the regression coefficients β and the variance σ^2.

2.1.2 Parameter Estimation

The linear model given in Eq. 2.9 has the unknown parameters β and σ which need to be estimated using given data. In the following part, the estimators $\hat{\beta}$ and $\hat{\sigma}$ are introduced, and their statistical properties are derived.

2.1.2.1 Estimation of the Regression Coefficients β

The two main methods for the estimation of the regression coefficients in the context of linear models are

- Method of Least Squares
- Method of Maximum Likelihood

For Gaussian errors, the maximum likelihood estimator for the regression coefficients coincides with the least squares estimator.

2.1.2.1.1 The Method of Least Squares

The unknown regression coefficients β are estimated by minimizing the sum of squared error

$$\begin{aligned} \mathrm{LS}(\beta) &= \sum_{i=1}^{n} (y_i - x_i'\beta)^2 \\ &= \sum_{i=1}^{n} \epsilon_i^2 \\ &= \epsilon'\epsilon \end{aligned} \tag{2.10}$$

with respect to $\beta \in \mathbb{R}^p$. Rewriting of Eq.2.10 leads to the least squares criterion

$$\begin{aligned} \mathrm{LS}(\beta) &= \epsilon'\epsilon \\ &= (y - X\beta)'(y - X\beta) \\ &= y'y - 2y'X\beta + \beta'X'X\beta. \end{aligned}$$

The least squares criterion is minimized by setting its first derivative equal to zero and by showing that the matrix of second derivatives is positive definite. Applying the rules of differentiation we obtain

$$\frac{\partial LS(\beta)}{\partial \beta} = -2X'y + 2X'X\beta.$$

The second derivative is given by

$$\frac{\partial^2 LS(\beta)}{\partial \beta \partial \beta'} = 2X'X$$

Since $X \in \mathbb{R}^{n \times p}$ for $p = q + 1$ has full rank (per assumption), the matrix $X'X$ is positive definite. The least squares estimate $\hat{\beta}_{LS}$ is then obtained by solving the so-called *normal equations*

$$X'X\hat{\beta} = X'y. \tag{2.11}$$

Since $X'X$ is positive definite and invertible, the normal equations in Eq.2.11 have a unique solution given by the least squares estimate

$$\hat{\beta}_{LS} = (X'X)^{-1}X'y. \tag{2.12}$$

2.1.2.1.2 Maximum Likelihood Estimation

Assuming normally distributed errors, i.e. $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$, the maximum likelihood estimators for the unknown parameters β and σ^2 can be computed. Under the normality assumption the likelihood is defined as

$$L(\beta, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\Big(- \frac{1}{2\sigma^2}(y - X\beta)'(y - X\beta)\Big).$$

The log-likelihood is then given by

$$l(\beta, \sigma^2) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}(y - X\beta)'(y - X\beta).$$

Thus, maximizing the log-likelihood with respect to β is equivalent to minimizing the least squares criterion given in Eq.2.10. The maximum likelihood estimator of β is therefore equivalent to the least squares estimator in Eq.2.12.

2.1.2.1.3 The Hat Matrix

Using the least squares estimator $\hat{\beta}_{LS} = (X'X)^{-1}X'y$, we can estimate the mean of y by

$$\widehat{\mathbb{E}[y]} = \hat{y} = X\hat{\beta}_{LS}$$

.

This results in

$$\hat{y} = X(X'X)^{-1}X'y = Hy, \tag{2.13}$$

with the matrix $H \in \mathbb{R}^{n \times n}$, 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

$$\hat{\epsilon} = y - \hat{y} = (I - H)y.$$

The hat matrix H has the following useful properties:

- H is symmetric.

- H is idempotent ($H^2 = HH = H$)
- The rank of H is equal to its trace.
- $\frac{1}{n} \leq h_{ii} \leq \frac{1}{r}$, where r represents the number of rows in X with different x_i. If all rows are different, then $r = 1$.
- The matrix $(I - H)$ is also idempotent and symmetric, with $rk(I - H) = n - p$.

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

2.1.2.2 Estimation of the Variance σ^2

The estimation of the variance σ^2 is necessary for the construction of confidence intervals of the regression coefficients and for the construction of prediction intervals. It is further used in all kinds of statistical tests.

2.1.2.2.1 Maximum Likelihood Estimation

The variance σ^2 can be estimated using the maximum likelihood method by differentiation of the log-likelihood $l(\beta, \sigma^2)$ with respect to σ^2. The derivative is given by

$$\frac{\partial l(\beta, \sigma^2)}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4}(y - X\beta)'(y - X\beta).$$

Substituting the maximum likelihood estimator $\hat{\beta}$ for β results in the maximum likelihood estimator for the variance σ^2 given by

$$\hat{\sigma}^2_{ML} = \frac{(y - X\hat{\beta})'(y - X\hat{\beta})}{n} = \frac{\hat{\epsilon}'\hat{\epsilon}}{n}. \tag{2.14}$$

This estimator for σ^2 is rarely used since it is biased, i.e. $\mathbb{E}[\sigma^2_{ML}] \neq \sigma^2$.

2.1.2.2.2 Restricted Maximum Likelihood Estimation

Using $\mathbb{E}(\hat{\epsilon}'\hat{\epsilon}) = (n - p)\sigma^2$, we obtain for the restricted maximum likelihood estimation of the variance σ^2 the following:

$$\hat{\sigma}^2_{REML} = \frac{1}{n - p}\hat{\epsilon}'\hat{\epsilon}, \tag{2.15}$$

which is the commonly used estimator for σ^2. The restricted maximum likelihood estimator for the variance is in general less biased. Therefore, it is a better choice to use this estimator.

