DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

ROBUST NONLINEAR REGRESSION METHODS

by

Burak DİLBER

July, 2019

İZMİR

ROBUST NONLINEAR REGRESSION METHODS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Master of Science in
Statistics, Statistics Program

by
Burak DİLBER

July, 2019
İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"ROBUST NONLINEAR REGRESSION METHODS"** completed by **BURAK DİLBER** under supervision of **ASSOC. PROF. DR. A. FIRAT ÖZDEMİR** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. A. Fırat Özdemir

Supervisor

Assoc. Prof. Dr. Hafız Savaş SARAÇ

(Jury Member)

Asst. Prof. Dr. Engin YILDIZTEPE

(Jury Member)

Prof.Dr. Kadriye ERTEKİN

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

I would like to express my special thanks to my supervisor Assoc. Prof. Dr. A. Fırat ÖZDEMİR for his important advices, helpful comments, patience and continuous support since the beginning of my study.

I would like to acknowledge my dissertation committee members, Assoc. Prof. Dr. Hakan Savaş SAZAK and Asst. Prof. Dr. Engin YILDIZTEPE, for their constructive comments and suggestions.

I would like to thank to Gözde NAVRUZ for her helpful, constructive comments and encouring support with my most sincere feelings.

I am also grateful to Sezer BAYSAL, Sami AKDENİZ and Hasan SAĞIROĞLU for their patience, support and positive manners during the arduous research process.

This thesis is dedicated to my parents İ. Yaşar and Zeynep DİLBER for their love, endless support and encouragement. I also would like to thank to my brothers Cüneyt and Ertuğrul DİLBER for their moral and support throughout this research.

Burak DİLBER

ROBUST NONLINEAR REGRESSION METHODS

ABSTRACT

Regression analysis is a statistical method for modelling the relationship between two or more variables. Ordinary least squares regression, which is the most commonly used approach to determine relationship between variables may be misleading in the presence of outliers, or when there is heteroscedasticity and non-normality. Even if the underlying assumptions such as normality and homoscedasticity are hold, the relationship between dependent variable and independent variable(s) may be nonlinear. In such cases, using nonparametric regression methods is more appropriate since the shape of the regression function is not needed to be predefined and there is no significant assumptions as in parametric regression situation.

The nonparametric regression estimators are called as smoothers and in this thesis four of them are investigated: Kernel smoothing, locally weighted scatter plot smoothing (LOWESS), the running interval smoother (RIS) and constrained b-spline smoothing (COBS). While the running interval smoother predicts the dependent variable by using different location estimators, COBS predicts by using quantiles and the other methods predict the dependent variable using weighted mean.

Robust nonlinear regression methods are also blended to create alternative methods. The smoothers and these alternative methods are compared with a simulation study by using theoretical distributions. Furthermore, the methods are examined graphically to understand how the methods can model the relationship between variables. The predicted values of dependent variable corresponding to new observations are calculated as well. COBS and RIS with NO estimator outperformed the other methods in terms of mean squared error (MSE).

Keywords: Robust nonlinear regression, nonparametric regression, quantile estimators, the running interval smoother, constrained b-spline smoothing

DAYANIKLI DOĞRUSAL OLMAYAN REGRESYON YÖNTEMLERİ

ÖZ

Regresyon analizi iki veya daha fazla değişken arasındaki ilişkiyi modellemek için kullanılan istatistiksel bir yöntemdir. Değişkenler arasındaki ilişkiyi tanımlamak için kullanılan en yaygın yaklaşım olan en küçük kareler regresyonu, aykırı değerler, heterojen varyanslılık ve normal olmayan dağılım durumlarında yanıltıcı olabilir. Homojenlik ve normallik gibi temel varsayımlar sağlansa bile bağımlı ve bağımsız değişkenler arasındaki ilişki doğrusal olmayabilir. Bu durumlarda önceden tanımlı olmayan regresyon fonksiyonun şekline ve parametrik regresyon durumundaki varsayımlara ihtiyaç duyulmadığı için parametrik olmayan regresyon yöntemlerini kullanmak daha uygun olacaktır.

Parametrik olmayan regresyon yöntemleri düzleştiriciler olarak isimlendirilir ve bu tezde dört tane yöntem incelenmiştir: Çekirdek düzleştirici, yerel olarak ağırlıklandırılmış dağılım grafiği düzleştirici (LOWESS), hareketli aralık düzleştirici (RIS) ve kısıtlı b-spline düzleştirici (COBS). Hareketli aralık düzleştirici farklı yerel kestiriciler ve COBS kantiller kullanarak bağımlı değişkeni tahmin ederken diğer yöntemler ağırlıklı ortalama kullanarak bağımlı değişkeni tahmin eder.

Dayanıklı doğrusal olmayan regresyon yöntemleri harmanlanarak alternatif yöntemler oluşturulmuştur. Düzleştiriciler ve bu alternatif yöntemler teorik dağılımlar kullanılarak bir simülasyon çalışması ile karşılaştırılmıştır. Ayrıca yöntemler değişkenler arasındaki ilişkinin nasıl modellediğini anlamak için grafiksel olarak incelenmiştir. Buna ek olarak yeni gözlemlere karşılık gelen bağımlı değişkenin tahminlenen değerleri hesaplanmıştır. COBS ve NO kestiricisi ile kullanılan RIS, hata kareler ortalaması (MSE) açısından diğer yöntemlerden daha iyi performans göstermektedir.

Anahtar Kelimeler: Dayanıklı doğrusal olmayan regresyon, parametrik olmayan regresyon, kantil kestiriciler, hareketli aralık kestiricisi, kısıtlı b-spline düzleştiricisi

CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER ONE
INTRODUCTION

Regression analysis is commonly used to determine the relationship between variables. It is based on the nineteenth century and it shows the conditional distribution of the dependent variable for certain values of the independent variable(s). Regression analysis is applied in order to determine the relationship between various variables in various fields such as social sciences, medicine, engineering, science and educational sciences.

Regression analysis is divided into two groups as parametric and non-parametric regression. The most important feature of parametric regression is that the shape of regression function is known in advance. It is generally based on the least squares estimation method and for all values of the independent variable, it is necessary to provide some important assumptions such as the error variances are constant, no autocorrelation between the error terms, the error terms conform to the normal distribution, and if there is multiple regression analysis, there is no multicollinearity between the independent variables. In the absence of these assumptions, inferences about regression function do not fit the truth and the results might be misinterpreted.

Non-parametric regression is used to explain the nonlinear relationship between variables. In the nonparametric regression analysis, the shape of the function is not pre-defined and there are no significant assumptions as in parametric regression analysis. The only assumption is that the mean of error terms is zero and the variance is a finite number. In this way, flexibility is provided to reveal the relationship between variables. In addition to all these advantages, there are some difficulties in the application of nonparametric regression. First of all, a certain number is not given, but a large data set is needed. It also requires intensive computer use. Compared in terms of efficiency; it should be kept in mind that nonparametric regression is less effective than parametric regression analysis.

The estimators used in the nonparametric regression are called smoothers. The basic idea of smoothing is the use locally weighted mean, the estimated value of the dependent variable at a given point x is determined by taking a weighted mean of points in the neighborhood of x. In nonparametric regression, there are methods used to find the locally weighted mean. In this thesis, the focus is on few methods that appear to have considerable practical value when the focus is on robust measures of location and robust quantile estimator. These methods are kernel smoothing, locally weighted scatter plot smoothing (LOWESS), the running interval smoother and constrained b-spline smoothing (COBS).

In the kernel smoothing method, weights are obtained by using the kernel functions, while the local mean is found. Kernel function should be continuous, bounded and symmetric real function. Changing the kernel function does not affect the result much. In this method, the local neighborhood is called the bandwidth. Bandwidth selection is a very important issue. The bandwidth selection for each kernel function is determined by different techniques.

In the locally weighted scatter plot smoothing method, weights are obtained using the tricube weight function (Cleveland, 1979). Weights are used to estimate the dependent variable with the least squares method. In this method, local neighborhood is called span. The span is between 0 and 1. The span, controls the raggedness of the smooth. As the span value increases, we get a straight line. If there is no certainty about what span value will be, a reasonable start can be made for iteration by taking span=0.5.

In the running interval smoother method, independent variables (x_i), which is close to the point of interest of x, is determined. Then, a conditional measure of location of the y values corresponding to these x_i is computed. This conditional measure of location is estimated using different estimators. In this thesis, it is estimated by using Harrell-Davis (HD) (Harrell & Davis, 1982) and NO (Navruz, 2019) quantile estimators. In this method, local neighborhood is called span. Often the choice span=1

and span=0.8 and gives good results, but both larger and smaller values might be better, particularly when the number of observations is small.

The other non-parametric regression method is constrained b-spline smoothing (COBS). The main principle involves the expansion of smoothing splines with conditional quantile function estimation, a method introduced by Bosch et al. (1995). Constrained b-spline smoothing uses splines. Regression splines are special functions defined by piecewise polynomials. The region defining the pieces is separated by a sequence of knots or breakpoints (Hastie & Tibshirani, 1990). The aim is to force the piecewise polynomials to merge smoothly with the knots. B-splines are a spline function with minimum support according to the given degree and smoothness. On the other hand, this method works based on the estimated function of the constraints. In this method, the smoothing parameter selection is extremely important to check the smoothness.

Nonparametric regression is a method based on advanced mathematical background. For this reason, mathematical theorem and proofs are not given much in order for the thesis to be understandable. In order to be understandable, the non-parametric regression model with one explanatory variable is discussed in explanations and estimates.

Alternative methods are obtained by blending two different nonparametric regression methods. The comparison was made between the alternative methods by using different quantile estimators. First, a simulation study was performed with mean squared error (MSE) criterion and distributions and the performances of the methods were evaluated and compared. Then, comparisons between methods were made by using a real data set. In addition, fitted values were obtained for new observations.

The second chapter explains the basic concepts used in non-parametric regression analysis and non-parametric regression estimators as well. The kernel smoothing and the locally weighted scatter plot smoothing (LOWESS) are described, and the selection of the smoothing parameter is examined in detail. Then, the running interval smoother

method, which might be used with different location and quantile estimators, is introduced. Finally, the constrained b-spline smoothing (COBS) method is given at the end of this chapter.

The third chapter has two parts: design of the simulation study and real data example. The first part is about how the study design was set with the combinations of non-normality, heteroscedasticity and unequal sample sizes. At the end of this chapter, a real data set example is investigated. R programming language is used in all computations. The R codes used in the application are in the appendices section. The R statistical software can be downloaded from https://cran.r-project.org/bin/windows/base/.

The last chapter concludes the findings of the simulation studies and give some suggestions about the use and preference of the nonparametric regression estimators considered in this study.

CHAPTER TWO
NONPARAMETRIC REGRESSION METHODS (SMOOTHERS)

Nonparametric regression deals with the problem of estimating a conditional measure of location related to dependent variable (y) when independent variables are known. It is assumed that this conditional measure of location is given by some unknown function m. The main problem is to estimate the unknown function m.

Methods in which the unknown m function is estimated are generally based on what are called smoothing methods. The smoothing techniques detect points that are close to the $(x_1, x_2, \dots, x_p)$ values from n observations, and the corresponding y values and any conditional measure of location are computed. The estimator $\hat{m}$ called a smoother is $\hat{m}(x_1, x_2, \dots, x_p)$ an estimate of the measure of location associated with *y* at the point$(x_1, x_2, \dots, x_p)$.

The goal is to get the estimation of some conditional measure of location for y given x. Let weights (w_i) be some measure of how close x_i is to the point of interest. Then generally, the estimate of $m(x)$ is taken to be

$$\hat{m}(x) = \sum w_i y_i \tag{2.1}$$

and the goal is to choose the w_i in some reasonable manner.

Let W be a weight function with the following properties:

1. $W(x) > 0$ $\quad for\ |x| < 1,$
2. $W(-x) = W(x),$
3. $W(x)\ is\ a\ nonincreasing\ function$ $\quad for\ x \geq 0,$
4. $W(x) = 0$ $\quad for\ |x| \geq 1.$

Let $0 < f \leq 1$ and let r be fn rounded to the nearest integer. For each x_i and weights $w_k(x_i)$ are determined by using weight function W for all x_k , $k = 1, \dots, n,$.

This process is done by centering and scaling in x_i point. Thus, the point at which W is the first zero is in the nearest r^{th} intersection of x. The initial fitted value, $\hat{y}_i$, at each x_i point is applied to the data by weighted least squares method using the weights $w_k(x_i)$ is the estimated value of a d^{th} polynomial. A process for computing the initial fitted values is called **locally weighted regression.** Based on the size of the residual obtained with $y_i - \hat{y}_i$, a different weight set δ_i is defined for each (x_i, y_i)**.** Accordingly, large residuals receive in small weights and small residuals receive in large weights. As before, the new fitted values are computed. However, $\delta_i w_k(x_i)$ is used instead of $w_k(x_i)$. The process of computing new weights and new fitted values is repeated several times. The entire procedure, including the initial computation and the iterations, is referred to as **robust locally weighted regression** (Cleveland, 1979).

The model for locally weighted regression is

$$y_i = g(x_i) + \varepsilon_i \tag{2.2}$$

where g is a smooth function and the ε_i are random variables with mean 0 and variance σ^2. Within such a framework, $\hat{y}_i$ is an estimate of $g(x_i)$. The assumption of smoothness makes it possible to be used in forming $\hat{y}_i$ in the neighborhood of (x_i, y_i). The points whose x-axis are close to x_i play a large role in the determination of $\hat{y}_i$, while points far away play a lesser role. Increasing f increases the neighborhood of effective points and hence the smoothness of smoothed points tends to increase.

There are four smoothing methods in the literature. These methods are kernel smoothing, locally weighted scatter plot smoothing (LOWESS), the running interval smoother (RIS) and constrained b-spline smoothing (COBS). Methods are described in this chapter and examples are shown at the end of each method.

2.1 Kernel Smoothing

One of the methods used to estimate the conditional mean of y, given x is known is the kernel smoothing method. In this method weights (w_i) are determined by a kernel function $K(u)$. The kernel function must provide the following requirements;

- continuous,
- bounded,
- symmetric.

Let the kernel $K(u)$ be in this requirements such that

$$\int K(u)du = 1. \tag{2.3}$$

.

2.1.1 Choice of Kernel Function

Some of the functions used for this method are shown in Table 2.1. The most preferred among these functions is the Epanechnikov kernel.

Table 2.1 Kernel functions

Kernel Function	K(u)	Efficiency
Epanechnikov	$\frac{3}{4}(1-u^2)$ $\quad \lvert u\rvert \leq 1$	1
Biweight (Quartic)	$\frac{15}{16}(1-u^2)^2$ $\quad \lvert u\rvert \leq 1$	0.9939
Triangular	$1-\lvert u\rvert$ $\quad \lvert u\rvert \leq 1$	0.9859
Gaussian	$\frac{1}{\sqrt{2\pi}}e^{(-\frac{1}{2}u^2)}$ $\quad \lvert u\rvert \leq 1$	0.9512
Rectangular (Uniform, Box)	$\frac{1}{2}$ $\quad \lvert u\rvert \leq 1$	0.9295

The distribution patterns of these functions are shown in Figure 2.1. The common feature of these functions is that they are bounded, symmetric and continuous.

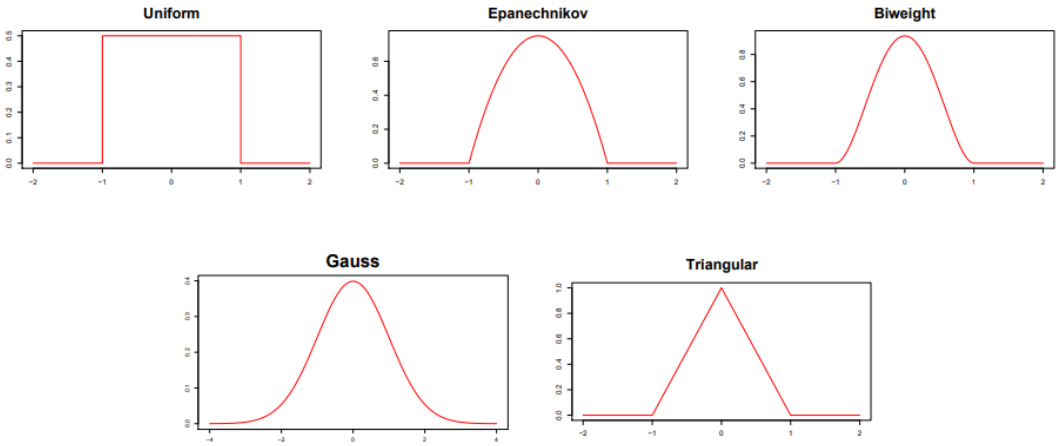

Figure 2.1 Kernel functions

2.1.2 Kernel Regression

The unknown function $m(x)$ can be estimated with different estimators. These estimators are Nadaraya – Watson kernel estimator, Gasser – Müller kernel estimator, linear interpolation and k nearest – neighbour estimator.

2.1.2.1 Nadaraya – Watson (N-W) Kernel Estimator

The most preferred estimator of unknown $m(x)$ is the Nadaraya-Watson. For this estimator, weights are calculated first and then the unknown function $m(x)$ is estimated.

An estimate of $m(x)$ where

$$w_i = \frac{K\left(\frac{x - x_i}{h}\right)}{\sum K\left(\frac{x - x_i}{h}\right)} \tag{2.4}$$

and h is the non-negative bandwidth which controls the size of the local neighborhood.

The Nadaraya-Watson estimator that estimates the unknown $m(x)$ function is defined by:

$$\widehat{m}(x) = \sum \frac{K\left(\frac{x - x_i}{h}\right)}{\sum K\left(\frac{x - x_i}{h}\right)} y_i. \quad (2.5)$$

Nadaraya-Watson estimator is often used in conjunction with Epanechnikov and Gaussian kernel functions (Nadaraya, 1964).

2.1.2.2 Gasser – Müller (G-M) Kernel Estimator

Gasser-Müller (G-M) estimator is also known as integral kernel estimator. According to this estimator, the sum of the weights is 1, so there is no need for denominator. Assuming that the data is sorted according to the variable x, the following estimation is obtained.

G-M kernel estimator is defined by:

$$\widehat{m}(x) = \sum \int_{s_{i-1}}^{s_i} K(\frac{u - x}{h})\, du\, y_i \quad (2.6)$$

$$s_i = \frac{x_i + x_{i+1}}{2} \quad (2.7)$$

The Gasser-Müller estimator is a robust estimate against outliers.

Compared to N-W and G-M estimators, the N-W estimator has a significant bias, particularly in the boundary regions. The G-M estimator reduces this bias but increases the variance (Gasser & Müller, 1979).

2.1.2.3 Linear Interpolation

This method uses the triangular kernel function to estimate the function of $m(x)$. The linear interpolation method is a family of b-spline estimators. This method estimates the linear b-spline regression model.

An estimate of $m(x)$ where

$$w_i = K\left(\frac{x - \frac{i}{n}}{\frac{1}{n}}\right). \tag{2.8}$$

The linear interpolation that estimates the unknown $m(x)$ function is defined by:

$$\widehat{m}(x) = \sum K\left(\frac{x - \frac{i}{n}}{\frac{1}{n}}\right) y_i = \sum B_{1,i}(x)\, y_i. \tag{2.9}$$

2.1.2.4 K Nearest – Neighbour (K-NN) Estimator

The estimation by the kernel regression method is defined as a weighted average of the y values in a fixed neighborhood of x, and the distance (width) of this neighboor is determined by the bandwidth. The k-NN method takes a weighted average of y values corresponding to x in the estimation, but the significant difference here is that this distance is variable. In other words, at a point x, the function of $m(x)$ is to be estimated; the average of y values for the x observation of the closest to this point is calculated.

K-NN estimator is used in conjunction with triangular kernel function.

An estimate of $m(x)$ where

$$w_i = \frac{K\left(\frac{x - x_i}{h}\right)}{\sum K\left(\frac{x - x_i}{h}\right)} \tag{2.10}$$

and h is the non-negative bandwidth which controls the size of the local neighborhood (Hardle, 1994).

K - NN estimator defined by:

$$\widehat{m}(x) = \sum \frac{K\left(\frac{x - x_i}{h}\right)}{\sum K\left(\frac{x - x_i}{h}\right)} y_i. \quad (2.11)$$

2.1.3 Bandwidth Selection

The most commonly used method is cross-validation (CV). The basic goal behind cross-validation (CV) is to hold part of the sample to assess the performance of an estimator. In here, n estimation errors are summarized in least squares CV by:

$$CV(h) = \frac{1}{n} \sum [y_i - \widehat{m}(x)]^2. \quad (2.12)$$

Estimation of the CV for bandwidth is calculated by minimizing the function.

The Table 2.2 shows how bandwidth is calculated for the Epanechnikov and Gaussian kernel functions.

Table 2.2 Bandwidth selection of kernel functions

$h = \min(s, \frac{IQR}{1.34})$	(Epanechnikov Kernel Function)
$h = \left(1.06 * \min\left(s, \frac{IQR}{1.34}\right)\right) * n^{\frac{1}{5}}$	(Gaussian Kernel Function)

2.1.4 R Function "kerreg"

"kerreg" is applied via the R package "WRS2".

The R function

library("WRS2")

"kerreg(x,y,pyhat=FALSE,pts=NA,plotit=TRUE,theta=50,phi=25,expand=.5,

scale=FALSE,zscale=FALSE,eout=FALSE,xout=FALSE,outfun=out,np=100,xlab="X",ylab="Y",zlab="Z",varfun=pbvar,e.pow=TRUE,pr=TRUE,ticktype="simple",pch='.',...)".

This function computes locally weighted regression with Epanechnikov kernel. If the argument pyhat=T, the function returns $\hat{m}(x_i)$ values. If eout = T, the function removes any outliers in the observations (x_i, y_i) according to the method specified in the outfun parameter. If xout = T, only the outliers in x_i values are removed. Set plotit = F to suppress the plot. The arguments theta and phi can be used to rotate a three-dimensional plot. The xlab, ylab and zlab arguments are used to determine the labels on the x-axis, y-axis, and z-axis. The np argument determines how many x values are defined when creating the smoothness.

2.1.5 "predict" Function for Kernel Smoothing

"predict" function is used to estimate new observations. The R function

"predict(object, newdata = NULL)"

where the argument object is fitted by kernel smoothing. The new data used to obtain predictions.

2.1.6 An Example

Kernel smoothing with Epanechnikov is examined graphically to understand how the methods can model the relationship between variables. The data set is from article by Joyner at al. "Peak horizontal acceleration and velocity from strong-motion records including records from the 1979 Imperial Valley, California earthquake. (1981)". This data gives peak accelerations measured at various observation stations for 23 earthquakes in California. The data have been used by various workers to estimate the attenuating affect of distance on ground acceleration. Dependent variable is station – hypocenter distance (km) and independent variable is peak acceleration (g).

The plot is shown in Figure 2.2.

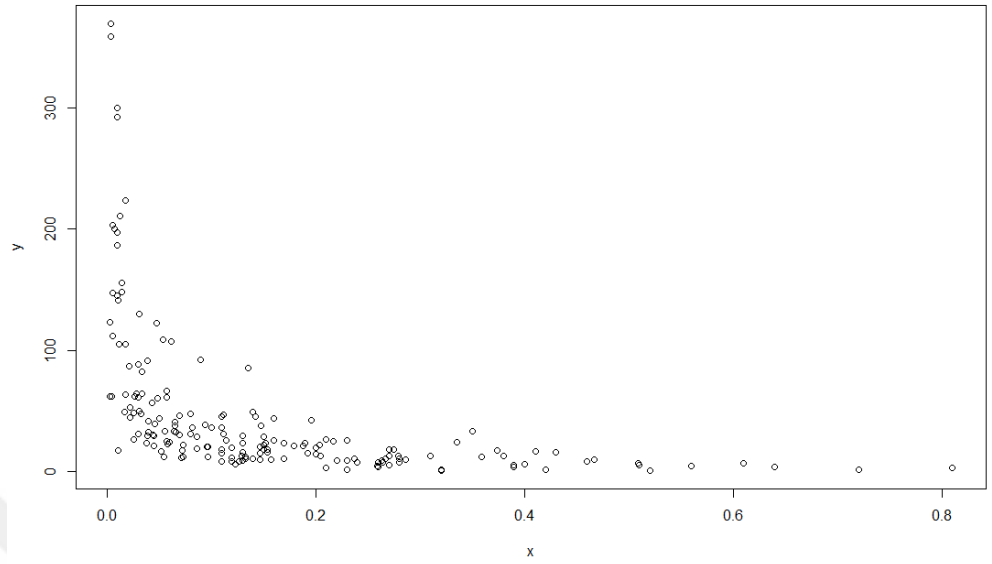

Figure 2.2 Plot of the data set

The line for the case where "Epanechnikov Kernel" is shown in Figure 2.3.

