Nonparametric logistic regression using smoothing splines

by

Morne Joubert

Submitted in partial fulfillment of the requirements for the degree

Magister Scientiae

In the Faculty of Natural & Agricultural Sciences

University of Pretoria

Pretoria

31 August 2012

SUPERVISORS: DR F KANFER and MR S MILLARD

Declaration

I, Morne Joubert declare that the thesis/dissertation, which I hereby submit for the degree MSc Mathematical Statistics at the University of Pretoria, is my own work and has not previously been submitted by me for a degree at this or any other tertiary institution.

Signature: ..…….. Date: ……..

Abstract

Logistic regression is a well-established technique for modeling a discrete response variable as a function of explanatory variables. The aim of this study is to introduce nonparametric regression using smoothing splines. Nonparametric regression yields a more flexible way of estimating curves and provides a larger set of functions to work from, which is not limited to a family of functions as in the parametric regression framework.

Nonparametric regression falls into the framework of the general additive model with the natural cubic spline as the solution to the penalised least square criterion. Models are fitted using natural cubic splines. B-splines will be implemented due to their computational advantages gained from their almost orthogonal structure.

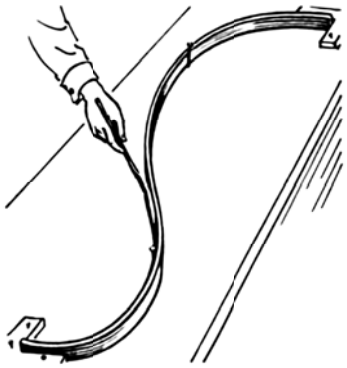
Keywords:

Software, PROC IML within SAS, will be written to estimate these models.

Table of Contents

1. Introduction

In earlier days, architects used to draw curves by hand. They made use of a tool called a **spline,** also called a flexible curve**,** that consists of a long strip fixed in a position at a number of points that relaxes to form and hold a smooth curve passing through those points. The curve, say f, is then traced to paper.



http://upload.wikimedia.org/wikipedia/commons/f/fd/Spline_%28PSF%29.png

Figure 1 Spline

The elasticity of the spline and the control points will cause the spline to take on the shape that minimises the energy required for bending it between the fixed points; this being the smoothest possible shape.

One can measure the energy strain applied to the wood by means of the following

$$\int_{\mathbf{a}}^{\mathbf{b}} \{\mathbf{f}''(\mathbf{t})\}^{\mathbf{2}}\mathbf{dt}. \qquad (1.1)$$

Usually, in a regression setting, the least squared criterion is minimized to solve the regression. We will now introduce a penalty term to measure curvature when minimising the least squared criterion. The penalised least squared criterion is

$$\sum_{j=1}^{n}[y_j - f(x_j)]^2 + \lambda \int [f''(x)]^2 \, dx, \qquad (1.2)$$

with λ a trade-off between the sum of squared criterion and the energy due to the curvature of the function.

The above mentioned idea will be used together with logistic regression. Logistic regression is a technique for making predictions where the dependant variable is binary and the independent variables are continuous or discrete.

In chapter 2 we will look at some standard techniques that will be used in later chapters. Linear and logistic regression is revised for completeness. A cubic spline, natural cubic splines and B-splines are described here and applied to a nonparametric regression setting.

Chapter 3 focuses on the application of the above mentioned techniques and theory. SAS IML programs were written to produce the examples given in this chapter from first principles, except where it was specified otherwise. (SAS Publishers, (2004)) and (SAS Publishers, (2006)). South African heart disease data will be used to compare Gini coefficients between different models.

In conclusion Chapter 4 focuses on the proofs within the content of the natural cubic splines and smoothing spline by using the B-splines.

2. Standard Techniques

2.1 Introduction

In this chapter we will apply standard techniques that will be used in later chapters. Linear- and logistic regression is revised for completeness sake. A cubic spline, natural cubic splines and B-splines are described here and applied to a nonparametric regression setting. The smoothing spline will be described through nonparametric regression by using natural cubic splines as well as B-splines and the use of these in logistic- and linear regression will be demonstrated.

Let

$\boldsymbol{x} = (1\ x_1 \dots x_k)$: $1 \times k+1$ vector with $x_{ij},\ i = 1, \dots, k$ a set of k explanatory variables and

$\boldsymbol{\beta}` = (\beta_0\ \beta_1 \dots \beta_k)$: $1 \times k+1$ a vector of parameters to be estimated.

For a random sample of size n we have

$$\boldsymbol{Y} = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix}, \qquad \boldsymbol{X} = \begin{pmatrix} 1 & x_{11} & \dots & x_{1k} \\ \vdots & \vdots & \dots & \vdots \\ 1 & x_{n1} & \cdots & x_{nk} \end{pmatrix}$$

with $\boldsymbol{Y}$ a column vector of n response and $\boldsymbol{X}$ a matrix with k corresponding explanatory vectors including an intercept as first column.

It is assumed that $E(\boldsymbol{Y}) = \boldsymbol{X\beta}$.

2.2 Linear regression

The conditional expectation of a response variable is modelled as a linear function in the parameters as a function of the explanatory variables. This function can be used to predict the outcome of variable $\boldsymbol{y}$. The parameters are estimated by using equation 2.2.1.

Let Y is a continuous response variable, the linear regression model is given by

$$\mathrm{E(y)} = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k.$$

The ordinary least square (OLS) estimates for β is given by

$$\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{Y} \qquad (2.2.1)$$

(Hastie, Tibshirani and Friedman (2009), p.44 - 45).

2.3 Binary logistic regression

The expected value of a binary response variable is modelled through a set of explanatory variables. The model can be used to predict the outcome of an event. The parameters in the binary logistic regression model are estimated using the iterative least square equations as stated in equation 2.3.1.

Consider the binary variable Y,

$$Y = \begin{cases} 1 \; if \; an \; event \; of \; \text{interest} \; happens \\ 0 \; \; if \; the \; event \; does \; not \; happen \end{cases}$$

with $P(Y = 1) = \; p$ implying $P(Y = 0) = \; 1 - p, p > 0.$

The logistic regression model is

$$\log(\frac{p}{1-p}) = \; \beta_0 + \; \beta_1 x_1 + \cdots + \beta_k x_k$$

with $\boldsymbol{\beta}` = (\beta_0 \, \beta_1 \ldots \beta_k)$ a vector of regression parameters**.**

Consider a random sample and let

$$\boldsymbol{W}_{n\times n} = \begin{bmatrix} p & 0 & \cdots & 0 & 0 \\ 0 & p & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & p & 0 \\ 0 & 0 & \ldots & 0 & p \end{bmatrix}$$

be a diagonal matrix with p on the diagonal and select the vector $\boldsymbol{p}_{1\times n} = (p \ldots p)$`**.**

The iterative least square estimator for $\boldsymbol{\beta}$ is given by the update rule

$$\widehat{\boldsymbol{\beta}}_{new} = \; (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{W}\,\boldsymbol{z}$$

$$\text{where } \; \boldsymbol{z} = \left(\boldsymbol{X}\widehat{\boldsymbol{\beta}}_{\boldsymbol{old}} + \boldsymbol{W}^{-1}(\boldsymbol{Y} - \boldsymbol{p})\right) \qquad (2.3.1)$$

(Hastie, Tibshirani and Friedman (2009), p.121).

2.4 Cubic splines

A spline can be expressed as a linear combination of a set of basis functions. The whole idea behind this builds upon the idea that the set of splines forms a linear space, which usually has some form of simple base (Grgic (2008), p.13).

In this paragraph we will consider global linear functions, global cubic functions, piecewise constant functions and piecewise linear functions. A cubic spline will also be considered.

The red line in figure 2 presents the nonlinear function, $f(X) = \frac{cos(X)}{2}$. For this discussion, $f(X)$ will be considered as the true model. A random error, that is distributed normally with a variance of 1 and a mean of 0, was added to the function to obtain data as indicated by the blue dots. The purple line represents the fitted linear regression function. The generated data points can be found in Appendix A of this document.

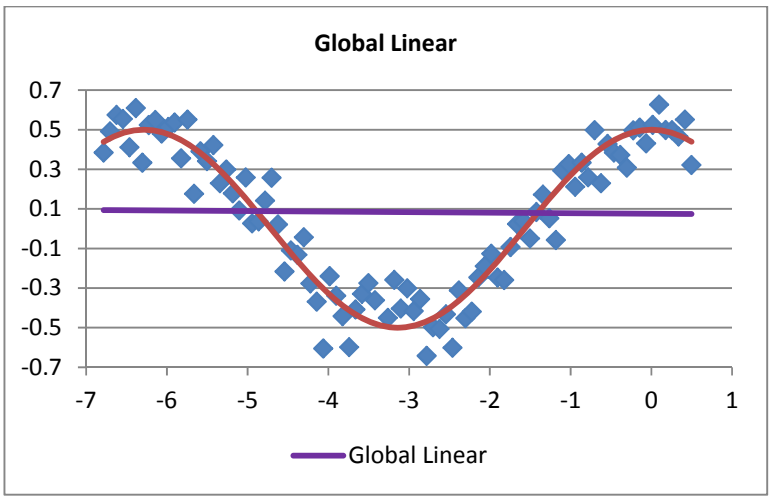


Figure 2 Global Linear

Clearly the straight line is not a good representation of the data, and therefore also not of $f(X)$ (Hastie, Tibshirani and Friedman (2009), p.142).

In order to address the nonlinearity, the vector of inputs $\boldsymbol{X}$ will be replaced with transformations (expansion), $h_m(X)$, $m = 1, \dots, M$ and

$$f(X) = \sum_{m=1}^{M} \beta_m h_m(X). \qquad (2.4.1)$$

The function $f(X)$ is a linear basis expansion in X transformations (expansions) on X, while $h_m(X)$ is the m'th transformation (expansion) of X and β_m being the coefficient of

$h_m(X)$.The vector of regression parameters can be defined as $\beta = (\beta_1 \ldots . \beta_m)$. Typical transformation selections are power transformations; leading to polynomial solutions.

An improvement may be established by replacing the global fit with a localised fit. This will be done by selecting knots which divide the definition region of X into sub regions, representing $f(X)$ by a separate polynomial in each sub region. In figure 3 to figure 7 knots were selected at $\varepsilon_1 = -5$ and $\varepsilon_2 = -1$ indicated with vertical green lines.

Polynomials of different degrees will be considered next, starting with a piecewise constant function, that is $h_1(X) = 1$. This is indicated by the purple lines, piecewise constants, on figure 3 (Hastie, Tibshirani and Friedman (2009), p.142).

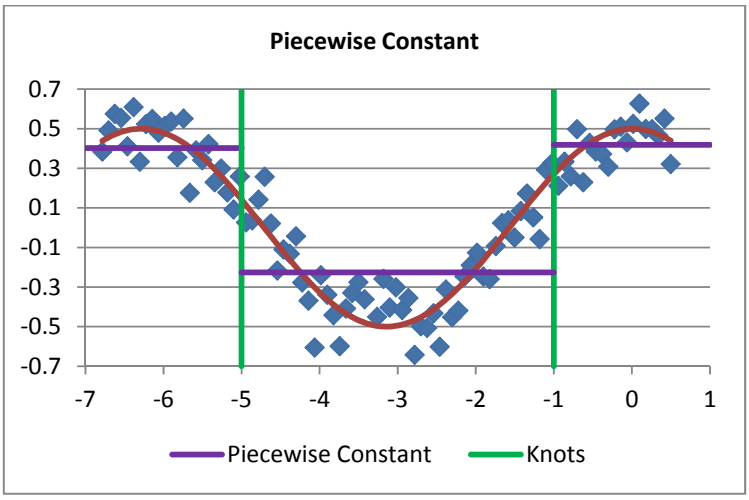


Figure 3 Piecewise Constant

A piecewise linear function (Hastie, Tibshirani and Friedman (2009), p.142) is fitted next. This entails selecting $h_1(x) = 1$ and $h_2(x) = x$. The fitted linear lines are represented in figure 4.

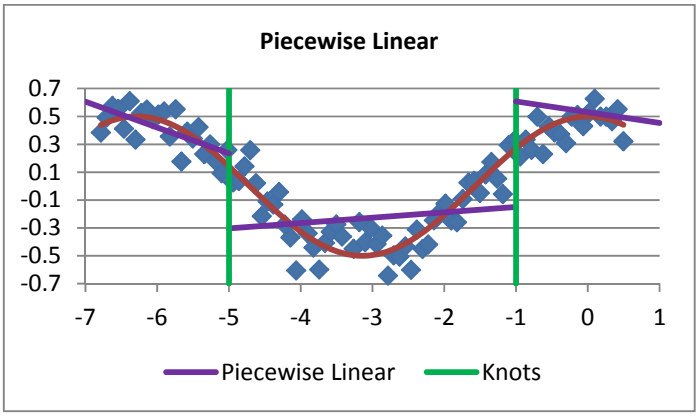


Figure 4 Piecewise Linear

The fitted lines in figure 3 and figure 4 are discontinuous at the knots, which is not ideal.

The discontinuities can be removed by introducing truncated piecewise linear basis function transformations, selecting $h_1(X) = 1,\ h_2(X) = X$, $h_3(X) = (X - \varepsilon_1)_+$ and $h_4(X) = (X - \varepsilon_2)_+$, where t_+ indicates the positive part of t (Hastie, Tibshirani and Friedman (2009), p.142). Figure 5 clearly shows that the piecewise linear transformations are continuous.

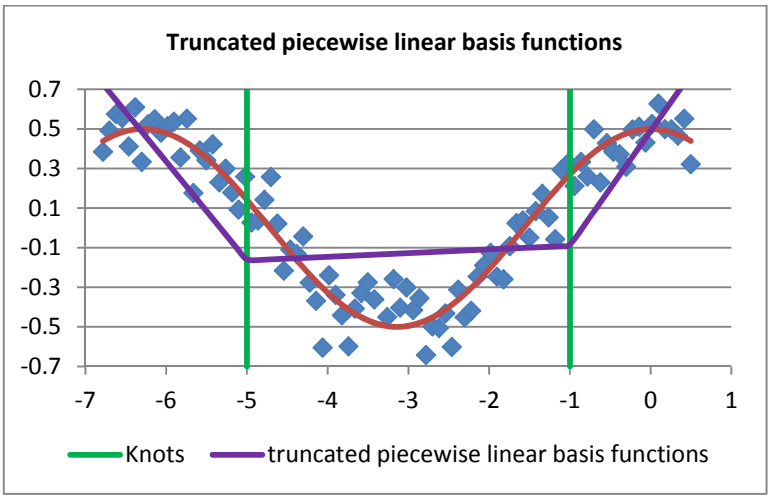


Figure 5 Truncated piecewise linear basis functions

The transformation $h_1(X) = (X - \varepsilon_1)_+$, is given in figure 6 (Hastie, Tibshirani and Friedman (2009), p.142).

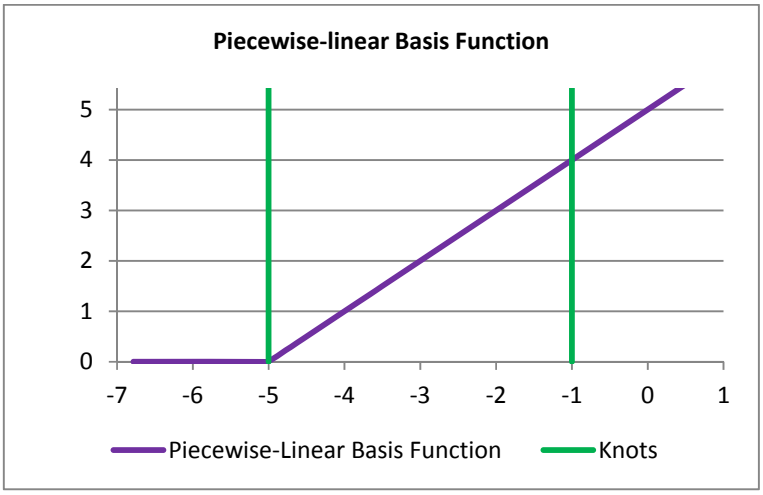


Figure 6 Piecewise-linear Basis Function

The continuous piecewise linear fit in figure 5 is not a smooth fit, which is not preferable and can be improved by increasing the order of the local polynomial.

A cubic spline with knots at ε_1 and ε_2 is suggested (Hastie, Tibshirani and Friedman (2009), p.143).

This involves using the following transformations (basis functions):

$$h_1(X) = 1\ , \qquad h_2(X) = X, \qquad h_3(X) = X^2,$$

$$h_4(X) = X^3, \qquad h_5(X) = (X - \varepsilon_1)_+^3\ , \qquad h_6(X) = (X - \varepsilon_2)_+^3$$

The resulting $f(X)$ is indicated by the purple line in figure 7. Note that the function $f(X)$ is continuous in the second derivative (Hastie, Tibshirani and Friedman (2009), p.143).

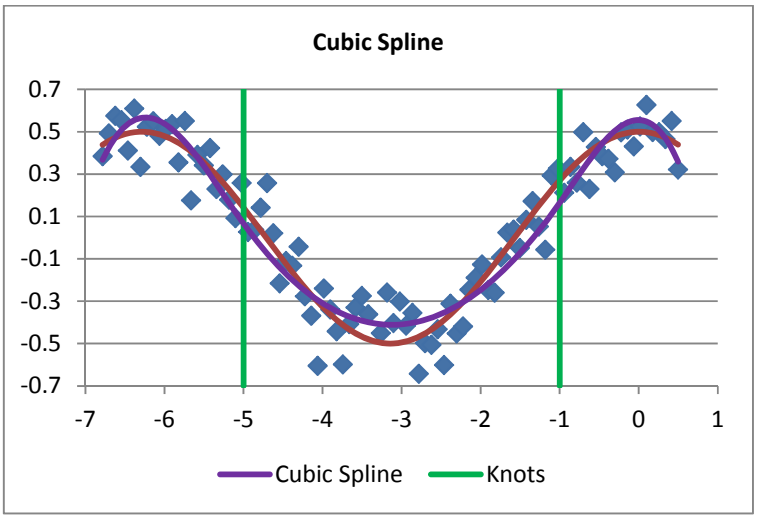


Figure 7 Cubic Spline

The fit of polynomial functions to data tends to be erratic near boundaries. This will be measured by calculating the pointwise variance for different fits to the data.

Let $\boldsymbol{H}_x = [\, h_1(X), \ldots, \, h_M(X)]$.

The pointwise variance was calculated as follows:

The variance matrix of the fitted values with a constant error variance is given by $\boldsymbol{H}\ (\boldsymbol{H^t}\ \boldsymbol{H}\ )^{-1}\boldsymbol{H^t}\boldsymbol{\sigma^2}$ since

$$\begin{aligned} cov(\hat{y}, \hat{y}'\ ) \quad &= \ cov\big(h\hat{b}, \hat{b}'h'\big) \\ &= h\ cov\big(\hat{b}, \hat{b}'\big)h' \\ &= h\ (h'h)^{-1}\sigma^2 h' \\ &= h\ (h'h)^{-1}h'\sigma^2. \end{aligned}$$

Let the pointwise variance be $\boldsymbol{H^X((H^X)^T H^X)^{-1}(H^X)^T}$.

The pointwise variance, for the cubic spline in figure 7, is indicated by the blue line in figure 8. The variance increases dramatically below the knot $\varepsilon_1 = -5$ and above the knot $\varepsilon_2 = -1$. The pointwise variance for the natural cubic spline that is described in section 2.5, is lower in the boundary areas if compared to that of the cubic spline. The natural cubic spline will therefore be used instead of the cubic spline (Hastie, Tibshirani and Friedman (2009), p.145).

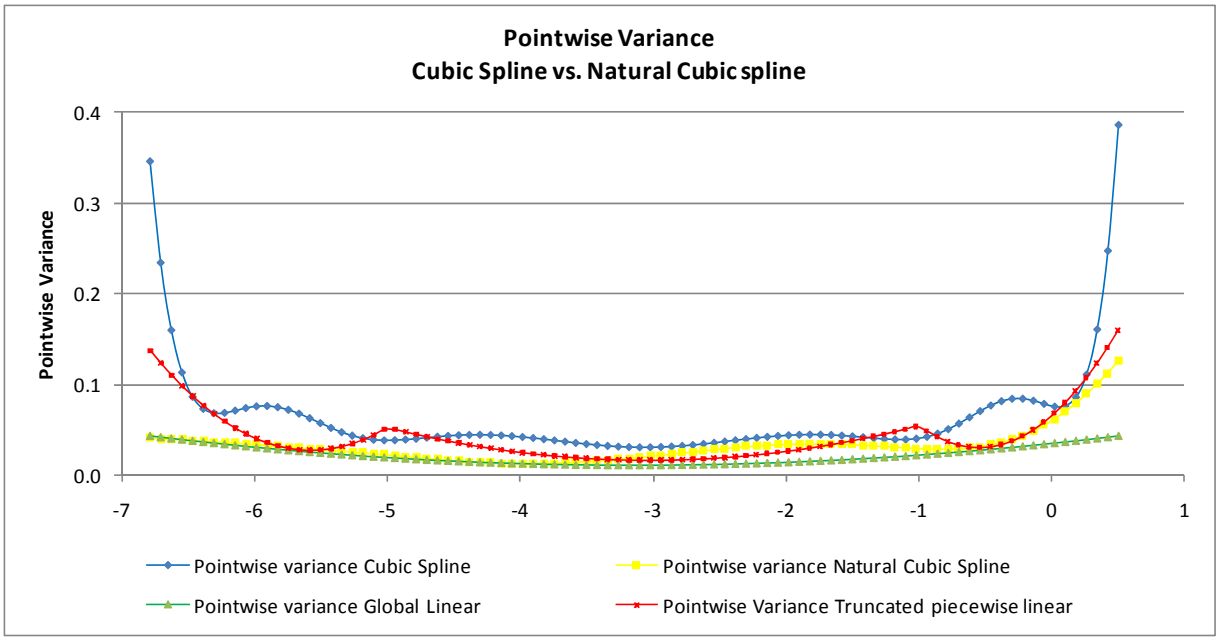


Figure 8 Pointwise variance – Cubic spline vs. Natural Cubic spline

2.5 Natural cubic splines

The natural cubic spline is a cubic spline which is continuous and has continuous first and second derivatives, but with the additional boundary constraints. The additional constraints require that the function is linear beyond the boundary knots.

The basis functions transformation of the natural cubic spline are

$$h_1(X) = 1\,, \quad h_2(X) = X,$$

for order 1 and 2 , and

$$h_{k+2}(X) = d_k(X) - d_{K-1}(X)$$

for order 3 and higher where

$$d_k(X) = \frac{(X-\varepsilon_k)_+^3 - (X-\varepsilon_K)_+^3}{\varepsilon_K - \varepsilon_k} \qquad (2.5.1)$$

(Hastie, Tibshirani and Friedman (2009), p.145).

Example 2.1

A multiple regression was fitted on $h_1(X)$ to $h_4(X)$ in order to obtain the natural cubic spline fit. These basis functions were applied to a dataset containing 94 observations (as previously used in figure 2 to figure 5 and figure 7).

The natural cubic spline transformation was applied to the data using the knots $\varepsilon_1 = -6.8$, $\varepsilon_2 = -5$, $\varepsilon_3 = -1$, and $\varepsilon_4 = 0.5$. The knots ε_1 and ε_4 are boundary knots.

The basis transformations are:

$$\begin{aligned} h_1(X) &= 1, h_2(X) = X \\ h_3(X) &= d_1(X) - d_3(X) \\ &= \frac{(X-\varepsilon_1)_+^3 - (X-\varepsilon_4)_+^3}{\varepsilon_4 - \varepsilon_1} - \frac{(X-\varepsilon_3)_+^3 - (X-\varepsilon_4)_+^3}{\varepsilon_4 - \varepsilon_3} \\ h_4(X) &= d_2(X) - d_3(X) \\ &= \frac{(X-\varepsilon_2)_+^3 - (X-\varepsilon_4)_+^3}{\varepsilon_4 - \varepsilon_2} - \frac{(X-\varepsilon_3)_+^3 - (X-\varepsilon_4)_+^3}{\varepsilon_4 - \varepsilon_3} \end{aligned}$$

Figure 9 shows the estimated natural cubic spline the purple line and $f(X)$ (red line). The estimated spline (purple line) is linear beyond the boundary knots ($\varepsilon_1 = -6.8$ and $\varepsilon_4 = 0.5$).

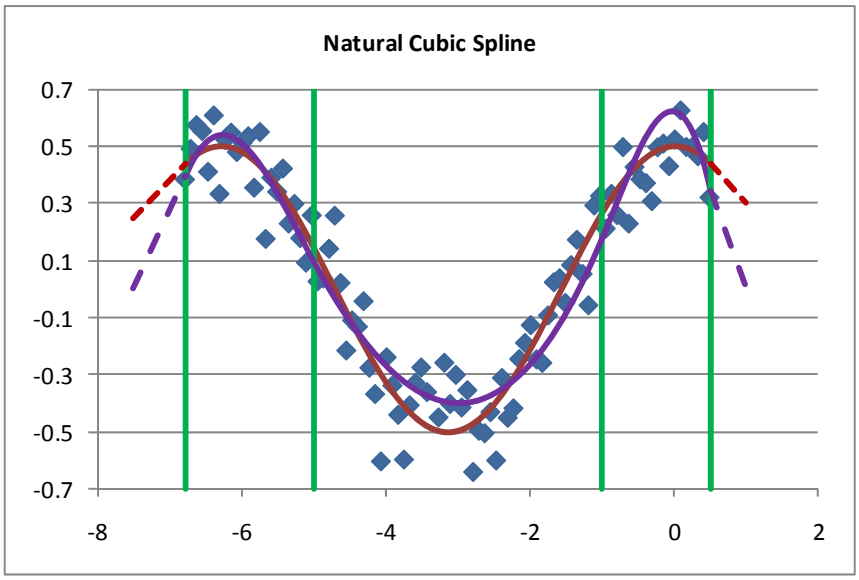


Figure 9 Natural Cubic Spline

2.6 B-splines

In this subsection we will introduce a superior and well-conditioned representation where a spline will be expressed as a linear combination of a set of basis functions, called B-splines. The whole idea behind this builds upon the fact that the set of splines with fixed knots $\varepsilon_1, \dots, \varepsilon_m$ forms a linear space, and as such, usually has some sort of simple base (Grgic (2008), p.13).

"The B-splines are the simplest possible splines of each order and as orthogonal as possible" (Ohlsson and Johannsen (2010) , p.108).

The B-spline transformation of order zero is:

$$\mathrm{B}_{0,i}(t) = \begin{cases} 1 & \text{if } \varepsilon_i \le t < \varepsilon_{i+1} \quad \text{for } i = 1, \dots, m-2 \\ 1 & \text{if } \varepsilon_i \le t \le \varepsilon_{i+1} \quad \text{for } i = m-1 \\ 0 & \text{otherwise} \end{cases} \qquad (2.6.1)$$

The B-splines of order $k+1$ are:

$$\mathrm{B}_{k+1,1}(t) = \frac{\varepsilon_2 - \mathrm{t}}{\varepsilon_2 - \varepsilon_1}\mathrm{B}_{k,1}(t) \qquad (2.6.2)$$

$$\mathrm{B}_{k+1,i}(t) = \frac{t - \varepsilon_{\max(i-k-1,1)}}{\varepsilon_{(\min(i,m))} - \varepsilon_{\max(i-k-1,1)}}\mathrm{B}_{k,\mathrm{i}-1}(\mathrm{t}) + \frac{\varepsilon_{\min(i+1,m)} - t}{\varepsilon_{(\min(i+1,m))} - \varepsilon_{\max(i-k,1)}}\mathrm{B}_{k,i}(\mathrm{t}),$$

$$i = 2, \dots, m+k-1 \qquad (2.6.3)$$

$$\mathrm{B}_{k+1,m+k}(t) = \frac{t-\varepsilon_{\max(m-1,1)}}{\varepsilon_m - \varepsilon_{m-1}} \mathrm{B}_{k,m+k-1}(t) \qquad (2.6.4)$$

(Grgic (2008), p.14).

