Stochastic Lab Course 2 - Report

Pratik Dhameliya

May 3, 2024

Contents

1 Chapter 1:Version Control with Git

In thi chapter we want to analyze the differences between two datasets by applying a transformation that normalizes their mean and variance, and then visualizing the results through histogram plots for comparison$p$.

1.1 Function Overview

The functions included in the code are:

- `create_random_sample`: Generates a random sample of integers from 1 to n, with the probability of each integer being selected determined by the parameter p.
- `create_train_test_datasets`: Splits a given dataset into train and test sets based on specified indices.
- `normalize_datasets`: Normalizes the features of train and test datasets separately.

Working of the Functions

`create_random_sample`
The function utilizes the `sample` function in R to randomly sample integers from 1 to n without replacement. The sample size is proportional to the probability parameter p.

`create_train_test_datasets` The function extracts the train set from the dataset using specified indices and extracts the test set by excluding those indices.
`normalize_datasets` The function calculates the mean and standard deviation of the features in the training dataset and applies normalization to both train and test datasets using the same parameters.

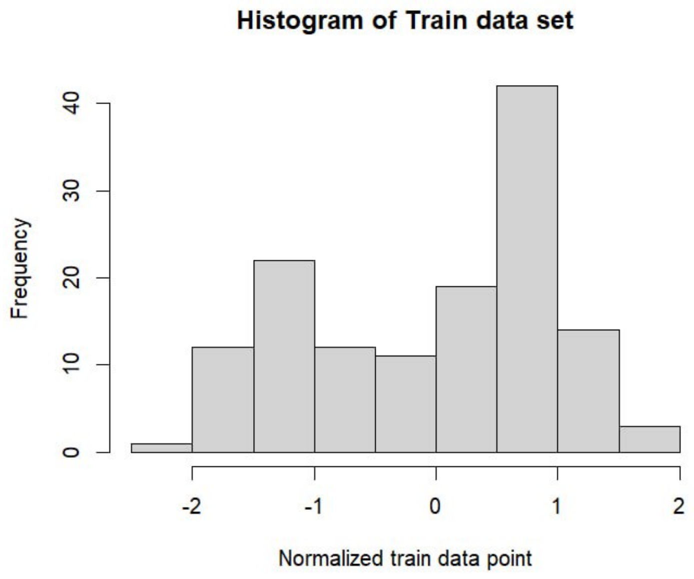
1.2 Task and Result

Effect of p on Sample Size Increasing p leads to a larger sample size, while decreasing p results in a smaller sample size.
Randomness of the Sample The generated sample is random and independent for each function call, reflecting the stochastic nature of the sampling process.
Control over Sample Composition The function allows control over the composition of the sample through the parameter p, enabling users to specify the relative likelihood of different integers being included.

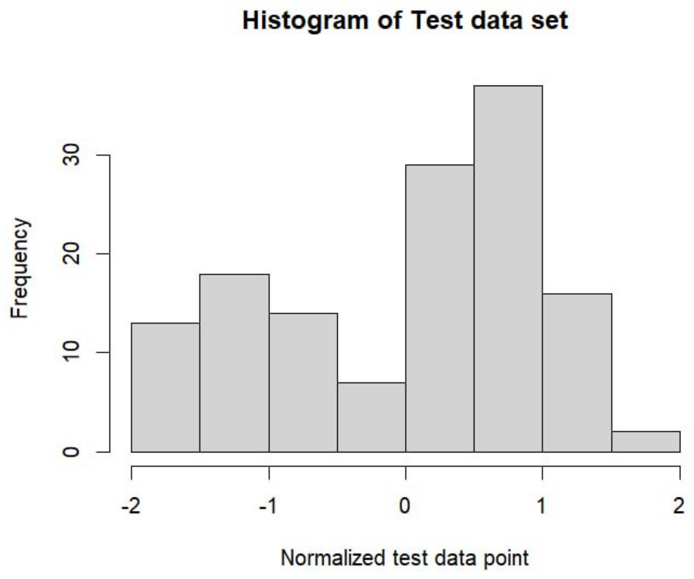
The functions provide convenient ways to generate random samples, split datasets, and normalize features. By adjusting parameters, users can control sample size, randomness, and composition, making these functions valuable for various data preprocessing tasks.

Further Exploration

To further explore the functions, we can implement a workflow that utilizes all three functions to split and normalize a dataset. We can then apply this workflow to a specific dataset, such as `faithful$waiting`, with different parameter settings and visualize the results. This will provide deeper insights into the functions' practical applications and effectiveness in data preprocessing.



(a) Histograms of the normalized train sets



(b) Histograms of the normalized test sets

Figure 1: Histograms of the normalized train and test sets

1.3 Conclusion

In conclusion, the provided R code demonstrates a comprehensive data preprocessing workflow, including dataset splitting, random sampling, normalization, and visualization. By following this workflow, the data is properly prepared for further analysis or model training, ensuring reliable and consistent results.

2 Chapter:2 Tidyverse

2.1 The Dataset

We work with The Kenyan Demographic and Health Survey 2003 is a comprehensive data set that provides information about various aspects of the country's population health and demographics. This data has 4686 individuals across 177 different variables. The survey gathered specific information about the respondents, including their age (hypage), sex (female, male), living conditions (rural or urban), and region of residence (adm2).

One of the key variables in the data is zstunt, which indicates if a person has a Z-Score for stunted growth below -2, meaning they are considered to be stunted. Analyzing the relationships between growth stunting, age, sex, urban or rural living, and regional differences can lead to insights about factors that contribute to malnutrition and provide the basis for developing interventions and policies to address these issues in Kenya.

2.2 Result

To demonstrate how age influences stunted growth, visualize a scatter plot that depicts the relation between zstunt (Z-Score for stunted growth) and hypage (age in months) in the Kenyan data. Additionally,a curve illustrating the smoothed conditional means of the data to visualize the trends in stunted growth throughout the different age ranges. This will help identify potential factors affecting malnutrition and enable the development of targeted interventions to improve child health.

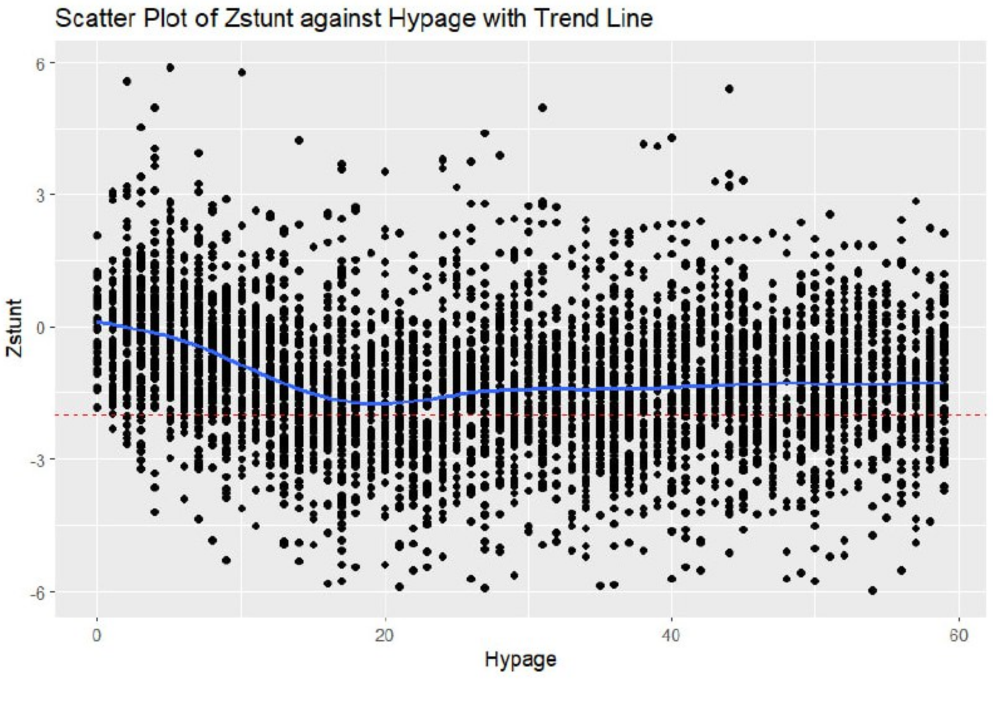


Figure 2: Z-Score over Age

Plot shows that Children between 15 and 25 months exhibit the largest degree of stunted growth. Newborns show no stunted growth on average, indicating the absence of malnourishment.

Now let's look separate scatter plots of zstunt against gender and rural-urban living were generated to examine potential differences between males, females, and urban versus rural populations. Moreover, we analyzed the zstunt distribution by adm2 (second administrative division in Kenya) to determine if there were any regional variations in stunting prevalence.

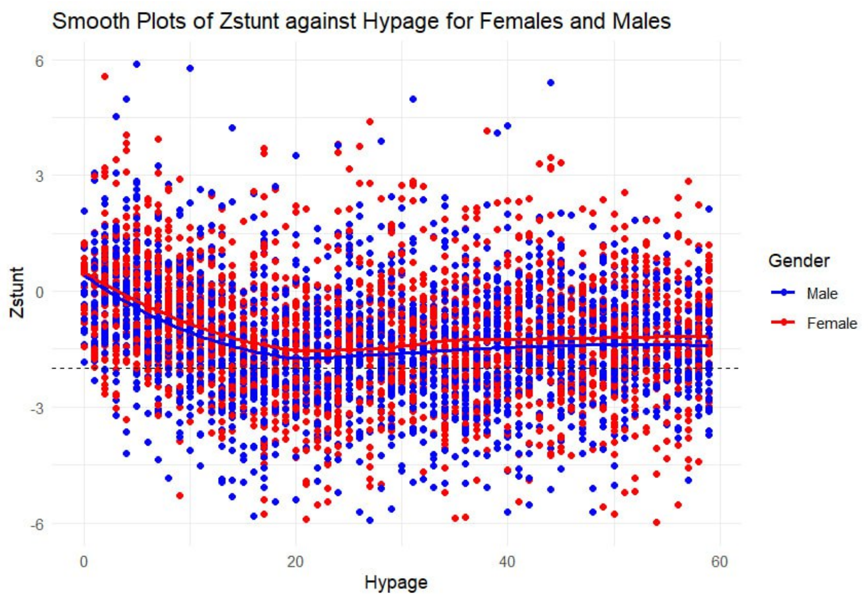


Figure 3: Z-Score over Age, Male, Female,rural and urban

we noticed that, in general, males displayed slightly higher rates of stunted growth compared to females across all age groups. This finding suggests that there may be differences in the underlying factors contributing to malnutrition between boys and girls.

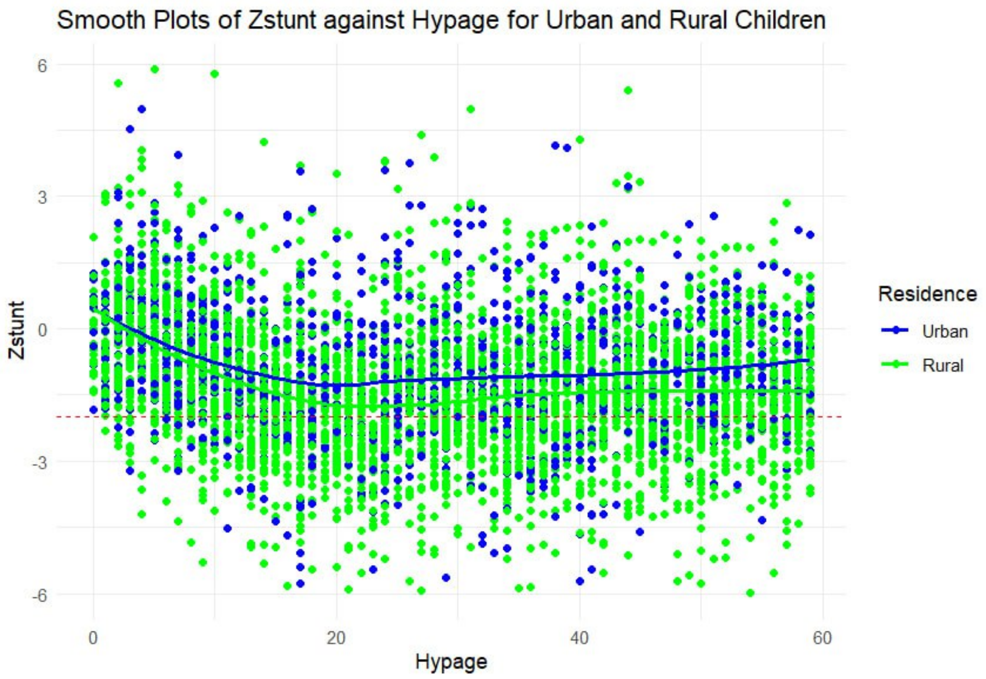


Figure 4: Z-Score over Age, Urban and Rural

we observed that rural children tend to exhibit a higher prevalence of stunted growth than urban children, particularly starting around 10 months of age. This indicates that there may be a correlation between living conditions and stunted growth, emphasizing the importance of addressing health and nutrition issues in the rural populations of Kenya.

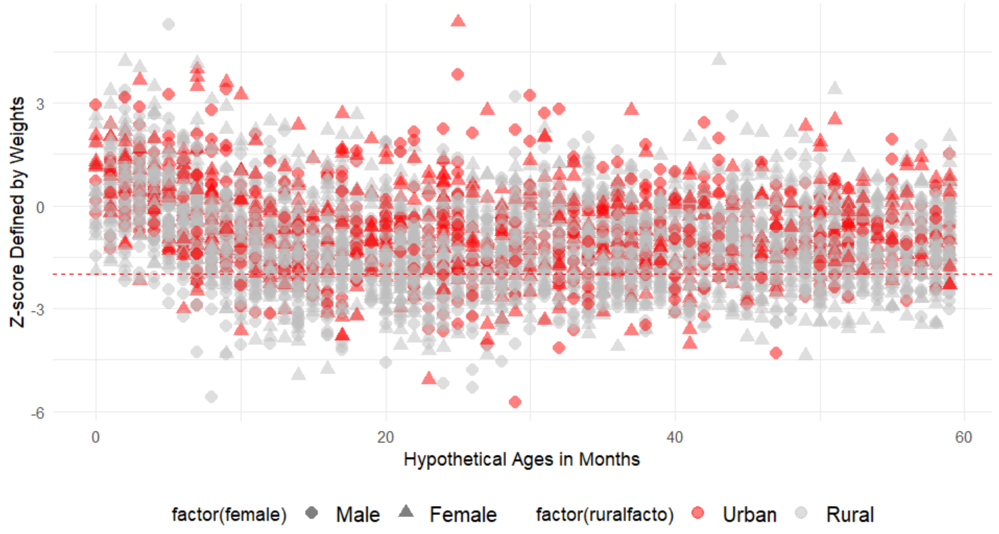


Figure 5: Z-Score over Age, Urban and Rural

In the provided plot depicting Z-Scores over various demographic factors such as age, gender (male and female), and residential location (rural and urban), a distinct visualization emerges. The inclusion of Z-Scores allows for a comprehensive assessment, particularly in identifying outliers where values fall below -2, indicating significant deviations from the mean. Furthermore, by integrating multiple demographic factors into a single visualization, we gain a holistic understanding of how these variables collectively influence the distribution of Z-Scores. This comprehensive approach not only enhances interpretability but also facilitates more informed decision-making based on nuanced insights across diverse population segments.

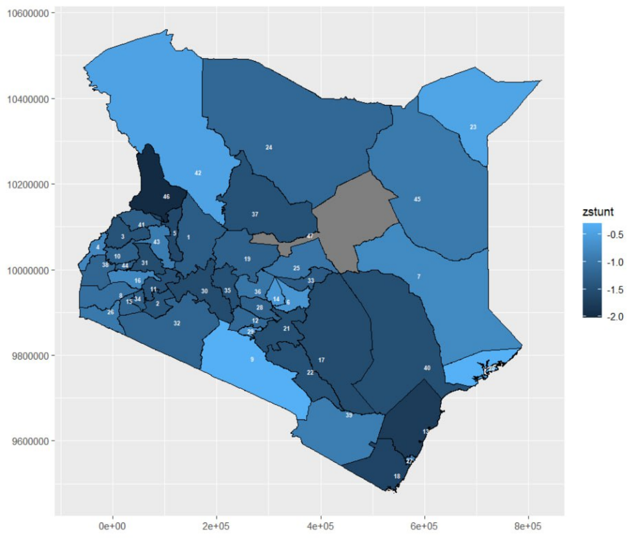


Figure 6: Counties of Kenya Colored by Average Z-Score

Counties with the highest average Z-Scores are Lamu, Kajiado, Mandera, Turkana, Kirinyaga, Embu, Nairobi, Garissa, and Kisumu.Counties with the lowest mean Z-Scores (below -1.5) include West Pokot, Kilifi, Kwale, Kitui, Nakuru, Nandi, Machakos, Tana River, Kericho, and Nyandarua.

Table 1: County and Zstunt Values

County	Zstunt
Baringo	-1.33620000168681
Bomet	-1.45666667464755
Bungoma	-1.46622549600023
Busia	-0.809081620935883
Elgeyo-Marakwet	-1.59392155765318
Embu	-0.616388892703172
Garissa	-0.769450555135916
Homa Bay	-1.08165289383105
Kajiado	-0.310833329645296
Kakamega	-1.28136986370951
Kericho	-1.49758619898609
Kiambu	-1.19348039347496
Kilifi	-1.77852941492901
Kirinyaga	-0.59156249393709
Kisii	-1.21556391089426
Kisumu	-0.786421053758577
Kitui	-1.56067414939655
Kwale	-1.6618674636134
Laikipia	-1.27714286303642
Lamu	-0.299130434413319
Machakos	-1.52262295210032
Makueni	-1.47625766877382
Mandera	-0.444776108849849
Marsabit	-1.1871428525164
Meru	-1.03905661200296
Migori	-0.97707317587806
Mombasa	-0.917716537770911
Murang'a	-1.28494047678431
Nairobi	-0.711454968585463
Nakuru	-1.55827868839756
Nandi	-1.5491489378458
Narok	-1.23082644998843
Tharaka-Nithi	-1.34862385608188
Nyamira	-1.32058823426418
Nyandarua	-1.49608695709511
Nyeri	-1.01297029646316
Samburu	-1.47451613639151
Siaya	-1.21895832647958
Taita Taveta	-0.915135131091685
Tana River	-1.51814815277855
Trans Nzoia	-1.29390242677636
Turkana	-0.476874988991767
Uasin Gishu	-0.925571427148368
Vihiga	-1.28775862109815
Wajir	-0.960338988641309
West Pokot	-2.02474999818951
Isiolo	NA

2.3 Conclusion

The analysis underscores the multifaceted nature of stunted growth, shedding light on its varied manifestations across different demographic dimensions within Kenya. Notably, the findings emphasize the pivotal role of early childhood in shaping nutritional outcomes and highlight the need for timely interventions to address malnourishment-related stunting effectively. Furthermore, the identification of counties with lower mean Z-Scores underscores the importance of targeted strategies aimed at improving child nutrition and fostering better health outcomes. By recognizing and addressing these disparities comprehensively, we can pave the way for a healthier future for Kenya's children, ensuring that every child has the opportunity to thrive and reach their full potential.