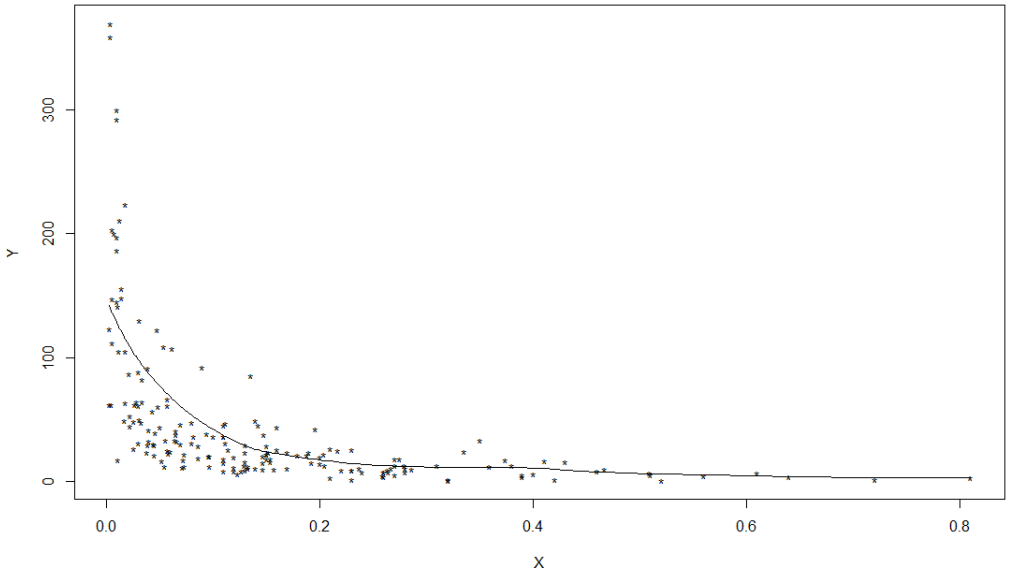

Figure 2.3 The regression line of Epanechnikov kernel

The line for the case where "Epanechnikov Kernel" and ”xout=TRUE” is shown in Figure 2.4.

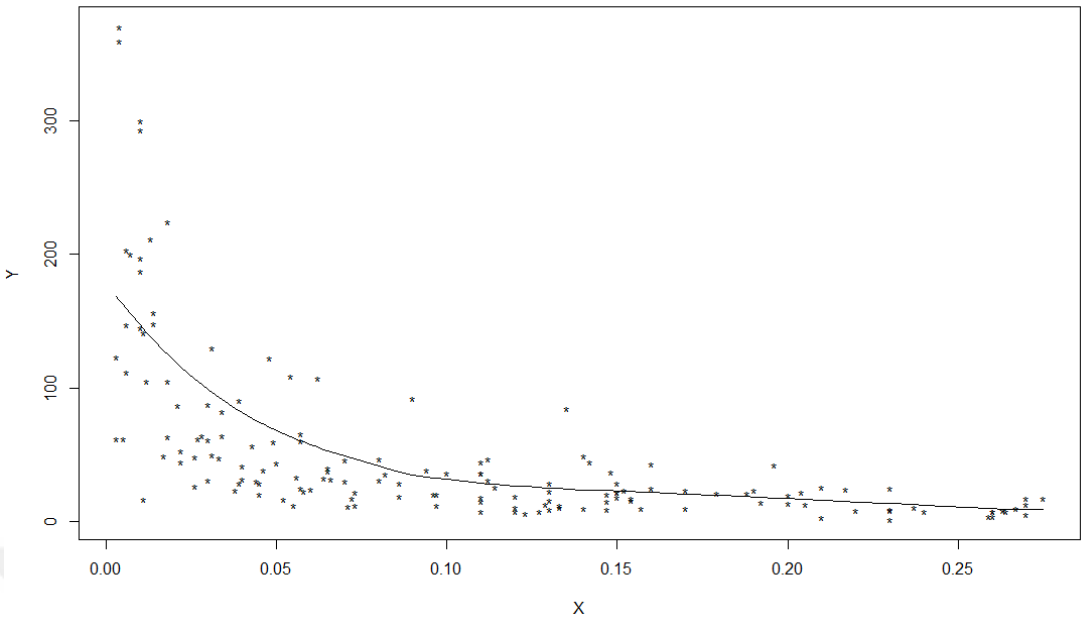

Figure 2.4 The regression line of Epanechnikov kernel with x_i outliers

The line for the case where "Epanechnikov Kernel" and ”eout=TRUE” is shown in Figure 2.5.

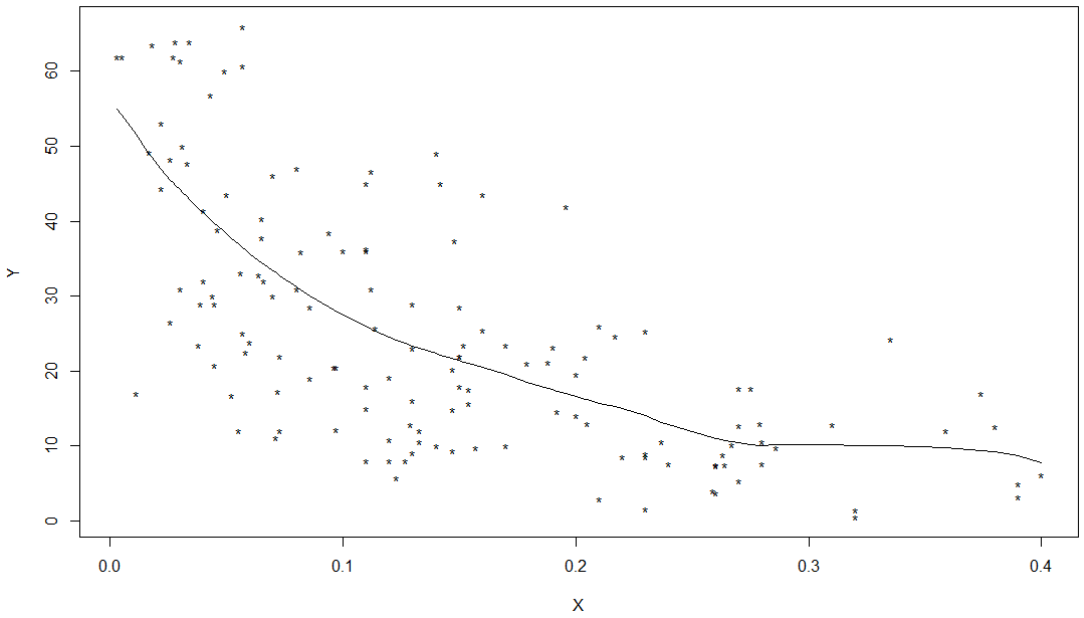

Figure 2.5 The regression line of Epanechnikov kernel with x_i and y_i outliers

2.2 Locally Weighted Scatter Plot Smoothing (LOWESS)

Let h_i be the r^{th} smallest number among $|x_i - x_j|$, for $j = 1, \dots, n$. For $k = 1, \dots, n$, let

$$w_k(x_i) = W\left(h_i^{-1}(x_k - x_i)\right). \tag{2.13}$$

The **tricube weight function**

$$W(x) = \begin{cases} (1 - |x|^3)^3, & for\ |x| < 1 \\ 0, & for\ |x| \geq 1 \end{cases} \tag{2.14}$$

Locally weighted regression and robust locally weighted regression are defined by the sequential process as follows:

1. For each i compute the estimates, $\hat{\beta}_j(x_i)$, $j = 0, \dots . d$, of the parameters in a polynomial regression of degree d of y_k on x_k, which is fit by weighted least squares with weight $w_k(x_i)$ for (x_k, y_k). Thus the, $\hat{\beta}_j(x_i)$ are the values of β_j that minimize

$$\sum_{k=1}^{n} w_k(x_i)\,(y_k - \beta_0 - \beta_1 x_k - \cdots - \beta_d x_k^d)^2. \tag{2.15}$$

The smoothed point at x_i using locally weighted regression of degree d is $(x_i, \hat{y}_i)$, where $\hat{y}_i$ is the fitted value of the regression at x_i. Thus

$$\hat{y}_i = \sum_{j=0}^{d} \hat{\beta}_j(x_i)\, x_i^j = \sum_{k=1}^{n} r_k(x_i) y_k \tag{2.16}$$

where $r_k(x_i)$ does not depend on y_j, j = 1, ..., n. We have used the notation "$r_k(x_i)$" to remind us that these are the coefficients for the y_k that arise from the regression.

2. Let B be the **bisquare weight function** that is defined by

$$B(x) = \begin{cases} (1-x^2)^2, & for\ |x| < 1 \\ 0, & for\ |x| \geq 1 \end{cases}. \tag{2.17}$$

Let

$$e_i = y_i - \hat{y}_i \tag{2.18}$$

be the residuals from the current fitted values. Let s be the median of the $|e_i|$. Define robustness weights by

$$\delta_k = B\left(\frac{e_k}{6s}\right). \tag{2.19}$$

3. New $\hat{y}_i$ values are calculated for each i by fitting a d^{th} degree polynomial using weighted least squares with weight $\delta_k w_k(x_i)$ at (x_k, y_k).
4. Steps 2 and 3 are repeated a total of t times. The $\hat{y}_i$ values obtained are robust locally weighted regression fitted values.

After the robustness weights δ_k is determined, the fitted value at x can be calculated by fitting a polynomial using the weights $\delta_k w_k(x_i)$. Thus the fitted values could, for example, be calculated and plotted at an equally spaced set of points on the horizontal axis (Cleveland, 1979).

2.2.1 LOESS

The loess fitting method is defined as $\hat{y}_i$, which is the estimate of g for a specific value of x. The smoothness of the loess fit is related to the specification of the degree property of the neighboring parameter and the specification of conditional parametric estimators.

Let $\Delta_i(x)$ be the Euclidean distance of x to x_i; in the space of the estimators. Let $\Delta_i(x)$ be the values of these distances ordered from smallest to largest, and let

$$W(u) = \begin{cases} (1-u^3)^3, & for\ 0 \le u < 1 \\ 0, & for\ u \ge 1 \end{cases} \quad (2.20)$$

be the **tricube weight function.**

Weight is defined for (x_i, y_i) by

$$w_k(x_i) = W\left(\frac{\Delta_i(x)}{\Delta_q(x)}\right) \quad (2.21)$$

where $\Delta_q(x)$ is the largest $\Delta_i(x)$ value among the retained points (Cleveland and Loader, 1981).

There are three items that the user must select in order to carry out robust locally weighted regression: d, the order of the polynomial that is locally fit to each point on the scatterplot; t, the number of iterations of the robust fitting procedure; and f, the parameter used to determine the amount of smoothing.

2.2.2 Choosing "d"

Choosing d to be 1 appears to strike a good balance between ease of calculation and the need for flexibility to increase molds in the data. The case $d = 0$ is the simplest, computationally, however, the case d=1 gives better results. Because the slope is related to the case d = 1. For $d = 2$, the computational considerations begin to eliminate the need for flexibility. Taking $d = 1$ always provides sufficient smoothed points and ease of calculation.

2.2.3 Choosing "t"

The purpose of performing robust iterations is to determine the convergence criteria and iterates until the criterion is satisfied. Many studies have shown that two iterations are sufficient for almost all situations.

2.2.4 Choosing "f"

It is of great importance to determine the value of f in order to obtain the most likely value that will minimize the variability in the smoothed points without disturbing the structure of the data. f is a span value. The span, f, controls the raggedness of the smooth. As the f value increases, we get a straight line. If there is no certainty about what f will be, a reasonable start can be made for iteration by taking f = 0.5.

2.2.5 R Functions "lowess", "loess", "loess.smooth", "scatter.smooth", "lplot"

2.2.5.1 "lowess" Function

The R function

"lowess(x,y,f=2/3,iter=3)"

computes LOWESS where the argument span is f. The span, defaults to 2/3. Scatter plot of points can be create with R commands

"plot(x,y)
lines(lowess(x,y))".

Normally a local linear polynomial fit is used.

2.2.5.2 "loess" Function

The loess function can be used for one or more estimators. The R function

"loess(formula, data, model = FALSE, span = 0.75, degree = 2,
parametric = FALSE, drop.square = FALSE, normalize = TRUE, ...)"

where the argument span. The span, defaults to 0.75. Parameter of degree, defaults to 2.

2.2.5.3 R Functions "loess.smooth" and "scatter.smooth"

R functions plot and add a smooth curve computed by loess on a scatter plot.

"loess.smooth(x, y, span = 2/3, degree = 1,
family = c("symmetric", "gaussian"), evaluation = 50, ...)"

"scatter.smooth(x, y = NULL, span = 2/3, degree = 1,
family = c("symmetric", "gaussian"), xlab = NULL, ylab = NULL,
evaluation = 50, ...)"

2.2.5.4 "lplot" Function

"lplot" is applied via the R package "WRS2".

The R function

library("WRS2")

"lplot(x,y,span=.75,pyhat=F,eout=F,xout=F,outfun=out,plotit=T,
expand=.5,low.span=2/3,varfun=pbvar,
scale=F,xlab="X",ylab="Y",zlab="",theta=50,phi=25)",

where the argument span is low.span If eout = T is determined, this function eliminates any outlier values among the (x, y). Outlier values are determined by the argument outfun. If xout=T is determined, the function removes leverage points among the x_i values only. The arguments theta and phi are used to rotate a three-dimensional plot.

2.2.6 "predict" Function for LOWESS

"predict" function is used to estimate new observations. The R function

"predict(object, newdata = NULL, se = FALSE,
na.action = na.pass, ...)"

where the argument object is fitted by loess. The argument newdata is an optional data frame in which to look for variables with which to predict, or a matrix or vector containing exactly the variables needs for prediction. If missing, the original data points are used. If se = FALSE, a vector giving the prediction for each row of newdata (or the original data). If se = TRUE, a list containing components.

2.2.7 An Example

LOWESS is examined graphically to understand how the methods can model the relationship between variables using the Joyner-Boore attenuation data.

The line for the case where "span = 0.5" is shown in Figure 2.6.

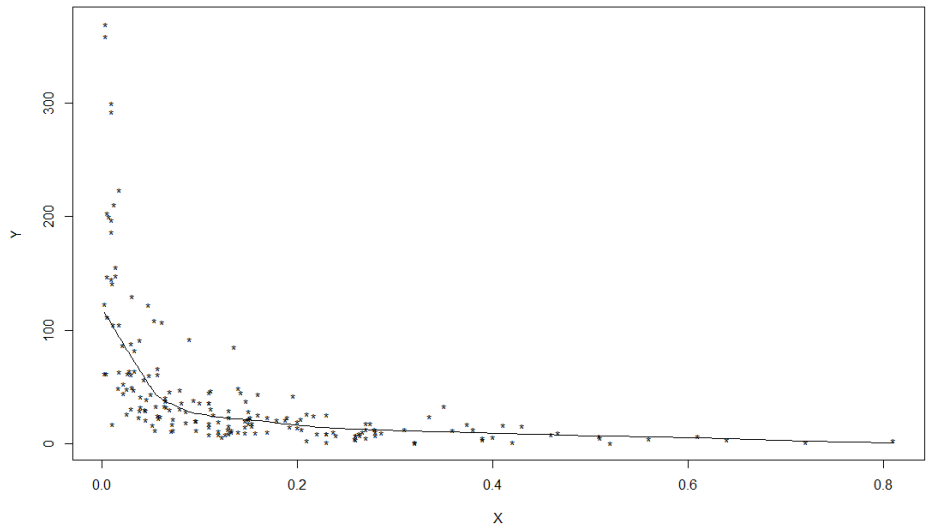

Figure 2.6 The regression line of LOWESS where span=0.5

The line for the case where "span = 0.5" and "xout=TRUE" is shown in Figure 2.7.

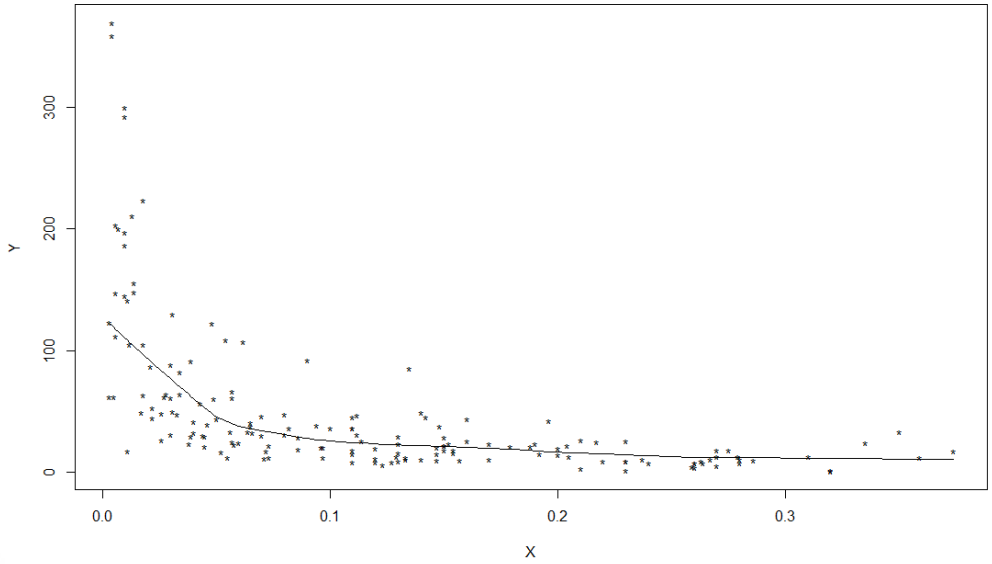

Figure 2.7 The regression line of LOWESS with x_i outliers

The line for the case where "span = 0.5" and “eout=TRUE” is shown in Figure 2.8.

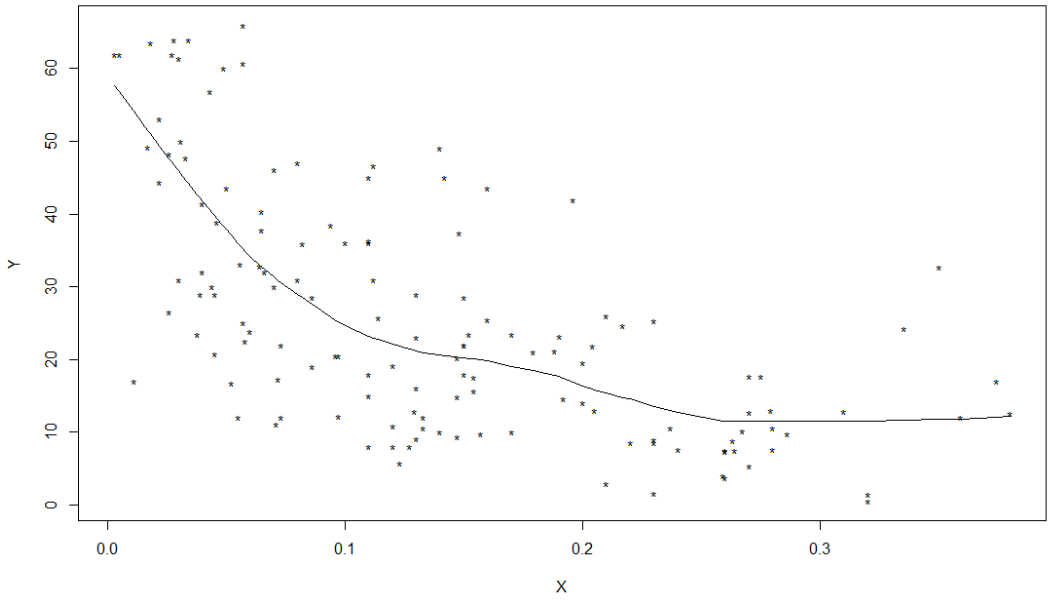

Figure 2.8 The regression line of LOWESS with x_i and y_i outliers

2.3 The Running Interval Smoother (RIS)

Firstly, independent variables (X_i), which is close to the point of interest of x, is determined. Then, a conditional measure of location of the y values corresponding to these x_i is computed. For the running interval smoother, this conditional measure of location is estimated according to an estimator.

The points x_i is said to be close to x if

$$|x - X_i| \leq f * MADN \tag{2.22}$$

where $MADN$ is computed using $x_1, \dots, x_n$ and f is a number between 0 and 1 and plays the role of a span.

$$MADN = \frac{MAD_x}{0.6745} \tag{2.23}$$

To create a line belonging for running interval smoother, x_j values that are close to x_i are first determined, then a measure of location is estimated for the corresponding values of these values and label this result $\hat{y}_i$. So we now have the following n pairs of numbers: $(\hat{x}_1, \hat{y}_1), \dots, (\hat{x}_n, \hat{y}_n)$. The smooth is the line formed by connecting these points (Wilcox, 2012).

The span, f, controls the roughness of the line. As the f value increases a smooth will be a straight, horizontal line. However, if f is too close to zero, the result is a very ragged line. Often the choice $f = 1$ and $f = 0.8$ and gives good results, but both larger and smaller values might be better, particularly when n is small.

Median absolute deviation (MAD) statistic is one of the methods used to determine outliers. To compute it, first sample median is computed, M, subtructed it from every observed value and then taken absolute values. In symbols, compute

$$|x_1 - M|, \dots, |x_n - M| \tag{2.24}$$

The median of the n values just computed is MAD. Its finite sample breakdown point is 0.5.

RIS works with any estimator. These estimators are included in the literature. In addition to these estimators, there is a new estimator recommended. In here, the new quantile estimator is described.

2.3.1 NO Quantile Estimator

With the aim of improving the performance in the lower and upper quantiles especially with small sample sizes, a new quantile estimator which is again a weighted average of all order statistics is introduced (Navruz, 2019):

$$\begin{aligned} NO_q = {} & [B(0;n,q)2q + B(1;n,q)q]X_{(1)} + B(0;n,q)(2-3q)X_{(2)} \\ & - B(0;n,q)(1-q)X_{(3)} \\ & + \sum_{i=1}^{n-2} [B(i;n,q)(1-q) + B(1+1;n,q)q]X_{(i+1)} \\ & - B(n;n,q)qX_{(n-2)} + B(n;n,q)(3q-1)X_{(n-1)} \\ & + [B(n-1;n,q)(1-q) + B(n;n,q)(2-2q)]X_n \end{aligned} \tag{2.25}$$

$B(i;n,q)$, $i = 0,1,2,\dots,n$ are the binomial probabilities with probability of success q and n is the sample size.

2.3.2 R Functions "rplot", "runmean", "rungen", "runhat"

These functions are applied via the R package "WRS2".

2.3.2.1 "rplot" Function

The R function

library("WRS2")

"rplot(x,y, est = tmean, scat = T, fr = NA, plotit = T, pyhat = F, efr = 0.5, theta = 50, phi = 25, scale = F, expand = 0.5, SEED = T, varfun = pbvar, nmin = 0, xout = F, outfun = out, eout = F, xlab = "X", ylab = "Y", zlab = "")"

computes a running-interval smooth. The argument est can be any R function, by default estimator is trimmed mean with 0.2. The argument fr corresponds to the span (f) and defaults to 0.8. If pyhat = T, the function returns the values of $\hat{y}_i$. To avoid the scatterplot, set scat=F. If eout = T, the function eliminates any outliers in the observations (x_i, y_i) according to the method specified in the outfun parameter. If xout = T, only the outliers in x_i values are removed.

2.3.2.2 runmean Function

The function

library("WRS2")

"runmean(x, y, fr=1, tr=.2, pyhat=F, eout=F, outfun=out, xout=F, xlab="x", ylab="y")"

and is used to estimate the trimmed mean of y corresponding to $x_i,\ i = 1, \ldots, n$. The argument fr is the span value and takes the value 1 by default, and the argument tr is the amount of trimming which defaults to 0.2.

2.3.2.3 rungen Function

The rungen function is used to estimate different conditional measure of location associated with y. It has the form

library("WRS2")

"rungen(x,y,est=onestep,fr=1,plotit=T,scat=T,pyhat=F,eout=F,xout=F, outfun=out, …)".

The argument est can be different estimators. If this argument is not specified, it defaults to the modified one-step M estimator. The last argument, ... , can be any additional arguments required by the function est. For example, the command rungen(x,y,est=hd,q=0.75) would result in a running interval smoother that predicts the 0.75 quantile of *y* given *x* with Harrell -Davis quantile estimator. The command rungen(x,y,est=mean,tr=0.3) would result in a running interval smoother based on the 30% trimmed mean.

2.3.2.4 runhat Function

The function

library("WRS2")

"runhat(x,y,pts=x,est=onestep,fr=1, ...)"

is provided for computing $\hat{m}(x)$ for each of the values stored in the argument pts. The argument est defaults to the function one-step M estimator. If, for example, it is desired to compute $\hat{m}(x)$ for *x*=2 and x=4, using a 0.40 quantile of *y* given x with Harrell – Davis quantile estimator, the command runhat(x,y,pts=c(2,4),est=hd,q=0.4) accomplishes this goal.