Example 2.2: Illustration of B-splines

The B-spline transformation will now be illustrated by choosing the 5 knots ($M = 5$) as $\varepsilon_1 = -5, \varepsilon_2 = -3, \varepsilon_3 = 0, \varepsilon_4 = 3$ and $\varepsilon_5 = 5$ and the data in table 1 will be used.

Table 1 Data points for B-spline

t	-5	-4	-3	-2	-1	0	1	2	3	4	5
y	-48	-14	2	6	4	2	6	22	56	114	202

B-splines transformation of order 0 are:

$$B_{0,1}(t) = \begin{cases} 1 & [-5,-3) \\ 0 & \text{otherwise} \end{cases}$$

$$B_{0,2}(t) = \begin{cases} 1 & [-3,0) \\ 0 & \text{otherwise} \end{cases}$$

$$B_{0,3}(t) = \begin{cases} 1 & [0,3) \\ 0 & \text{otherwise} \end{cases}$$

$$B_{0,4}(t) = \begin{cases} 1 & [3,5] \\ 0 & \text{otherwise} \end{cases}$$

B-splines of order one are ($k = 0$):

$$B_{1,1}(t) = \frac{\varepsilon_2 - t}{\varepsilon_2 - \varepsilon_1} B_{0,1}(t)$$

$$B_{1,2}(t) = \frac{t - \varepsilon_1}{\varepsilon_2 - \varepsilon_1} B_{0,1}(t) + \frac{\varepsilon_3 - t}{\varepsilon_3 - \varepsilon_2} B_{0,2}(t)$$

$$B_{1,3}(t) = \frac{t - \varepsilon_2}{\varepsilon_3 - \varepsilon_2} B_{0,2}(t) + \frac{\varepsilon_4 - t}{\varepsilon_4 - \varepsilon_3} B_{0,3}(t)$$

$$B_{1,4}(t) = \frac{t - \varepsilon_3}{\varepsilon_4 - \varepsilon_3} B_{0,3}(t) + \frac{\varepsilon_5 - t}{\varepsilon_5 - \varepsilon_4} B_{0,4}(t)$$

$$B_{1,5}(t) = \frac{t - \varepsilon_4}{\varepsilon_5 - \varepsilon_4} B_{0,4}(t)$$

B-splines of order two are ($k = 1$):

$$B_{2,1}(t) = \frac{\varepsilon_2 - t}{\varepsilon_2 - \varepsilon_1} B_{1,1}(t)$$

$$B_{2,2}(t) = \frac{t - \varepsilon_1}{\varepsilon_2 - \varepsilon_1} B_{1,1}(t) + \frac{\varepsilon_3 - t}{\varepsilon_3 - \varepsilon_1} B_{1,2}(t)$$

$$B_{2,3}(t) = \frac{t - \varepsilon_1}{\varepsilon_3 - \varepsilon_1} B_{1,2}(t) + \frac{\varepsilon_4 - t}{\varepsilon_4 - \varepsilon_2} B_{1,3}(t)$$

$$B_{2,4}(t) = \frac{t - \varepsilon_2}{\varepsilon_4 - \varepsilon_2} B_{1,3}(t) + \frac{\varepsilon_5 - t}{\varepsilon_5 - \varepsilon_3} B_{1,4}(t)$$

$$B_{2,5}(t) = \frac{t - \varepsilon_3}{\varepsilon_5 - \varepsilon_3} B_{1,4}(t) + \frac{\varepsilon_5 - t}{\varepsilon_5 - \varepsilon_4} B_{1,5}(t)$$

$$B_{2,6}(t) = \frac{t - \varepsilon_4}{\varepsilon_5 - \varepsilon_4} B_{1,5}(t)$$

B-splines of order 3 are ($k = 2$):

$$B_{3,1}(t) = \frac{\varepsilon_2 - t}{\varepsilon_2 - \varepsilon_1} B_{2,1}(t)$$

$$B_{3,2}(t) = \frac{t - \varepsilon_1}{\varepsilon_2 - \varepsilon_1} B_{2,1}(t) + \frac{\varepsilon_3 - t}{\varepsilon_3 - \varepsilon_1} B_{2,2}(t)$$

…

$$B_{3,4}(t) = \frac{t - \varepsilon_1}{\varepsilon_4 - \varepsilon_1} B_{2,3}(t) + \frac{\varepsilon_5 - t}{\varepsilon_5 - \varepsilon_2} B_{2,4}(t)$$

…

$$B_{3,7}(t) = \frac{t - \varepsilon_4}{\varepsilon_5 - \varepsilon_4} B_{2,6}(t)$$

Table 2 presents numerical B-spline values for the dataset

(where b01 represents the parameter $\beta_{0,1}$ etc. in table 2)

Table 2 B-spline calculations

t	y	b01	b02	b03	b04	b11	b12	b13	b14	b15
-5	-48	1	0	0	0	1	0	0	0	0
-4	-14	1	0	0	0	0.5	0.5	0	0	0
-3	2	0	1	0	0	0	1	0	0	0
-2	6	0	1	0	0	0	0.666667	0.333333	0	0
-1	4	0	1	0	0	0	0.333333	0.666667	0	0
0	2	0	0	1	0	0	0	1	0	0
1	6	0	0	1	0	0	0	0.666667	0.333333	0
2	22	0	0	1	0	0	0	0.333333	0.666667	0
3	56	0	0	0	1	0	0	0	1	0
4	114	0	0	0	1	0	0	0	0.5	0.5
5	202	0	0	0	1	0	0	0	0	1

b21	b22	b23	b24	b25	b26	b31	b32	b33	b34	b35	b36	b37
1	0	0	0	0	0	1	0	0	0	0	0	0
0.25	0.65	0.1	0	0	0	0.125	0.645	0.2175	0.0125	0	0	0
0	0.6	0.4	0	0	0	0	0.36	0.54	0.1	0	0	0
0	0.266667	0.677778	0.055556	0	0	0	0.106667	0.583611	0.302778	0.006944	0	0
0	0.066667	0.711111	0.222222	0	0	0	0.013333	0.408889	0.522222	0.055556	0	0
0	0	0.5	0.5	0	0	0	0	0.1875	0.625	0.1875	0	0
0	0	0.222222	0.711111	0.066667	0	0	0	0.055556	0.522222	0.408889	0.013333	0
0	0	0.055556	0.677778	0.266667	0	0	0	0.006944	0.302778	0.583611	0.106667	0
0	0	0	0.4	0.6	0	0	0	0	0.1	0.54	0.36	0
0	0	0	0.1	0.65	0.25	0	0	0	0.0125	0.2175	0.645	0.125
0	0	0	0	0	1	0	0	0	0	0	0	1

The calculated B-splines of order 0, 1, 2 and 3 are sketched in figure 10 (Grgic (2008), p.15).

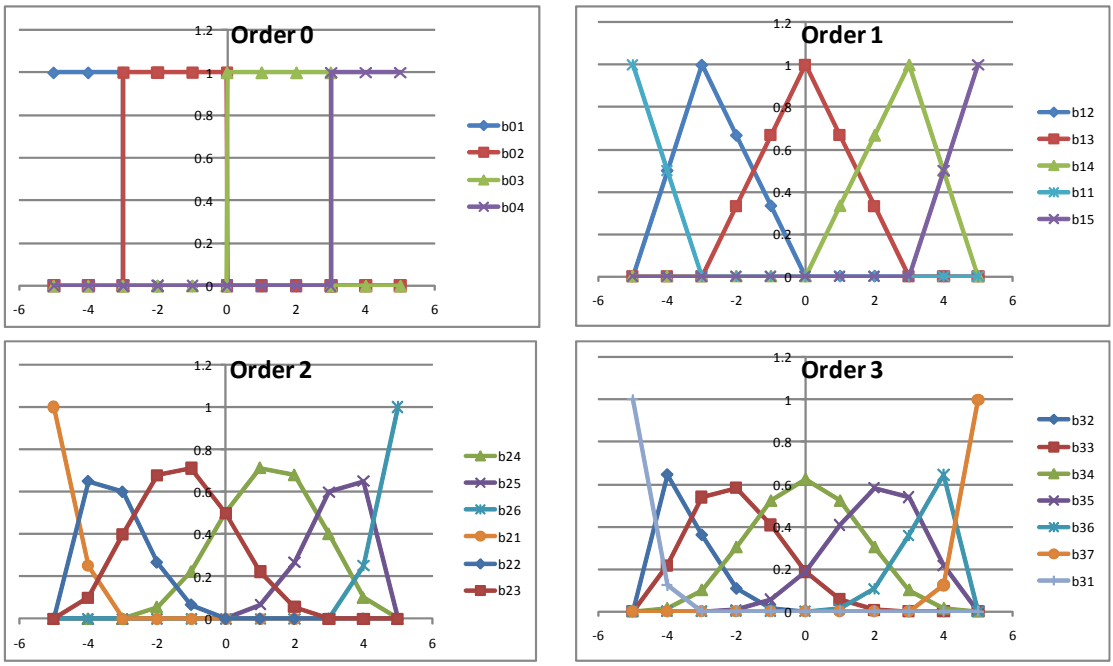


Figure 10 B-splines order 0 to 3

Note the following interesting properties that generally characterise B-splines. They are all positive and locally supported, which means that a k'th order B-spline is strictly positive only on a part of the domain, $(\varepsilon_{\max(i-k,1)}, \varepsilon_{\min(i+1,m)})$. The latter may be seen as some sort of orthogonality and is one of the reasons that makes the B-spline representation well-conditioned especially for calculations. Also note that the B-splines transformations are normalized, i.e. they add up to 1 along the domain, for each order.

The B-splines were used to fit the spline in figure 11 to the data in table 1.

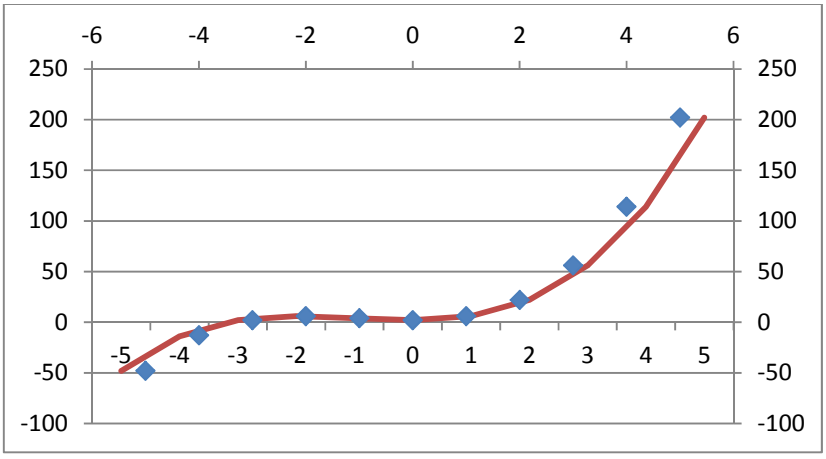


Figure 11 Fitted B-spline

Example 2.3 (continuation of example 2.1)

The fitted curve to the B-spline transformed data of example 2.1 is given in figure 12.

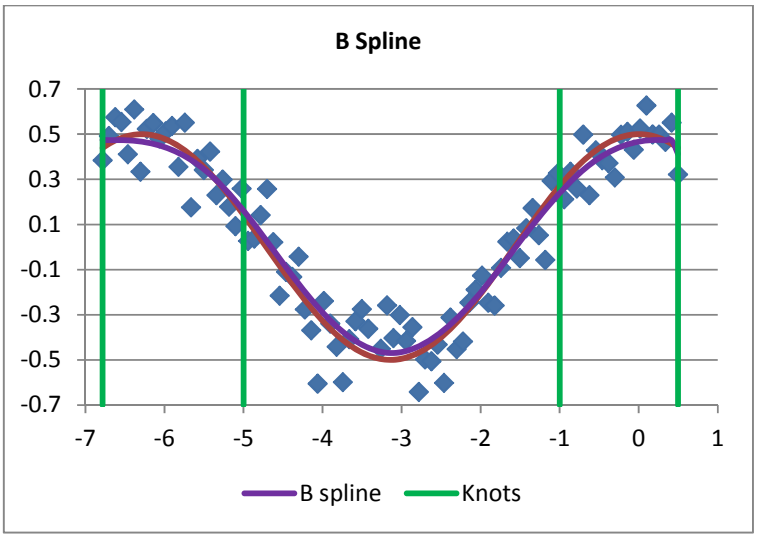


Figure 12 Fitted B-spline (Example 2.3)

2.7 Nonparametric regression

This section overviews traditional nonlinear regression, which is defined for a sample of size n as

$$y_i = f(\boldsymbol{\beta}, \boldsymbol{x_i}) + \varepsilon_i, \qquad i = 1, \ldots, n$$

with $\boldsymbol{\beta} = (\beta_1 \ldots \beta_p)'$ a column vector of regression parameter values and $\boldsymbol{x_i}' = (x_1 \ldots x_k)$ a vector of predictors (explanatory variables) for the ith of n observations. The ith random component ε_i is assumed to be independently normally distributed with mean 0 and constant variance σ^2, $\varepsilon_i \sim NID(0, \sigma^2)$, (Fox (2002), p.1).

The function $f(.)$ is assumed known and relates the predictors to the average value of the response Y over the domain X. Multiple linear regression is a specific case of $f(\boldsymbol{\beta}, \boldsymbol{x_i}) = \boldsymbol{x}'\boldsymbol{\beta}$. This clearly identifies the known pre-specified linear structure as well as the fact that it is a linear function of the parameters. In this case the estimation of $f(\boldsymbol{\beta}, \boldsymbol{x_i})$ only requires the estimation of the regression parameters, $\boldsymbol{\beta}$.

The **general nonparametric regression** model is defined as,

$$\begin{aligned} y_i &= f(\boldsymbol{x_i}') + \varepsilon_i \\ &= f(x_{i1}, x_{i2}, \ldots, x_{ik}) + \varepsilon_i, \; i = 1, \ldots, n \end{aligned}$$

with $f(\boldsymbol{x}')$ unspecified; not known as in the case of traditional regression. It assumes that $f(\boldsymbol{x}')$ is a smooth continuous function with the same assumptions for the errors as traditional regression, that is $\varepsilon_i \sim NID(0, \sigma^2)$. The estimation of the regression function $f(\boldsymbol{x}')$ requires direct estimation rather than the estimation of parameters (Fox (2002), p.1).

Since it is difficult to fit general nonparametric regression model, more restrictive assumptions are usually implemented. Because of the difficulties of the high dimensional predictor space, the following structure

$$f(x_{i1}, x_{i2}, \ldots, x_{ik}) = f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_k(x_{ik})$$

is assumed which simplifies the functional form $f(\boldsymbol{x_i}')$ into the linear structure of univariate smooth functions $f_j(x), j = 1, \ldots, k$. The functions $f_j(x), j = 1, \ldots, k$ are also known as partial regression functions. The model is referred to as the **additive regression** model (Fox (2002), p.1).

Variations of the additive regression model includes, for example:

- semi-parametric model $y_i = \alpha + \beta_1 x_{i1} + f_2(x_{i2}) + \cdots + f_k(x_{ik}) + \varepsilon_i$
- interaction model $y_i = \alpha + f_{12}(x_{i1}, x_{i2}) + f_3(x_{i3}) + \cdots + f_k(x_{ik}) + \varepsilon_i$

(Fox (2002), p.1).

Generalized nonparametric regression is an extension of additive regression models in a similar way as linear models extend to generalized linear models. The random component and link functions (described in Appendix B on page 81) are the same as in generalized linear models, but the linear predictor of the generalized linear model

$$\eta_i = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \varepsilon_i \qquad (2.7.1)$$

is replaced in the most general case by an unspecified smooth function f of the predictors

$$\eta_i = f(x_{i1}, x_{i2}, \ldots, x_{ik}) \ .$$

We will now consider a more constrained case where f is the sum of smooth functions

$$f_j, j = 1, \ldots, k$$

and

$$\eta_i = \alpha + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_k(x_{ik}), \qquad (2.7.2)$$

also known as the generalized additive model or generalized regression additive model, (GAM) (Fox (2002), p.1).

2.7.1 Nonparametric logistic regression

The nonparametric logistic regression model, models the posterior probability of K classes via linear functions in x. The model is specified in terms of $K-1$ logit transformations

$$\log\left(\frac{\Pr(G=1|X=x)}{\Pr(G=K|X=x)}\right) = \propto_{10} + f_1(x)$$

$$\log\left(\frac{\Pr(G=2|X=x)}{\Pr(G=K|X=x)}\right) = \propto_{20} + f_2(x)$$

$$\vdots$$

$$\log\left(\frac{\Pr(G=K-1|X=x)}{\Pr(G=K|X=x)}\right) = \propto_{(K-1)0} + f_{K-1}(x)$$

or

$$\Pr(G=k|X=x) = \frac{\exp(\propto_{k0} + f_k(x))}{1 + \sum_{l=1}^{K-1} \exp(\propto_{l0} + f_l(x))}, k = 1, \ldots, K-1$$

$$\Pr(G = K|X = x) = \frac{1}{1 + \sum_{l=1}^{K-1} \exp(\propto_{l0} + f_l(x))},$$

(Hastie, Tibshirani and Friedman (2009), p.119).

It is known that for the case $K = 2$ the response variable Y has a Bernoulli distribution. In other words, y_i =1 if g_i=1 (if an event occurs for the ith observation) and y_i = 0 if g_i =2 (if an event does not occur for the ith observation) and let $\mu = p(Y = 1|x_{i1}, \dots, x_{ik})$, $i = 1, \dots, n$. In logistic regression μ is linked to the predictors via η_i =log$(\mu/(1-\mu)) = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik}$, which is a linear predictor.

For nonparametric logistic regression η_i =log$(\mu/(1-\mu)) = \alpha + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_k(x_{ik})$ with f_i, $i = 1, \dots, k$ smooth functions, it is clear that this is a special case of the generalized additive model.

2.8 Penalised sum of squares

Consider a set of co-ordinates (x_i, y_i), $i = 1, \dots, n$ and let f be a function such that the second derivative of f exists.

The penalised sum of squares criteria is defined as

$$\sum_{j=1}^{n} [y_j - f(x_j)]^2 + \lambda \int [f''(x)]^2 \, dx. \qquad (2.8.1)$$

The first term in equation 2.8.1 is the same as that of the squared error loss function. The second term is the roughness penalty and will penalise for the curvature of the function. Remember the second term represents the energy into the parameter λ and can be interpreted as follows:

$\lambda = 0$: f can be any function that interpolates the data,

$\lambda = \infty$: this is the simplest least square fit since no second order derivative can be tolerated.

These vary from very rough functions to very smooth functions (Hastie, Tibshirani and Friedman (2009), p.151).

2.9 Nonparametric linear regression by using natural cubic splines

The function which is the optimal solution of the penalised sum of squares is referred to as the smoothing spline. The optimal solution is a natural cubic spline with knots at the

unique observed values of $x = x_1, \ldots, x_n$ (Hastie, Tibshirani and Friedman (2009), p.151-152).

It is proven in section 4 that the natural cubic spline, with knots at each unique observed value, minimises the roughness penalty (Hastie and Tibshirani (1990), p.27). The natural cubic spline can be written as

$$f_i(x) = \sum_{j=1}^{n} \beta_j h_j(x), \qquad (2.9.1)$$

which is the natural basis function expansion. Note that there are n basis functions, h_j, $j = 1, \ldots, n$ which correspond to the n knots; a knot at each observation.

The basis functions $h_1(x), \ldots, h_n(x)$ are compiled over the observation into an $n \times n$ matrix,

$$\boldsymbol{H}_{\boldsymbol{n}\times\boldsymbol{n}} = \begin{bmatrix} h_1(x_1) & \cdots & h_n(x_1) \\ \vdots & \ddots & \vdots \\ h_1(x_n) & \cdots & h_n(x_n) \end{bmatrix}$$

where x_i represent the i'th observation and $h_j(x_i)$ represents the j'th basis function transformation on observation x_i .

Let $\boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$ where y_j, $j = 1, \ldots, n$ to the response for observation j,

$\boldsymbol{x} = (1 \; x_1 \ldots x_k)$ with, x_i, $i = 1, \ldots, k$ a set of k explanatory variables (notice that $k > n$ since transformations are done on each x_i, $i = 1, \ldots, n$),

$\boldsymbol{\beta}` = (\beta_0 \, \beta_1 \ldots \beta_k)$ a vector of parameters.

Let $\boldsymbol{\Omega}$ be an $n \times n$ matrix

$$\boldsymbol{\Omega}_{\boldsymbol{n}\times\boldsymbol{n}} = \begin{bmatrix} \Omega_{11} & \Omega_{12} & \cdots & \Omega_{1n} \\ \Omega_{21} & \Omega_{22} & \cdots & \Omega_{2n} \\ \vdots & & \ddots & \vdots \\ \Omega_{n1} & \Omega_{n2} & \cdots & \Omega_{nn} \end{bmatrix}$$

with $\Omega_{ij} = \int h_i''(x) h_j''(x) dx$, $i = 1, \ldots, n$ and $j = 1, \ldots, n$

The Penalised sum of squares can be written in the natural cubic spline case in matrix notation,

$$\sum_{j=1}^{n} \left[y_j - f(x_j)\right]^2 + \lambda \int \left[f''(x)\right]^2 dx$$

$$= \sum_{j=1}^{n}\left[y_j - \sum_{j=1}^{n}\beta_j h_j(x)\right]^2 + \lambda \int \left[\sum_{j=1}^{n}\beta_j h_j(x)''\right]^2 dx$$

$$= \sum_{j=1}^{n}\left[y_j - \sum_{j=1}^{n}\beta_j h_j(x)\right]^2 + \lambda \int (\sum_{i=1}^{n}\sum_{j=1}^{n}\beta_i\beta_j h_j''(x)h''_i(x))\, dx$$

$$= \sum_{j=1}^{n}\left[y_j - \sum_{j=1}^{n}\beta_j h_j(x)\right]^2 + \lambda \sum_{i=1}^{n}\sum_{j=1}^{n}\beta_i\,\beta_j \int h''_i(x)\, h''_j(x)dx$$

$$= (\boldsymbol{y} - \boldsymbol{H\beta})^T(\boldsymbol{y} - \boldsymbol{H\beta}) + \lambda\boldsymbol{\beta^T\Omega\beta}$$

The penalised sum of square criteria estimator for $\boldsymbol{\beta}$ is given by

$$\widehat{\boldsymbol{\beta}} = (\boldsymbol{H}'\boldsymbol{H} + \lambda\boldsymbol{\Omega})^{-1}\boldsymbol{H}'\boldsymbol{y} \qquad (2.9.2)$$

$$= \boldsymbol{H}\boldsymbol{S}_\lambda\boldsymbol{y} \qquad (2.9.3)$$

with $\boldsymbol{S}_\lambda$ known as the smoother matrix (Hastie, Tibshirani and Friedman (2009), p.152).

2.9.1 Average Mean Squared error

A global measure for choosing the smoothing parameter λ such as the average mean squared error (MSE) will be used (Hastie and Tibshirani (1990), p.42-43).

The MSE measures the discrepancy between a function and an estimated function and is defined as

$$MSE(\lambda) = \frac{1}{n}\sum_{i=1}^{n} E\{\widehat{f_\lambda}(x_i) - f(x_i)\}^2,$$

where $\widehat{f_\lambda}$ is the estimated function.

The average predicted squared error (PSE) is given by

$$PSE(\lambda) = \frac{1}{n}\sum_{i=1}^{n} E\{Y_i^* - \hat{f}(x_i)\}^2 .$$

Let

$$Y_i^* = f(x_i) + \varepsilon_i^* \text{ and } Y_i = f(x_i) + \varepsilon_i,$$

where Y_i^* is a new observation at x_i and not used in the determination of $\hat{f}$. Y_i^* is independent of Y_i since the observations are independent; therefore we have ε_i^* independent of ε_i.

MSE differs from PSE by a constant σ^2 (the residual variance) since

$$
\begin{aligned}
PSE(\lambda) &= \frac{1}{n}\sum_{i=1}^{n} E\{Y_i^* - \hat{f}(x_i)\}^2 \\
&= \frac{1}{n}\sum_{i=1}^{n} E\{Y_i^* - f(x_i) + f(x_i) - \hat{f}(x_i)\}^2 \\
&= \frac{1}{n}\sum_{i=1}^{n} [E\{Y_i^* - f(x_i)\}^2 - E\left(f(x_i) - \hat{f}(x_i)\right)^2 \\
&+ 2E\{Y_i^* - f(x_i)\}E\{f(x_i) - \hat{f}(x_i)\}] \text{ since } \hat{f}(x_i) \text{ is independent of } Y_i^* \\
&= \frac{1}{n}\sum_{i=1}^{n} [E\{f(x_i) + \varepsilon_i^* - f(x_i)\}^2 - E\{f(x_i) - \hat{f}(x_i)\}^2] \text{ since } E(\varepsilon_i^*) = 0 \text{ and } E(\varepsilon_i) = 0 \\
&= \frac{1}{n}\sum_{i=1}^{n} [E\{\varepsilon_i^*\}^2 - E\{f(x_i) - \hat{f}(x_i)\}^2] \\
&= \frac{1}{n}\sum_{i=1}^{n} \sigma^2 + \frac{1}{n}\sum_{i=1}^{n} E\{f(x_i) - \hat{f}(x_i)\}^2 \\
&= \sigma^2 + MSE(\lambda)
\end{aligned}
$$

2.9.2 Cross validation

Cross validation (CV) is a jackknife type of estimator for PSE. CV is obtained by leaving observation i out in calculating the estimate $\hat{f}^{-i}(x)$ for $f(x)$. The estimate $\hat{f}^{-i}(x)$ is based on the remaining $n-1$ points. Note that $\hat{f}^{-i}(x_i)$ is an estimate of y_i and $\hat{f}^{-i}(x_i)$ is obtained by not including (x_i, y_i).

2.9.2.1 Selecting the value of λ

Let $\hat{f}_\lambda$ be an estimation for f obtained under the penalized sum of squares. Indicate the cross validation measure with

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n} \{y_i - \hat{f}_\lambda^{-i}(x_i)\}^2$$

which is an estimate for PSE. By considering $CV(\lambda)$ over a range of λ values, the λ minimizing $CV(\lambda)$ should be selected. To see this we note that

$$
\begin{aligned}
E\{Y_i - \hat{f}_\lambda^{-i}(x_i)\}^2 &= E\{Y_i - f(x_i) + f(x_i) - \hat{f}_\lambda^{-i}(x_i)\}^2 \\
&= \sigma^2 + E\{f(x_i) - \hat{f}_\lambda^{-i}(x_i)\}^2
\end{aligned}
$$

The cross product term in $E\{Y_i - f(x_i)\}E\{f(x_i) - \hat{f}_\lambda^{-i}(x_i)\}$ is zero since $\hat{f}_\lambda^{-i}(x_i)$ doesn't involve Y_i. Similarly it can be shown that $E\{Y_i^* - \hat{f}_\lambda^{-i}(x_i)\}^2 = \sigma^2 + E\{f(x_i) - \hat{f}_\lambda(x_i)\}^2$.

Following this we assume that $\hat{f}_\lambda^{-i}(x_i) \approx \hat{f}_\lambda(x_i)$; leading to $CV(\lambda) \approx PSE(\lambda)$.

It is possible to determine $\hat{f}_\lambda^{-i}(x_i)$ from $\hat{f}_\lambda(x_i)$ by means of a correcting factor. The correcting factor depends only on $\boldsymbol{S}_\lambda$.