2.1.3 Model Selection Techniques, Model Choice Criteria and Regularization

Linear models can be used to generate models using a large number of input or predictor variables. The challenge is then to decide which of these variables to include in the model.

On way of comparing various models, i.e models using different sets of variables, is the use of model choice criteria, e.g. Mallow's CP or AIC. Generally, these criteria can be split in two parts. The first part measures the goodness of fit, e.g. using the sum of squared errors, while the second part measures the complexity of the model.

Most model choice criteria are based on the sum of squared prediction error $SPSE$. Therefore, the derivation of the sum of squared prediction error $SPSE$ is given first.

We assume given data $\{x_{i1}, \dots, x_{iq}; y_i\}$ for $i = 1, \dots, n$. Further, we assume that the expectation $\mathbb{E}[y_i] = \mu_i$ and the variance $\mathbb{V}ar[y_i] = \sigma^2$. Using the q variables x_i we can generate the corresponding design matrix X for the linear model $y = X\beta$. The least squares estimator for β is then given by

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

The data y can be interpreted as random variable. We can then define an estimator $\hat{y}$ for the vector μ of expectations $\mu_i = \mathbb{E}[y_i]$ by

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

The following properties the $\hat{y}$ hold:

- $\mathbb{E}[\hat{y}] = X(X'X)^{-1}X'\mathbb{E}[y]$
- $\mathbb{C}ov(\hat{y}) = \sigma^2 X(X'X)^{-1}X'$
- $\sum_{i=1}^{n} \mathrm{Var}[\hat{y}_i] = \sigma^2 \mathrm{tr}(X(X'X)^{-1}X') = |M|\sigma^2$, where M represents the number of used variables.
- Sum of Mean Squared Error

$$\begin{aligned} \text{SMSE} &= \sum_{i=1}^{n} \mathbb{E}[\hat{y}_i - \mu_i]^2 \\ &= \sum_{i=1}^{n} \mathbb{E}\big[(\hat{y}_i - \mathbb{E}[\hat{y}_i]) + (\mathbb{E}[\hat{y}_i] - \mu_i)\big]^2 \\ &= |M|\sigma^2 + \sum_{i=1}^{n} (\mathbb{E}[\hat{y}_i] - \mu_i)^2. \end{aligned} \tag{2.16}$$

If we now assume new data $\{x_{i1}, \dots, x_{ik};\ y_{n+i} = \mu_i + \epsilon_{n+i}\}$ for $i = 1, \dots, n$, we can use the estimator $\hat{y}$ as a prediction for these new observations. We can therefore derive the sum of the expected squared prediction errors SPSE, given by

$$
\begin{aligned}
\text{SPSE} &= \sum_{i=1}^{n} \mathbb{E}[y_{n+i} - \hat{y}_i]^2 \\
&= \sum_{i=1}^{n} \mathbb{E}\big[(y_{n+i} - \mu_i) - (\hat{y}_i - \mu_i)\big]^2 \\
&= \sum_{i=1}^{n} \mathbb{E}[y_{n+1} - \mu_i]^2 + 2\mathbb{E}[(y_{n+1} - \mu_i)(\hat{y}_i - \mu_i)] + \mathbb{E}[haty_i - \mu_i]^2 \\
&= \sum_{i=1}^{n} \mathbb{E}[y_{n+i} - \mu_i] + \sum_{i=1}^{n} (\mathbb{E}[\hat{y}_i] - \mu_i)^2 \\
&= n\sigma^2 + SMSE \\
&= n\sigma^2 + |M|\sigma^2 + \sum_{i=1}^{n} (\mathbb{E}[\hat{y}_i] - \mu_i)^2.
\end{aligned}
\tag{2.17}
$$

The sum of the expected squared prediction error can be split into three parts

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

- *Variance Term*
  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 variables.

- *Squared Bias Term*
  The last term can be interpreted as bias. It can be reduced by increasing the model complexity.

The sum of expected squared prediction error is an example of the bias-variance tradeoff, which is characteristic for all statistical models. It states that by increasing model complexity, the bias is reduced but instead the variance is increased. On the other side, by decreasing model complexity, the variance of the model is reduced, but the bias is increased. [1]

In practice, the true value for the SPSE is not accessible since μ_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

$$\widehat{\text{SPSE}} = \sum_{i=1}^{n} (y_{n+i} - \hat{y}_i)^2.$$

   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 error term depending on the estimated variance and the model complexity. The estimate is thus given by

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

Typically used model choice criteria follow the basic idea of the SPSE.

2.1.3.1 Model Choice Criteria

2.1.3.1.1 Corrected Coefficient of Determination R^2_{corr}

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

$$R^2 = 1 - \frac{\sum_{i=1}^{n} \hat{\epsilon}_i^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2},$$

where $\bar{y}$ is defined as the mean value of y. The major drawback of R^2 is that it will never decrease when further predictors 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

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

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.

2.1.3.1.2 Mallow's Cp

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

$$\mathrm{C}_p = \frac{\sum_{i=1}^{n} (y_i - \mathbb{E}[y_i])^2}{\hat{\sigma}^2} - n + 2|M|,$$

where M is again the number of used parameters. Lower values of Mallow's C_p correspond to a better model fit.

2.1.3.1.3 Akaike Information Criterion

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

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

where $l(\hat{\beta}_{ML}, \hat{\sigma}^2_{ML})$ is the value of the log-likelihood at its maximum. We again have the standard model choice criteria structure of a data dependent term, here the maximal log-likelihood, and a model dependent term given.

The log-likelihood for a linear model assuming Gaussian errors is given by

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

Therefore, neglecting the constant value n, the AIC evaluates to

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

Lower values of AIC correspond to a better model fit.