2.3.3 "predict" Function for RIS

"predict" function is used to estimate new observations. The R function

"predict(object, newdata = NULL, fr=1, est, ...)"

where the argument object is fitted by the running interval smoother. The argument newdata is an optional data frame in which to look for variables with which to predict, or a matrix or vector containing exactly the variables needs for prediction. The argument fr is the span value and takes the value 1 by default. The argument est can

be different estimators. The last argument, ... , can be any additional arguments required by the function est.

2.3.4 An Example

The running interval smoother is examined graphically to understand how the methods can model the relationship between variables using the Joyner-Boore attenuation data.

The quantile regression line for the case where "one step m estimator" and "span = 0.8" is shown in Figure 2.9.

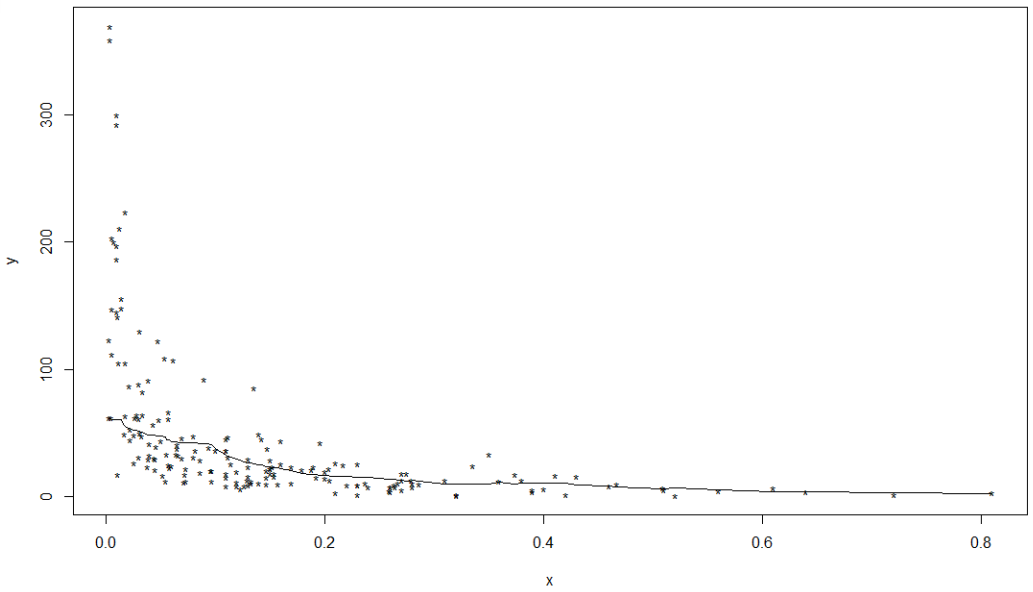

Figure 2.9 The quantile regression line of RIS with one step m estimator

The quantile regression line for the case where "trimmed mean", “tr=0.2” and "span = 0.8" is shown in Figure 2.10.

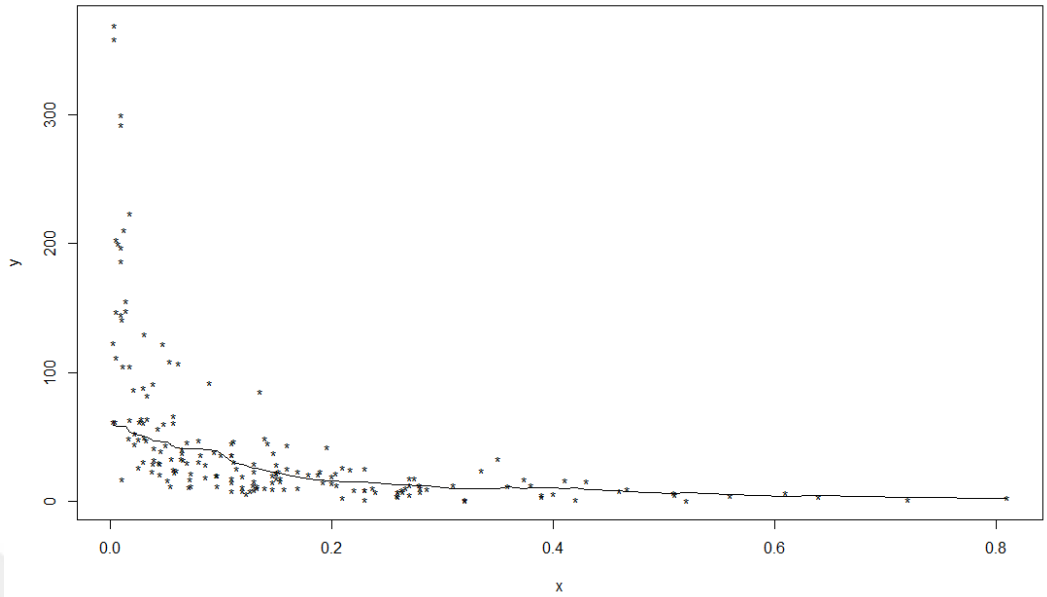

Figure 2.10 The quantile regression line of RIS with trimmed mean

The quantile regression line for the case where "Harrell-Davis quantile estimator", “quantile=0.2” and "span = 0.8" is shown in Figure 2.11.

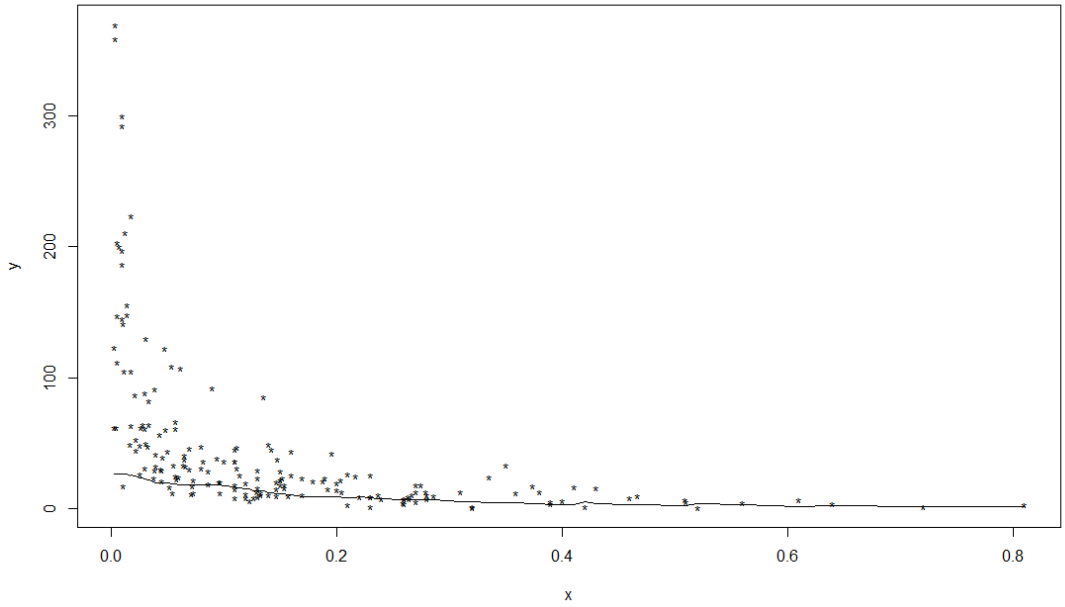

Figure 2.11 The quantile regression line of RIS with HD quantile estimator (q=0.2)

The quantile regression line for the case where "Harrell-Davis quantile estimator", “quantile=0.5” and "span = 0.8" is shown in Figure 2.12.

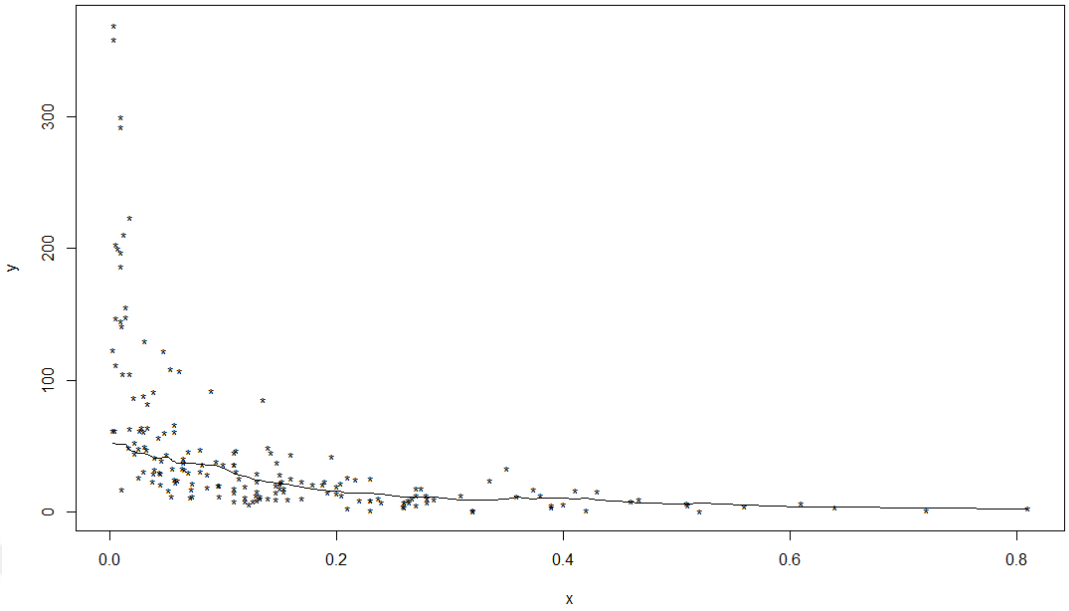

Figure 2.12 The quantile regression line of RIS with HD quantile estimator (q=0.5)

The quantile regression line for the case where "Harrell-Davis quantile estimator", “quantile=0.8” and "span = 0.8" is shown in Figure 2.13.

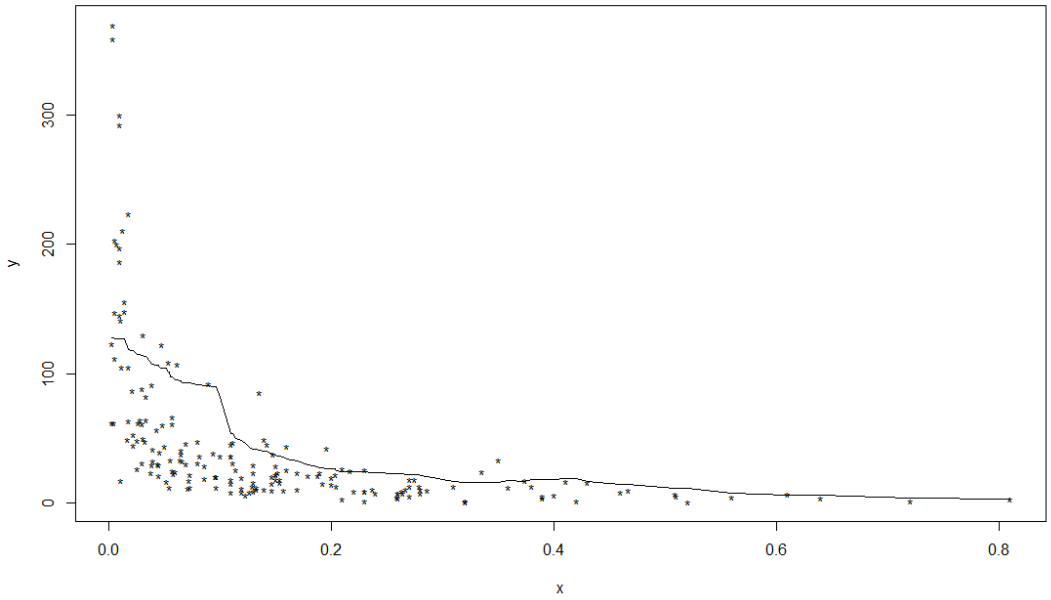

Figure 2.13 The quantile regression line of RIS with HD quantile estimator (q=0.8)

The quantile regression line for the case where "NO quantile estimator", “quantile=0.2” and "span = 0.8" is shown in Figure 2.14.

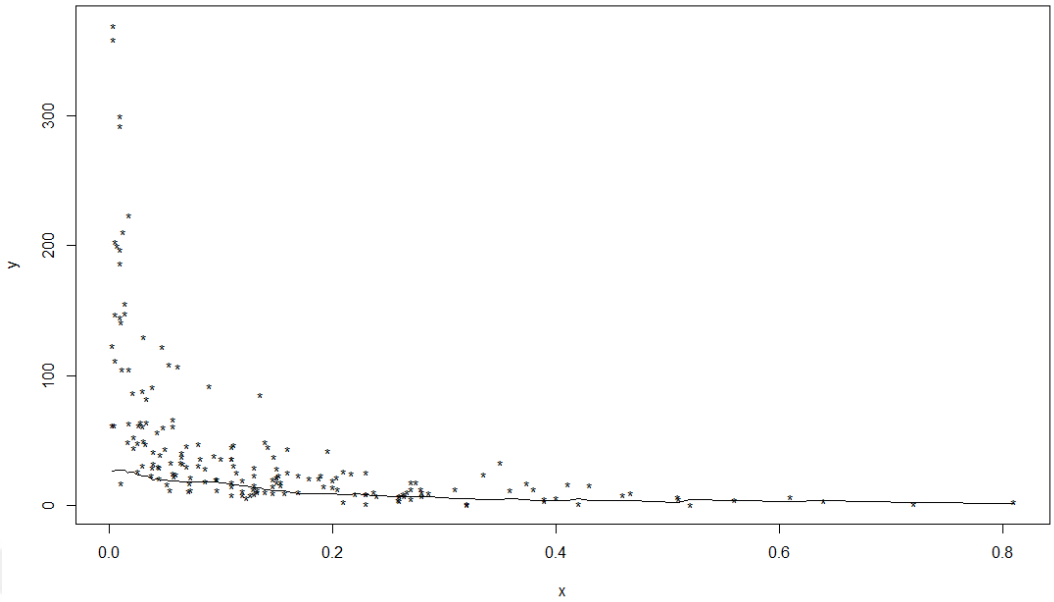

Figure 2.14 The quantile regression line of RIS with NO quantile estimator (q=0.2)

The quantile regression line for the case where "NO quantile estimator", “quantile=0.5” and "span = 0.8" is shown in Figure 2.15.

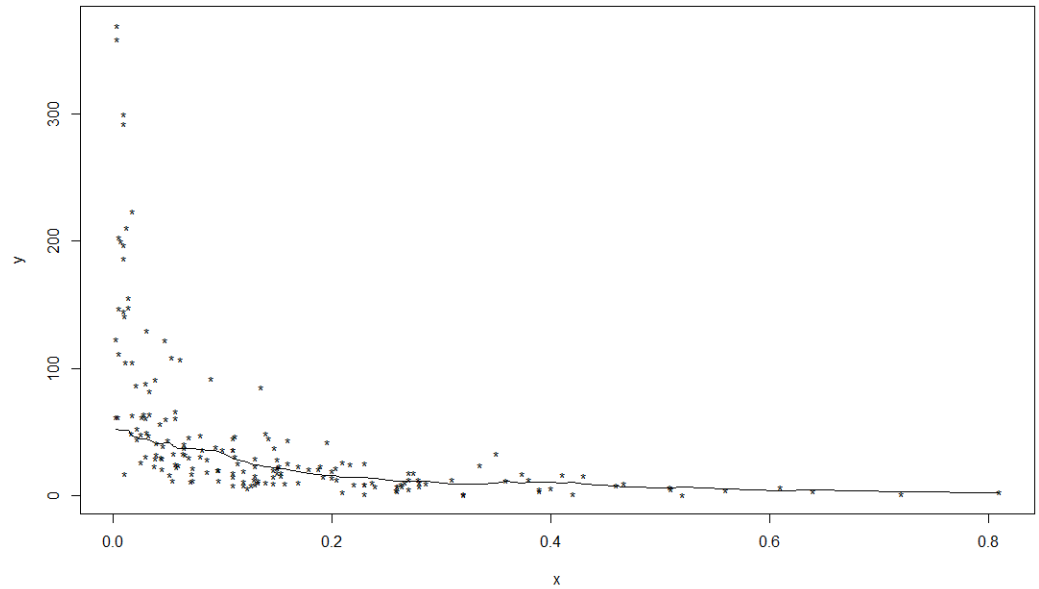

Figure 2.15 The quantile regression line of RIS with NO quantile estimator (q=0.5)

The quantile regression line for the case where "NO quantile estimator", “quantile=0.8” and "span = 0.8" is shown in Figure 2.16.

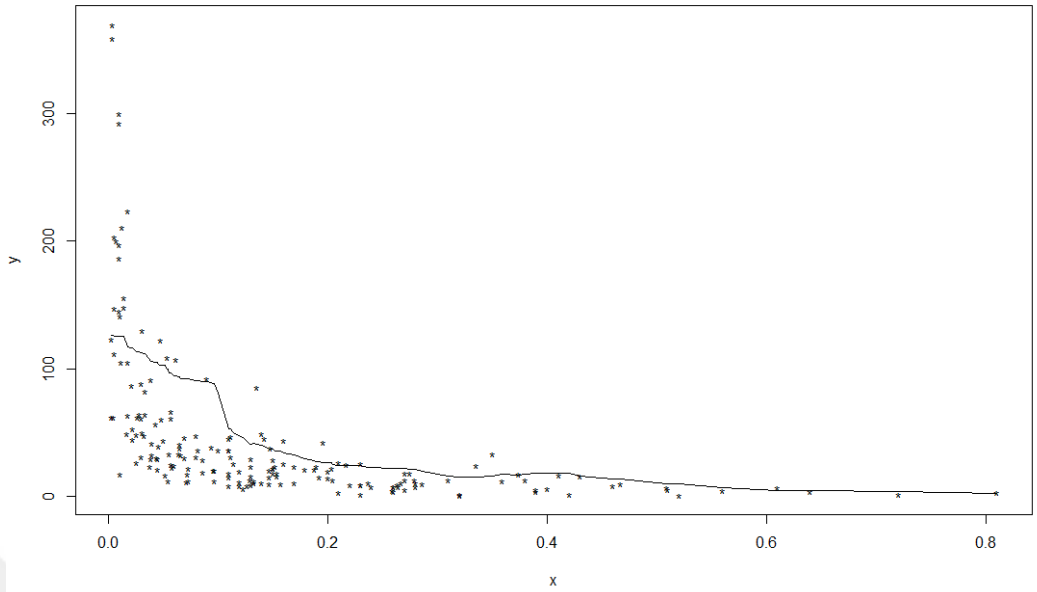

Figure 2.16 The quantile regression line of RIS with NO quantile estimator (q=0.8)

2.4 Constrained B-Spline Smoothing (COBS)

Constrained B-spline smoothing (COBS) method provides a way of dealing with quantiles (e.g., He & Ng, 1999; Koenker & Ng, 2005; Ng, 1996). This method works based on the estimated function of the constraints. In particular, COBS can include restrictions such as monotonicity, convexity, concavity and priodicity constraints. The main principle involves the expansion of smoothing splines with conditional quantile function estimation, a method introduced by Bosch et al. (1995).

In general, this technique of constrained b-spline smoothing is close to the b-splines methodology. It also combines b-splines with the method of smoothing splines.

2.4.1 Spline

Splines are used in applications that require data interpolation and / or smoothing. For interpolation, the spline functions are normally determined as minimization of appropriate roughness measurements that are subject to interpolation constraints. Smoothing splines can be seen as the generalizations of the interpolated splines in which the functions to minimize a weighted combination of the mean

squared approximation error over the observed data and the roughness measure. For a number of meaningful definitions of the roughness measure, it has been found that the spline functions are limited in nature, the primary cause of their use in calculations and representations.

Regression splines are special functions defined by piecewise polynomials. The region defining the pieces is separated by a sequence of knots or breakpoints (Hastie & Tibshirani, 1990). The aim is to force the piecewise polynomials to merge smoothly with the knots.

Splines can be named according to degrees. The simplest spline has degree 0 and it is also called a step function. The next simplest spline has degree 1 and it is also called a **linear spline**. The next simplest spline has degree 2 and it is also called a **quadratic spline**. A common spline is the **natural cubic spline** of degree 3. Cubic Splines has continuous first and second derivatives at the knots.

The simplest spline is a piecewise polynomial function where each polynomial has a single variable. Spline S takes values in the interval [a, b] and matches them to the real set of numbers $\mathbb{R}$.

$$\text{S: [a,b]} \rightarrow \mathbb{R}. \tag{2.26}$$

Since S is defined as part, choose k subintervals to perform partitioning [a, b]:

$$[t_i, t_{i+1}], \text{i} = 0, \dots, \text{k} - 1 \tag{2.27}$$

$$[a, b] = [t_0, t_1] \cup [t_1, t_2] \cup \dots \cup [t_{k-2}, t_{k-1}] \cup [t_{k-1}, t_k] \tag{2.28}$$

$$a = t_0 \leq t_1 \leq \cdots \leq t_{k-1} \leq t_k = b \tag{2.29}$$

Each of these subintervals is associated with a polynomial P_i,

$$P_i : [t_i, t_{i+1}] \rightarrow \mathbb{R}. \tag{2.30}$$

On the i^{th} subinterval of [a,b], S is defined by P_i,

$$\begin{aligned} S(t) &= P_0(t), t_0 \le t < t_1, \\ S(t) &= P_1(t), t_1 \le t < t_2, \\ &\vdots \\ S(t) &= P_{k-1}(t), t_{k-1} \le t < t_k. \end{aligned} \tag{2.31}$$

The given $k+1$ points t_j $(0 \le j \le k)$ are called **knots**. The vector $(t_0, t_1, \dots, t_k)$ is called a **knot vector** for the spline.

A spline of degree D is a function created by combining polynomial sections of degree D so that:

- the function is continuous,
- the function has D − 1 continuous derivatives.

$$(x - t_k)_+^D = \begin{cases} 0 & if\ x < t_k \\ (x - t_k)^D & if\ x \ge t_k \end{cases} \tag{2.32}$$

The equation for a spline of degree D with K knots:

$$y = \beta_0 + \sum_{d=1}^{D} \beta_d x^d + \sum_{k=1}^{K} b_k (x - t_k)_+^D \tag{2.33}$$

where b_k refers to the weight of each function of Dth degree and $(x - t_k)_+^D$ refers to the kth function of Dth degree with a knot at t_k.

2.4.1.1 Linear Splines

$$f(x) = \beta_0 + \beta_1 x + \sum_{k=1}^{K} b_k (x - t_k)_+^D. \tag{2.34}$$

For example; K=2 Knots

$$f(x) = \beta_0 + \beta_1 x + \sum_{k=1}^{2} b_k (x - t_k)_+^D$$

$$y = \beta_0 + \beta_1 x + b_1 (x - t_1)_+ + b_2 (x - t_2)_+$$

$$y = \beta_0 + \beta_1 x + \beta_2 (x - t)_+ + \beta_3 (x - t)_+$$

2.4.1.2 Quadratic Splines

For Example; K=3 Knots

$$y = \beta_0 + \sum_{d=1}^{2} \beta_d x^d + \sum_{k=1}^{3} b_k (x - t_k)_+^D$$

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + b_1 (x - t_1)_+^2 + b_2 (x - t_2)_+^2 + b_3 (x - t_3)_+^2$$

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 (x - t)_+^2 + \beta_4 (x - t)_+^2 + \beta_5 (x - t)_+^2$$

N = 1+D+K = 1+2+3 = 6 (Coefficients)

2.4.1.3 Cubic Splines

For Example; K=2 Knots

$$y = \beta_0 + \sum_{d=1}^{3} \beta_d x^d + \sum_{k=1}^{2} b_k (x - t_k)_+^D$$

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + b_1 (x - t_1)_+^3 + b_2 (x - t_2)_+^3$$

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - t)_+^3 + \beta_5 (x - t)_+^3$$

N = 1+D+K = 1+3+2 = 6 (Coefficients)

Splines computed from the piecewise polynomials may be numerically unstable because:

- The values may be very large, and
- correlation between the variables may be highly.

2.4.2 Basis Spline (B-Spline)

The basis spline (B-Spline) is a spline function with minimal support for the given degree, smoothness and domain partition. Any spline function of the given degree can be expressed as a linear combination of B-splines of this degree. B-splines can be used for curve fitting and numerical differentiation of experimental data.

For a given sequence of knots, there is, up to a scaling factor, a unique spline $B_{i,d}(x)$ satisfying

$$B_{i,d}(x) = \begin{cases} 0 & if\ x \le t_i\ or\ x \ge t_{i+d} \\ nonzero & otherwise \end{cases}. \qquad (2.35)$$

If we add the additional constraint that $\sum B_{i,d}(x) = 1$ for all x between the first and last knot, then the scaling factor of $B_{i,d}(x)$ becomes fixed. The resulting $B_{i,d}(x)$ spline functions are called B-splines.

