Index-Model Inference with Box-Cox Transformation

Hai Yen Luu

Facultat de Matemàtiques i Estadística
Universitat Politècnica de Catalunya

Supervisor:
Catalina Bolancé Losilla

A thesis submitted for the degree of
Master's Degree in Statistics and Operations Research

Barcelona, June 2023

Acknowledgements

I would like to thank my thesis supervisor Catalina Bolancé Losilla for all the help and guidance I received from her during the period I worked at this master's thesis. Her valuable insights, and constructive feedback have been a precious instrument that helped me shape this work and achieve my academic goals. I am also thankful to the other professors of this master's degree, who provided an advanced knowledge in the area of Statistics and Operations Research. I would like to extend my heartfelt gratitude to my husband Roberto, for his love, patience, and support throughout this journey. His constant encouragement, understanding, and sacrifice have been invaluable in helping me balance my academic and personal life. I would like to acknowledge the unwavering support of my family, whose encouragement and love have been a constant source of strength throughout the entire process. I am forever grateful for their presence in my life.

Abstract

The single-index model is a non-parametric model that is popular in statistics for the estimation of risk assessment, thanks to its flexibility. Many studies have approached the problem of finding the best estimators for the model parameters and smooth function. However, most of the real-world data is asymmetric. In this thesis, we consider the application of a Box-Cox transformation on the response variable to improve the performances of the single-index model. Indeed, the Box-Cox transformation addresses the skewness of the response and transform it into a normally-distributed variables. We consider also the log transformation of the response variable. We show through a simulation study that the parameters of the models fitted with the transformed response have more similarity with the true parameters, which shows that the model is more reliable. Finally, we perform a study on a real-world data set from the car insurance industry, where we build a single-index model to predict the expected cost per claim based on a set of variables, that concern the demographic characteristic and the driving style of the customer. We assess the performance of the model fitted on the original data as well the models that use the log and the Box-Cox transformation, that obtain better results than the original one.

Keywords single index model, Box-Cox transformation, car insurance

Contents

List of Figures

Chapter 1

Introduction

The proper assessment of the risk profile of customers is of paramount importance in the insurance market. Companies try to predict the likelihood that a customer will generate a certain cost in terms of claimed accident damages based on his/her history, his/her demographic characteristics and his/her relevant habits. This applies to various kinds of insurance industries, as well to banks when assessing the risk of insolvency of potential customers before conceding or denying a loan. For example, in the case of car insurance, several driving habits such as the average speed, respect for speed limits, adherence to traffic rules, and other factors like distance traveled and driving experience can significantly influence the likelihood of causing an accident and its severity. This value will influence the fee that the customers will pay to be insured. An insurance company runs a profitable business when the sum of the collected fees is higher than the paid claims. According to a report from Insurance Europe, the claims paid by insurance companies in Europe in 2020 amounted to a value of 2.8 bn€, daily, which sums up to over 1000 bn€ yearly. The total premiums paid by customers were about 200 billions higher. A full picture can be seen in Figure 1.1. The difference between what the customers pay and the claims that companies has to pay is needed to pay salaries, to keep offices working, and for other relevant spending and investments that companies have to sustain to operate, and, of course, to have a profit. However, a company can't just indiscriminately increase the fees in order to be sure that the cost of paid claims won't exceed the available incoming cash flow, as this will have the side effect that customers will flee the company to subscribe more convenient contracts with the competitors. Therefore, there is great attention on the research of models that are able to predict as precisely as possible the likelihood that a customer will cause an accident, how many accidents customers will cause, and the monetary expenses that these accidents generate. This is a typical problem that can be performed through the theory and tools of statistics. In particular, semi-parametric single-index models are a class of regression models that have received major attention in the context of risk quantification in insurances.

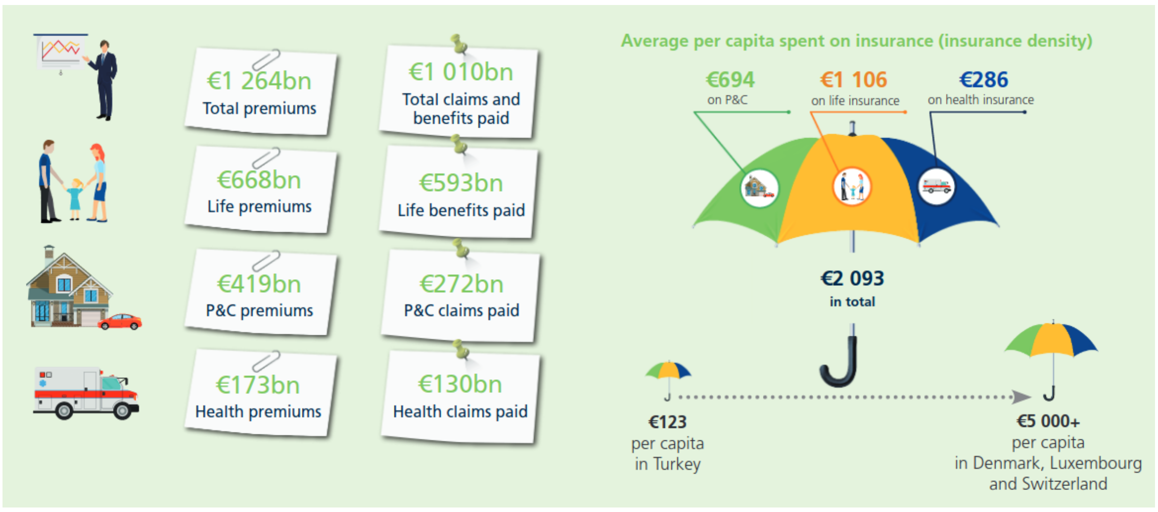

Figure 1.1: Statistics on the insurance market in Europe in 2020 (courtesy of Insurance Europe)

Differently from fully-parametric linear regression models, they include a non-parametric smoothing (or link) function that deals with the non-linearity of the model. Single-index model take this name because they predict the value of the dependent variable based on the application of the smoothing function to a single index. They have been extensively employed also in other sectors such as quantitative finance, portfolio management and asset pricing, where they are employed, for example, to build profitable stock portfolios that minimize the expected risk and maximize the expected returns.

In the context of this thesis, we are interested in the design of a single-index model with multiple predictors, with the final goal of being able to predict the expected cost per claim that customers are expected to generate. In particular, we consider the work of Bolancé et al. (2018), that is one of the first works to consider the prediction of cost per claim instead of the total cost per year. Analogously to Bolancé et al., we use a flexible maximum likelihood method to estimate the linear coefficients and a leave-one-out kernel to estimate the link function. However, given that skewness or non-normality of the response variable can cause a loss of the predictive performance of the single-index model, we extend their work with the application of a two-parameters Box-Cox transformation on the response variable, so to transform it into a normal-distributed variable. We compare the predictive performances of the model obtained with the transformed response with the model with the original response. We compare the model that has been transformed with the Box-Cox transformation also with a log-transformation of the response variable. Finally, we apply this methodology on a real-world case study from the car insurance industry, with the aim to predict the accident severity. To do so, we employ a rich anonymized data set provided by a Spanish company, that includes demographic data as well as data about the driving habits of the customers, that is called telematics data, given that has been collected by the insurance company through the black-box device that customers agree to have installed in their cars.

The remainder of this Thesis is organized as follows: Chapter 2 gives a general overview of the Theory related with the topic, provide some examples and presents the related literature. Chapter 3 gives the details of the Box-Cox transformation application in the single-index model. Experimental data and results, with a rich set of plots and graphics, are presented in Chapter 4. The conclusions and a suggestion for future work are given in 5. Finally, all the implementation details are situated in the Appendix A, where the most relevant `R` functions and code lines are reported.

Chapter 2

Theoretical Background and Related Work

In statistics, a regression model is a basic way to describes the relationship between one or more explained variables and a response variable. However, it is not always easy to implement to get a model which fit well for all observations, and unveil the right relationship between them. Single index model is a special model which combine the regression linear with a smooth function. Hence, it is more flexible to illustrate the relationship between the dependent variable and independent variables. Besides, the Box-Cox transformation can be useful in helping to achieve this goal by transforming the data into a more appropriate form for analysis.

2.1 Single-index model

A general linear regression model takes the form

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{2.1}$$

where $\mathbf{y}$ is the vector of observations values, $\mathbf{X}$ is the matrix of regressors (or explanatory variables, covariates, predictors, or independent variables), $\boldsymbol{\beta}$ is the parameter vector and $\boldsymbol{\varepsilon}$ is the error vector. The model presented in Equation 2.1 written in its extended form is as follows:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i \tag{2.2}$$

where index i refers to the observations and $1 \ldots p$ are used to distinguish the covariates and their parameter coefficients. The semi-parametric single-index model is defined as follows:

$$\mathbf{y} = \mathbf{f}(\mathbf{X}\boldsymbol{\theta}) + \boldsymbol{\varepsilon} \tag{2.3}$$

where $\mathbf{X}$ is the vector of explanatory variables, f is an unknown smooth function (non-parametric component), $\boldsymbol{\theta}$, the equivalent of $\boldsymbol{\beta}$ in simple linear regression, is the vector of parameters, also known as index, and ε is the vector of random variables with zero-mean conditional on X. So, the regression consists of a non-parametric function of the linear index θx from which the term index model arises. If we write Equation 2.3 making explicit all the components of vector x, we can have it in an extended form:

$$y_i = f(\theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \cdots + \theta_p x_{ip}) + \epsilon_i \tag{2.4}$$

The reader can appreciate by the comparison of Equations 2.1 and 2.3, or their extended forms, 2.2 and 2.4, respectively, that the only difference lies in the presence of the smooth function f, which makes non completely parametric the new model. The expected value of y is given by

$$E(y_i|x_i) = f(\theta' x_i) \tag{2.5}$$

An advantage of the single index model lies in the fact that it combines both parametric and non-parametric components, therefore it allows for flexibility in modeling the relationship between the response variable and predictor variables. This comes at the price of an increased complexity compared to a linear regression model, which can be fitted by means of a simple method like the least square errors, that consists in determining the values of the parameters $\beta_0 \ldots \beta_p$ that minimize the sum of the squares of the deviations between the expected values $\hat{y}_i$ and the actual values y_i. The single index model requires a much more complex estimation that provides both values for the linear predictor coefficients $\theta_0 \ldots \theta_p$ and the smooth function f. To this regard, a large variety of methods has been proposed in the literature, including kernel smoothing, spline smoothing, and penalized likelihood methods.

Starting from the origin, the single-index model was introduced by Sharpe (1963) and its early applications are found in the field of quantitative finances, however the possibilities of applying single-index models are broad. We mention, among many, portfolio optimization [Sharpe 1963, Elton and Gruber 1977], risk analysis and insurance in farm planning, [Collins and Barry 1986, Miranda and Glauber 1997], risk analysis in car insurance [Bolancé et al. 2018], and signal processing [Pananjady and Foster 2021]. In order to estimate the coefficient vector θ, Han (1987) and Sherman (1993) introduced estimators based on maximum rank correlation. Ichimura (1993) proposed the semi-parametric least squares and the weighted semi-parametric least squares estimation. In the same year, Klein and Spady (1993) introduced a quasi-maximum likelihood estimator for the case in which the response is a binary variable $y \in \{0, 1\}$, and Hardle et al. (1993) proposed an estimator based on the minimization of the residual sum of squares. Later, Horowitz and

Härdle (1996) proposed a non-iterative estimation method valid for the case of discrete independent variables. More recently, Delecroix et al. (2003) use the pseudo-maximum likelihood method to estimate the model with one bandwidth of kernel estimation, and Strzalkowska-Kominiak and Cao (2013) studied alternatives to the maximum likelihood estimation, based on a non-parametric estimation of the conditional distribution, for censored data. Finally, Bolancé et al. (2018) applied the flexible maximum likelihood method to estimate the parameter θ in single-index models. To estimate the link function f, which usually is not possible to known in its analytical form, most papers use a kernel density estimation approach, that is commonly used for analyzing and visualizing data when the underlying distribution is not known or is difficult to determine. Kernel Density Estimation is based on a bandwidth parameter, that determines the width of the kernel. The larger is the bandwidth, the more the data points are aggregated in the same kernel point. Smaller bandwidth make the estimation more precise but also more sensible to variations between individual points. For a better result, multiple bandwidths can be applied. We present, hereafter, the theoretical details of some of the discoveries from the papers mentioned above, especially for what regards the estimation of the parameters of the single-index model.

Remind that $x_1, x_2, \ldots, x_n$ represent the observed values of a random sequence of p-vectors, $X_1, X_2, \ldots, X_n$ and $\epsilon_1, \ldots, \epsilon_1$ are independent random variables with zero mean and bounded variance. The function f is the conditional density of Y given X, and θ is a p-variate unit vector. Hardle et al. proposed a way to estimate the vector θ in which they minimize the residual sum of squares, so that θ is estimated according to the function:

$$\hat{\theta} = argmin_{\theta}\left(\sum_{i=1}^{n}[Y_i - \hat{f}_i(\theta' X_i)]^2\right) \tag{2.6}$$

Beside estimating θ, they find the function of the density distribution of Y by means of a leave-one-out kernel estimator with one bandwidth. Let $u = \theta X$, where X is one of the observations x_i. The leave-one-out kernel density estimator when the pair (X_i, Y_i) is omitted is calculated as:

$$f_i(\hat{u}|\theta) = \frac{\sum_{j\neq i}^{n} Y_j K_h(u - \theta' x_j)}{\sum_{j=1}^{n} K_h(u - \theta' x_j)} \tag{2.7}$$

where h is the bandwidth, and K is a fixed kernel function.