2.1.3.1.4 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

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

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

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

Lower values of BIC correspond to a better model fit.

2.1.3.1.5 Cross Validation

The basic idea of cross validations is to split the existing data set into multiple smaller ones and to fit one model to each of these smaller data sets. These models are then evaluated by the calculation of the SPSE on the data which it was not trained on. The model which has the smallest error is then chosen as final estimate.

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 estimator trained on all data y and the hat matrix $H = X(X'X)^{-1}X'$. The cross validation score is then given by

$$\mathrm{CV} = \frac{1}{n}\sum_{i=1}^{n}(\frac{y_i - \hat{y}_i}{1 - h_{ii}})^2.$$

where h_{ii} denotes the diagonal elements of the hat matrix and $haty_i$ is defined as the prediction for the input x_i'. A lower cross validation score corresponds to a better model fit. [3]

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 H is numerically expensive to compute. In the GCV score, the diagonal elements of the hat matrix h_{ii} are replaced by the mean of the trace of H. The GCV score is then given by

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

The numerical advantage comes from the fact that the trace of a product of matrices is not changed when cyclically permuting the product of matrices, i.e. $\mathrm{trace}(H) = \mathrm{trace}(X(X'X)^{-1}X') = \mathrm{trace}(X'X(X'X)^{-1})$. The trace can therefore be computed from the product of two matrices of shape $p \times p$. [2]

2.1.3.2 Model Selection Techniques

To make use of the various model choice criteria, some algorithmic approach to model selection needs to be given. The most commonly used are given below. We always start with a candidate model. [2]

2.1.3.2.1 Forward Selection

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

2.1.3.2.2 Backward selection

We 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 choice criteria. The algorithm stops, if no further reduction is possible.

2.1.3.2.3 Step-wise Selection

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.

2.1.3.3 Regularization

Model selection can also be achieved using regularization techniques. In general, regularization restricts the parameter space by adding some penalty term depending on the complexity of the model. This leads to the penalized least squares criterion

$$\text{PLS}(\beta) = \|y - X\beta\|^2 + \lambda * \text{pen}(\beta)$$

where λ is the so-called smoothing parameter and $\text{pen}(\beta)$ is the penalty term. The two most commonly used forms of regularization are the Ridge Regression and the Lasso Regression. Both are explained in detail in the following.

2.1.3.3.1 Ridge Regression

In Ridge regression, the penalty term in the penalized least squares criterion is given by the squared L_2-norm of the coefficient vector β. The objective function to minimize is therefore given by

$$\text{PLS}(\beta) = \|y - X\beta\|^2 + \lambda\beta'\beta$$

which the closed form solution

$$\hat{\beta}_{PLS} = (X'X + \lambda I_p)^{-1}X'y,$$

where $I_p \in \mathbb{R}^{p\times p}$ is the identity matrix. The additional penalty term in Ridge regression leads to smaller parameter estimates $\hat{\beta}_{PLS}$ compared to the un-penalized estimate $\hat{\beta}_{LS}$. For large values of the smoothing parameter λ, the parameter estimates will be converge towards, but never reach, zero.

Ridge regression is commonly used for design matrices X that are close to nonlinear or when the input dimension is high. [4]

2.1.3.3.2 Lasso Regression

One drawback of Ridge regression is that it does not produce sparse solutions, i.e. all estimated coefficients will be different from zero. Lasso regression tackles this problem by the use of the L_1-norm as penalty term. The penalized least squares objective function is then given by

$$\text{PLS}(\beta) = \|y - X\beta\| + \lambda * \sum_{i=1}^{k} |\beta_i|,$$

where λ again plays the role of a smoothing parameter. No closed form solution is available for Lasso regression. The Lasso estimates $\hat{\beta}_{Lasso}$ are calculated using either quadratic programming techniques, given in [5], or least angle regression, given in [6]. Ridge regression penalized large coefficients much stronger than Lasso regression, while for small coefficient values, the Lasso penalty has much more influence. [5]

2.2 Splines

In it's most general definition a spline is a piece-wise polynomial defined on a sequence of knots. This definition is quite general. Therefore, there exist a large variety of splines, ranging from regression splines in [7], over B-splines in [8] to natural cubic splines and many more.

2.2.1 B-Splines

We lay the focus on the definition and use of B-splines, which are constructed from polynomial pieces in a recursive manner. The following figure shows a simple example of a B-spline of degree 1 on the left. It consists of two linear pieces, one from x_1 to x_2 and the other from x_2 to x_3. Everywhere else, the B-spline is equal to zero. This locality is a very attractive feature of B-splines. In the right part of the figure, a B-spline of degree 2 is shown, which consist of three quadratic pieces. At the joining points, the values as well as the first derivatives of the quadratic pieces are equal.