Since $\boldsymbol{S}_\lambda$ is constant preserving, that is $\boldsymbol{S}_\lambda \mathbf{1} = \mathbf{1}$ or $\sum_{i=1}^n S_{ij} = 1$, if S_{ij} is the (i,j) element of $\boldsymbol{S}_\lambda$. The fit, $\hat{f}_\lambda^{-i}(x_i)$, is obtained when the weight for the ith observation is set to zero and the remaining weights are increased such that the sum of these weights equal one (Hastie and Tibshirani (1990), p.46).

Since $\sum_{\substack{j=1 \\ j\neq i}}^n S_{ij(\lambda)} = 1 - S_{ii(\lambda)}$, the correcting factor (increasing factor) should be defined as $\frac{1}{1-S_{ii(\lambda)}}$, thus $\sum_{\substack{j=1 \\ j\neq i}}^n \frac{S_{ij(\lambda)}}{1-S_{ii(\lambda)}} = 1$.

Since S_λ is a smoother we have

$$\hat{f}_\lambda^{-i}(x_i) = \sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{S_{ij(\lambda)}}{1 - S_{ii(\lambda)}} y_j$$

from which follows

$$\hat{f}_\lambda^{-i}(x_i)\,(1 - S_{ii(\lambda)}) = \sum_{\substack{j=1 \\ j\neq i}}^{n} S_{ij(\lambda)}\, y_j$$

$$\hat{f}_\lambda^{-i}(x_i) = \sum_{\substack{j=1 \\ j\neq i}}^{n} S_{ij(\lambda)}\, y_j + S_{ii(\lambda)} \hat{f}_\lambda^{-i}(x_i).$$

This holds for a cubic smoothing spline as well (Hastie and Tibshirani (1990), 47). To see that this holds for a cubic smoothing spline, suppose that $\hat{f}_\lambda^{-i}$ minimizes the penalised least squares equation

$$\sum_{\substack{j=1 \\ i\neq j}}^{n} \{y_j - g(x_j)\}^2 + \lambda \int \{g''(x)\}\, dx$$

for sample size $n-1$ and suppose we add the point $\{x_i, \hat{f}_\lambda^{-i}(x_i)\}$. Then $\hat{f}_\lambda^{-i}$ results in the same value and once again minimise

$$\sum_{\substack{j=1 \\ i\neq j}}^{n}\{y_j - g(x_j)\}^2 + \lambda\int\{g''(x)\}\,dx \text{ for the sample of size } n.$$

Combining $\widehat{f_\lambda}(x_i) = \sum_{\substack{j=1 \\ j\neq i}}^{n} S_{ij}(\lambda)\, y_j + S_{ii}(\lambda)y_i$ and $\hat{f}_\lambda^{-i}(x_i) = \sum_{\substack{j=1 \\ j\neq i}}^{n} S_{ij(\lambda)}\, y_j + S_{ii(\lambda)}\hat{f}_\lambda^{-i}(x_i)$ we have

$$\begin{aligned}\hat{f}_\lambda^{-i}(x_i) &= \sum_{\substack{j=1 \\ j\neq i}}^{n} S_{ij}(\lambda)\, y_j + S_{ii}(\lambda)\hat{f}_\lambda^{-i}(x_i)\\ &= \widehat{f_\lambda}(x_i) - S_{ii}(\lambda)y_i + S_{ii}(\lambda)\widehat{f_\lambda}^{-i}(x_i).\end{aligned}$$

$$\hat{f}_\lambda^{-i}(x_i) = \widehat{f_\lambda}(x_i) + S_{ii}(\lambda)\hat{f}_\lambda^{-i}(x_i) - S_{ii}(\lambda)y_i + y_i - y_i$$

Adding y_i on both sides, and we can write

$$\begin{aligned}y_i - \hat{f}_\lambda^{-i}(x_i) - S_{ii}(\lambda)y_i + S_{ii}(\lambda)\hat{f}_\lambda^{-i}(x_i) &= -\widehat{f_\lambda}(x_i) + y_i\\ (y_i - \hat{f}_\lambda^{-i}(x_i)) - S_{ii}(\lambda)(y_i - \hat{f}_\lambda^{-i}(x_i)) &= -\widehat{f_\lambda}(x_i) + y_i\\ (y_i - \hat{f}_\lambda^{-i}(x_i)) - S_{ii}(\lambda)(y_i - \hat{f}_\lambda^{-i}(x_i)) &= y_i - \widehat{f_\lambda}(x_i)\\ (1 - S_{ii}(\lambda))(y_i - \hat{f}_\lambda^{-i}(x_i)) &= y_i - \widehat{f_\lambda}(x_i)\\ (y_i - \hat{f}_\lambda^{-i}(x_i)) &= \frac{y_i - \widehat{f_\lambda}(x_i)}{(1 - S_{ii}(\lambda))}\end{aligned}$$

The fitting statistics for $\hat{f}_\lambda^{-i}(x_i)$ can be computed from the fit $\widehat{f_\lambda}(x_i)$ and $S_{ii}(\lambda)$. We do not have to remove the ith point and re-compute the fit $\hat{f}_\lambda^{-i}$ to obtain fitting statistics. The cross validation measure can therefore be computed from $\widehat{f_\lambda}$ by substituting this back into the cross validation sum of square to get

$$\begin{aligned}CV(\lambda) &= \frac{1}{n}\sum_{i=1}^{n}\{y_i - \hat{f}_\lambda^{-i}(x_i)\}^2\\ &= \frac{1}{n}\sum_{i=1}^{n}\left\{\frac{y_i - \widehat{f_\lambda}(x_i)}{(1 - S_{ii}(\lambda))}\right\}^2 \qquad (2.9.2.1.1)\end{aligned}$$

(Hastie and Tibshirani (1990), p.46-48).

The degrees of freedom is the sum of the eigen values of $\boldsymbol{S}_\lambda$, and gives an indication of the amount of fitting that $\boldsymbol{S}_\lambda$ does (Hastie and Tibshirani (1990), p.52).

To find the eigenvalue decomposition of $\boldsymbol{S}_\lambda$ we will write $\boldsymbol{S}_\lambda$ in *Reinch* form. The smooth

$$\boldsymbol{S}_\lambda = \boldsymbol{H}\left(\boldsymbol{H}'\boldsymbol{H} + \lambda\boldsymbol{\Omega}\right)^{-1}\boldsymbol{H}'$$

where $\boldsymbol{H} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T$ be the singular value decomposition of $\boldsymbol{H}$.

Since $\boldsymbol{H}$ is a square $N \times N$ matrix, U an orthogonal matrix hence invertible with $\boldsymbol{U}^{-1} = \boldsymbol{U}'$

$$\begin{aligned}
\boldsymbol{S}_\lambda &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}'\left(\boldsymbol{V}\boldsymbol{D}^2\boldsymbol{V}' + \lambda\boldsymbol{\Omega}\right)^{-1}\boldsymbol{V}\boldsymbol{D}\boldsymbol{U}' \\
&= \boldsymbol{U}(\boldsymbol{D}^{-1}\boldsymbol{V}'\boldsymbol{V}\boldsymbol{D}^2\boldsymbol{V}'\boldsymbol{V}\boldsymbol{D}^{-1} + \lambda\boldsymbol{D}^{-1}\boldsymbol{V}'\boldsymbol{\Omega}\boldsymbol{V}\boldsymbol{D}^{-1})^{-1}\,\boldsymbol{U}' \\
&= \boldsymbol{U}(\boldsymbol{I} + \lambda\boldsymbol{D}^{-1}\boldsymbol{V}'\Omega\boldsymbol{V}\boldsymbol{D}^{-1})^{-1}\boldsymbol{U}' \\
&= (\boldsymbol{U}'\boldsymbol{U} + \lambda\boldsymbol{U}'\boldsymbol{D}^{-1}\boldsymbol{V}'\Omega\mathbf{V}\boldsymbol{D}^{-1}\boldsymbol{U})^{-1} \\
&= (\boldsymbol{I} + \lambda\boldsymbol{U}'\boldsymbol{D}^{-1}\boldsymbol{V}'\Omega\mathbf{V}\boldsymbol{D}^{-1}\boldsymbol{U})^{-1} \\
&= (\boldsymbol{I} + \lambda\boldsymbol{K})^{-1} \qquad (2.9.2.1.2)
\end{aligned}$$

where $\boldsymbol{K}$ does not depend on λ ,

$\boldsymbol{K}$ is positive semi-definite and we can write

$\boldsymbol{K} = \boldsymbol{U}\boldsymbol{G}\boldsymbol{U}'$ where $= diag(g_1, \ldots, g_n)$, g_k is the corresponding eigenvalues of $\boldsymbol{K}$.

The value for $\boldsymbol{G}$ is $\boldsymbol{D}^{-1}\boldsymbol{V}'\Omega\boldsymbol{V}\boldsymbol{D}^{-1}$.

We have

$$\boldsymbol{S}_\lambda = (\boldsymbol{I} + \lambda\boldsymbol{U}\boldsymbol{G}\boldsymbol{U}')^{-1}$$

$$= \boldsymbol{U}(\boldsymbol{I} + \lambda\boldsymbol{G})^{-1}\boldsymbol{U}'$$

(Hastie and Tibshirani (1990), p.57).

The eigenvalues of $\boldsymbol{S}_\lambda$ is $p_k(\lambda)$ where

$$p_k(\lambda) = \frac{1}{1+\lambda g_k} \text{ where } p_k(\lambda) = 1, k = 1, \ldots, n$$

and

$$df_\lambda = \sum_{i=1}^{n} p_k(\lambda) = trace(\boldsymbol{S}_\lambda) \qquad (2.9.2.1.3)$$

(Hastie, Tibshirani and Friedman (2009), p.154).

2.10 Nonparametric linear regression by using B-splines

The B-spline basis function provides a numerically superior alternative basis to the natural cubic spline when calculating a smoothing spline. The main feature of B-splines is that any given basis function is nonzero over a span of at most five distinct knots. In practise this means that the evaluation rarely gets out of hand, and resulting regression matrix is banded. We will therefore use B-splines to calculate a smoothing spline. (Hastie and Tibshirani (1990), p.25).

This is also indicated by (Ohlsson and Johannsen (2010), p.108) stating “The B-splines are the simplest possible splines of each order and as orthogonal as possible”.

The natural cubic spline transformations is therfore replaced by B-spline transformations.

The estimator for the parameter β, is given by $\hat{\beta} = (\boldsymbol{B}'\boldsymbol{B} + \lambda\boldsymbol{\Omega})^{-1}\boldsymbol{B}'\boldsymbol{y}$, that is replacing $\boldsymbol{B}$ with $\boldsymbol{H}$ in equation 2.9.2 where $\boldsymbol{B}$ is determined as in equation 2.6.1 to equation 2.6.4.The value for $\boldsymbol{\Omega}$ can be calculated as follows (Ohlsson and Johannsen (2010)).

Let $\varepsilon_1, \dots, \varepsilon_m$ be the knots at each unique value of $x\varepsilon\mathbb{R}$, $m \leq N$.

Set $\varepsilon_i = \varepsilon_m$ for $i \geq m+1$ and $\varepsilon_i = \varepsilon_1$ for $i \leq 1$ in $\boldsymbol{\Omega}$. (These values will occur in $\boldsymbol{\Omega}$ and therefore the definition)

Let

$$a_{2k} = \frac{2}{\varepsilon_{k+1} - \varepsilon_{k-1}};\ k = 1, \dots, m,$$

$$a_{3k} = \frac{3}{\varepsilon_{k+1} - \varepsilon_{k-2}};\ k = 1, \dots, m+1.$$

The values in $\boldsymbol{\Omega}$ is equal to 0 except for the following cases:

$$\Omega_{1k,k} = \frac{\varepsilon_{k+1} - \varepsilon_{k-1}}{3};\ k = 1, \dots, m;$$

$$\Omega_{1k,k+1} = \Omega_{1k+1,k} = \frac{\varepsilon_{k+1} - \varepsilon_{k}}{6};\ k = 1, \dots, m-1,$$

$$\Omega_{2k,k} = a_{2,k-1}^2\Omega_{1k-1,k-1} - 2a_{2,k-1}a_{2,k}\Omega_{1k-1,k} + a_{2,k}^2\Omega_{1k,k}$$

$$k = 1, \dots, m+1,$$

$$\Omega_{2k,k+1} = \Omega_{2k+1,k} = a_{2,k-1}a_{2,k}\Omega_{1k-1,k} - a_{2,k}^2\Omega_{1k,k} + a_{2,k}a_{2,k+1}\Omega_{1k,k+1}$$

$$k = 1, \dots, m,$$

$$\Omega_{2k,k+2} = \Omega_{2k+2,k} = -a_{2,k}a_{2,k+1}\Omega_{1k,k+1};\ k = 1, \dots, m-1,$$

$$\Omega_{3k,k} = a_{3,k-1}^2\Omega_{2k-1,k-1} - 2a_{3,k-1}a_{3,k}\,\Omega_{2k-1,k} + a_{3,k}^2\Omega_{2k,k}$$

$$k = 1, \dots, m+2,$$

$$\Omega_{3k,k+1} = \Omega_{3k+1,k} = a_{3,k-1}\,a_{3,k}\Omega_{2k-1,k} - a_{3,k-1}a_{3,k+1}\,\Omega_{2k-1,k+1}$$

$$- a_{3,k}^2\Omega_{2k,k} + a_{3,k}a_{3,k+1}\Omega_{2k,k+1}$$

$$k = 1, \dots, m+1,$$

$$\Omega_{3k,k+2} = \Omega_{3k+2,k} = a_{3,k-1}\,a_{3,k+1}\,\Omega_{2k-1,k+1} - a_{3,k}\,a_{3,k+1}\Omega_{2k,k+1}$$

$$+a_{3,k}a_{3,k+2}\Omega_{2k,k+2} + a_{3,k}\,a_{3,k+1}\,\Omega_{2k,k+1}$$

$$k = 1, \dots, m,$$

$$\Omega_{3k,k+3} = \Omega_{3k+3,k} = -a_{3,k}a_{3,k+2}\Omega_{2k,k+2}\ \ \mathrm{k} = 1, \dots, \mathrm{m}-1.$$

Example 2.4 (continuation of example 2.3)

The same data as in example 2.3 are used to fit a nonparametric linear regression by using B-splines on equal distant data (Hastie, Tibshirani and Friedman (2009), p.159).

Note that for this example equal distant knots are used.

Let Y, taking on value 0 or 1, be a binary variable and let $\boldsymbol{X}$ be a set of explanatory variables. The odds for $Y = 1$ is then

$$\frac{P(Y = 1|\boldsymbol{X} = \boldsymbol{x})}{P(Y = 0|\boldsymbol{X} = \boldsymbol{x})}.$$

The value of λ was varied between 0.1 and 1 in intervals of 0.001. The CV_λ is plotted against df_λ in figure 13. The graph shows that the minimum cross validation value occurs where the degrees of freedom is 6 (when rounded). The resulting $f(X)$ from the regression is indicated by the purple line. The smoothing spline $df = 6$ graph in figure 13 shows the smooth purple line where degrees of freedom is 6. The smooth purple line is close to the actual simulated function (the red line).

The smoothing spline $df = 14$ graph in figure 13 shows the fit when the degrees of freedom is 14. One can see that the purple line starts to follow the data (blue dots) , as seen by the curvature influenced by local data point. The smoothing spline $df = 4$ graph on figure 13 shows the fit when degrees of freedom of 4 was used indicating under fitting.

The values for CV_λ and λ for $df_\lambda = 4$, $df_\lambda = 6$ and $df_\lambda = 14$ are given in table 3.

Table 3 Cross validation, degrees of freedom and lamda (Example 2.4)

λ	df_λ	CV_λ
1.491	4	0.01004
0.318	6	0.00928
0.0115	14	0.01039

Figure 13 shows the fits for the different selections in table 3.

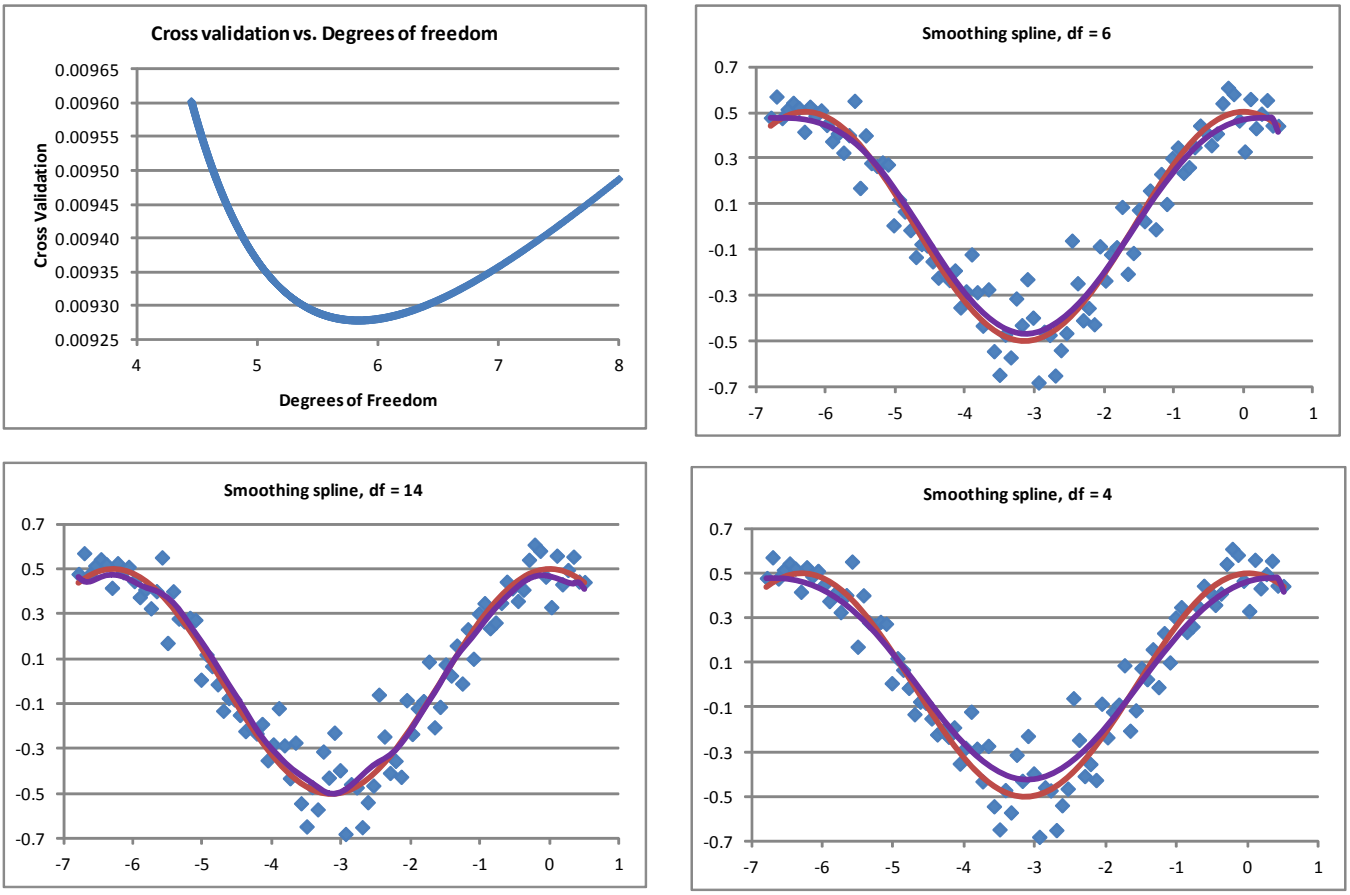


Figure 13 Smoothing spline

2.11 Nonparametric logistic regression by using B-splines

In section 2.10 smoothing splines were fitted to continuous data. In this section the outcome variable will be binary discrete variable. A logistic regression approach will be followed to fit a smoothing spline.

The nonparametric logistic regression model is given by

$$log\left(\frac{\Pr(Y=1|X=x)}{\Pr(Y=0|X=x)}\right)=f(x)$$

which implies

$$\Pr(Y=1|X=x)=\frac{e^{f(x)}}{1+e^{f(x)}}=p(x).$$

The penalised log likelihood, for a random sample of n observations, is

$$\begin{aligned}
l(f,\lambda) &= \sum_{i=1}^{n} y_i \log p(x_i) + (1-y_i)\log(1-\mathrm{p}(x_i)) - \tfrac{1}{2}\lambda\int\{f''(t)\}^2\,dt \\
&= \sum_{i=1}^{n} y_i \log\left(\tfrac{e^{f(x)}}{1+e^{f(x)}}\right) + (1-y_i)\log(1-(\tfrac{e^{f(x)}}{1+e^{f(x)}})) - \tfrac{1}{2}\lambda\int\{f''(t)\}^2\,dt \\
&= \sum_{i=1}^{n} y_i \log\left(\tfrac{e^{f(x)}}{1+e^{f(x)}}\right) + (1-y_i)\log((\tfrac{1}{1+e^{f(x)}})) - \tfrac{1}{2}\lambda\int\{f''(t)\}^2\,dt \\
&= \sum_{i=1}^{n} y_i \log\left(\tfrac{e^{f(x)}(1+e^{f(x)})}{1+e^{f(x)}}\right) + \log((\tfrac{1}{1+e^{f(x)}})) - \tfrac{1}{2}\lambda\int\{f''(t)\}^2\,dt \\
&= \sum_{i=1}^{n} y_i \log\left(e^{f(x)}\right) - \log((1+e^{f(x)})) - \tfrac{1}{2}\lambda\int\{f''(t)\}^2\,dt \\
&= \sum_{i=1}^{n} y_i f(x) - \log((1+e^{f(x)})) - \tfrac{1}{2}\lambda\int\{f''(t)\}^2\,dt. \qquad (2.11.1)
\end{aligned}$$

B-splines were used to represent $f(x)$ in equation 2.11.1. Let $Y_{N\times 1}$ be a vector of n binary responses and $\boldsymbol{B}$ a matrix containing the values of the B-spline transformations on $\boldsymbol{X}$.

As in section 2.10 the matrix $\boldsymbol{\Omega}$ is calculated.

Due to the way the B-splines are defined, the matrices $\mathbf{B}'\mathbf{W}\mathbf{B}$ and $\boldsymbol{\Omega}$ are banded, which simplifies the numerical solution of the linear equation system. $\mathbf{W}$ is described in section 2.3. We will solve the regression parameters by using

$$(\boldsymbol{B}'\boldsymbol{W}\boldsymbol{B}+\lambda\boldsymbol{\Omega})\,\beta=\boldsymbol{B}'\boldsymbol{W}\boldsymbol{y}. \qquad (2.11.2)$$

(Ohlsson and Johannsen (2010) , p.110)

The derivatives $l(f,\lambda)$ are

$$\frac{d(l(\beta))}{d\theta} = \boldsymbol{B}(\boldsymbol{y}-\boldsymbol{p}) - \lambda\boldsymbol{\Omega}\beta$$

$$\frac{d^2(l(\beta))}{d\beta d\beta^T} = -\boldsymbol{B}'\boldsymbol{W}\boldsymbol{B} - \lambda\boldsymbol{\Omega}.$$

From which the Newton Rhapson update rule it follows that

$$\begin{aligned}\beta_{new} &= \beta_{old} + (\boldsymbol{B}'\boldsymbol{W}\boldsymbol{B} + \lambda\boldsymbol{\Omega})^{-1}\boldsymbol{H}'(\boldsymbol{y}-\boldsymbol{p}) \\ &= (\boldsymbol{B}'\boldsymbol{W}\boldsymbol{B} + \lambda\boldsymbol{\Omega})^{-1}\boldsymbol{B}'\boldsymbol{W}\big(\boldsymbol{B}\beta_{old} + \boldsymbol{W}^{-1}(\boldsymbol{y}-\boldsymbol{p})\big) \\ &= (\boldsymbol{B}'\boldsymbol{W}\boldsymbol{B} + \lambda\boldsymbol{\Omega})^{-1}\boldsymbol{B}'\boldsymbol{W}\boldsymbol{z}\end{aligned}$$

where $\boldsymbol{z} = \boldsymbol{B}\beta_{old} + \boldsymbol{W}^{-1}(\mathbf{y}-\mathbf{p})$. (2.11.3)

The cross validation is given by

$$CV_\lambda = \frac{1}{N} w_i \sum_{i=1}^{N} \left(\frac{y_i - \hat{f}(x_i)}{1 - S_\lambda(i,i)}\right)^2. \qquad (2.11.4)$$

The smoother matrix is $\boldsymbol{S}_\lambda = \boldsymbol{B}(\boldsymbol{B}'\boldsymbol{W}\boldsymbol{B} + \lambda\boldsymbol{\Omega})^{-1}\boldsymbol{B}'\boldsymbol{W}$ with effective degree of freedom $df_\lambda = trace(\boldsymbol{S}_\lambda)$.

The following example from (Hastie and Tibshirani (1987), p.261) show how to fit a nonparametric binary logistic regression by means of a smoothing spline. B-splines will be used to fit a smoothing spline on nonequal distant data.

A smoothing spline was fitted to systolic blood pressure (SBP) on the South African heart disease dataset (This data set is described on page 38). The coronary heart disease variable is binary and will be used as the response.

The cross validation criteruim was calculated by varying λ from 0.0001 to 1 in steps of 0.0001.The region around the minimum of the cross validation is plotted against the degrees of freedom in figure 14. Table 4 contains the values for the cross validation, degrees of freedom and λ used in the smoothing spline calculation.

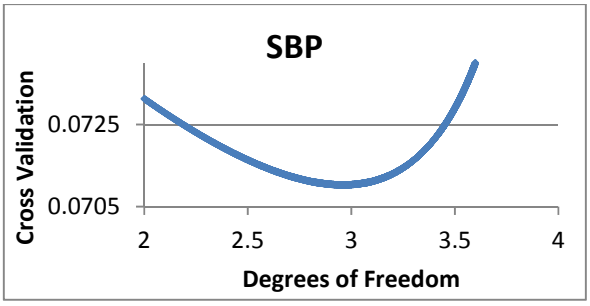


Figure 14 Systolic blood pressure, Degrees of Freedom

Table 4 Cross validation, degrees of freedom and lamda (SBP)

Variable	λ	CV	df
sbp	0.89361	0.071031	3

The confidence interval band for the smoothing spline fitted is plotted in figure 15 using

$$\mathrm{CI} = \mathrm{f(X)} \pm 2(\mathrm{Diagonal}\sqrt{(\mathrm{X}(\mathrm{X}'\mathrm{WX})^{-1}\mathrm{X}')}\ ).$$

Figure 15 shows the resulting smoothing spline and the confidence intervals. SBP is non linear as we can see in figure 15.

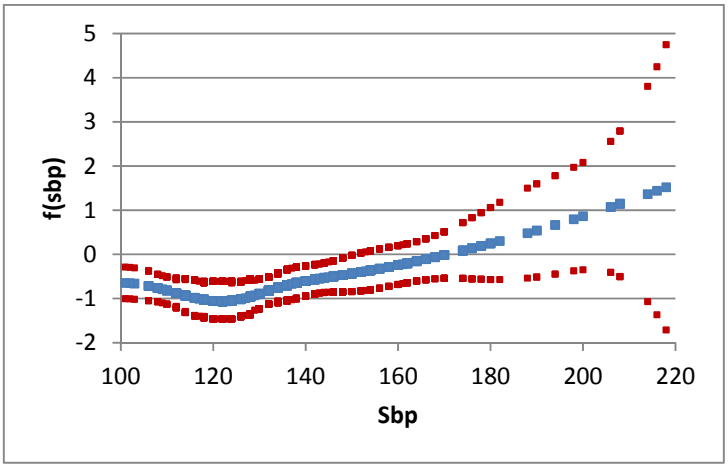


Figure 15 f(Systolic blood pressure) vs. SBP

2.12 Generalized additive models

Section 2.11 described a nonparametric logistic regression by using B-splines on a single rating variable, this idea will be extended to more than one rating variable in this section. The algorithm that we will use for parameter estimation is called the **local scoring algorithm**. The main idea behind this algorithm is to reduce the estimation problem to the one variable case discussed in section 2.11.