Klein and Spady (1993) proposed an estimator of the semi-parametric single-index model by maximum likelihood estimation of θ, according to Equation 2.8:

$$L_n(\beta) = \sum_{i=1}^{n}\left[Y_i\, ln(\hat{f}(\theta' X_i) + (1 - Y_i)\, ln(1 - \hat{f}_i(\theta' X_i))\right] \tag{2.8}$$

where $\hat{f}_i$ indicates the leave-one-out kernel estimator of f.

We focus now on the contribution of Bolancé et al. (2018), which is at the basis of the work contained in this thesis. They applied the flexible maximum likelihood function $\tilde{L}$ to estimate the parameter θ, as following:

$$\tilde{L}_n(\theta) = \prod_{i=1}^{n} f_\theta(Y_i|\theta' X_i) \tag{2.9}$$

Furthermore, they denote that maximizing $\tilde{L}$ is equivalent to maximizing its logarithm. They define the function $\tilde{l}$:

$$\tilde{l}_n(\theta) = \frac{1}{n} log(\tilde{L}_n(\theta)) = \frac{1}{n}\sum_{i=1}^{n} log f_\theta(Y_i|\theta' X_i) \tag{2.10}$$

and they state that the ideal estimator $\tilde{\theta}$ would be:

$$\tilde{\theta}_n = \underset{\theta}{argmax}\, \tilde{l}_n(\theta) = \underset{\theta}{argmax}\, \frac{1}{n}\sum_{i=1}^{n} log f_\theta(Y_i)|\theta' X_i) \tag{2.11}$$

To estimate $f_\theta(Y_i|\theta' X_i)$, they use the kernel estimator with two bandwidths with a leave-one-out procedure, hence the $i - th$ component will be left out when we estimate the estimator of set (Y_i, X_i). The first bandwidth constructs a preliminary estimator of f, which estimate θ, and the second for a final estimator of f. The kernel conditional density, following Hall et al. (1999), is calculated as follows:

$$f_\theta^{-i}(Y_i)|\theta' X_i) = \frac{\hat{r}_{-i}(\theta' X_i, Y_i)}{\hat{s}_{-i}(\theta' X_i)} \tag{2.12}$$

in which $\hat{r}_{-i}$ is defined as:

$$\hat{r}_{-i}\left(\theta' X_i, Y_i\right) = \frac{1}{h_1 h_2} \sum_{j=1, i\neq j}^{n} K\left(\frac{\theta' X_i - \theta' X_j}{h_1}\right) K\left(\frac{Y_i - Y_j}{h_2}\right) \tag{2.13}$$

and $\hat{s}_{-i}$ is:

$$\hat{s}_{-i}(\theta' X_i, Y_i) = \frac{1}{h_1} \sum_{j=1, i\neq j}^{n} K\left(\frac{\theta' X_i - \theta' X_j}{h_1}\right) \tag{2.14}$$

In particular, h_1 and h_2 are two positive bandwidths. At this point, the leave-one-out conditional likelihood is:

$$\hat{l}_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} log \hat{f}_{\theta_i}(Y_i|\theta' X_i) \tag{2.15}$$

and the final maximum conditional likelihood estimator can be calculated as:

$$\hat{\theta}_n = \underset{\theta}{argmax}\, \hat{l}_n(\theta) \tag{2.16}$$

Alternatives to single-index model are non-parametric regression [Härdle 1990], which is more general, and the generalised additive model [Hastie 2017]. However, as Bolancé et al. (2018) point out, both presents considerable drawbacks, such as the curse of dimensionality (when the number of independent variable increases, the estimation becomes more difficult), and the relationship between covariates and response is not easily explainable. Thus, we consider the adoption of the single-index model to be the best option for our problem.

2.2 The Box-Cox transformation

The Box-Cox transformation is used to cope with anomalies such as non-additivity, non-normality and heteroscedasticity, by transforming the response so that it takes a normal distribution. The method was presented for the first time in the article "An analysis of transformations" by Box and Cox (1964). It presents a family of power transformations that incorporate and extend the traditional options to help researchers to easily find the optimal normalizing transformation for each variable.

In general, the data, and in particular the response variable, can be transformed by the following Box-Cox function with one or two parameters. The one parameter Box-Cox transformation is given by:

$$y^{\lambda} = \begin{cases} \frac{y^{\lambda}-1}{\lambda} & \text{if } \lambda \neq 0, y > 0 \\ \log(y) & \text{if } \lambda = 0, y > 0 \end{cases} . \tag{2.17}$$

while the two parameters Box-Cox transformation is:

$$y^{\lambda} = \begin{cases} \frac{(y+\lambda_2)^{\lambda_1}-1}{\lambda_1} & \text{if } \lambda_1 \neq 0, y > -\lambda_2 \\ \ln(y+\lambda_2) & \text{if } \lambda_1 = 0, y > -\lambda_2 \end{cases} . \tag{2.18}$$

where y is the original response, y^{λ} is the transformed response, and λ is the power parameter. The value of λ can be estimated using maximum likelihood estimation, which involves selecting the value of lambda that maximizes the likelihood of the transformed data being normally distributed. When $\lambda = 0$, the transformation becomes log transformation. Hence, the transformations are defined:

$$T(Y) = \left[\begin{array}{rl} log(y) & \text{if } y > 0 \\ log(y+\lambda_2) & \text{if } y > -\lambda_2 \end{array} \right. . \tag{2.19}$$

To the best of our knowledge, we did not find any work in the literature that apply the Box-Cox transformation to improve the performances of the single-index model. In any case, we present hereafter an overview of some papers that apply the Box-Cox transformation to deal with skewness of responses and to improve predictive performances of models on a wide set of problems. Relevant examples are visualization and image processing, as in Maciejewski et al. (2012) and Cheddad (2020), improvement in time series forecasting accuracy, for which we cite Proietti and Lütkepohl (2013), Lee et al. (2013), Terasaka and Hosoya (2007), financial modelling, as in Pajor (2009), and quality control and operations management, like in Wu et al. (2014), Sennaroğlu and Şenvar (2015), Bicego and Baldo (2016). These pieces of work demonstrated the effectiveness of the Box-Cox transformation on a variety of domains. Furthermore, they are all from recent researches developed in the last two decades, which shows that that the interest in the application of the technique remains high among the research community. However, there are also some articles that show the limits of the Box-Cox transformation and propose solutions to fix them. For example, Zhang and Yang (2017) mentioned some challenges related to memory and storage capacity, as well as processing time in big data, and proposed ways to handle with this. Moreover, there are many papers that modify the Box-Cox function to adapt it also for negative values, for example Hawkins and Weisberg (2017), Weisberg (2001), and John and Draper (1980). Finally, there are also articles that show that the transformation is not very useful in specific contexts, because it is not possible to find a value of λ that produces a normally distributed data. This was the case of Nelson Jr and Granger (1979).

Chapter 3

Application of Box-Cox to the single index model

From the study of Bolancé et al. (2018), we observe that the skewness of the data has a significant impact on the accuracy of the results. In fact, the outcomes for the non-normal distribution data are worse than the ones for normally distributed data. To re-examine, we generate the data sets in different distributions of 100 samples with size n=100 and n=500, then run again the algorithm following the method of Bolancé et al. (2018), we get the results which showed in the table below.

		Normal	Logistic	Lognormal	Weibull	Champernowne ($\alpha = 1$)	Champernowne ($\alpha = 2$)
n=100							
Bias	θ_2	-0.0022	-0.1682	0.1697	-2.7905	-2.1894	0.5369
	θ_3	0.0116	0.1345	0.2367	1.1376	2.4972	-1.9249
MSE	θ_2	0.0389	1.1306	1.8237	508.3906	198.4770	1810.825
	θ_3	0.0192	6.6961	5.4549	153.7398	191.0748	3600.027
n=500							
Bias	θ_2	0,0052	0,0099	-0,0153	-1,0060	-3,7711	-0,2375
	θ_3	0,0013	-0,0001	-0,0115	-1,0420	-2,2358	-0,0639
MSE	θ_2	0,0050	0,0077	0,0313	10,1076	181,6413	0,7377
	θ_3	0,0029	0,0045	0,0206	21,6555	42,8694	0,5732

Table 3.1: The simulated results by 100 samples with size = 100 and size = 500

Following the outcomes, the results are improved when the size of the sample are greater. Especially, the values of bias and mean squared error of estimation with the data which is in normal distribution shape (normal distribution, and logistic distribution) is very smaller than the ones whose distribution is asymmetrical (log-normal, Weibull, Champernowne distribution). Therefore, our research question is whether we can obtain a better estimation if we transform the response from non-normal distribution to normal distribution. Hence, to explore this possibility, one of the most popular methods is the Box-Cox transformation. This method involves applying a power transformation to the data in order to achieve a normal distribution.

3.1 Description of the application of Box-Cox to single index model

Based on the problem mentioned above, our idea is transform the data from skewed to symmetric, then we can estimate the link function and the parameters. We apply the Box-Cox transformation to change the dependent data from its skewness state to the normal distribution, so that the predictive qualities of the model can be enhanced. First of all, based on the available data, we find the optimal parameters λ, then use these parameters to calculate the new values. The distribution of the set of value is expected to be normality.

In general, when u is a variable which depends on v, or $u = h(v)$, so that $v = h^{-1}(u)$ where h^{-1} is the inverse function of the function h, the density function of u can be calculated according to the following relation:

$$f(u) = f(h^{-1}(u))\frac{dv}{du} \tag{3.1}$$

Let $T(Y)$ be the Box-Cox transformation function of the dependent variable Y, let $g(T(Y))$ be the density function of $T(Y)$, and let $T'(y)$ be the derivative of $T(Y)$. We have:

$$f(y|\theta x) = g(T(y)|\theta x) \cdot T'(y) \tag{3.2}$$

From the Equation 3.2, instead of estimating directly $f(y|\theta x)$, we evaluate $g(T(y)|\theta x)$. Recalling Equation 2.12, we can calculate also the leave-one-out conditional density of $T(Y_i)$, given X_i, $g_\theta^{-i}(T(Y_i)|\theta' X_i)$, as:

$$\hat{g}_\theta^{-i}(T(Y_i)|\theta' X_i) = \frac{\hat{r}^{-i}(\theta X_i, T(Y_i))}{\hat{s}^{-i}(\theta X_i)} \tag{3.3}$$

Then, we have that:

$$\hat{f}_\theta^{-i}(T(Y_i)|\theta' X_i) = \hat{g}_\theta^{-i}(T(Y_i)|\theta' X_i) \cdot T'(Y_i) = \frac{\hat{r}^{-i}(\theta X_i, T(Y_i))}{\hat{s}^{-i}(\theta X_i)} \cdot T'(Y_i) \tag{3.4}$$

where, according to Equations 2.13 and 2.14, $\hat{r}^{-i}$ is defined as

$$\hat{r}^{-i}(\theta' X_i, T(Y_i)) = \frac{1}{h_1 h_2} \sum_{j=1, i\neq j}^{n} K\left(\frac{\theta' X_i - \theta' X_j}{h_1}\right) K\left(\frac{T(Y_i) - T(Y_j)}{h_2}\right) \tag{3.5}$$

and $\hat{s}^{-i}$ is computed as:

$$\hat{s}^{-i}(\theta' X_i, T(Y_i)) = \frac{1}{h_1} \sum_{j=1, i\neq j}^{n} K\left(\frac{\theta' X_i - \theta' X_j}{h_1}\right) \tag{3.6}$$

The leave-one-out conditional likelihood function is defined as:

$$\hat{L}_n(\theta) = \prod_{i=1}^{n} f_\theta^{-i}(Y_i|\theta' X_i) = \prod_{i=1}^{n} g^{-i}(T(Y_i)|\theta X_i) \cdot T'(Y_i) \tag{3.7}$$

To estimate the parameters, we find the values of θ that maximize the value of the log function $\hat{l}_n$, defined as:

$$\hat{l}_n(\theta) = log(\hat{L}_n(\theta)) = \sum_{i=1}^{n} log(g^{-i}(T(Y_i)|\theta X_i)) + \sum_{i=1}^{n} log(T'(Y_i)) \tag{3.8}$$

Since the transformed response $T'(Y_i)$ is not dependent on θ, the derivative of $log(T'(Y))$ by θ is equal to zero, therefore, the estimator is not depend on $T'(Y)$. This is translated into the following equation:

$$\hat{\theta}_n = \underset{\theta}{\operatorname{argmax}}\ \hat{l}_n(\theta) = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} log(g^{-i}(T(Y_i)|\theta X_i)) \tag{3.9}$$

which applies when:

$$\frac{\partial \hat{l}_n(\theta)}{\partial \theta} = \frac{\partial \sum_{i=1}^{n} log(g^{-i}(T(Y_i)|\theta X_i))}{\partial \theta} = 0 \tag{3.10}$$

3.2 Simulation study

In order to verify our assumption on the properties of the estimator, we ran the data set with 100 samples of size n=500. Specifically, we used the `R` programming language and a variety of statistical packages, including `EnvStats`, `stats`, and `ggplot2`, to perform the analysis. Hereafter, we define as θ_0 the initial parameters, and θ the estimated parameters. The algorithm is constructed to create the random values for the independent variables x and for the response y, which is obtained by the application of a set of different smooth functions, namely lognormal, Weibull, Champernowne, Pareto, and exponential distribution, to the linear combination $\theta_0 x$. For $\theta_0 = \{\theta_{01}\theta_{02}, \theta_{03}\}$, we fix the values $\{1, 1.3, 0.5\}$. After that, we fit a single-index model by estimating the parameters $theta$ and the smooth function f. To verify the efficiency of the method, we calculate the bias and the mean squared error of the estimated θ and the real θ_0.