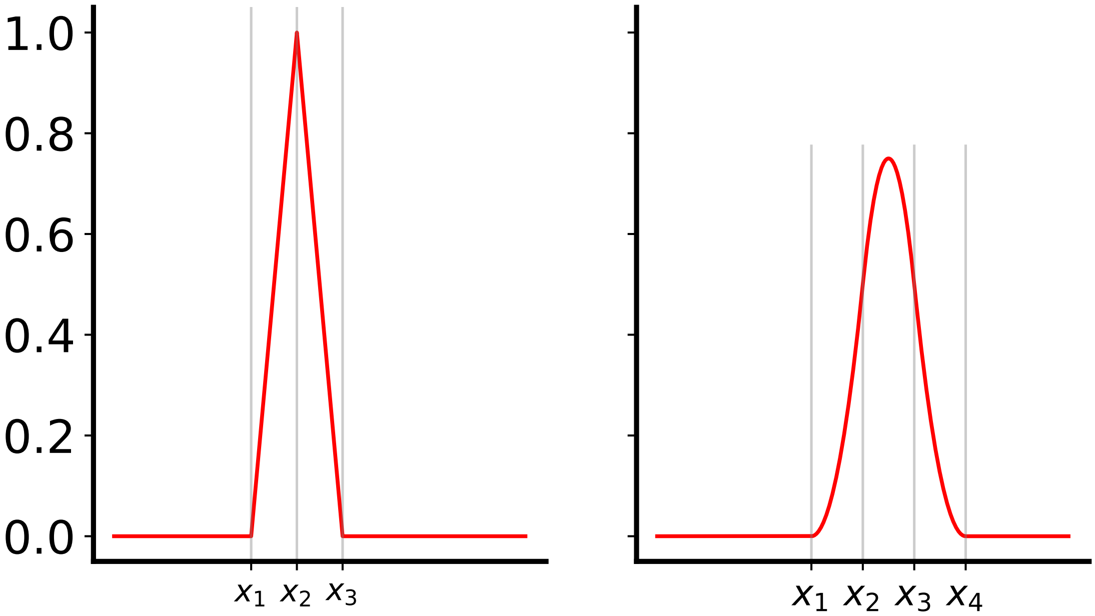

Abbildung 2.1: Linear and Quadratic Spline

The general properties of a B-spline of degree m are the following:

- It consists of $m + 1$ polynomial pieces of degree m, e.g. a cubic spline $(m = 3)$ consists of 4 cubic pieces.
- The pieces join at m inner knots.
- At these knots, the derivatives up to order $m - 1$ are continuous.

- The B-spline is positive on the domain spanned by $m + 2$ knots, everywhere else it is zero.
- At every given x, only $m + 1$ B-splines are non-zero.

The collection of k B-splines of degree m over a sequence of $k + 2(m - 1)$ knots is called B-spline basis. The $2(m - 1)$-knots are the boundary knots while the k 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.

A smooth function can then be represented using the basis function approach given by

$$f(x) = \sum_{i=1}^{k} B_i^m(x)\beta_i$$

using the B-spline basis $B_i^m(x)$ which is defined recursively using the knot sequence $x_i = \{x_1, \ldots, x_{k+2(m-1)}\}$ according to [8] as follows:

$$B_i^m(x) = \frac{x - x_i}{x_{i+m} - x_i} B_i^{m-1}(x) + \frac{x_{i+m+1} - x}{x_{i+m+1} - x_{i+1}} B_{i+1}^{m-1}(x)$$

and

$$B_i^0(x) = \begin{cases} 1, & x_i \leq x < x_{i+1} \\ 0, & \text{otherwise} \end{cases}.$$

The use of B-splines as basis functions for uni-variate or non-parametric regression is very attractive. A linear combination of cubic B-splines gives a smooth curve (first and second order derivatives are continuous). A further advantage of B-splines is that once the basis is given, the coefficients can be estimated using the Least Squares algorithm. The least squares formulation for splines is the given by

$$Q(\beta) = \|y - X\beta\|^2$$

for the B-spline basis $X \in \mathbb{R}^{n \times k}$ for n data points and k splines, which leads to the estimated coefficients

$$\beta_{LS} = (X^T X)^{-1} X^T y$$

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 calculate e.g confidence intervals for the regression coefficients and the prediction.

B-splines of appropriate order produce smooths curves, where the smoothness is determined by the number of splines used. For a low number, the curve will be quite smooth, but with a large data error. When using a high number of splines, 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. It is therefore necessary to introduce some kind of regularization. [8]

2.2.2 P-Splines

P-splines where introduced by Eilers and Marx in [9] to tackle the problem introduced above. Eilers and Marx simplified and generalized the idea of [**osullivan1986statistical**], who introduced a penalty on the integral of the squared second derivative of the estimated spline to penalized wiggly function estimates. Eilers and Marx proposed to use equidistant knots and to base the penalty on finite differences of order d of the coefficients of adjacent B-splines. In [**osullivan1986statistical**], O'Sullivan implicitly used a modified second-order penalty. The finite differences of order 1 for k splines are given by

$$\Delta^1 \beta = \sum_{j=2}^{k-1} \beta_j - \beta_{j-1}$$

and in matrix form

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

This leads to the penalized least squares formulation

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

where $\|y - X\beta\|^2$ is the mean squared error on the data for the spline smooth, $\mathcal{J}_s(\beta; d) = \beta^T D_d^T D_d \beta$ is the smoothness penalty term and λ_s is the smoothness parameter effect of the smoothing penalty. The estimated coefficients are then given by

$$\beta_{LS,p} = (X^T X + \lambda_s D_d^T D_d)^{-1} X^T y$$

The main advantage of P-splines is their easy set up. This advantage is diminished if uneven knot placement is chosen.

2.2.3 Tensor-Product Splines

Tensor-product splines can be seen as the multi-dimensional extension of univariate splines. The basic approach is to start with a spline basis for each dimension and construct the tensor-product spline from these.

We examine an example for two input dimensions x_1 and x_2. Assume that we have bases available for representing the functions $f_1(x_1)$ and $f_2(x_2)$ given by

$$f_1(x_1) = \sum_{i=1}^{k_1} \alpha_i a_i(x_1), \quad f_2(x_2) = \sum_{j=1}^{k_2} \beta_j b_j(x_2),$$

where $\alpha_i \in \mathbb{R}^{k_1}$ and $\beta_j \in \mathbb{R}^{k_2}$ are the coefficients and $a_i(x_1)$ and $b_j(x_2)$ are the known basis functions. 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

$$\alpha_i(x_2) = \sum_{j=1}^{k_2} \beta_{ij} b_j(x_2)$$

which leads to

$$f_{1,2}(x_1, x_2) = \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} \beta_{ij} b_j(x_2) a_i(x_1).$$

For any set of data (x_{1i}, x_{2i}) for $i = 1, \ldots, n$ there is a relationship between the model matrix X of the tensor-product smooth and the marginal model matrices X_1 and X_2, which is given by

$$X_i = X_1 \otimes X_2.$$

where $\otimes$ indicates the use of the Kronecker product, X_1 denotes the B-spline basis for dimension 1 and X_2 denotes the B-spline basis for dimension 2. [**wood2006GAM**] The model matrix for the tensor-product smooth is therefore given by the Kronecker product of the marginal model matrices.