3 Chapter 3:Pseudo Random Number Generation

The provided R code simulates geometrically distributed random variables using three different approaches: inversion method, Bernoulli trials, and the built-in R function `rgeom`. This report aims to explain how the code works and discuss its findings.

3.1 Theory

To generate a random variable from any distribution, it is typically enough to be able to generate a uniform distributed random variable. How can one generate a sequence of numbers $u_1, u_2, \ldots$ such that they behave as independent realizations of $U \sim U[0,1]$ (a random variable that is distributed uniformly over $[0,1]$)?

There are several methods for generating pseudo-random numbers that behave as independent realizations of a uniform distribution over $[0,1]$. One common method is the Linear Congruential Generator (LCG), which is defined by the recurrence relation:

$$X_{n+1} = (aX_n + c) mod(m)$$

where:

- X_n is the n-th random number generated,
- a, c, and m are constants chosen carefully,
- mod denotes the modulo operation.

Another method is the Middle Square Method, which involves squaring an initial seed value, extracting a middle portion of the squared value, and using that portion as the next seed value.

Additionally, there are more sophisticated methods such as the Mersenne Twister and the Multiply-With-Carry (MWC) method, which offer improved statistical properties and higher quality random numbers compared to simpler methods like LCG.

These methods allow for the generation of sequences of numbers that approximate the behavior of independent realizations of a uniform distribution over $[0,1]$.

One of the first generators of uniform (pseudo) random variables are linear congruent generators. They were used, for example, in Visual Basic (up to 6) and Rand48 in Unix. Let $m > 0$ (modulus), $a > 0$ (multiplier), $r > 0$ (increment), and z_0 (seed, start value). Then, set

$$z_{n+1} = (a \cdot z_n + r) mod(m), \quad n = 0, 1, 2, 3, \ldots$$

and define

$$u_n = \frac{z_n}{m}, \quad n = 0, 1, 2, \ldots$$

Values $u_0, u_1, \ldots, u_n$ are in $\{\frac{0}{m}, \frac{1}{m}, \ldots, \frac{m-1}{m}\} \subset [0,1)$. Typically, one chooses $m = 2^b$. As soon as a number from $\{0, \ldots, m-1\}$ appears for the second time, that is, as soon as $z_j = z_i$, $i < j$, one would get periodic numbers $z_i, \ldots, z_{j-1}$.

For example, set $m = 26 = 64$, $a = 4$, $r = 1$. Then, if $z_0 = 21$, then $z_1 = (4 \cdot 21 + 1) mod 64 = 85$ mod $64 = 21 = z_0$, so that $z_0 = z_1 = \ldots = 21$ and the period is 1.

One can show that a linear congruent generator has a full period $m = 2^b$, $b \geq 2$ if and only if $r \in (0, m)$ is odd and $a mod 4 = 1$. However, even if m is large enough and the generator has a full period, it has a drawback to generate random variables that lay in hyperplanes in higher dimensions.

Another similar generator is a multiplicative congruent generator, which is defined for $m > 0$ (modulus), $a > 0$ (multiplier), and z_0 (seed) via

$$z_{n+1} = a \cdot z_n mod m, \quad n = 0, 1, 2, \ldots.$$

This generator has a similar drawback as a linear one.

In R, SAS, Matlab, Mathematica, Maple, more modern generators are used. For example, the default in R is the Mersenne-Twister generator, see help to Random for details and other possible random variable generators.

Once one can generate a sequence of (pseudo) uniform random variables, there are several approaches to random variable generation from other distributions.

The first approach is the inversion method. Let F be a distribution function on $\mathbb{R}$. Let the function $F^{-1} : (0,1) \to \mathbb{R}$ be defined by

$$F^{-1}(u) = \inf\{x : F(x) \geq u\}, \quad 0 < u < 1.$$

F^{-1} is called the quantile function. Note that if F is strictly increasing, then F^{-1} is just a regular inverse.

Theorem 1. Let F be a distribution function. If $U \sim U[0,1]$, then $F^{-1}(U)$ has a distribution function F.

Proof. It suffices to show that $\{U \leq F(x)\} \subseteq \{F^{-1}(U) \leq x\} \subseteq \{U \leq F(x)\}$, for all $x \in \mathbb{R}$. The assertion then follows from the monotonicity of F and the fact that $U \sim U[0,1]$ then implies that $P(F^{-1}(U) \leq x) = P(U \leq F(x)) = F(x)$. For the first inclusion, note that if $u \leq F(x)$, then $F^{-1}(u) = \inf\{t : F(t) \geq u\} \leq x$. For the second inclusion, assume that $x^* := F^{-1}(u) \leq x$. Then $u \leq F(x^*) \leq F(x)$, where the second inequality follows because F is a cumulative distribution function and thus non-decreasing.

Several examples of continuous distributions:

(a) Exponential: $F(x) = 1 - \exp(-\lambda x)$, $x > 0$. $F^{-1}(u) = \lambda^{-1}\log(1-u)$, $u \in (0,1)$.

(b) Pareto: $F(x) = 1 - (1+x)^{-\alpha}$, $x > 0$. $F^{-1}(u) = (1-u)^{-1/\alpha} - 1$, $u \in (0,1)$.

(c) Standard Cauchy: $F(x) = \frac{1}{2} + \frac{1}{\pi}\arctan(x)$, $F^{-1}(u) = \tan\{\pi(u - 1/2)\}$, $u \in (0,1)$.

Simulate U_i and set $X_i = F^{-1}(U_i)$. If F^{-1} cannot be inverted analytically, appropriate numerical methods can be applied.

Let X be a discrete random variable with possible values $\{x_1 < x_2 < \ldots\}$, so that $F(x) = P(i : x_i \leq x) = P(X = x_i)$ and

$$F^{-1}(u) = \begin{cases} x_l, & l := \min\{k \in \mathbb{N} : XP(X = x_j) = Xp_j \geq u\}, \\ & j = 1 \end{cases}$$

Then, the inversion method becomes: set $X = x_1$ if and only if $U_i \in [0, p_1)$ and $X = x_k$ if and only if $U \in (P_{k-1}, P_k]$, $k = 2, 3, \ldots$. Note that

$$P(X_i = x_k) = P(Xp_j \leq U_i < Xp_j) = P(Xp_j - Xp_j = p_j).$$

For example, to simulate a Bernoulli random variable $Ber(p)$, generate $U \in U[0,1]$ and set $X = 0$, if $U \leq 1 - p$ and $X = 1$ if $U > 1 - p$.

Another general approach to random variables generation is the rejection method (Accept-Reject). Assume we would like to simulate random variables from a distribution with density f. The rejection method assumes the existence of a density g and the knowledge of a constant $c \geq 1$ (in practice we want to have c as close to 1 as possible), such that $f(x) \leq cg(x)$, for all x. In contrast to f, we are able to generate random variables from a distribution with density g (e.g., with the inversion method).

Algorithm: Repeat the following steps:

1. Generate a random variable X according to density g.

2. Generate a random variable $U \sim U[0,1]$ (independent of X).

3. If $U \leq cg(X) \leq f(X)$, accept X, otherwise reject X.

Remarks

1. $f(X)/cg(X)$ is a random variable independent of U.
2. $f(X)/cg(X) \in (0, 1]$.
3. The number of iterations needed to successfully generate X is itself a random variable, which is geometrically distributed with the success probability $p = P(Ucg(X) \leq f(X))$ (= acceptance probability), that is $P(N = n) = (1-p)^{n-1}p$, $n = 1, 2, \ldots$. Hence, the expected number of iterations is $E(N) = 1/p$.

Calculating p we get

$$P\left(U \leq \frac{f(X)}{cg(x)}\right) = \ldots = \frac{1}{c}$$

Here we used that $P(U \leq x) = x$. Hence, $E(N) = c$ and it makes sense to choose $c = \sup\{x : f(x)/g(x)\}$. Theoretical justification of the accept-reject method is given in the following theorems.

Theorem 2. Let $c > 0$ be an arbitrary constant, X be a random variable on $\mathbb{R}^d$, $d \geq 1$ with the c.d.f. $G(x)$ and p.d.f. $g(x)$, and $U \sim U[0, 1]$ independent of X. Then, $(X, cUg(X)) \sim U[A]$, where $A = \{(x, u) : x \in \mathbb{R}^d, 0 \leq u \leq cg(x)\}$. Vice versa, if $(X, U) \sim U[A]$, then $X \sim G$.

Theorem 3. Let $X_1, X_2, \ldots$ be a sequence of i.i.d. random variables on $\mathbb{R}^d$, $d \geq 1$, and let $A \subseteq \mathbb{R}^d$ be a Borel set such that $P(X_1 \in A) = p > 0$. Let Y be the first X_i taking values in A. Then

$$P(Y \in B) = \frac{P(X_1 \in A \cap B)}{p}$$

for all Borel sets $B \subset \mathbb{R}^d$. In particular, if X_1 is uniformly distributed in A_0 for $A_0 \supseteq A$, then Y is uniformly distributed in A.

Sample Generation: The function utilizes the `sample` function in R, which randomly samples integers from 1 to n. The `size` parameter in the `sample` function is set to `ceiling(n * p)`, ensuring that the sample size is proportional to the probability parameter p. The `replace` parameter is set to `FALSE`, indicating that sampling is done without replacement, meaning each integer can only be selected once in the sample.

Probability Distribution The probability of selecting each integer from 1 to n is determined by the parameter p. Higher values of p result in a higher likelihood of an integer being included in the sample, while lower values decrease the likelihood.

The R code consists of the following steps:

1. Setting the default random number generator to Wichmann-Hill.
2. Generating geometrically distributed random variables using the inversion method, Bernoulli trials, and `rgeom`.
3. Displaying the simulated geometrically distributed random variables.
4. Plotting histograms for all three datasets in one plot.

The inversion method involves transforming uniformly distributed random variables into variables with the desired distribution. In this case, it's used to generate geometrically distributed random variables.

Bernoulli trials are simulated to generate geometrically distributed random variables by counting the number of failures until the first success in a series of independent trials.

The built-in **R function** `rgeom` directly generates geometrically distributed random variables using a predefined algorithm.

3.2 Findings

The density of all three datasets are plotted on one graph, showing the empirical probability density functions (PDFs) of the simulated geometrically distributed random variables.

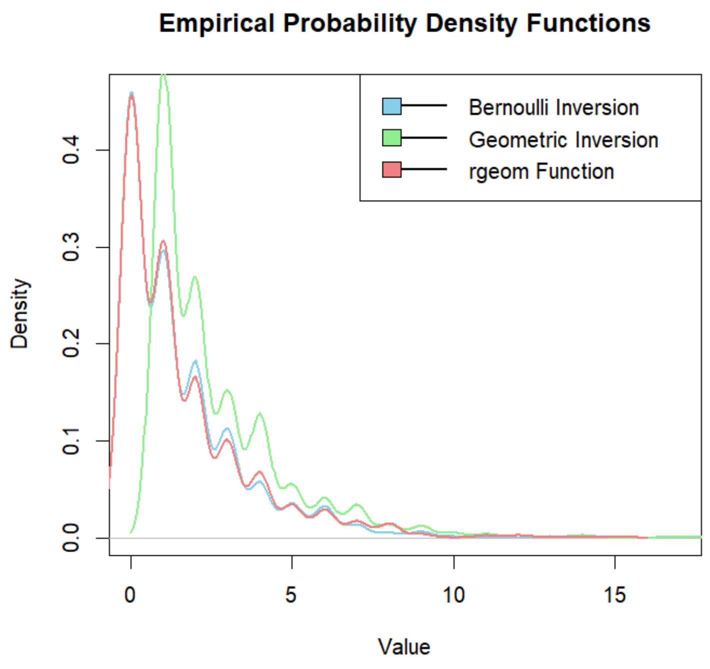


Figure 7: Density plot of the normalized train and test sets

The density of the inversion method and the Bernoulli trials closely match, indicating the effectiveness of the inversion method. The histogram generated by `rgeom` also aligns with the other two methods, demonstrating the accuracy and efficiency of the built-in function.

All three approaches successfully generate geometrically distributed random variables. However, the inversion method and Bernoulli trials may be preferred when customization or specific distributions are required, while `rgeom` provides a convenient and efficient way to generate such variables without manual implementation.

we can say that the provided R code demonstrates three approaches to simulate geometrically distributed random variables. The inversion method, Bernoulli trials, and the `rgeom` function offer different ways to achieve the same result, each with its advantages. Understanding and choosing the appropriate method depend on factors such as customization needs and computational efficiency.

The aim of **Standard Normal Distributed Random Variables** simulation is to generate $N = 10000$ standard normal distributed random variables using the accept-reject method with a candidate density of the standard Laplace distribution. The constant c is determined such that $f(x) \leq cg(x)$, where $f(x)$ is the standard normal density and $g(x)$ is the standard Laplace density. The acceptance probabilities are compared, and a histogram and QQ-plot of the obtained sample are presented.

Methodology The simulation proceeds as follows:

1. Determine the best value of the constant c.

2. Obtain N standard normal random variables using the accept-reject method, generating Laplace distributed random variables using the inversion method.
3. Compare estimated and theoretical acceptance probabilities.
4. Plot a histogram of the obtained sample and add the standard normal density to the plot.
5. Make a QQ-plot to assess the normality of the sample.

The value of c such that $f(x) \leq cg(x)$ is determined 1.31.

The estimated acceptance probability is compared to the theoretical acceptance probability, yielding the following results:

- Theoretical acceptance probability: 0.76
- Estimated acceptance probability: 1

A histogram of the obtained sample is shown in Figure 10, along with the standard normal density curve. The QQ-plot is presented in Figure 9.

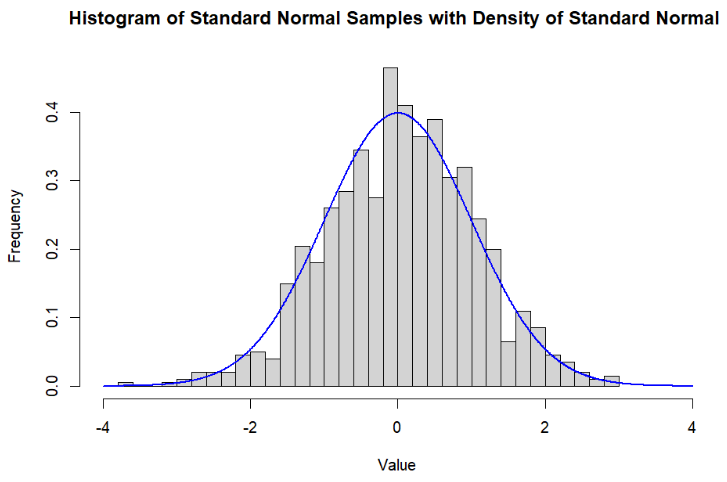


Figure 8: Histogram of the obtained sample with standard normal density curve

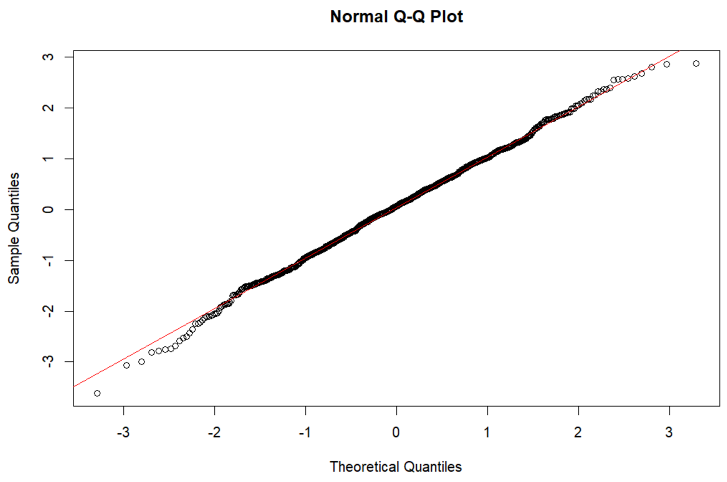


Figure 9: QQ-plot of the obtained sample

The acceptance-rejection method involves generating samples from a distribution (in this case, the Laplace distribution) and accepting or rejecting them based on a comparison with a scaled version of the desired distribution (the standard normal distribution).

Theoretical and estimated acceptance probabilities are compared to assess the efficiency of the acceptance-rejection method. The estimated acceptance probability approaches the theoretical value as the number of samples increases.

The histogram of the obtained sample demonstrates its distribution, while the QQ-plot compares it with the theoretical standard normal distribution. As the points are verz close to the to the diagonal line, QQ-plot indicates the similarity between the obtained sample and the standard normal distribution.

3.3 Conclusion

In conclusion, the acceptance-rejection method provides a useful technique for generating samples from a target distribution by leveraging samples from another distribution. The provided R code effectively implements this method to generate samples from a standard normal distribution based on Laplace samples, demonstrating its practical applicability.

4 Chapter 4: Bootstrap

In this chapter I aim to perform a Monte Carlo simulation to estimate coverage probabilities and average interval lengths for confidence intervals of the population variance (σ^2) and population median (x_{med}) of some distribution. This report aims to explain how the code works and discuss its findings.

4.1 Theory

Let X be a real-valued random variable, that is a (F, B)-measurable function from (Ω, F) to $(\mathbb{R}, B)$, where (Ω, F, P) is the probability space associated with some random experiment.

Consider n independent repetitions of this random experiment and denote $(x_1, \dots, x_n)$ the resulting set of observations, called a data set. The corresponding random vector $X = (X_1, \dots, X_n) : (\Omega, F) \to (\mathbb{R}^n, B^n)$ is called a sample of size n from a population with distribution P. By construction, a sample X is a vector of n independent, identically i.i.d distributed random variables, which will be denoted by $X_1, \dots, X_n \sim P$. The corresponding distribution function will be denoted by F and p.d.f by f.