n=500			Lognormal distribution Orig.	Log	Box-Cox	Weibull distribution Orig.	Log	Box-Cox
Bias	V1	θ_2	-0.0153	0.0052	0.0018	-1.0061	-0.0250	-0.0043
		θ_3	-0.0116	0.0013	-0.0016	-1.0420	-0.0105	-0.0145
MSE	V1	θ_2	0.0206	0.0050	0.0033	10.1076	0.0546	0.0992
		θ_3	0.0206	0.0029	0.0017	21.6555	0.0232	0.0178
n=500			Champernowne ($\alpha = 1$) Orig.	Log	Box-Cox	Champernowne ($\alpha = 2$) Orig.	Log	Box-Cox
Bias	V1	θ_2	-3.7712	0.0160	0.0134	-0.2375	0.0009	0.0022
		θ_3	-2.2358	0.0052	-0.0009	-0.0639	0.0004	-0.0024
MSE	V1	θ_2	181.6414	0.0179	0.0156	0.7377	0.0020	0.0023
		θ_3	42.8694	0.0076	0.0072	0.5732	0.0009	0.0012
n=500			Pareto distribution Orig.	Log	Box-Cox	Exponential distribution Orig.	Log	Box-Cox
Bias	V1	θ_2	-1.6966	0.0054	0.0034	-0.2845	-0.0133	-0.0066
		θ_3	-1.0196	0.0001	-0.0012	-0.4875	-0.004	0.0029
MSE	V1	θ_2	60.8870	0.0016	0.0013	5.7300	0.0051	0.0042
		θ_3	13.8100	0.0006	0.0004	5.9300	0.0033	0.0029

Table 3.2: Bias and MSE of the estimators $\hat{\theta}$ in different distributions

The results are shown in the Table 3.2. The table is composed by six boxes, each of them representing the results obtained by the different smoothing function (Lognormal, Weibull, Champernowne with $\alpha = 1$ and $\alpha = 2$, Pareto and Exponential). Inside each box, the columns repreent the results obtained for the original data (Orig.), the data transformed with the logarithm in columns Log(Y) and the data transformed with Box-Cox in the columns Box-Cox(Y). The rows contains the values of the Bias and the Mean Square Error (MSA) of the estimators, for the parameters θ_2 and θ_3. From the results, we can observe there is a big difference between the results of the non-transformed data and the transformed data. The values of the bias and the MSE obtained when we use the transformations are much lower than those we get from the original data, which indicate a much better fit of the model. Moreover, the Box-Cox transformation gives results that are slightly better than the ones obtained by the log transformation for most distributions.
After we get the parameters corresponding to each sample, we use the statistical power to verify the effect of the variance, in particular whether it is different from zero and whether the estimated parameters are identical to the initial parameters θ_0. The first null hypothesis is: H_0: $\theta_{jk} = 0$, where $k = 1, \ldots, d$, with d that is the number of the independent variables, $j = 1, 2, \ldots, n$, with n that is the number of replications or the number of samples, and the

alternative hypothesis is H_1: $\theta_{jk} > 0$. In order to find the p-value that we need to answer the hypothesis test, we calculate the critical Z-value $Z = \frac{\hat{\theta_j}}{se(\hat{\theta_j})}$, where $\hat{\theta_j}$ is the estimated parameter of each sample j and $se(\hat{\theta_j})$ indicates the estimated standard error. After having the p-value for the hypothesis testing for each sample, we determine how many times we reject the null hypothesis for all samples. In the table, we record the outcomes of estimated power, that is the percentage of times in which we reject the null hypothesis or when $p - value \leq 0.025$. According to the results, the proportion of rejection is higher when we transform the data by log and Box-Cox transformation, with the average p-value very near to 0. Hence the estimated parameters are significantly different from zero, and the percentage is higher with the transformed data.

On the other hand, we want to test if the estimated parameters are considered to be the same as the initial parameters. To do so, we take as as null hypothesis H_0: $\theta_{jk} = \theta_{0k}$ and the alternative hypothesis: $\theta_{jk} \neq \theta_{0k}$. We test to verify if the estimated θ_{jk} is equal to the θ_{0k} and to know how many samples get the $p - value > 0.025$.
The statistic for this test is $Z = \frac{\hat{\theta_{jk}} - \theta_{0k}}{se(\hat{\theta_{jk}} - \theta_{0k})}$
We verify that the average p-value of all samples in all distributions are greater than 0.1. The estimated power in the table is the percentage of samples in which the null hypothesis is not rejected. It shows an improvement of the proportion of acceptance of the null hypothesis with the transformation. However, the results do not change much in the case of the exponential distribution. The results are presented in Table 3.3.

Lognormal distribution						
	Orig.		Log		Box-Cox	
	est. power	avg. p-value	est. power	avg. p-value	est. power	avg. p-value
θ_2=0.0	1.000	0.000	1.000	0.000	1.000	0.000
θ_3=0.0	0.920	0.013	1.000	0.000	1.000	0.000
θ_2=1.3	0.880	0.249	0.910	0.224	0.880	0.223
θ_3=0.5	0.850	0.172	0.830	0.218	0.890	0.211
Weibull distribution						
	Orig.		Log		Box-Cox	
	est. power	avg. p-value	est. power	avg. p-value	est. power	avg. p-value
θ_2=0.0	0.600	0.078	0.990	0.002	1.000	0.000
θ_3=0.0	0.460	0.109	0.960	0.003	0.960	0.005
θ_2=1.3	0.430	0.120	0.840	0.194	0.840	0.181
θ_3=0.5	0.560	0.119	0.820	0.200	0.860	0.201
Champernowne ($\alpha = 1$) Distribution						
	Orig.		Log		Box-Cox	
	est. power	avg. p-value	est. power	avg. p-value	est. power	avg. p-value
θ_2=0.0	0.620	0.092	1.000	0.000	1.000	0.000
θ_3=0.0	0.450	0.112	0.990	0.0003	1.000	0.0002
θ_2=1.3	0.490	0.121	0.740	0.173	0.780	0.180
θ_3=0.5	0.600	0.127	0.750	0.192	0.790	0.207
Champernowne ($\alpha = 2$) Distribution						
	Orig.		Log		Box-Cox	
	est. power	avg. p-value	est. power	avg. p-value	est. power	avg. p-value
θ_2=0.0	0.940	0.011	1.000	0.000	1.000	0.000
θ_3=0.0	0.910	0.023	1.000	0.000	1.000	0.000
θ_2=1.3	0.780	0.200	0.850	0.189	0.800	0.180
θ_3=0.5	0.830	0.211	0.830	0.193	0.830	0.206
Pareto Distribution						
	Orig.		Log		Box-Cox	
	est. power	avg. p-value	est. power	avg. p-value	est. power	avg. p-value
θ_2=0.0	0.790	0.038	1.000	0.000	1.000	0.000
θ_3=0.0	0.660	0.075	1.000	0.000	1.000	0.000
θ_2=1.3	0.780	0.194	0.840	0.227	0.860	0.211
θ_3=0.5	0.850	0.193	0.850	0.216	0.900	0.220
Exponential Distribution						
	Orig.		Log		Box-Cox	
	est. power	avg. p-value	est. power	avg. p-value	est. power	avg. p-value
θ_2=0.0	0.880	0.022	1.000	0.000	1.000	0.000
θ_3=0.0	0.840	0.032	1.000	0.000	1.000	0.000
θ_2=1.3	0.800	0.214	0.850	0.213	0.840	0.210
θ_3=0.5	0.820	0.204	0.800	0.186	0.800	0.194

Table 3.3: The power statistics

Chapter 4

Predicting automobile accident costs through a single-index model with the Box-Cox transformation

For the experimental chapter of this thesis, we use data from a Spanish car insurance. In car insurances, car owners pay a yearly fee to a company in order to be insured against traffic-related accidents. In Europe, mandatory car insurance has to cover for the damages the customer may cause to properties of others, their health or their lives. The company gets paid a yearly fee by the customer, that is usually in the range of hundreds to thousands of euros, in exchange of the legal binding to repay damages in case of accidents, which can range up to millions of euros. In this case study, we try to build a single index model that predicts the cost per claim that a customer is likely to generate depending on his/her demographic data and his/her driving habits. Data is collected from the general profile of the customer (non-telematics data) and from its driving habits (telematics data), thanks to a black-box device that companies install on the cars of the customers. The model may reveal that certain categories of drivers generate higher claim costs compared to other categories who exhibit more cautious driving habits. By accurately predicting the cost per claim, insurance companies can better manage their financial risks and optimize pricing strategies, offering personalized insurance premiums based on individual risk profiles.

4.1 Data set

The data set is from a Spanish insurance company and contains a sample of 489 clients who hold a car insurance contract and submitted at least one claim in the analyzed period of one year. The given information for each policyholder is presented in Table 4.1. In the table, a horizontal line divide the response, which is is expressed in terms of total cost, number of claims and cost per claim generated by the customers, from the independent variables. The three ways in which the response is expressed are related, but in this analysis the objective is to being able to predict the expected cost per claim. The independent variables are divided in two categories: telematic covariates and non-telematic covariates. The telematic covariates are collected by the insurance company through a device installed in the cars of the customers, and are tkm, nightkm, urbankm, and speedkm, whilethe non-telematic covariates are gender, age, agelic, agecar, and parking.

Parameter	Meaning
total_cost	total cost for all claims in thousand of euros
Nclaims	the numbers of claims
cost	cost per claim in thousands of euros. To simplify the analysis, we divide the values by 1000
gender	the gender of clients
age	age of the clients in years
agelic	age of driving licence in years
agecar	age of the car in years
parking	is 1 if car is parked in the garage at night, otherwise the value is 0
tkm	annual kilometers driven in thousands. To simplify the analysis, we divide the values by 1000.
nightkm	kilometers driven at night. However, we transform this variables to be the percentage of night kilometers on the total annual kilometers in the analysis.
urbankm	percentage of kilometers on urban roads
speedkm	percentage of kilometers above the speed limit

Table 4.1: Information contained in the data set used for the case study

The descriptive statistics of the cost per claim and other variables are presented in Table 4.2. The statistics shown are the mean, the standard deviation and the quantiles. We can see that the maximum cost per claim is 130870,3 euros, but other critical statistics such as mean, minimum, median, the 25th and 75th percentiles are quite small, all under 2 thousand euros.

	Mean	STD	Min	Q25	Median	Q75	Max
cost	1.8097	6.1911	0.0177	0.4172	0.8182	1.8785	130.8703
age	27.0092	3.2457	20.5859	24.4955	26.8199	29.8864	34.0670
agelic	6.4280	2.8333	2.0014	4.3367	5.8645	7.9917	14.6858
agecar	8.9162	4.1620	2.1109	5.7768	7.9425	11.3703	20.4682
parking	0.7628	0.4258	0.0000	1.0000	1.0000	1.0000	1.0000
tkm	8.3564	4.5304	1.2198	5.1743	7.5493	10.6352	35.1049
nightkm	7.5138	6.5030	0.0438	2.9789	5.8408	9.9537	42.8302
urbankm	27.1275	14.1629	3.8099	16.5655	24.4008	35.2445	80.6586
nightkm	7.2033	7.0997	0.1223	2.2857	4.9691	9.4033	48.0024

Table 4.2: Descriptive statistics of the variables of the data

A preliminary exploratory data analysis step, that help us to understand visually the impact of each factor on the cost per claim, consists in generating a series of plots to verify the distribution of the outcome.

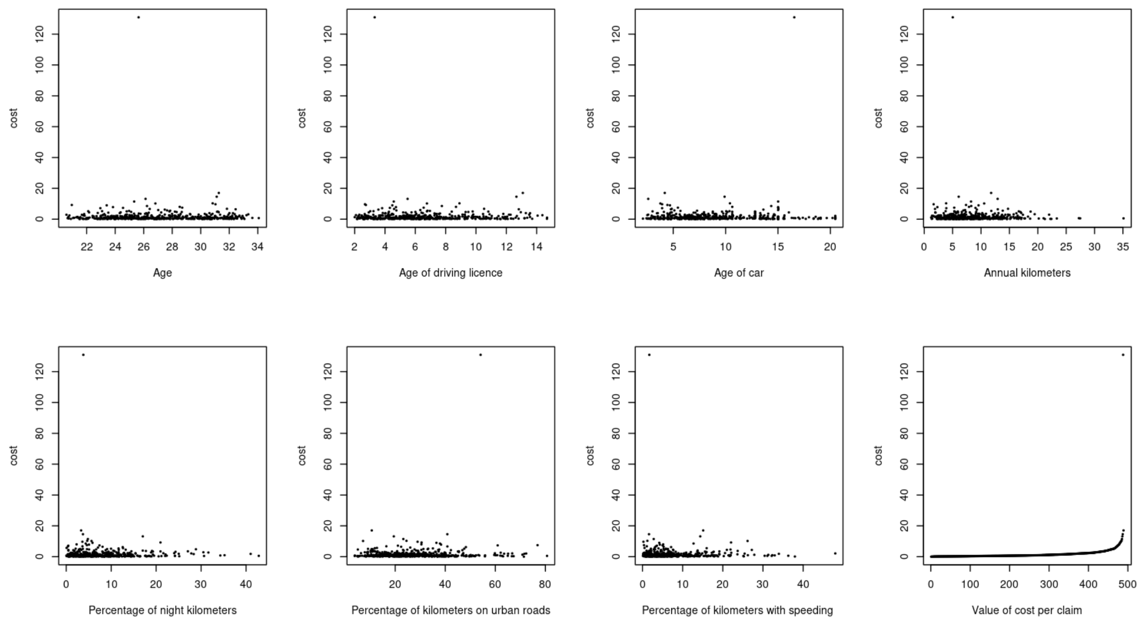

Figure 4.1: Dispersion plots of the relationship between cost per claim and independent variables

Figure 4.1 represents the distribution of the cost depending on the explanatory variables age, agelic, agecar, tkm, nightkm, urbankm and speedkm. We don't take into account the presence of parking, given that it is a binary variable takes only two values, zero and one. According to the plots in Figure 4.1, the pattern of association between the variables is not readily apparent. The last plot, at the bottom right, represents just the sorted responses in increasing order. We can notice that the shape is quite flat, except from the right tail and, in particular, the extreme value. The figures show the symptoms of non-normality in the response and the need for a transformation of the variable. In most of cases, cost per claim is under €30000, however, the extreme value with a single claim that costed 130000€. Probably, it is the case of a customer that was involved in a serious accident on the road which involved a much higher cost than average. We also noticed that this accident with a very high cost was also the only one generated by this customer.