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 if the enormous increase in used basis functions. A smoothness penalty term for tensor-product splines can also be constructed using the Kronecker product. Further explanations are given in Chapter 3.

2.3 Structured Additive Regression

We have again some given data $\{x_{i1}, \ldots, x_{iq}; y_i\}$ for $i = 1, \ldots, n$, where we want to model the generally non-linear relationship between the input data $\{x_{i1}, \ldots, x_{iq}\}$ and the output y by some function $f(x_1, \ldots, x_q)$. Using, e.g. high-dimensional tensor-product splines, to model the function is computationally expensive, since the number of regression coefficients increases exponentially.

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

$$f(x_1, \ldots, x_q) = f_1(x_1) + \cdots + f_q(x_q). \tag{2.18}$$

Hence, we use one smooth function per input dimension and assume an additive structure. [2] Using the concepts introduced in Chapter 2.2, we obtain for each smooth function a linear model

$$f_i(x_i) = X_i \beta_i$$

where $X_i \in \mathbb{R}^{n \times k_i}$ is the B-spline basis using k_i splines for input dimension i and $\beta_i \in \mathbb{R}^{k_i}$ are the coefficients to be estimated. We can also use the already described penalization approaches given in Chapter 2.2.2.

The model given in Eq. 2.18 does not contain interaction terms between variables. Nevertheless, these can be easily introduced for 2 dimensions using tensor-product splines without an overflowing increase of the number of regression coefficients.

We can then write the structured additive model in matrix notation as

$$y = X_1\beta_1 + \cdots + X_q\beta_q + \sum_{i=1}^{n_{interact}} X_{tps,i}\beta_{tps,i} + \epsilon \tag{2.19}$$

using the error term ϵ and $n_{interact}$ as the number of interactions to include via tensor-product spline bases $X_{tps,i}$. This can be solved using ordinary least squares. If we choose to include penalization, we can solve this using penalized least squares.

Using the notation in Eq. 2.19 the theoretical framework of linear models can be applied to structured additive regression models. Therefore, the assumptions given in Chapter 2.1.1 on the error term, as well as on the model functions are used. [10]

3 Mathematical Framework

3.1 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

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

Figure 3.1 shows a B-spline smooth using $k = 10$ 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 $\hat{\beta}_{LS}$.

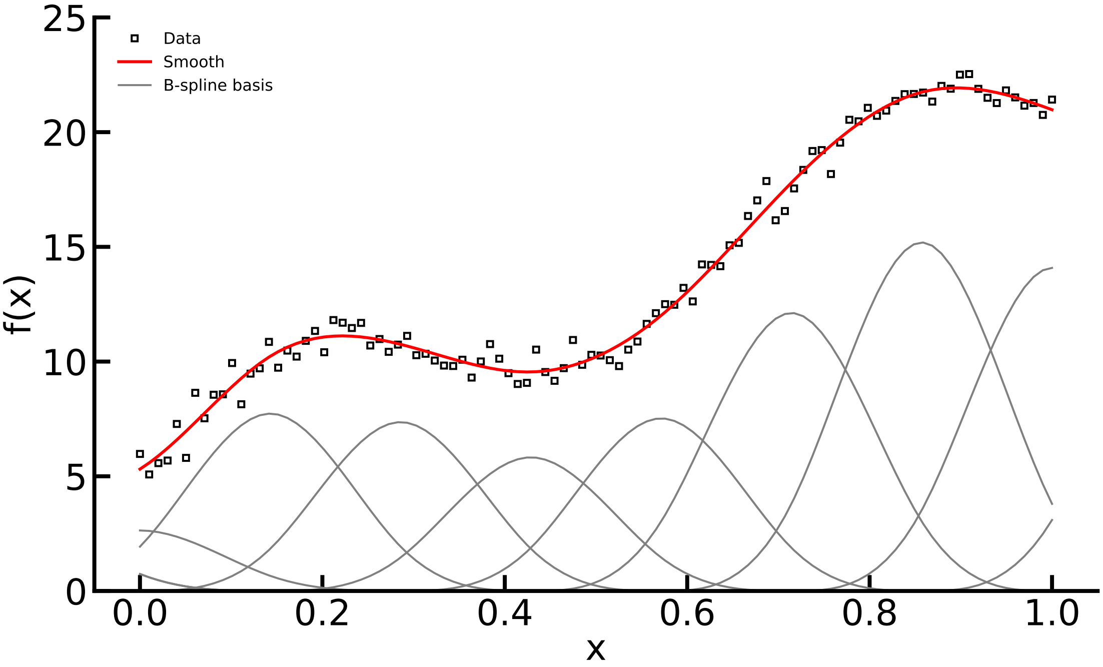

Abbildung 3.1: B-spline smooth and basis functions

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 an example of the bias-variance dilemma and depicted in Figure 3.2. [11]