Alternatively, the B-splines can be identified by construction via the Cox-de Boor recursion formula. Given a knot sequence $\ldots, t_0, t_1, t_2, \ldots$, then the B-splines of degree 0 are defined by

$$B_{i,0}(x) = \begin{cases} 1 & if \quad t_i \le x < t_{i+1} \\ 0 & otherwise \end{cases}. \tag{2.36}$$

Note that these satisfy $\sum B_{i,0}(x) = 1$ for all x because for any x exactly one of the $B_{i,0}(x) = 1$, and all the others are zero.

The higher degree B-splines are defined by recursion

$$B_{i,d}(x) = \frac{x - t_i}{t_{i+d} - t_i} B_{i,d-1}(x) + \frac{t_{i+d+1} - x}{t_{i+d+1} - t_{i+1}} B_{i+1,d-1}(x). \tag{2.37}$$

A B-spline of degree n is a parametric curve composed of a linear combination of basis B-splines $B_{i,n}(x)$ of degree n given by

$$B(x) = \sum_{i=0}^{N+d} \beta_i B_{i,d}(x), \qquad x \epsilon [t_0, t_{N+1}] \tag{2.38}$$

where N is the number of internal knots and $\{t_i\}_{i=0}^{N+2(d+1)}$ are the knot vectors.

For example; Quadratic b-spline (N=3 internal Knots)

$$B(x) = \sum_{i=0}^{3+2} \beta_i B_{i,d}(x)$$

$$B(x) = \beta_0 B_{0,2}(x) + \beta_1 B_{1,2}(x) + \beta_2 B_{2,2}(x) + \beta_3 B_{3,2}(x) + \beta_4 B_{4,2}(x) + \beta_5 B_{5,2}(x)$$

Coefficients = N+n+1=3+2+1=6

$$B_{i,d}(x) = \frac{x - t_i}{t_{i+d} - t_i} B_{i,d-1}(x) + \frac{t_{i+d+1} - x}{t_{i+d+1} - t_{i+1}} B_{i+1,d-1}(x)$$

$$B_{3,2}(x) = \frac{x - t_3}{t_5 - t_3} B_{3,1}(x) + \frac{t_6 - x}{t_6 - t_4} B_{4,1}(x)$$

$$B_{3,1}(x) = \frac{x - t_3}{t_4 - t_3} B_{3,0}(x) + \frac{t_5 - x}{t_5 - t_4} B_{4,0}(x)$$

$$B_{4,1}(x) = \frac{x - t_4}{t_5 - t_4} B_{4,0}(x) + \frac{t_6 - x}{t_6 - t_5} B_{5,0}(x)$$

2.4.3 B-Spline Smoothing

Let $\rho_\tau(u) = u(\tau - I(u < 0))$, where the indicator function, $u = [1 + (2\tau - 1)sgn(u)]|u|$

$$I(u < 0) = \begin{cases} 1 & u < 0 \\ 0 & otherwise \end{cases}. \qquad (2.39)$$

The goal is to estimate the τ quantile of *y* given *x* by finding a function *g(x)* that minimizes

$$\sum \rho_\tau(y_i - g(x_i)) + \lambda \sum |g^\iota(x_{i+1}^+) - g^\iota(x_i^+)| \qquad (2.40)$$

$$\sum \rho_\tau(y_i - g(x_i)) + \lambda \int |g^\iota(x)|\, dx \qquad (2.41)$$

based on the random sample $(x_1, \dots, y_1), \dots, (x_n, \dots, y_n)$ where λ is a scalar that controls smoothness. λ can be taken from zero to infinity. They show that $\hat{g}(x)$ is a linear smoothing spline for the first function while $\hat{g}(x)$ is a quadratic smoothing spline for the second function (Eilers & Marx, 1996).

The B-spline representation $g(x_i)$ smooth function becomes

$$g(x_i) = \sum_{j=0}^{N+d} \beta_j B_{j,d}(x) \tag{2.42}$$

where N is the number of internal knots, $B_{j,d}(x)$ are the B-spline basis functions β_j are the coefficients for the B-spline basis functions and *d* is a degree for the smoothing spline.

2.4.4 Choice of Smoothing Parameter or Knots

If a fully automatic smoothness is required, we must resolve the λ smoothing parameter or the problem of selecting knot vector $T = \{t_i\}_{i=1}^{N+2(d+1)}$ in the case of B-spline regression. Asymptotically, the Akaike information criterion (AIC) for the smoothing splines is usually used in the plane curves based on the least squares. For small sample sizes Akaike information criterion and Schwarz-type information criterion (SIC) show similar characteristics.

For COBS method, there is a function called cobs which is used in R programming language. When the argument lambda is supplied with a negative value, COBS computes the smoothing spline with λ chosen to minimize a Schwarz-type information criterion used in Koenker et al. (1994), and He, Ng & Portnoy (1998). SIC is defined as

$$SIC(\lambda) = \log\left(\frac{1}{n}\sum \rho_\tau\left(y_i - \hat{g}(x_i)\right)\right) + \frac{1}{2}p_\lambda\left(\frac{\log(n)}{n}\right) \tag{2.43}$$

where p_λ is the number of interpolated data points and serves as dimensionality measure of the fitted model.

When lambda is provided with a positive number, it will be used as the value of the smoothing parameter. No efforts will be made to choose the optimal λ. This option allows the user to experiment with various fits of different smoothness.

Akaike information criterion (AIC) is used for the choice of knots. Compute the regression B-Splines for N internal knots from knots. Select N that corresponds to the smallest

$$AIC(T) = \log\left(\frac{1}{n}\sum \rho_\tau\left(y_i - \hat{g}(x_i)\right)\right) + \frac{2(N+d+1)}{n} \tag{2.44}$$

where T are the knot vectors, N is the number of internal knots in T, d is the degree of spline used.

2.4.5 More on Other Quantiles

Conditional quantile functions beyond the median may provide a more complete picture of the relationship between the response and the covariate. COBS provides estimates of the τ^{th} conditional quantile via the tau argument. AIC and SIC criteria may not perform as well in choosing the smoothing parameters for extreme quantiles (τ close to 0 or 1) as for the median. User intervention is recommended in such situations (He & Ng, 1999).

2.4.6 Imposing Additional Constraints

Many qualitative restrictions on the fitted curves can be incorporated easily by the addition of equality or inequality constraints as described below (He & Ng, 1999):

- Monotonicity constraints (increasing and decreasing),

- convexity constraints,
- concavity constraints,
- periodicity constraints.

2.4.7 R Function "cobs"

COBS is applied via the R package "cobs".

In case it helps, the R function

library("cobs")

"cobs(x, y, constraint = "none", knots, method = "quantile", degree = 2, tau = 0.5, la
mbda = 0, ic = "SIC", knots.add = FALSE, …)"

is provided that plots the smooth (assuming that the R package cobs has been installed). The argument "tau" determines the quantile that will be used and defaults to the median. The argument "constraint" determines the kind of constraint. These constraints are increase, decrease, convex, concave and periodic; defaults to none. The argument "degree" determines degree of the splines; 1 for linear spline and 2 for quadratic spline; defaults to 2. Quantile level is desired by the argument "tau"; defaults to 0.5 (median). The argument "lambda" can be equal 0, in this case it uses regression B-splines. If lambda is greater than 0, it uses smoothing B-spline with the given λ and If lambda is less than 0, it uses smoothing B-spline with λ chosen by a Schwarz-type information criterion. The argument "ic" determines string indicating the information criterion used in knot deletion and addition for the regression B-spline method.

2.4.8 R Function "qsmcobs"

"qsmcobs" is applied via the R package "WRS2".

In case it helps, the R function

library(“WRS2”)

“qsmcobs(x, y, qval=0.5, xlab=“X”, ylab=“Y”, FIT=T, pc=“.”, plotit=T, xout=F, outfun=out, …)”

is provided that plots the smooth (assuming that the R package cobs has been installed). The argument qval determines the quantile that will be used and defaults to the median. There are two options regarding how the plot is created. The default approach, when FIT=T, is to estimate the quantiles of *y*, given *x*, using the predict command associated with the R command cobs. The second, when the argument FIT=F, estimates the quantile of *y* for each observed x_i and plots the results.

2.4.9 “predict” Function for COBS

Compute predicted values and simultaneous or pointwise confidence bounds for cobs objects.

“predict(object,z,deriv=0L,minz=knots[1],maxz=knots[knots],nz=100,…)”

The argument object determines object of class cobs. The argument z determines vector of grid points at which the fitted values are evaluated; defaults to an equally spaced grid with nz grid points between minz and maxz. Note that now z may lie outside of the knots interval which was not allowed originally. The argument deriv determines scalar integer specifying (the order of) the derivative that should be computed. The argument nz determines number of grid points in z if that is not given; defaults to 100.

2.4.10 An Example

COBS is examined graphically to understand how the methods can model the relationship between variables using the Joyner-Boore attenuation data.

The line of quadratic b-splines are shown in Figure 2.17 when lambda is 0 and the quantile value is 0.5.

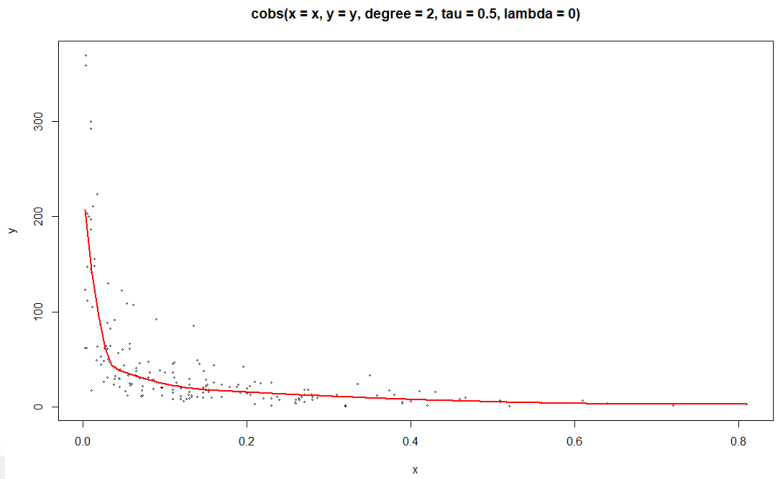

Figure 2.17 The quantile regression line of COBS (degree=2, quantile=0.5, lambda=0)

The line of linear b-splines are shown in Figure 2.18 when lambda is 0 and the quantile value is 0.5.

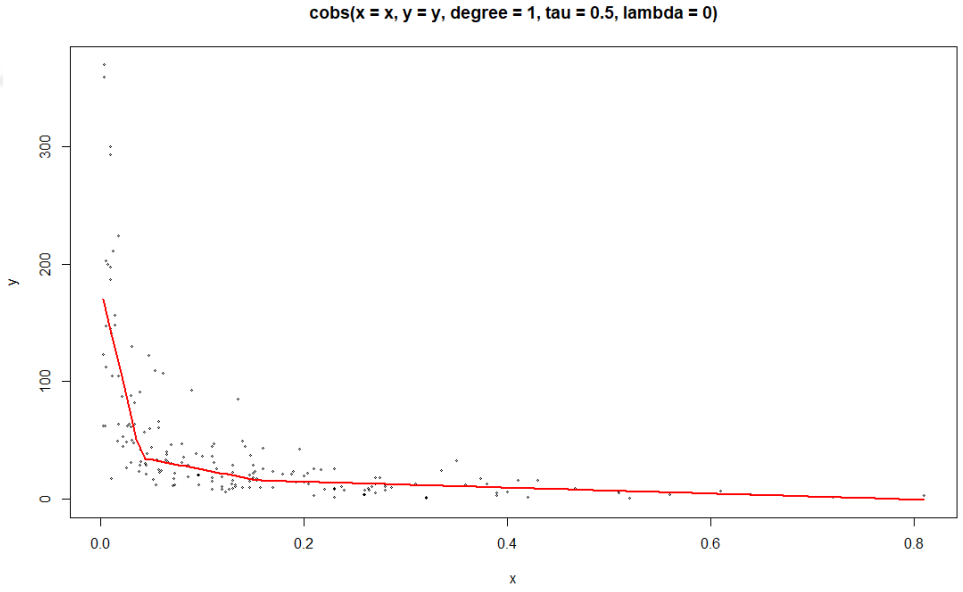

Figure 2.18 The quantile regression line of COBS (degree=1, quantile=0.5, lambda=0)

For quadratic b-spline, when the argument lambda is supplied with a negative value, COBS computes the smoothing spline with λ chosen to minimize a Schwarz-type information criterion. The line of quadratic b-splines are shown in Figure 2.19 when lambda is 0.00702 and the quantile value is 0.5.

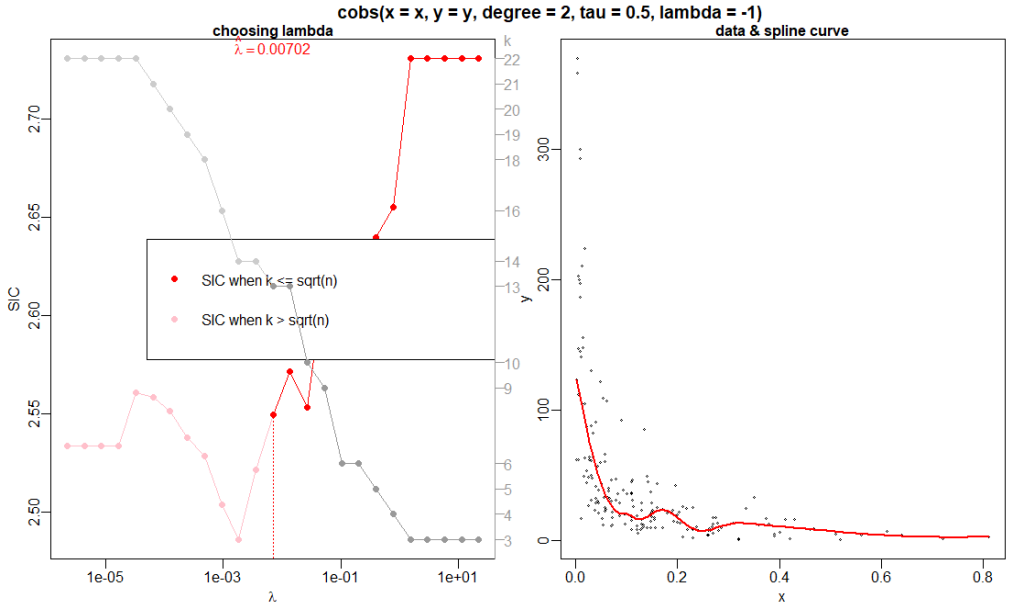

Figure 2.19 The quantile regression line of COBS (degree=2, quantile=0.5, lambda=0.00702)

For linear b-spline, when the argument lambda is supplied with a negative value, COBS computes the smoothing spline with λ chosen to minimize a Schwarz-type information criterion. The line of linear b-splines are shown in Figure 2.18 when lambda is 0.0922 and the quantile value is 0.5.

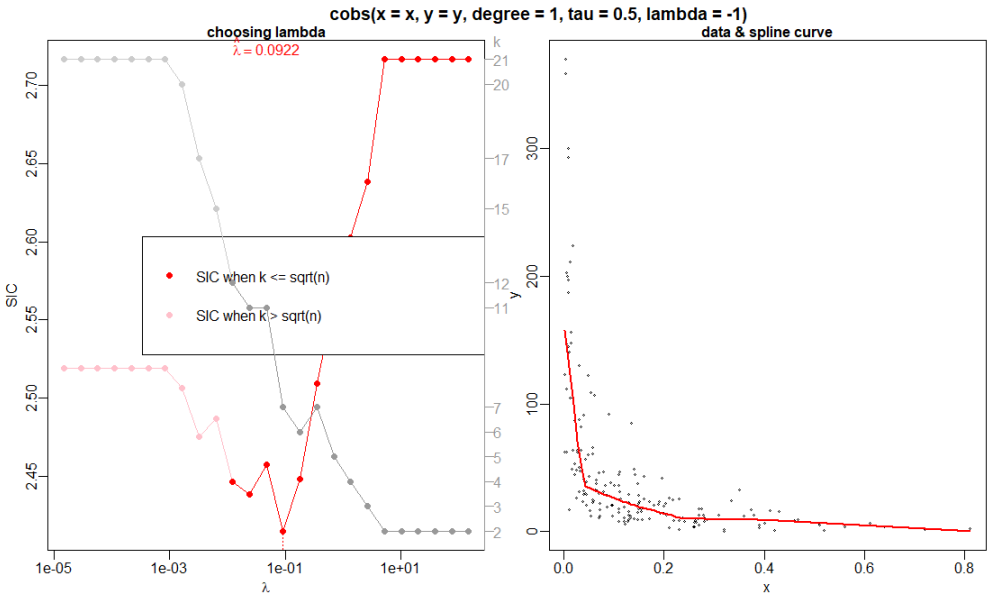

Figure 2.20 The quantile regression line of COBS (degree=1, quantile=0.5, lambda=0.0922)

The line of quadratic b-splines are shown in Figure 2.21 when lambda is 0 and the quantile value is 0.2.

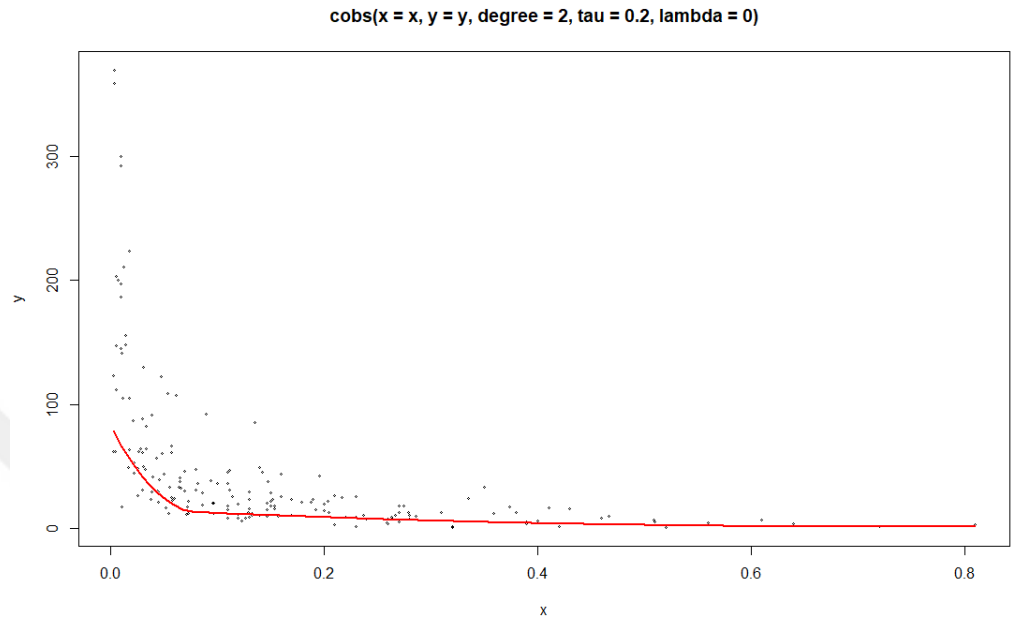

Figure 2.21 The quantile regression line of COBS (degree=2, quantile=0.2, lambda=0)

The line of quadratic b-splines are shown in Figure 2.22 when lambda is 0 and the quantile value is 0.8.

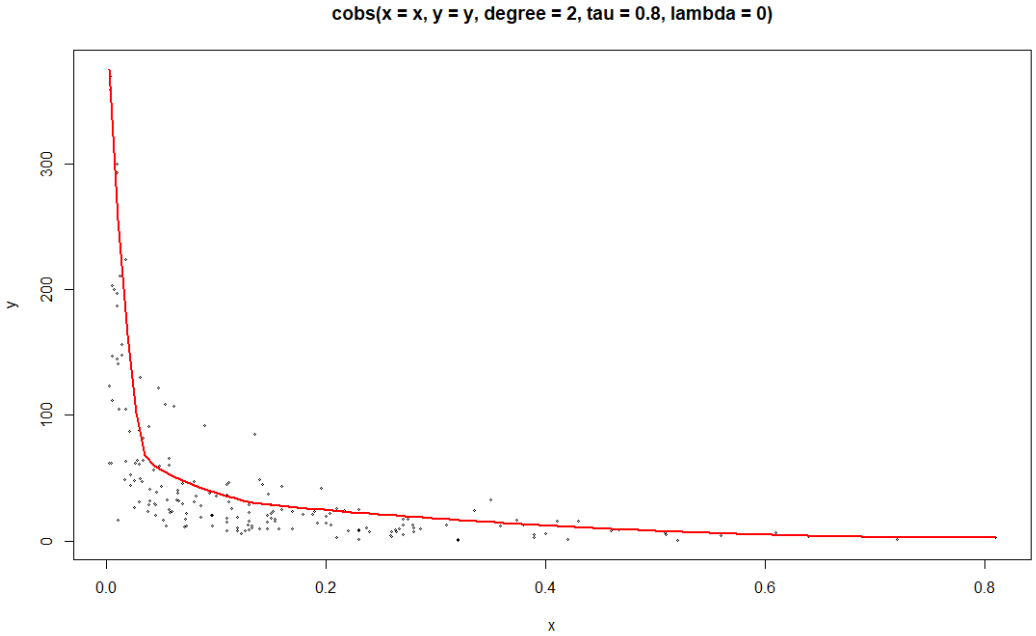

Figure 2.22 The quantile regression line of COBS (degree=2, quantile=0.8, lambda=0)

CHAPTER THREE
APPLICATION

In this chapter of the thesis, constrained b-spline smoothing (COBS), the running interval smoother (RIS) with HD and NO quantile estimators and alternative methods are compared with a simulation study. These alternative methods consist of a blend of two smoothers: RUNLOW (the running interval smoother and locally weighted scatter plot smoothing) recommended by Wilcox (2016), COBSLOW (constrained b-spline smoothing and locally weighted scatter plot smoothing) and RUNCOBS - COBSRUN (the running interval smoother and constrained b-spline smoothing). In this simulation study, the methods created by blending of RIS were used with HD and NO estimators. Furthermore, the methods are investigated graphically to understand how the methods can model the relationship between variables using a real data set. The predicted values of dependent variable corresponding to new observations are calculated as well.

3.1 Design of Simulation Study

Simulations were used to compare the small-sample properties of COBS, RIS and alternative methods based on K = 10000 replications and sample size n=25, n=50. The data were generated according to the model

$$Y = X + \lambda\,(X)\epsilon \tag{3.1}$$

where X and ϵ have one of four distributions: normal, symmetric and heavy-tailed, asymmetric and light-tailed, and asymmetric and heavy-tailed. The distribution for the X and ϵ were taken to be one of four g-and-h distributions (Hoaglin, 1985) that contain the standard normal distribution as a special case. Let Z be a random variable which is generated from standard normal distribution. When $g \neq 0$, with the transformation

$$X = \frac{\exp(gz) - 1}{g}\,\exp(hZ^2/2) \tag{3.2}$$

and when $g = 0$, with the transformation

$$X = Zexp(hZ^2/2) \tag{3.3}$$

the data is generated from g-and-h distribution.

Seven different g-and-h distributions are used:

- Standard normal distribution ($g = h = 0$),
- symmetric heavy tailed distribution ($g = 0, h = 0.2$),
- symmetric heavy tailed distribution ($g = 0, h = 0.5$),
- asymmetric light tailed distribution ($g = 0.2, h = 0$),
- asymmetric light tailed distribution ($g = 0.5, h = 0$),
- asymmetric heavy tailed distribution ($g = 0.2, h = 0.2$),
- asymmetric heavy tailed distribution ($g = 0.5, h = 0.5$).

Table 3.1 shows the skewness (κ_1) and kurtosis (κ_2) for each distribution.

Table 3.1 Some properties of the g-and-h distribution

g	h	κ_1	κ_2
0	0	0	3
0	0.2	0	21.46
0.2	0	0.61	3.68
0.2	0.2	2.81	155.98

In this simulation study, there are three different variance patterns (VP): $\lambda\,(X) = 1$, $\lambda\,(X) = |X| + 1$, and $\lambda\,(X) = 1/(|X| + 1)$. These three choices are called VP1, VP2 and VP3.

Details about the criterion used to compare these methods are as follows. The criterion was mean squared error, which was estimated with

$$MSE = \frac{1}{nK}\sum_{k=1}^{K}\sum_{i=1}^{n}(\theta_{qik} - \hat{\theta}_{qik})^2 \tag{3.4}$$

where now, for the k^{th} replication, θ_{qik} is the true conditional q^{th} quantile of Y given $X=X_i$ and again $\hat{\theta}_{qik}$ is the estimate of θ_{qik} based on robust nonlinear regression methods.

In this simulation study, methods are compared with three different quantile values: q=0.1, q=0.5 and q=0.9. The sample size of n=25 and n=50 are taken from each distribution. R programming language is used for this simulation study.