In section 2.7 we have discussed the form of the additive model that is given by

$$Y = \alpha + \sum_{j=1}^{p} f_j(X_j) + \varepsilon$$

(Hastie, Tibshirani and Friedman (2009), p.297).

The minimiser of the penalised sum of squares,

$$PRSS(\alpha, f_1, \dots f_p) = \sum_{i=1}^{N} \left(y_i - \alpha - \sum_{j=1}^{p} f_j(x_{ij}) \right)^2 + \sum_{j=1}^{p} \lambda_j \int f_j''(t_j)^2 dt_j, \qquad (2.12.1)$$

is an additive cubic spline model. Each of the functions f_j is a cubic spline in the component X_j. Knots will be selected at each of the unique values of $x_{ij},\ i = 1, \dots, N$.

To solve equation 2.12.1 we will assume that $\sum_{i=1}^{N} f_j(x_{ij}) = 0$. In this case

$$\bar{y} = average(y_i).$$

The value for $\bar{y}$ will not change in the iterative process that will follow. Apply a cubic smoothing spline

$$S_j \text{ to } \{y_i - \bar{y} - \textstyle\sum_{k \neq j} \widehat{f_k}(x_{ik})\}_1^N$$

to obtain a new estimate of $\hat{f}_j$. Do this for each predictor in turn, using the current estimates of the other functions $\widehat{f_k}$ when computing $y_i - \bar{y} - \sum_{k \neq j} \widehat{f_k}(x_{ik})$. Repeat this process until the estimates for $\hat{f}_j$ stabilizes. This process is known as the **backfitting algorithm**. (Hastie, Tibshirani and Friedman (2009), p.298).

At every iterative step $y_i - \bar{y} - \sum_{k \neq j} \widehat{f_k}(x_{ik})$ is used as the response variable and a smoothing spline is fitted to estimate a new value for $\hat{f}_j$ for every j. The methodology described in section 2.10, nonparametric linear regression by using B-spline, is repeatedly applied here.

This can be extended to the **generalized additive logistic model** which has the form

$$\log\left(\frac{Pr(Y = 1|X)}{Pr(Y = 0|X}\right) = \alpha + \sum_{j=1}^{p} f_j(X_j) + \varepsilon$$

(Hastie, Tibshirani and Friedman (2009), p.299).

The functions $f_1, \dots, f_p$ are estimated by using the **local scoring algorithm**:

- Compute starting values $\hat{\alpha} = \log\left(\bar{y}(1-\bar{y})\right)$
- Define $\widehat{\eta_i} = \hat{\alpha} + \sum_j \widehat{f_j}(x_{ij})$ and $\widehat{p_i} = 1/(1+\exp(-\widehat{\eta_i}))$
- Iterate the following:
    - Construct the working target variable $\widehat{z_i} = \widehat{\eta_i} + \frac{(y_i - \widehat{p_i})}{\widehat{p_i}(1-\widehat{p_i})}$
    - Construct weights $w_i = \widehat{p_i}(1-\widehat{p_i})$
    - Fit an additive model to the targets z_i with weights w_i, using the backfitting algorithm described above.
- Continue in this fashion until the estimates for $\hat{f}_j$ stabilizes.

The local scoring algorithm gives us a way to solve nonparametric logistic regression models with multiple variables (generalized additive logistic model). This is very useful when practical examples are considered. Example 3.1 shows that the Gini coefficient can be increased from 56.26% to 63.5% by using “additive logistic regression using local scoring algorithm” instead of “typical logistic regression”.

The Gini coefficient measures the discrimination between an event occurring or not occurring. There are two reasons for the increase in Gini coefficiant. Typical logistic regression divides variables into groups, treating values in the same interval identical. Nonparametric logistic regression will fit a curve to the data, which causes the curve to follow the data more closely than when intervals are used. A second reason is that non-linearities are picked up in variables and therefore more variables are entered into the model.

3. Applications

SAS IML programs were written to produce the examples given in this chapter from first principles, except where it was specified otherwise. (SAS Publishers, (2004)) and (SAS Publishers, (2006)). Appendix C – E contains some of the macros written in IML.

We will perform a logistic regression using natural cubic spline transformations on South African heart disease data. Then we will fit a typical logistic regression to South African heart disease data. An additive logistic regression model will then be fitted by using the local scoring algorithm (using PROC GAM procedure in SAS). A comparison will be drawn between the three models (Hastie, Tibshirani and Friedman (2009), p.146-148).

The Relative spinal bone mineral density data were also used in section 3.5 to fit a smoothing spline by using a linear regression and B-spline transformations.

3.1 Logistic regression using natural cubic spline transformations on South African heart disease data

We will commence by doing a logistic regression using natural cubic spline transformations on South African heart disease data.

A retrospective sample of males in a heart-disease high-risk region of the Western Cape, South Africa were considered. Many of the coronary heart disease (CHD) positive men have undergone blood pressure reduction treatment and other programs to reduce their risk factors after their CHD event. In some cases the measurements were made after these treatments. The variables and a description of the variables used is in table 5:

Table 5 South African heart disease data variable abbreviations

Variable	Description
SBP	systolic blood pressure
Tobacco	cumulative tobacco (kg)
Ldl	low density lipoprotein cholesterol
Famhist	family history of heart disease (Present, Absent)
Obesity	
Alcohol	current alcohol consumption
Age	age at onset
CHD	response, coronary heart disease

Knots were chosen at the minimum value, maximum value and at each quartile for each X_i. The knots used in the final model are indicated in table 6.

Table 6 South African heart disease data knots

x_i	Variable	ε_{1i} = Min	ε_{2i} = Quartile 1	ε_{3i} = Quartile 2	ε_{4i} = Quartile 3	ε_{5i} = Max
x_1	SBP	101	124	134	148	218
x_2	Tobacco	0	0.05	2	5.5	31.2
x_3	Ldl	0.98	3.28	4.34	5.8	15.33
x_4	Obesity	14.7	22.95	25.805	28.5	46.58
x_5	Age	15	31	45	55	64

A binary variable was created from x_6 and was called Famhist:

$$Binary(Famhist) = \begin{cases} 1\ if\ x_6 \text{ contains the value } present \\ 0\ if\ x_6 \text{contains the value } absent \end{cases}$$

The transformations done on X_i for a natural cubic spline with K knots are:

$$h_1(X_i) = 1\,, \quad h_2(X_i) = X_i,$$

$$h_{k+2}(X_i) = d_k(X_i) - d_{K-1}(X_i)$$

where

$$d_k(X) = \frac{(X_i-\varepsilon_{ki})_+^3-(X_i-\varepsilon_{Ki})_+^3}{\varepsilon_{Ki}-\varepsilon_{ki}} \text{ for all } k = 1,2,3 \text{ and } i = 1,\dots,5$$

Non-linearities are explored in the functions using natural cubic splines. A natural cubic spline was fitted to each of the X_i's by applying the transformations that follow to each X_i. The knots, $\varepsilon_{1i},\dots,\varepsilon_{5i}$ differ for each X_i as indicated in the above table.

$$h_2(X_i) = X_i$$

$$h_3(X_i) = \frac{(X_i-\varepsilon_{1i})_+^3-(X_i-\varepsilon_{5i})_+^3}{\varepsilon_{5i}-\varepsilon_{1i}} - \frac{(X_i-\varepsilon_{4i})_+^3-(X_i-\varepsilon_{5i})_+^3}{\varepsilon_{5i}-\varepsilon_{4i}}$$

$$h_4(X_i) = \frac{(X_i-\varepsilon_{2i})_+^3-(X_i-\varepsilon_{5i})_+^3}{\varepsilon_{5i}-\varepsilon_{2i}} - \frac{(X_i-\varepsilon_{4i})_+^3-(X_i-\varepsilon_{5i})_+^3}{\varepsilon_{5i}-\varepsilon_{4i}}$$

$$h_5(X_i) = \frac{(X_i - \varepsilon_{3i})_+^3 - (X_i - \varepsilon_{5i})_+^3}{\varepsilon_{5i} - \varepsilon_{3i}} - \frac{(X_i - \varepsilon_{4i})_+^3 - (X_i - \varepsilon_{5i})_+^3}{\varepsilon_{5i} - \varepsilon_{4i}}$$

The model applied to the South African heart disease data was

$$\begin{aligned}
&logit\big(Pr(\boldsymbol{CHD}|\boldsymbol{X})\big) \\
&= \beta_0 + f_1(SBP) + f_2(TOBACCO) + f_3(LDL) + f_4(OBESITY) + f_5(AGE) + f_6(FAMHIST) \\
&= \beta_0 + \beta_1 h_2(SPB) + \cdots + \beta_4 h_5(SBP) + \beta_5 h_2(TOBACCO) + \cdots + \beta_8 h_5(TOBACCO) + \cdots + \\
&\quad + \beta_9 h_2(LDL) + \cdots + \beta_{12} h_5(LDL) + \beta_{13} h_2(OBESITY) + \cdots + \beta_{16} h_5(OBESITY) + \cdots + \\
&\quad + \beta_{17} h_2(AGE) + \cdots + \beta_{20} h_5(AGE) + \beta_{21} Binary(FAMHIST) \\
&= \boldsymbol{H\beta}.
\end{aligned}$$

To estimate the beta values in the above logistic regression we will create a matrix $\boldsymbol{H}_{\boldsymbol{462\times 22}}$, that is a matrix with 462 rows and 22 columns.

Let

$$\boldsymbol{K}_{\boldsymbol{462\times 4}}^{x_i} = \begin{matrix} \vdots \\ h_2(x_i) \\ \vdots \end{matrix} \quad \dots \quad \begin{matrix} \vdots \\ h_5(x_i) \\ \vdots \end{matrix} \text{ for } i = 1, \dots, 5 \qquad (3.1.1)$$

where the rows consist of the transformations applied to the observations.

The values in the columns of the matrices $\boldsymbol{K}_{\boldsymbol{462\times 4}}^{x_i}$, $i = 1, \dots, 5$ were standardised by subtracting the mean of the column and dividing by the standard deviation of that column.

Now

$(\boldsymbol{H}_{\boldsymbol{462\times 21}}) =$

$$(\boldsymbol{1}_{\boldsymbol{462\times 1}} \quad \boldsymbol{K}_{\boldsymbol{462\times 4}}^{\boldsymbol{sbp}} \quad \boldsymbol{K}_{\boldsymbol{462\times 4}}^{\boldsymbol{tobacco}} \quad \boldsymbol{K}_{\boldsymbol{462\times 4}}^{\boldsymbol{ldl}} \quad \boldsymbol{K}_{\boldsymbol{462\times 4}}^{\boldsymbol{obesity}} \quad \boldsymbol{K}_{\boldsymbol{462\times 4}}^{\boldsymbol{age}} \quad \boldsymbol{Binary}(\boldsymbol{FAMHIST})_{\boldsymbol{462\times 1}})$$

Let $\boldsymbol{Y}_{\boldsymbol{462\times 1}}$ be the observed values of CHD. CHD is a binary response variable.

Let $\boldsymbol{p}_{\boldsymbol{462\times 1}} = \exp\big(\boldsymbol{H}\widehat{\boldsymbol{\beta}}\big)/(1 + \exp(\boldsymbol{H}\widehat{\boldsymbol{\beta}}))$ be the vector of fitted probabilities $\widehat{Pr}(\boldsymbol{Y}|\boldsymbol{X})$ for each observation and p_j be the elements of $\boldsymbol{p}$ and y_j the elements of $\boldsymbol{Y}_{\boldsymbol{462\times 1}}$ for $j = 1$ to 462. Let $\boldsymbol{W}_{\boldsymbol{462\times 462}}$ be a diagonal weight matrix with diagonal elements $\widehat{Pr}(\boldsymbol{Y}|\boldsymbol{X})\big(1 - \widehat{Pr}(\boldsymbol{Y}|\boldsymbol{X})\big)$.

The original variables were transformed and used in a standard logistic content.

Estimate the beta values using a logistic regression with matrix $\boldsymbol{H}$ by applying iterative least squares, using the following formulas:

$$\beta_{new} = \quad (\boldsymbol{H'WH})^{-1}\boldsymbol{H'W}\,\boldsymbol{z}$$

with

$$\boldsymbol{z} = (\boldsymbol{H}\beta_{old} + \boldsymbol{W^{-1}}(\boldsymbol{y} - \boldsymbol{p}).$$

The estimated beta values from the logistic regression is presented in table 7.

Table 7 South African heart disease data beta values

Variable	Natural Cubic Spline	$\boldsymbol{\beta}$	Beta value
Intercept		β_0	-1.504
SBP	**h2**	β_1	-3.204
SBP	**h3**	β_2	17.395
SBP	**h4**	β_3	-28.701
SBP	**h5**	β_4	14.476
Tobacco	**h2**	β_5	6.063
Tobacco	**h3**	β_6	-4361.516
Tobacco	**h4**	β_7	4421.662
Tobacco	**h5**	β_8	-65.107
Ldl	**h2**	β_9	1.660
Ldl	**h3**	β_{10}	-4.186
Ldl	**h4**	β_{11}	3.715
Ldl	**h5**	β_{12}	-0.597
Obesity	**h2**	β_{13}	-1.908
Obesity	**h3**	β_{14}	3.695
Obesity	**h4**	β_{15}	-3.387
Obesity	**h5**	β_{16}	1.353
Age	**h2**	β_{17}	4.012
Age	**h3**	β_{18}	-9.791
Age	**h4**	β_{19}	8.719
Age	**h5**	β_{20}	-1.896
Binary(Famhist)		β_{21}	1.078

The Akaike information criterion (AIC) gives an indication of the amount of information that is lost when a model is used to describe reality. The AIC decreases as the number of parameters increase. This will discourage the model to overfit. The ideal is the model with the minimum AIC. The technique to utilise is to remove variables one by one and record the AIC. If the variable causes an increase in AIC the variable should be removed from the model.

$$AIC = 2 \times number\ of\ rows\ in\ beta - 2 \times \text{Log Likelihood}$$

The AIC statistic was used to select the variables in the table 7. All the remaining terms in the model will cause the AIC to increase if any variable is removed from the model. One variable at a time was removed from the full mode. The information indicated in Table 8 shows how the AIC will increase if the variable in the model is excluded. A point to note is that if a variable is dropped then all its transformations are excluded.

Table 8 South African heart disease data variable selection

		DF	Deviance	AIC	Likelihood Ratio Test Statistic	Log Likelihood	BIC	Standard error
	Full model		458.090	502.090		-229.045	593.072	0.247
Variable excluded:	**SBP**	4	467.166	503.166	9.076	-233.583	577.606	0.253
	Tobacco	4	470.477	506.477	12.386	-235.238	580.917	0.253
	Ldl	4	472.395	508.395	14.304	-236.197	582.835	0.275
	Famhist	1	479.444	521.444	21.354	-239.722	608.291	0.288
	Obesity	4	466.237	502.237	8.148	-233.119	576.678	0.251
	Age	4	481.857	517.857	23.768	-240.929	592.297	0.271

Alcohol will be excluded if considered in table 8 since the AIC will increase if the variable is included in the model.

For completeness the following formulas were used in the above table:

$$\text{Log Likelihood} = \log\{\prod_{j=1}^{462} [p_j{}^{(y_j)} \ \ (1-p_j)^{(1-y_j)}]\}$$

$$AIC = 2 \times number\ of\ rows\ in\ beta - 2 \times \text{Log Likelihood}$$

$$BIC = number\ of\ rows\ in\ beta \times \log(N) - 2 \times \text{Log Likelihood}$$

$$Deviance = -2 \times \text{Log Likelihood}$$

The functions

$$\hat{f}_1(SBP) = \hat{\beta}_1 h_2(SPB) + \cdots + \hat{\beta}_4 h_5(SBP)$$

$$\hat{f}_2(TOBACCO) = \widehat{\beta_5} h_2(TOBACCO) + \cdots + \widehat{\beta_8} h_5(TOBACCO)$$

$$\hat{f}_3(LDL) = \hat{\beta}_9 h_2(LDL) + \cdots + \widehat{\beta_{12}} h_5(LDL)$$

$$\hat{f}_4(OBESITY) = \widehat{\beta_{13}} h_2(OBESITY) + \cdots + \widehat{\beta_{16}} h_5(OBESITY)$$

$$\widehat{f_5}(AGE) = \widehat{\beta_{17}}\, h_2(AGE) + \cdots + \widehat{\beta_{20}} h_5(AGE)$$

$$\widehat{f_6}(FAMHIST) = \widehat{\beta_{21}} Binary(FAMHIST)$$

were calculate and are displayed separately in figure 16 using a blue line against the corresponding X value.

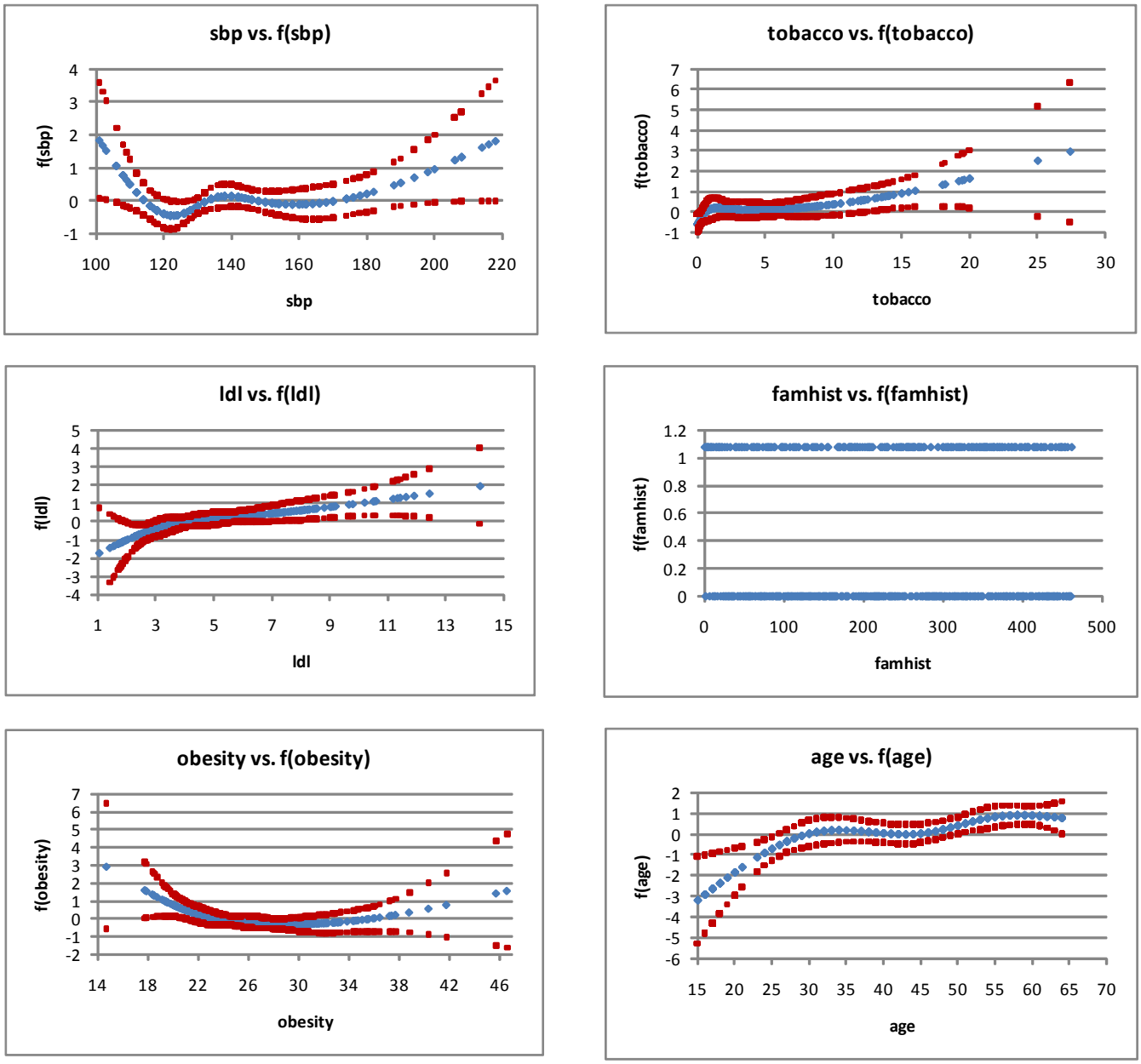


Figure 16 Logistic regression using natural cubic spline transformations

The variables obesity and SBP are nonlinear as we can see from figure 16.

The further you move from the mean the greater the variance. The red line in figure 16 moves further from the fitted function when moving away from the mean.

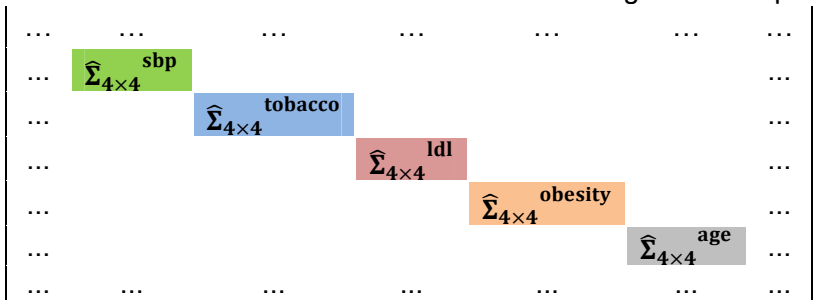
The values for $\widehat{\boldsymbol{\Sigma}} = (\boldsymbol{H}\grave{}\boldsymbol{W}\boldsymbol{H})^{-1}$ is displayed in table 9 where $(\widehat{\boldsymbol{\beta}})$ is equal to $\boldsymbol{\Sigma}$.

Table 9 Pointwise standard deviation

Intercept	0.04	0.05	-0.28	0.45	-0.22	-0.12	95.99	-97.43	1.54	-0.04	0.16	-0.18	0.05	0.04	-0.11	0.12	-0.05	-0.15	0.35	-0.28	0.05	-0.03
h2_sbp	0.05	1.89	-9.92	15.67	-7.51	0.50	-482.00	489.99	-8.44	-0.19	0.75	-0.83	0.23	0.10	-0.24	0.17	-0.02	-0.24	0.58	-0.46	0.07	-0.05
h3_sbp	-0.28	-9.92	56.62	-94.79	47.42	-2.12	2153.40	-2190.84	39.36	0.75	-2.58	2.45	-0.39	-0.29	0.50	0.07	-0.31	0.99	-2.37	1.97	-0.32	0.27
h4_sbp	0.45	15.67	-94.79	166.09	-86.02	2.86	-3127.61	3184.56	-59.57	-0.84	2.09	-0.95	-0.68	0.05	0.48	-1.75	1.22	-1.17	2.72	-2.37	0.43	-0.43
h5_sbp	-0.22	-7.51	47.42	-86.02	45.71	-1.19	1405.67	-1432.36	27.79	0.25	-0.12	-0.86	0.91	0.16	-0.78	1.55	-0.90	0.41	-0.90	0.83	-0.17	0.21
h2_tobacco	-0.12	0.50	-2.12	2.86	-1.19	16.76	-14488.17	14711.59	-238.34	0.19	-0.74	0.37	0.18	-0.31	1.24	-1.83	0.89	-0.81	2.20	-1.89	0.38	0.07
h3_tobacco	95.99	-482.00	2153.40	-3127.61	1405.67	-14488.17	13057566.00	-13264139.00	219520.25	-196.05	899.43	-882.19	173.54	310.67	-1137.75	1636.86	-785.28	519.03	-1498.88	1341.44	-288.91	-64.28
h4_tobacco	-97.43	489.99	-2190.84	3184.56	-1432.36	14711.59	-13264139.00	13474032.00	-223041.80	199.41	-916.14	901.93	-179.82	-315.81	1155.81	-1662.35	797.33	-525.35	1517.84	-1359.13	293.06	65.25
h5_tobacco	1.54	-8.44	39.36	-59.57	27.79	-238.34	219520.25	-223041.80	3735.20	-3.52	17.34	-20.00	6.07	5.42	-19.18	27.15	-12.86	7.03	-20.91	19.37	-4.48	-1.03
h2_ldl	-0.04	-0.19	0.75	-0.84	0.25	0.19	-196.05	199.41	-3.52	3.80	-19.31	28.77	-12.84	-0.03	-0.04	0.23	-0.17	0.00	-0.11	0.22	-0.11	0.02
h3_ldl	0.16	0.75	-2.58	2.09	-0.12	-0.74	899.43	-916.14	17.34	-19.31	104.12	-161.78	75.02	0.18	-0.11	-0.59	0.58	-0.02	0.39	-0.87	0.50	-0.10
h4_ldl	-0.18	-0.83	2.45	-0.95	-0.86	0.37	-882.19	901.93	-20.00	28.77	-161.78	260.26	-124.50	-0.27	0.20	0.67	-0.69	0.16	-0.81	1.41	-0.74	0.10
h5_ldl	0.05	0.23	-0.39	-0.68	0.91	0.18	173.54	-179.82	6.07	-12.84	75.02	-124.50	61.15	0.11	-0.08	-0.23	0.24	-0.12	0.47	-0.70	0.34	-0.02
h2_obesity	0.04	0.10	-0.29	0.05	0.16	-0.31	310.67	-315.81	5.42	-0.03	0.18	-0.27	0.11	2.15	-6.30	8.09	-3.73	-0.43	0.90	-0.60	0.05	-0.02
h3_obesity	-0.11	-0.24	0.50	0.48	-0.78	1.24	-1137.75	1155.81	-19.18	-0.04	-0.11	0.20	-0.08	-6.30	19.65	-26.90	13.02	0.97	-2.04	1.31	-0.08	0.06
h4_obesity	0.12	0.17	0.07	-1.75	1.55	-1.83	1636.86	-1662.35	27.15	0.23	-0.59	0.67	-0.23	8.09	-26.90	40.01	-20.64	-1.04	2.30	-1.51	0.07	-0.06
h5_obesity	-0.05	-0.02	-0.31	1.22	-0.90	0.89	-785.28	797.33	-12.86	-0.17	0.58	-0.69	0.24	-3.73	13.02	-20.64	11.15	0.44	-1.06	0.73	-0.04	0.03
h2_age	-0.15	-0.24	0.99	-1.17	0.41	-0.81	519.03	-525.35	7.03	0.00	-0.02	0.16	-0.12	-0.43	0.97	-1.04	0.44	2.52	-6.68	5.56	-1.03	0.02
h3_age	0.35	0.58	-2.37	2.72	-0.90	2.20	-1498.88	1517.84	-20.91	-0.11	0.39	-0.81	0.47	0.90	-2.04	2.30	-1.06	-6.68	19.43	-17.18	3.55	-0.06
h4_age	-0.28	-0.46	1.97	-2.37	0.83	-1.89	1341.44	-1359.13	19.37	0.22	-0.87	1.41	-0.70	-0.60	1.31	-1.51	0.73	5.56	-17.18	15.92	-3.62	0.05
h5_age	0.05	0.07	-0.32	0.43	-0.17	0.38	-288.91	293.06	-4.48	-0.11	0.50	-0.74	0.34	0.05	-0.08	0.07	-0.04	-1.03	3.55	-3.62	1.00	-0.01
Binary(famhist)	-0.03	-0.05	0.27	-0.43	0.21	0.07	-64.28	65.25	-1.03	0.02	-0.10	0.10	-0.02	-0.02	0.06	-0.06	0.03	0.02	-0.06	0.05	-0.01	0.06

Table 10 Sigma values

The sub matrices are selected from the above matrix using the corresponding colour:



The matrices $\boldsymbol{K}^{x_i}$, $i = 1, \ldots, 5$ were created in equation 3.1.1. The column vectors containing the pointwise variance for the functions $\widehat{f}_i$, $i = 1, .., 5$ are calculated using

$$covar\left(\widehat{f}_1(x_i)\right)_{462\times 1} = \boldsymbol{K}^{x_i}\boldsymbol{\Sigma}^{x_i}(\boldsymbol{K}^{x_i})^{\boldsymbol{T}}$$

The pointwise standard deviation is calculated using

$$\widehat{f}_i(x_i) \pm 2\sqrt{covar\left(\widehat{f}_i(x_i)\right)}, \; i = 1, \ldots, 5.$$

3.2 Typical binary logistic regression on South African heart disease data

We will now be fitting a typical logistic regression to South African heart disease data.