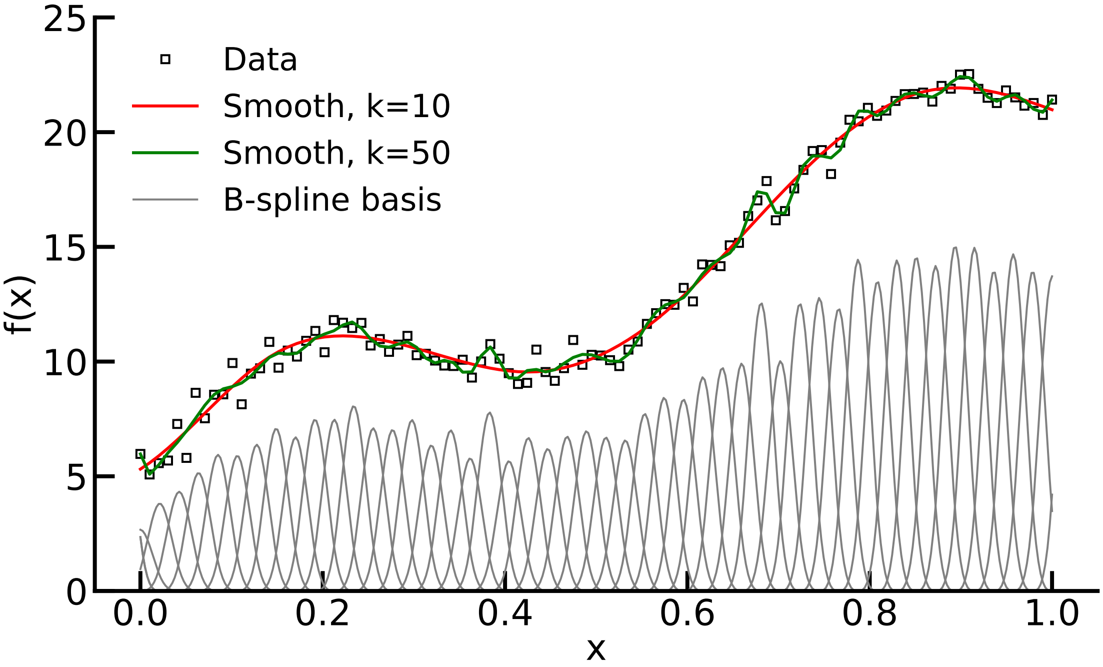

Abbildung 3.2: B-spline smooth and basis functions for larger number of splines

To overcome this, Eilers and Marx 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. [9]

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 effect of the penalty. The matrix D_d is called penalty matrix. 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. [2]

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. [2]

Figure 3.3 shows the behavior of P-splines using $k = 50$ 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, i.e. the 2^{nd} derivative is smaller, and finally, for very large values of λ, it approaches a straight line.

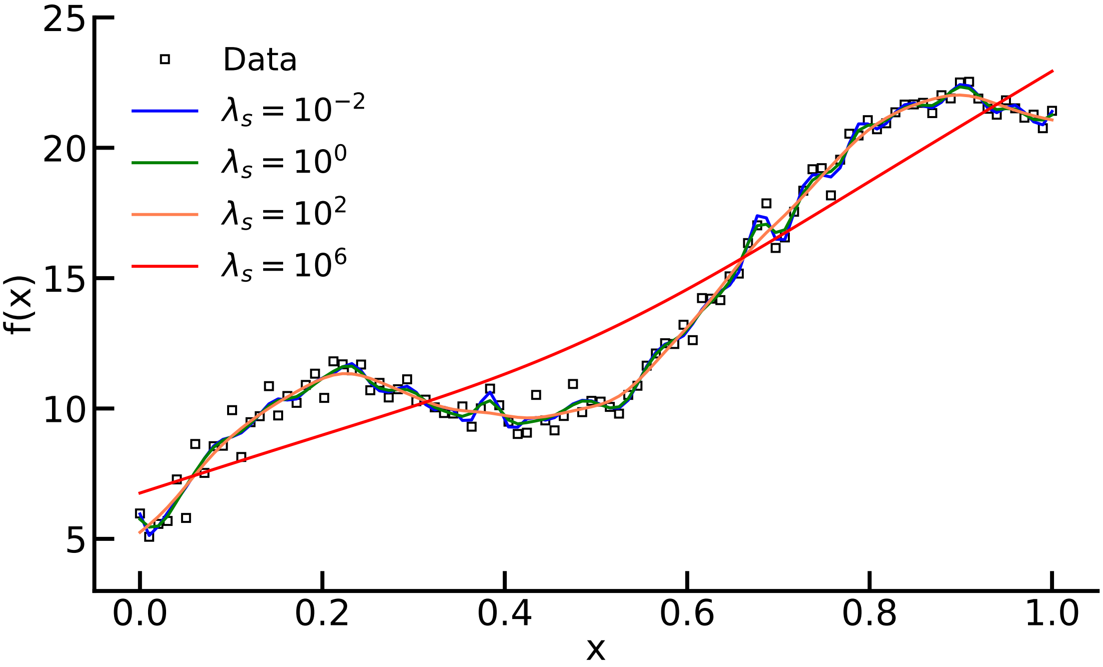

Abbildung 3.3: P-spline smooth for different λ_s

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. [12]

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

$$Q(y; \beta) = \|y - X\beta\|^2 + \lambda_s \mathcal{J}_s(\beta; d) + \lambda_c \mathcal{J}_c(\beta; c) \tag{3.1}$$

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 constrained 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 unconstrained least squares estimate $\hat{\beta}_{LS}$ is a valid candidate. Now the calculation of the constrained 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*. [12]

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.

Figure 3.4 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 constrained, red smooth follows the a priori known behavior of monotonicity far better than the blue, unconstrained smooth.