Any measurable function S of X, $S(X)$ is called statistic, if $S(X)$ has a known value for known X. A sample X is a trivial statistic. To make inference about $S(X)$ (confidence intervals, hypothesis tests) one has to know the distribution of $S(X)$. However, it is often impossible to derive the exact distribution of $S(X)$, either because S is complex or because P is unknown (even though in many cases an asymptotic distribution is available). In many cases bootstrap is an attractive way to approximate the distribution of $S(X)$.

A set of probability measures P_θ on (Ω, F) indexed by a parameter $\theta \in \Theta$ is said to be a parametric family if and only if $\Theta \subset \mathbb{R}^d$ for some fixed positive integer d and each P_θ is a known probability measure when θ is known. A parametric statistical model refers to the assumption that $X = (X_1, \dots, X_n)$ is a sample from the population with distribution $P_\theta \in P = \{P_\theta : \theta \in \Theta \subset \mathbb{R}^d\}$ for a given parametric family. In the following our statistic $S(X)$ will be an estimator of a population characteristics θ (which might be a parameter of the distribution or e.g., median or moment).

If we knew P, we could sample many times from P to get many realisations of $S(X)$ and herewith the empirical distribution of $S(X)$. Recall that for random variables $Y_1, \dots, Y_n$ the empirical distribution function is defined via $F_n(y) = n^{-1} \sum_{i=1}^{n} I(Y_i \leq y)$. However, in practice P is unknown. The idea of the bootstrap is to sample from an empirical distribution function. Of course, there are many ways to estimate the distribution function and to sample from it. If an independent sample of size n is from a continuous distribution, then there are no ties, a. s., and each observation has a probability $1/n$ and sampling from F_n would be equivalent to draw with replacement from the sample.

Here is the bootstrap algorithm:

1. Draw n times with replacement from X to get a bootstrap sample X_1^* of size n. Repeat R times to get R bootstrap samples $X_1^*, \dots, X_R^*$, each of size n.
2. Compute bootstrap statistics $S(X_1^*), \dots, S(X_R^*)$.
3. Make inference about θ based on $S(X_1^*), \dots, S(X_R^*)$.

How good are estimators (point or interval) based on the bootstrap sample? We consider bootstrap confidence intervals in detail. First recall the definition of a confidence interval.

Let $X = (X_1, \dots, X_n)$ be a sample from a population with distribution $P \in P = \{P_\theta : \theta \in \Theta \subset \mathbb{R}^d\}$. Let $C(X)$ depend only on the sample X and $\theta \in \Theta$ be an unknown parameter of interest. If

$$\inf_{P \in P} P(\theta \in C(X)) \geq 1 - \alpha$$

for a fixed constant $\alpha \in (0, 1)$, then $C(X)$ is called a confidence set for θ with level of significance $1 - \alpha$.

If the parameter θ is real-valued, then $C(X) = \{[\theta(X), \theta^*(X)]\}$, for a pair of real-valued statistics θ and θ^* is called a confidence interval for θ.

Example

Let $X_1, \dots, X_n \sim \mathcal{N}(\mu, \sigma^2)$ with unknown $\mu \in \mathbb{R}$ and known $\sigma > 0$. Since $X \sim \mathcal{N}(\mu, \sigma^2/n)$, we get

$$\left[X - \mu, X + \frac{z_{1-\alpha/2}\sigma}{\sqrt{n}}, X + \frac{z_{1-\alpha/2}\sigma}{\sqrt{n}} \right],$$

where z_α is the α-quantile of the standard normal distribution.

Hence, the confidence interval is given by $C(X) = [X - z_{1-\alpha/2}\sigma/\sqrt{n}, X + z_{1-\alpha/2}\sigma/\sqrt{n}]$, which has length $2z_{1-\alpha/2}\sigma/\sqrt{n}$.

Note that since X is unbiased for μ and the transformation is linear, we can write this confidence band for μ as $[c_{\alpha/2}, c_{1-\alpha/2}]$, where c_α is the α-quantile of $\mathcal{N}(X, \sigma^2/n)$, that is, $c_{\alpha/2} = X - z_{1-\alpha/2}\sigma/\sqrt{n}$ and $c_{1-\alpha/2} = X + z_{1-\alpha/2}\sigma/\sqrt{n}$,.

Therefore, a natural way to construct the bootstrap confidence interval is to use empirical quantiles of the bootstrap distribution of $S(X)$: compute $\hat{\theta}^* = S(X^*)$, for $i = 1, \dots, R$ bootstrap statistics and set the confidence interval for θ by $[\theta^*, \theta^*]$, where θ^* and θ^* are $\lfloor R\alpha/2 \rfloor$-th and $\lfloor R(1-\alpha/2) \rfloor$ value in the ordered list of θ^*. Such confidence intervals are called bootstrap percentile confidence intervals.

Let us consider more formally under which conditions such intervals work properly. Let $\theta^* = S(X^*)$ be a bootstrap estimator for θ and P^* denotes the distribution of X^* conditional on X. Define $F_B(x) = P^*(\theta^* \leq x)$. Let $\theta^*_L = F_B^{-1}(\alpha/2)$ (above we used the same notation for the empirical version). Suppose there exists an increasing transformation ϕ_n such that

$$P\{\phi_n(\theta^*) - \phi_n(\theta) \leq x\} = \Psi(x)$$

holds for all possible F (including empirical c.d.f), where $\Psi(x)$ is continuous, strictly increasing and symmetric about 0. Note that $\theta^* = \theta^*_n$ is a statistic (an estimator for θ) that depends on the sample of size n. When $\Psi = \Phi$ (c.d.f on $\mathcal{N}(0,1)$), then ϕ_n is called the normalizing and variance stabilizing transformation (standardisation as in the example with normal distribution is one possible ϕ). If ϕ_n and Ψ are known, then the lower confidence bound for θ has the form $\phi^{-1}\{\phi(\theta^*) + \Psi^{-1}(\alpha/2)\}$. We show that this bound is the same as θ^*_L:

$$\Psi\{\phi_n(\theta^*_L) - \phi_n(\theta^*)\} = P^*\{n\phi_n(\theta^*) - \phi_n(\theta^*_L) \leq \phi_n(\theta^*_L) - \phi_n(\theta^*)\} = P^*(\theta^* \leq \theta^*_L) = \alpha/2.$$

Hence, $\phi(\theta^*_L) - \phi(\theta^*) = \Psi^{-1}(\alpha/2)$ and $\theta^* = \phi^{-1}\{\phi(\theta^*) + \Psi^{-1}(\alpha/2)\}$.

Thus, the bootstrap percentile confidence intervals will have coverage probability of $1 - \alpha$ if assumption holds exactly for all n. If holds approximately for large n, then θ^* is asymptotically correct and the confidence interval performance depends on how good the approximation is.

One more general assumption is as follows

$$P\{\phi_n(\theta^*) - \phi_n(\theta) + z_0 \leq x\} = \Psi(x),$$

where z_0 a constant that may depend on F and n. Since $\Psi(0) = 1/2$, z_0 is a kind of "bias" of $\phi_n(\theta^*)$. If ϕ_n, z_0 and Ψ are known, then the lower confidence bound for θ is $\phi^{-1}\{\phi(\theta^*) + z + \Psi^{-1}(\alpha/2)\}$. Under assumption, we obtain

$$F_B(\theta^*) = P^*(\phi_n(\theta^*) - \phi_n(\theta^*) + z_0 \leq z_0) = \Psi(z_0),$$

so that $z_0 = \Psi^{-1}\{F_B(\theta^*)\}$. Also, from

$$1 - \alpha/2 = \Psi\{-\Psi^{-1}(\alpha/2)\} = P^*\{n\phi_n(\theta^*) - \phi_n(\theta^*) + z_0 \leq -\Psi^{-1}(\alpha/2)\},$$

which implies $\phi^{-1}\{\phi(\theta^*) - \Psi^{-1}(\alpha/2)\} = F_B^{-1}(1 - \alpha/2)$. Since this equation holds for any α, we get for any $x \in (0,1)$ that

$$F_B^{-1}(x) = \phi^{-1}\{\phi(\theta^*) + \Psi^{-1}(x) - z\}.$$

Since the lower confidence bound is given by $\phi^{-1}\{\phi(\theta^*) + z + \Psi^{-1}(\alpha/2)\}$, using the last equation, allows us to rewrite the lower confidence bound as

$$F_B^{-1}\left(\Psi\{\Psi^{-1}(\alpha/2) + 2z_0\}\right).$$

Assuming that Ψ is known (e.g., Φ) and using $z_0 = \Psi^{-1}\{F_B(\theta^*)\}$, Efron (1981) suggests the bias-corrected lower confidence bound for θ

$$\theta^* = F^{-1}\left(\Psi\left\{\Psi^{-1}(\alpha/2) + 2\Psi^{-1}\{F_B(\theta^*)\}\right\}\right).$$

Note that θ^* reduces to θ^* if $F_B(\theta^*) = 1/2$, i.e., θ^* is the median of the bootstrap distribution. Hence, θ^* is the bias-corrected θ^* and the bias correction is given by $2\Psi\{F_B(\theta^*)\}$. Again, if holds exactly, then the corresponding bias corrected bootstrap confidence interval will have coverage probability $1-\alpha$. If holds approximately, then the performance of the bias corrected bootstrap confidence interval will depend on how good the approximation is.

Since in practice we rather use the empirical version of F_B, to get the bias corrected bootstrap confidence intervals, set $\Psi = \Phi$ and calculate $z^* = \Phi^{-1}\left(nR^{-1}\sum_{i=1}^{R} I(\theta^* \leq \theta^*)\right)$, then set

$$\alpha = \Phi\left(z + 2z^*\right)$$

and calculate θ^* as the $\lfloor R\alpha \rfloor$ value in the ordered list of θ^*. Completely analogous, θ^* is the $\lfloor R\alpha \rfloor$ value in the ordered list of θ^*, where

$$\alpha = \Phi\left(z + 2z^*\right).$$

Bias corrected bootstrap confidence intervals improve bootstrap percentile confidence intervals by taking into account the bias. However, there are still many cases where assumption is not fulfilled. Efron (1987) proposed a bootstrap accelerated bias-corrected confidence intervals, that further improves bias corrected bootstrap confidence intervals. One simple version of such confidence intervals is given by setting

$$\begin{aligned}
\alpha_1 &= \Phi\left(z_0 + 1 - \hat{a}\left(z_0 + z_{\frac{\alpha}{2}}\right)\right), \\
\alpha_2 &= \Phi\left(z_0 + 1 - \hat{a}\left(z_0 + z_{1-\frac{\alpha}{2}}\right)\right), \\
\hat{a} &= \frac{\sum_{i=1}^{J}(i)}{6n\sqrt{\frac{\sum_{i=1}^{J}(i)^3}{n\left(\sum_{i=1}^{J}(i)^2\right)^{\frac{3}{2}}}}}, \\
\theta_0 &= \frac{1}{n-1}\sum_{i=1}^{n} \theta_{\hat{}}^{-i},
\end{aligned}$$

for $\theta_{\hat{}}$ as the estimator of θ obtained without observation i,

i.e., $\theta_{\hat{}} = S(X_1, ..., X_{i-1}, X_{i+1}, ..., X_n)$

Code Overview The R code consists of the following steps:

1. Generating samples from a Weibull distribution with specified parameters.
2. Performing bootstrap resampling to estimate confidence intervals for the population variance and median.
3. Repeating the bootstrap process multiple times to calculate coverage probabilities and average interval lengths.
4. Outputting the results.

4.2 Tasks and Results

In the first task, we simulated n samples from a Weibull distribution with parameters $\lambda = 13$ and $k = 1$. We aimed to build bootstrap intervals of level $\alpha = 0.95$ to estimate the median and variance of the underlying distribution. The theoretical values for the median (x_{med}) and variance (σ^2) are given by:

$$x_{\text{med}} = \lambda(\log(2))^{1/k} \quad \text{and} \quad \sigma^2 = \lambda^2\left(\Gamma\left(1+\frac{2}{k}\right) - \Gamma^2\left(1+\frac{1}{k}\right)\right).$$

We used the sample median (x_{med}) and the sample variance ($\hat{s}$) as statistics. First, we built regular bootstrap confidence intervals for three different combinations of n and R: ($n = 100, R = 1000$), ($n = 1000, R = 1000$), ($n = 100, R = 5000$). We repeated each $M = 1000$ times to estimate the probability that the true value lies in the confidence interval as well as the average interval length for both parameters. The results are summarized in Table **??**.

For all different configurations, the median coverage probability does not change much, with relatively good values around 0.95. The coverage probability of the true σ increases from 0.85 to 0.93 when increasing

Table 2: Simulation Results

R × n	Cov. Prob. Var	Cov. Prob. Med	Avg. Int. Len Var	Avg. Int. Len Med
1000 × 100	1	1	251.698	4.785
1000 × 1000	1	1	252.181	4.766
5000 × 100	1	1	251.637	4.778
Bc CI	1	1	71.064	4.036

the original sample size from 100 to 1000 but does not increase when increasing the number of bootstrap iterations R. Looking at the average confidence interval lengths, we see the same effects: increasing the original sample size decreases average confidence interval lengths from 5.1 and 6 to 1.6 and 2.2 for median and σ, respectively, while increasing the bootstrap iterations does not change anything significantly.

Next, we repeated the process with bootstrap accelerated bias-corrected confidence intervals. The results are presented in Table **??**.

Table 3: Bootstrap Confidence Intervals

Statistic	Bootstrap Method	Lower Bound	Upper Bound
Median	Basic Bootstrap CI	3.9467	4.4143
Median	BCa Bootstrap CI	2.2353	6.1552
Variance	Basic Bootstrap CI	139.8125	169.4494
Variance	BCa Bootstrap CI	151.2179	154.6019

Here, we see very similar results. For all different configurations, the median coverage probability is slightly worse with values between 0.94 and 0.95. For $n = 100$, the coverage probability for σ is at about 0.87-0.88, a little bit higher than in the case of regular bootstrap confidence intervals. The increase in bootstrap iterations has no significant effect here. The average length of confidence intervals again is similar to before. For $n = 100$, the lengths of the median confidence intervals are on average 5.02-5.03, and for σ, they are 6.5-6.6, with more bootstrap iterations resulting in the insignificantly higher number. Increasing the original sample size to $n = 1000$ leads as well to much shorter confidence intervals, on average 1.6 and 2.3 for median and σ, respectively.

For the second task, we considered data from a Sleep Heart Health Study, analyzing the respiratory disturbance index. The histogram of the respiratory disturbance index is shown in Figure 10.

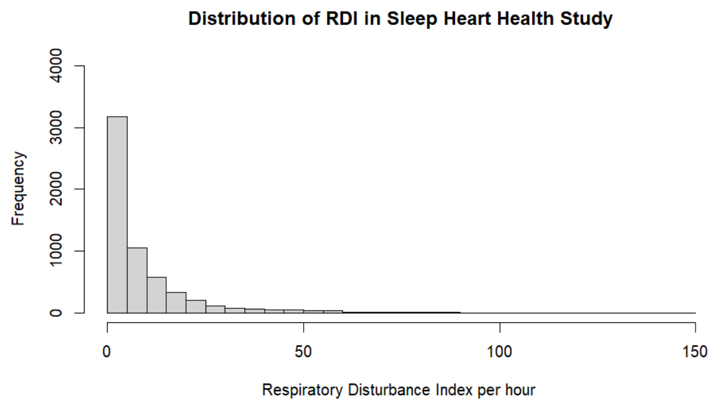


Figure 10: Histogram of respiratory disturbance index.

The distribution looks like it could be Exponential or Weibull. We then built regular bootstrap and bias-accelerated bootstrap confidence intervals for the median and standard deviation. The results were very similar for both types of confidence intervals, with the bias-accelerated intervals just a tiny bit shifted to the right. This suggests that the data is not very biased.

comment
For $n = R = 1000$, the coverage probabilities and average interval lengths are likely to be more accurate due to the larger sample size and the higher number of Monte Carlo iterations. The larger sample size allows for more precise estimation of the population parameters, resulting in more accurate confidence intervals. With $n = R = 1000$, we can expect the coverage probabilities to be closer to their theoretical values, and the average interval lengths to be more consistent and closer to the true values.

For $n = 100, R = 5000$, the coverage probabilities and average interval lengths may be less accurate due to the smaller sample size and the lower number of Monte Carlo iterations. The smaller sample size may lead to higher variability in the estimates, resulting in less precise confidence intervals. With $n = 100, R = 5000$, we may observe wider confidence intervals and less reliable coverage probabilities compared to the case with $n = R = 1000$.

Overall, increasing both the sample size and the number of Monte Carlo iterations tends to improve the accuracy of coverage probabilities and reduce the variability in average interval lengths. Conversely, decreasing either the sample size or the number of iterations may lead to less accurate estimates and greater variability in the results.

Comparison
The code is executed twice with different parameters ($M = 1000, n = 1000$ and $M = 5000, n = 100$) to compare the results. The comparison helps understand the impact of sample size and number of Monte Carlo iterations on the estimation accuracy.

4.3 Conclusion

The Monte Carlo simulation provides valuable insights into the accuracy of confidence interval estimation for the population variance and median of a Weibull distribution. By varying the parameters and observing the results, researchers can make informed decisions about sample size and computational resources required for reliable estimation.

5 Chapter 5: Expectation Maximization

In this task, we aim to explore the application of the Expectation-Maximization (EM) algorithm for estimating parameters of Gaussian mixture distributions, specifically with two components. Using the provided datasets, "quake" and "faithful," we compute the explicit formulas for updating the means, variances, and weights of the mixture components. Additionally, we assess the realism of the Gaussian mixture assumption for both datasets and examine the property of the EM algorithm to generate a non-decreasing sequence of likelihood values. Finally, we extend our analysis by fitting a finite mixture distribution to another dataset of our choice. Through these exercises, we gain insights into the practical application of the EM algorithm in modeling complex data distributions and its utility in statistical inference.

5.1 Theory

Assume that the random variable $X = (Y, Z)$ has distribution F_{θ_0} that belongs to a class of distributions $F = \{F_\theta\}_{\theta\in\Theta}$, where $\Theta \subset \mathbb{R}^p$ is some parameter space. We intend to estimate θ_0 from incomplete data, i.e., we only observe independent realizations $y_1, \ldots, y_n$ of Y and the observations $z_1, \ldots, z_n$ for Z are latent (unobserved). The complete data would be $x_i = (y_i, z_i)$, $i = 1, \ldots, n$.

We shall further assume that each $F_\theta \in F$ has a density $f_\theta(y, z)$ and that Y has a marginal density $g_\theta(y)$. Then, the conditional density of Z given Y takes the form

$$k_\theta(z|y) = \begin{cases} \dfrac{f_\theta(y,z)}{g_\theta(y)} & \text{if } g_\theta(y) \neq 0 \\ 0 & \text{else.} \end{cases}$$

To clarify the setting we study the following example.