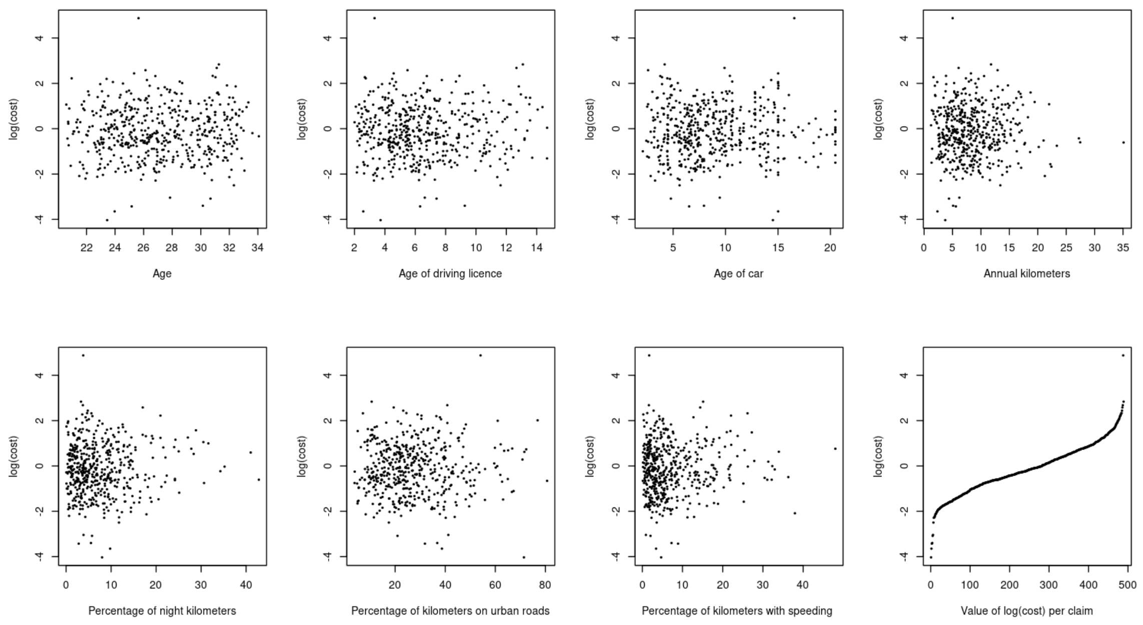

Figure 4.2: Dispersion plots of the relationship between the log-transformed of cost per claim and independent variables

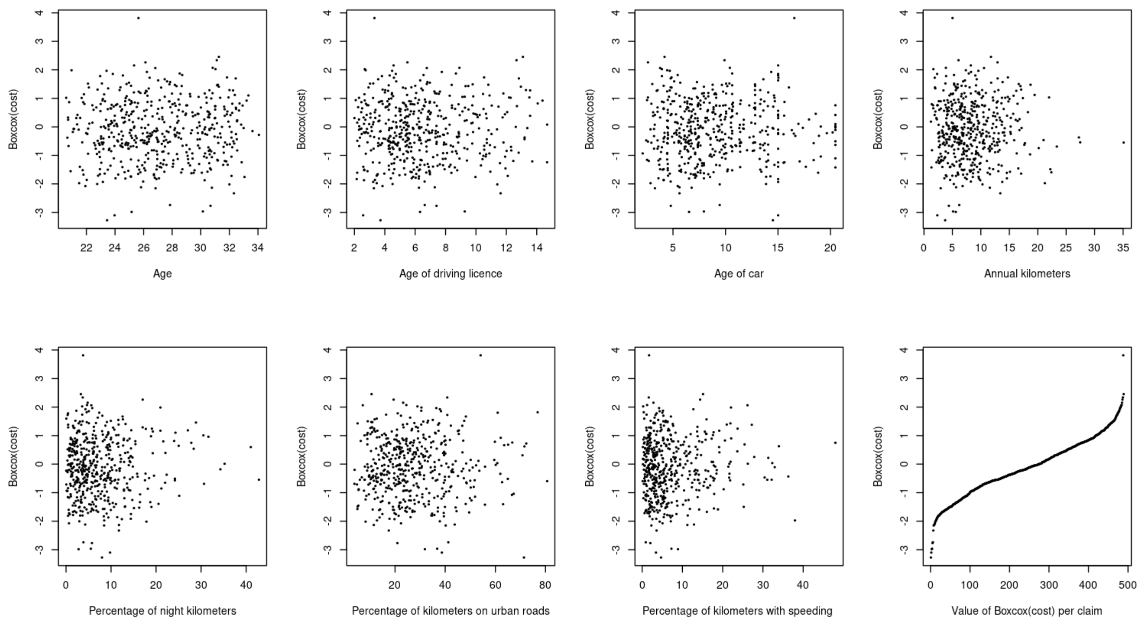

Figure 4.3: Dispersion plots of the relationship between Box-Cox transform of cost per claim and independent variables

According to the methodology presented in the previous sections, we transform the cost per claim by both the log transformation and by the Box-Cox transformation that we defined in Chapter 3. In particular, the application of the two-parameters Box-Cox transformation with λ_1 and λ_2 resulted in the optimal values of $\lambda_1 = -0.10506$ and $\lambda_2 = 0.04225$. We recall that the value of the cost transformed by log transformation is a special case of the Box-Cox transformation when the parameter λ_1 is equal to zero. In this case, $\lambda_1 \neq 0$, so it makes sense to apply the Box-Cox transformation. After we transform the cost by the log transformation and by the Box-Cox transformation, we obtain, respectively, the new sets of plots in Figure 4.2 and in Figure 4.3. From the new plots, we can observe that the mean of transformed cost per claim seems to remain constant for different values of the independent factors. With both transformations, the response takes a shape that is nearer to an increasing line. We can realize that both log and Box-Cox transformations produce plots that look like similar, however, the range of transformed cost by Box-Cox transformation is smaller than the one by log transformation. Additionally, by plotting the histogram for the cost and the transformed cost, we get Figure 4.4. Apparently, the cost distribution becomes normal with the log and the Box-Cox transformation.
To get a more statistically significant confirmation beyond the visual observations done so far, we also perform the Shapiro-Wilk normality test with a significance level of 0.05 to test the normality of the transformed data. If the statistic of the test provides a p-value

$p > 0.05$, than the transformed data is not significantly different from normal distribution. In the case of the log transformation, the result of the test gives $p = 0.02712$, indeed we can't affirm that the log-transformed response is normally distributed. On the other hand, the result executed on the response transformed by the two-parameter Box-Cox transformation gives a p-value $p = 0.3655$, so that we can confirm that it is normally distributed, as we expected, given that we got $\lambda_1 \neq 0$. Thus, the Box-Cox transformation is more appropriate than a simple log transformation to address non-normality in this data set.

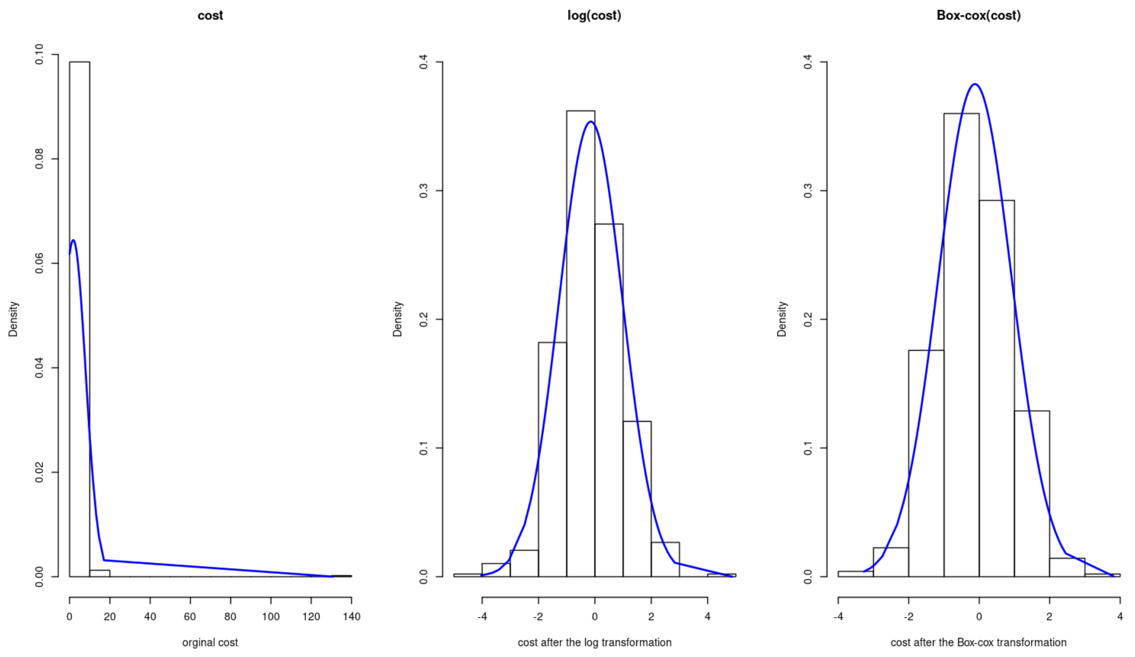

Figure 4.4: The histogram of the distribution of cost per claim. From left to right: without any transformation, with log transformation and with Box-Cox transformation with the $\lambda_1 = -0.10506$ and the $\lambda_2 = 0.04225$

In order to verify whether the extreme value has an impact on the model estimation, we try to repeat the procedure after we take the extreme value out from the data set. The new parameters λ of the Box-Cox transformation on the new dataset change to $\lambda_1 = -0.00998$ and $\lambda_2 = 0.01225$. In this case, λ_1 is much nearer to zero than in the previous case. However, according to the descriptive statistics in Table 4.3, the maximum of the original cost, with and without extreme value, are respectively 130.87(thousand euros) and 17.03(thousand euros), and the mean cost changed just about 0.3 (thousand euros). Similarly, the analogous statistics computed after the response has been transformed by the log transformation and by the Box-Cox transformation do not show any significant changes. The maximum values have the small difference about 1 unit, while the mean values appear to be stable. This applies not only the mean of the cost, but to all other statistics, that keep stable with and without the extreme value.

		Min.	1st-Qu	Median	Mean	3rd-Qu	Max.
With extreme value	Cost	0.018	0.420	0.820	1.810	1.880	130.870
	Log	-4.030	-0.870	-0.200	-0.140	0.630	4.870
	BC	-3.270	-0.810	-0.150	-0.120	0.630	3.810
Without extreme value	Cost	0.018	0.420	0.810	1.540	1.870	17.030
	Log	-4.030	-0.880	-0.200	-0.150	0.630	2.830
	BC	-3.560	-0.850	-0.190	-0.130	0.630	2.790

Table 4.3: Descriptive statistics of cost per claim and transformed cost per claim with and without extreme value

Analogously to what we did on the dataset that included the extreme value, we also execute the Shapiro-Wilk normality test to verify the normal distribution of the data sets. In this case, the p-values obtained for the cost transformed by the log transformation and by the Box-Cox transformation are, respectively, 0.1814 and 0.3442. Therefore, in this case, both the responses transformed by log transformation and by Box-cox transformation are normal distributed.

4.2 Results

Our goal is to analyze the impact of the independent variables on the cost per claim. Since we have two types of data, non telematics and telematics, we consider three different models. The first one, includes all variables, the second one includes only non-telematics variables, and the last model uses only telematics variables. According to the methodology presented in this thesis, we set "speedkm" as the variable with the constrained coefficient $\theta_1 = 1$, for the model with all variables and the models with only telematic factors, while the value of the other parameters θ_i are determined by the model. We do this because we suppose that high speed is one of the main reasons that lead to sever road accidents. Hence, high speed will produce claims with higher cost than those caused by people that drive at lower speed, that can better control their vehicles, and are less likely to generate impacts with severe damages. For the models with only non-telematic variables, we constraint the coefficient of the variable "age" to be equal to 1.