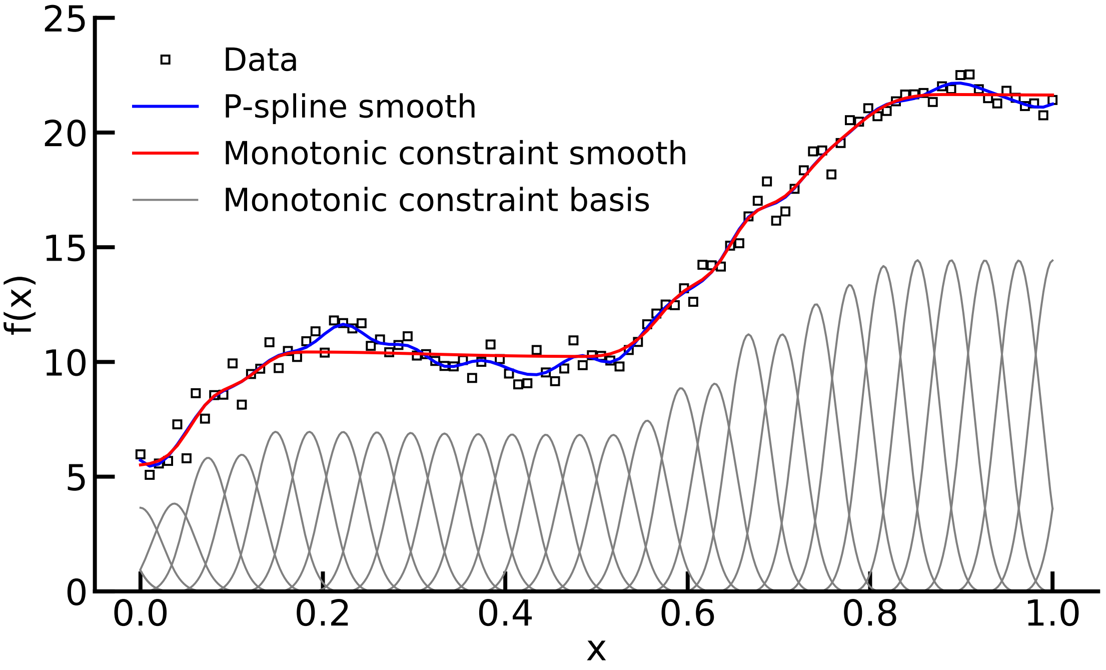

Abbildung 3.4: Monotonicity constrained smooth

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 as well as the weight matrix V 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.

3.2 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.

3.2.1 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. [12] [13]

3.2.2 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

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

Therefore, the penalty is only applied if the second order difference of adjacent coefficients is either positive or negative, respectively. [13]

3.2.3 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. First, we need to find the index j_{peak} of the spline, which has the maximal value around this peak. The index j_{peak} is now used as splitting point for the weight matrix V. All coefficients j for $j < j_{peak}$ are constrained to be monotonic increasing, while all coefficients j for $j > j_{peak}$ are constrained to be monotonic decreasing. The coefficient at position j_{peak} stays unconstrained. [13] 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:

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

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 j_{valley}, then constraining all splines for $j < j_{valley}$ to be monotonic decreasing and all splines for $j > j_{valley}$ to be monotonic increasing. The coefficient at position j_{valley} stays unconstrained. The weights v_j for the weight matrix are the given by

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

3.2.4 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.

3.2.5 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 Kronecker products.

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 2. Using the Kronecker product, we generate the expanded difference matrix $D^1_{d,exp} = I_{d_2} \otimes D^1_d$ for I_{d_2} as identity matrix of dimensions $d_2 \times d_2$ and $D^2_{d,exp} = D^2_d \otimes I_{d_1}$ 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 $D^1_{d,exp}$ and $D^2_{d,exp}$ 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 for the constraint given by c then has the form

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

with $D^1_{c,exp} = I_{d_2} \otimes D^1_{unimodal}$ and $D^2_{c,exp} = D^2_{unimodal} \otimes I_{d_1}$ as individual penalty terms and V_1 and V_2 as weight matrices. The same constrained penalized least squares objective function as in Equation 3.1 can now be used to estimate the coefficients $\beta_{PLS,c}$. [2]

3.3 Other constraints

3.3.1 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 constrained 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 constrained to be larger than 2, can also be enforced.

3.4 Extension to Multiple Dimensions

The extension from one input to multiple inputs uses the concept of additive models given in Chapter *Additive Models*. Given input data $\{x_{i1}, \dots, x_{ip}, y_i\}$ for $i = 1, \dots, n$ and p as the number of inputs, 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 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 $k_{smooths} = p * k + \frac{p(p-1)}{2} k^2$. [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

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

The data term in the constrained penalized least squares objective function given in Equation 3.1 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 $\mathcal{J}_s(\beta, d)$ is now given as

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

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. d is a vector, determining the order d_i of the smoothness constraint for each individual smooth i.

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

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

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

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

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

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

$$f(x_1, x_2) = 2 \exp\big(-\frac{(x_1 - 0.25)^2}{0.08}\big) + x_2^2$$

for $x_1 \in [0, 1]$ and $x_2 \in [0, 1]$ and random Gaussian noise with $\sigma_{noise} = 0.1$. We therefore expect a peak in dimension 1 as well as increasing behavior for dimension 2. Using this knowledge, we create a model using the following smooths:

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

The fit for this model as well as the individual smooths $s_1(x_1)$ and $s_2(x_2)$ is shown in Figure 3.5. The model fits the data quite well and holds the specified constraints for the individual smooths.