Natural cubic splines transformations were applied to each variable and the newly created variables were entered into a logistic regression. We would like to make a comparison to a typical logistic regression where the variables are not transformed and a global linear fit is used.

A matrix $\boldsymbol{X}$ was created using all the x_i's on the South African heart disease data set as colums.

$$\boldsymbol{X}_{\boldsymbol{462\times 8}} =$$

$$(\mathbf{1}_{462\times 1}\ \boldsymbol{tobacco}_{\ 462\times 1}\ \boldsymbol{ldl}_{\boldsymbol{462\times 1}}\ \boldsymbol{Binary}(\boldsymbol{FAMHIST})_{\boldsymbol{462\times 1}}\ \boldsymbol{Age}_{\boldsymbol{462\times 1}}).$$

Let

- $\boldsymbol{\beta} = (\beta_0 \ldots \beta_4)$
- $\boldsymbol{Y}_{\boldsymbol{462\times 1}}$ be the observed values of CHD. CHD is a Binary response variable
- $\boldsymbol{p}_{\boldsymbol{462\times 1}} = \exp\left(\boldsymbol{H}\widehat{\boldsymbol{\beta}}\right)/(1 + \exp(\boldsymbol{H}\widehat{\boldsymbol{\beta}}))$ is the vector of fitted probabilities $\widehat{Pr}(\boldsymbol{Y}|\boldsymbol{X})$ for each observation
- Let p_j be the elements of $\boldsymbol{p}$ and y_j the elements of $\boldsymbol{Y}_{\boldsymbol{462\times 1}}$ for $j = 1\ to\ 462$
- $\boldsymbol{W}_{\boldsymbol{462\times 462}}$ be a diagonal weight matrix with diagonal elements $\widehat{Pr}(\boldsymbol{Y}|\boldsymbol{X})\left(1 - \widehat{Pr}(\boldsymbol{Y}|\boldsymbol{X})\right)$

We will now solve the beta's in the logistic regression by applying the following formulas iteratively:

$$\beta(new) = (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}\grave{}\boldsymbol{W}\,\boldsymbol{z}$$

with

$$\boldsymbol{z} = (\boldsymbol{X}\beta(old) + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p}).$$

The form of the model is

$$logit\big(Pr(Chd|X)\big).$$

The standard error, Z-score and probability of the Z-score in table 11 are calculated using

$$\boldsymbol{standard\ error}_{8\times1} = \sqrt{diagonla((\boldsymbol{X}`\boldsymbol{W}\boldsymbol{X})^{-1})}$$
$$\boldsymbol{Z}_{8\times1} = \boldsymbol{\beta}_{8\times1}/\boldsymbol{standard\ error}_{8\times1}$$
$$\boldsymbol{PZ}_{8\times1} = 2 \times (1 - \text{probnorm}(\text{abs}(\mathbf{Z}))).$$

The final model is

$$logit(Pr(Chd \mid X))$$
$$= \beta_0 + \beta_1 tobacco + \beta_2 ldl + \beta_3\, Binary(FAMHIST) + \beta_4\, age$$
$$= \boldsymbol{H\beta}.$$

The fitted coefficients are shown in table 11.

Table 11 Typical logistic regression variable selection

Variable	Coefficient	Standard Error	Z-Score	Probability of Z-score
intercept	-4.204	0.498	-8.436	0.000
tobacco	0.081	0.026	3.163	0.002
ldl	0.168	0.054	3.093	0.002
$\boldsymbol{Binary(FAMHIST)}$	0.924	0.223	4.141	0.000
age	0.044	0.010	4.520	0.000

The probability of the Z-score to be nonsignificant is high for SBP, obesity and alcohol if entered into the above model. These variables were dropped from the model and the new model is shown.

3.3 Additive logistic regression on South African heart disease data by using the local scoring algorithm

We will fit an additive logistic model to the South African heart disease data by using the local scoring algorithm described in section 2.12.

Proc GAM in SAS was used to fit the additive logistic regression model. The AIC statistic was used to select the variables in the model. All the remaining terms in the model will cause the AIC to increase, if any one of the variable is removed from the model. One variable at a time was removed from the full model and the below table shows how the AIC will increase if the variable in the model is excluded.

Table 12 Local scoring algorithm variable selection

		DF	Deviance	AIC	Likelihood Ratio Test Statistic	Log Likelihood	BIC	N
	Full Model		449.24	493.235388		-224.62	566.479635	462
Variable excluded:	Sbp	4	457.48	493.48212	8.25	-228.74	553.409231	462
	Tobacco	4	468.08	504.083967	18.85	-234.04	564.011078	462
	Ldl	4	466.39	502.38521	17.15	-233.19	562.312321	462
	Famhist	1	468.42	510.417761	19.18	-234.21	580.332724	462
	Obesity	4	457.90	493.903797	8.67	-228.95	553.830908	462
	Age	4	473.95	509.95001	24.71	-236.98	569.877121	462

The Gini coefficient for the above model is 63.5%.

3.4 Comparing the three different logistic regression models

A comparison will be drawn between the logistic regression using natural cubic spline transformations, the normal logistic regression model and the additive model using the natural scoring algorithm.

The variables included in the logistic regression model using the different approaches are displayed in table 13.

Table 13 Variable comparison between three methods used

Variables included when using:		
Global linear fit	Natural cubic splines	Local scoring algorithm
binary_famhist	binary_famhist	binary_famhist
ldl	Ldl	Ldl
age	Age	Age
tobacco	tobacco	tobacco
	obesity	obesity
	SBP	SBP

Obesity and SBP were not included when using a Global linear fit, but included when natural cubic splines and local scoring algorithm were fitted. The graphs for SBP and obsesity when using natural cubic splines are given in figure 17 below

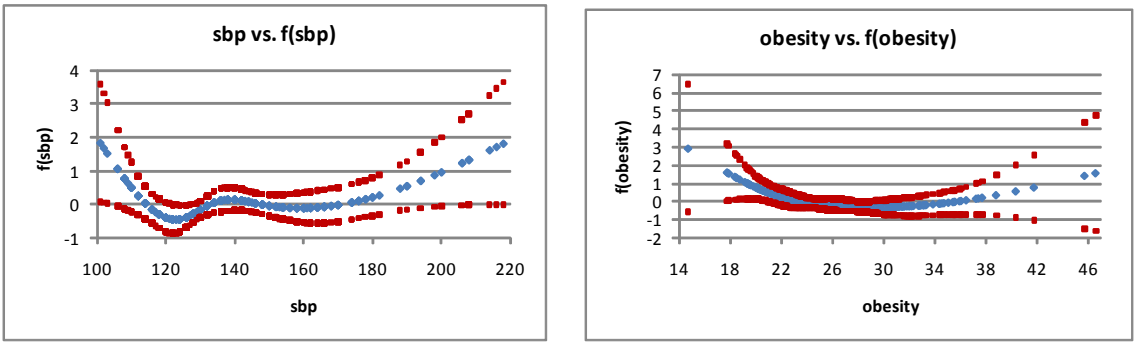


Figure 17 Non-linear variables

We can see that their contributions are nonlinear . This explains why they were not included when a linear fit was used.

The Gini's coefficients were calculated for each model and compared in the below table.

Table 14 Gini comparison

Model	Gini
Typical Logistic regression	56.26%
Logistic regression using Natural cubic splines	62.04%
Additive logistic regression using local scoring algorithm	63.5%

There is a substantial increase in the Gini coefficient when the nonparametric models are used. This means that we will discriminate better between $chd = 1$ and $chd = 0$ with the logistic regression using nonparametric models.

3.5 Fitting a smoothing spline by using a linear regression and B-spline transformations.

Relative spinal bone mineral density measurements on 261 North American adolescents were considered for this analysis. Each value is the difference in relative spinal bone mineral density measurement taken on two consecutive visits, divided by the average. The age is the average age over the two visits. (Hastie, Tibshirani and Friedman (2009), p.152).

The following variables are on the relative spinal bone mineral density measurements data set and a description will follow:

idnum:	identifies the child, and hence the repeat measurements
age:	average age of child when measurements were taken
gender:	male or female
spnbmd:	relative spinal bone mineral density measurement

The cross validation was calculated and plotted against the degrees of freedom for males and females separately.

The CV_λ for males is plotted against df_λ in figure 18. The graph shows that the minimum CV_λ is equal to 0.00173986, where df_λ is equal to 19.29. This leads to a λ of 0.0339 at this point and will be used for the smoothing spline calculation for males.

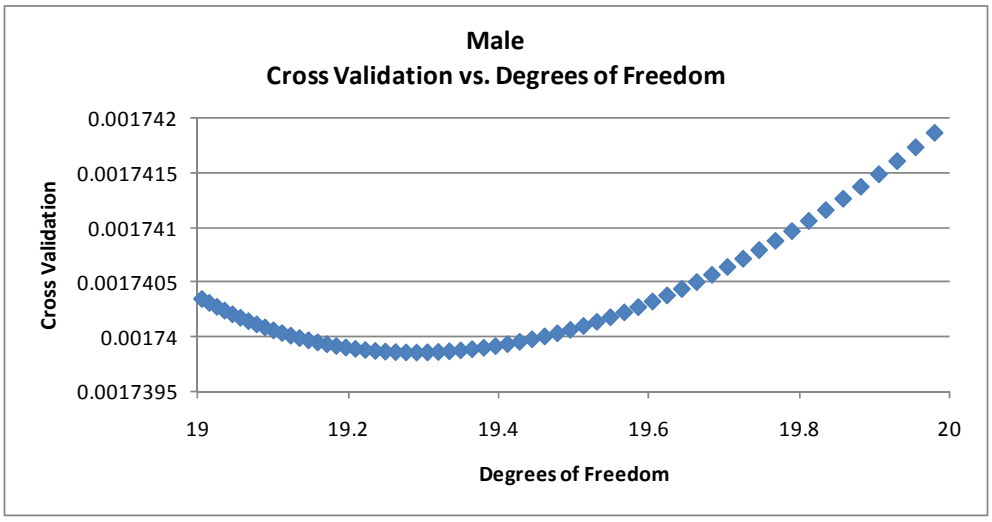


Figure 18 Male – Cross validation vs. Degrees of Freedom

The CV_λ for females is plotted against df_λ in the figure 19. The graph shows that the minimum CV_λ is equal to 0.001412, where df_λ is equal to 20.75. This leads to a λ of 0.0238 at this point and will be used for the smoothing spline calculation for females.

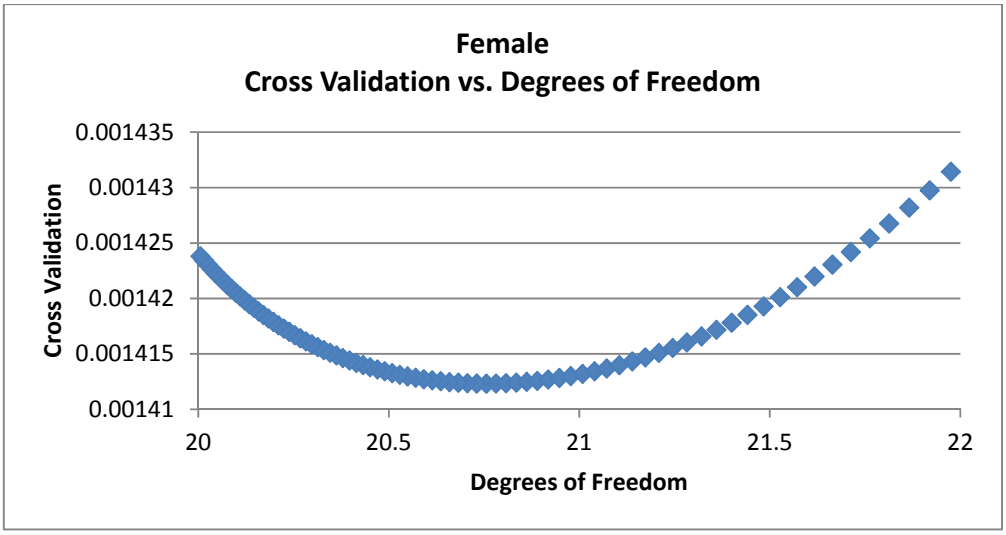


Figure 19 Female – Cross validation vs. degrees of Freedom

A nonparametric linear regression was applied to male and female on the relative spinal bone mineral density data to calculate smoothing splines. The smoothing splines and data points for males and female are indicated in figure 20.

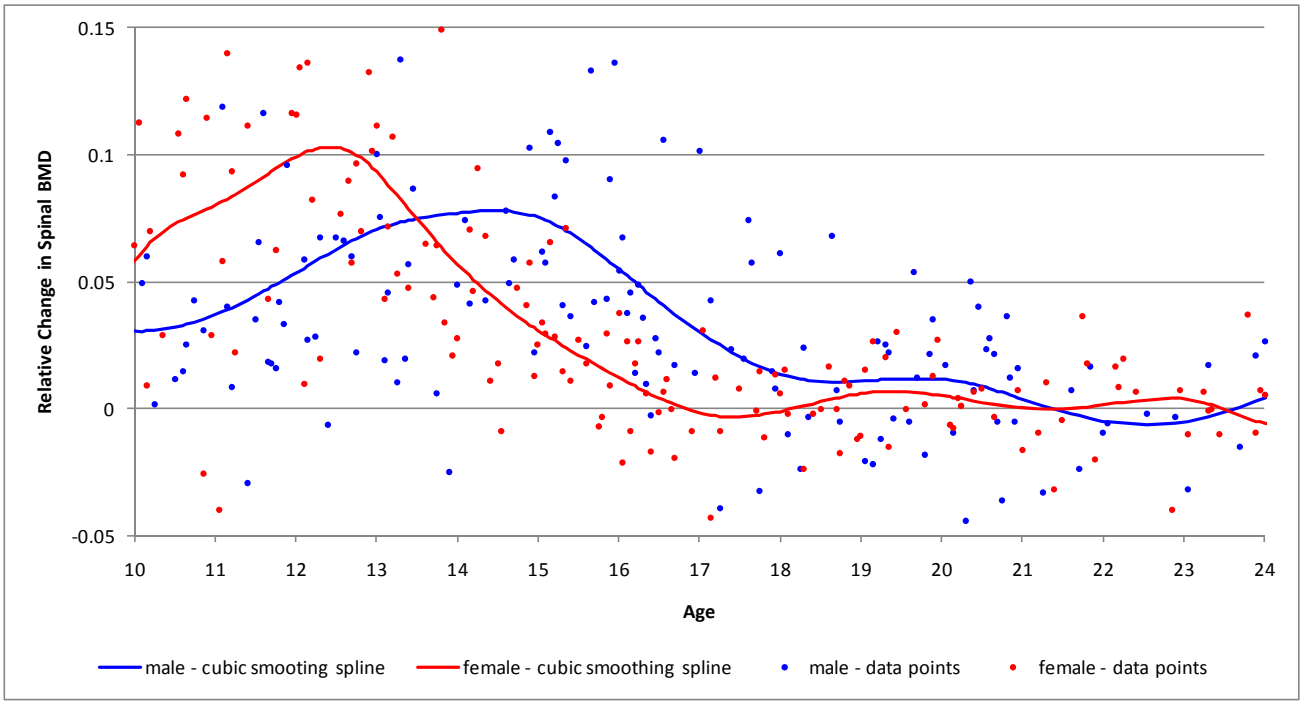


Figure 20 Relative change in Spinal BMD

The graphs shows that the growth spurt for females precedes that of males by about two years (Hastie, Tibshirani and Friedman (2009), p.152-p.153).

4. Proofs

This chapter is for completeness sake. Some of the proofs are shown here as referenced.

4.1 Natural cubic spline basis function expansion

The natural cubic spline was described in chapter 2 and used in chapter 3. In this section we proof certain result regarding natural cubic splines.

Polynomial functions behave erratic near the boundaries. The variance of the cubic spline is reduced near the boundaries knots by constraining the model to be linear beyond the boundary knots.

A cubic spline has a 4th order polynomial basis function with K interior knots and will be restricted by placing boundary constraints through linear restrictions on some of the parameters.

Lemma 4.1.1 Linear boundary conditions for the natural cubic spline

(Coetzee, J. (2009), p.63)

The cubic Spline's basis function expansion is given by

$$g(\varepsilon) = \sum_{i=0}^{3} \beta_i \varepsilon^i + \sum_{i=1}^{K} \theta_i (\varepsilon - \varepsilon_i)_+^3 \text{ for } \varepsilon \in [a, b].$$

The boundary restrictions imposed on the Cubic Spline basis function are two linear restrictions that ensure that the second and third derivates are zero at the boundaries and these restrictions are

$$\sum_{i=1}^{K} \theta_i = 0$$

and

$$\sum_{i=1}^{K} \theta_i \varepsilon_i = 0.$$

We will now show that the second and third derivatives are zero at the boundaries.

Ensure that the second and third derivatives are equal to zero in the interval $[\varepsilon_0, \varepsilon_1]$ by having the following boundary conditions

$$\beta_2 = \beta_3 = 0.$$

We then have

$$g(\varepsilon) = \beta_0 + \beta_1\varepsilon + \sum_{i=1}^{K} \theta_i(\varepsilon - \varepsilon_i)_+^3 \text{ for } \varepsilon \in [a, b].$$

We will now apply the boundary condition in the interval $[\varepsilon_K, \varepsilon_{K+1}]$.

We see that for $\varepsilon \geq \varepsilon_K$ it follows that $(\varepsilon - \varepsilon_i)_+^3 = (\varepsilon - \varepsilon_i)_i^3$.

Therefore

$$g(\varepsilon) = \beta_0 + \beta_1\varepsilon + \sum_{i=1}^{K} \theta_i(\varepsilon - \varepsilon_i)^3$$

$$g(\varepsilon) = \beta_0 + \beta_1\varepsilon + \sum_{i=1}^{K} \theta_i(\varepsilon - \varepsilon_i)(\varepsilon^2 - 2\varepsilon\varepsilon_i + \varepsilon_i^2)$$

$$g(\varepsilon) = \beta_0 + \beta_1\varepsilon + \sum_{i=1}^{K} \theta_i(\varepsilon^3 - 3\varepsilon^2\varepsilon_i + 3\varepsilon\varepsilon^2{}_i + \varepsilon_i^3).$$

The terms

$$\varepsilon^3 \sum_{i=1}^{K} \theta_i \text{ and } -3\varepsilon^2 \sum_{i=1}^{K} \theta_i\varepsilon_i \text{ must equal zero.}$$

This implies

$$\sum_{i=1}^{K} \theta_i = 0 \text{ and } \sum_{i=1}^{K} \theta_i\varepsilon_i = 0 \text{ when } \varepsilon \neq 0.$$

If $\varepsilon = 0$ we have

$$g(\varepsilon) = \beta_0 + \beta_1\varepsilon + \sum_{i=1}^{K} \theta_i\varepsilon_i^3. \qquad (4.1.1)$$

Equation 4.1.1 has a second and third derivative equal to zero in the upper bound.

Lemma 4.1.2 Expressions for θ_K and θ_{K-1}

(Coetzee, J. (2009), p.64)

Within the context of Lemma 4.1.1.

$$\theta_K = \frac{\theta_1(\varepsilon_{K-1} - \varepsilon_1)}{(\varepsilon_K - \varepsilon_{K-1})} + \frac{\theta_2(\varepsilon_{K-1} - \varepsilon_2)}{(\varepsilon_K - \varepsilon_{K-1})} + \cdots + \frac{\theta_{K-3}(\varepsilon_{K-1} - \varepsilon_{K-3})}{(\varepsilon_K - \varepsilon_{K-1})} + \frac{\theta_{K-2}(\varepsilon_{K-1} - \varepsilon_{K-2})}{(\varepsilon_K - \varepsilon_{K-1})}$$

and

$$\theta_{K-1} = \frac{-\theta_1(\varepsilon_K - \varepsilon_1)}{(\varepsilon_K - \varepsilon_{K-1})} - \frac{\theta_2(\varepsilon_K - \varepsilon_2)}{(\varepsilon_K - \varepsilon_{K-1})} - \cdots - \frac{\theta_{K-3}(\varepsilon_K - \varepsilon_{K-3})}{(\varepsilon_K - \varepsilon_{K-1})} - \frac{\theta_{K-2}(\varepsilon_K - \varepsilon_{K-2})}{(\varepsilon_K - \varepsilon_{K-1})}$$

which are two expressions in terms of all the other θ's.

The above Lemma will now be proofed.

From the previous lemma $\sum_{i=1}^{K} \theta_i = 0$ and writing it in another form gives

$$\theta_{K-1} = -\theta_1 - \theta_2 - \cdots - \theta_{K-3} - \theta_{K-2} - \theta_{K.}$$

Substitute this into $\sum_{i=1}^{K} \theta_i \varepsilon_i = 0$,

$$\theta_1(\varepsilon_1 - \varepsilon_{K-1}) + \theta_2(\varepsilon_2 - \varepsilon_{K-1}) + \cdots + \theta_{K-3}(\varepsilon_{K-3} - \varepsilon_{K-1}) + \theta_{K-2}(\varepsilon_{K-2} - \varepsilon_{K-1}) + \theta_K(\varepsilon_K - \varepsilon_{K-1}) = 0.$$

By simplifying and finding an expression for θ_K we obtain

$$\theta_K = \frac{\theta_1(\varepsilon_{K-1} - \varepsilon_1)}{(\varepsilon_K - \varepsilon_{K-1})} + \frac{\theta_2(\varepsilon_{K-1} - \varepsilon_2)}{(\varepsilon_K - \varepsilon_{K-1})} + \cdots + \frac{\theta_{K-3}(\varepsilon_{K-1} - \varepsilon_{K-3})}{(\varepsilon_K - \varepsilon_{K-1})} + \frac{\theta_{K-2}(\varepsilon_{K-1} - \varepsilon_{K-2})}{(\varepsilon_K - \varepsilon_{K-1})}.$$

In a similar way we can find and expression for θ_{K-1}.

From $\sum_{i=1}^{K} \theta_i = 0$ we obtain

$$\theta_K = -\theta_1 - \theta_2 - \cdots - \theta_{K-3} - \theta_{K-2} - \theta_{K-1.}$$

Substitute this into $\sum_{i=1}^{K} \theta_i \varepsilon_i = 0$ to get

$$\theta_1(\varepsilon_1 - \varepsilon_K) + \theta_2(\varepsilon_2 - \varepsilon_K) + \cdots + \theta_{K-3}(\varepsilon_{K-3} - \varepsilon_K) + \theta_{K-2}(\varepsilon_{K-2} - \varepsilon_K) + \theta_K(\varepsilon_{K-1} - \varepsilon_K) = 0.$$

By simplifying and finding an expression for θ_{K-1}we obtain

$$\theta_{K-1} = \frac{-\theta_1(\varepsilon_K - \varepsilon_1)}{(\varepsilon_K - \varepsilon_{K-1})} - \frac{\theta_2(\varepsilon_K - \varepsilon_2)}{(\varepsilon_K - \varepsilon_{K-1})} - \cdots - \frac{\theta_{K-3}(\varepsilon_K - \varepsilon_{K-3})}{(\varepsilon_K - \varepsilon_{K-1})} - \frac{\theta_{K-2}(\varepsilon_K - \varepsilon_{K-2})}{(\varepsilon_K - \varepsilon_{K-1})}.$$

Theorem 4.1.1 The basis function expansion for the natural cubic spline

(Coetzee, J. (2009), p.66)

The cubic spline has basis function expansion

$$g(\varepsilon) = \sum_{i=0}^{3} \beta_i \varepsilon^i + \sum_{i=1}^{K} \theta_i (\varepsilon - \varepsilon_i)_+^3 \text{ for } \varepsilon \in [a, b].$$

If the restrictions from Lemma 4.1.1 and Lemma 4.1.2 is imposed, this can be written as

$$g(\varepsilon) = \beta_0 H_1(\varepsilon) + \beta_2 H_2(\varepsilon) + \sum_{k=1}^{K-2} \theta_k H_{k+2}(\varepsilon)$$

with $k = 1, \ldots, K-2$ and where

$$H_1(\varepsilon) = 1, \quad H_2(\varepsilon) = \varepsilon \text{ and } H_{k+2}(\varepsilon) = d_k(\varepsilon) - d_{K-1}(\varepsilon)$$

$$d_K(\varepsilon) = \frac{(\varepsilon - \varepsilon_k)_+^3 - (\varepsilon - \varepsilon_K)_+^3}{\varepsilon_K - \varepsilon_k}.$$

Proof

The cubic spline has basis function expansion

$g(\varepsilon) = \sum_{i=0}^{3} \beta_i \varepsilon^i + \sum_{i=1}^{K} \theta_i (\varepsilon - \varepsilon_i)_+^3$ for $\varepsilon \in [a, b]$.

It is known that the natural cubic splines second and third derivative must be zero at the boundaries to ensure the function is linear for $\varepsilon \leq \varepsilon_1$ and $\varepsilon \geq \varepsilon_K$.