	All variables			Only non-telematics			Only telematics		
	Orig.	Log	Box-Cox	Orig.	Log	Box-Cox	Orig.	Log	Box-Cox
speedkm	1.000	1.000	1.000				1.000	1.000	1.000
age	-0.294*	0.153*	0.148*	1.000	1.000	1.000			
agelic	0.176*	0.097*	0.093**	-0.246*	-0.246*	-0.203*			
agecar	-0.406*	-0.107*	-0.080*	0.066*	0.074*	0.079 *			
parking	-3.024*	-0.162	0.240	0.845*	-0.655*	0.746*			
tkm	-0.138*	-0.044*	-0.144*				-0.053**	-0.423*	-0.218*
nightkm	0.106*	0.117*	0.107*				0.223*	0.089*	0.114*
urbankm	0.005	0.141*	0.104*				-0.024*	0.080*	0.127*
Significant at 1% level $*$ and at 5% level $**$									

Table 4.4: Estimated parameters and their significance for the single index-model for the data set with the original cost, the log transformation of cost, and the Box-cox transformation of cost with the $\lambda_1 = -0.1050598$ and the $\lambda_2 = 0.04225$

First of all, we analyse the data set that includes also the extreme value. The outcome of the fitted parameters is shown in Table 4.4. The table presents the estimated coefficients for the three models that include, respectively, all variables, only non-telematics, and only telematics variables. Within each model, the estimations are divided by groups, according the transformation that is applied to the cost per claim, namely, original (not transformed) response, log-transformed response, and responses transformed with the Box-Cox transformation with $\lambda_1 = -0.1050598$ and the $\lambda_2 = 0.04225$. Overall, we can observe that the estimated parameters by log transformation and Box-Cox transformation are similar, but for both the results differ significantly from those collected by original data. Considering telematic variables, there are slight differences between the model that takes into account all variables and the model that consider only telematics ones. From the data presented in the table, we can observe that the variable "tkm" gets negative sign in all models. Probably, this is due to the fact that when people drive more kilometers, they can improve their driving skills, then make less damage. Meanwhile, the "nightkm" parameter influences positively and strongly on the cost in all models, so if people drive more in the night time, the cost per claim gets higher. This might be due, for example, from the fact that people drive faster at night because there is less traffic at night, as well as darkness reduces visibility and tiredness might be higher. The "urbankm" gets negative values in the only telematics model with original data, but it is positive in other data set. The values show that the accident in urban area may cost more than in suburban, since there may be more people involved and other factors around as shops, vehicle, etc in urban area.

On the other hand, there is a big difference in the estimated parameters between the model with all variables and the model with only non-telematic factors. In the model with all variables, the parameters of "parking" are not significant, however, their values are significant, positive, and high in the model with only non-telematics variables. The positive correlation might be counter-intuitive, given that cars are usually safer in garage. For this reason, we have a doubt whether the cars that are parked carefully at night may be of luxury type, so that value of the costs generated by the claims is higher than the ones that are parked on the street, which may be cars with less monetary value. The parameters of "agelic" are positive in the models with all variables, however, they are negatively correlated with cost in the model with just non-telematics predictors. This appears reasonable, given that the customers who have the license since more years will have more experience in driving, so they may produce less damage than inexperienced drivers who got the license since a few years. Similarly, the results in the models with all variables suggests that older cars generate lower costs per claim. On the contrary, the models with only non-telematics values gave outcomes with positive value, on the other words, the accidents tend to cost more as the car get older.

We perform the same procedure also after the removal of data entry with the extreme value of cost 130870 €. Results of the estimated coefficients are show in Table 4.5. In general, we can observe that there is a slight change of the estimated coefficients in the set of data without the extreme value. We notice that all values of coefficients seem stable in both the models that use only non-telematics and only telematics variables between the sample with extreme value and without extreme value. Meanwhile, the values of the parameters of the model that takes into account the whole set of variables, incur a more noticeable change, even if still modest, if we remove the extreme value.

	All variables			**Only non-telematics**			**Only telematics**		
	Orig.	Log	Box-Cox	Orig.	Log	Box-Cox	Orig.	Log	Box-Cox
speedkm	1.000	1.000	1.000				1.000	1.000	1.000
age	-0.125*	0.112**	0.153*	1.000	1.000	1.000			
agelic	0.093**	0.158**	0.099*	-0.247*	-0.182*	-0.185*			
agecar	-0.292*	-0.087*	-0.107*	0.066*	0.087*	0.090*			
parking	-0.258	0.455	-0.157	0.845*	0.661*	0.686*			
tkm	-0.003*	-0.140*	-0.045*				-0.053**	-0.220*	-0.207 *
nightkm	0.138*	0.106*	0.118*				0.223*	0.115*	0.120*
urbankm	0.019**	0.103*	0.141*				-0.024	0.128*	0.074*
Significant at 1% level $*$ and at 5% level $**$									

Table 4.5: Estimated parameters and their significance for the single index-model for the data set without extreme value, with the orginal cost, the log transformation of cost, and the Box-cox transformation of cost with the $\lambda_1 = -0.00998$ and the $\lambda_2 = 0.01225$

Chapter 5

Conclusions

This master thesis concerns the application of the Box-Cox transformation in the context of single-index models. First, we gave a general overview of the theory related with the topic, including a report of the relevant literature, in particular the proposed estimation techniques and we provided details of our application of the Box-Cox transformation for the single-index model. In details, we employ the transformation on the response variable when it is normally distributed. For the experimental section, we included both a simulation study and experiments on a real-world data set, concerning the prediction of the cost per claim in automobile insurance.

In details, we extend the single-index model for the risk assessment in car insurance from Bolancé et al. (2018), that employ the flexible maximum conditional likelihood for the estimation of the index parameters and the leave-one-out-kernel method for the smooth function. We noticed that when response data belong to a normal distribution, the model provides better results than non-normalized data. For this reason, in this thesis we study the application of the Box-Cox transformation on the response variable.

In particular, we apply the two-parameters Box-Cox transformation on the response variable. The two parameters, named λ_1 and λ_2 are determined in order to maximize the likelihood of the transformed data being normally distributed. We remind that the log transformation, which is widely used in statistics, is a special case of the Box-Cox transformation with $\lambda = 0$. For this reason, we also apply the log transformation and use it for comparison with the Box-Cox transformation. To test our approach, we perform a simulation study on synthetic data and a statistical analysis on a real-world data set.

Regarding the simulation study, we generate responses with different distributions and we show that, for all studied distributions, both the log and the Box-Cox transformation improve by a large measure the predictive accuracy of the single-index model compared to the model fitted on the original response as expected. However, the Box-Cox transformation provides slightly better results than log transformation.

The thesis is completed by a real-world example on car insurance data, which is a concrete application of single-index models, given that they can be used to predict variables such as the expected cost, number of claims or cost per claims the customers are likely to generate, based on a set of data about demographic characteristics and driving style. In our study, we focused on the prediction of the impact of different telematic and non-telematic variables on the cost per claim. We consider the model estimated on the original response and the transformed response by the log and the Box-Cox transformation. In our case, the resulting values for the Box-Cox parameters are $\lambda_1 = -0.10506$ and $\lambda_2 = 0.04225$, which are different from zero. We fitted distinct models that contain all variables, only telematic ones and only non-telematic ones. Finally, given the presence of an extreme value, a data entry with a very high claim cost, we repeated the whole procedure after the removal of the extreme value from de data set. In this case, the difference between the application of the Box-Cox transformation and the log transformation is less significant, given that $\lambda_1 = -0.00998$ and $\lambda_2 = 0.01225$. In total, we considered a 18 different models on the data set, obtained as 3 variable selections (all, telematic, non-telematic) $\times$ 3 transformations (none, log, Box-Cox) $\times$ 2 dataset versions (with and without extreme value).

The models fitted on the response transformed by the Box-Cox method gives slightly different results from the ones obtained from the log transformation, and remarkably different from the models with original data. In comparison with the model that leaves out the extreme value, the parameter values do not change much. Thus, it seems that the extreme value does not influence the estimators of the models. However, the Box-Cox transformation is capable to transform into a normal shape the response variable both in presence and absence of the extreme value, while the log transformation fails in doing so in presence of the extreme value. Thus, the application of our two-parameters Box-Cox transformation is a more reliable methodology than a simple log transformation and, unsurprisingly, than the usage of the original response.

Future work that we plan to realize concerns further studies on the application of the Box-Cox to other real-world data sets that are relevant for the single-index model, as well as the generation of new simulated data sets. Additionally, we plan to combine the methodology with different estimation techniques for the single-index parameters and the smooth function.

Bibliography

M. Bicego and S. Baldo. Properties of the box–cox transformation for pattern classification. *Neurocomputing*, 218:390–400, 2016.

C. Bolancé, R. Cao, and M. Guillén. Flexible maximum conditional likelihood estimation for single-index models to predict accident severity with telematics data. *IREA–Working Papers, 2018, IR18/29*, 2018.

G. E. Box and D. R. Cox. An analysis of transformations. *Journal of the Royal Statistical Society: Series B (Methodological)*, 26(2):211–243, 1964.

A. Cheddad. On box-cox transformation for image normality and pattern classification. *IEEE Access*, 8:154975–154983, 2020.

R. A. Collins and P. J. Barry. Risk analysis with single-index portfolio models: An application to farm planning. *American Journal of Agricultural Economics*, 68(1):152–161, 1986.

M. Delecroix, W. Härdle, and M. Hristache. Efficient estimation in conditional single-index regression. *Journal of Multivariate Analysis*, 86(2):213–226, 2003.

E. J. Elton and M. J. Gruber. Risk reduction and portfolio size: An analytical solution. *The Journal of Business*, 50(4):415–437, 1977.

P. Hall, R. C. Wolff, and Q. Yao. Methods for estimating a conditional distribution function. *Journal of the American Statistical association*, 94(445):154–163, 1999.

A. K. Han. Non-parametric analysis of a generalized regression model: the maximum rank correlation estimator. *Journal of Econometrics*, 35(2-3):303–316, 1987.

W. Härdle. *Applied nonparametric regression*. Number 19. Cambridge university press, 1990.

W. Hardle, P. Hall, and H. Ichimura. Optimal smoothing in single-index models. *The annals of Statistics*, 21(1):157–178, 1993.

T. J. Hastie. Generalized additive models. In *Statistical models in S*, pages 249–307. Routledge, 2017.

D. Hawkins and S. Weisberg. Combining the box-cox power and generalised log transformations to accommodate nonpositive responses in linear and mixed-effects linear models. *South African Statistical Journal*, 51(2):317–328, 2017.

J. L. Horowitz and W. Härdle. Direct semiparametric estimation of single-index models with discrete covariates. *Journal of the American Statistical Association*, 91(436):1632–1640, 1996.

H. Ichimura. Semiparametric least squares (sls) and weighted sls estimation of single-index models. *Journal of econometrics*, 58(1-2):71–120, 1993.

Insurance Europe. Statistics. URL `https://insuranceeurope.eu/statistics`.

J. John and N. R. Draper. An alternative family of transformations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 29(2):190–197, 1980.

R. W. Klein and R. H. Spady. An efficient semiparametric estimator for binary response models. *Econometrica: Journal of the Econometric Society*, pages 387–421, 1993.

M. H. Lee, H. J. Sadaei, and Suhartono. Improving taiex forecasting using fuzzy time series with box–cox power transformation. *Journal of Applied Statistics*, 40(11):2407–2422, 2013.

R. Maciejewski, A. Pattath, S. Ko, R. Hafen, W. S. Cleveland, and D. S. Ebert. Automated box-cox transformations for improved visual encoding. *IEEE transactions on visualization and computer graphics*, 19(1):130–140, 2012.

M. J. Miranda and J. W. Glauber. Systemic risk, reinsurance, and the failure of crop insurance markets. *American journal of agricultural economics*, 79(1):206–215, 1997.

H. L. Nelson Jr and C. Granger. Experience with using the box-cox transformation when forecasting economic time series. *Journal of Econometrics*, 10(1):57–69, 1979.

A. Pajor. Bayesian analysis of the box-cox transformation in stochastic volatility models. *Dynamic Econometric Models*, 9:81–90, 2009.

A. Pananjady and D. P. Foster. Single-index models in the high signal regime. *IEEE Transactions on Information Theory*, 67(6):4092–4124, 2021.

T. Proietti and H. Lütkepohl. Does the box–cox transformation help in forecasting macroeconomic time series? *International Journal of Forecasting*, 29(1):88–99, 2013.

B. Sennaroğlu and Ö. Şenvar. Performance comparison of box-cox transformation and weighted variance methods with weibull distribution. *Journal of Aeronautics and Space Technologies*, 8(2):49–55, 2015.

W. F. Sharpe. A simplified model for portfolio analysis. *Management science*, 9(2):277–293, 1963.

R. P. Sherman. The limiting distribution of the maximum rank correlation estimator. *Econometrica: Journal of the Econometric Society*, pages 123–137, 1993.

E. Strzalkowska-Kominiak and R. Cao. Maximum likelihood estimation for conditional distribution single-index models under censoring. *Journal of Multivariate Analysis*, 114: 74–98, 2013.

T. Terasaka and Y. Hosoya. A modified box-cox transformation in the multivariate arma model. *Journal of the Japan Statistical Society*, 37(1):1–28, 2007.

S. Weisberg. Yeo-johnson power transformations. *Department of Applied Statistics, University of Minnesota. Retrieved June*, 1:2003, 2001.

C.-H. Wu, S. Lin, D. Yang, and W. Pearn. Box-cox transformation approach for evaluating non-normal processes capability based on the cpk index. *Journal of Testing and Evaluation*, 42(4), 2014.

T. Zhang and B. Yang. Box–cox transformation in big data. *Technometrics*, 59(2):189–201, 2017.

Appendix A

R code

In this section we present the code of our application. For sake of simplicity, we present just the case of 100 samples with size n=500.