3.2 Simulation Results

3.2.1 Mean Squared Error (MSE) Criterion

Table 3.2 MSE values (x has g=0 and h=0, ϵ has g=0 and h=0)

					RUNCOBS		COBSRUN		RIS		RUNLOW	
			COBSLOW	COBS	HD	NO	HD	NO	HD	NO	HD	NO
n=25	q=0.1	VP1	4.366995	2.249919	5.028552	4.517154	4.036047	4.033806	3.11357	2.601816	4.568843	4.051114
		VP2	10.3075	6.948166	12.11509	10.51111	9.315843	9.306645	8.671754	7.170572	10.88537	9.187708
		VP3	2.912716	0.9840256	3.290064	3.040986	2.675905	2.674552	1.684151	1.400975	3.021416	2.787525
	q=0.5	VP1	3.036332	0.8959453	2.753874	2.751507	2.694652	2.69716	0.9733737	0.9707291	2.696014	2.69473
		VP2	6.053478	2.862207	5.659451	5.657542	5.506504	5.508993	3.100983	3.083434	5.393313	5.389333
		VP3	2.346417	0.3843433	2.140234	2.135724	2.060974	2.062924	0.4382661	0.4380508	2.111235	2.107011
	q=0.9	VP1	4.42113	2.253201	5.038856	4.529436	4.096927	4.093688	3.10504	2.598	4.559523	4.042609
		VP2	10.47558	6.923932	12.24911	10.63238	9.459957	9.453457	8.697131	7.193585	10.94792	9.236507
		VP3	2.926867	0.9791324	3.281576	3.041914	2.688366	2.687365	1.669005	1.390325	3.002142	2.774946
n=50	q=0.1	VP1	4.600138	2.528965	5.169941	4.771546	4.231345	4.23193	3.248583	2.923573	4.762495	4.378578
		VP2	11.237	8.570979	12.70147	11.32932	9.899772	9.905806	9.23266	8.165444	11.57333	10.15669
		VP3	2.976826	1.052497	3.311599	3.137107	2.716193	2.717123	1.723098	1.55768	3.059021	2.90857
	q=0.5	VP1	3.008537	0.9520275	2.721601	2.720786	2.649158	2.651091	1.019646	1.018523	2.672826	2.673009
		VP2	5.811858	3.238769	5.456794	5.454839	5.337805	5.339728	3.384143	3.375882	5.248825	5.248838
		VP3	2.373394	0.4015328	2.150507	2.149056	2.042024	2.043752	0.4528658	0.4526453	2.122171	2.120805
	q=0.9	VP1	4.602405	2.536172	5.161818	4.773272	4.248057	4.24941	3.240751	2.919535	4.748236	4.366667
		VP2	11.28958	8.557723	12.72057	11.40273	9.972002	9.975376	9.237667	8.179951	11.59607	10.18049
		VP3	2.993406	1.052509	3.32333	3.147526	2.729104	2.729844	1.726772	1.561311	3.063079	2.912356

Table 3.3 MSE values (x has g=0 and h=0, ϵ has g=0.2 and h=0)

					RUNCOBS		COBSRUN		RIS		RUNLOW	
			COBSLOW	COBS	HD	NO	HD	NO	HD	NO	HD	NO
n=25	q=0.1	VP1	4.267807	2.170352	4.826067	4.467965	3.946646	3.944592	2.993213	2.60373	4.42198	4.045728
		VP2	9.915238	6.667444	11.41191	10.37056	9.039522	9.038918	8.269809	7.199157	10.35307	9.132302
		VP3	2.854489	0.9411034	3.179439	2.999471	2.612174	2.611639	1.623121	1.398459	2.937349	2.761817
	q=0.5	VP1	3.114011	0.9611846	2.831555	2.828993	2.774579	2.776559	1.034457	1.031339	2.774857	2.7739
		VP2	6.226862	3.081341	5.829197	5.825149	5.67715	5.680758	3.31076	3.290857	5.552455	5.551371
		VP3	2.38224	0.4137177	2.171934	2.167289	2.095072	2.097096	0.465153	0.464908	2.143198	2.138932
	q=0.9	VP1	4.8467	2.651005	5.674587	4.877553	4.505945	4.49981	3.579204	2.844158	5.081406	4.299915
		VP2	11.77407	7.995532	14.12419	11.54462	10.53272	10.52176	9.981268	7.761318	12.4668	9.908382
		VP3	3.084975	1.157751	3.552199	3.182218	2.859183	2.85672	1.873854	1.490253	3.209873	2.873613
n=50	q=0.1	VP1	4.473972	2.433763	4.957996	4.662931	4.123746	4.12511	3.137387	2.884947	4.601014	4.3115
		VP2	10.69861	8.120104	11.89142	11.00267	9.581946	9.587074	8.868514	8.094882	11.00529	9.965437
		VP3	2.934608	1.008124	3.240509	3.096171	2.66377	2.664734	1.682983	1.546296	3.008335	2.885982
	q=0.5	VP1	3.076886	1.020263	2.789568	2.788441	2.714468	2.716095	1.084773	1.08349	2.741651	2.741489
		VP2	6.090375	3.474682	5.751176	5.748998	5.610168	5.612165	3.620698	3.610582	5.522683	5.522317
		VP3	2.392298	0.428818	2.165825	2.164248	2.059469	2.060964	0.477757	0.4775337	2.135786	2.134245
	q=0.9	VP1	4.981392	2.904074	5.742156	5.161216	4.602548	4.603248	3.628178	3.183121	5.208203	4.661966
		VP2	12.85488	10.00036	14.80982	12.67129	11.19375	11.19729	10.51627	8.981764	13.30556	11.18053
		VP3	3.132261	1.198043	3.55762	3.31033	2.880199	2.88079	1.883513	1.667285	3.244713	3.036663

Table 3.4 MSE values (x has g=0 and h=0, ϵ has g=0.5 and h=0)

					RUNCOBS		COBSRUN		RIS		RUNLOW	
			COBSLOW	COBS	HD	NO	HD	NO	HD	NO	HD	NO
n=25	q=0.1	VP1	4.610888	2.519289	5.110692	4.867293	4.304248	4.303699	3.344402	3.049497	4.732282	4.445007
		VP2	11.2296	7.983286	12.60908	12.22306	10.49452	10.49714	9.542565	8.941088	11.57471	10.74184
		VP3	2.995787	1.084297	3.300713	3.156683	2.746016	2.745827	1.771261	1.587198	3.073169	2.929045
	q=0.5	VP1	3.525923	1.353123	3.233854	3.229126	3.18473	3.187689	1.413667	1.407262	3.168133	3.164515
		VP2	7.936378	4.370182	7.58996	7.577339	7.374565	7.373916	4.660149	4.616968	7.209992	7.20015
		VP3	2.55344	0.5866731	2.334168	2.329105	2.267606	2.26939	0.625042	0.6241083	2.30364	2.299117
	q=0.9	VP1	6.354904	4.109365	7.852415	6.055208	5.884884	5.873507	5.188896	3.725799	6.827258	5.188181
		VP2	17.28425	12.46085	21.41185	15.45404	15.03009	15.03019	14.83272	10.33112	18.27017	12.91463
		VP3	3.666852	1.822583	4.559633	3.717337	3.466928	3.463449	2.636445	1.897233	3.983048	3.270347
n=50	q=0.1	VP1	4.801327	2.776138	5.263617	5.050518	4.462165	4.463467	3.518667	3.321656	4.934772	4.710583
		VP2	11.66755	9.235945	12.74311	12.28274	10.75463	10.75997	10.04598	9.548009	11.9932	11.2622
		VP3	3.071933	1.141969	3.385701	3.260855	2.791691	2.792649	1.845492	1.730212	3.16465	3.05613
	q=0.5	VP1	3.509211	1.44874	3.20723	3.205263	3.149905	3.151682	1.492446	1.489685	3.155623	3.153723
		VP2	7.638035	4.918449	7.294747	7.290357	7.162619	7.164634	5.083993	5.061708	7.037453	7.030369
		VP3	2.599168	0.6172478	2.358205	2.356643	2.264971	2.266485	0.651629	0.6512725	2.327645	2.326019
	q=0.9	VP1	6.319969	4.243442	7.759526	6.497554	5.842089	5.841485	5.028941	4.182683	6.821537	5.710052
		VP2	18.18367	15.14932	22.00457	17.27762	15.356	15.36636	15.01439	12.05088	19.04582	14.80712
		VP3	3.633208	1.728798	4.409194	3.893794	3.397355	3.397645	2.479762	2.10285	3.891221	3.484898

Table 3.5 MSE values (x has g=0 and h=0, ϵ has g=0 and h=0.2)

					RUNCOBS		COBSRUN		RIS		RUNLOW	
			COBSLOW	COBS	HD	NO	HD	NO	HD	NO	HD	NO
n=25	q=0.1	VP1	7.118861	4.891313	8.814261	6.801799	6.673534	6.667671	6.050084	4.515573	7.742651	5.972481
		VP2	19.87342	15.09117	24.59343	18.53294	17.77187	17.79414	17.83358	13.30957	21.23367	15.80109
		VP3	3.978431	2.130277	4.878855	3.974726	3.739361	3.739438	2.958232	2.218785	4.299506	3.57046
	q=0.5	VP1	4.15918	1.948789	3.91087	3.910289	3.817501	3.819768	2.036818	2.029815	3.830305	3.831291
		VP2	10.39126	6.373965	10.27343	10.28199	9.786443	9.787725	6.80508	6.755222	9.697878	9.714651
		VP3	8.723994	0.8287153	2.602009	2.597921	2.517918	2.520239	0.879713	0.8787152	2.565836	2.562958
	q=0.9	VP1	7.148377	4.849704	8.774795	6.811367	6.696755	6.696366	6.023119	4.509539	7.671922	5.960304
		VP2	19.96388	14.97834	24.70543	18.46608	17.84061	17.85424	17.71665	13.23327	21.10496	15.71907
		VP3	3.997549	2.128025	4.896999	3.996152	3.763121	3.76283	2.980882	2.232742	4.308819	3.573049
n=50	q=0.1	VP1	7.064661	4.973176	8.606823	7.25229	6.529845	6.531068	5.850675	5.010197	7.647387	6.514539
		VP2	20.41189	17.61408	25.0125	19.81828	17.75342	17.75718	17.89418	14.91423	21.67232	17.40254
		VP3	3.880448	1.974711	4.692054	4.158964	3.618795	3.619494	2.801327	2.440282	4.198976	3.78205
	q=0.5	VP1	4.161277	2.070309	3.883398	3.883246	3.796933	3.798728	2.141933	2.139122	3.820645	3.820901
		VP2	9.955319	6.995347	9.776936	9.782853	9.468883	9.470944	7.235389	7.212882	9.386602	9.388718
		VP3	2.861122	0.8682989	2.632154	2.630787	2.523888	2.525399	0.918277	0.9179761	2.60158	2.600285
	q=0.9	VP1	7.002199	4.92492	8.616535	7.254302	6.504932	6.505347	5.849073	5.017542	7.622235	6.503403
		VP2	20.12397	17.52361	24.66225	19.68014	17.60471	17.62455	17.73009	14.80628	21.4701	17.29938
		VP3	3.861286	1.965312	4.662217	4.140695	3. 607525	3.607945	2.784778	2.429949	4.1656	3.762457

Table 3.6 MSE values (x has g=0 and h=0, ϵ has g=0 and h=0.5)

					RUNCOBS		COBSRUN		RIS		RUNLOW	
			COBSLOW	COBS	HD	NO	HD	NO	HD	NO	HD	NO
n=25	q=0.1	VP1	90.80125	92.16226	312.4691	86.84447	87.84742	87.91645	171.6835	73.02464	182.1526	79.87066
		VP2	521.9184	643.5242	901.5248	264.7312	518.0943	509.8564	579.6152	239.3349	583.6169	243.97
		VP3	22.47674	21.89912	30.70096	16.85333	22.17743	22.038	21.86117	14.01074	24.57818	15.49754
	q=0.5	VP1	29.76791	26.38563	29.89852	29.89136	29.40321	29.40742	26.83827	26.72298	29.42268	29.40219
		VP2	113.6852	102.2637	115.3788	115.3032	112.9438	112.96	104.6106	103.0229	112.8536	112.8713
		VP3	10.42181	8.258196	10.28213	10.27582	10.1235	10.12624	8.326363	8.298001	10.15524	10.14936
	q=0.9	VP1	65.87946	62.17643	98.67343	55.66905	60.23249	60.02334	69.6326	45.5165	76.28881	49.34035
		VP2	388.0386	394.1404	546.718	178.9076	321.434	307.3937	326.9072	147.1932	392.0868	157.0176
		VP3	40.67322	73.78299	108.3748	34.39028	46.64014	47.026	65.0905	28.98709	64.85942	31.97828
n=50	q=0.1	VP1	68.788	61.74646	96.50622	51.5517	47.85332	47.6732	54.64422	43.96973	63.29174	47.25849
		VP2	151.891	177.6485	310.965	153.1162	138.3204	138.9176	179.3753	127.1028	189.6075	133.9131
		VP3	17.45061	15.88608	26.50053	18.15388	16.83101	16.83006	18.44454	15.29635	20.85142	16.77509
	q=0.5	VP1	30.77777	28.23478	30.70786	30.74239	30.40912	30.41294	28.36822	28.29399	30.42625	30.42642
		VP2	117.1753	111.8749	118.3361	118.4925	116.6644	116.6684	112.53	112.2902	116.5519	116.5549
		VP3	12.43519	10.4387	12.21844	12.21811	12.10015	12.10167	10.47779	10.46908	12.17054	12.16839
	q=0.9	VP1	42.64592	42.20838	68.14457	43.13988	40.40521	40.40257	46.01534	37.45604	52.38997	39.94799
		VP2	155.3236	477.8187	266.9387	140.5892	132.3591	132.1991	158.4502	118.4235	203.1041	129.3866
		VP3	20.50191	19.18765	41.25393	21.58379	19.83618	19.88172	24.36363	18.2261	38.96279	20.05777

Table 3.7 MSE values MSE (x has g=0 and h=0, ϵ has g=0.2 and h=0.2)

					RUNCOBS		COBSRUN		RIS		RUNLOW	
			COBSLOW	COBS	HD	NO	HD	NO	HD	NO	HD	NO
n=25	q=0.1	VP1	6.752643	4.512798	7.935553	6.749393	6.352377	6.349126	5.632338	4.620211	7.166455	6.046644
		VP2	18.69847	14.27416	22.13241	18.59306	17.15995	17.15692	16.86722	14.03775	19.72562	16.0955
		VP3	3.86263	1.980131	4.551134	3.985565	3.636321	3.637433	2.811624	2.288757	4.118063	3.624314
	q=0.5	VP1	4.397952	2.17641	4.140271	4.139086	4.052705	4.056596	2.267429	2.258853	4.062139	4.059989
		VP2	11.15795	6.98377	11.07202	11.0806	10.55103	10.55358	7.426909	7.362588	10.44488	10.44309
		VP3	2.912576	0.920942	2.697917	2.693073	2.620902	2.622584	0.971731	0.9705771	2.664294	2.659697
	q=0.9	VP1	8.538271	6.134152	10.88152	7.594577	7.867181	7.863174	7.355163	5.001911	9.264324	6.524534
		VP2	25.61558	19.73816	31.9356	20.98425	22.06859	22.19954	22.29221	14.76845	26.77332	17.64981
		VP3	4.485806	2.651528	5.732533	4.299795	4.2182	4.214779	3.528537	2.420952	4.905811	3.780406
n=50	q=0.1	VP1	6.853089	4.765879	8.000651	7.160339	6.42314	6.423921	5.693826	5.10423	7.311789	6.551845
		VP2	19.54121	16.66701	22.74496	19.91683	17.57135	17.57682	17.44334	15.47897	20.59364	17.65466
		VP3	3.862589	1.93295	4.479785	4.122868	3.60017	3.600757	2.744632	2.477924	4.100997	3.810945
	q=0.5	VP1	4.413295	2.340672	4.145541	4.144651	4.054793	4.056245	2.407659	2.404839	4.079595	4.078731
		VP2	10.85938	7.870803	10.65631	10.66242	10.37021	10.37355	8.108439	8.079475	10.27717	10.27885
		VP3	2.977344	0.9883223	2.74665	2.745185	2.64234	2.643996	1.035821	1.035418	2.716738	2.715185
	q=0.9	VP1	7.912751	5.909821	10.33408	8.111129	7.280877	7.285179	6.823596	5.558032	8.857337	7.107909
		VP2	24.54564	22.18783	31.66464	22.89745	20.58988	20.5951	21.26362	16.6205	26.1187	19.65861
		VP3	4.173355	2.305673	5.345241	4.524461	3.921852	3.921852	3.189628	2.678131	4.630973	4.033608

Table 3.8 MSE values (x has g=0 and h=0, ϵ has g=0.5 and h=0.5)

					RUNCOBS		COBSRUN		RIS		RUNLOW	
			COBSLOW	COBS	HD	NO	HD	NO	HD	NO	HD	NO
n=25	q=0.1	VP1	129.7687	126.3304	136.9864	131.3287	128.6474	128.6386	129.3588	125.9947	132.159	125.4178
		VP2	257.4839	219.6746	319.3679	299.3416	257.4906	257.8097	229.4948	219.9301	278.5409	242.6135
		VP3	192.7784	188.3864	195.9518	193.3687	192.5395	192.5202	192.4567	190.655	193.9527	191.2924
	q=0.5	VP1	1811.392	1806.33	1813.015	1812.624	1811.058	1811.05	1807.487	1808.261	1810.936	1809.851
		VP2	583.9433	567.3846	588.3292	586.7459	583.1024	583.1164	570.0964	568.0523	582.7617	580.4459
		VP3	58.09048	55.44884	57.97034	57.83173	57.82183	57.8058	55.80928	55.69085	57.76604	57.71169
	q=0.9	VP1	588.8232	706.0947	1253.536	312.3659	552.5666	557.0624	705.4047	256.9838	741.9211	287.3863
		VP2	1178.401	1939.248	3413.015	841.8906	977.8837	999.8013	2000.654	673.189	2147.695	779.7373
		VP3	233.7194	306.6127	331.9365	84.16058	233.6091	232.458	215.0336	74.82774	236.101	79.16615
n=50	q=0.1	VP1	6943.524	6944.805	6950.933	6944.818	6942.137	6942.144	6943.222	6947.582	6945.853	6942.071
		VP2	509.9806	511.8314	534.8175	526.4215	505.6387	505.6227	511.0238	515.1028	518.8356	504.4017
		VP3	6829.367	6827.169	6833.038	6830.582	6828.803	6828.83	6828.439	6827.332	6831.004	6829.122
	q=0.5	VP1	199.8534	197.418	199.8249	200.0311	199.5012	199.5019	197.3365	197.1776	199.4356	199.4191
		VP2	1136.953	1125.255	1173.623	1190.612	1136.535	1136.538	1038.631	983.7236	1136.31	1136.266
		VP3	301.2457	299.1911	301.0046	301.1793	300.8969	300.8985	299.1739	299.3331	300.9188	300.9113
	q=0.9	VP1	160.3544	344.0921	446.2342	163.0583	151.656	151.7701	228.1126	135.9953	250.6846	151.838
		VP2	777.3764	949.5187	2143.13	735.0795	705.373	705.3259	1000.57	616.5355	1085.001	674.9467
		VP3	128.0534	126.7164	285.2517	120.3177	115.3115	115.2674	133.4224	109.4224	149.5694	113.0849

Table 3.9 MSE values MSE (x has g=0.2 and h=0.2, ϵ has g=0.2 and h=0.2)

					RUNCOBS		COBSRUN		RIS		RUNLOW	
			COBSLOW	COBS	HD	NO	HD	NO	HD	NO	HD	NO
n=25	q=0.1	VP1	9.752094	4.458646	11.06711	9.874195	9.295791	9.285842	5.476068	4.508428	10.16142	9.051173
		VP2	28.9888	16.40173	33.17784	29.70752	26.55761	26.53961	17.3349	14.5521	28.63087	24.9437
		VP3	6.554861	1.866436	7.324913	6.733798	6.277892	6.277318	2.716604	2.218206	6.85198	6.337229
	q=0.5	VP1	7.099075	2.175105	6.83523	6.834838	6.677611	6.680955	2.197546	2.188995	6.714378	6.720553
		VP2	19.61886	8.497014	21.00632	21.014	18.63297	18.61306	7.974942	7.900408	18.57419	18.634
		VP3	5.529515	0.8836328	5.27978	5.274717	5.184558	5.18755	0.931688	0.9305323	5.238446	5.233298
	q=0.9	VP1	11.47087	6.198845	13.2425	10.07816	10.5451	10.55549	7.177707	4.850629	11.29458	8.69884
		VP2	40.17549	22.16435	45.07788	33.49292	34.57834	34.63323	23.14871	15.49519	35.3627	26.30596
		VP3	6.992262	2.601274	8.238974	6.812818	6.628732	6.626402	3.574332	2.428527	7.25768	6.192285
n=50	q=0.1	VP1	9.528304	4.688912	10.8706	10.00481	8.961919	8.963252	5.541637	4.954018	10.02095	9.253969
		VP2	30.18218	20.87756	34.15091	31.27287	26.92574	26.95759	19.34256	17.28479	29.44886	25.99683
		VP3	6.657682	1.826329	7.363608	6.982452	6.324143	6.326532	2.677636	2.405937	6.946105	6.644653
	q=0.5	VP1	7.156336	2.304096	6.879031	6.882026	6.6928	6.695626	2.318465	2.314219	6.718124	6.721732
		VP2	19.71723	10.18521	21.69164	21.71307	18.90396	18.90681	9.266174	9.212404	18.9391	18.97035
		VP3	5.596805	0.9432722	5.324359	5.323438	5.191225	5.193851	0.993641	0.9931132	5.269896	5.268058
	q=0.9	VP1	11.09902	5.798883	13.3456	11.04745	10.13854	10.15299	6.716938	5.44115	11.24014	9.487856
		VP2	47.09648	27.76176	52.82142	41.99781	39.07656	38.64695	23.53467	18.02883	40.77293	32.52071
		VP3	6.581597	2.132311	7.782909	6.97591	6.172182	6.174337	3.112666	2.601049	6.879659	6.325499

Table 3.10 MSE values (x has g=0.5 and h=0.5, ϵ has g=0.5 and h=0.5)

					RUNCOBS		COBSRUN		RIS		RUNLOW	
			COBSLOW	COBS	HD	NO	HD	NO	HD	NO	HD	NO
n=25	q=0.1	VP1	297.0718	118.8283	304.6844	305.8924	295.1332	295.1172	112.7643	147.9567	298.0483	291.3282
		VP2	21059.41	975.4208	30387.43	30624.74	20990.98	20992.42	824.3184	927.5128	20965.44	20939.56
		VP3	186.6033	38.78027	190.0983	186.3725	184.9688	184.9976	40.33308	38.49724	187.1812	183.3374
	q=0.5	VP1	379.2644	76.82274	383.5959	383.5675	378.5497	378.5335	73.32431	73.01311	378.9289	378.9575
		VP2	10438.63	428.207	17213.21	17214.48	10407.69	10411.81	302.817	296.1285	10382.62	10388.63
		VP3	299.0677	22.23661	299.0844	299.0465	298.6973	298.7028	21.33725	21.18757	298.6745	298.6098
	q=0.9	VP1	1481.895	516.1904	2699.181	1284.101	1435.664	1461.202	928.6721	393.0557	2120.203	1231.038
		VP2	6201.852	1411.17	6082.503	4369.683	4100.834	4124.923	1285.827	501.3549	4357.684	3550.333
		VP3	446.1489	288.7282	778.1252	348.0212	511.9379	511.195	312.9019	106.3682	604.7589	345.5507
n=50	q=0.1	VP1	347.711	102.2554	360.5127	356.3615	346.3401	346.3464	102.4298	103.299	351.4641	347.3332
		VP2	21747.15	1391.852	22081.44	21978.23	21663.74	21665.07	1136.221	1130.666	21599.7	21574.08
		VP3	296.8036	46.2022	301.2569	298.8972	296.187	296.1869	48.02854	47.0509	298.4085	296.7268
	q=0.5	VP1	318.5525	110.8355	321.9573	321.9635	318.0418	318.0491	107.8536	107.5718	318.1475	318.2165
		VP2	230264.6	1061.297	230696.7	230697.1	230212	230212.3	901.4073	897.2672	230210.9	230212.5
		VP3	1394.194	61.62084	1393.862	1393.864	1393.788	1393.79	61.36323	61.24069	1393.764	1393.746
	q=0.9	VP1	505.4778	122.5121	663.2076	464.8336	456.6015	455.318	136.7393	91.35663	505.3509	430.988
		VP2	7742.953	1994.445	7992.741	6443.459	5432.174	5666.179	1154.632	788.057	4560.236	4225.774
		VP3	249.0118	63.52442	326.0206	246.8317	244.2024	244.1633	82.62595	60.269	264.5786	241.2559

Simulation results are reported in Table 3.2 where X and ϵ have standard normal distribution. For q=0.5, COBS, RIS with NO and RIS with HD give close results. The RIS with NO estimator gives better results than COBS for VP2 and extreme quantiles as the sample size increases.