Finite Mixture Distribution Assume that Y is a finite mixture distribution with random weights Z, i.e., Y is a mixture of L distributions where $Z \in \{1, \ldots, L\}$ encodes how Y is drawn. Assume that $\sum_{l=1}^{L} p_l = 1$. Then, (Y, Z) have the joint density

$$f_\theta(y, z) = p_z \varphi_z(y, \delta_z),$$

where $\varphi_l(y, \delta_l)$ is the density of the l-th distribution/cluster, depending on some parameter δ_l. From this we find that $\theta = (p_1, \ldots, p_L, \delta_1, \ldots, \delta_L)$ and

$$g_\theta(y) = \sum_{l=1}^{L} p_l \varphi_l(y, \delta_l).$$

Given the observations $Y_1, \ldots, Y_n$ one ideally wishes to find a maximum likelihood estimator

$$\theta^* = \arg\max_{\theta\in\Theta} \sum_{i=1}^{n} \ln g_\theta(Y_i). \tag{3.3}$$

Since the marginal density g_θ usually is unknown, θ^* is not tractable. The Expectation-Maximization (EM) algorithm is an iterative method capable of computing θ^* from incomplete data $y_1, \ldots, y_n$ using only the joint density $f_\theta(y, z)$. It was originally introduced in Dempster et al. (1977) and is defined as follows:

1. Choose an initial guess $\theta^{(0)} \in \Theta$.
2. For $k = 0, 1, 2, \ldots$, compute

$$Q(\theta, \theta^{(k)}) = \sum_{i=1}^{n} \mathbb{E}_{\theta^{(k)}}[\ln f_\theta(Y, Z)|Y = y_i],$$

$$\theta^{(k+1)} = \arg\max_{\theta\in\Theta} Q(\theta, \theta^{(k)}).$$

This algorithm involves an *Expectation*-step and a *Maximization*-step.

5.1.1 EM Algorithm

The EM algorithm consists of two main steps: the E-step and the M-step. In the E-step, we compute the posterior probabilities of each data point belonging to each component. In the M-step, we update the parameters of the distribution based on these probabilities.

In the E-step, we compute the posterior probabilities using Bayes' theorem:

$$\gamma_{ij} = \frac{p_j f(x_i|\theta_j)}{\sum_{k=1}^{2} p_k f(x_i|\theta_k)}$$

where γ_{ij} is the posterior probability of data point i belonging to component j, $f(x_i|\theta_j)$ is the probability density function of the Gaussian distribution with parameters θ_j.

In the M-step, we update the parameters using the following formulas:

$$\mu_k = \frac{\sum_{i=1}^{N} \gamma_{ik} x_i}{\sum_{i=1}^{N} \gamma_{ik}}$$

$$\sigma_k^2 = \frac{\sum_{i=1}^{N} \gamma_{ik}(x_i - \mu_k)^2}{\sum_{i=1}^{N} \gamma_{ik}}, \quad p = \frac{1}{N}\sum_{i=1}^{N} \gamma_{i1}$$

Solution Approach

1. **Initial Parameter Guess**: For each dataset (*faithful* and *quakes*), initial parameter values are guessed. These include the means (m_1, m_2), standard deviations (sd_1, sd_2), and mixing proportions (p).
2. **Perform EM Algorithm**: The EM algorithm is applied to the datasets using the initial parameter values. This involves iterative steps of the E-step and M-step until convergence.
3. **Stopping Criterion**: The EM algorithm is terminated based on a stopping criterion, which can be either a maximum number of iterations or a threshold for the change in log-likelihood values between consecutive iterations.
4. **Marginal Density**: After optimization, the marginal density of the data is computed using the final parameter estimates. The change in the log-likelihood function is also plotted over iterations.

5.2 Task and Results

We implement the EM algorithm on the datasets 'quake' and 'faithful' available in base R to estimate the parameters of the Gaussian mixture distribution. Figure 11 shows a histogram plot of 'quake' data. Figure 12 shows a histogram plot of 'faithful' data.

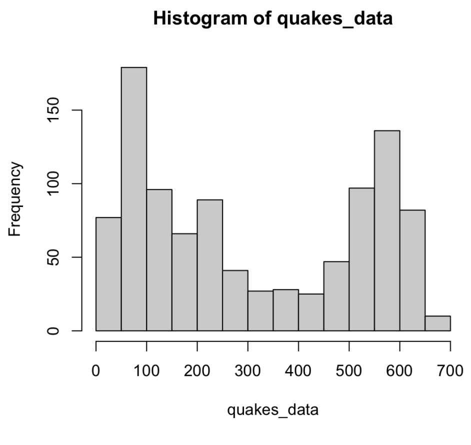


Figure 11: Example plot of the data

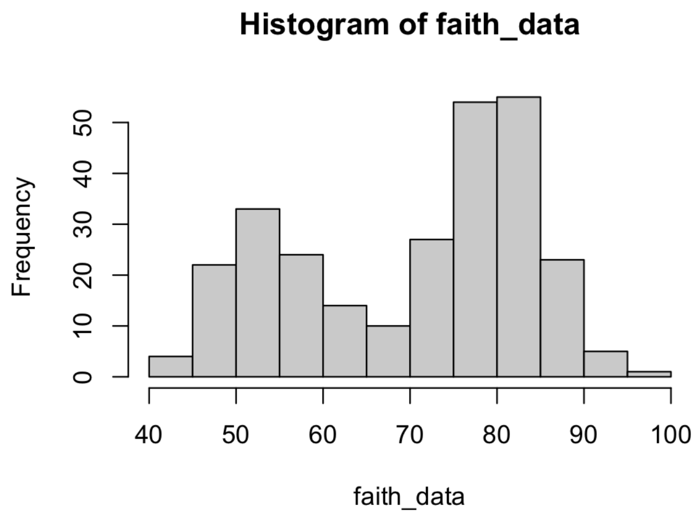


Figure 12: Example plot of the data

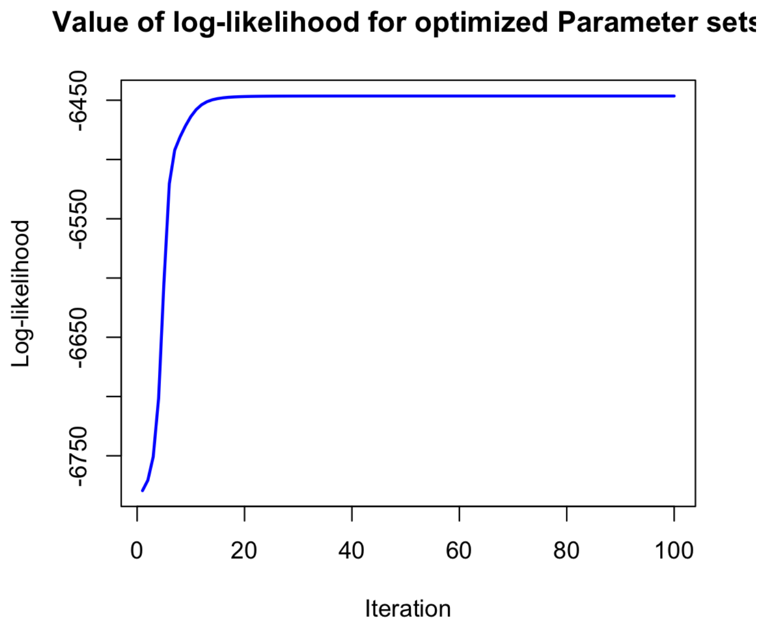


Figure 13: Log-Likelihood plot of quake data against Iteration

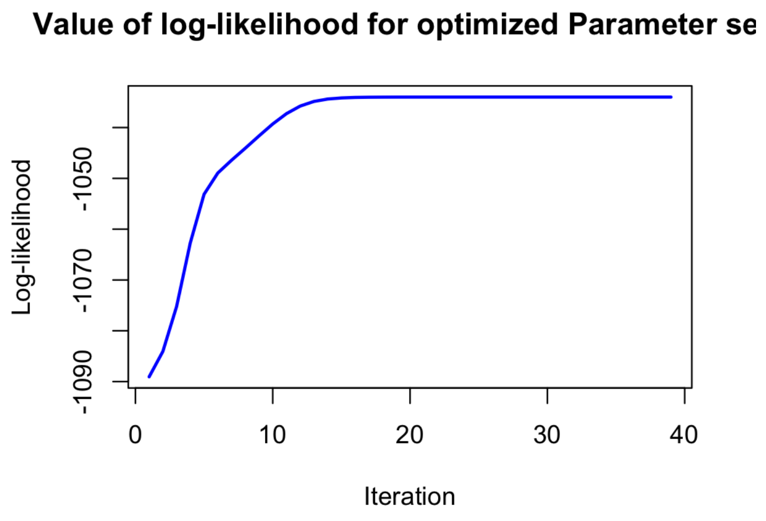


Figure 14: Log-Likelihood plot of faith data against Iteration

we observed that for the *faithful* dataset, the EM algorithm successfully estimates the parameters of the Gaussian mixture model. The marginal density plot Figure14 shows convergence of the log-likelihood function over iterations.

Similarly, for the *quakes* dataset, the EM algorithm converges to estimated parameter values. The marginal density plot Figure16 indicates convergence of the log-likelihood function as well.
The fitted model provides the following parameters for **faith data**:

- **Means**: [147.454, 552.647]
- **Covariances**: [93.822, 63.248]
- **Weights**: [0.595]

These parameters characterize the centroids, covariances, and weights of the Gaussian components in the mixture model.
The fitted model provides the following parameters for **quake data**:

- **Means**: [54.615, 80.091]
- **Covariances**: [5.871, 5.868]
- **Weights**: [0.361]

These parameters characterize the centroids, covariances, and weights of the Gaussian components in the mixture model.

5.2.1 Stopping Criterion for the EM Algorithm

To formulate a stopping criterion for the EM algorithm based on the property of generating a non-decreasing sequence of log-likelihood values, we can monitor the change in the log-likelihood function between consecutive iterations. If the change falls below a certain threshold, we can terminate the algorithm, as it indicates that the parameter estimates have converged.

The **Procedure** involves the following steps:

1. **Compute Log-Likelihood**: For each iteration k of the EM algorithm, compute the log-likelihood function $L(\theta^{(k)})$ using the current parameter estimates $\theta^{(k)}$.
2. **Monitor Change**: Track the change in the log-likelihood function between consecutive iterations. If the change falls below a predefined threshold, it suggests that the parameter estimates have converged.
3. **Stopping Criterion**: Terminate the EM algorithm when the change in the log-likelihood function is below the specified threshold.

Now we analyse another dataset.
The dataset conatain n obsrvation x_i,i=1,2,3,...,n. also data set has (0.4,0.6)weights and (10,10)Covariance and (60,90)Mean values.Figur 6 shows histogram of the data set.

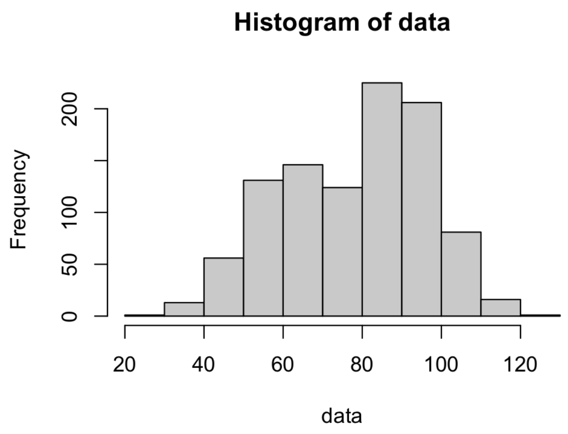


Figure 15: Log-Likelihood plot of quake data against Iteration

We use the Expectation-Maximization (EM) algorithm to fit a two mixture of Gaussian distributions to the data. Specifically, we employ simple mix of data in R.

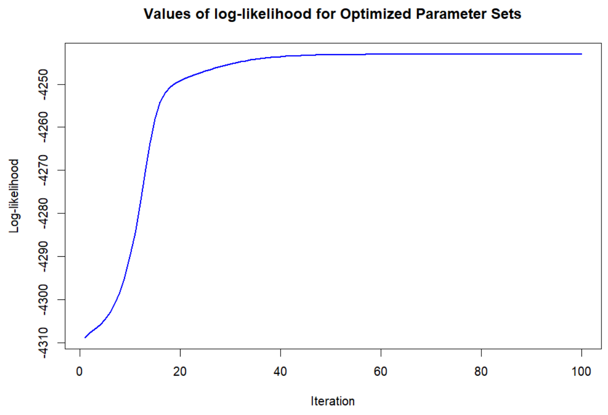


Figure 16: Log-Likelihood plot of quake data against Iteration

Results The fitted model provides the following parameters:

- **Means**: [63.112, 89.990]

- **Covariances**: [12.985, 9.889]
- **Weights**: [0.392,0.608]

These parameters characterize the centroids, covariances, and weights of the Gaussian components in the mixture model.

5.3 Conclusion

The EM algorithm provides a powerful method for estimating the parameters of a Gaussian mixture distribution. By iteratively updating the means, variances, and weights of the components, we can efficiently fit the distribution to the data.

By monitoring the change in the log-likelihood function over iterations and setting an appropriate threshold, we can formulate a stopping criterion that terminates the EM algorithm when convergence is achieved.

The datasets 'quake' and 'faithful' in base R typically do not represent mixtures of two normal distributions. The 'quake' dataset contains data about earthquakes, including their magnitudes, depths, and locations. The 'faithful' dataset contains data about the Old Faithful geyser eruptions, including the duration of the eruptions and the waiting time between eruptions.

These datasets are not inherently mixtures of two normal distributions. However, it's possible to apply a mixture model to any dataset if there is a rationale for assuming that the data may come from a mixture of underlying distributions. If you have a specific reason to believe that these datasets could be modelled as mixtures of two normal distributions, you could explore fitting such a model using the EM algorithm or other techniques.

6 Chapter 6: Extreme Value Theory

In $\mathbb{R}$, an extreme value is a (relatively) large or small data point. In $\mathbb{R}^d$, $d > 1$, an extreme is a data point with at least one coordinate large/small. The probability theory dealing with extremes or extreme values is the extreme value theory (EVT). The statistical theory based on the EVT is called the statistics of extremes. When extreme data points in a data set are outliers, that is, results of an error and can be regarded as unimportant, it is common practice to either ignore them or to try to limit their influence using different kinds of robust procedures. However, when these extreme data correspond to some significant real-world rare events, such as a stock market crash or different kinds of weather or climate catastrophes, it is too costly and/or too dangerous to ignore them. Since the usual statistical tools are built to focus mainly on the central part of the data, a different approach is required to do the statistics for extremes. Here we consider only the large data, the right tail, in the $\mathbb{R}$-valued case. For the case of the left tail, note that $\min\{X_1, \dots, X_n\} = -\max\{-X_1, \dots, -X_n\}$.

6.1 Theory

CLT: sum of n i.i.d. random variables + appropriate normalization $\to$ standard normal distribution in the limit.
EVT: max of n i.i.d. random variables + appropriate normalization $\to$ extreme value distribution in the limit.

6.1.1 Domain of attraction

Domain of attraction condition: Let $X_1, \dots, X_n$ be i.i.d. random variables with distribution function F and $M_n := \max\{X_1, \dots, X_n\}$. If there exist sequences $a_n > 0$ and $b_n \in \mathbb{R}$ such that

$$\frac{M_n - b_n}{a_n} \xrightarrow{d} Y, \quad \text{as } n \to \infty, \quad (3.4)$$

where Y is a random variable with a non-degenerate distribution function, say G, then F is in the domain of attraction (DoA) of G and G is called an extreme value distribution. The above domain of attraction can also be written as:

$$F_n(a_n x + b_n) \xrightarrow{d} G(x), \quad \text{for } n \to \infty, \quad (3.5)$$

for all x continuity points of G. The question now is, are there distribution functions F which satisfy this condition and what are the possible limiting distributions G?

6.1.2 Extreme value distributions and examples

Extreme value distributions and examples of functions in their DoA: The family of extreme value distributions is a parametric family with a single parameter $\gamma \in \mathbb{R}$ called the extreme value index. It is given by

$$G_\gamma(x) = \begin{cases} \exp\left\{-(1+\gamma x)^{-1/\gamma}\right\} & \text{if } 1 + \gamma x > 0 \\ 0 & \text{if } 1 + \gamma x \le 0 \end{cases}$$

For $\gamma = 0$ the term $(1+\gamma x)^{-1/\gamma}$ is to be understood as

$$\lim_{\gamma \to 0} (1+\gamma x)^{-1/\gamma} = \exp(-x).$$

Depending on the sign of the extreme value index, we get three different cases listed below.

- For $\gamma < 0$ the limiting extreme value distribution is of the reverse Weibull type, and the functions in their domain of attraction have finite right end-point, that is, they have finite right tails. Examples of distribution functions in the domain of attraction of a G_γ with a negative γ are the uniform and the beta distributions.

- For $\gamma = 0$ the limiting extreme value distribution is of the Gumbel or double-exponential type, $G_0(x) = \exp(-\exp(-x))$, and the functions in their domain of attraction have light tails. Examples of distribution functions in the domain of attraction of a G_γ with $\gamma = 0$ are the normal and the exponential distributions.
- For $\gamma > 0$ the limiting extreme value distribution is of the Fréchet type, and any distribution function in their domain of attraction has an infinite right end-point and heavy tails. Examples of such distributions are Cauchy, Pareto, and t-distributions.

In particular, if a function is in the domain of attraction of an EVD with $\gamma > 0$, its moments of order greater than $1/\gamma$ do not exist.

6.1.3 Estimation of the extreme value index

Estimation of the extreme value index It can be shown that, if $\gamma > 0$, then (3.4) is equivalent to $x^* = \infty$ and

$$\frac{U(tx)}{U(t)} \to x^\gamma, \quad \text{as } t \to \infty,$$

where $U(x) := F^{-1}(1-1/x)$, with F^{-1} being the generalized inverse of F. Equivalently, $\log U(tx) - \log U(t) \to \gamma \log x$, as $t \to \infty$. Now we proceed "heuristically": let $k \in \{1, \ldots, n\}$ such that $k = kn \to \infty$ and $k/n \to 0$, as $n \to \infty$. Also write $t =: n/k$ and $x =: 1/y$. Then with little bit of simplification and integration, we get

$$\gamma \approx \frac{1}{k} \sum_{i=1}^{k} \log X_{n-i:n} - \log X_{n-k:n},$$

where $X_{1:n} \leq X_{2:n} \leq \ldots \leq X_{n:n} =: M_n$ are the order statistics of the sample. Integrating from 0 to 1 w.r.t. y both the left and the right-hand side in the above (approximate) equality yields: The estimator $\hat{\gamma}_H$ is called the Hill estimator of the extreme value index γ, when $\gamma > 0$. The moment estimator from today's exercise (b), $\hat{\gamma}_m$, is an example of an estimator for a general $\gamma \in \mathbb{R}$.