A.1 Prerequisite and Data

First of all, we load the needed libraries

```
setwd("path/to/folder")
library(RGCCA)
library(EnvStats)
library(mvtnorm)
```

Then, we declare and define the set of variables needed in our study. First, we generate the independent variable X, and divide it in 100 samples. In each sample, X is a 500×3 matrix, in other words, X is combined by three explanatory variables. The initial parameters θ are assigned with the values $(1, 1.3, 0.5)$. Depending on the distribution, we have the values of the dependent variable (Y) given by the values of X and θ. There are 500 values Y in each sample.

```
# NUMBER OF REPLICAS
nrep<-100
# Sample size
n<-500
# Number of explanatory variables
k=3
x3<-matrix(0,n,(nrep*k))
sig0<-diag(k)

set.seed(123)
```

```
for (i in 1:nrep){x3[,(k*(i-1)+1):(k*i)]<- rmvnorm(n, mean =
 ↪  rep(0, k), sigma = sig0)}

# Index
index<-matrix(0,n,nrep)
dep_norm0<-matrix(0,n,nrep)
b0<-as.matrix(c(1,1.3,0.5))
for (j in 1:nrep){
 for (i in 1:n){
   index[i,j]<-x3[i,(k*(j-1)+1):(k*j)]%*%b0
   dep_norm0[i,j]<-rnorm(1, mean = index[i,j], sd =
    ↪  abs(index[i,j]))
 }
}
dep_norm0<-exp(dep_norm0) #to generate index values following
 ↪  the lognormal distribution.
# non-parametric index estimation
b_ini<-matrix(1,nrep,k)
```

After having the values of the dependent variable Y, we start to estimate the parameter λ, that we need to transform the variable Y. In this context, we want to use Box-Cox transformation with two parameters, therefore, we estimate the two parameters λ_1 and λ_2 and store them as a 100×2 matrix (line 12).

```
# Box-Cox
l12<-matrix(0,nrep,2)
dlambda=0.01
for (j in 1:nrep){
 Y<-dep_norm0[,j]
 lambda2=seq((-min(Y)+0.01),(-min(Y)+0.01+3),dlambda)
 lambda1=rep(0,length(lambda2))
 liklambda=rep(0,length(lambda2))
 for(i in 1:length(lambda2)){
   boxcox.fit=boxcox((Y+lambda2[i]), lambda =c(-4, 1),
    ↪  objective.name = "Log-Likelihood", optimize = TRUE)
   lambda1[i]=boxcox.fit$lambda;
   liklambda[i]=boxcox.fit$objective
 }
 l12[j,1]=lambda1[which.max(liklambda)]
 l12[j,2]=lambda2[which.max(liklambda)]
}
```

A.2 Single index function

This section shows the algorithm of the single index with the aim to estimate the parameters θ of the function. We will get various values of θ, h_1, and h_2 as the bandwidths of the kernel functions, then we also store those data to do the inference analysis in the next part.

```
singleindex<-function(nn,t2=b_ini[,2:k], type, c1=5e-324,
 ↪  MM=nrep){
  dim1<-ncol(t2)
  tediffh<-array(0, dim=c(MM,dim1))
  dd<-array(0, dim=c(MM,3))
  mh<-array(0, dim=c(MM,2))
  it<-array(0, dim=c(MM,1))
  l<-1
  while(l<=MM){
    datosX<-x3[,(k*(l-1)+1):(k*l)]
    Xv<-datosX
    Y1<-dep_norm0[,l]
    Y1<-(((Y1+l12[l,2])**l12[l,1])-1)/l12[l,1]
    # kernel function, its derivatives and its integral;
     ↪  Gaussian
    Ke<-function(x){return(dnorm(x))}
    dKe<-function(x){
      r<--dnorm(x)*x
      return(r)}
    ddKe<-function(x){
      r<-dnorm(x)*(x-1)*(x+1)
      return(r)}
    pKe<-function(x){return(pnorm(x))}

    likelih<-function(argumn,h1,h2){
      Likelihood<-rep(1,times=nn)
      for(k in 1:nn){
        help1<-c(Ke(c(sum(c(1,argumn)*Xv[k,])
         ↪  -colSums(c(1,argumn)*t(Xv)))/h1)*Ke((Y1[k]-Y1)/h2))
        help2<-c(Ke(c(sum(c(1,argumn)*Xv[k,])
         ↪  -colSums(c(1,argumn)*t(Xv)))/h1))
        term1<-sum(help1[-k])/(h2*sum(help2[-k]))
        if(sum(help1[-k])==0){term1<-0}
        logDensity<-log(max(term1,c1))
         Likelihood[k]<-logDensity
      }
      return(-sum(Likelihood))
    }

    # likelihhod function needed to optimize in (h1,h2)
    likeliarg<-function(hh,ar){
    return(likelih(argumn=ar,h1=hh[1],h2=hh[2]))
    }
```

```
#likelihood function needed to optimize in (h,
 ↪  sd(X*theta),h*sd(Z)).

likelihnew<-function(ar,hh){ return(likelih(argumn=ar,
 ↪  h1=hh*sd(ColSums(c(1,ar)*t(Xv)))
, h2=hh*sdkm(Y1=Y1)))}
# standard deviation of Z
sdkm<-function(Y1=Y1){
  return(sqrt((sum(Y1^2)/nn)-(sum(Y1)/nn)^2))
}
#########################################################
# choosing start parameters for optimization
h1A<-1
h2A<-1
teA<-rep(1,times=dim1)
te<-rep(0,times=dim1)
b<-c(1,teA)
sdX<-sd(colSums(b%*%t(Xv)))
h1start<-sdX*nn^(-2/13)
sdY1<-sdkm(Y1=Y1)
sdY2<-IQR(Y1)/1.349
sdY<-min(sdY1,sdY2)
h2start<-sdY*nn^(-4/13)
count<-0
iter<-count
# we optimize first in the parameter theta then in
 ↪  bandwidths
# the optimization goes until the distances between
 ↪  consecutive steps <0.001 or 200 steps were reached.

  while((count<200)&((sum(abs(teA-te)/dim1)>0.0001)|
  (abs(h1A-h1start)>0.001)|(abs(h2A-h2start)>0.001))){
  h1A<-h1start
  h2A<-h2start
  te<-teA
   ↪  teA<-optim(par=c(te),fn=likelih,h1=h1start,h2=h2start)
  $par      if(likelih(argumn=teA,h1=h1A,h2=h2A) >
   ↪  likelih(argumn=te,
  h1=h1A,h2=h2A)){
    teA<-te
  }
  # minimize -loglikelihood in (h1,h2)
  b<-c(1,teA)
  sdX<-sd(colSums(b%*%t(Xv)))
  lowX<-0.01*sdX*nn^(-2/13)
  upX<-10*sdX*nn^(-2/13)
  sdY1<-sdkm(Y1=Y1)
```

```
        sdY2<-IQR(Y1)/1.349
        sdY<-min(sdY1,sdY2)
        lowY<-0.01*sdY*nn^(-4/13)
        upY<-10*sdY*nn^(-4/13)
         ↪  ha<-optim(par=c(h1start,h2start),fn=likeliarg,
        lower=c(lowX,lowY),upper=c(upX,upY),
        method="L-BFGS-B",ar=teA)
        h1start<-ha$par[1]
        h2start<-ha$par[2]
        print("count="); print(count)
        count<-count+1
        d1<-sum(abs(teA-te)/dim1)
        d2<-abs(h1A-h1start)
        d3<-abs(h1A-h1start)
      }
      # get values (c(h1start,h2start,teA))
      tediffh[l,]<-c(teA)
      mh[l,]<-c(h1start,h2start)
      print("l="); print(l)
      dd[l,]<-as.matrix(c(d1,d2,d3))
      it[l]<-count
      l<-l+1
    }
      #################### printing
       ↪  results#######################
    print(iter)
    mtheta<-as.matrix(tediffh)
    mh<-as.matrix(mh)
    result<-cbind(mtheta,mh,it,l12)
    colnames(result)<-c("theta2","theta3", "h1",
     ↪  "h2","niter","l1","l2")
    write.csv(result,file =
     ↪  "result_lognormal_500_bc.txt",row.names = F)
    bias1<-colSums(tediffh)/MM-b0[2:k]
    print("bias of theta:")
    print(bias1)
    sthe<-var(tediffh)
    print("covariance of theta:")
    print(sthe)
    print("MSE")
    mse1<-diag(sthe)+bias1^2
    print(mse1)
  }
```

A.3 Execution

We present hereafter an example of execution with sample size 500. The code is encapsulated into functions, so the only user-defined command needed to start the application is the following:

```
tiempo.ini<-Sys.time()
singleindex(nn=500,type=1)
tiempo.fin<-Sys.time()
tiempo<-tiempo.fin-tiempo.ini
tiempo
```