Simulation results are reported in Table 3.3 where X has standard normal distribution and ϵ has asymmetric (g=0.2) light tailed distribution. For q=0.5, COBS, RIS with NO and RIS with HD give close results. The RIS with NO estimator gives better results than COBS for VP2 and extreme quantiles as the sample size increases.

Simulation results are reported in Table 3.4 where X has standard normal distribution and ϵ has asymmetric (g=0.5) light tailed distribution. For q=0.5, COBS, RIS with NO and RIS with HD give close results. COBS and RIS with NO estimator can be preferred for extreme quantiles. However, RIS with NO gives better results for VP1 and VP2 when q=0.9.

Simulation results are reported in Table 3.5 where X has standard normal distribution and ϵ has symmetric heavy tailed (h=0.2) distribution. For q=0.5, COBS, RIS with NO and RIS with HD give close results. For extreme quantiles where VP2, RUNLOW with NO estimator competes well with COBS as the sample size increases. However, RIS with NO estimator can also be preferred in here.

Simulation results are reported in Table 3.6 where X has standard normal distribution and ϵ has symmetric heavy tailed (h=0.5) distribution. For q=0.5, COBS, RIS with NO and RIS with HD give close results. For extreme quantiles, RIS with NO must be used. RUNLOW with NO gives better results than COBS. COBS and RIS with NO estimator can be preferred for VP3 and extreme quantiles as the sample size increases.

Simulation results are reported in Table 3.7 where X has standard normal distribution and ϵ has asymmetric heavy tailed (g=0.2, h=0.2) distribution. For q=0.5,

COBS, RIS with NO and RIS with HD give close results. For q=0.9, RIS with NO estimator can be preferred and RUNLOW with NO gives better than COBS for VP2.

Simulation results are reported in Table 3.8 where X has standard normal distribution and ϵ has asymmetric heavy tailed (g=0.5, h=0.5) distribution. For q=0.5, COBS, RIS with NO and RIS with HD give close results. For q=0.9, RIS with NO estimator can be used.

Simulation results are reported in Table 3.9 where X and ϵ have asymmetric heavy tailed (g=0.2, h=0.2) distribution. For q=0.5, COBS, RIS with NO and RIS with HD give close results. For extreme quantiles, COBS and RIS with NO give close results. However, RIS with NO estimator can be preferred for q=0.9. COBS gives better than other methods as the sample size increase.

Simulation results are reported in Table 3.10 where X and ϵ have asymmetric heavy tailed (g=0.5, h=0.5) distribution. For VP1 and VP3, COBS, RIS with NO and RIS with HD give close results when q=0.5. For VP2, the best result is RIS with NO.

3.3 Real Data Example

The real data set is from Census of Canada by Blishen et al. (1971). The prestige data set has 102 observations and two variables. These variables are prestige and income. The observations are occupations. Income variable is average income of incumbents, dollars, in 1971 and prestige variable is Pineo-Porter prestige score for occupation, from a social survey conducted in the mid-1960s.

Figure 3.1 shows the relationship between income and prestige.

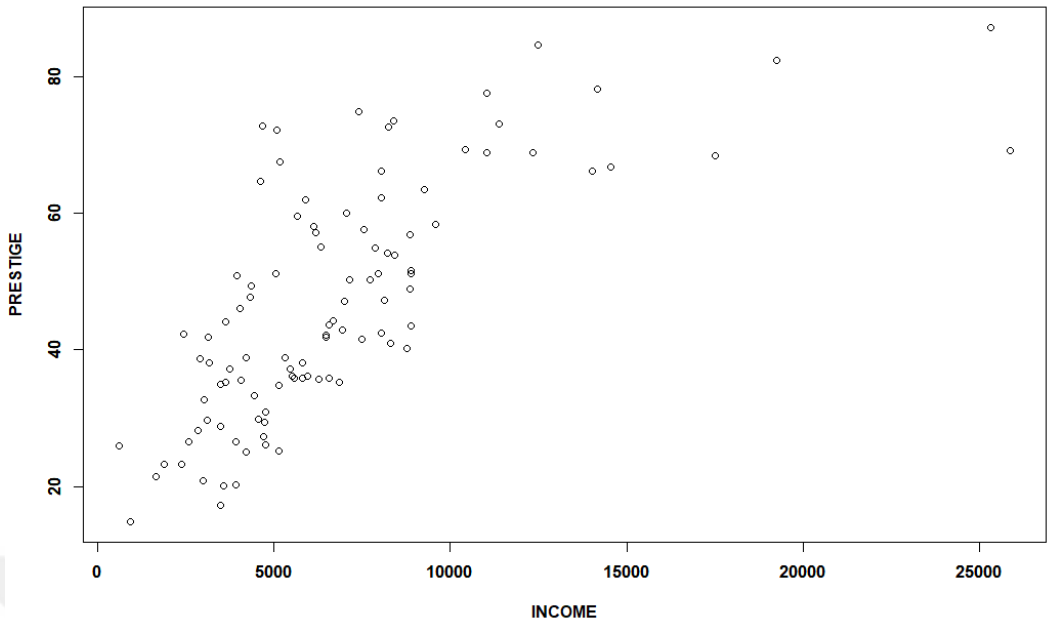

Figure 3.1 Association between income and prestige

Four methods are investigated in Figure 3.2. These methods are RIS-NO with span=1, LOWESS with span=0.8, kernel smoothing with Epanechnikov kernel and COBS with lambda=0. On the other hand, MSE values of these methods are shown in Table 3.11.

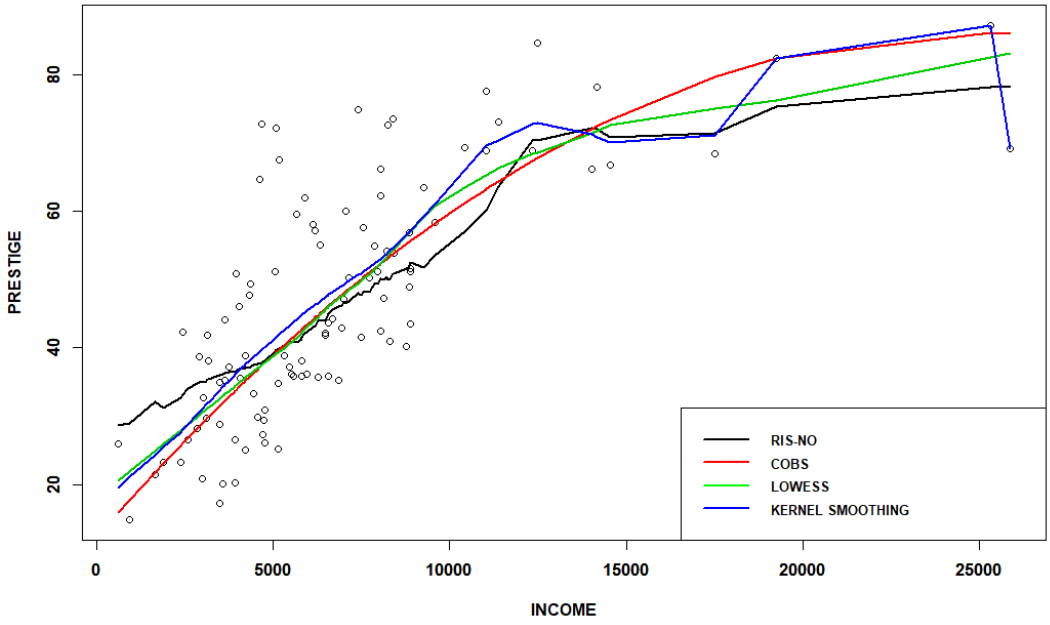

Figure 3.2 Comparisons of four methods (RIS-NO with span=1, LOWESS with span=0.8, Epanechnikov kernel smoothing and COBS with lambda=0)

Table 3.11 MSE values for four methods (RIS-NO with span=1, LOWESS with span=0.8, Epanechnikov kernel smoothing and COBS with lambda=0)

Methods	**MSE Values**
RIS-NO	130.1699
COBS	124.4609
LOWESS	121.952
Kernel Smoothing	112.9521

Four different methods are compared in Figure 3.3. These methods are RIS-NO with span=0.6, LOWESS with span=0.8, kernel smoothing with Epanechnikov kernel and COBS with lambda chosen SIC. On the other hand, MSE values of these methods are shown in Table 3.12.

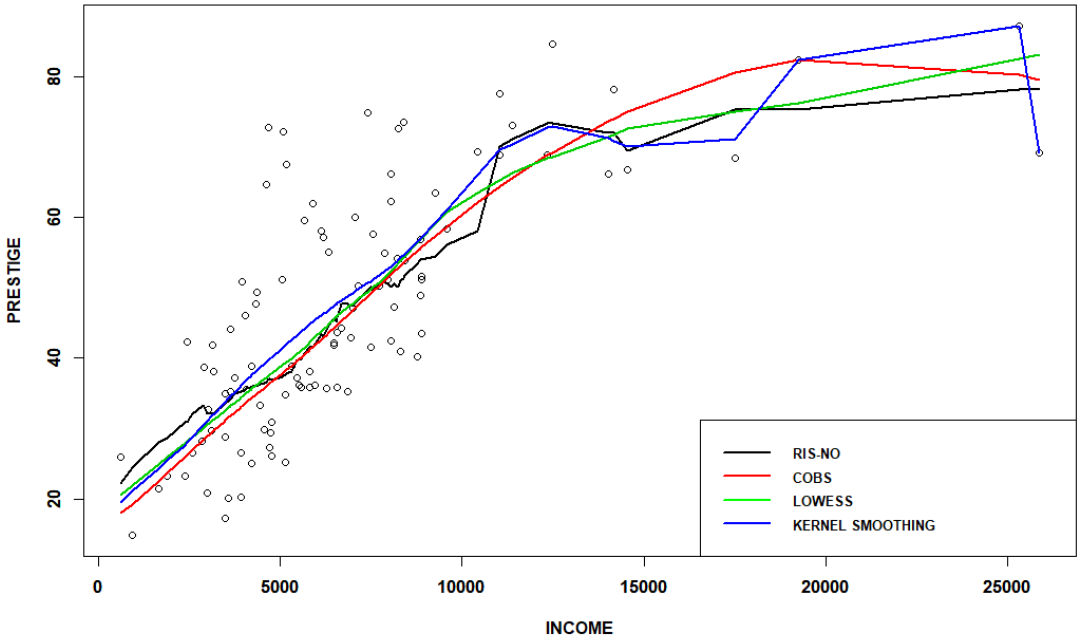

Figure 3.3 Comparisons of four methods (RIS-NO with span=0.6, LOWESS with span=0.8, Epanechnikov kernel smoothing and COBS with lambda=SIC)

Table 3.12 MSE values for four methods (RIS-NO with span=0.6, LOWESS with span=0.8, Epanechnikov kernel smoothing and COBS with lambda=SIC)

Methods	**MSE Values**
RIS-NO	124.3158
COBS	126.0781
LOWESS	121.952
Kernel Smoothing	112.9521

Four different methods are compared in Figure 3.4. These methods are RIS-NO with span=0.6, LOWESS with span=0.8, kernel smoothing with Epanechnikov kernel and COBS with lambda chosen SIC and constrained=increase. On the other hand, MSE values of these methods are shown in Table 3.13.

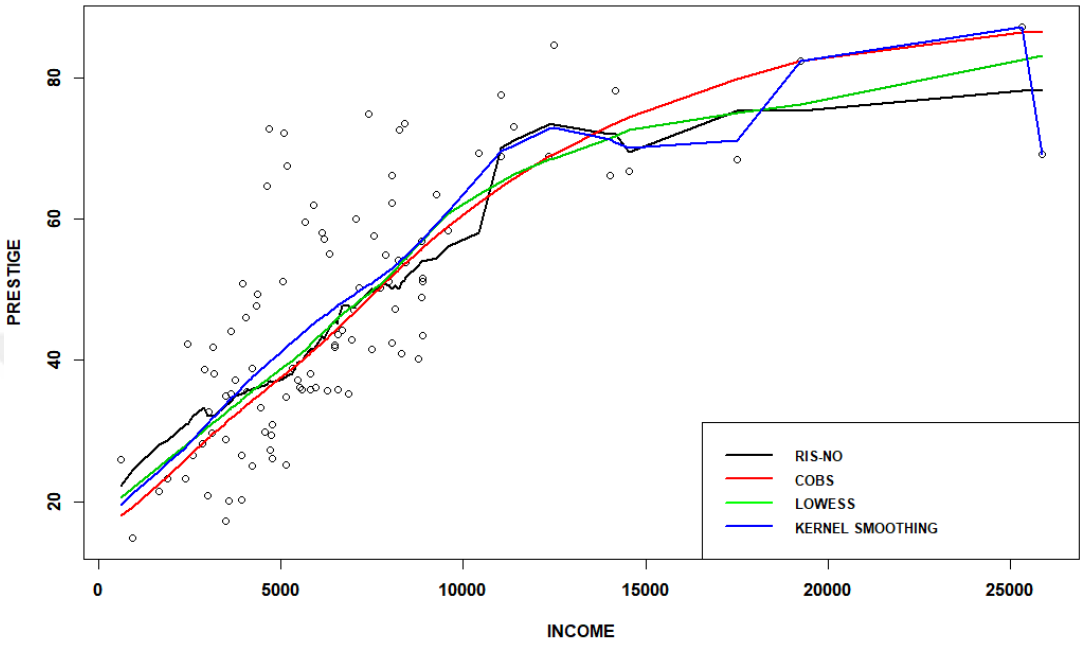

Figure 3.4 Comparisons of four methods (RIS-NO with span=0.6, LOWESS with span=0.8, Epanechnikov kernel smoothing and COBS with lambda=SIC and constrained=increase)

Table 3.13 MSE values for four methods (RIS-NO with span=0.6, LOWESS with span=0.8, Epanechnikov kernel smoothing and COBS with lambda=SIC and constrained=increase)

Methods	**MSE Values**
RIS-NO	124.3158
COBS	127.5089
LOWESS	121.952
Kernel Smoothing	112.9521

Four different methods are compared in Figure 3.5. These methods are RIS-NO with span=0.6, LOWESS with span=0.8, kernel smoothing with Gaussian kernel and COBS with lambda chosen SIC. On the other hand, MSE values of these methods are shown in Table 3.14.

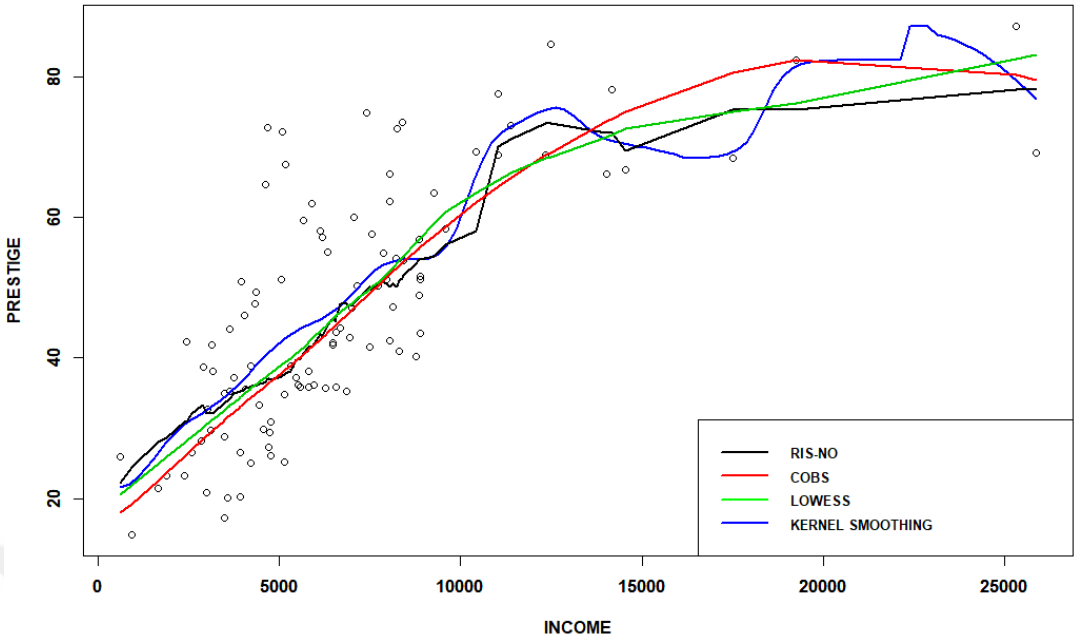

Figure 3.5 Comparisons of four methods (RIS-NO with span=0.6, LOWESS with span=0.8, Gaussian kernel smoothing and COBS with lambda=SIC)

Table 3.14 MSE values for four methods (RIS-NO with span=0.6, LOWESS with span=0.8, Gaussian kernel smoothing and COBS with lambda=SIC)

Methods	**MSE Values**
RIS-NO	124.3158
COBS	126.0781
LOWESS	121.952
Kernel Smoothing	482.6155

The RIS, LOWESS, kernel smoothing and COBS were compared with different parameters. In general, it is observed that prestige status increases as income level increases. When the income level is 15000 or more, it can be said that there is little or no association between prestige and income level.

Two different methods are compared in Figure 3.6. These methods are RIS with NO quantile estimator and quantile value is 0.1 and COBS with lambda=0 and quantile value is 0.1. On the other hand, MSE values of these methods are shown in Table 3.15.

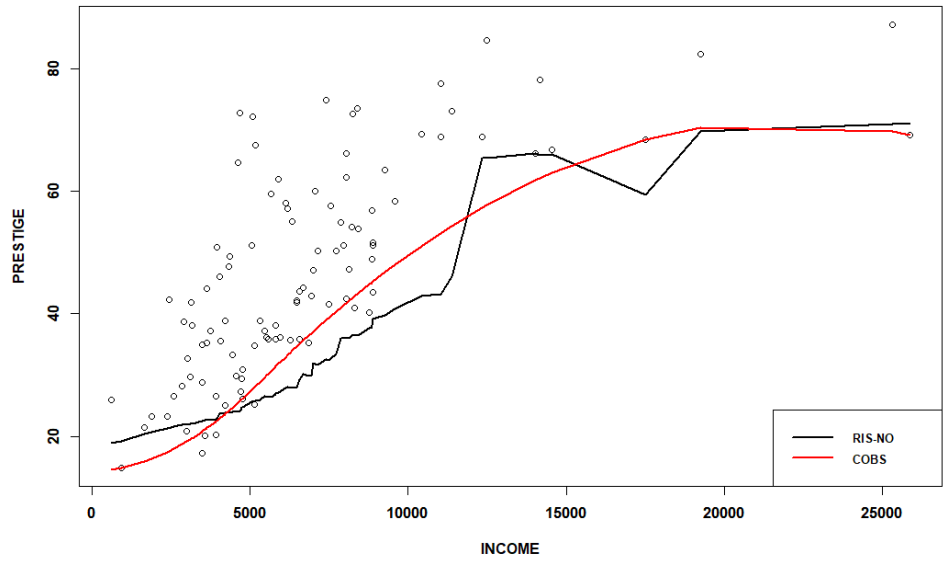

Figure 3.6 Comparisons of two methods (RIS-NO and COBS with lambda=0, quantile value is 0.1)

Table 3.15 MSE values for two methods (RIS-NO and COBS with lambda=0, quantile value is 0.1)

Methods	**MSE Values**
RIS-NO	362.3545
COBS	277.8881

Two different methods are compared in Figure 3.7. These methods are RIS with NO quantile estimator and quantile value is 0.9 and COBS with lambda=0 and quantile value is 0.9. On the other hand, MSE values of these methods are shown in Table 3.16.

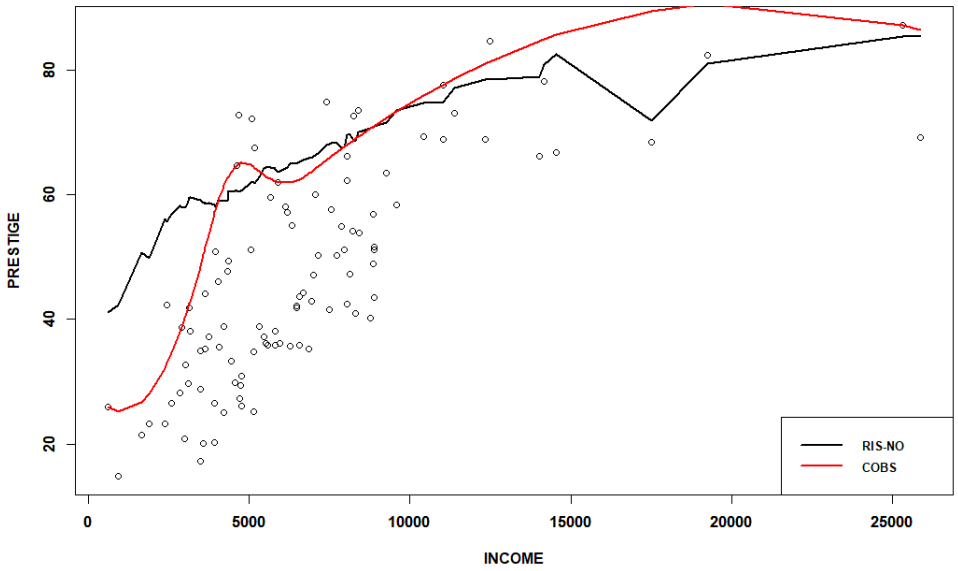

Figure 3.7 Comparisons of two methods (RIS-NO and COBS with lambda=0, quantile value is 0.9)

Table 3.16 MSE values for two methods (RIS-NO and COBS with lambda=0, quantile value is 0.9)

Methods	**MSE Values**
RIS-NO	476.7572
COBS	374.1045

Two different methods are compared in Figure 3.8. These methods are the RIS with HD quantile estimator and quantile value is 0.1 and COBS with lambda=0 and quantile value is 0.1. On the other hand, MSE values of these methods are shown in Table 3.17.

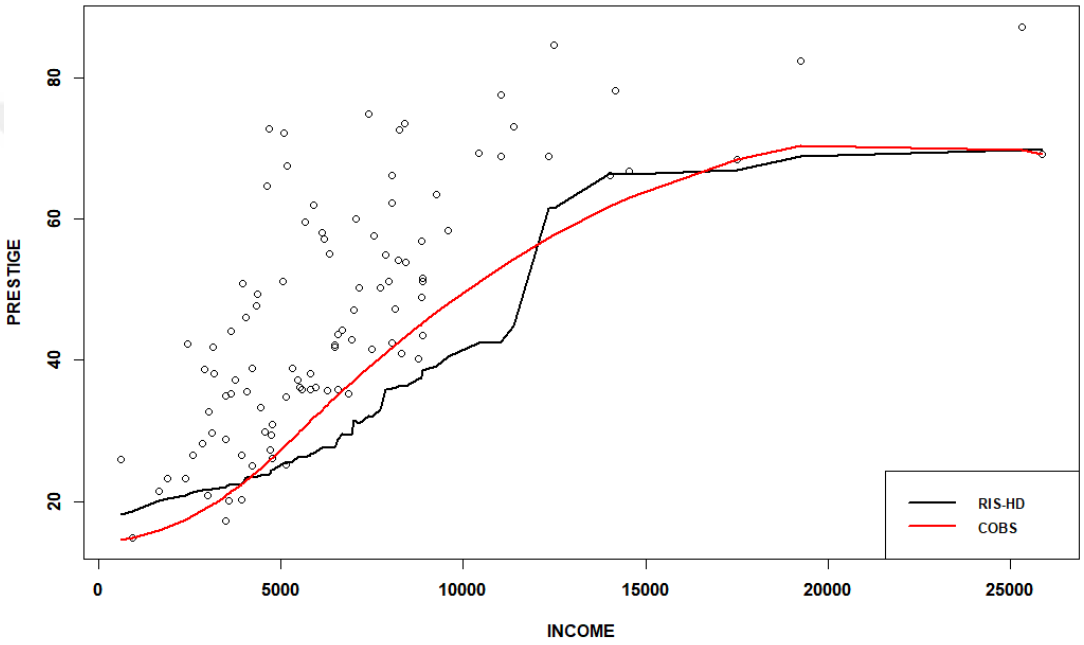

Figure 3.8 Comparisons of two methods (RIS-HD and COBS with lambda=0, quantile value is 0.1)

Table 3.17 MSE values for two methods (RIS-HD and COBS with lambda=0, quantile value is 0.1)

Methods	**MSE Values**
RIS-NO	374.1397
COBS	277.8881

Two different methods are compared in Figure 3.9. These methods RIS with HD quantile estimator and quantile value is 0.9 and COBS with lambda=0 and quantile value is 0.9. On the other hand, MSE values of these methods are shown in Table 3.18.