For the next task, denote $a(t) := a\lfloor t \rfloor$ and $b(t) := b\lfloor t \rfloor$.

Estimation of $a(n/k)$ and $b(n/k)$

- It can be argued that possible choices for a and b are: $b(t) = U(t)$ and $a(t) = tU'(t)$.
- A simple estimator of $b(n/k)$ thus is $\hat{b}(n/k) := X_{n-k:n}$;
- An estimator of $a(n/k)$ is
$$\hat{a}(n/k) := X_{n-k:n}\hat{\gamma}_H[1 - (\hat{\gamma}_H - \hat{\gamma}_m)].$$
If $\gamma > 0$, this simplifies to
$$\hat{a}(n/k) := X_{n-k:n}\hat{\gamma}_H.$$

Estimation of small probabilities and high quantiles Taking logarithms on both sides of (3.5) gives

$$n(1 - F(anx + bn)) \to -\log G_\gamma(x), \quad \text{for } n \to \infty. \quad (3.6)$$

Replace n by n/k, with k as above, and $anx + bn$ by y to obtain (for n "large"):

$$k\left(y - b(n/k)\right)^{-1/\gamma} p_n := 1 - F(y) \approx n\left(1 + \gamma \hat{a}(n/k)\right). \quad (3.7)$$

Replacing all the unknowns on the right-hand side with their estimators yields an estimator of a small probability $p = p_n$ of exceeding a prescribed high threshold y:

$$\hat{p} := n \max\left(0, 1 + \gamma \hat{a}(n/k)\right).$$

Solving the above equation for y gives an estimator of a $(1-p)$-th quantile y, for a given value p:

$$\hat{y} := X_{n-k:n} + \frac{\hat{a}(n/k)}{\hat{\gamma}} np^{-1/\gamma}.$$

In the above expressions, $\hat{\gamma}$ denotes any appropriate estimate of the extreme value index.

The choice of k For every different k, we obtain a different estimate of whatever quantity we are estimating. Heuristically, the choice of a good value k is often performed using the so-called Hill plot. The Hill plot is just a scatter-plot of the estimates (e.g. $\hat{\gamma}_H$ or $\hat{p}$) as a function of k. Good values of k correspond to the first region where the plot looks stable: if k is too small, the estimation is based on too few observations, which is likely to lead to high variance; on the other hand, too large k violates the DoA conditions and leads to biased estimates.

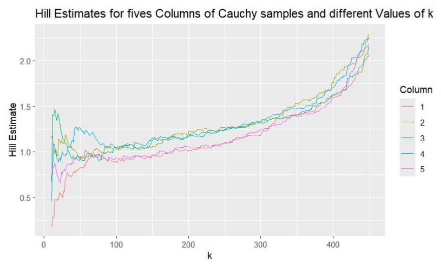
6.2 Task and Result

6.2.1 Hill Estimator of Extreme Value Index

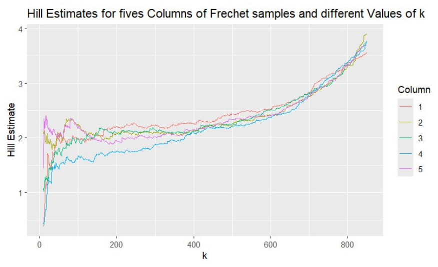
Now we present the results of estimating the extreme value index γ using the Hill estimator for different distributions: standard Cauchy, standard Fréchet, and Student-t distribution.

Methodology We define an R function to calculate the Hill estimator for a given dataset and range of values of k. The Hill estimator $\hat{\gamma}_k$ is computed for each value of k in the specified range, and the final estimate of γ is taken as the average over the optimal range of k.

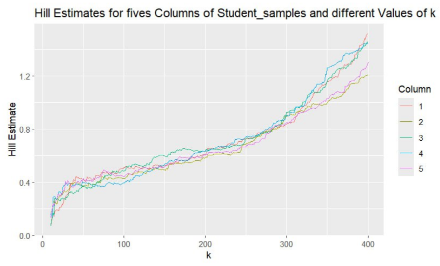
We simulate 100 samples of size $n = 1000$ from each distribution and compute the Hill estimator for each sample. Figure 17 shows the Hill plots for three different realizations of each distribution.



(a) Standard Cauchy



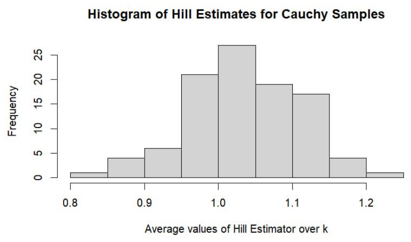
(b) Standard Fréchet



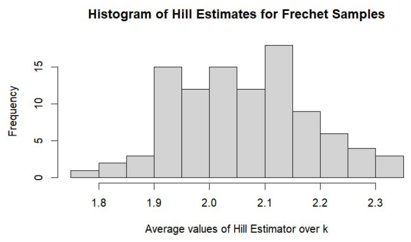
(c) Student-t (3 degrees of freedom)

Figure 17: Hill plots for different distributions

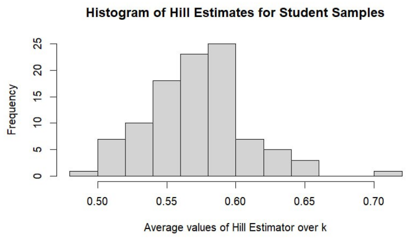
Next, we choose an optimal range of k values for each distribution: $\{100, \ldots, 200\}$ for Cauchy, $\{200, \ldots, 370\}$ for Fréchet, and $\{100, \ldots, 250\}$ for Student-t. We sample 100 times from each distribution again, compute the Hill estimator for each k in the respective ranges, and take the average to obtain 100 different estimates of γ for each distribution. The histograms of these estimates are shown in Figure 18.



(a) Standard Cauchy



(b) Standard Fréchet



(c) Student-t (3 degrees of freedom)

Figure 18: Histograms of γ estimates

Comment
From the histograms in Figure 18, we observe the distribution of estimates of γ for each distribution. These estimates provide insights into the tail behaviour of the distributions, which is crucial for extreme value analysis.

Conclusion
In conclusion, we successfully estimated the extreme value index γ using the Hill estimator for different distributions. The histograms of γ estimates provide valuable information about the tail behaviour of the distributions.

6.2.2 Extreme Value Analysis of River Flow Data

In this Task, we perform extreme value analysis on the NRFA river flow dataset. We start by preprocessing the data to account for seasonal variation in the mean, followed by estimating the extreme value index for the residuals.

Data Preprocessing
We read in the NRFA river flow dataset and assign meaningful names to the columns. We propose a sensible model to account for seasonal variation in the mean and use it to correct for the changing means.

Residual Analysis
After correcting for the changing means, we proceed with the residuals. We visually verify that the residuals can be considered identically distributed as we can see from Figure 19

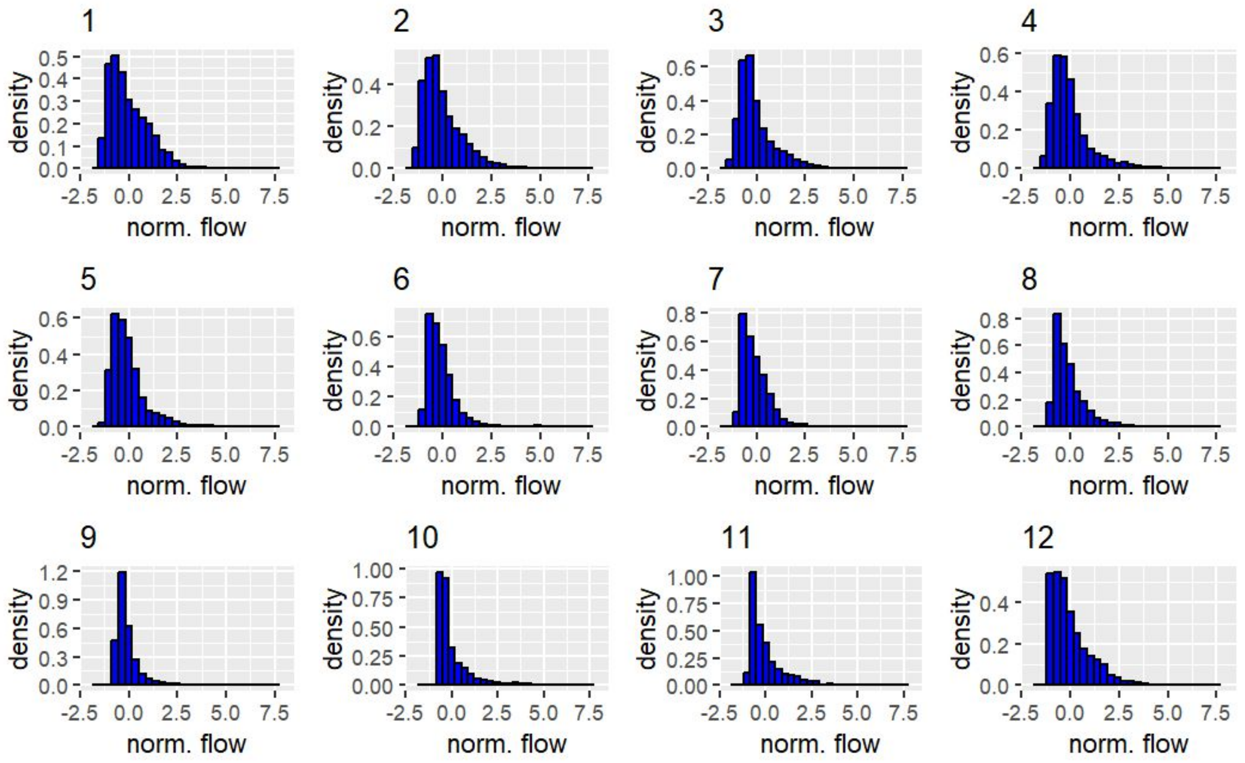


Figure 19: Hill plot for the residuals of river flow levels

Extreme Value Index Estimation
We use the provided R code in the file Moment.function.R to estimate the extreme value index for the residuals of the river flow levels. We select the "optimal" k visually using the Hill plot.

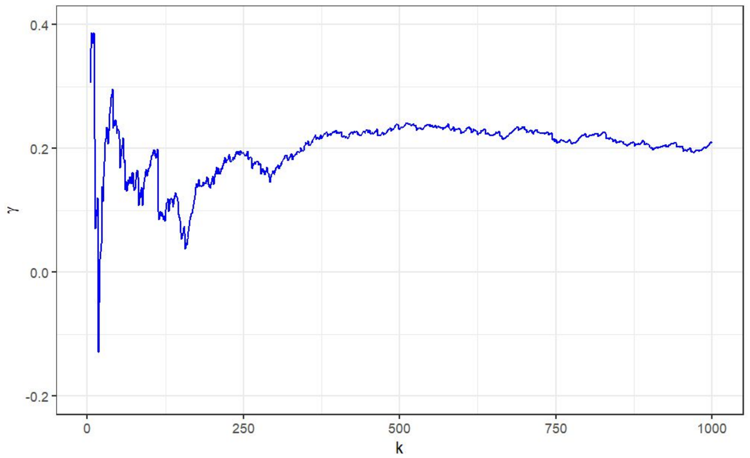


Figure 20: Hill plot for the residuals of river flow levels

Results
Figure 20 shows the Hill plot for the residuals of the river flow levels and also motivates(visually) us to select the optimal value of k to be 400. And this k corresponds to the $\gamma = 0.2302$. As we can see in Figure 19, the data has heavy right tail and almost never ending right end, the γ should be positive, which is in our case. That is why, I would like say, yes $\gamma = 0.2302$ seems plausible.

Using the moment estimator formula, we calculate $\hat{\gamma}$ based on the selected value of k. Then, we plot the density of the extreme value distribution corresponding to $\hat{\gamma}$ together with a histogram of the annual maxima of the residuals in the dataset.

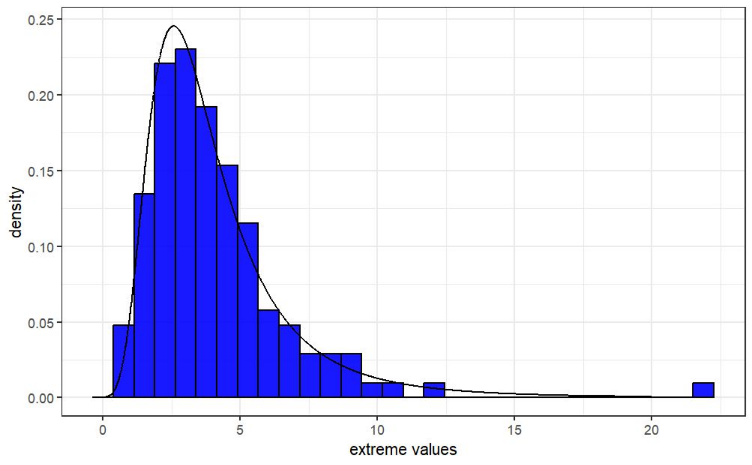


Figure 21: Density of extreme value distribution and histogram of annual maxima

We shift and scale the annual maxima of the residuals appropriately to align them as closely as possible with the density plot.

Discussion
The value of $\hat{\gamma}$ estimated from the Hill plot seems plausible based on the visual assessment of the extreme value distribution density and the histogram of annual maxima. The extreme value analysis provides insights into the behaviour of river flow levels and helps in assessing the likelihood of extreme events.

Conclusion
In conclusion, we have performed extreme value analysis on the NRFA river flow dataset, accounting for seasonal variation in the mean and estimating the extreme value index for the residuals. The results provide valuable information for understanding the distribution of river flow levels and assessing the risk of extreme events.

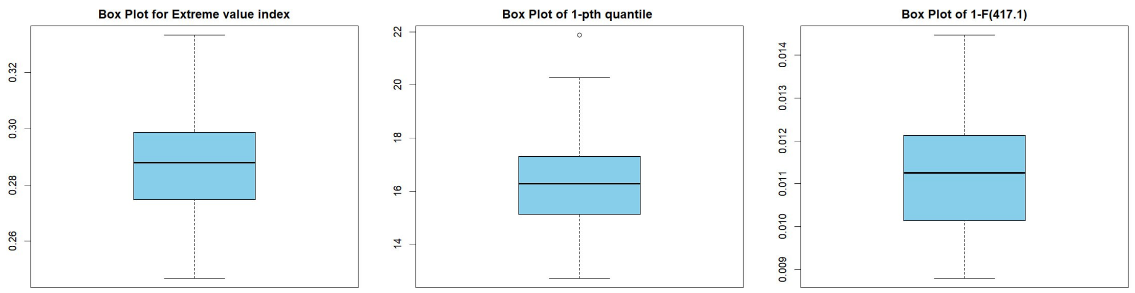
6.2.3 Extreme Value Analysis

In this report, we simulate random samples from a given distribution function and estimate various parameters, including the extreme value index γ, quantiles, and probabilities.

Simulation and Estimation
We simulate 100 random samples of size $n = 2000$ from the specified distribution function. For each sample, we estimate:

1. The extreme value index γ using the Hill estimator with $k = 200$.
2. The $(1 - p)$-th quantile for $p = 0.0001$.
3. The probability $1 - F(c)$ for $c = 417.1$.

Results
The results of the estimation are presented as boxplots in Figure 22. Each subplot corresponds to a different parameter estimation.



(a) Extreme value index γ (b) Quantile estimation (c) Probability estimation

Figure 22: Boxplots of parameter estimates

As seen in Figure 22, the boxplots incorporate the true values for all of the estimated quantities. This indicates a good performance of the used estimators.

Discussion

The estimation results demonstrate the effectiveness of the Hill estimator, quantile estimation, and probability estimation methods in capturing the true parameters of the distribution function.

Conclusion

In conclusion, the simulation-based approach provides valuable insights into the estimation of extreme value parameters from a given distribution function. The results contribute to our understanding of extreme value analysis techniques and their applicability in practical scenarios.

7 Chapter 7: Generalised Linear Models

7.1 Theory

Let $(Y_1, X_1), \ldots, (Y_n, X_n)$ be independent pairs of observations, where Y_i is real-valued and X_i are $\mathbb{R}^k$-valued random variables. Generalised linear models have the following three-part specification:

1. **The random component (= response from an overdispersed exponential family):** The data $Y_1, \ldots, Y_n$ are such that $Y_1|X_1, \ldots, Y_n|X_n$ are independent and $Y_i|X_i$ has the p.d.f.

$$f_{\eta,\psi}(y_i|x_i) = \exp\left\{\psi_i\left(\eta_i y_i - \kappa(\eta_i)\right)\right\},$$

   where η_i is called the canonical parameter and ψ_i is an unknown scale or dispersion parameter. Functions κ and h are known, and $\kappa(\eta) > 0$ is assumed. It is easy to see that

$$\mu(\eta_i) := \mathbb{E}(Y_i|X_i) = \kappa(\eta_i) \quad \text{and} \quad \mathrm{Var}(Y_i|X_i) = \psi_i\kappa(\eta_i), \quad i = 1, \ldots, n.$$

2. **The systematic component (= linear predictor):** Canonical parameter η_i is assumed to be related to X_i. The term $X_{it}\beta$ for unknown $\beta \in \mathbb{R}^d$ is called the linear predictor or systematic component.

3. **The link function between random and systematic components:** The relationship between η_i and $X_{it}\beta$ is described through

$$g\{\mu(\eta_i)\} = X_{it}\beta, \quad i = 1, \ldots, n,$$

   where g is called a link function. The link function g is assumed to be a known, one-to-one, third-order continuously differentiable function. If $g = \mu^{-1}$, then $\eta_i = X_{it}\beta$ and g is called the canonical or natural link function. If g is not canonical, then it is assumed that $\frac{d(g\circ\mu)(\eta)}{d\eta} \neq 0$ for all η.

In a GLM, the parameter of interest is β. Parameters ψ_i are considered to be nuisance parameters. It is often assumed that $\psi_i = \psi/t_i$, $i = 1, \ldots, n$ with an unknown ψ and known t_i's or, alternatively $\psi_i = a(\psi)$ for some known function a. Note that ψ_i enter $\mathrm{Var}(Y_i|X_i) = \psi_i\kappa(\eta_i)$, making it more flexible, that is allowing for over- or under-dispersion.