A.4 Inference power analysis

This section combines the code for inference power analysis. We need to use the parameters θ, bandwidths, and the Box-Cox parameters λ that we estimated before. For each set of dependent variables, we need to do four hypothesis testing: $\theta_2 = 0$, $\theta_3 = 0$, $theta_2 = \theta_{02} = 1.3$, and $theta_3 = \theta_{03} = 0.5$. The code below is for the set of data in which the original dependent variable is generated following the log normal distribution with the application of Box-Cox transformation.

```
library(RGCCA)
library(mvtnorm)
# NUMBER OF REPLICAS
nrep<-100
# Sample size
n<-500
# Number of explanatory variables
k=3
x3<-matrix(0,n,(nrep*k))
sig0<-diag(k)
set.seed(123)
for (i in 1:nrep){x3[,(k*(i-1)+1):(k*i)]<-rmvnorm(n, mean =
 ↪  rep(0, k), sigma = sig0)}
# Index
index<-matrix(0,n,nrep)
dep_norm0<-matrix(0,n,nrep)
b0<-as.matrix(c(1,1.3,0.5))
for (j in 1:nrep){
  for (i in 1:n){
    index[i,j]<-x3[i,(k*(j-1)+1):(k*j)]%*%b0
    dep_norm0[i,j]<-rnorm(1, mean = index[i,j],
    sd = abs(index[i,j]))
  }
}
 #Generating index values following the lognormal distribution
dep_norm0<-exp(dep_norm0)

# INFERENCE ANALYSIS
tiempo.ini<-Sys.time()
nn<-n
Ke<-function(x){return(dnorm(x))}
dKe<-function(x){
  r<--dnorm(x)*x
  return(r)}
ddKe<-function(x){
  r<-dnorm(x)*(x-1)*(x+1)
  return(r)}
pKe<-function(x){return(pnorm(x))}
c1=5e-324
condens<-function(argumn,h1,h2){
  argumn<-argumn[-1]
  cdens<-rep(0,times=nn)
  for(k in 1:nn){
# to define conditional density and survival function we need
 ↪   following help functions

    help1<-c(Ke(c(sum(c(1,argumn)*Xv[k,])-colSums(c(1,argumn)
    *t(Xv)))/h1)*Ke((Y1[k]-Y1)/h2))

    help2<-c(Ke(c(sum(c(1,argumn)*Xv[k,])-colSums(c(1,argumn)
    *t(Xv)))/h1))

    dens<-sum(help1[-k])/(h2*(sum(help2[-k]+c1)))
    cdens[k]<-(max(c1,dens))
  }
  return(cdens)
}
ere<-function(argumn,h1,h2){
  argumn<-argumn[-1]
  ss<-rep(0,times=nn)
  for(k in 1:nn){

    help3<-c(Ke(c(sum(c(1,argumn)*Xv[k,])-colSums(c(1,argumn)*
    t(Xv)))/h1)*Ke((Y1[k]-Y1)/h2))
    ss[k]<-sum(help3[-k])/((nn-1)*h1*h2)
  }
  return(ss)
}
ese<-function(argumn,h1,h2){
```

```
  argumn<-argumn[-1]
  ss<-rep(0,times=nn)
  for(k in 1:nn){

    help3<-c(Ke(c(sum(c(1,argumn)*Xv[k,])-colSums(c(1,argumn)
    *t(Xv)))/h1))
    ss[k]<-sum(help3[-k])/((nn-1)*h1)
  }
  return(ss)
}
argumn<-b0
nn<-n
h<-2
i<-1
ddere<-function(argumn,h1,h2,i){
  argumn<-argumn[-1]
  ss<-array(0,dim=c(length(argumn),length(argumn),nn))
  for(h in 1:nn){

    help5<-t(t(Xv[i,]-Xv[h,]))%*%(Xv[i,]-Xv[h,]) * Ke((Y1[i] -
     ↪  Y1[h])/h2) * ddKe(c(sum(c(1,argumn)*Xv[i,])  -
     ↪  sum(c(1,argumn) * Xv[h,]))/h1)
    ss[,,h]<-help5[2:k,2:k]
  }
  rr<-apply(ss[,,-i],1:2,sum)/((nn-1)*(h1**3)*h2)
  return(rr)
}

ddese<-function(argumn,h1,h2,i){
  argumn<-argumn[-1]
  ss<-array(0,dim=c(length(argumn),length(argumn),nn))
  for(h in 1:nn){

    help5<-t(t(Xv[i,]-Xv[h,]))%*%(Xv[i,]-Xv[h,]) *
    ddKe(c(sum(c(1,argumn)*Xv[i,]) -
     ↪  sum(c(1,argumn)*Xv[h,]))/h1)
    ss[,,h]<-help5[2:k,2:k]
  }
  rr<-apply(ss[,,-i],1:2,sum)/((nn-1)*(h1**3))
  return(rr)
}
dere<-function(argumn,h1,h2,i){
  argumn<-argumn[-1]
  ss<-matrix(0,nn,length(argumn))
  for(h in 1:nn){

    help6<-(Xv[i,]-Xv[h,])*Ke((Y1[i]-Y1[h])/h2) * dKe(c(
    sum(c(1,argumn)*Xv[i,]) - sum(c(1,argumn)*Xv[h,])) / h1)
    ss[h,]<-help6[2:k]
  }
  rr<-colSums(ss[-i,])/((nn-1)*(h1**2)*h2)
  return(rr)
}

dese<-function(argumn,h1,h2,i){
  argumn<-argumn[-1]
  ss<-matrix(0,nn,length(argumn))
  for(h in 1:nn){
    help7<-(Xv[i,]-Xv[h,])*dKe(c(sum(c(1,argumn)*Xv[i,])
    - sum(c(1,argumn)*Xv[h,]))/h1)
    ss[h,]<-help7[2:k]
  }
  rr<-colSums(ss[-i,])/((nn-1)*(h1**2))
  return(rr)
}
result<-read.table("result_lognormal_500_bc.txt",header =
 ↪  TRUE,sep = ",")
l12<-result[,6:7]
kk<-k
dimX<-kk
z1<-matrix(0,nrep,1)
STE<-matrix(0,nrep,1)

c1=5e-324
# Power analyisis. H0 false
# H0: Theta2=0

fderiv<-matrix(0,nn,dimX-1)
sderiv<-array(0,dim=c((dimX-1),(dimX-1),nn))
for (i in 1:nrep){
  theta<-cbind(1,result[i,1:(kk-1)])
  h1<-result[i,kk]
  h2<-result[i,kk+1]
  theta<-t(as.matrix(theta))
  datosX<-x3[,(kk*(i-1)+1):(kk*i)]

  Xv<-datosX

  Y1<-dep_norm0[,i]
  Y1<-(((Y1+l12[i,2])**l12[i,1])-1)/l12[i,1]

  si<-ese(theta,h1,h2)
  ri<-ere(theta,h1,h2)
  for(m in 1:nn){

    fderiv[m,]<-(dere(theta,h1,h2,m)*si[m]-ri[m]
    *dese(theta,h1,h2,m))/((si[m]**2)+c1)

    sderiv[,,m]<-(ddere(theta,h1,h2,m)/(ri[m]+c1))
    -((t(t(dere(theta,h1,h2,m)))%*%dere(theta,h1,h2,m))
    /((ri[m]**2)+c1))-(ddese(theta,h1,h2,m)/(si[m]+c1))
     ↪  +((t(t(dese(theta,h1,h2,m)))%*%dese(theta,h1,h2,m))
    /((si[m]**2)+c1))
  }
  fderiv<-fderiv/condens(theta,h1,h2)
  Sigma1<-(1/nn)*t(fderiv)%*%fderiv
  Itheta=apply(sderiv,1:2,sum)/nn
  Sigma2=solve(Itheta)
  Sigma<-Sigma2%*%Sigma1%*%Sigma2/n

  STE[i]<-sqrt(diag(Sigma))[1]
  z1[i]<-(theta[2])/STE[i]
  print(i)
}

pvalue<-1-pnorm(abs(z1))
ind<-(as.matrix(pvalue<=0.025))
p1<-mean(as.numeric(ind[,1]))
print("p-value")
p1
mean(pvalue)
# H0: Theta3=0
fderiv<-matrix(0,nn,dimX-1)
sderiv<-array(0,dim=c((dimX-1),(dimX-1),nn))
for (i in 1:nrep){
  theta<-cbind(1,result[i,1:(kk-1)])
  h1<-result[i,kk]
  h2<-result[i,kk+1]
  theta<-t(as.matrix(theta))
  datosX<-x3[,(kk*(i-1)+1):(kk*i)]
  Xv<-datosX
  Y1<-dep_norm0[,i]
  Y1<-(((Y1+l12[i,2])**l12[i,1])-1)/l12[i,1]

  si<-ese(theta,h1,h2)
  ri<-ere(theta,h1,h2)

  for(m in 1:nn){

    fderiv[m,]<-(dere(theta,h1,h2,m)*si[m]-ri[m]
    *dese(theta,h1,h2,m))/((si[m]**2)+c1)

    sderiv[,,m]<-(ddere(theta,h1,h2,m)/(ri[m]+c1))
    -((t(t(dere(theta,h1,h2,m)))%*%dere(theta,h1,h2,m))
    /((ri[m]**2)+c1))-(ddese(theta,h1,h2,m)/(si[m]+c1))
    +((t(t(dese(theta,h1,h2,m)))%*%dese(theta,h1,h2,m))
    /((si[m]**2)+c1))
  }
  fderiv<-fderiv/condens(theta,h1,h2)
  Sigma1<-(1/nn)*t(fderiv)%*%fderiv
  Itheta=apply(sderiv,1:2,sum)/nn
  Sigma2=solve(Itheta)
  Sigma<-Sigma2%*%Sigma1%*%Sigma2/n
  STE[i]<-sqrt(diag(Sigma))[2]
  z1[i]<-(theta[3])/STE[i]
  print(i)
}
hist(z1)
pvalue<-1-pnorm(abs(z1))
ind<-(as.matrix(pvalue<=0.025))
p1<-mean(as.numeric(ind[,1]))
print("p-value")
p1
mean(pvalue)
tiempo.fin<-Sys.time()
tiempo<-tiempo.fin-tiempo.ini
tiempo

# Conficence analyisis. H0 true
# H0: Theta2=1.3
fderiv<-matrix(0,nn,dimX-1)
sderiv<-array(0,dim=c((dimX-1),(dimX-1),nn))
for (i in 1:nrep){
  theta<-cbind(1,result[i,1:(kk-1)])
  h1<-result[i,kk]
  h2<-result[i,kk+1]
  theta<-t(as.matrix(theta))
  datosX<-x3[,(kk*(i-1)+1):(kk*i)]

  Xv<-datosX

  Y1<-dep_norm0[,i]
  Y1<-(((Y1+l12[i,2])**l12[i,1])-1)/l12[i,1]

  si<-ese(theta,h1,h2)
  ri<-ere(theta,h1,h2)
  for(m in 1:nn){
    fderiv[m,]<-(dere(theta,h1,h2,m)*si[m]-ri[m]
    *dese(theta,h1,h2,m))/((si[m]**2)+c1)

    sderiv[,,m]<-(ddere(theta,h1,h2,m)/(ri[m]+c1))
    -((t(t(dere(theta,h1,h2,m)))%*%dere(theta,h1,h2,m))
    /((ri[m]**2)+c1))-
      (ddese(theta,h1,h2,m)/(si[m]+c1))
       ↪  +((t(t(dese(theta,h1,h2,m)))%*%dese(theta,h1,h2,m))
      /((si[m]**2)+c1))
  }
  fderiv<-fderiv/condens(theta,h1,h2)
  Sigma1<-(1/nn)*t(fderiv)%*%fderiv
  Itheta=apply(sderiv,1:2,sum)/nn
  Sigma2=solve(Itheta)
  Sigma<-Sigma2%*%Sigma1%*%Sigma2/n
  STE[i]<-sqrt(diag(Sigma))[1]
  z1[i]<-(theta[2]-1.3)/STE[i]
  print(i)
}
hist(z1)
pvalue<-1-pnorm(abs(z1))
ind<-(as.matrix(pvalue>0.025))
p1<-mean(as.numeric(ind[,1]))
print("1-p-value")
p1
mean(pvalue)
tiempo.fin<-Sys.time()
tiempo<-tiempo.fin-tiempo.ini
tiempo

tiempo.ini<-Sys.time()
# H0: Theta3=0.5
fderiv<-matrix(0,nn,dimX-1)
sderiv<-array(0,dim=c((dimX-1),(dimX-1),nn))
for (i in 1:nrep){
  theta<-cbind(1,result[i,1:(kk-1)])
  h1<-result[i,kk]
  h2<-result[i,kk+1]
  theta<-t(as.matrix(theta))
  datosX<-x3[,(kk*(i-1)+1):(kk*i)]

  Xv<-datosX

  Y1<-dep_norm0[,i]
  Y1<-(((Y1+l12[i,2])**l12[i,1])-1)/l12[i,1]
  si<-ese(theta,h1,h2)
  ri<-ere(theta,h1,h2)

  for(m in 1:nn){

    fderiv[m,]<-(dere(theta,h1,h2,m)*si[m]-ri[m]
    *dese(theta,h1,h2,m))/((si[m]**2)+c1)

    sderiv[,,m]<-(ddere(theta,h1,h2,m)/(ri[m]+c1))
    -((t(t(dere(theta,h1,h2,m)))%*%dere(theta,h1,h2,m))/
    ((ri[m]**2)+c1))-(ddese(theta,h1,h2,m)/(si[m]+c1))
    +((t(t(dese(theta,h1,h2,m)))%*%dese(theta,h1,h2,m))
    /((si[m]**2)+c1))
  }
  fderiv<-fderiv/condens(theta,h1,h2)
  Sigma1<-(1/nn)*t(fderiv)%*%fderiv
  Itheta=apply(sderiv,1:2,sum)/nn
  Sigma2=solve(Itheta
  Sigma<-Sigma2%*%Sigma1%*%Sigma2/n
  STE[i]<-sqrt(diag(Sigma))[2]
  z1[i]<-(theta[3]-0.5)/STE[i]
  print(i)
}

pvalue<-1-pnorm(abs(z1))
ind<-(as.matrix(pvalue>0.025))
p1<-mean(as.numeric(ind[,1]))
print("1-p-value")
p1
mean(pvalue)
tiempo.fin<-Sys.time()
tiempo<-tiempo.fin-tiempo.ini
tiempo
```