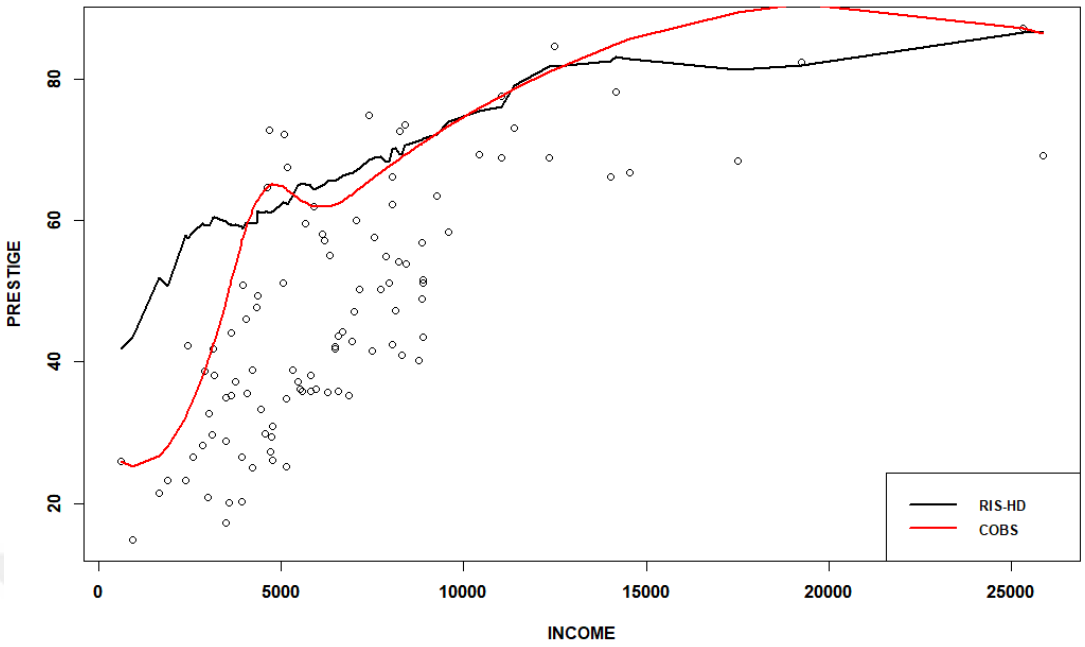

Figure 3.9 Comparisons of two methods (RIS-HD and COBS with lambda=0, quantile value is 0.9)

Table 3.18 MSE Values for two methods (RIS-HD and COBS with lambda=0, quantile value is 0.9)

Methods	**MSE Values**
RIS-NO	508.3362
COBS	374.1045

The RIS and COBS were compared at different quantile points. For q=0.1, when the income level was between 5000 and 20000, there was a positive association between the variables. For q=0.9, there is an increasing association between prestige and income when income is less than 5000.

Four different alternative methods are compared in Figure 3.10. These methods are COBSLOW, COBSRUN, RUNCOBS and RUNLOW. The span values used for each method were 0.2 and the quantile estimator is HD. The quantile values are 0.5. On the other hand, MSE values of these methods are shown in Table 3.19.

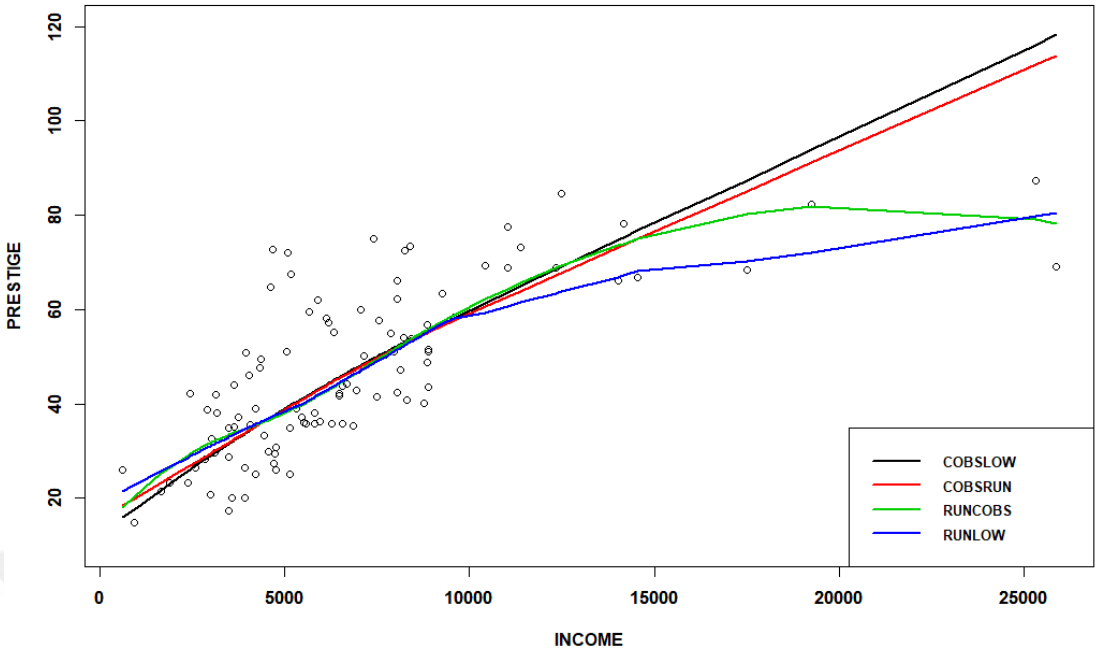

Figure 3.10 Comparisons of alternative methods (quantile estimator is HD and quantile value is 0.5)

Table 3.19 MSE values for alternative methods (quantile estimator is HD and quantile value is 0.5)

Methods	**MSE Values**
COBSLOW	886.0293
COBSRUN	843.8688
RUNCOBS	720.4396
RUNLOW	680.248

Four different alternative methods are compared in Figure 3.11. These methods are COBSLOW, COBSRUN, RUNCOBS and RUNLOW. The span values used for each method are 0.2 and the quantile estimator is HD. The quantile values are 0.1. On the other hand, MSE values of these methods are shown in Table 3.20.

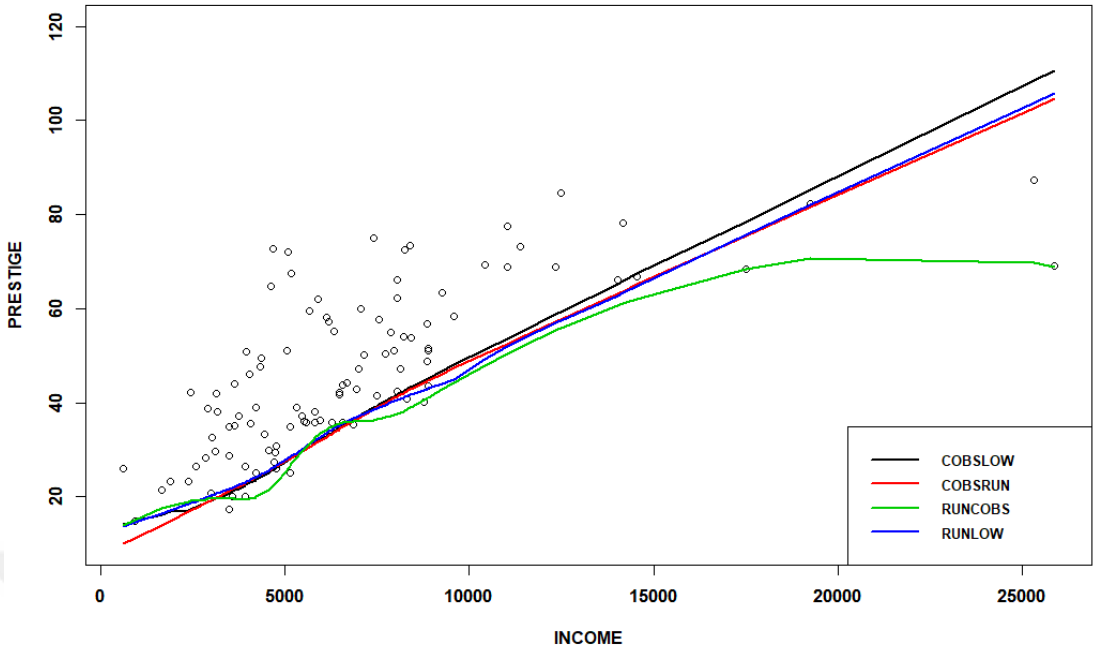

Figure 3.11 Comparisons of alternative methods (quantile estimator is HD and quantile value is 0.1)

Table 3.20 MSE values for alternative methods (quantile estimator is HD and quantile value is 0.1)

Methods	**MSE Values**
COBSLOW	1006.782
COBSRUN	982.844
RUNCOBS	927.1054
RUNLOW	955.9284

Four different alternative methods are compared in Figure 3.12. These methods are COBSLOW, COBSRUN, RUNCOBS and RUNLOW. The span values used for each method are 0.2 and the quantile estimator is HD. The quantile values are 0.9. On the other hand, MSE values of these methods are shown in Table 3.21.

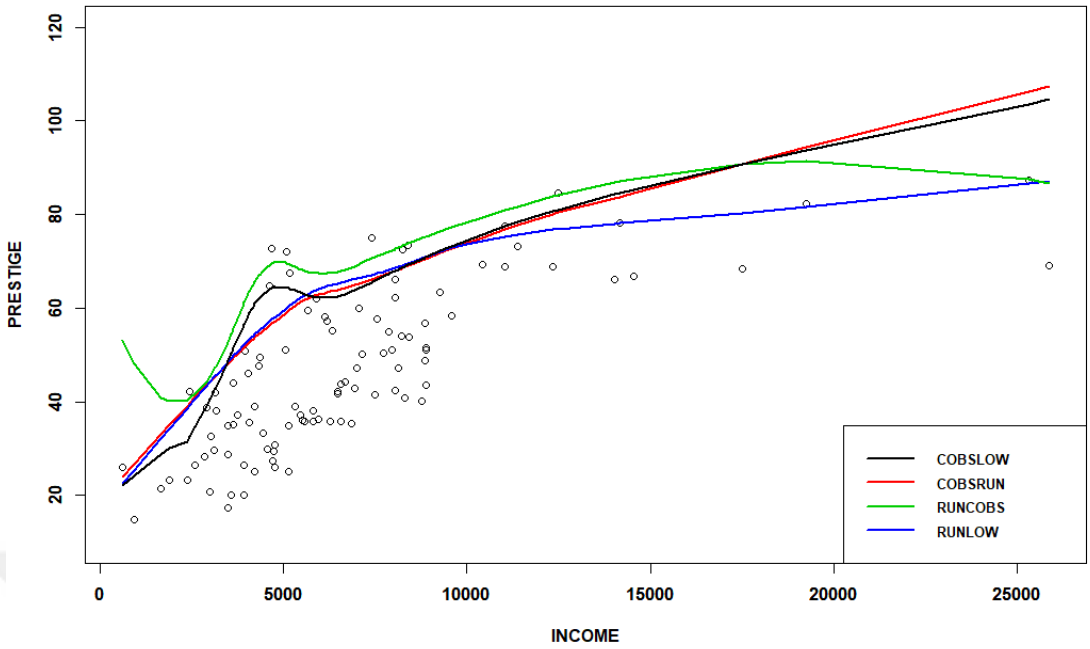

Figure 3.12 Comparisons of alternative methods (quantile estimator is HD and quantile value is 0.9)

Table 3.21 MSE values for alternative methods (quantile estimator is HD and quantile value is 0.9)

Methods	**MSE Values**
COBSLOW	1034.648
COBSRUN	999.3591
RUNCOBS	1092.653
RUNLOW	932.4174

Four different alternative methods are compared in Figure 3.13. These methods are COBSLOW, COBSRUN, RUNCOBS and RUNLOW. The span values used for each method are 0.2 and the quantile estimator is NO. The quantile values are 0.5. On the other hand, MSE values of these methods are shown in Table 3.22.

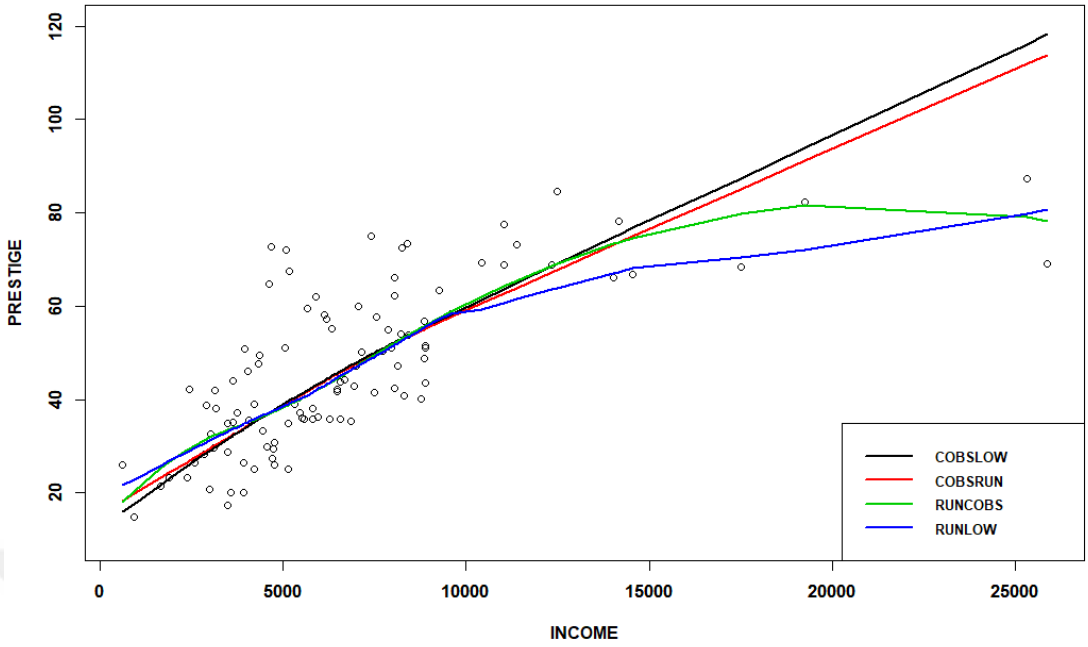

Figure 3.13 Comparisons of alternative methods (quantile estimator is NO and quantile value is 0.5)

Table 3.22 MSE values for alternative methods (quantile estimator is NO and quantile value is 0.5)

Methods	**MSE Values**
COBSLOW	886.0293
COBSRUN	843.8688
RUNCOBS	720.4396
RUNLOW	680.0873

Four different alternative methods are compared in Figure 3.14. These methods are COBSLOW, COBSRUN, RUNCOBS and RUNLOW. The span values used for each method are 0.2 and the quantile estimator is NO. The quantile values are 0.1. On the other hand, MSE values of these methods are shown in Table 3.23.

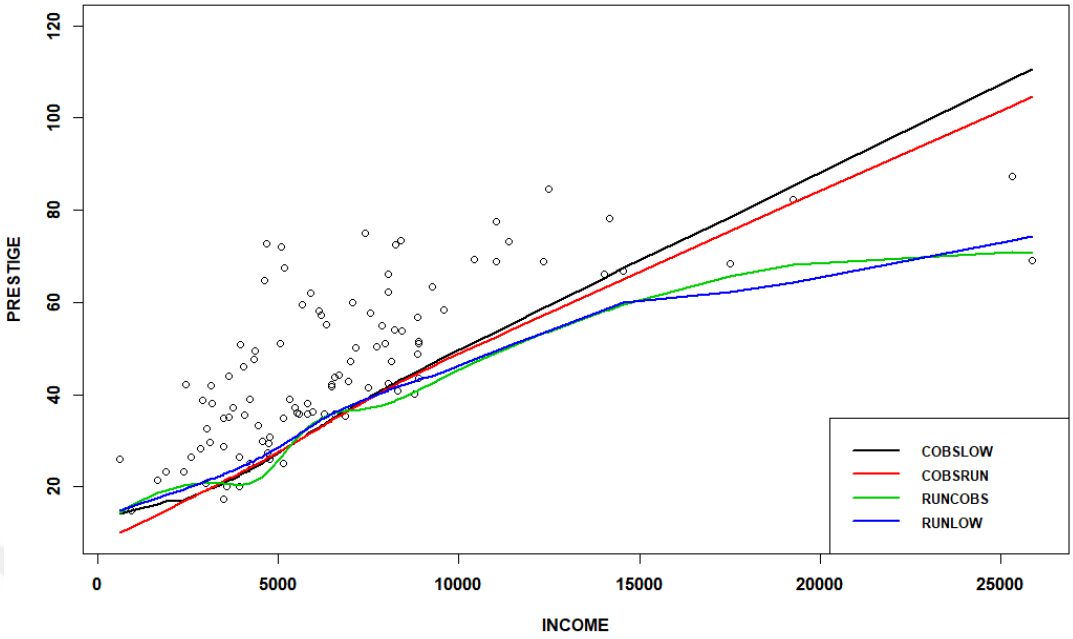

Figure 3.14 Comparisons of alternative methods (quantile estimator is NO and quantile value is 0.1)

Table 3.23 MSE values for alternative methods (quantile estimator is NO and quantile value is 0.1)

Methods	**MSE Values**
COBSLOW	1006.782
COBSRUN	982.6199
RUNCOBS	888.0313
RUNLOW	826.1281

Four different alternative methods are compared in Figure 3.15. These methods are COBSLOW, COBSRUN, RUNCOBS and RUNLOW. The span values used for each method are 0.2 and the quantile estimator is NO. The quantile values are 0.9. On the other hand, MSE values of these methods are shown in Table 3.24.

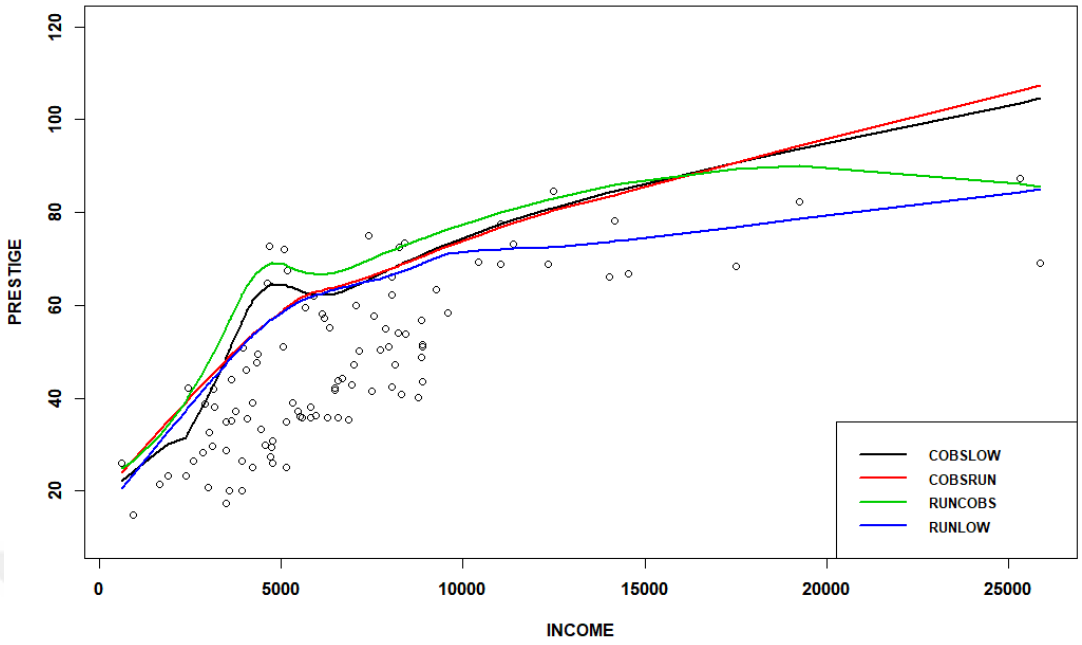

Figure 3.15 Comparisons of alternative methods (quantile estimator is NO and quantile value is 0.9)

Table 3.24 MSE values for alternative methods (quantile estimator is NO and quantile value is 0.9)

Methods	**MSE Values**
COBSLOW	1034.648
COBSRUN	999.4557
RUNCOBS	1080.119
RUNLOW	861.9016

Alternative methods were compared for different quantile points. As the income level increases for q=0.5, it can be said that there is a positive association between the variables in COBSLOW and COBSRUN methods. In RUNCOBS and RUNLOW methods, there is an increasing association when income is below 15000. When it is 15000 or more, it can be said that there is little or no association. For q=0.1, it is observed that there is a positive association between the variables in COBSLOW, COBSRUN and RUNLOW methods. In the RUNCOBS method, there is an increasing association when income is below 20000. When it is 20000 and over, it can be said that there is little or no association. There is an increasing association between the variables for q=0.9.

3.3.1 Prediction for Real Data

Table 3.25 shows the predicted values of new observations for nonparametric regression methods. Empty cells under observed Y values column corresponds to X values that are not included in the original data set.

Table 3.25 Predicted values of new observations for nonparametric regression methods

X Values	Kernel Smoothing	COBS	LOWESS	RIS		Observed Y Values
				HD	NO	
100	18.98944	12.86257	18.29912	26.98898	27.09654	
400	20.36111	14.62521	19.58155	27.88682	27.98479	
611	21.32585	15.85207	20.48193	28.65451	28.68170	
700	21.73278	16.36637	20.86127	29.36563	29.38397	
918	22.72957	17.61812	21.78891	28.92744	28.95649	14.8
1656	26.10462	21.77149	24.90739	32.08584	32.09165	21.5
2400	29.50476	25.82699	28.02151	33.09929	33.09768	
3500	34.50801	31.58086	32.64856	35.95203	35.94757	
4000	36.76438	34.10075	34.78762	36.88477	36.89475	
5000	41.20083	38.96145	38.94108	39.44755	39.45181	
6600	47.97789	46.24193	46.06508	45.02082	45.02473	
7405	51.22906	49.67379	49.43464	47.92653	47.92246	74.9
8500	55.44386	54.09359	55.03662	50.83423	50.82740	
9271	58.25678	57.03386	59.10804	51.66207	51.66086	63.4
10000	60.78513	59.68341	61.81173	56.07528	56.10738	
11377	65.11336	64.34190	65.98488	63.60286	63.54700	73.1
12351	67.76686	67.36364	68.08655	70.23215	70.35371	68.8
13000	69.34520	69.25135	69.29129	71.11793	71.29170	
13500	70.46362	70.63708	70.31785	71.91082	72.06094	
14558	72.57751	73.37250	72.72001	70.19382	70.75000	66.7
15000	73.36857	74.43613	73.71795	70.19382	70.75000	
16500	75.70152	77.69794	74.85004	71.14803	71.66875	
18000	77.57684	80.42252	75.27469	75.35000	75.35000	
20000	79.50942	83.21960	77.10947	75.35000	75.35000	
23000	81.48270	85.62444	80.61614	78.15000	78.15000	
25000	82.42616	86.03382	82.95773	78.15000	78.15000	
25879	82.79964	85.91163	83.95416	78.15000	78.15000	69.1
26000	82.85055	85.88036	84.08934	78.15000	78.15000	
27000	83.27137	85.48813	85.18886	78.15000	78.15000	
28000	83.69218	84.85712	86.26180	78.15000	78.15000	

CHAPTER FOUR
CONCLUSION

In this thesis, four nonparametric regression methods and some alternative two-step methods that can be used to model nonlinear relationship were compared. After introducing the fundamental structure of the conventional nonparametric regression methods a simulation study was carried out by using g-h distributions with different skewness and kurtosis levels. The performance of the methods were also monitored with graphs using a real data set and the predicted values of the new observations that are not included in the original data set were computed.

In the simulation study, the methods were compared in terms of mean squared error (MSE). In general, the COBS and the running interval smoother with NO quantile estimator gave better results than the others. That means they are more efficient than the other methods. These two methods gave similar results for all g-h distributions but increasing the kurtosis and skewness (non-normality) resulted in better efficiency values for RIS with NO quantile estimator.

Performances of all methods were also compared by sketching graphs. The relationship among variables might change depending on the quantile points considered. Degree of association might change depending on the quantile point. Using more than one quantile point let the researcher observe different patterns and RIS and COBS enable this.

Fitted values of new observations were computed. In the literature, there were R functions in which new observations for COBS, kernel smoothing and LOWESS can be predicted but there was no such a function for RIS. In the study, an R function for predicting the fitted value of a new observation for RIS is developed. A researcher can modify the quantile point and the estimator by using this function.

Constrained B-Spline Smoothing (COBS) and Running Interval Smoother (RIS) with NO quantile estimator are suggested as efficient, flexible and robust tools of modelling nonlinear relationship among variables.