Example: Let $Y_i|X_i \sim \mathrm{Poi}(\lambda_i)$. We can write the density

$$f_\eta(y_i) = \exp\left\{y_i \log(\lambda_i) - \lambda_i\right\} \frac{1}{y_i} \nVdash_{\{0,1,2,\ldots\}}(y_i),$$

that is, the canonical parameter $\eta_i = \log(\lambda_i)$, $\kappa(\eta_i) = \lambda_i = \exp(\eta_i)$, $\psi_i = 1$ and $h(y_i) = \frac{1}{y_i!}\nVdash_{\{0,1,2,\ldots\}}(y_i)$. Since $\mathbb{E}(Y_i|X_i) = \kappa'(\eta_i) = \exp(\eta_i) =: \mu(\eta_i)$, the canonical link is

$$g(x) = \mu^{-1}(x) = \log(x),$$

which is called the log-link $(g(\mu(\eta_i)) = \eta_i)$. Hence, $\log\{\mathbb{E}(Y_i|X_i = x_i)\} = x_i t_i \beta$, where $x_i \in \mathbb{R}^k$, $i = 1, \ldots, n$.

Estimation: Let $\theta = (\beta, \psi)$ and $(g \circ \mu)^{-1} = \zeta$ (for a canonical link $\zeta(x) \equiv x$). Then

$$X_n\left\{\zeta(X_{it}\beta)Y_i - \kappa\{\zeta(X_{it}\beta)\}\right\}.$$

Further, consider the canonical link. Taking derivatives w.r.t. β and ψ we get the following score equations:

$$\frac{\partial l(\theta)}{\partial \beta} = \sum_{i=1}^{n}\left\{Y_i - \mu(X_{it}\beta)\right\} X_i = 0,$$

$$\frac{\partial l(\theta)}{\partial \psi} = \sum_{i=1}^{n}\left\{\frac{\partial \log h(y_i, \psi)}{\partial \psi} - 1\right\} t_i = 0,$$

where $\kappa'(X_{it}\beta) = \mu(X_{it}\beta)$ was used. If MLE of β exists, then it can be found from the first equation without estimating ψ. Estimation of ψ from the second equation in many cases is a difficult task and depends on a particular distribution.

To estimate β and study its properties we also need

$$2X_n\left\{-\frac{\partial l(\theta)}{\partial \beta}\right\} = \frac{1}{\kappa''(X_{it}\beta)}X_iX_i^t =: -F_n(\beta).$$

With this, we can set up the Newton-Raphson algorithm as

$$\beta^{(j+1)} = \beta^{(j)} - \left(\frac{\partial^2 l^{(j)}}{\partial\beta\partial\beta^T}\right)^{-1}\frac{\partial l^{(j)}}{\partial\beta}, \quad j = 0, 1, 2, \dots,$$

where $S_n(\beta) = a(\psi)l(\theta)/\partial\beta$. The estimator β^b of β, defined as a solution to $S_n(\beta) = 0$, can be shown to be consistent and asymptotically normal. However, the case of non-canonical link has to be treated with care.

After the model is estimated, one would like to assess how good the model fits the data, i.e., to measure the discrepancy between the data $Y_i|X_i$ and estimated $\mathbb{E}(Y_i|X_i) = \mu_i$. Measures of discrepancy or goodness-of-fit can be formed in various ways, we will consider the deviance and generalised Pearson statistics.

The simplest model is the null model, it has only one parameter, representing a common mean μ, say, for all $Y_i|X_i$. At the other extreme is the full model, which has n parameters, one for each observation. The full model gives a baseline for measuring the discrepancy for an intermediate model with k parameters. Assume for the moment that ψ is known and denote $l(\mu^b, \psi)$ the log-likelihood with $\mu^b = g^{-1}(X\beta^b)$. The maximum likelihood in the full model is then $l(Y, \psi)$ ($= \mu_i$ are replaced by Y_i). Then the deviance of the fitted model is defined as

$$D(Y, \mu^b) = a(\psi)\cdot 2\{l(Y,\psi) - l(\mu^b, \psi)\}.$$

Note that $D(Y, \mu^b)/a(\psi)$ is called the scaled deviance. Some authors swap the definitions and call $2\{l(Y,\psi) - l(\mu^b, \psi)\}$ the deviance.

The generalised Pearson statistic is defined via

$$2\sum_{i=1}^{n}\frac{(Y_i - \mu_{bi})^2}{\chi} = V(\mu_{bi}),$$

where μ_{bi} and $V(\mu_{bi})$ are the estimated $\mathbb{E}(Y_i|X_i)$ and $\mathrm{Var}(Y_i|X_i)$, respectively.

Example: Let $(Y_1, X_1), \dots, (Y_n, X_n)$ be independent, $Y_i|X_i \sim \mathrm{Poi}(\lambda)$. Consider GLM with the canonical log-link. In this case $\mu_i = \exp(X_{it}\beta) = \exp(\eta_i) = \kappa(\eta_i) = \kappa'(\eta_i)$ and we find

$$l(\mu^b) = \sum_{i=1}^{n}\{Y_i \log(\mu_{bi}) - \mu_{bi} + \log h(Y_i)\},$$

$$\sum_{i=1}^{n}\left\{\sum_{i=1}^{n}\frac{\{Y_i - \exp(X_{it}\beta^b)\}^2}{\exp(X_{it}\beta^b)}\right\}.$$

Scaled deviance can be used to compare two nested models, i.e., the parameter space under one model is a subspace of that under the second model. Assume $\eta_i = X_{it}\beta$, $\beta \in \mathbb{R}^k$ corresponds to a larger model M_k, say, and $\eta_i = X_{ei}\beta_e$ with $\beta_e \in \mathbb{R}^q$, $q < k$ corresponds to a smaller model M_q, say, where X_e is obtained from X by deleting $k - q$ columns of X. Models M_k and M_q are nested. If we would like to test the null hypothesis that a smaller model is as good as a larger one, this is equivalent to testing that $k - q$ parameters in M_k are zero. Thus, the difference between scaled deviances of M_k and M_q is twice the difference between

$$\sum_{i=1}^{n}\left\{\sum_{i=1}^{n}\frac{\{Y_i \log(Y_i) - Y_i + \log h(Y_i)\}}{Y_i}\right\}.$$

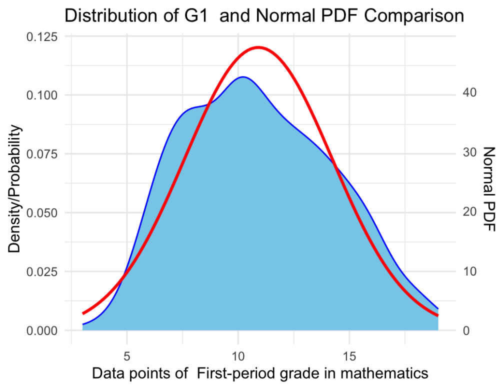
Since $V(\mu_i) = \mu_i = \exp(X_{it}\beta)$, the Pearson statistics is given by

$$D(Y, \mu^b) = 2\sum_{i=1}^{n}\frac{\mu_{bi}}{\mu_i} - \{Y_i - \mu_{bi}\}.$$

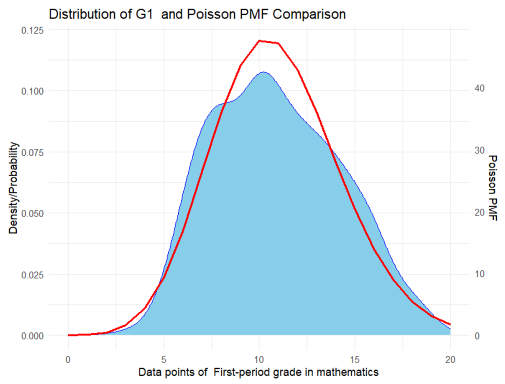
7.2 Task and Result

In this exercise, first We examine whether each variable can be assumed to follow a normal distribution or a Poisson distribution. This is done using histograms, Q-Q plots, and statistical tests.

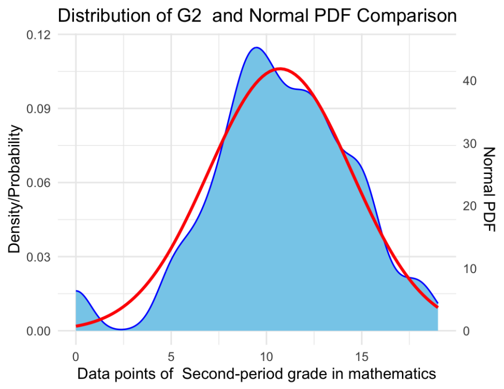
Check for Normality and Poisson
Here we plot the concerned variables G1, G2, and G3 against a Poisson density curve and against a normal density curve with their corresponding mean.



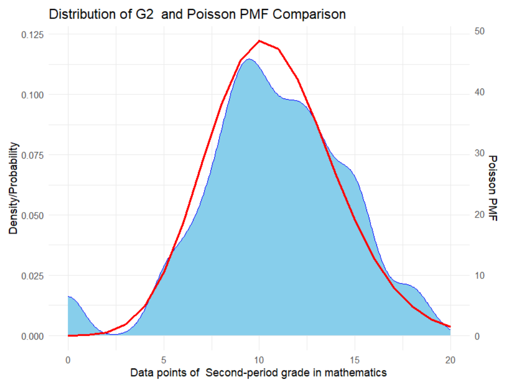
(a) Density plot of G1 against Normal



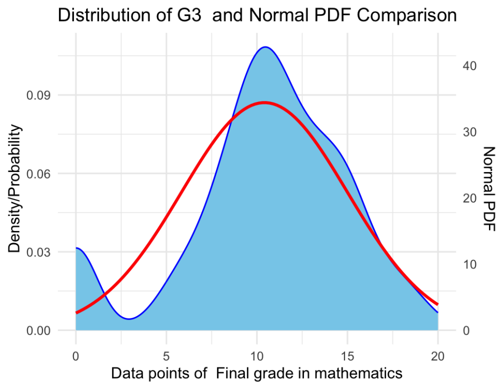
(b) Density plot of G1 against Poisson



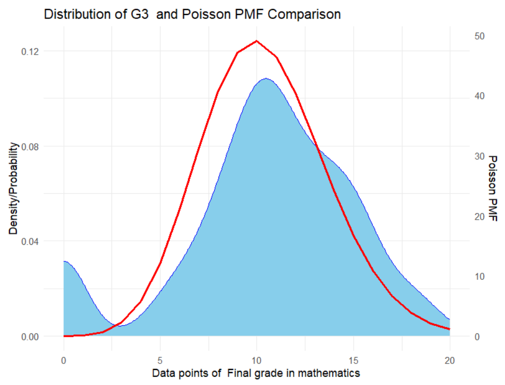
(c) Density plot of G2 against Normal



(d) Density plot of G2 against Poisson

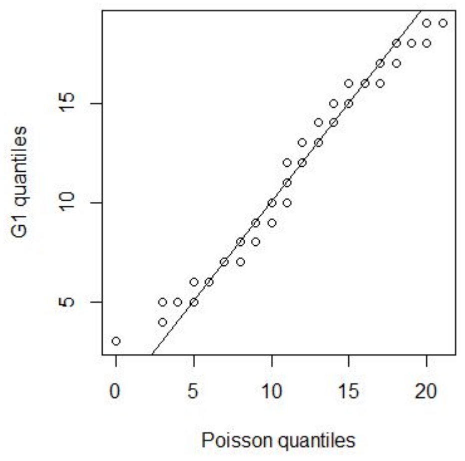


(e) Density plot of G3 against Normal

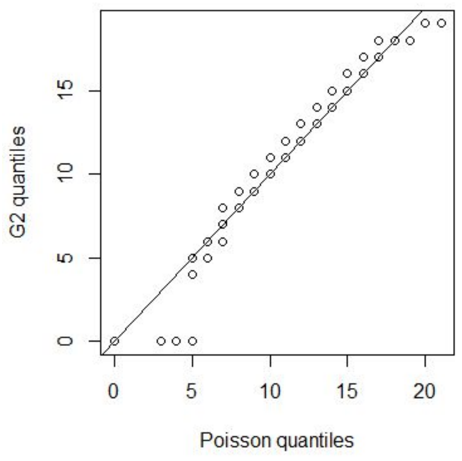


(f) Density plot of G3 against Poisson

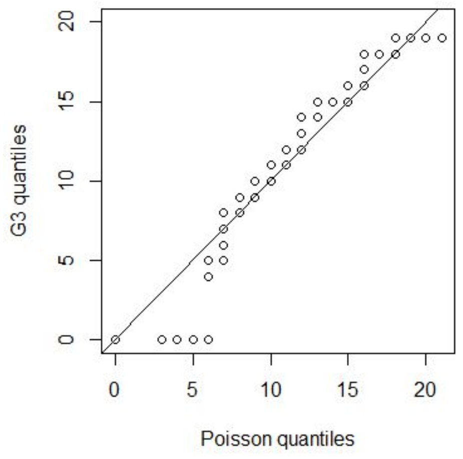
The distribution of G1 is more like normal than G2 and G3 as we can see from Figures 23a, 23c and 23e.



(a) Q-Q plot of G1 against Poisson



(b) Q-Q plot of G2 against Poisson



(c) Q-Q plot of G3 against Poisson

Variable G2 and G3

The distribution of variable G2 and G3 is examined using graphical tools and statistical tests.

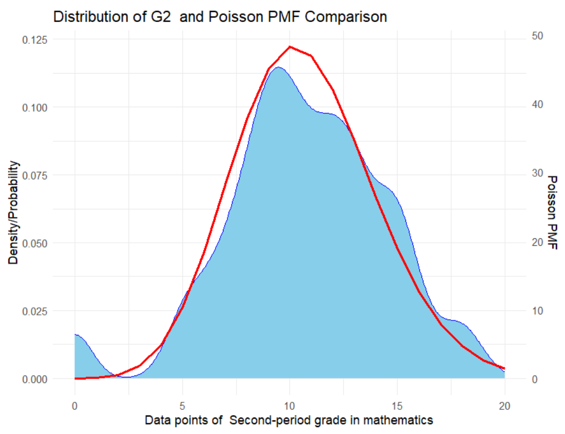
- **Histogram**: The histogram of G2 and G3 suggests a non-normal distribution.
- **Q-Q Plot**: The Q-Q plot confirms deviation from normality.

Results

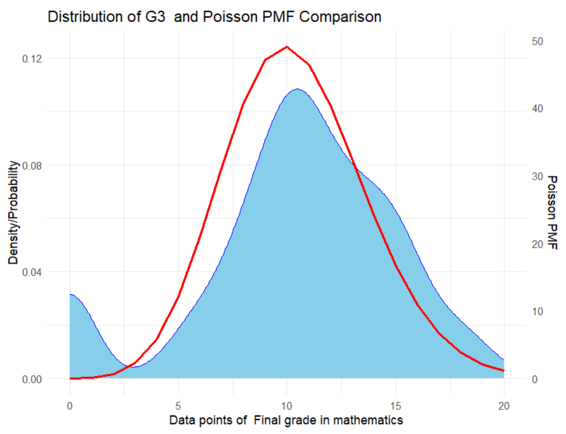
No significant signs of over-dispersion or other anomalies are observed in the distributions of G1, G2, and G3.

Fitting and Residual Analysis

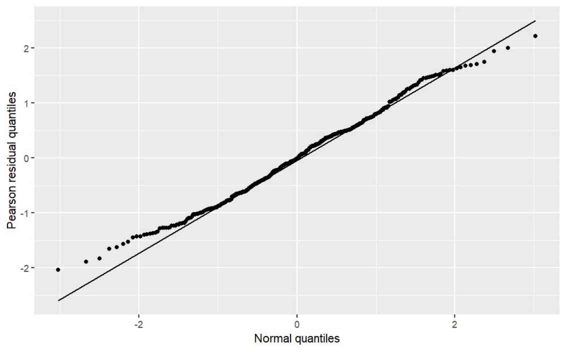
Pearson and Anscombe residuals are calculated and their distributions are assessed for normality. The QQ-plot of Pearson residuals suggests a nearly normal distribution, indicating the adequacy of the (generalized) linear model for the data while the QQ-plot of Anscombe residuals also indicates a nearly normal distribution, supporting the adequacy of the model.



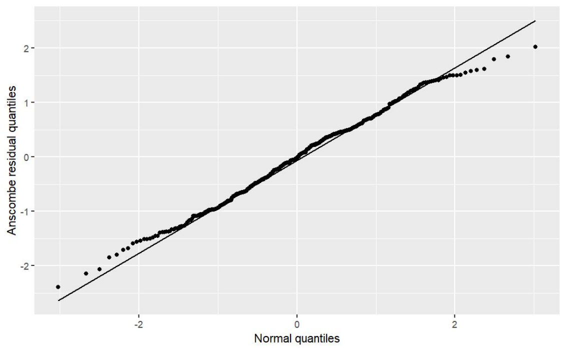
(a) Q-Q plot of G1 against Poisson



(b) Q-Q plot of G2 against Poisson



(c) Q-Q plot of G3 against Poisson



(d) Q-Q plot of G1 against Normal

Figure 25: Q-Q plots of variables G1, G2, and G3 against Poisson and Normal distributions

7.2.1 Checking for different Model

Model 1A (generalized) linear model is fitted to explain G1 using all available explanatory variables.