By imposing the linear restrictions and using $\beta_2 = \beta_3 = 0$ on the coefficient we obtain

$$g(\varepsilon) = \beta_0 + \beta_1 \varepsilon + \theta_1(\varepsilon - \varepsilon_1)_+^3 + \theta_2(\varepsilon - \varepsilon_2)_+^3 + \theta_3(\varepsilon - \varepsilon_3)_+^3 + \cdots + \theta_{K-1}(\varepsilon - \varepsilon_{K-1})_+^3 + \theta_K(\varepsilon - \varepsilon_K)_+^3.$$

This shows that

$$H_1(\varepsilon) = 1, \quad H_2(\varepsilon) = \varepsilon.$$

Now by substituting the expression for θ_{K-1} and θ_K obtained in Lemma 4.1.2 into $g(\varepsilon)$ we obtain

$$g(\varepsilon) = \beta_0 + \beta_1 \varepsilon + \theta_1(\varepsilon - \varepsilon_1)_+^3 + \theta_2(\varepsilon - \varepsilon_2)_+^3 + \theta_3(\varepsilon - \varepsilon_3)_+^3 + \cdots \theta_{K-2}(\varepsilon - \varepsilon_{K-2})_+^3$$

$$+ \left[\frac{-\theta_1(\varepsilon_K - \varepsilon_1)}{(\varepsilon_K - \varepsilon_{K-1})} - \frac{\theta_2(\varepsilon_K - \varepsilon_2)}{(\varepsilon_K - \varepsilon_{K-1})} - \cdots - \frac{\theta_{K-3}(\varepsilon_K - \varepsilon_{K-3})}{(\varepsilon_K - \varepsilon_{K-1})} - \frac{\theta_{K-2}(\varepsilon_K - \varepsilon_{K-2})}{(\varepsilon_K - \varepsilon_{K-1})}\right](\varepsilon - \varepsilon_{K-1})_+^3$$

$$+ [\frac{\theta_1(\varepsilon_{K-1} - \varepsilon_1)}{(\varepsilon_K - \varepsilon_{K-1})} + \frac{\theta_2(\varepsilon_{K-1} - \varepsilon_2)}{(\varepsilon_K - \varepsilon_{K-1})} + \cdots + \frac{\theta_{K-3}(\varepsilon_{K-1} - \varepsilon_{K-3})}{(\varepsilon_K - \varepsilon_{K-1})} + \frac{\theta_{K-2}(\varepsilon_{K-1} - \varepsilon_{K-2})}{(\varepsilon_K - \varepsilon_{K-1})}](\varepsilon - \varepsilon_K)_+^3$$

$$= \beta_0 + \beta_1 \varepsilon + \theta_1[(\varepsilon - \varepsilon_1)_+^3 - \frac{(\varepsilon_K - \varepsilon_1)(\varepsilon - \varepsilon_{K-1})_+^3}{(\varepsilon_K - \varepsilon_{K-1})} + \frac{(\varepsilon_{K-1} - \varepsilon_1)(\varepsilon - \varepsilon_K)_+^3}{(\varepsilon_K - \varepsilon_{K-1})}]$$

$$+ \theta_2[(\varepsilon - \varepsilon_2)_+^3 - \frac{(\varepsilon_K - \varepsilon_2)(\varepsilon - \varepsilon_{K-1})_+^3}{(\varepsilon_K - \varepsilon_{K-1})} + \frac{(\varepsilon_{K-1} - \varepsilon_2)(\varepsilon - \varepsilon_K)_+^3}{(\varepsilon_K - \varepsilon_{K-1})}]$$

$$+ \theta_{K-2}[(\varepsilon - \varepsilon_{K-2})_+^3 - \frac{(\varepsilon_K - \varepsilon_{K-2})(\varepsilon - \varepsilon_{K-1})_+^3}{(\varepsilon_K - \varepsilon_{K-1})} + \frac{(\varepsilon_{K-1} - \varepsilon_{K-2})(\varepsilon - \varepsilon_K)_+^3}{(\varepsilon_K - \varepsilon_{K-1})}]$$

which is equivalent to

$$\beta_0 + \beta_1\varepsilon + \theta_1^*[\frac{(\varepsilon-\varepsilon_1)_+^3}{(\varepsilon_K-\varepsilon_1)} - \frac{(\varepsilon-\varepsilon_{K-1})_+^3}{(\varepsilon_K-\varepsilon_{K-1})} + \frac{(\varepsilon_{K-1}-\varepsilon_1)(\varepsilon-\varepsilon_K)_+^3}{(\varepsilon_K-\varepsilon_1)(\varepsilon_K-\varepsilon_{K-1})}]$$

$$+\theta_2^*[\frac{(\varepsilon-\varepsilon_2)_+^3}{(\varepsilon_K-\varepsilon_2)} - \frac{(\varepsilon-\varepsilon_{K-1})_+^3}{(\varepsilon_K-\varepsilon_{K-1})} + \frac{(\varepsilon_{K-1}-\varepsilon_2)(\varepsilon-\varepsilon_K)_+^3}{(\varepsilon_K-\varepsilon_1)(\varepsilon_K-\varepsilon_{K-1})}]$$

$$+\theta_{K-2}^*[\frac{(\varepsilon-\varepsilon_{K-2})_+^3}{(\varepsilon_K-\varepsilon_{K-2})} - \frac{(\varepsilon-\varepsilon_{K-1})_+^3}{(\varepsilon_K-\varepsilon_{K-1})} + \frac{(\varepsilon_{K-1}-\varepsilon_{K-2})(\varepsilon-\varepsilon_K)_+^3}{(\varepsilon_K-\varepsilon_{K-2})(\varepsilon_K-\varepsilon_{K-1})}].$$

or

$$g(\varepsilon) = \beta_0 H_1(\varepsilon) + \beta_2 H_2(\varepsilon) + \theta_1^* H_3(\varepsilon) + \cdots + \theta_{K-2}^* H_K(\varepsilon)$$

with $k = 1, \ldots, K-2$ and where

$$H_1(\varepsilon) = 1, \quad H_2(\varepsilon) = \varepsilon \text{ and } H_{k+2}(\varepsilon) = d_k(\varepsilon) - d_{K-1}(\varepsilon)$$

$$d_K(\varepsilon) = \frac{(\varepsilon-\varepsilon_k)_+^3 - (\varepsilon-\varepsilon_K)_+^3}{\varepsilon_K-\varepsilon_k}.$$

Theorem 4.1.2 The basis function representation of the natural cubic spline

(Coetzee, J. (2009), p.69)

$$H_{k+2}(\varepsilon) = d_k(\varepsilon) - d_{K-1}(\varepsilon) \text{ and}$$

$$d_K(\varepsilon) = \frac{(\varepsilon-\varepsilon_k)_+^3 - (\varepsilon-\varepsilon_K)_+^3}{\varepsilon_K-\varepsilon_k}$$

ensures that the second and third order derivatives of the functions is equal to zero for $\varepsilon \geq \varepsilon_K$.

Proof

Let

$$I(\varepsilon > \varepsilon_k) = \begin{cases} 1 & \text{if } \varepsilon > \varepsilon_k \\ 0 & otherwise \end{cases}.$$

$$H_{k+2}(\varepsilon) = \frac{(\varepsilon-\varepsilon_k)_+^3}{\varepsilon_K-\varepsilon_k} - \frac{(\varepsilon-\varepsilon_{k-1})_+^3}{\varepsilon_K-\varepsilon_{K-1}} + \frac{(\varepsilon_{K-1}-\varepsilon_k)(\varepsilon-\varepsilon_K)_+^3}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})}$$

$$= \frac{(\varepsilon-\varepsilon_k)^3}{\varepsilon_K-\varepsilon_k} I(\varepsilon > \varepsilon_k) - \frac{(\varepsilon-\varepsilon_{k-1})^3}{\varepsilon_K-\varepsilon_{K-1}} I(\varepsilon > \varepsilon_{k-1}) + \frac{(\varepsilon_{K-1}-\varepsilon_k)(\varepsilon-\varepsilon_K)^3}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})} I(\varepsilon > \varepsilon_k)$$

$$H`_{k+2}(\varepsilon) = \frac{3(\varepsilon-\varepsilon_k)^2}{\varepsilon_K-\varepsilon_k} I(\varepsilon>\varepsilon_k) - \frac{3(\varepsilon-\varepsilon_{k-1})^2}{\varepsilon_K-\varepsilon_{K-1}} I(\varepsilon>\varepsilon_{k-1})$$

$$+\frac{3(\varepsilon_{K-1}-\varepsilon_k)(\varepsilon-\varepsilon_K)^2}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})} I(\varepsilon>\varepsilon_k)$$

$$H``_{k+2}(\varepsilon) = \frac{6(\varepsilon-\varepsilon_k)}{\varepsilon_K-\varepsilon_k} I(\varepsilon>\varepsilon_k) - \frac{6(\varepsilon-\varepsilon_{k-1})}{\varepsilon_K-\varepsilon_{K-1}} I(\varepsilon>\varepsilon_{k-1})$$

$$+\frac{6(\varepsilon_{K-1}-\varepsilon_k)(\varepsilon-\varepsilon_K)}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})} I(\varepsilon>\varepsilon_k).$$

If $\varepsilon \geq \varepsilon_k$ then $t = \varepsilon_k + m$ where $m \geq 0$

$$H``_{k+2}(\varepsilon) = \frac{6(\varepsilon_K+m-\varepsilon_k)}{\varepsilon_K-\varepsilon_k} - \frac{6(\varepsilon_k+m-\varepsilon_{k-1})}{\varepsilon_K-\varepsilon_{K-1}} + \frac{6(\varepsilon_{K-1}-\varepsilon_k)(\varepsilon_k+m-\varepsilon_K)}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})}$$

$$= \frac{6[(\varepsilon_K-\varepsilon_k)+m](\varepsilon_K-\varepsilon_{K-1}) - 6[(\varepsilon_K-\varepsilon_{K-1})+m](\varepsilon_K-\varepsilon_k)}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})} + \frac{6(\varepsilon_{K-1}-\varepsilon_k)m}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})}$$

$$= \frac{6m(\varepsilon_K-\varepsilon_{K-1}) - 6m(\varepsilon_K-\varepsilon_k)}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})} + \frac{6m(\varepsilon_{K-1}-\varepsilon_k)}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})}$$

$$= 0.$$

Therefore any of the basis functions of the form $H_{k+2}(\varepsilon) = d_k(\varepsilon) - d_{K-1}(\varepsilon)$ will have a second order derivative equal to zero for $\varepsilon \geq \varepsilon_k$.

Similarly

$$H```_{k+2}(\varepsilon) = \frac{6}{(\varepsilon_K-\varepsilon_k)} - \frac{6}{(\varepsilon_K-\varepsilon_{K-1})} + \frac{6(\varepsilon_{K-1}-\varepsilon_k)}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})}$$

$$= \frac{-6(\varepsilon_{K-1}-\varepsilon_k)}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})} + \frac{6(\varepsilon_{K-1}-\varepsilon_k)}{(\varepsilon_K-\varepsilon_k)(\varepsilon_K-\varepsilon_{K-1})}$$

$$= 0.$$

The third order derivative for these basis functions is zero for any ε.

4.2 Smoothing spline by using B-splines.

We will prove that $(\boldsymbol{B}\grave{}\boldsymbol{W}\boldsymbol{B} + \lambda\Omega)\boldsymbol{\beta} = \boldsymbol{B}\grave{}\boldsymbol{W}\boldsymbol{y}$ for a smoothing spline using a B-spline. Then we will prove how to obtain the values inside the matrix omega for the smoothing spline when a B-spline is used.

The proof of certain lemmas and theorems will be followed by the proof of the main result.

Lemma 4.2 1
(Ohlsson and Johannsen (2010) , p.139).

Let

- $u_1 < \cdots < u_m$ be a set of points
- $y_1, \ldots, y_m$ be real numbers
- $\mathfrak{X}$ be a set of twice continuously differentiable functions f such that $f(u_k) = y_k, k = 1, \ldots, m$

Let $s \in \mathfrak{X}$ have the property

$$\int_a^b s``(x)\, h``(x)dx = 0\,, \quad a < u_1, \qquad b > u_m \qquad (4.2.1)$$

h is a twice continuous differentiable function,

we have $h(u_k) = 0, k = 1, \ldots, m.$

Then for any $f \in \mathfrak{X}$

$$\int_{\boldsymbol{a}}^{\boldsymbol{b}} (s``(x))^{\boldsymbol{2}} \leq \int_{\boldsymbol{a}}^{\boldsymbol{b}} (f``(x))^{\boldsymbol{2}}\,, \quad a < u_1, \qquad b > u_m$$

Proof

Let $h(x) = f(x) - s(x).$

We have $h(u_k) = 0, k = 1, \ldots, m$ and so

$$\begin{aligned}\int_{\boldsymbol{a}}^{\boldsymbol{b}} (f``(x))^{\boldsymbol{2}}\, dx &= \int_a^b \left((s``(x))^{\boldsymbol{2}} + 2s``(x)h``(x) + (h``(x))^{\boldsymbol{2}}\right)dx \\ &= \int_a^b (s``(x))^{\boldsymbol{2}}dx + \int_a^b (h``(x))^{\boldsymbol{2}}dx \geq \int_a^b (s``(x))^{\boldsymbol{2}}dx\,.\end{aligned}$$

We will now look at property 4.2.1.

Let h be twice continuously differentiable and it does not satisfy $h(u_k) = 0, k = 1 \dots m.$

Let the new notation be $u_0 = a$ and $u_{m+1} = b$. Then

$\int_a^b s``(x)h``(x)\,dx = \lim_{\epsilon\to 0}\sum_{k=1}^{m}\int_{u_k+\epsilon}^{u_{k+1}-\epsilon} s``(x)h``(x)dx$.Let s be four time continuous differentiable for $x \neq u_1, \dots, u_m$.

Use integration by parts. Do this twice to get

$$\int_{u_k+\epsilon}^{u_{k+1}-\epsilon} s``(x)h``(x)dx = s``(u_{k+1}-\epsilon)h`(u_{k+1}-\epsilon) - s``(u_k+\epsilon)h`(u_k+\epsilon)$$

$$-s```(u_{k+1}-\epsilon)h(u_{k+1}-\epsilon) - s```(u_k+\epsilon)h(u_k+\epsilon)$$

$$+\int_{u_k+\epsilon}^{u_{k+1}-\epsilon}\frac{d^4}{dx^4}s(x)h(x)dx.$$

If $\frac{d^4s(x)}{dx^4}$=0 for $x \neq u_1, \dots, u_m$ then the last integral in the above equation will disappear. We will assume this, it will apply that $s```(x)$ is constant between the knots, and let $\epsilon\to 0$. We can now see that

$$\int_a^b s``(x)h``(x)dx = \sum_{k=0}^{m}[s``(u_{k+1})h`(u_{k+1}) - s``(u_k)h`(u_k)]$$

$$+\sum_{k=0}^{m}[s```(u_{k+1}-)h(u_{k+1}) - s```(u_k+)h(u_k)]$$

$$= s``(u_{m+1})h`(u_{m+1}) - s``(u_0)h`(u_0)$$

$$+\sum_{k=1}^{m}(s```(u_k+) - s```(u_k-))h(u_k).$$

If $s``(x)$ = 0 for $x \le u_1$ and $x \ge u_m$ then equation 4.2.1 will hold when $h(u_k) = 0, k = 1, \dots, m.$

Summarize the above in the following lemma:

Lemma 4.2.2

(Ohlsson and Johannsen (2010), p.140)

Let s be four times continuous differentiable, except at the points $u_1, \dots, u_m$, where it is twice continuously differentiable. Let s satisfies the following conditions

(i) $\frac{d^4}{dx^4}s(x) = 0\ for\ x \neq u_1, \dots, u_m$

and

$$(ii) \qquad s``(x) = 0 \; for \; x \leq \; u_1 \; and \; x \geq u_m.$$

Then for any twice continuously differentiable function h and for any $a < \; u_1, b > u_m$

$$\int_a^b s``(x)h``(x)\,dx = \sum_{k=1}^{m} d_k h(u_k) \qquad (4.2.2)$$

where $d_k = \; s```(u_k +) - s```(u_k -)$.

From (i) we can see that s must be a cubic polynomial on each of the intervals (u_k , u_{k+1}) and if s is twice continuously differentiable, we can see that s must be a cubic spline. Condition (ii) says it is natural cubic spline. Using condition (i) and (ii) we can construct an expression for a natural cubic spline.

Use the fact that s is twice differentiable and that it satisfies (i) and (ii) in lemma 4.2.2. From (i) it follows that $s```(x)$ is a piecewise constant between any two points u_k and u_{k+1}. Using (ii), $s```(x)$ must be identically zero for $x < \; u_1 \; and \; x > u_m$. Now, if $x \neq u_1, \dots, u_m$

$$s```(x) = \sum_{k=1}^{m} d_k I_{\{u_k < x\}.} \qquad (4.2.3)$$

with d_k as in Lemma 4.2.2.

We can see that

$$\sum_{k=1}^{m} d_k = 0, \quad \sum_{k_1}^{m} d_k u_k = 0\,. \qquad (4.2.4)$$

The first equality follows from the fact that

$$\sum_{k} d_k = s```(b) - s``(a) = 0$$

where

$a < \; u_1, b > \; u_m$.

The second holds from $\sum_k d_k u_k = -\int_a^b s^{```}(x)dx = s^{``}(a) - s^{``}(b) = 0$.

We now derive and expression from $s^{``}(x)$, starting from 4.2.3. The primitive function of $t \to d_k I_{\{u_k<x\}}$ is, if $x \neq u_k$, $x \to b_k I_{\{u_k<x\}} + (d_k x + b_k) I_{\{x>u_k\}}$. For some constant b_k and c_k. Thus, if $x \neq u_1, \dots, u_m$

$$s^{``}(x) = \sum_{k=1}^{m} b_k I_{\{u_k<x\}} + (d_k x + c_k) I_{\{x>u_k\}} + a),$$

for some constant a. By letting $x \to u_k$ from above and below, we have, due to the continuity of $s^{``}(x)$, that $b_k = d_k u_k + c_k$. From this we get

$$s^{``}(x) = \sum_{k=1}^{m} d_k u_k I_{\{u_k<x\}} + (d_k x) I_{\{x>u_k\}} + b)$$

where $b = \sum_{k=1}^{m} c_k + a$. Since for $x > u_m$, $s^{``}(x) = \sum_{k=1}^{m} d_k x + b$, we conclude from 4.2.4 we have

$$\begin{aligned} s^{``}(x) &= \textstyle\sum_{k=1}^{m} d_k u_k I_{\{u_k<x\}} + \sum_{k=1}^{m} d_k x I_{\{x>u_k\}} \\ &= \textstyle\sum_{k=1}^{m} d_k u_k I_{\{u_k<x\}} + x(\sum_{k=1}^{m} d_k - \sum_{k=1}^{m} d_k I_{\{x<u_k\}}) \\ &= \textstyle\sum_{k=1}^{m} d_k (u_k - x) I_{\{x<u_k\}} . \end{aligned}$$

On the other hand

$$\begin{aligned} s^{``}(x) &= \sum_{k=1}^{m} d_k u_k I_{\{u_k<x\}} + \sum_{k=1}^{m} d_k x I_{\{x>u_k\}} \\ &= \sum_{k=1}^{m} d_k u_k - \sum_{k=1}^{m} d_k u_k I_{\{x>u_k\}} + \sum_{k=1}^{m} d_k x \ I_{\{x>u_k\}} \\ &= \textstyle\sum_{k=1}^{m} d_k (x - u_k) \ I_{\{x>u_k\}} . \end{aligned}$$

We thus have

$$2s^{``}(x) = \textstyle\sum_{k=1}^{m} d_k (u_k - x) I_{\{x<u_k\}} + \sum_{k=1}^{m} d_k (x - u_k) \ I_{\{x>u_k\}}$$

$$= \sum_{k=1}^{m} d_k |x - u_k|$$

and so

$$s``(x) = \frac{1}{2} \sum_{k=1}^{m} d_k |x - u_k| \ . \qquad (4.2.5)$$

From 4.2.5 the following lemma is easily shown, using 4.2.4 again.

Lemma 4.2.3

(Ohlsson and Johannsen (2010), p.142)

Assume that the twice continuously differential functions satisfies (i) and (ii) in Lemma 4.2.2. Then

$$s(x) = \frac{1}{12} \sum_{k=1}^{m} d_k |x - u_k|^3 + a_0 + a_1 x \qquad (4.2.6)$$

for some constant a_0 and a_1.

We have a natural cubic spline since it is twice continuously differentiable and between the knots it is a cubic polynomial. The condition 4.2.4 implies that it is natural. The next lemma, that uses this representation, a simple expression for the integrated squared second derivative can be easily derived.

Lemma 4.2.4

(Ohlsson and Johannsen (2010), p.142)

With $s(x)$ as in 4.2. 6

$$\int_a^b (s``(x))^2 \, dx = \frac{1}{12} \sum_{j=1}^{m} \sum_{k=1}^{m} d_j d_k |u_j - u_k|^3.$$

Proof

Taking $h = s$ in 4.2.2 we get

$$\int_a^b (s``(x))^2 \, dx = \sum_{j=1}^{m} d_j \, s(u_j)$$

$$= \frac{1}{12} \sum_{j=1}^{m} d_j \sum_{k=1}^{m} d_k |u_j - u_k|^3 + a_0 \sum_{j=1}^{m} d_j + a_1 \sum_{j=1}^{m} d_j u_j. \quad (4.2.7)$$

Thus 4.2.7 follows from 4.2.4.

We are now ready to prove the following basic result.

Theorem 4.2.1

(Ohlsson and Johannsen (2010), p.143)

For any $u_1, \dots, u_m$ with $u_1 < \cdots < u_m$ and any real numbers $y_1, \dots, y_m$ there exists a unique natural cubic spline $s(x)$, such that $s(u_j) = y_j, j = 1, \dots, m$.

Proof

Put $e_{jk} = \frac{|u_j - u_k|^3}{12}$. By Lemma 4.2.3 we may represent natural cubic splines as in 4.2.6. The condition $s(u_j) = y_j, j = 1, \dots, m$ then becomes

$$e_{11}d_1 + e_{12}d_2 + \cdots + e_{1m}d_m + a_0 + a_1u_1 = y_1$$

$$e_{21}d_1 + e_{22}d_2 + \cdots + e_{2m}d_m + a_0 + a_1u_2 = y_2$$

...

$$e_{m1}d_1 + e_{m2}d_2 + \cdots + e_{mm}d_m + a_0 + a_1u_m = y_m.$$

These are m equations with $m + 2$ unknowns, but if we add those in 4.2.4 , we get $m + 2$ equations. We introduce the matrix

$$\boldsymbol{E} = \begin{pmatrix} e_{11} & \cdots & e_{1n} \\ \vdots & \ddots & \vdots \\ e_{n1} & \cdots & e_{nn} \end{pmatrix} \quad (4.2.8)$$

and the vectors

$$\boldsymbol{d} = \begin{pmatrix} d_1 \\ \vdots \\ d_m \end{pmatrix}, \quad \mathbf{1} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}, \quad \boldsymbol{u} = \begin{pmatrix} u_1 \\ \vdots \\ u_m \end{pmatrix}, \quad \boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}.$$

We may then write 4.2.8 plus 4.2.4 as

$$\begin{pmatrix} \boldsymbol{E} & \mathbf{1} & \boldsymbol{u} \\ \mathbf{1}` & 0 & 0 \\ \boldsymbol{u}` & 0 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{d} \\ a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{y} \\ 0 \\ 0 \end{pmatrix}. \quad (4.2.9)$$

If we can show that these equations have a unique solution, then the natural cubic spline 4.2.6 with parameters $d_1, \dots, d_m, a_0, a_1$ corresponding to the solution will be a

(unique) interpolating natural cubic spline. It suffices to show that the matrix on the left in 4.2.9 has full rank. To do this, we show that (Using **0** to denote a vector of m zeroes)

$$\begin{pmatrix} \boldsymbol{E} & \mathbf{1} & \boldsymbol{u} \\ \mathbf{1}` & 0 & 0 \\ \boldsymbol{u}` & 0 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{d} \\ a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ 0 \\ 0 \end{pmatrix} \qquad (4.2.10)$$

implies

$$\begin{pmatrix} \boldsymbol{d} \\ a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ 0 \\ 0 \end{pmatrix}. \qquad (4.2.11)$$

If 4.2.10 holds, we have

$$(\boldsymbol{d}` \quad a_0 \quad a_1) \begin{pmatrix} \boldsymbol{E} & \mathbf{1} & \boldsymbol{u} \\ \mathbf{1}` & 0 & 0 \\ \boldsymbol{u}` & 0 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{d} \\ a_0 \\ a_1 \end{pmatrix} = 0, \qquad (4.2.12)$$

but if $s(x)$is the natural cubic spline with the parameters $d_1, \dots, d_m, a_0, a_1$ in 4.2.12, then by lemma 4.2.4 and 4.2.4

$$(\boldsymbol{d}` \quad a_0 \quad a_1) \begin{pmatrix} \boldsymbol{E} & \mathbf{1} & \boldsymbol{u} \\ \mathbf{1}` & 0 & 0 \\ \boldsymbol{u}` & 0 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{d} \\ a_0 \\ a_1 \end{pmatrix}$$

$$= \sum_{j=1}^{m} \sum_{k=1}^{m} d_j d_k e_{jk} + 2a_0 \sum_{j=1}^{m} d_j + 2a_1 \sum_{j=1}^{m} d_j u_j$$

$$= \int (s(x))^2 dx \ . \qquad (4.2.13)$$

But 4.2.12 and 4.2.13 put together implies that $s(x)$ must be linear, which means that the jumps in the third derivative, i.e. the d_j, are all zero. Furthermore, using this in 4.2.8 we get

$$a_0 + a_1 u_1 = 0$$

$$a_0 + a_1 u_2 = 0$$

...

$$a_0 + a_1 u_m = 0.$$

Since the points $u_1 \dots u_m$are distinct, this implies that $a_1 = a_0 = 0$.

Finally, we have the results stating that among all twice differentiable functions with given values at certain points, the interpolating natural cubic spline minimize the integrated squared second derivative.

Theorem 4.2.2

(Ohlsson and Johannsen (2010), p.145)

Let $u_1 < \cdots < u_m$, let $f(.)$ be any twice continuously differentiable function and let $s(.)$ be the natural cubic spline satisfying $s(u_j) = f(u_j)$, $j = 1, \ldots, m$. Then, for any $a \leq u_1$ and $b \geq u_m$

$$\int_a^b (s``(x))^2 dx \leq \int_a^b (f``(x))^2 \, dx. \qquad (4.2.14)$$

Proof

It follows from Lemma 4.2.2 that s satisfies 4.2.1 for any twice differentiable h with $h(u_k) = 0, k = 1, \ldots, m$. Thus the result follows from Lemma 4.2.1.

We will proof that $(\boldsymbol{B}`\boldsymbol{W}\boldsymbol{B} + \lambda\Omega)\boldsymbol{\beta} = \boldsymbol{B}`\boldsymbol{W}\boldsymbol{y}$.

We are looking for a B-splines that minimizing the penalised deviance, which is given by

$$\Delta(f) = D(y, u) + \lambda \int_a^b (f``(x))^2 \, dx.$$

In the normal case this is

$$\Delta(s) = \sum_i w_i(y_i - s(x_i)^2) + \lambda \int_a^b (s``(x))^2 dx.$$

For a natural cubic spline, which is linear outside $[z_1, z_m]$we have

$$\int_a^b (s``(x))^2 dx = \int_{z_1}^{z_m} (s``(x))^2 dx.$$

Therefore only $s(x)$, $z_1 \leq x \leq z_m$is used in the Penalised deviance. On the interval $[z_1, z_m]$ we can write $s(x)$ as

$$s(x) = \sum_{j=1}^{m+2} \beta_j B_j(x)$$

where $B_1(x) \dots B_{m+2}(x)$ are cubic B-splines with knots $z_1, \dots, z_m$. Using this, the penalised deviance may be considered as a function of the parameters $\beta_1, \dots, \beta_{m+2}$ and becomes

$$\Delta\beta = \sum_i w_i(y_i - \sum_{j=1}^{m+2} \beta_j B_j(x))^2 + \lambda \sum_{j=1}^{m+2} \sum_{k=1}^{m+2} \beta_j \beta_k \, \Omega_{\mathrm{j,k}}$$

where

$$\Omega_{\mathrm{jk}} = \int_{z_1}^{z_m} B``_j(x) B``_k(x) dx.$$

The number Ω_{jk}will be calculated in a later section.

To find the minimizing $\beta_1, \dots, \beta_{m+2}$ we calculate the partial derivative

$$\frac{d\Delta}{d\beta_l} = -2 \sum_i w_i \left(y_i - \sum_{j=1}^{m+2} \beta_j B_j(x)\right) B_l(x_i) + 2\lambda \sum_{j=1}^{m+2} \beta_j \Omega_{\mathrm{jl}}.$$

Letting $I_k$denote the set of $i$for which $x_i = z_i$ we get

$$-2 \sum_i w_i \left(y_i - \sum_{j=1}^{m+2} \beta_j B_j(x_i)\right) B_l(x_i) = -2 \sum_{k=1}^{m} \sum_{i \epsilon I_k} w_i \left(y_i - \sum_{j=1}^{m+2} \beta_j B_j(z_k)\right) B_l(z_k)$$

$$= -2 \sum_{k=1..m} \widetilde{w_k} \left(\widetilde{y_k} - \sum_{j=1}^{m+2} \beta_j B_j(z_k)\right) B_l(z_k)$$

where $\widetilde{w_k} = \sum_{i \epsilon I_k} w_i$ and $\widetilde{y_k} = \frac{1}{\widetilde{w_k}} \sum_{i \epsilon I_k} w_i \, y_i$.

Setting the partial derivatives equal to zero we obtain the equations

$$\sum_{k=1}^{m} \sum_{j=1}^{m+2} \widetilde{w_k} \, \beta_j B_j(z_k) B_l(z_k) + \lambda \sum_{j=1}^{m+2} \beta_j \Omega_{\mathrm{jl}}$$

$$= \sum_{k=1}^{m} \widetilde{w_k} \, \widetilde{y_k} B_l(z_k), l = 1, \dots, m+2.$$

The $m \times (\mathrm{m}+2)$ matrix $\boldsymbol{B}$

$$\boldsymbol{B} = \begin{pmatrix} B_1(z_1) & \cdots & B_{m+2}(z_1) \\ \vdots & \ddots & \vdots \\ B_1(z_m) & \cdots & B_{m+2}(z_m) \end{pmatrix}$$

is introduced.