A.5 The analysis of car accident cost

Finally, we analyse the real data on the automobile insurance sector. We have three variable groups (all, telematic, non-telematic) and three transformations (non, log, Box-Cox transformation) and two data set versions (with and without extreme value).

```
library(RGCCA)
library(e1071)
library(MultiSkew)
library(robustbase)
library(EnvStats)
ComputeModel <- function(data_file, transform = "none",
 ↪  remove_extreme = FALSE, selected_variables = "all")
{data<-read.table(data_file,header=TRUE,sep=",")
  # Parameters
  if(remove_extreme){
    data<-data[-488,]
  }
  n<-nrow(data)
  cost<-data$cost

  # Explanatory variables
  mix1<-data$Nclaims
  mix2<-data$age
  mix3<-data$gender
  mix4<-data$agelic
  mix5<-data$agecar
  mix6<-data$parking
  mix7<-data$tkm/1000
  mix8<-data$nightkm
  mix8<-(mix8/data$tkm)*100
  mix9<-data$urbankm
  mix10<-data$speedkm

  # Dependent variable
  cost<-cost/1000
  # Transformations
  Log_cost<-log(cost)

  data2<-data.frame(as.matrix(cbind(cost, Log_cost, mix2,
   ↪  mix4,
  mix5, mix6, mix7, mix8, mix9, mix10)))

  # Descriptive statistics
  Means<-colMeans(data2)
  Variances<-diag(var(data2))
  Desviacion<-sqrt(Variances)
  Q<-apply(data2,2,quantile)
  # Los concatenamos por filas
  Estadisticos<-rbind(Means,Desviacion,Q)
  Estadisticos<-t(Estadisticos)
  colnames(Estadisticos)<-c('Means', 'STD', 'Min', 'Q25',
  'Median', 'Q75', 'Max')
  knitr::kable(Estadisticos,digits=8, caption="Estad??sticos
   ↪  Descriptivos")

  # Scatterplots
  # layout(matrix(c(1,2,3,4,5,6,7,0), 2, 4, byrow = TRUE))
  # plot(mix2,Log_cost,pch=19, cex=.3, xlab="Age",
   ↪  ylab="log(cost)")
  # plot(mix4,Log_cost,pch=19, cex=.3, xlab="Age of driving
   ↪  licence", ylab="log(cost)")
  # plot(mix5,Log_cost,pch=19, cex=.3, xlab="Age of car",
   ↪  ylab="log(cost)")
  # plot(mix7,Log_cost,pch=19, cex=.3, xlab="Annual
   ↪  kilometers", ylab="log(cost)")
  # plot(mix8,Log_cost,pch=19, cex=.3, xlab="Percentage of
   ↪  night kilometers", ylab="log(cost)")
  # plot(mix9,Log_cost,pch=19, cex=.3, xlab="Percentage of
   ↪  kilometers on urban roads", ylab="log(cost)")
  # plot(mix10,Log_cost,pch=19, cex=.3, xlab="Percentage of
   ↪  kilometers with speeding ", ylab="log(cost)")

  # Covariates in the trhree estimated models: TO SELECT ONE
   ↪  VECTOR
  if(selected_variables == "all")
  {
    dataX<-as.matrix(cbind(mix10, mix2, mix4, mix5, mix6,
     ↪  mix7, mix8, mix9)) #All variables
  }else if(selected_variables == "telematics")
  {
    dataX<-as.matrix(cbind(mix10,mix7,mix8,mix9)) #Only
     ↪  telematics
  }else if(selected_variables == "no_telematics")
  {
    dataX<-as.matrix(cbind(mix2,mix4,mix5,mix6)) #Only no
     ↪  telematics
  }

  dimX<-ncol(dataX)
  nn<-nrow(dataX)

  Y<-cost
  l12<-matrix(0,1,2)
  dlambda=0.01
  lambda2=seq((-min(Y)+0.01),(-min(Y)+0.01+3),dlambda)
  lambda1=rep(0,length(lambda2))
  liklambda=rep(0,length(lambda2))
  for(i in 1:length(lambda2)){
    boxcox.fit=boxcox((Y+lambda2[i]), lambda =c(-4, 1),
     ↪  objective.name = "Log-Likelihood", optimize = TRUE)
    lambda1[i]=boxcox.fit$lambda;
    liklambda[i]=boxcox.fit$objective
  }
```

```
l12[1]=lambda1[which.max(liklambda)]
l12[2]=lambda2[which.max(liklambda)]
l12
# Initial parameters
ini<-lm(Y~dataX)
summary(ini)

ini<-as.matrix(ini$coefficients)
ini<-ini[-1]
ini<-as.matrix(ini/ini[1])
ini<-t(ini)

# kernel function, its derivatives and its integral;
 ↪  Gaussian
Ke<-function(x){return(dnorm(x))}
dKe<-function(x){
  r<--dnorm(x)*x
  return(r)}
ddKe<-function(x){
  r<-dnorm(x)*(x-1)*(x+1)
  return(r)}
pKe<-function(x){return(pnorm(x))}

# R rounds all numbers smaller than c1 to 0.
# That's why I introduce c1 so that the likelihood is well
 ↪  defined.
c1=5e-324

# nn=sample size
singleindex<-function(nn,t2=ini[-1], c1=5e-324){
  dim1<-length(t2)
  one<-1
  # save results for estimated parameters
  # when two selected bandwidths
  tediffh<-array(0, dim=c(dim1,1))
  Xv<-dataX

  #define theta, while theta1=1 (always! Model restriction)
  thetav<-c(one,t2)
  print(thetav)

  # generate data
  score1<-colSums(thetav*t(Xv))
  if(transform == "boxcox")
  {
    Y1<-(((Y+l12[2])**l12[1])-1)/l12[1]
    print(paste("Boxcox parameters, lambda_1 = ", l12[1], ",
      ↪  lambda_2 = ", l12[2], sep=""))
  }
  else if (transform == "log")
  {
    Y1<-Log_cost
  } else if (transform == "none")
  {
    Y1<-Y
  }

  # likelihood function. It returns -loglikelihood and hence
   ↪  we have minimization problem
  #Theta=argumn
  #h1 is associated with Theta'*x and h2 with y.
  likelih<-function(argumn,h1,h2){

    Likelihood<-rep(1,times=nn)

    for(k in 1:nn){

      # to define conditional density and survival function
       ↪  we need following help functions

      help1<-c(Ke(c(sum(c(one,argumn) * Xv[k,])
           - colSums(c(one,argumn) * t(Xv)))/h1) *
            ↪  Ke((Y1[k]-Y1)/h2))
      help2<-c(Ke(c(sum(c(one,argumn) * Xv[k,])
           - colSums(c(one,argumn) * t(Xv)))/h1))

      term1<-sum(help1[-k])/(h2*sum(help2[-k]))

      if(sum(help1[-k])==0){term1<-0}

      logDensity<-log(max(term1,c1))

      Likelihood[k]<-logDensity
    }
    return(-sum(Likelihood))
  }


  # likelihhod function needed to optimize in (h1,h2)

  likeliarg<-function(hh,ar){
    return(likelih(argumn=ar,h1=hh[1],h2=hh[2]))
  }
```

```
# likelihood function needed to optimize in (h
 ↪  sd(X*theta),h*sd(Z))
likelihnew<-function(ar,hh){
  return(likelih(argumn=ar,
   ↪  h1=hh*sd(colSums(c(one,ar)*t(Xv))),
   ↪  h2=hh*sdkm(Y1=Y1)))
}
# standard deviation of Z
sdkm<-function(Y1=Y1){
  return(sqrt((sum(Y1^2)/nn)-(sum(Y1)/nn)^2))
}

#########################################################
# choosing start parameters for optimization
h1start<-nn^(-2/13)
h2start<-nn^(-4/13)

h1A<-1
h2A<-1
teA<-rep(0,times=dim1)
te<-rep(1,times=dim1)
count<-0
while((count<50) & ((sum(abs(teA-te)/dim1)>0.000001) |
 ↪  (abs(h1A-h1start)>0.001) |
                  (abs(h2A-h2start)>0.1))){
  h1A<-h1start
  h2A<-h2start
  te<-teA

  teA<-optim(par=c(te),
   ↪  fn=likelih,h1=h1start,h2=h2start)$par

  # this next line is "just in case". It checks if the new
   ↪  found
  # parameter is smaller than the previous one (recall
   ↪  that we seach for minimum)

  if(likelih(argumn=teA, h1=h1A,h2=h2A) >
   ↪  likelih(argumn=te, h1=h1A, h2=h2A)){
    teA<-te
  }

  # minimize -loglikelihood in (h1,h2)

  sdX1<-sd(colSums(ini%*%t(Xv)))
  sdX2<-IQR(colSums(ini%*%t(Xv)))/1.349
  sdX<-min(sdX1,sdX2)
```

```
      lowX<-0.1*sdX*nn^(-2/13)
      upX<-4*sdX*nn^(-2/13)
      sdY1<-sdkm(Y1=Y1)
      sdY2<-IQR(Y1)/1.349
      sdY<-min(sdY1,sdY2)
      lowY<-0.1*sdY*nn^(-4/13)
      upY<-4*sdY*nn^(-4/13)

      ha<-optim(par=c(h1start,h2start),
       ↪  fn=likeliarg,lower=c(lowX,upX), upper=c(lowY,upY),
       ↪  method="L-BFGS-B",ar=teA)

      h1start<-ha$par[1]
      h2start<-ha$par[2]
      print("count="); print(count)
      count<-count+1
    }
    tediffh<-c(teA)
    ##################### printing results
     ↪  ########################
    print("Smoothing parameters estimated index parameters")
    result<-c(h1start,h2start,tediffh)
    return(result)
  }

  tiempo.ini<-Sys.time()
  par<-singleindex(nn=nn)
  tiempo.fin<-Sys.time()
  tiempo<-tiempo.fin-tiempo.ini
  tiempo
  # Printing the estimated parameter
  par
  # Likelihood function
  likeli<-function(Xv,Y1,argumn,h1,h2){
    argumn<-argumn[-1]
    Likelihood<-rep(1,times=nn)

    for(k in 1:nn){
      # to define conditional density and survival function we
       ↪  need following help functions

      help1<-c(Ke(c(sum(c(1,argumn) * Xv[k,]) -
       ↪  colSums(c(1,argumn) * t(Xv)))/h1) *
       ↪  Ke((Y1[k]-Y1)/h2))
      help2<-c(Ke(c(sum(c(1,argumn) * Xv[k,]) -
       ↪  colSums(c(1,argumn) * t(Xv)))/h1))
      term1<-sum(help1[-k])/(h2*sum(help2[-k]))

      if(sum(help1[-k])==0){term1<-0}
      logDensity<-log(max(term1,c1))
      Likelihood[k]<-logDensity
    }
    return(sum(Likelihood))
  }

  # Optimal parameters
  theta<-c(1,par[2:dimX+1])
  theta<-as.matrix(theta)

  h1<-par[1]
  h2<-par[2]
  h1
  h2

  # Log-likelihood optimal value
  logLik<-likeli(dataX,Y,theta,h1,h2)

  # Conditioned density
  condens<-function(argumn,h1,h2){
    argumn<-argumn[-1]
    cdens<-rep(0,times=nn)
    for(k in 1:nn){

      # to define conditional density and survival function we
       ↪  need following help functions

      help1<-c(Ke(c(sum(c(1,argumn) * Xv[k,]) -
       ↪  colSums(c(1,argumn)*t(Xv)))/h1) * Ke((Y1[k]-Y1)/h2))
      help2<-c(Ke(c(sum(c(1,argumn) * Xv[k,]) -
       ↪  colSums(c(1,argumn) * t(Xv)))/h1))
      dens<-sum(help1[-k])/(h2*(sum(help2[-k]+c1)))
      cdens[k]<-(max(c1,dens))
    }
    return(cdens)
  }

  # Functions to estimate variance-covariance matrix
  ere<-function(argumn,h1,h2){
    argumn<-argumn[-1]
    ss<-rep(0,times=nn)
    for(k in 1:nn){
      help3<-c(Ke(c(sum(c(1,argumn) * Xv[k,])
      - colSums(c(1,argumn) * t(Xv)))/h1) * Ke((Y1[k]-Y1)/h2))
      ss[k]<-sum(help3[-k])/((nn-1)*h1*h2)
    }
    return(ss)
  }

  ese<-function(argumn,h1,h2){
    argumn<-argumn[-1]
    ss<-rep(0,times=nn)
    for(k in 1:nn){
      help3<-c(Ke(c(sum(c(1,argumn)*Xv[k,])
      -colSums(c(1,argumn)*t(Xv)))/h1))
      ss[k]<-sum(help3[-k])/((nn-1)*h1)
    }
    return(ss)
  }

  nn<-n
  ddere<-function(argumn,h1,h2,i){
    argumn<-argumn[-1]
    ss<-array(0,dim=c(length(argumn),length(argumn),nn))
    for(h in 1:nn){
      help5<-t(t(Xv[i,]-Xv[h,])) %*% (Xv[i,]-Xv[h,]) *
       ↪  Ke((Y1[i]-Y1[h])/h2) *
       ↪  ddKe(c(sum(c(1,argumn)*Xv[i,]) - sum(c(1,argumn) *
       ↪  Xv[h,]))/h1)
      ss[,,h]<-help5[2:dimX,2:dimX]
    }
    rr<-apply(ss[,,-i],1:2,sum)/((nn-1)*(h1**3)*h2)
    return(rr)
  }

  ddese<-function(argumn,h1,h2,i){
    argumn<-argumn[-1]
    ss<-array(0,dim=c(length(argumn),length(argumn),nn))
    for(h in 1:nn){
      help5 <- t(t(Xv[i,]-Xv[h,])) %*% (Xv[i,]-Xv[h,]) *
       ↪  ddKe(c(sum(c(1,argumn)*Xv[i,]) -
       ↪  sum(c(1,argumn)*Xv[h,]))/h1)
      ss[,,h]<-help5[2:dimX,2:dimX]
    }
    rr<-apply(ss[,,-i],1:2,sum)/((nn-1)*(h1**3))
    return(rr)
  }
  dere<-function(argumn,h1,h2,i){
    argumn<-argumn[-1]
    ss<-matrix(0,nn,length(argumn))
    for(h in 1:nn){
```

```
    help6 < -(Xv[i,]-Xv[h,]) * Ke((Y1[i]-Y1[h])/h2) *
     ↪  dKe(c(sum(c(1,argumn) * Xv[i,]) -
     ↪  sum(c(1,argumn)*Xv[h,]))/h1)

    ss[h,]<-help6[2:dimX]
  }
  rr<-colSums(ss[-i,])/((nn-1)*(h1**2)*h2)
  return(rr)
}

dese<-function(argumn,h1,h2,i){
  argumn<-argumn[-1]
  ss<-matrix(0,nn,length(argumn))
  for(h in 1:nn){
    help7<-(Xv[i,]-Xv[h,]) * dKe(c(sum(c(1,argumn)*Xv[i,]) -
     ↪  sum(c(1,argumn)*Xv[h,]))/h1)

    ss[h,]<-help7[2:dimX]
  }
  rr<-colSums(ss[-i,])/((nn-1)*(h1**2))
  return(rr)
}


# Ploting index
index<-dataX%*%theta
par(mfrow=c(1,1))
plot(density(index))
Xv<-dataX
Y1<- (((Y+l12[2])**l12[1])-1)/l12[1]

# Calculating variance-covariance matrix
fderiv<-matrix(0,nn,dimX-1)
sderiv<-array(0,dim=c((dimX-1),(dimX-1),nn))

si<-ese(theta,h1,h2)
ri<-ere(theta,h1,h2)

for(m in 1:nn){
  fderiv[m,]<-(dere(theta,h1,h2,m) * si[m]-ri[m] *
   ↪  dese(theta,h1,h2,m)) / ((si[m]**2)+c1)
  sderiv[,,m]<-(ddere(theta,h1,h2,m) / (ri[m]+c1)) -
   ↪  ((t(t(dere(theta,h1,h2,m))) %*% dere(theta,h1,h2,m)) /
   ↪  ((ri[m]**2)+c1)) -
    (ddese(theta,h1,h2,m) / (si[m]+c1)) +
     ↪  ((t(t(dese(theta,h1,h2,m))) %*% dese(theta,h1,h2,m))
     ↪  / ((si[m]**2)+c1))
    }

    fderiv<-fderiv/condens(theta,h1,h2)
    Sigma1<-(1/nn)*t(fderiv)%*%fderiv
    Itheta=apply(sderiv,1:2,sum)/nn
    Sigma2=solve(Itheta)

    # Variance-covariance matrix of parameters
    Sigma<-Sigma2%*%Sigma1%*%t(Sigma2)/n

    # Standard errors
    STE<-sqrt(diag(Sigma))

    # Z statistic for normal asymptotic inference on index
     ↪  parameters
    z<-theta[-1]/STE
    STE
    z
    # P-values
    pval<-1-pnorm(abs(z))
    pval
    # RESULTS
    print("Estimation Results")
    Results<-cbind(theta,c(0,STE),c(0,z),c(0,pval))
    # Results<-t(Results)
    colnames(Results)<-c('Coefficients','STE','Z','p-value')
    print(knitr::kable(Results,digits=6, caption="Estimation
     ↪  Results"))
  }
```

In order to execute the code above, we just have to call the function `ComputeModel` specifing the value for the four parameters function: `data_file`, that contains the path of the .csv file with the data, `transform` that can be set to "none", "log", or "boxcox", depending on the desired response transformation, `remove_extreme` that if set to TRUE takes away the extreme value discussed in Chapter 4, and `selected_variables` that can be set to "all", "telematics", or "no_telematics". The following snippet of code executes all the combinations discussed in this work.

```
remove_extreme_types = c(FALSE,TRUE)
transform_types = c("none","log","boxcox")
selected_variables_types =
 ↪  c("all","telematics","no_telematics")
```

```
for(rr in remove_extreme_types)
{
  for (tt in transform_types)
  {
    for (sv in selected_variables_types)
    {
      print(paste("remove_extreme: ",rr,", transform_type:
       ↪  ",tt,", selected_variables_type: ",sv, sep=""))
      ComputeModel("data_cost.csv", tt, rr, sv)
    }
  }
}
```