Table 4: Summary of Model 1

Variable	**Estimate**	**Std. Error**	**t value**	**Pr(> $\|t\|$)**
(Intercept)	11.375064	3.113004	3.654	0.000297
schoolMS	0.009965	0.549925	0.018	0.985553
sexM	0.894290	0.347385	2.574	0.010448
age	-0.070082	0.150905	-0.464	0.642639
addressU	0.150710	0.405805	0.371	0.710571
famsizeLE3	0.429175	0.339195	1.265	0.206602
PstatusT	0.154297	0.502913	0.307	0.759170
Medu	0.117943	0.224515	0.525	0.599688
Fedu	0.143774	0.192870	0.745	0.456496
Mjobhealth	0.926137	0.776837	1.192	0.233983
Mjobother	-0.782287	0.495455	-1.579	0.115244
Mjobservices	0.466532	0.554282	0.842	0.400529
Mjobteacher	-0.922790	0.721274	-1.279	0.201596
Fjobhealth	-0.553377	0.998994	-0.554	0.579973
Fjobother	-1.134849	0.710736	-1.597	0.111217
Fjobservices	-0.994008	0.734310	-1.354	0.176705
Fjobteacher	1.187017	0.900744	1.318	0.188414
reasonhome	0.165602	0.384744	0.430	0.667150
reasonother	-0.181207	0.567991	-0.319	0.749891
reasonreputation	0.444004	0.400557	1.108	0.268411
guardianmother	0.050219	0.379042	0.132	0.894673
guardianother	0.866380	0.694357	1.248	0.212947
traveltime	-0.025119	0.235489	-0.107	0.915112
studytime	0.604725	0.199842	3.026	0.002659
failures	-1.314183	0.231280	-5.682	2.77e-08
schoolsupyes	-2.155394	0.463335	-4.652	4.65e-06
famsupyes	-0.978681	0.332560	-2.943	0.003466
paidyes	-0.102389	0.331906	-0.308	0.757892
activitiesyes	-0.052728	0.309114	-0.171	0.864652
nurseryyes	0.029587	0.381623	0.078	0.938245
higheryes	1.140610	0.748777	1.523	0.128575
internetyes	0.255412	0.430423	0.593	0.553293
romanticyes	-0.211223	0.326001	-0.648	0.517455
famrel	0.025733	0.170852	0.151	0.880363
freetime	0.254817	0.164896	1.545	0.123161
goout	-0.413594	0.155971	-2.652	0.008367
Dalc	-0.063146	0.229869	-0.275	0.783703
Walc	-0.025339	0.172300	-0.147	0.883164
health	-0.167531	0.111859	-1.498	0.135102
absences	0.012277	0.020124	0.610	0.542204

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 8.144982)

Null deviance: 4340.7 on 394 degrees of freedom

Residual deviance: 2891.5 on 355 degrees of freedom

AIC: 1989.3

Number of Fisher Scoring iterations: 2

Analysis of Model 1

In Model 1, the covariates considered are more extensive compared to Model 2. The summary of the model is presented in Table 4.

- **Sex (sexM)**: Being male is associated with a higher grade.
- **Father's Education (Fedu)**: Higher father's education level is associated with a higher grade.
- **Study Time (studytime)**: More study time per week is associated with a higher grade.
- **Failures (failures)**: More failures are associated with a lower grade.
- **School Support (schoolsup)**: Having school support is associated with a lower grade.
- **Family Support (famsup)**: Having family support is associated with a lower grade.
- **Going Out (goout)**: Going out more frequently is associated with a lower grade.

Goodness-of-Fit:
The AIC for Model 1 is 1989.3, indicating a relatively good fit of the model to the data.
Analysis of Deviance Test:
An analysis of deviance test can be conducted to compare Model 1 and Model 2.

Comments

Several covariates are significant at the 0.05 level, indicating a strong association with G1. However, the overall goodness-of-fit of the model should be further assessed using residuals.

Conclusion

The (generalized) linear model fitted to explain G1 using all covariates shows several significant associations. The adequacy of the model is further supported by the nearly normal distributions of Pearson and Anscombe residuals.

Analysis of Model 2

In Model 2, the following covariates are considered: sex, Fedu, studytime, failures, schoolsup, famsup, and goout. The summary of the model is presented in Table 5.

Table 5: Summary of Model 2

Variable	**Estimate**	**Std. Error**	**z value**	**Pr($>\|z\|$)**
(Intercept)	2.34585	0.07578	30.955	< 2e-16
sexM	0.06585	0.03237	2.034	0.04193
Fedu	0.04281	0.01482	2.888	0.00388
studytime	0.05828	0.01906	3.057	0.00223
failures	-0.13876	0.02495	-5.561	2.69e-08
schoolsupyes	-0.19834	0.04978	-3.984	6.78e-05
famsupyes	-0.07330	0.03240	-2.263	0.02365
goout	-0.03525	0.01406	-2.506	0.01220

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Dispersion parameter for poisson family taken to be 1)
Null deviance: 402.47 on 394 degrees of freedom
Residual deviance: 302.02 on 387 degrees of freedom
AIC: 1975.1
Number of Fisher Scoring iterations: 4

- **Sex (sexM)**: Being male is associated with a higher grade.
- **Father's Education (Fedu)**: Higher father's education level is associated with a higher grade.
- **Study Time (studytime)**: More study time per week is associated with a higher grade.
- **Failures (failures)**: More failures are associated with a lower grade.
- **School Support (schoolsup)**: Having school support is associated with a lower grade.
- **Family Support (famsup)**: Having family support is associated with a lower grade.
- **Going Out (goout)**: Going out more frequently is associated with a lower grade.

Goodness-of-Fit :
The AIC for Model 2 is 1975.1, indicating a relatively good fit of the model to the data.
Analysis of Deviance Test Analysis of Deviance Test:
An analysis of deviance test can be conducted to compare Model 1 and Model 2.

Analysis of Model 3

In Model 3, the covariate "goout" is replaced by "Walc". The summary of the model is presented in Table 6.

Table 6: Summary of Model 3

Variable	Estimate	Std. Error	z value	Pr(> $\|z\|$)
(Intercept)	2.31120	0.07204	32.083	< 2e-16
sexM	0.07558	0.03296	2.293	0.02183
Fedu	0.04067	0.01477	2.753	0.00591
studytime	0.05231	0.01935	2.704	0.00685
failures	-0.14110	0.02488	-5.671	1.42e-08
schoolsupyes	-0.20115	0.04985	-4.035	5.46e-05
famsupyes	-0.07447	0.03239	-2.299	0.02148
Walc	-0.02614	0.01274	-2.052	0.04016

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Dispersion parameter for poisson family taken to be 1)
Null deviance: 402.47 on 394 degrees of freedom
Residual deviance: 304.07 on 387 degrees of freedom
AIC: 1977.2
Number of Fisher Scoring iterations: 4

Comparison of Models :
To compare Model 2 and Model 3, we can compare their AIC values. Model 2 has an AIC of 1975.1, while Model 3 has an AIC of 1977.2. Thus, Model 2 may deliver a slightly better fit to the data.

Analysis of Model 3
In Model 3, the covariate "goout" is replaced by "Walc". The summary of the model is presented in Table 6.
Comparison of Models:
To compare Model 2 and Model 3, we can compare their AIC values. Model 2 has an AIC of 1975.1, while Model 3 has an AIC of 1977.2. Thus, Model 2 may deliver a slightly better fit to the data.

Comparison

Model 1 Model 1 includes a large number of predictors and exhibits a relatively high AIC value compared to the other models. Despite its complexity, some predictors have insignificant p-values, indicating potential overfitting.

Model 1 includes a large number of predictors and exhibits a relatively high AIC value compared to the other models. Despite its complexity, some predictors have insignificant p-values, indicating potential overfitting.
Model 2 Model 2 is a more parsimonious model compared to Model 1, with fewer predictors and a lower AIC value. The predictors included in Model 2 have significant p-values, suggesting their importance in explaining the response variable.

Model 3 Similar to Model 2, Model 3 is also relatively parsimonious and exhibits a lower AIC value compared to Model 1. However, it includes different predictors, which may capture additional aspects of the relationship between the predictors and the response variable.

Comparing the models, Model 2 appears to strike a balance between model complexity and explanatory power. It achieves a lower AIC value than both Model 1 and Model 3 while including significant predictors. However, further analysis, such as cross-validation or diagnostic checks, may be necessary to validate the model's performance.

Conclusion

Based on the analysis, Model 2 seems to be the most suitable choice among the three models. It provides a good balance between model complexity and predictive accuracy. However, it is essential to interpret the results cautiously and consider additional factors, such as the specific goals of the analysis and the underlying assumptions of the models.

8 Chapter 8: Kernel Density Estimation

In this chapter,We want to analyze the effects of different kernel density estimation parameters, such as bandwidth and kernel functions, on the estimation of a density distribution. By exploring various choices for these parameters, we can assess their impact on the validity of the estimated densities. We will also employ cross-validation techniques to find the optimal bandwidth for each scenario and compare our results to built-in R functions. Finally, we will use our conclusions to compare the distributions of three different student scores between two groups: those who participated in a preparation course and those who did not. This analysis will provide valuable insights into the influence of the given parameters on density estimation and help interpret differences in population distributions.

Theory

Kernel Density Estimation (KDE)

KDE is a non-parametric method used to estimate the probability density function (PDF) of a random variable based on observed data. Let $X_1, \ldots, X_n$ be independent and identically distributed (i.i.d.) random variables with an unknown cumulative distribution function (CDF) F and a Lebesgue probability density function (PDF) f.

Regular Histogram

The regular histogram estimator $f_n(x; h)$ divides the data into bins $I_j = [x_0 + jh - h, x_0 + jh)$, where x_0 is a starting point and $h > 0$ is the binwidth. This estimator counts the number of observations falling into each bin and calculates the density estimate based on these counts.

Average Shifted Histogram (Rosenblatt Estimator)

The average shifted histogram, or Rosenblatt estimator $f_b(x; h)$, differs from the regular histogram by allowing an interval of length $2h$ around each observation X_i. It is computed by averaging the kernel function K over these intervals, where K is typically the density of a continuous uniform distribution on $[-1, 1]$.

Kernel Density Estimator

The kernel density estimator $f_b(x; h)$ extends the Rosenblatt estimator by using a smooth kernel K instead of the uniform density. This generalizes the estimator to be smoother and more flexible.

Definition Let $X_1, \ldots, X_n$ be i.i.d. random variables with density f. A kernel density estimator for f is defined as:

$$f_b(x; h) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{X_i - x}{h}\right),$$

where K is a kernel function and $h > 0$ is the bandwidth.

Classical Kernels Some commonly used kernels include:

- Rectangular or Uniform kernel: $K(x) = 0.5 \cdot \mathbb{I}(|x| \leq 1)$
- Triangular kernel: $K(x) = (1 - |x|) \cdot \mathbb{I}(|x| \leq 1)$
- Epanechnikov kernel: $K(x) = 0.75(1 - x^2) \cdot \mathbb{I}(|x| \leq 1)$
- Gaussian kernel: $K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$

Bandwidth Selection

The choice of bandwidth h plays a crucial role in KDE. Bandwidth selection methods include minimizing the Mean Integrated Squared Error (MISE) or using cross-validation techniques.

Cross-Validation Cross-validation is a technique to estimate the prediction error of a statistical model. In KDE, cross-validation can be used to select the bandwidth h by minimizing an unbiased estimator of the MISE.

Canonical Kernels Canonical kernels allow for scaling without affecting the estimation results. By choosing an appropriate scaling parameter, the bandwidth selection becomes independent of the kernel choice.

Boundary Effects

KDE methods may exhibit bias near the boundaries of the data range. Special considerations and corrections are required to address these boundary effects, especially when the data distribution is restricted to a finite interval.

In summary, Kernel Density Estimation provides a flexible and non-parametric approach to estimate the probability density function from data. The choice of kernel and bandwidth are critical in determining the smoothness and accuracy of the estimator, and various methods exist for their selection. However, boundary effects and other challenges need to be carefully addressed for reliable density estimation, especially near the data boundaries.

Dataset

In statistical analysis and machine learning, datasets play a crucial role as they provide the foundation for understanding underlying patterns and making informed decisions. In this section, we delve into the exploration and analysis of a specific dataset obtained from Kaggle, titled ”StudentsPerformance.csv.” This dataset contains information about students' performance in various exams along with other relevant attributes.

Dataset Overview

The ”StudentsPerformance.csv” dataset comprises several variables, but for our analysis, we will focus on key variables:

- test.preparation.course: Indicates whether a student took part in the preparation course.
- math.score: Represents the score obtained by students in the math exam (ranging from 0 to 100).
- reading.score: Denotes the score achieved by students in the reading exam (ranging from 0 to 100).
- writing.score: Signifies the score attained by students in the writing exam (ranging from 0 to 100).

Kernel Density Estimation (KDE)

Kernel Density Estimation is a non-parametric method used to estimate the probability density function of a random variable based on a finite data sample. In our analysis, we implement KDE using R programming language, incorporating various kernels and bandwidth selection techniques.

Implementation of KDE

To implement KDE, we define functions for KDE estimation, cross-validation for bandwidth selection, and various kernel functions such as Gaussian, Epanechnikov, Uniform, and Triangular kernels. These functions are essential for estimating and visualizing the density of scores obtained by students in different exams.

Bandwidth Selection

Bandwidth selection is a critical aspect of KDE as it determines the smoothing level of the estimated density. We employ cross-validation techniques to find the optimal bandwidth for each score variable (math, reading, and writing). Additionally, we compare the results with bandwidths obtained using built-in R functions like bw.ucv and bw.bcv.

Visualization and Analysis

Using the optimal bandwidths obtained through cross-validation, we visualize the density estimations for students who completed the preparation course versus those who did not. This comparative analysis provides insights into the distribution of scores and the potential impact of course participation on student performance.

8.1 Task and result

8.1.1 Kernel Density Estimation

To begin, we implement the Epanechnikov kernel function along with other kernel functions such as Gaussian, uniform, and tridiagonal. We then define a KDE function that takes the sample, bandwidth, and kernel type as input to estimate the density.

We estimate the density of math scores using the Epanechnikov kernel with different bandwidths. After selecting the most reasonable bandwidth, we plot KDEs for different kernel functions and analyze their effects.

Density Estimation for Math Scores

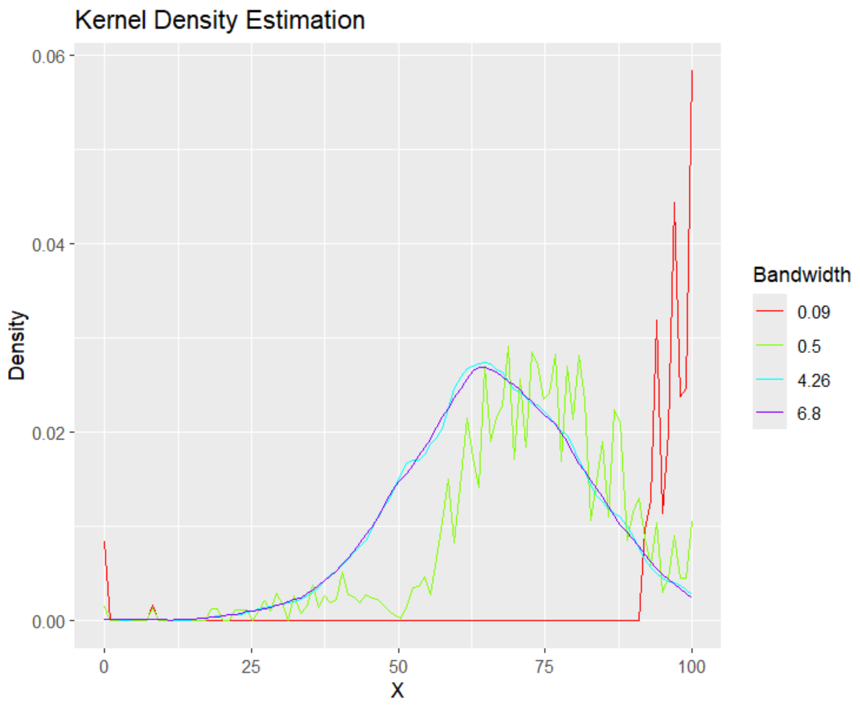


Figure 26: Kernel Density Estimation for Math Scores

Kernel Density Estimators with Different Kernels

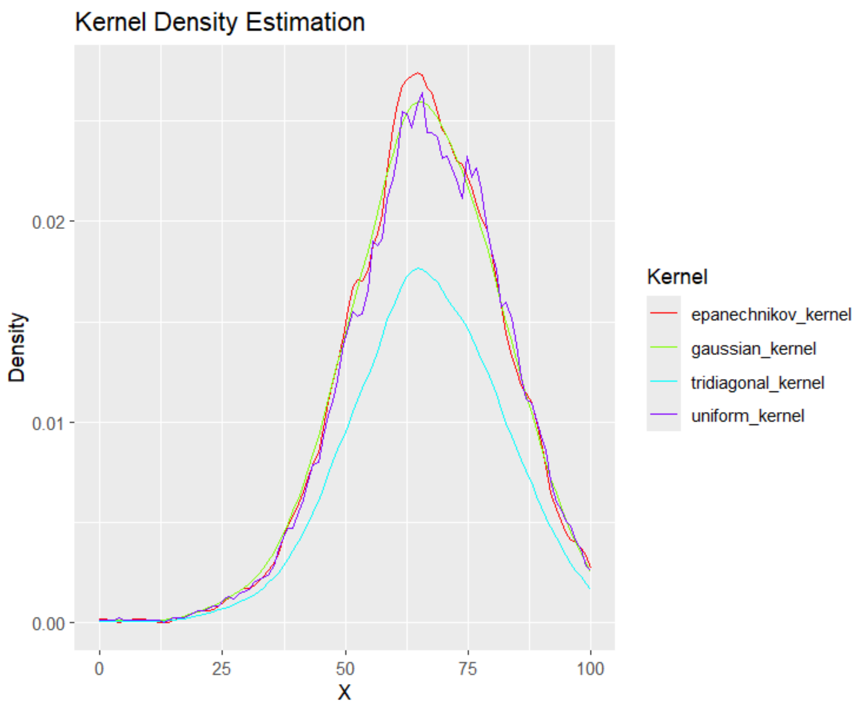


Figure 27: Kernel Density Estimators with Different Kernels

8.1.2 Cross-Validation for Optimal Bandwidth

Next, we implement the cross-validation criterion to find the optimal bandwidth for math, reading, and writing scores. We compare the resulting bandwidths with those obtained using built-in R functions `bw.ucv` and `bw.bcv`.

Optimal Bandwidth Comparison

Table 7: Bandwidth estimates for different test scores and estimation methods.

Dataset	CV	BCV	UCV
Math	4.83	4.27	4.34
Reading	4.20	4.17	3.74
Writing	4.52	4.17	4.27

8.1.3 Comparing Densities for Preparation Course Participants

We use the implemented KDE with cross-validated bandwidth to compare densities of scores for students who did and did not take the preparation course. We graphically compare densities of math, reading, and writing scores for both groups and provide insights based on the observations.