Let $\boldsymbol{W}$ denote the $m \times m$ diagonal matrix with $\widetilde{w_j}$ on the main diagonal and let Ω denote the symmetric $(m+2) \times (m+2)$ matrix with elements Ω_{ij}. Furthermore let $\boldsymbol{\beta}$ and $\boldsymbol{y}$ denote the column vectors with elements β_j and $\widetilde{y_k}$ respectively. The above may then be written as

$$(\boldsymbol{B}`\boldsymbol{W}\boldsymbol{B} + \lambda\boldsymbol{\Omega})\boldsymbol{\beta} = \boldsymbol{B}`\boldsymbol{W}\boldsymbol{y}.$$

Due to the way the B-splines are defined, the matrix $\boldsymbol{B}`\boldsymbol{W}\boldsymbol{B}$ and $\boldsymbol{\Omega}$ are banded which simplifies the numerical solution of the linear equation system.

We will now be defining certain ideas and then firstly proof that the B-spline is continuous differentiable. We will then prove that the B-spline of a certain order form a base for the spline of that order. We then compute the values in $\boldsymbol{\Omega}$ for the B-spline.

The start is to define a base for the step function with jumps at the knots. For $k = 1, \ldots, m-2$, put

$$B_{0,k}(x) = \begin{cases} 1, & u_k \leq x \leq u_{k+1} \\ 0, & \text{otherwise} \end{cases} \qquad (4.2.15)$$

and furthermore

$$B_{0,m-1}(x) = \begin{cases} 1, & u_{m-1} \leq x \leq u_m \\ 0, & \text{otherwise} \end{cases} . \qquad (4.2.16)$$

The step function can be written as a linear combination of the functions $B_{0,k}(x)$.

For $j \geq 0$ we define the B-splines recursively by

$$B_{j+1,k}(x) = \frac{x - u_{k-j-1}}{u_k - u_{k-j-1}} B_{j,k-1}(x) + \frac{u_{k+1} - x}{u_{k+1} - u_{k+j}} B_{j,k}(x) \qquad (4.2.17)$$

and 4.2.17 is equal to 0 if $k \leq 0$ or if $k \geq m + j$.

Let $u_k = u_1$ for $k \leq 0$ and $u_k = u_m$ for $k \geq m + 1$.

We can see that $B_{j,k}(x)$ is positive on (u_{k-j}, u_{j+1}) and 0 otherwise

It follows from the next proposition that $B_{j,k}(x)$ is continuously differentiable.

Proposition 4.2.1.

For $j \geq 1$ and $x \neq u_1, \ldots, u_m$ that

$$B`_{j+1,k}(x) = \frac{j+1}{u_k - u_{k-j-1}} B_{j,k-1}(x) - \frac{j+1}{u_{k+1} - u_{k-j}} B_{j,k}(x). \qquad (4.2.18)$$

Proof

For $j = 0$ we can seed it from the differentiating in 4.2.17.

Now assuming that 4.2.18 holds for j, we shall prove that it also holds for $j+1$.

Differentiating 4.2.17 we get

$$B`_{j+1,k}(x) = \frac{1}{u_k - u_{k-j-1}} B_{j,k-1}(x) + \frac{x - u_{k-j-1}}{u_k - u_{k-j-1}} B`_{j,k-1}(x) - \frac{1}{u_{k+1} - u_{k-j}} B_{j,k}(x) + \frac{u_{k+1} - x}{u_{k+1} - u_{k-j}} B`_{j,k}(x). \qquad (4.2.19)$$

By doing mathematical induction and also 4.2.17 this becomes

$$\frac{1}{u_k - u_{k-j-1}} \left(\frac{x - u_{k-j-1}}{u_{k-1} - u_{k-j-1}} B_{j-1,k-2}(x) + \frac{u_k - x}{u_k - u_{k-j}} B_{j-1,k-1}(x) \right)$$

$$+ \frac{x - u_{k-j-1}}{u_k - u_{k-j-1}} \left(\frac{j}{u_{k-1} - u_{k-j-1}} B_{j-1,k-2}(x) - \frac{j}{u_k - u_{k-j}} B_{j-1,k-1}(x) \right)$$

$$- \frac{1}{u_{k+1} - u_{k-j}} \left(\frac{x - u_{k-j}}{u_k - u_{k-j}} B_{j-1,k-1}(x) + \frac{u_{k+1} - x}{u_{k+1} - u_{k-j+1}} B_{j-1,k}(x) \right)$$

$$+ \frac{u_k + 1 - x}{u_k + 1 - u_{k-j}} \left(\frac{j}{u_k - u_{k-j}} B_{j-1,k-1}(x) - \frac{j}{u_{k+1} - u_{k-j+1}} B_{j-1,k}(x) \right).$$

This is simplified to become

$$\frac{j+1}{u_k - u_{k-j-1}} \left(\frac{x - u_{k-j-1}}{u_{k-1} - u_{k-j-1}} B_{j-1,k-2}(x) + \frac{u_k - x}{u_k - u_{k-j}} B_{j-1,k-1}(x) \right)$$

$$- \frac{j+1}{u_{k+1} - u_{k-j}} \left(\frac{x - u_{k-j}}{u_k - u_{k-j}} B_{j-1,k-1}(x) + \frac{u_{k+1} - x}{u_{k+1} - u_{k-j+1}} B_{j-1,k}(x) \right)$$

$$= \frac{j+1}{u_k - u_{k-j-1}} B_{j,k-1}(x) - \frac{j+1}{u_{k+1} - u_{k-j}} B_{j,k}(x). \qquad (4.2.20)$$

Thus 4.2.18 holds for any $j \geq 0$

From the following proposition we will see that the cubic B-spline are twice continuous differentiable and that the quadratic B-spline are continuous differentiable follows from the next proposition.

Proposition 4.2.2 For $j \geq 0$ and $x \neq u_1, \dots, u_m$

$$B``_{j+1,k}(x) = \frac{j+1}{u_k - u_{k-j-1}} B`_{j,k-1}(x) - \frac{j+1}{u_{k+1} - u_{k-j}} B`_{j,k}(x)\,. \qquad (4.2.21)$$

Proof

This follows directly from Proposition 4.2.1.

We prove the following important property before proving that the B-spline of a certain order form a base for the spline of that order.

Proposition 4.2.3

For any $j \geq 0$

$$\sum_k B_{j,k}(x) = 1\,,\ u_1 \leq x \leq u_m\,. \qquad (4.2.22)$$

Proof

Make use of induction. For $j = 0$ it is true.

Let it hold for a certain j 4.2.17 gives

$$\begin{aligned} \textstyle\sum_k B_{j+1,k}(x) &= \sum_k \frac{x - u_{k-j-1}}{u_k - u_{k-j-1}} B_{j,k-1}(x) + \sum_k \frac{u_{k+1} - x}{u_{k+1} - u_{k-j}} B_{j,k}(x) \\ &= \sum_k \frac{x - u_{k-j}}{u_{k+1} - u_{k-j}} B_{j,k}(x) + \sum_k \frac{u_{k+1} - x}{u_{k+1} - u_{k-j}} B_{j,k}(x) \\ &= \sum_k B_{j,k}(x) = 1. \qquad (4.2.23) \end{aligned}$$

The proof of the basis theorem uses the following lemma.

Lemma 4.2.5

Suppose $s(x)$ is a linear combination of B-splines of order $j+1$,

$$s(x) = \sum_k \propto_k B_{j+1,k}(x)$$

then

$$s`(x) = \sum_k (j+1)\frac{\propto_{k+1} - \propto_k}{u_{k+1} - u_{k-j}} B_{j,k}(x).$$

Proof

This is a consequence of Proposition 4.2.1.

Theorem 4.2.3

For a given set of knots, a spline of order j may be written as

$$s(x) = \sum_{k=1}^{m=j+1} \beta_k B_{j,k}$$

for unique constants $\beta_1, \ldots, \beta_{m+j-1}$.

Proof

For $j = 1$ we may write

$$s(x) = \sum_{k=1}^{m} \beta_k B_{1,k}(x) \qquad (4.2.24)$$

with $\beta_k = s(u_k)$.

The two sides of 4.2.24 coincides since they are both linear splines and agree at the knots.

Let it hold for $j \geq 1$ and consider a spline s of order $j+1$. Since $s`$ is a spline of order j, we can write it as

$$s`(x) = \sum_k \beta_k B_{j,k}.$$

Now let $\propto_1 = 0$ and define $\propto_2, \ldots, \propto_{m+j-1}$ recursively by

$$\propto_{j+1} = \propto_j + \beta_j \frac{u_{k+1} - u_{k-j}}{j+1}.$$

If we define

$$S(x) = \sum_{k=1}^{m+j-1} \propto_k B_{j+1,k}(x),$$

then by lemma 4.2.5 we have $S`(x) = s`(x)$ and $S(u_1) = 0$. Thus, using Proposition 4.2.3.

$$\begin{aligned} s(x) &= s(u_1) + \int_{u_1}^{x} s`(y)dy = s(u_1) + S(x) \\ &= \sum_{k=1}^{m+j-1} s(u_1)B_{j+1,k}(x) + \sum_{k=1}^{m+j-1} \propto_k B_{j+1,k}(x) \\ &= \sum_{k=1}^{m+j-1} (s(u_1) + \propto_k) B_{j+1,k}(x) \ . \end{aligned}$$

Given that $\propto_1 = 0$ the coefficients $\propto_2, \dots, \propto_{m+j-1}$ are determined by the values of $\beta_1, \dots, \beta_{m+j-2}$. If we set $\propto_1$ to anything else it would still hold that $S(x) - S(u_1)$ would be determined by $\beta_1, \dots, \beta_{m+j-2}$. Thus uniqueness for splines of degree j implies uniqueness for splines of degree $j + 1$.

Computation of the $\boldsymbol{\Omega}$ matrix follow. Utilizing the recursion formulae derived above. Put

$$\Omega_{1\mathrm{kl}} = \int B_{1,k}(x)B_{1,l}(x)dx,$$

$$\Omega_{2\mathrm{kl}} = \int B_{2k}^{`}(x)B_{2l}^{`}(x)dx,$$

$$\Omega_{3\mathrm{kl}} = \int B_{3k}^{``}(x)B_{3l}^{``}(x)dx.$$

Let $u_1, \dots, u_m$ denote the knots. Recall the convention $u_k = u_1$ for $k \leq 0$ and $u_k = u_m$ for $k \geq m + 1$. With these in mind put

$$a_{2k} = \frac{2}{u_{k+1} - u_{k-1}}, k = 1, \dots, m$$

$$a_{3k} = \frac{3}{u_{k+1} - u_{k-2}}, k = 1, \dots, m + 1.$$

The following proposition describes how to compute the $\boldsymbol{\Omega}$ matrix. Note that, due to the convention $B_{jk} = 0$ for $k \leq 0$ and $k \geq m + 1$, certain Ω_{jkl} occurring in the formulae below are zero.

Proposition 4.2.4

(Ohlsson and Johannsen (2010), p.151)

The numbers Ω_{jkl} can be computed successfully as follows

The values in Ω is equal to 0 except for the following cases

$$\Omega_{1k,k} = \frac{u_{k+1} - u_{k-1}}{3}, k = 1, \dots, m$$

$$\Omega_{1k,k+1} = \Omega_{1k+1,k} = \frac{u_{k+1} - u_k}{6}, k = 1, \dots, m-1$$

$$\Omega_{2k,k} = a_{2,k-1}^2 \Omega_{1k-1,k-1} - 2a_{2,k-1}a_{2,k}\Omega_{1k-1,k} + a_{2,k}^2 \Omega_{1k,k}$$

$$k = 1, \dots, m+1$$

$$\Omega_{2k,k+1} = \Omega_{2k+1,k} = a_{2,k-1}a_{2,k}\Omega_{1k-1,k} - a_{2,k}^2 \Omega_{1k,k} + a_{2,k}a_{2,k+1}\Omega_{1k,k+1}$$

$$k = 1, \dots, m$$

$$\Omega_{2k,k+2} = \Omega_{2k+2,k} = -a_{2,k}a_{2,k+1}\Omega_{1k,k+1};\ k = 1, \dots, m-1$$

$$\Omega_{3k,k} = a_{3,k-1}^2 \Omega_{2k-1,k-1} - 2a_{3,k-1}a_{3,k}\,\Omega_{2k-1,k} + a_{3,k}^2 \Omega_{2k,k}$$

$$k = 1, \dots, m+2$$

$$\Omega_{3k,k+1} = \Omega_{3k+1,k} = a_{3,k-1}\,a_{3,k}\Omega_{2k-1,k} - a_{3,k-1}a_{3,k+1}\,\Omega_{2k-1,k+1}$$

$$- a_{3,k}^2 \Omega_{2k,k} + a_{3,k}a_{3,k+1}\Omega_{2k,k+1}$$

$$k = 1, \dots, m+1$$

$$\Omega_{3k,k+2} = \Omega_{3k+2,k} = a_{3,k-1}\,a_{3,k+1}\,\Omega_{2k-1,k+1} - a_{3,k}\,a_{3,k+1}\Omega_{2k,k+1}$$

$$+a_{3,k}a_{3,k+2}\Omega_{2k,k+2} + a_{3,k}\,a_{3,k+1}\,\Omega_{2k,k+1}$$

$$k = 1, \dots, m$$

$$\Omega_{3k,k+3} = \Omega_{3k+3,k} = -a_{3,k}a_{3,k+2}\Omega_{2k,k+2}\ \ k = 1, \dots, m-1.$$

Proof

As the computation in the various cases are similar, we do some of them.

Using the basic recursion formula we have

$$\begin{aligned}
\Omega_{1kk} &= \int B_{1,k}(x)B_{1,k}(x)dx \\
&= \int (\frac{x- u_{k-1}}{u_k- u_{k-1}} B_{0,k-1}(x) + \frac{u_{k+1}-x}{u_{k+1}- u_k} B_{0,k}(x))^2 dx \\
&= \int [\left(\frac{x- u_{k-1}}{u_k- u_{k-1}}\right)^2 B_{0,k-1}(x) + (\frac{u_{k+1}-x}{u_{k+1}- u_k})^2 B_{0,k}(x)]\, dx \\
&= \int_{u_k-1}^{u_k} \left(\frac{x- u_{k-1}}{u_k- u_{k-1}}\right)^2 dx + \int_{u_k}^{u_{k+1}} (\frac{u_{k+1}-x}{u_{k+1}- u_k})^2 dx \\
&= \frac{u_k-u_{k-1}}{3} + \frac{u_{k+1}-u_k}{3} \\
&= \frac{u_{k+1}-u_{k-1}}{3}\,.
\end{aligned}$$

Using proposition 4.2.1 we get

$$\begin{aligned}
\Omega_{2,k,k+1} &= \int B`_{2,k}(x)B`_{2,k+1}(x)dx \\
&= \int (a_{2,k-1}\, B_{1,k-1}(x) - a_{2,k}B_{1,k}(x)) \left(a_{2,k}B_{1,k}(x) - a_{2,k+1}B_{1,k+1}(x)\right) dx \\
&= a_{2,k-1}a_{2,k}\Omega_{1k-1,k} - a_{2,k}^2\Omega_{1k,k} + a_{2,k}a_{2,k+1}\Omega_{1k,k+1}\,.
\end{aligned}$$

Using proposition 4.2.2 we get

$$\begin{aligned}
\Omega_{3,k,k+2} &= \int B``_{3,k}(x)B``_{3,k+2}(x)dx \\
&= \int (a_{3,k-1}\, B`_{2,k-1}(x) - a_{3,k}B`_{2,k}(x)) \left(a_{3,k+1}B`_{2,k+1}(x) - a_{3,k+2}B`_{2,k+2}(x)\right) dx \\
&= a_{3,k-1}\, a_{3,k+1}\, \Omega_{2k-1,k+1} - a_{3,k}\, a_{3,k+1}\Omega_{2k,k+1} + a_{3,k}a_{3,k+2}\Omega_{2k,k+2} + a_{3,k}\, a_{3,k+1}\, \Omega_{2k,k+1}.
\end{aligned}$$

5. Conclusion

B splines, cubic splines and natural cubic splines were compared and it was concluded that B splines is the most orthogonal splines. B splines were therefore used in the computation of the natural cubic spline. The optimal solution of the penalized sum of square is a smoothing spline or a natural cubic spline with knots at unique X values.

Nonparametric linear regression and nonparametric logistic regression were computed by solving the smoothing spline. The optimal smoothing value is selected by using cross validation. Nonparametric regression was solved for a single explanatory variable.. The nonparametric regression, for multiple explanatory values, was solved using the local scoring algorithm.

A comparison was drawn between the logistic regression, logistic regression using natural cubic splines and nonparametric logistic regression using the local scoring algorithm. The use of the nonparametric logistic regression model increased the Gini coefficient when applied to the South African heart disease data set. The improved Gini coefficient points to the better discrimination ability of the nonparametric approach.

SAS software was developed to obtain estimates for these models. The software gives further insight into the techniques described and resolves some implementation issues.

Generalized additive models, GAM, with smooth functions specially structured were also considered. These models were structured as logistic regression models. The optimum solution of the GAM was achieved by applying the B-spline as preferred choice deducted from theoretical considerations. The estimated model parameters were obtained by using the local scoring algorithm.

The advantage of fitting GAM models were demonstrated in the applications through increased Gini indices. It was also demonstrated that the GAM models followed the data more closely.

References

Coetzee, J. (2009), A Roughness Penalty Approach to Nonparametric Regression, Unpublished MCom essay, Department of Statistics, University of Pretoria.

Fox, J. (2002), Nonparametric regression, Appendix to An R and S-PLUS Companion to Applied Regression.

Grgic, V. (2008). Smoothing splines in non-life insurance pricing, Examensarbete 2008:3, Department of Mathematical Statistics, Stockholm University.

Hastie, T., Tibshirani, R. (1987). Nonparametric Logistic and Proportional Odds Regression, *Journal of the Royal Statistical Society Series C (Applied Statistics),* 36(3): 260-276

Hastie, T., Tibshirani, R. (1990), Generalized Additive Models, Monographs on Statistics and Applied Probability 43, Chapman & Hall/CRC

Hastie, T., Tibshirani, R., Friedman, J. (2009). The Elements of statistical learning, Second Edition, Springer Series in Statistics.

Ohlsson, E., Johannsen, B., (2010). Non–Life Insurance Pricing with Generalized Linear models, First Edition, Springer European Actuarial Academy.

SAS Publishers, (2004), The SAS/IML 9.1 Users Guide.

SAS Publishers, (2006), SQL Processing with the SAS System

Appendix A: Data used in examples

X	Y
-6.78319	0.200187
-6.70319	0.420399
-6.62319	0.462835
-6.54319	0.593302
-6.46319	0.501491
-6.38319	0.426591
-6.30319	0.510364
-6.22319	0.496568
-6.14319	0.473283
-6.06319	0.424294
-5.98319	0.4251
-5.90319	0.323516
-5.82319	0.432165
-5.74319	0.490004
-5.66319	0.576899
-5.58319	0.393625
-5.50319	0.315579
-5.42319	0.430061
-5.34319	0.356185
-5.26319	0.299021
-5.18319	0.265149
-5.10319	0.084859
-5.02319	0.078071
-4.94319	-0.07473
-4.86319	0.056993
-4.78319	0.120655
-4.70319	0.011055
-4.62319	-0.05375
-4.54319	-0.17615
-4.46319	-0.31095
-4.38319	-0.21262
-4.30319	-0.31589
-4.22319	-0.31708
-4.14319	-0.33089
-4.06319	-0.24326
-3.98319	-0.16606
-3.90319	-0.31108
-3.82319	-0.39706

-3.74319	-0.5528
-3.66319	-0.37371
-3.58319	-0.39473
-3.50319	-0.40378
-3.42319	-0.40497
-3.34319	-0.51036
-3.26319	-0.33094
-3.18319	-0.52791
-3.10319	-0.55727
-3.02319	-0.41842
-2.94319	-0.5023
-2.86319	-0.49542
-2.78319	-0.67924
-2.70319	-0.40497
-2.62319	-0.42123
-2.54319	-0.4481
-2.46319	-0.38957
-2.38319	-0.24392
-2.30319	-0.39468
-2.22319	-0.27361
-2.14319	-0.33393
-2.06319	-0.26542
-1.98319	-0.08777
-1.90319	-0.37744
-1.82319	-0.01643
-1.74319	-0.05473
-1.66319	0.008861
-1.58319	-0.09968
-1.50319	0.090492
-1.42319	-0.04426
-1.34319	0.005322
-1.26319	0.340338
-1.18319	-0.00228
-1.10319	0.218881
-1.02319	0.174486
-0.94319	0.254025
-0.86319	0.314767
-0.78319	0.346958
-0.70319	0.499408
-0.62319	0.406023
-0.54319	0.494836

-0.46319	0.330517
-0.38319	0.531323
-0.30319	0.274598
-0.22319	0.513126
-0.14319	0.509466
-0.06319	0.439689
0.016815	0.39868
0.096815	0.347559
0.176815	0.573732
0.256815	0.641722
0.336815	0.545771
0.416815	0.531407
0.496815	0.395529

Appendix B: Random component, link function and systematic component

The generalized linear model consists of a random component, systematic component and link function. The link function links the random component and the systematic component. The response Y is assumed to have exponential density

$$\rho_{Y(y|\theta|\phi)=\exp\left\{\frac{y\theta-b(\theta)}{a(\phi)}+c(y,\phi)\right\}.}$$

The random component consists of θ, the natural parameter, and ϕ, the dispersion parameter. The mean of the response $E(Y) = \mu$ is related to the covariates $X_1 \ldots X_p$ by $g(\mu) = \eta$ the systematic component. If $\eta = \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k$ the systematic component is called the linear predictor. The function g is known as the link function.