REFERENCES

Bosch, G. (1995). *Flexible working time; collective bargaining and government intervention.* Paris: OECD.

Cheng, F., & Wang, X., & Barsky, B.A. (2000). Quadratic b-spline curve interpolation. *Computers and Mathematics with Applications, 41*, 39-50.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association, 74,* 829–836.

Cleveland, W. S., & Devlin, S. J. & Grosse, E. (1991). Regression by local fitting. *Journal of Econometrics, 37,* 87-114.

Cleveland, W. S., & Grosse, E. (1991). Computational methods for local regression. *Statistics and Computing, 1,* 47-62.

Gasser T., & Müller H. G. (1979). Kernel estimation of regression functions. *Lecture Notes in Mathematics, 757,* 24-68.

Hafen, R. P. (2010). *local regression models: advancements, applications, and new methods.* Phd Thesis, Purdue University, Indiana.

Hardle, W. (1994). Applied nonparametric methods. *The Handbook of Econometrics, 4,* 2295-2339.

Harrell, F.E., & Davis, C.E. (1982). A new distribution-free quantile estimator. *Biometrika, 69,* 635-640.

Hastie, T., & Tibshirani R. (1990). *Generalized additive models*. London: Chapman and Hall.

Hastie, T., & Tibshirani, R., & Friedman, J. (2008). *The elements of statistical learning* (2nd ed.). California: Stanford.

He, X., & Ng, P. (1999). COBS: Qualitatively constrained smoothing via linear programming. *Computational Statistics, 14,* 315-337.

Hoaglin, D. C. (1985). Summarizing shape numerically: The g-and-h distribution. In *Exploring data tables trends and shapes.* New York: Wiley.

Joyner, W. B., & Boore, D. M., & Porcella, R. D. (1981). Peak horizontal acceleration and velocity from strong-motion records including records from the 1979 Imperial Valley, California earthquake. *Bulletin of the Seismological Society of America, 71,* 2011-2038.

Koenker, R., & Bassett, G. (1978). Regression quantiles. *Econometrica, 46, 33-50.*

Koenker, R., & Ng, P., & Portnoy, S. (1994), Quantile smoothing splines. *Biometrika, 81,* 673-680.

Koenker, R., & Ng, P. (2005). Inequality constrained quantile regression. *The Indian Journal of Statistics, 67,* 418-440.

Koenker, R., & Hallock, K. F. (2001). Quantile regression. *Journal of Economic Perspectives, 15,* 143-156.

Laurini, M. P., & Moura, M. (2009). Constrained smoothing B-splines for the term structure of interest rates. *Insurance: Mathematics and Economics, 46,* 339–350.

Lyche, T., & Mørken, K. (2008). *Spline methods draft.* Oslo: University of Oslo.

Marx, B. D., & Eilers, P. H. (1996). Flexible smoothing with b-splines and penalties. *Statistical Science, 11,* 89-121.

Meyer, K. (2005). Random regression analyses using b-splines to model growth of Australian angus cattle. *EDP Sciences, 37,* 473–500.

Nadaraya, E.A. (1964). On estimating regression. *Teoriya Veroyatnostei i ee Primeniya, 9,* 157-159.

Navruz, G. (2019). *Analysis of robust approaches in the two independent samples case*. Phd Thesis, Dokuz Eylül University, İzmir.

Schucany, W. R. (2004*).* Kernel smoothers: An overview of curve estimators for the first graduate course in nonparametric statistics. *Institute of Mathematical Statistics, 19,* 663–675.

Tezcan, N. (2009). *Tahmine parametrik olmayan regresyon yöntemiyle yaklaşım.* Phd Thesis, İstanbul University, İstanbul.

Ucal, M. Ş. (2006). Ekonometrik model seçim kriterleri üzerine kısa bir inceleme. *C.Ü. İktisadi ve İdari Bilimler Dergisi, 7,* 41-57.

Wilcox, R. R. (2012). *Introduction to robust estimation and hypothesis testing (3rd ed.).* San Diego, CA: Academic Press.

Wilcox, R. R. (2012*). Modern statistics for the social and behavioral sciences*. Los Angeles: CRC Press.

Wilcox, R. R. (2016). Comparisons of two quantile regression smoothers. *Journal of Modern Applied Statistical Methods, 5,* 62-77.

APPENDICES

R codes for simulation results

```
K=10000
n=25

mse_cobslow<-c()
mse_cobs<-c()
mse_runcobshd<-c()
mse_runcobsno<-c()
mse_cobsrunhd<-c()
mse_cobsrunno<-c()
mse_runhd<-c()
mse_runno<-c()
mse_runlowhd<-c()
mse_runlowno<-c()

max_cobslow<-c()
max_cobs<-c()
max_runcobshd<-c()
max_runcobsno<-c()
max_cobsrunhd<-c()
max_cobsrunno<-c()
max_runhd<-c()
max_runno<-c()
max_runlowhd<-c()
max_runlowno<-c()

for(k in 1:K){

  x<-ghdist(n,g=0,h=0) #g-and-h distribution
```

```
e<-ghdist(n,g=0,h=0) #g-and-h distribution

vp1<-1
vp2<-abs(x)+1
vp3<-1/(abs(x)+1)

y<-x+(e*vp1)

cobslow_est<-cobslow(x,y,qval=.5,plotit=F,pyhat=T,LP=TRUE)  #cobs and lowess

cobs_est<-qsmcobs(x,y,qval=.5,plotit = F,FIT=F)  #COBS

runcobs_hd_est<-runcobshd(x,y,qval=.5,plotit=F,pyhat=T,LP=TRUE)      #ris and cobs with HD

runcobs_no_est<-runcobsno(x,y,qval=.5,plotit=F,pyhat=T,LP=TRUE)      #ris and cobs with NO

cobsrun_hd_est<-cobsrunhd(x,y,qval=.5,plotit=F,pyhat=T,LP=TRUE)    #cobs and ris with HD

cobsrun_no_est<-cobsrunno(x,y,qval=.5,plotit=F,pyhat=T,LP=TRUE)    #cobs and ris with No

run_hd_est<-rungen(x,y,q=.5,LP=F,plotit = F,pyhat=T,est=hd)  #RIS with HD

run_no_est<-rungen(x,y,q=.5,LP=F,plotit = F,pyhat=T,est=no)  #RIS with NO

runlow_hd_est<-qhdsm(x,y,q=.5,LP=T,plotit=F,pyhat = T) #ris and lowess with HD

runlow_no_est<-qhdsmno(x,y,q=.5,LP=T,plotit=F,pyhat = T) #ris and lowess with NO
```

```
  mse_cobslow[k]<-sum((cobslow_est-y)^2)
  mse_cobs[k]<-sum((cobs_est$yhat-y)^2)
  mse_runcobshd[k]<-sum((runcobs_hd_est-y)^2)
  mse_runcobsno[k]<-sum((runcobs_no_est-y)^2)
  mse_cobsrunhd[k]<-sum((cobsrun_hd_est-y)^2)
  mse_cobsrunno[k]<-sum((cobsrun_no_est-y)^2)
  mse_runhd[k]<-sum((run_hd_est$output-y)^2)
  mse_runno[k]<-sum((run_no_est$output-y)^2)
  mse_runlowhd[k]<-sum((runlow_hd_est-y)^2)
  mse_runlowno[k]<-sum((runlow_no_est-y)^2)

  max_cobslow[k]<-max(abs(cobslow_est-y))
  max_cobs[k]<-max(abs(cobs_est$yhat-y))
  max_runcobshd[k]<-max(abs(runcobs_hd_est-y))
  max_runcobsno[k]<-max(abs(runcobs_no_est-y))
  max_cobsrunhd[k]<-max(abs(cobsrun_hd_est-y))
  max_cobsrunno[k]<-max(abs(cobsrun_no_est-y))
  max_runhd[k]<-max(abs(run_hd_est$output-y))
  max_runno[k]<-max(abs(run_no_est$output-y))
  max_runlowhd[k]<-max(abs(runlow_hd_est-y))
  max_runlowno[k]<-max(abs(runlow_no_est-y))

  print(k)

}

sum(mse_cobslow)/(n*K)
sum(mse_cobs)/(n*K)
sum(mse_runcobshd)/(n*K)
sum(mse_runcobsno)/(n*K)
```

```
sum(mse_cobsrunhd)/(n*K)
sum(mse_cobsrunno)/(n*K)
sum(mse_runhd)/(n*K)
sum(mse_runno)/(n*K)
sum(mse_runlowhd)/(n*K)
sum(mse_runlowno)/(n*K)

sum(max_cobslow)/K
sum(max_cobs)/K
sum(max_runcobshd)/K
sum(max_runcobsno)/K
sum(max_cobsrunhd)/K
sum(max_cobsrunno)/K
sum(max_runhd)/K
sum(max_runno)/K
sum(max_runlowhd)/K
sum(max_runlowno)/K
```

#R codes for figure 3.2 and table 3.11

```
library(carData)
data("Prestige")

x<-Prestige$income
y<-Prestige$prestige

plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)
```

```
sbs<-cobs(x,y,lambda=0)
#MSE
sum((y-sbs$fitted)^2)/102

fitlow<-lplot(x,y,pyhat=TRUE,fr=0.8)
##MSE
sum((y-fitlow$yhat.values)^2)/102

fitker<-kerreg(x,y,pyhat=TRUE,np=0)
##MSE
sum((y[orderx]-fitker)^2)/102

fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=1,est=no,q=0.5)
#MSE
sum((y-fitrun$output)^2)/102

lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx],sbs$fitted[orderx],col=2,lwd=2)
lines(x[orderx],fitlow$yhat.values[orderx],col=3,lwd=2)
lines(x[orderx],fitker,col=4,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS","LOWESS","KERNEL
SMOOTHING"),col=c("black","red","green","blue"),lty=1,cex=0.8,y.intersp=0.6,lw
d=2,text.font=2)
```

#R codes for figure 3.3 and table 3.12

```
library(carData)
data("Prestige")

x<-Prestige$income
y<-Prestige$prestige
```

```
plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)

sbs<-cobs(x,y,lambda=-1)
#MSE
sum((y-sbs$fitted)^2)/102

fitlow<-lplot(x,y,pyhat=TRUE,fr=0.8)
##MSE
sum((y-fitlow$yhat.values)^2)/102

fitker<-kerreg(x,y,pyhat=TRUE,np=0)
##MSE
sum((y[orderx]-fitker)^2)/102

fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=0.6,est=no,q=0.5)
#MSE
sum((y-fitrun$output)^2)/102

lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx], sbs$fitted[orderx],col=2,lwd=2)
lines(x[orderx],fitlow$yhat.values[orderx],col=3,lwd=2)
lines(x[orderx],fitker,col=4,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS","LOWESS","KERNEL
SMOOTHING"),col=c("black","red","green","blue"),lty=1,cex=0.8,y.intersp=0.6,lw
d=2,text.font=2)
```

#R codes for figure 3.4 and table 3.13

```
library(carData)
data("Prestige")

x<-Prestige$income
y<-Prestige$prestige

plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)

sbs<-cobs(x,y,lambda=0,constrained=”increase”)
#MSE
sum((y-sbs$fitted)^2)/102

fitlow<-lplot(x,y,pyhat=TRUE,fr=0.8)
##MSE
sum((y-fitlow$yhat.values)^2)/102

fitker<-kerreg(x,y,pyhat=TRUE,np=0)
##MSE
sum((y[orderx]-fitker)^2)/102

fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=0.6,est=no,q=0.5)
#MSE
sum((y-fitrun$output)^2)/102

lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx],sbs$fitted[orderx],col=2,lwd=2)
```

```
lines(x[orderx],fitlow$yhat.values[orderx],col=3,lwd=2)
lines(x[orderx],fitker,col=4,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS","LOWESS","KERNEL
SMOOTHING"),col=c("black","red","green","blue"),lty=1,cex=0.8,y.intersp=0.6,lw
d=2,text.font=2)
```

#R codes for figure 3.5 and table 3.14

```
library(carData)
data("Prestige")

x<-Prestige$income
y<-Prestige$prestige

plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)

sbs<-cobs(x,y,lambda=-1)
#MSE
sum((y-sbs$fitted)^2)/102

fitlow<-lplot(x,y,pyhat=TRUE,fr=0.8)
##MSE
sum((y-fitlow$yhat.values)^2)/102

lines(ksmooth(x,y,kernel="normal",bandwidth = 2000),col=4,lwd=2)
kerrr<-ksmooth(x,y,kernel="normal",bandwidth = 2000)
#MSE
sum((y[orderx]-kerrr$y[orderx])^2)/102
```

```
fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=0.6,est=no,q=0.5)
#MSE
sum((y-fitrun$output)^2)/102

fitcobs<-qsmcobs(x,y,lambda=0,FIT=FALSE,qval=0.5)
#MSE
sum((y-fitcobs$yhat)^2)/102

lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx],sbs$fitted[orderx],col=2,lwd=2)
lines(x[orderx],fitlow$yhat.values[orderx],col=3,lwd=2)
lines(x[orderx],fitker,col=4,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS","LOWESS","KERNEL
SMOOTHING"),col=c("black","red","green","blue"),lty=1,cex=0.8,y.intersp=0.6,lw
d=2,text.font=2)
```

#R codes for figure 3.6 and table 3.15

```
library(carData)
data("Prestige")

x<-Prestige$income
y<-Prestige$prestige

plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)

sbs<-cobs(x,y,lambda=0,tau=0.1)
#MSE
```

```
sum((y-sbs$fitted)^2)/102

fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=1,est=no,q=0.1)
#MSE
sum((y-fitrun$output)^2)/102

lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx],sbs$fitted[orderx],col=2,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS"),col=c("black","red"),lty=1,
cex=0.8, y.intersp=0.6, lwd=2, text.font=2)
```

#R codes for figure 3.7 and table 3.16

```
library(carData)
data("Prestige")

x<-Prestige$income
y<-Prestige$prestige

plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)

sbs<-cobs(x,y,lambda=0,tau=0.9)
#MSE
sum((y-sbs$fitted)^2)/102

fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=1,est=no,q=0.9)
#MSE
sum((y-fitrun$output)^2)/102
```

```
lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx],sbs$fitted[orderx],col=2,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS"),col=c("black","red"),lty=1,
cex=0.8, y.intersp=0.6, lwd=2, text.font=2)
```

#R codes for figure 3.8 and table 3.17

```
library(carData)
data("Prestige")

x<-Prestige$income
y<-Prestige$prestige

plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)

sbs<-cobs(x,y,lambda=0,tau=0.1)
#MSE
sum((y-sbs$fitted)^2)/102

fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=1,est=hd,q=0.1)
#MSE
sum((y-fitrun$output)^2)/102

lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx],sbs$fitted[orderx],col=2,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS"),col=c("black","red"),lty=1,
cex=0.8, y.intersp=0.6, lwd=2, text.font=2)
```

#R codes for figure 3.9 and table 3.18

```
library(carData)
data("Prestige")

x<-Prestige$income
y<-Prestige$prestige

plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)

sbs<-cobs(x,y,lambda=0,tau=0.9)
#MSE
sum((y-sbs$fitted)^2)/102

fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=1,est=hd,q=0.9)
#MSE
sum((y-fitrun$output)^2)/102

lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx],sbs$fitted[orderx],col=2,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS"),col=c("black","red"),lty=1,
cex=0.8, y.intersp=0.6, lwd=2, text.font=2)
```

#R codes for figure 3.10 and table 3.19

```
x<-Prestige$income
y<-Prestige$prestige

orderx<-order(x)
```

```
fitcobslow<-cobslow(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.5)
#mse
sum((y-fitcobslow)^2)/102

fitcobsrun<-cobsrunhd(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.5)
#mse
sum((y-fitcobsrun)^2)/102

fitruncobs<-runcobshd(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.5)
#mse
sum((y-fitruncobs)^2)/102

fitrunlow<-qhdsm(x,y,LP=TRUE,pyhat = TRUE,fr=0.2,qval = 0.5)
#mse
sum((y-fitrunlow)^2)/102

plot(x,y,ylim=c(10,120),xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

lines(x[orderx],fitcobslow,lwd=2)
lines(x[orderx],fitcobsrun,col=2,lwd=2)
lines(x[orderx],fitruncobs,col=3,lwd=2)
lines(x[orderx],fitrunlow,col=4,lwd=2)
legend("bottomright",legend=c("COBSLOW","COBSRUN","RUNCOBS","RUNLO
W"),col=1:4,lty=1,cex=0.8,y.intersp=0.6,lwd=2,text.font = 2)
```

#R codes for figure 3.9 and table 3.18

```
library(carData)
data("Prestige")
```

```
x<-Prestige$income
y<-Prestige$prestige

plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)

sbs<-cobs(x,y,lambda=0,tau=0.9)
#MSE
sum((y-sbs$fitted)^2)/102

fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=1,est=hd,q=0.9)
#MSE
sum((y-fitrun$output)^2)/102

lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx],sbs$fitted[orderx],col=2,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS"),col=c("black","red"),lty=1,
cex=0.8, y.intersp=0.6, lwd=2, text.font=2)
```

#R codes for figure 3.11 and table 3.20

```
x<-Prestige$income
y<-Prestige$prestige

orderx<-order(x)

fitcobslow<-cobslow(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.1)
#mse
sum((y-fitcobslow)^2)/102
```

```
fitcobsrun<-cobsrunhd(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.1)
#mse
sum((y-fitcobsrun)^2)/102

fitruncobs<-runcobshd(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.1)
#mse
sum((y-fitruncobs)^2)/102

fitrunlow<-qhdsm(x,y,LP=TRUE,pyhat = TRUE,fr=0.2,qval = 0.1)
#mse
sum((y-fitrunlow)^2)/102

plot(x,y,ylim=c(10,120),xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

lines(x[orderx],fitcobslow,lwd=2)
lines(x[orderx],fitcobsrun,col=2,lwd=2)
lines(x[orderx],fitruncobs,col=3,lwd=2)
lines(x[orderx],fitrunlow,col=4,lwd=2)
legend("bottomright",legend=c("COBSLOW","COBSRUN","RUNCOBS","RUNLO
W"),col=1:4,lty=1,cex=0.8,y.intersp=0.6,lwd=2,text.font = 2)
```

#R codes for figure 3.9 and table 3.18

```
library(carData)
data("Prestige")

x<-Prestige$income
y<-Prestige$prestige

plot(x,y,xlab="INCOME",ylab="PRESTIGE",font.lab=2)
```

```
axis(1, font=2)
axis(2, font=2)

orderx<-order(x)

sbs<-cobs(x,y,lambda=0,tau=0.9)
#MSE
sum((y-sbs$fitted)^2)/102

fitrun<-rungen(x,y,pyhat=TRUE,LP=FALSE,fr=1,est=hd,q=0.9)
#MSE
sum((y-fitrun$output)^2)/102

lines(x[orderx],fitrun$output[orderx],col=1,lwd=2)
lines(x[orderx],sbs$fitted[orderx],col=2,lwd=2)
legend("bottomright",legend=c("RIS-NO","COBS"),col=c("black","red"),lty=1,
cex=0.8, y.intersp=0.6, lwd=2, text.font=2)
```

#R codes for figure 3.12 and table 3.21

```
x<-Prestige$income
y<-Prestige$prestige

orderx<-order(x)

fitcobslow<-cobslow(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.9)
#mse
sum((y-fitcobslow)^2)/102

fitcobsrun<-cobsrunhd(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.9)
#mse
sum((y-fitcobsrun)^2)/102
```

```
fitruncobs<-runcobshd(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.9)
#mse
sum((y-fitruncobs)^2)/102

fitrunlow<-qhdsm(x,y,LP=TRUE,pyhat = TRUE,fr=0.2,qval = 0.9)
#mse
sum((y-fitrunlow)^2)/102

plot(x,y,ylim=c(10,120),xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

lines(x[orderx],fitcobslow,lwd=2)
lines(x[orderx],fitcobsrun,col=2,lwd=2)
lines(x[orderx],fitruncobs,col=3,lwd=2)
lines(x[orderx],fitrunlow,col=4,lwd=2)
legend("bottomright",legend=c("COBSLOW","COBSRUN","RUNCOBS","RUNLO
W"),col=1:4,lty=1,cex=0.8,y.intersp=0.6,lwd=2,text.font = 2)
```

#R codes for figure 3.13 and table 3.22

```
x<-Prestige$income
y<-Prestige$prestige

orderx<-order(x)

fitcobslow<-cobslow(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.5)
#mse
sum((y-fitcobslow)^2)/102

fitcobsrun<-cobsrunno(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.5)
#mse
sum((y-fitcobsrun)^2)/102
```

```
fitruncobs<-runcobsno(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.5)
#mse
sum((y-fitruncobs)^2)/102

fitrunlow<-qhdsmno(x,y,LP=TRUE,pyhat = TRUE,fr=0.2,qval = 0.5)
#mse
sum((y-fitrunlow)^2)/102

plot(x,y,ylim=c(10,120),xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

lines(x[orderx],fitcobslow,lwd=2)
lines(x[orderx],fitcobsrun,col=2,lwd=2)
lines(x[orderx],fitruncobs,col=3,lwd=2)
lines(x[orderx],fitrunlow,col=4,lwd=2)
legend("bottomright",legend=c("COBSLOW","COBSRUN","RUNCOBS","RUNLO
W"),col=1:4,lty=1,cex=0.8,y.intersp=0.6,lwd=2,text.font = 2)
```

#R codes for figure 3.14 and table 3.23

```
x<-Prestige$income
y<-Prestige$prestige

orderx<-order(x)

fitcobslow<-cobslow(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.1)
#mse
sum((y-fitcobslow)^2)/102

fitcobsrun<-cobsrunno(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.1)
#mse
```

```
sum((y-fitcobsrun)^2)/102

fitruncobs<-runcobsno(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.1)
#mse
sum((y-fitruncobs)^2)/102

fitrunlow<-qhdsmno(x,y,LP=TRUE,pyhat = TRUE,fr=0.2,qval = 0.1)
#mse
sum((y-fitrunlow)^2)/102

plot(x,y,ylim=c(10,120),xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

lines(x[orderx],fitcobslow,lwd=2)
lines(x[orderx],fitcobsrun,col=2,lwd=2)
lines(x[orderx],fitruncobs,col=3,lwd=2)
lines(x[orderx],fitrunlow,col=4,lwd=2)
legend("bottomright",legend=c("COBSLOW","COBSRUN","RUNCOBS","RUNLO
W"),col=1:4,lty=1,cex=0.8,y.intersp=0.6,lwd=2,text.font = 2)

#R codes for figure 3.15 and table 3.24
x<-Prestige$income
y<-Prestige$prestige

orderx<-order(x)

fitcobslow<-cobslow(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.9)
#mse
sum((y-fitcobslow)^2)/102

fitcobsrun<-cobsrunno(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.9)
```

```
#mse
sum((y-fitcobsrun)^2)/102

fitruncobs<-runcobsno(x,y,LP=TRUE,pyhat=TRUE,fr=0.2,qval = 0.9)
#mse
sum((y-fitruncobs)^2)/102

fitrunlow<-qhdsmno(x,y,LP=TRUE,pyhat = TRUE,fr=0.2,qval = 0.9)
#mse
sum((y-fitrunlow)^2)/102

plot(x,y,ylim=c(10,120),xlab="INCOME",ylab="PRESTIGE",font.lab=2)
axis(1, font=2)
axis(2, font=2)

lines(x[orderx],fitcobslow,lwd=2)
lines(x[orderx],fitcobsrun,col=2,lwd=2)
lines(x[orderx],fitruncobs,col=3,lwd=2)
lines(x[orderx],fitrunlow,col=4,lwd=2)
legend("bottomright",legend=c("COBSLOW","COBSRUN","RUNCOBS","RUNLO
W"),col=1:4,lty=1,cex=0.8,y.intersp=0.6,lwd=2,text.font = 2)
```

#R codes for predict

```
x<-Prestige$income
y<-Prestige$prestige
xg<-
(xx=c(100,400,611,700,918,1656,2400,3500,4000,5000,6600,7405,8500,9271,10000
,11377,12351,13000,13500,14558,15000,16500,18000,20000,23000,25000,25879,26
000,27000,28000))
```

```
##new observation (RIS)

r1<-rungen1(x,y,pyhat = T,plotit = T,LP=F,est=hd,q=0.5)
r2<-rungen1(x,y,pyhat = T,plotit = T,LP=F,est=no,q=0.5)

predict(r1,xg,est=hd,q=0.5)
predict(r1,xg,est=no,q=0.5)

#new observation (COBS)

sbs<-cobs(x,y,lambda = 0)
predict(sbs,xg)

#new observation (LOWESS)

lwmethod<-
loess(y~x,degree=1,family="symmetric",surface="direct",model=TRUE)
predict(lwmethod,xg)

#new observation (Kernel Smoothing)

ss<-sreg(x,y,method="GCV")
predict(ss,xg)
```