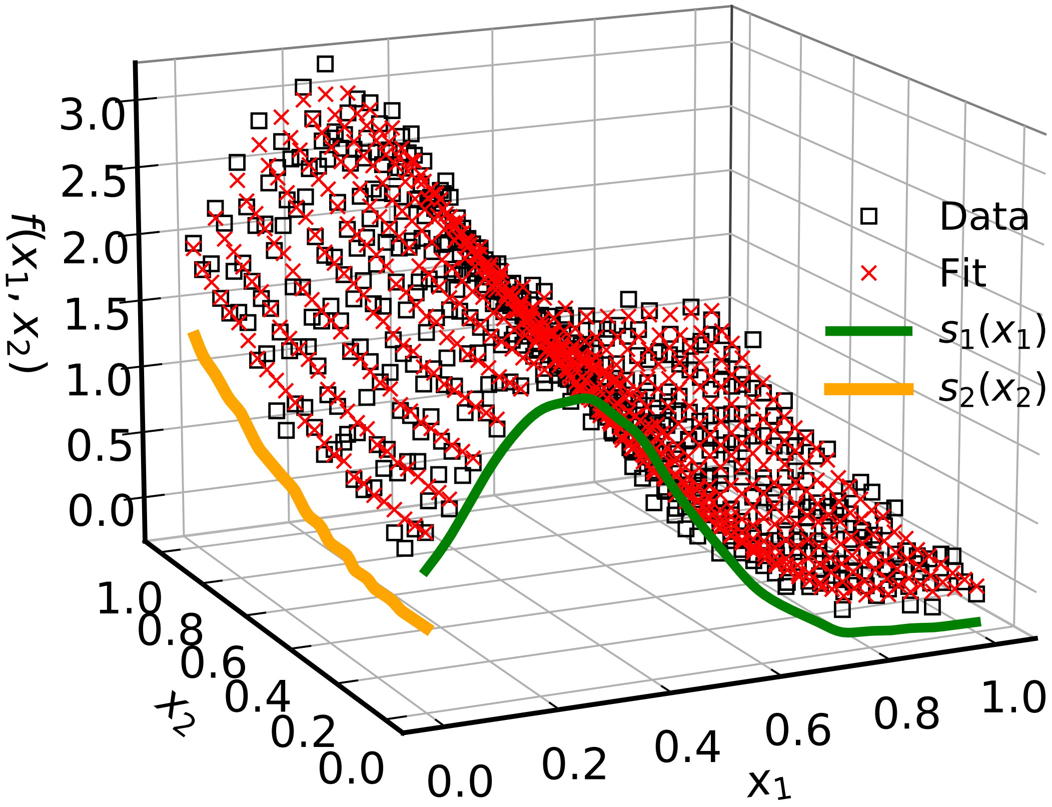

Abbildung 3.5: Test function for constrained penalized least squares fit in 2 dimension

4 Results

This chapter is reserved for many plots and results.

Should contain the following:

- Equidistant vs. Non-equidistant Knot Placement
- Grid vs. non-grid data
- Different Noise Levels
- Phasendiagram: $\sigma^2 vs. \lambda_c$
- Ebner Data

5 Outline

Just the outline. E.g. adaptive knot placement (as in source), using Genetic Algorithms to find the best model structure, etc..

Literatur

[1] C. M. Bishop, *Pattern recognition and machine learning.* springer, 2006.

[2] L. Fahrmeir, T. Kneib, S. Lang und B. Marx, *Regression models.* Springer, 2013, S. 21–72.

[3] G. H. Golub, M. Heath und G. Wahba, „Generalized cross-validation as a method for choosing a good ridge parameter“, *Technometrics*, Jg. 21, Nr. 2, S. 215–223, 1979.

[4] A. E. Hoerl und R. W. Kennard, „Ridge regression: Biased estimation for nonorthogonal problems“, *Technometrics*, Jg. 12, Nr. 1, S. 55–67, 1970.

[5] R. Tibshirani, „Regression shrinkage and selection via the lasso“, *Journal of the Royal Statistical Society: Series B (Methodological)*, Jg. 58, Nr. 1, S. 267–288, 1996.

[6] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani u. a., „Least angle regression“, *The Annals of statistics*, Jg. 32, Nr. 2, S. 407–499, 2004.

[7] R. L. Eubank und C. H. Spiegelman, „Testing the goodness of fit of a linear model via nonparametric regression techniques“, *Journal of the American Statistical Association*, Jg. 85, Nr. 410, S. 387–392, 1990.

[8] C. De Boor, C. De Boor, E.-U. Mathématicien, C. De Boor und C. De Boor, *A practical guide to splines.* springer-verlag New York, 1978, Bd. 27.

[9] P. H. Eilers und B. D. Marx, „Flexible smoothing with B-splines and penalties“, *Statistical science*, S. 89–102, 1996.

[10] L. Fahrmeir, T. Kneib und S. Lang, „Penalized structured additive regression for space-time data: a Bayesian perspective“, *Statistica Sinica*, S. 731–761, 2004.

[11] C. Sammut und G. I. Webb, *Encyclopedia of machine learning.* Springer Science & Business Media, 2011.

[12] B. Hofner, J. Müller und T. Hothorn, „Monotonicity-constrained species distribution models“, *Ecology*, Jg. 92, Nr. 10, S. 1895–1901, 2011.

[13] P. H. Eilers, „Unimodal smoothing“, *Journal of Chemometrics: A Journal of the Chemometrics Society*, Jg. 19, Nr. 5-7, S. 317–328, 2005.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass die vorliegende Arbeit gemäß dem Code of Conduct – Regeln zur Sicherung guter wissenschaftlicher Praxis (in der aktuellen Fassung des jeweiligen Mitteilungsblattes der TU Wien), insbesondere ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel, angefertigt wurde. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In– noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

<Vienna, 23. September 2020>

<Jakob Weber>