Appendix C: SAS code, B-spline macro

```
/*calculate the B-spline*/

%macro b_splines(data =.,var=.,y=.);

/*knots*/

proc sql;
create table original_data as
select
row,
&var. ,
&y.
from &data.;
```

```
quit;

data original_data_knot;
set original_data;

/*quantiles*/
if row in ( ;

/*0.98     3.28  4.34  5.8   15.33*/

run ;

proc sql;
      select count(&var.)
            into :num_knots
      from original_data_knot;
quit;

proc sql;
      select count(&var.)-1
            into :n_knots_min_one
      from original_data_knot;
quit;

proc sql;
      select count(&var.)+1
            into :n_knots_plus_one
      from original_data_knot;
quit;


%DO k = 1 %TO &num_knots.;

      data original_data_&k.;
            set  original_data_knot;
            if _N_ = &k.;
      run;

      proc sql;
            select &var.
                  into : knot_value
            from original_data_&k.;
      quit;


    proc datasets nolist nowarn nodetails;
      delete original_data_&k.;
      run;

            data original_data;
        set  original_data;
```

```
/*         if  <= &var. < &knot_value. then h_&k. = (&var. -
&knot_value.)**3;*/
/*        else  h_&k. = 0;*/
           e&k. = &knot_value.;
           run;

%end;

proc iml;
use original_data;
read all into xdat;
x = xdat[,2];
knots = xdat[1,4:ncol(xdat)]`;

/*b zero*/
b_zero = j(nrow(x),nrow(knots)-1,0);
do j = 1 to nrow(x);
do i = 1 to nrow(knots)-1;
     if x[j,] >= knots[i,] then do;
     if x[j,] < knots[i+1,] then b_zero[j,i]=1;
     end;
end;
end;

/*print b_zero;*/

/*b_one*/
k=0;
m = nrow(knots);
j = nrow(x);
b_one = j(j,m+k,0);

do j = 1 to nrow(x);
     b_one[j,1] =  (knots[2,]-x[j,1])/(knots[2,]-
knots[1,])#b_zero[j,1] ;
end;

do j = 1 to nrow(x);
do i = 2 to m+k-1;

b_one[j,i] =
((x[j,1] - knots[max(i-k-1,1),])/(knots[min(i,m),]-knots[max(i-k-
1,1),]))#b_zero[j,i-1]+
(( knots[min(i+1,m),] -x[j,1] )/(knots[min(i+1,m),]-knots[max(i-
k,1),]))#b_zero[j,i]
;

end;
end;

do j = 1 to nrow(x);
```

```
      b_one[j,m+k] =  (x[j,1]-knots[max(m-1,1),])/(knots[m,]-knots[(m-
1),])#b_zero[j,m+k-1] ;
end;

/*print b_one;*/

/*b_two*/
k=1;
m = nrow(knots);
j = nrow(x);
b_two = j(j,m+k,0);

do j = 1 to nrow(x);
      b_two[j,1] =  (knots[2,]-x[j,1])/(knots[2,]-knots[1,])#b_one[j,1]
;
end;

do j = 1 to nrow(x);
do i = 2 to m+k-1;

b_two[j,i] =
((x[j,1] - knots[max(i-k-1,1),])/(knots[min(i,m),]-knots[max(i-k-
1,1),]))#b_one[j,i-1]+
(( knots[min(i+1,m),] -x[j,1] )/(knots[min(i+1,m),]-knots[max(i-
k,1),]))#b_one[j,i]
;

end;
end;

do j = 1 to nrow(x);
      b_two[j,m+k] =  (x[j,1]-knots[max(m-1,1),])/(knots[m,]-knots[(m-
1),])#b_one[j,m+k-1] ;
end;

/*print b_two;*/


/*b_three*/
k=2;
m = nrow(knots);
j = nrow(x);
b_three = j(j,m+k,0);

do j = 1 to nrow(x);
      b_three[j,1] =  (knots[2,]-x[j,1])/(knots[2,]-
knots[1,])#b_two[j,1] ;
end;

do j = 1 to nrow(x);
do i = 2 to m+k-1;
```

```
b_three[j,i] =
((x[j,1] - knots[max(i-k-1,1),])/(knots[min(i,m),]-knots[max(i-k-
1,1),]))#b_two[j,i-1]+
(( knots[min(i+1,m),] -x[j,1] )/(knots[min(i+1,m),]-knots[max(i-
k,1),]))#b_two[j,i]
;

end;
end;

do j = 1 to nrow(x);
     b_three[j,m+k] =  (x[j,1]-knots[max(m-1,1),])/(knots[m,]-
knots[(m-1),])#b_two[j,m+k-1] ;
end;

/*---try ---------*/

a = inv(b_three`*b_three);
print a;
print b_three;
/*call svd(U,Q,V,b_three);*/

/*do i = 1 to nrow(q);*/
/*if q[i,]< 0.01 then q[i,] = 0.01;*/
/*end;*/

/*New_b_three = U*diag(Q)*V`;*/
/*S = inv( I(nrow(xstd)) +
lamda#(u`*inv(diag(Q))*v`*omega_ridge*V*inv(diag(Q))*U));*/
/*a  = inv(New_b_three`*New_b_three);*/

/*--------------*/

 create b_three from b_three; append from b_three;

quit;

%mend;

/**/
/*omega*/
/*B-spline omega*/

%macro omega_bspline(data =.,var=.,y=. );

proc iml;
use ORIGINAL_DATA ;
read all into xdat;
knots = xdat[1,4:ncol(xdat)]`;
m = nrow(knots);

/*a_twee - bl 151 Olshon*/
```

```
a_twee = j(m,1,0);

a_twee[1,] = 2/(knots[2,] - knots[1,]);
a_twee[m,] = 2/(knots[m,] - knots[m-1,]);

do k = 2 to m-1;
a_twee[k,] = 2/(knots[K+1,] - knots[K-1,]);
end;

/*a_drie*/
a_drie = j(m+1,1,0);
a_drie[1,] = 3/(knots[2,] - knots[1,]);
a_drie[2,] = 3/(knots[3,] - knots[K-2,]);
a_drie[m,] = 3/(knots[m,] - knots[m-2,]);
a_drie[m+1,] = 3/(knots[m,] - knots[m-1,]);

do k = 3 to m-1;
a_drie[k,] = 3/(knots[K+1,] - knots[K-2,]);
end;

/*print a_twee a_drie;*/
omega_een= j(m,m,0);

omega_een[1,1] = (knots[2,] - knots[1,])/3;
omega_een[m,m] = (knots[m,] - knots[m-1,])/3;
do k = 2 to m-1;
 omega_een[k,k] = (knots[K+1,] - knots[K-1,])/3;
end;

do k = 1 to m-1;
omega_een[k,k+1] = (knots[K+1,] - knots[K,])/6;
/*is the following correct - not as in text book*/
omega_een[k+1,k] = (knots[K+1,] - knots[K,])/6;
end;

/*print omega_een;*/

/*omega_twee*/
omega_twee = j(m+1,m+1,0);


omega_twee[1,1] =
(a_twee[1,]##2)#(omega_een[1,1])
-2#(a_twee[1,])#(a_twee[1,])#(omega_een[1,1])
+(a_twee[1,]##2)#(omega_een[1,1])
;
omega_twee[m+1,m+1] =
(a_twee[m,]##2)#(omega_een[m,m])
-2#(a_twee[m,])#(a_twee[m,])#(omega_een[m,m])
+(a_twee[m,]##2)#(omega_een[m,m])
;
do k = 2 to m;
```

```
omega_twee[k,k] =
(a_twee[k-1,]##2)#(omega_een[k-1,k-1])
-2#(a_twee[k-1,])#(a_twee[k,])#(omega_een[k-1,k])
+(a_twee[k,]##2)#(omega_een[k,k])
;
end;


omega_twee[1,2]=
 (a_twee[1,])#(a_twee[1,])#(omega_een[1,1])
-((a_twee[1,])##2)#(omega_een[1,1])
+(a_twee[1,])#(a_twee[2,])#(omega_een[1,2])
;
omega_twee[2,1]=omega_twee[1,2];

omega_twee[m,m+1]=
 (a_twee[m-1,])#(a_twee[m,])#(omega_een[m-1,m])
-((a_twee[m,])##2)#(omega_een[m,m])
+(a_twee[m,])#(a_twee[m,])#(omega_een[m,m])
;
omega_twee[m+1,m]=omega_twee[m,m+1];


do k = 2 to m-1 ;
omega_twee[k,k+1]=
 (a_twee[k-1,])#(a_twee[k,])#(omega_een[k-1,k])
-((a_twee[k,])##2)#(omega_een[k,k])
+(a_twee[k,])#(a_twee[k+1,])#(omega_een[k,k+1])
;
omega_twee[k+1,k]=omega_twee[k,k+1];
end;

do k = 1 to m-1;
omega_twee[k,k+2]= -(a_twee[k,])#(a_twee[k+1,])#(omega_een[k,k+1]);
omega_twee[k+2,k]=omega_twee[k,k+2];
end;

/*print omega_twee a_twee;*/

omega_drie = j(m+2,m+2,0);

/*1 m+1 m+2*/

omega_drie[1,1] =
((a_drie[1,])##2)#(omega_twee[1,1])
-2#(a_drie[1,])#(a_drie[1,])#(omega_twee[1,1])
+((a_drie[1,])##2)#(omega_twee[1,1])
;
omega_drie[m+1,m+1] =
((a_drie[m,])##2)#(omega_twee[m,m])
-2#(a_drie[m,])#(a_drie[m,])#(omega_twee[m,m])
+((a_drie[m,])##2)#(omega_twee[m,m])
```

```
;
omega_drie[m+2,m+2] =
((a_drie[m,])##2)#(omega_twee[m,m])
-2#(a_drie[m,])#(a_drie[m,])#(omega_twee[m,m])
+((a_drie[m,])##2)#(omega_twee[m,m])
;

do k = 2 to m;
omega_drie[k,k] =
((a_drie[k-1,])##2)#(omega_twee[k-1,k-1])
-2#(a_drie[k-1,])#(a_drie[k,])#(omega_twee[k-1,k])
+((a_drie[k,])##2)#(omega_twee[k,k])
;
end;

omega_drie[1,2] =
(a_drie[1,])#(a_drie[1,])#(omega_twee[1,1])
-(a_drie[1,])#(a_drie[2,])#(omega_twee[1,2])
-((a_drie[1,])##2)#(omega_twee[1,1])
+(a_drie[1,])#(a_drie[2,])#(omega_twee[1,2])
;

omega_drie[m+1,m+2] =
(a_drie[m,])#(a_drie[m,])#(omega_twee[m,m])
-(a_drie[m,])#(a_drie[m,])#(omega_twee[m,m])
-((a_drie[m,])##2)#(omega_twee[m,m])
+(a_drie[m,])#(a_drie[m,])#(omega_twee[m,m])
;

omega_drie[m,m+1] =
(a_drie[m-1,])#(a_drie[m,])#(omega_twee[m-1,m])
-(a_drie[m-1,])#(a_drie[m+1,])#(omega_twee[m-1,m])
-((a_drie[m-1,])##2)#(omega_twee[m,m])
+(a_drie[m,])#(a_drie[m,])#(omega_twee[m,m])
;


do k = 2 to m-1;
omega_drie[k,k+1] =
(a_drie[k-1,])#(a_drie[k,])#(omega_twee[k-1,k])
-(a_drie[k-1,])#(a_drie[k+1,])#(omega_twee[k-1,k+1])
-((a_drie[k-1,])##2)#(omega_twee[k,k])
+(a_drie[k,])#(a_drie[k+1,])#(omega_twee[k,k+1])
;
omega_drie[k+1,k] = omega_drie[k,k+1];
end;

 omega_drie[1,3]=
 (a_drie[1,])#(a_drie[2,])#(omega_twee[2,2])
-(a_drie[1,])#(a_drie[2,])#(omega_twee[1,2])
+(a_drie[1,])#(a_drie[3,])#(omega_twee[1,3])
;
```

```
omega_drie[m,m+2]=
 (a_drie[m-1,])#(a_drie[m,])#(omega_twee[m-1,m])
-(a_drie[m,])#(a_drie[m,])#(omega_twee[m,m])
+(a_drie[m,])#(a_drie[m,])#(omega_twee[m,m])
;

 omega_drie[m-1,m+1]=
 (a_drie[m-2,])#(a_drie[m,])#(omega_twee[m-2,m])
-(a_drie[m-1,])#(a_drie[m,])#(omega_twee[m-1,m])
+(a_drie[m-1,])#(a_drie[m,])#(omega_twee[m-1,m])
;

do k = 2 to m-2;
 omega_drie[k,k+2]=
 (a_drie[k-1,])#(a_drie[k+1,])#(omega_twee[k-1,k+1])
-(a_drie[k,])#(a_drie[k+1,])#(omega_twee[k,k+1])
+(a_drie[k,])#(a_drie[k+2,])#(omega_twee[k,k+2])
;
omega_drie[k+2,k]=omega_drie[k,k+2];
end;

omega_drie[m-1,m+2] =
-(a_drie[m-1,])#(a_drie[m,])#(omega_twee[m-1,m]);

do k = 1 to m-2;
omega_drie[k,k+3] =
-(a_drie[k,])#(a_drie[k+2,])#(omega_twee[k,k+2])
;
omega_drie[k+3,k] = omega_drie[k,k+3];
end;

create omega_drie from omega_drie; append from omega_drie;

quit;

data omega_drie;
set omega_drie;
if  col1 = . then col1 = 0;
if  col2 = . then col2 = 0;
if  col3 = . then col3 = 0;
if  col4 = . then col4 = 0;
if  col5 = . then col5 = 0;
if  col6 = . then col6 = 0;
if  col7 = . then col7 = 0;
/*if  col8 = . then col8 = 0;*/
/*if  col9 = . then col9 = 0;*/
/*if  col10 = . then col10 = 0;*/
/*if  col11 = . then col11 = 0;*/
/*if  col12 = . then col12 = 0;*/
/*if  col13 = . then col13 = 0;*/
/*if  col14 = . then col14 = 0;*/
```

```
/*if  col15 = . then col15 = 0;*/
/*if  col16 = . then col16 = 0;*/
/*if  col17 = . then col17 = 0;*/
/*if  col18 = . then col18 = 0;*/
/*if  col19 = . then col19 = 0;*/
/*if  col20 = . then col20 = 0;*/
run;

%mend;

/*standardize variables */
%macro standardize_variable(data=.,y=.,var=.);

data b;
set &data.;
keep &var.;
run;

proc iml ;
use b;
read all into A;

use &data;
read all into all_data;

Standerdize_var = (A - A[:])/sqrt((((A -A[:])##2)[+,])/((nrow(A))-1))
;

f = all_data[,1]||Standerdize_var;

create standard from f [ colname = { "row" , "standard_&var." } ];
append from f ;

proc sort data =  standard;
by row;
run ;

proc sort data = &data.;
by row;
run;

data &data.;
merge &data. standard;
by row;
run;

data &data. (drop = standard_&var.) ;
set &data.;
&var. = standard_&var.;
run;
quit;
%mend;
```

Appendix D: Logistic regression using natural cubic spline transformations on South African heart disease data

```
libname examples "C:\Documents and
Settings\a138617\Desktop\Nonparametric logistic regression";

data EXAMPLES.SAHEARTDIS;
set EXAMPLES.SAHEARTDIS_original;
wt=1;
row = _n_;
run;

%macro natural_cubic_spline(data =, var = , weight = , groups = ,
output = ,y=);

data _tmp1(keep = &var &weight);
  set &data;
run;

/*create quantiles*/

proc univariate data = &data noprint;
  var &var;
  weight &weight;
  output out = _tmp2
  pctlpre = decile_
  pctlpts = 0 to 100 by %sysevalf(100 / &groups);
run;
proc contents data = _tmp2 out = _tmp3;
run ;

proc sql;
     select count(NAME)
          into :num
     from _tmp3;
quit;

%DO k = 1 %TO &num.;
     data _tmp3_&k.;
          set  _tmp3;
          if _N_ = &k.;
     run;

     proc sql;
          select NAME
               into :dec_name
          from _tmp3_&k.;
     quit;

     proc sql;
          select &dec_name.
```

```
                into :dec_value
            from _tmp2;
      quit;

/*create cubic splines*/

data &data;
set  &data;

x = &var.;

if X > &dec_value. then h&k. = (X- &dec_value.)**3;
else h&k. = 0;

e&k. = &dec_value.;
run;

%end;

/*create natural cubic splines*/

data &data;
set  &data;
n2 = X;
n3 = (h1-h5)/(e5-e1) - (h4-h5)/(e5-e4);
n4 = (h2-h5)/(e5-e2) - (h4-h5)/(e5-e4);
n5 = (h3-h5)/(e5-e3) - (h4-h5)/(e5-e4);
run;

/*regression for natural cubic splines*/
data regression_data;
set &data;
keep row &y.  n2 n3 n4 n5;
run;

proc sql;
create table regression_data as
select row, &y.,  n2, n3, n4, n5
from regression_data;
quit;

proc iml ;

use regression_data ;
read all into xdat;

n=nrow(xdat);
l = ncol(xdat);

x = xdat[,3:l];
y = xdat[,2] ;
```

```
/*print x y;*/

*Start the algorithm by choosing betas to be zero;
bhi = J(ncol(x),1,0);
print bhi;
bhold = bhi;
diff = 99999 ;

do i = 1 to 10  until (diff < 0.000001);
  pr1 = exp(x*bhold);*Get all the entries for the Newton Raphson
method - y X p W;
  pr2 = 1/(1+exp(x*bhold)) ;
  p = pr1#pr2 ;

/*print p ;*/

  w1 = p#(1-p) ;
  w = diag(w1) ;
/*print w ;*/

z  = x*bhold + inv(w)*(y-p);
bhnew = inv(x`*w*x)*x`*w*z ;

print i bhold bhnew;

diff=abs(max(bhnew-bhold));
bhold=bhnew;
end;


/*print bh ;*/
yhp = x*bhnew ;

h = x#bhnew`;
/*print h;*/
/*print yhp;*/
/*print x;*/
/*print bh; */


data_p = xdat[,1] || yhp ;

nm = { "row"  "yh_&var."} ;

create data_y_&var. from data_p [ colname = nm ];
append from data_p ;

bk = { "1_bh1" "n2_bh2" "n3_bh3" "n4_bh4" "n5_bh5"  } ;
create h_&var. from h [colname = bk];
append from h;

quit;
```

```
proc sort data = &data.;
by row;
run;
proc sort data = data_y_&var.;
by row;
run;

data &data.;
merge &data. (in=a) data_y_&var. (in=b);
by row;
run;

/*clean up data set */
data &data.;
set &data.;
drop
x
h1 h2 h3 h4 h5
e1 e2 e3 e4 e5
n2 n3 n4 n5

;
run;

proc gplot data = &data.;
     plot
                yh_&var. * &var.;

run;


%mend;


%natural_cubic_spline(data = EXAMPLES.SAHEARTDIS,var = adiposity ,
weight = wt , groups = 4 ,output = test,y=chd);
%natural_cubic_spline(data = EXAMPLES.SAHEARTDIS,var = age , weight =
wt , groups = 4 ,output = test,y=chd);
%natural_cubic_spline(data = EXAMPLES.SAHEARTDIS,var = alcohol ,
weight = wt , groups = 4 ,output = test,y=chd);
%natural_cubic_spline(data = EXAMPLES.SAHEARTDIS,var = ldl , weight =
wt , groups = 4 ,output = test,y=chd);
%natural_cubic_spline(data = EXAMPLES.SAHEARTDIS,var = obesity ,
weight = wt , groups = 4 ,output = test,y=chd);
%natural_cubic_spline(data = EXAMPLES.SAHEARTDIS,var = sbp , weight =
wt , groups = 4 ,output = test,y=chd);
%natural_cubic_spline(data = EXAMPLES.SAHEARTDIS,var = tobacco ,
weight = wt , groups = 4 ,output = test,y=chd);
%natural_cubic_spline(data = EXAMPLES.SAHEARTDIS,var = typea , weight
= wt , groups = 4 ,output = test,y=chd);
/*create  binary variables with naming convension: yh_&var.*/
```

```
data EXAMPLES.SAHEARTDIS;
set EXAMPLES.SAHEARTDIS;

     if famhist = "Absent"  then yh_famhist= 0;
else if famhist = "Present" then yh_famhist= 1;

run;



/*logistic regression*/

%macro logistic_regression_spline(y=.,data=. );
/*proc contents data = &data. out= logistic_data_list;*/
/*run;*/

/*data logistic_data_list;*/
/*set logistic_data_list; */
/*if substrn(name,1,3) = "yh_" or name = "row" or name = "&y." ; */
/*keep name;*/
/*run ;*/
/**/
/*    proc sql;*/
/*         select NAME*/
/*              into :logistic_list*/
/*         from logistic_data_list; */
/*    quit;*/

proc sql;
create table logistic_data as
select
row, &y.,
yh_sbp,
yh_tobacco,
yh_ldl,
yh_famhist,
yh_obesity,
yh_age

/*yh_alcohol,*/
/*yh_adiposity,*/
/*yh_typea*/

/*&logistic_list.*/
/*change this*/
from &data.;
quit;

proc iml;

use logistic_data ;
```

```
read all into xdat;

n=nrow(xdat);
l = ncol(xdat);

x = J(n,1,1) || xdat[,3:l];
y = xdat[,2] ;

/*print x y;*/

*Start the algorithm by choosing betas to be zero;
bhi = J(ncol(x),1,0);
print bhi;
bhold = bhi;
diff = 99999 ;

do i = 1 to 10  until (diff < 0.000001);
  pr1 = exp(x*bhold);*Get all the entries for the Newton Raphson
method - y X p W;
  pr2 = 1/(1+exp(x*bhold)) ;
  p = pr1#pr2 ;

/*print p ;*/

  w1 = p#(1-p) ;
  w = diag(w1) ;
/*print w ;*/

z  = x*bhold + inv(w)*(y-p);
bhnew = inv(x`*w*x)*x`*w*z ;

print i bhold bhnew;

diff=abs(max(bhnew-bhold));
bhold=bhnew;
end;

*Calculate the standard error with Maximum likelihood estimators;
eq = exp(x*bhnew);
prob = eq/(1+eq);
compare = round(prob) || y;
accurate= compare[,1] = compare[,2];
std_error= 1 - (accurate[+,]/nrow(accurate));
print std_error;


/*aic and bic*/

  p   = exp(x * bhnew) / (1 + exp(x * bhnew));
  ll  = log(((p ## y) # ((1 - p) ## (1 - y)))[#, ]);
  aic = 2 * nrow(bhnew) - 2 * ll;
  bic = nrow(bhnew) * log(nrow(y)) - 2 * ll;
```

```
  print (ll||aic||bic)
        [colname = {"Log Likelihood", "AIC", "BIC"}
         format  = 8.3];


  /*aicc*/
n = nrow(y);
aicc = aic + (2*(nrow(bhnew))*(nrow(bhnew)+1))/(n-nrow(bhnew)-1);
print aicc n ;


/*get correlation matrix*/


/*get vif*/


/*quashi aic*/


/*---------*/
/*create f =h*beta and draw it , create confidance intervals*/
f = x#bhnew`;
predict = x * bhnew;
/*print  bhnew;*/
/*print x y f;*/


/*create for final model selested*/
data_f = xdat[,1]||f||predict;

nk = { "row" "beta_intercept" "yhbeta_sbp" "yhbeta_tobacco"
"yhbeta_ldl" "yhbeta_famhist" "yhbeta_obesity" "yhbeta_age"
"predict"};

create data_f from data_f [ colname = nk ];
append from data_f ;

quit;

%mend;

%logistic_regression_spline(y=chd , data=EXAMPLES.SAHEARTDIS);



/*draw the final graphs*/

%macro draw(data);

proc sort data = data_f;
```

```
by row;
run;
proc sort data = &data.;
by row;
run;
data &data.;
merge &data.(in=a) data_f;
by row;
if a;
run;

proc contents data = &data. out = temp noprint;
run;

data temp;
set temp;
if substrn(name,1,7) in ("yhbeta_");
keep name;
run;

proc sql;
     select count(NAME)
          into :num
     from temp;
quit;

%DO k = 1 %TO &num.;
     data temp_&k.;
          set  temp;
          if _N_ = &k.;
     run;
     proc sql;
          select NAME
               into :fbeta_name
          from temp_&k.;
     quit;

%let lente = %sysfunc(length(&fbeta_name.))- 7;
%let var = %sysfunc(substrn(&fbeta_name.,8,&lente.));

proc gplot data = &data.;
     plot &fbeta_name. * &var.;
run;

%end;
%mend;
%draw(EXAMPLES.SAHEARTDIS);
```

Appendix E: SAS code, cross validation on a natural cubic spline (example 2.4)

```
data examples.cos_data;
set a;
run;
proc sort data = examples.cos_data nodupkey;
by x;
run;

data examples.cos_data;
set examples.cos_data;
row = _n_;
run;

%macro natural_cubic_spline_no_knots(data =, var =  , output = ,y=);

proc sql;
create table original_data as
select
row,
&var. ,
&y.
from &data.;
quit;

proc sql;
     select count(&var.)
          into :num_knots
     from original_data;
quit;

proc sql;
     select count(&var.)-1
          into :n_knots_min_one
     from original_data;
quit;

proc sql;
     select count(&var.)+1
          into :n_knots_plus_one
     from original_data;
quit;


%DO k = 1 %TO &num_knots.;

     data original_data_&k.;
          set  original_data;
          if _N_ = &k.;
     run;
```

```
      proc sql;
            select &var.
                  into : knot_value
            from original_data_&k.;
      quit;


    proc datasets nolist nowarn nodetails;
      delete original_data_&k.;
      run;

            data &data;
         set  &data;
            if &var. > &knot_value. then h_&k. = (&var. -
&knot_value.)**3;
         else  h_&k. = 0;
            e&k. = &knot_value.;
            run;
%end;
%DO  J = 1 %TO  &num_knots.-2;

      data &data;
      set  &data;
      n_&j. = (h_&j.-
%sysfunc(compress(h_&num_knots.)))/(%sysfunc(compress(e&num_knots.))-
e&j.)
      -(%sysfunc(compress(h_&n_knots_min_one.))-
%sysfunc(compress(h_&num_knots.)))/(%sysfunc(compress(e&num_knots.))
      -%sysfunc(compress(e&n_knots_min_one.)));
      run;
%END;

data &data;
set  &data;
%sysfunc(compress(n_&n_knots_min_one.)) = &var.;
run;

%mend;

%natural_cubic_spline_no_knots(data =examples.cos_data,var = x ,output
= test,y=y);


/*new omega*/

%macro omega(data =.,var=.,y=. );
proc contents data = &data. out  = knot_names noprint;
run;

data knot_names (keep = name);
set knot_names;
```

```
if substrn(name,1,1) eq "e";
run;

      proc sql;
            select name
                  into :knotname separated by " "
            from knot_names;
      quit;

data knot_val;
set &data.;
keep
&knotname.
;
if _n_ = 1;
run;


proc iml;

use knot_val;
read all into knot_val;

/*print knot_val;*/

omega = j(ncol(knot_val)-2,ncol(knot_val)-2,0);

            do x= 1 to ncol(knot_val)-2;
            do y= 1 to ncol(knot_val)-2;

mx = (knot_val[,y]<>knot_val[,x]);
mi = min(knot_val[,y]><knot_val[,x]);

omega[x,y] =
((18#(knot_val[,ncol(knot_val)-1]-mx)#(knot_val[,ncol(knot_val)-1]-
mi)##2
+ 6#(mx-mi)##3
- 6#(knot_val[,ncol(knot_val)-1]-mi)##3)
+ 12#(knot_val[,ncol(knot_val)-1]-
knot_val[,y])#(knot_val[,ncol(knot_val)-1]-
knot_val[,x])#(knot_val[,ncol(knot_val)]-knot_val[,ncol(knot_val)-1]))
/((knot_val[,ncol(knot_val)]-knot_val[,y])#(knot_val[,ncol(knot_val)]-
knot_val[,x]))
;

            end;
            end;



/**/
```

```
/*do x= 1 to ncol(knot_val);*/
/*do y= 1 to ncol(knot_val);*/
/*if omega[x,y] = . then omega[x,y]=0;*/
/*end;*/
/*end;*/

/*put 0 's on inside - for the transformations 1 and x  */

omega = j(nrow(omega),2,0)||omega;
omega = j(2,ncol(omega),0)//omega;

create EXAMPLES.omega_male_new from omega; append from omega;
quit;

%mend;

%omega(data =examples.cos_data,var=x,y=y);


%macro iml_regression_data(data =.,var=.,y=.);

proc contents data = &data. out = logistic_names  noprint;
run ;

proc sort data = logistic_names ;
by varnum;
run;

data logistic_names;
set logistic_names;
where substrn(name,1,2) in ("n_" );
f_name = "f_"||name;
run;

proc sql;
select name into :logitic_list separated by ',' from logistic_names;
quit;

proc sql;
select f_name into :logitic_f_list separated by '","' from
logistic_names;
quit;

proc sql;
create table logistic_data as
select
row, &y.,
&logitic_list.
from &data.;
quit;
quit;
```

```
%mend;
%iml_regression_data(data =examples.cos_data,var=x,y=y);



/*linear regresion */
%macro ncs_lamda_omega(lamda=.);
proc iml;
use examples.omega_male_new ;
read all into omega;
use logistic_data ;
read all into xdat;

/*ridge regression:*/

n=nrow(xdat);
l = ncol(xdat);

/*een x die res van die splines*/
/*no interceprt x is the first value then the transformations follow*/

x =   xdat[,l]||xdat[,3:l-1];
y = xdat[,2] ;
/*x_leaveone = x[1:nrow(x)-1,];*/

one = j(nrow(x),1,1);
mean_x = one`*x/n;

/*x_centred = x_leaveone-one*mean_x;*/

m = nrow(x);
one = j(nrow(x),1,1);
mean_x = one`*x/m;
sigma = (x-one*mean_x)`*(x-one*mean_x)/(m-1);
std_x = sqrt(vecdiag(sigma));
d = diag(std_x);
std_x = repeat(std_x`,m,1);
xstd = j(n,1,1)||(x-one*mean_x)/std_x;

omega_ridge = omega[1:nrow(omega),1:ncol(omega)];
lamda = &lamda.;

beta = inv(xstd`*xstd + lamda#omega_ridge)*xstd`*y;

/*print beta;*/
f_sf = xstd*beta;
fitted_SF= f_sf||xdat[1:nrow(xdat),l];
create fitted_SF from fitted_SF [ colname = {"f_sf" "x"}]; append from
fitted_SF;

print f_sf;
```

```
call svd(U,Q,V,xstd);

S = inv( I(nrow(xstd)) +
lamda#(u`*inv(diag(Q))*v`*omega_ridge*V*inv(diag(Q))*U));

f = S*y[1:nrow(y),];
fitted= f||xdat[1:nrow(xdat),l];
create fitted from fitted [ colname = {"f" "x"}]; append from fitted;

/*BL 155*/
df = trace(S);
eigenval_s = eigval(S);
eigenvec_s = EIGVEC(S);
/*print df eigenval_s eigenvEC_s;*/

/*cv bl 161*/
do j = 1 to ncol(s);
CV_temp = CV_temp||((y[j,] - f[j,])/(1-s[j,j]))##2 ;
end;
CV = CV_temp[,+]/ncol(CV_temp);


/* gcv - bl 244*/

do j = 1 to ncol(s);
GCV_temp = GCV_temp||((y[j,] - f[j,])/(1-(trace(s)/ncol(s))))##2 ;
end;
GCV = GCV_temp[,+]/ncol(GCV_temp);
crosval= cv||gcv||df||lamda;
create crosval from crosval [ colname = {"cv" "gcv" "df" "lamda"}];
append from crosval;

var= inv(x`*x);
ci_x = 2#((vecdiag(x*var*x`))##(1/2));
pointwise_variance_css = vecdiag(x*var*x`);
print pointwise_variance_css ci_x;

fitted= f_sf||xdat[1:nrow(xdat),l]||pointwise_variance_css||ci_x;;
create fitted_ci from fitted [ colname = {"f" "x" "pv" "ci"}]; append
from fitted;

quit;

/*proc gplot data = fitted;*/
/*plot f*x;*/
/*run;*/

/*proc gplot data = fitted_SF;*/
/*plot f_sf*x;*/
/*run;*/
```

```
/*create confidance intervals*/

%mend;



%macro loop_ncs_lamda_omega();

data all_crosval;
run;

data L_val;
do L =  to ;
output;
end;
run;

proc sql;
     select count(L)
          into :num
     from L_val;
quit;
%DO k = 1 %TO &num.;

     data L_val_&k.;
          set  L_val;
          if _N_ = &k.;
     run;

     proc sql;
          select L
               into :lamda_value
          from L_val_&k.;
     quit;

  proc datasets nolist nowarn nodetails;
     delete L_val_&k.;
     run;

%ncs_lamda_omega(lamda=&lamda_value.);

data  all_crosval;
set all_crosval crosval;
run;

%end;

%mend;
%loop_ncs_lamda_omega();
```

```
proc gplot data = all_crosval;
plot cv*df;
run;

/*hier*/

proc sort data =  all_crosval;
by df;
run;

data all_crosval_24;
set all_crosval;
run ;

data mornej.all_crosval_all;
set all_crosval_24 all_crosval_23 all_crosval_22 all_crosval_21
all_crosval_20 all_crosval_19 all_crosval_18 all_crosval_17
all_crosval_16 all_crosval_15 all_crosval_14 all_crosval_13
all_crosval_12 all_crosval_11 all_crosval_10 all_crosval_9
all_crosval_8 all_crosval_7 all_crosval_6 all_crosval_5 all_crosval_4
all_crosval_3  all_crosval_2 all_crosval_1 ;
run ;
 /*Data Source:  EXAMPLES.ALL_CROSVAL_COS_NCS */


proc gplot data = EXAMPLES.ALL_CROSVAL_COS_NCS;
plot cv*df;
run;
```