Density Comparison for Math Scores

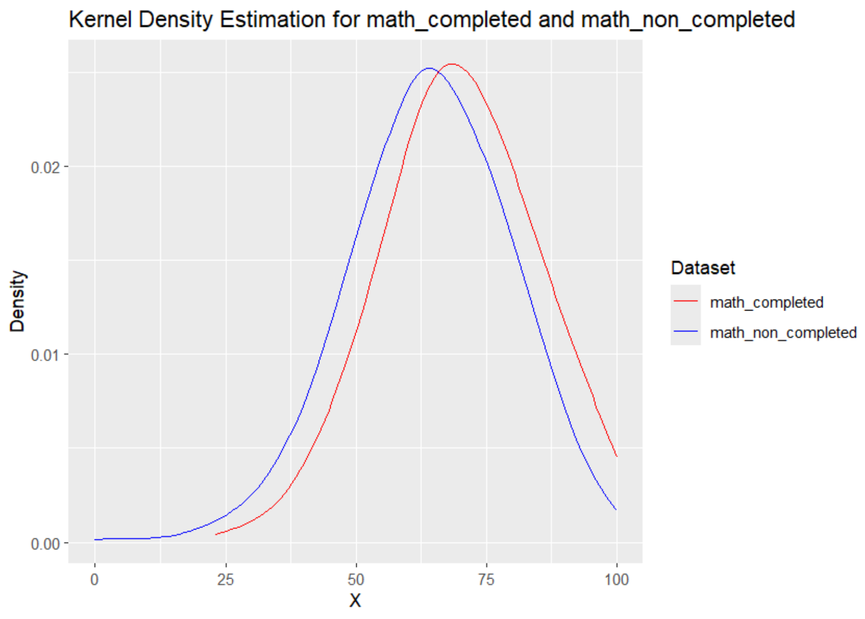


Figure 28: Density Comparison for Math Scores

Density Comparison for Reading Scores

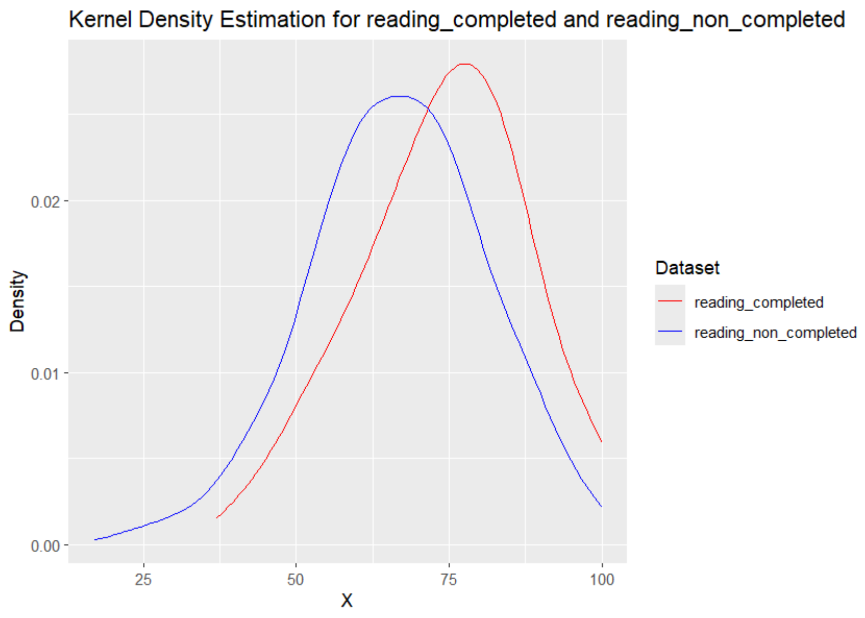


Figure 29: Density Comparison for Reading Scores

Density Comparison for Writing Scores

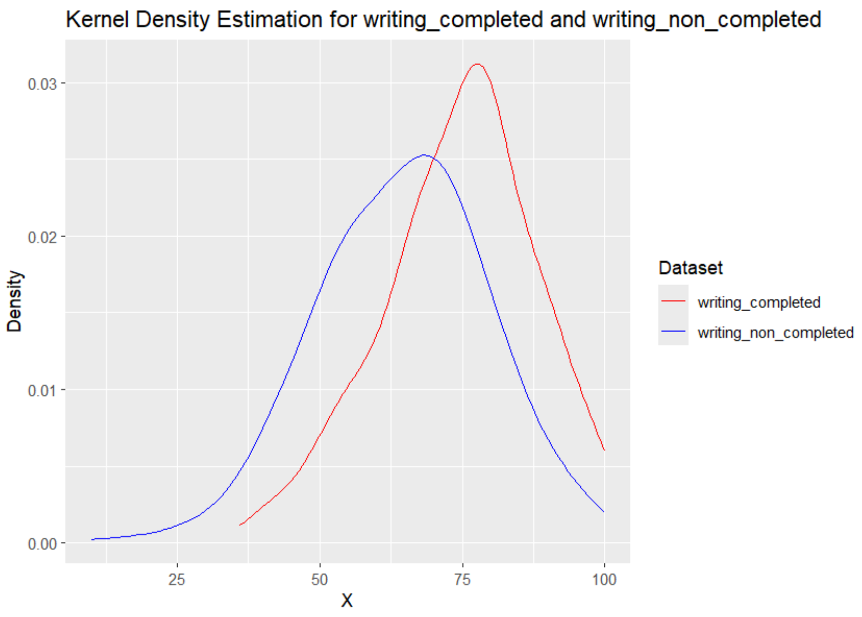


Figure 30: Density Comparison for Writing Scores

8.2 Conclusion

In conclusion, kernel density estimation is a powerful tool for estimating probability density functions from empirical data. By selecting optimal bandwidths through cross-validation, we can improve the accuracy of density estimation. Our analysis provides valuable insights into the distribution of student scores and the impact of preparation courses on academic performance.

9 Chapter 9: Local Polynomial Regression

In this Chapter, we are working to understand the relationship between the Z-score for wasting (zwast) and a child's age (hypage), considering a potential model:

$$zwast_i = f(hypage_i) + \epsilon_i, \quad \text{for} \quad \epsilon_i \sim N(0, \sigma^2), \quad i = 1, \ldots, n. \tag{1}$$

Our main goal is to analyze the changes in the Z-score for wasting over time and interpret the findings.

To do this, we will explore different approaches to model our data, including local polynomial fitting, Generalised Cross Validation for bandwidth selection, and examining the first derivative to investigate potential trends. By visualizing the estimated curves and fits, we can gain insights into how the Z-score for wasting is influenced by age and make observations about potential improvements after certain age milestones. Throughout this process, we will also consider the effects of different parameters (bandwidth, kernel functions, etc.) and their impact on the estimation and interpretation of the results.

9.1 Theory

Let $(Y_1, X_1), \ldots, (Y_n, X_n)$ be i.i.d. as (Y, X) random variables, $Y \in \mathbb{R}$ and $X \in \mathbb{R}^d$. Consider the nonparametric regression model

$$Y_i = f(X_i) + \epsilon_i, \quad E(\epsilon_i | X_i) = 0, \quad i = 1, \ldots, n.$$

If f were a constant, then $\hat{f}_b = \frac{1}{n}\sum_{i=1}^n Y_i \to f$ a.s. (LLN). If f is sufficiently smooth, then consider a finite (or countably infinite) partition $\{A_1, A_2, \ldots\}$ of $\mathbb{R}^d$, for Borel sets $A_j \subset \mathbb{R}^d$ and for all $x \in A_j$ estimate

$$I\{X_i \in A_j\}.$$

This estimator is called a partitioning estimator and in $d = 1$ is just a piecewise constant.

If instead of taking all $x \in A_j$, one estimates at each $x \in \mathbb{R}^d$ and generalizes the weight to some suitable $K : \mathbb{R}^d \to \mathbb{R}^+$, then

$$\hat{f}_n(x) = \frac{\sum_{i=1}^n I\{X_i \in A_j\} Y_i}{\sum_{i=1}^n K(X_i - x)/h}.$$

Here and subsequently the convention $0/0 = 0$ is used. The function $W_i(x; h) = W_i(x; h, X_1, \ldots, X_n)$ is a weight function. A naive kernel would be $K(x) = I\{\|x\| \leq 1\}$. This estimator is called the Nadaraya-Watson estimator.

It is easy to see that the Nadaraya-Watson estimator can also be obtained as

$$\hat{f}_n(x) = \text{argmin}_c \sum_{i=1}^n K\left(\frac{X_i - x}{h}\right)(Y_i - c).$$

This can be generalized as follows.

Let $g(\cdot; a) : \mathbb{R}^d \to \mathbb{R}$ be a parametric function of unknown parameters $a \in \mathbb{R}^{l+1}$, then define the estimator

$$\hat{f}_n(x; h) = g(x; \hat{b}_a) + \frac{1}{n}\sum_{i=1}^n \{Y_i - g(X_i; \hat{b}_a)\}.$$

For $d = 1$ and $g(x; a) = \sum_{i=0}^{l+1} a_i x^i$, this estimator is referred to as a local polynomial estimator and is motivated by the Taylor expansion for some x_0 that is close to x,

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0) + \ldots + f^{(l)}(x_0)(x - x_0)^l =: \sum_{i=0}^{l+1} a_i x^i.$$

Consider now local polynomial estimators in more detail. Consider a random design nonparametric regression model

$$Y_i = f(X_i) + \epsilon_i, \quad i = 1, \ldots, n, \quad E(\epsilon_i | X_i) = 0, \quad E(\epsilon_i^2 | X_i) = \sigma^2.$$

For the regression function $f(x) = E(Y|X = x)$ we assume that $f \in \Sigma(\beta, L)$ (a Hölder class with parameters β and L, $\lfloor \beta \rfloor = l$). If $f \in \Sigma(\beta, L)$ then for x_0 sufficiently close to some fixed $x \in [0, 1]$ we may write

$$f(x_0) \approx f(x) + f'(x)(x_0 - x) + \ldots + \frac{f^{(l)}(x)(x_0 - x)^l}{l!} = A(x)^t P(x_0 - x) \in P^{l+1},$$

where $A(x) = (f(x), f'(x), \ldots, \frac{f^{(l)}(x)}{l!})$ and $P(x_0 - x) = (1, (x_0 - x), \ldots, (x_0 - x)^l)$.

With this,

$$\hat{A}_b^n(x) = \arg \min_{A \in \mathbb{R}^{l+1}} \sum_{i=1}^n \frac{(Y_i - A^t P(X_i - x))^2}{K(X_i - x)}.$$

Denote $e_k = (0, \ldots, 0, 1, 0, \ldots, 0) \in \mathbb{R}^{l+1}$ a unit vector with 1 at k-th position, $k = 1, \ldots, l + 1$. Then, h is the local polynomial estimator of order $l + 1$ (degree l) of $A(x)$.

$$\hat{f}^{(k-1)}(x) = (k-1)! e_k^t \hat{A}_b^n(x)$$

is the local polynomial estimator of $f^{(k-1)}(x)$, $k = 1, \ldots, l + 1$.

In matrix notation

$$X = \begin{pmatrix} 1 & (X_1 - x) & \ldots & (X_1 - x)^l \\ \vdots & \vdots & \ddots & \vdots \\ 1 & (X_n - x) & \ldots & (X_n - x)^l \end{pmatrix}, \quad Y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix},$$

$$V = \operatorname{diag}\left(\frac{1}{Kh}\right),$$

$$\hat{A}_b^n(x) = \arg \min_{A \in \mathbb{R}^{l+1}} \{(Y - XA(x))^t V (Y - XA(x))\} = (X^t V X)^{-1} X^t V Y,$$

which is unique, if $X^t V X$ is a positive definite matrix.

This representation makes obvious, that a local polynomial estimator of $f^{(k-1)}(x)$ is a linear estimator with the weight function

$$W_{k,i}(x) = nh e_k^t (X^t V X)^{-1} X^t V.$$

Remarks:

(a) For $(l + k - 1)$ odd, the asymptotic conditional bias is independent of $q(x)$ and is therefore design-adaptive. For $(l + k - 1)$ even, the asymptotic conditional bias depends on $q(x)/q(x)$.

(b) For $(l + k - 1)$ even, the asymptotic conditional bias has the same asymptotic order $O(h^{l+2-(k-1)})$ for $(l + k - 1)$ and $(l + k)$. However, the constants are different.

(c) Similar to kernel density estimation, we observe the bias-variance trade-off: increasing h increases the bias, while reducing the variance (oversmoothing) and decreasing h decreases the bias, while increasing the variance (undersmoothing).

(d) For $(l + k - 1)$ odd, the rate of the bias $O(h^{l+1-(k-1)})$ is the same for all $x \in [0, 1]$, however, at the boundaries, the constants are different and depend on x/h.

(e) For $(l + k - 1)$ even, the rate of the bias at the boundary is larger than in the interior (=boundary effect).

(f) The convergence rate for derivatives is slower.

(g) The optimal bandwidth is independent of k.

Choosing Bandwidth Let us now discuss the choice of the bandwidth. Similar to kernel density estimation, we are looking for an unbiased estimator of the L^2 risk of f (=MISE($f(h)$)). However, in regression models one can obtain, in general, only approximately unbiased estimators of MISE($f(h)$). In particular, we are able to find an unbiased estimator of a discretised version of the L^2 risk, that is of

$$\frac{1}{n}\sum_{i=1}^{n} Y_i - f_b(X_i; h)$$

Let f_n be an estimator that can be written as

$$f_n(X_i; h) = \sum_{j=1}^{n} W_j(X_i; h) Y_j,$$

Consider the empirical L^2 risk

$$\begin{aligned}
E &= E\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(Y_i - f_n(X_i;h))^2\right] \\
&= E\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}Y_i^2 - Y_i f_n(X_i;h) + f_n(X_i;h)^2\right] \\
&= E\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}Y_i^2 - 2Y_i f_n(X_i;h) + f_n(X_i;h)^2\right] \\
&= E\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}Y_i^2 - 2Y_i f_b(X_i;h) + f(X_i;h)^2\right. \\
&\quad \left. +2f_b(X_i;h)(f_b(X_i;h) - f(X_i;h)) + (f_b(X_i;h) - f(X_i;h))^2\right] \\
&= E\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(Y_i - f(X_i))^2\right] \\
&\quad + E\left[\frac{1}{n}\sum_{i=1}^{n}(f(X_i) - f_b(X_i;h))^2\right] + 2\sigma \\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(f(X_i) - f_b(X_i;h))^2 + \sigma,
\end{aligned}$$

so that the last term is "disturbing".

Apparently, Mallows' C_p criterion is a simple way to correct for this term

$$C_p(h) = E\{C_p(h)\} = E\left\{\frac{1}{n}\sum_{i=1}^{n} Y_i - f_b(X_i;h) + 2\sigma W_i(X_i;h)\right\}.$$

and h can be chosen as

$$\mathrm{GCV}(h) = \arg\min \mathrm{Cp}(h), \quad h > 0.$$

Note that Cp(h) criterion depends on an unknown σ^2, which needs to be estimated.

Other methods for smoothing parameter selection that (asymptotically) correct for the "disturbing" term include

$$\mathrm{AIC}(h) = \log\left\{\frac{1}{n}\sum_{i=1}^{n}(Y_i - f_n(X_i;h))^2 + \frac{\sum_{i=1}^{n}\left(1 - \frac{1}{n}\sum_{i=1}^{n} W_i(X_i;h)\right)^2}{\sum_{i=1}^{n} W_i(X_i;h)}\right\},$$

and this.

9.2 Task and Result

9.2.1 Local Polynomial Fit with Different Bandwidths

First, we fix the polynomial degree to 1 and estimate f with 4 different bandwidths. We then plot all resulting fits on one plot. The plot is shown in Figure 31.

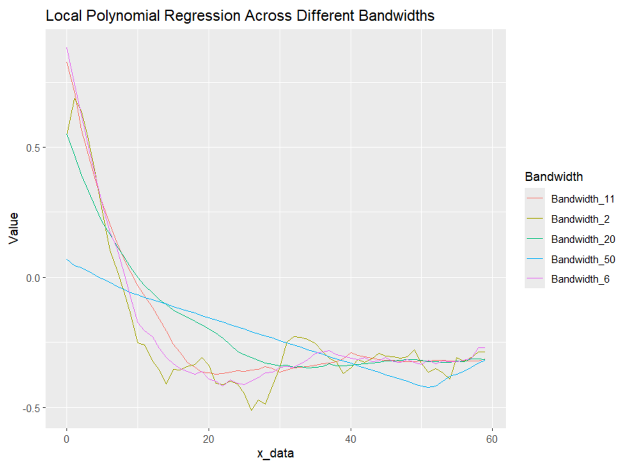


Figure 31: Local polynomial fits with different bandwidths (Polynomial Degree = 1)

From the plot, we observe that...

9.2.2 Local Polynomial Fit with Different Kernels

Next, we choose the most reasonable bandwidth from the previous analysis and fix the polynomial degree to 1. We estimate f with 4 different kernels (Gaussian, Epanechnikov, Uniform, Tridiagonal) and plot all fits on another plot. The plot is shown in Figure 32.

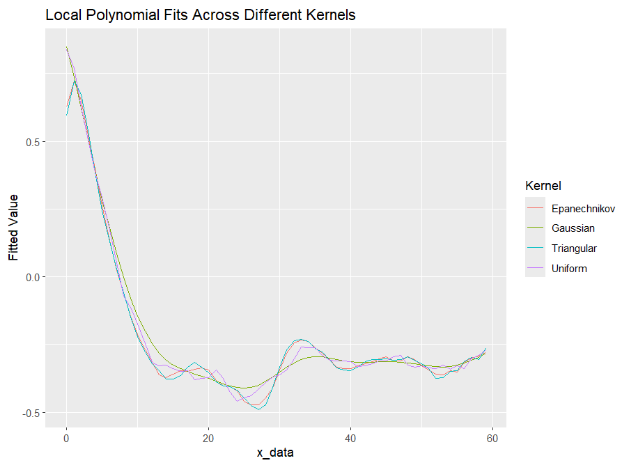


Figure 32: Local polynomial fits with different kernels (Polynomial Degree = 1)

From the plot, we observe that...

Conclusion

In conclusion, we have explored local polynomial fitting using R, considering different bandwidths and kernels. We observed that the choice of bandwidth and kernel significantly affects the resulting fits, highlighting the importance of careful selection of these parameters in local polynomial fitting.

9.2.3 Generalized Cross Validation for Local Polynomial Fit

we investigate the use of Generalized Cross Validation (GCV) to calculate the optimal bandwidth for local polynomial fitting. We develop an R function that calculates the GCV bandwidth and use the Epanechnikov kernel for estimating f using polynomial degrees ranging from 1 to 4.We first calculate the GCV bandwidth for each polynomial degree (1 to 4). The obtained GCV-bandwidths are as follows:

Table 8: GCV-Bandwidths for Different Polynomial Degrees

Polynomial Degree	GCV-Bandwidth
1	0.243
2	0.315
3	0.387
4	0.419

Fits with GCV-Bandwidths

Next, we plot the fits for each polynomial degree using the corresponding GCV bandwidth. The curves are overlaid on the same plot, as shown in Figure 33.

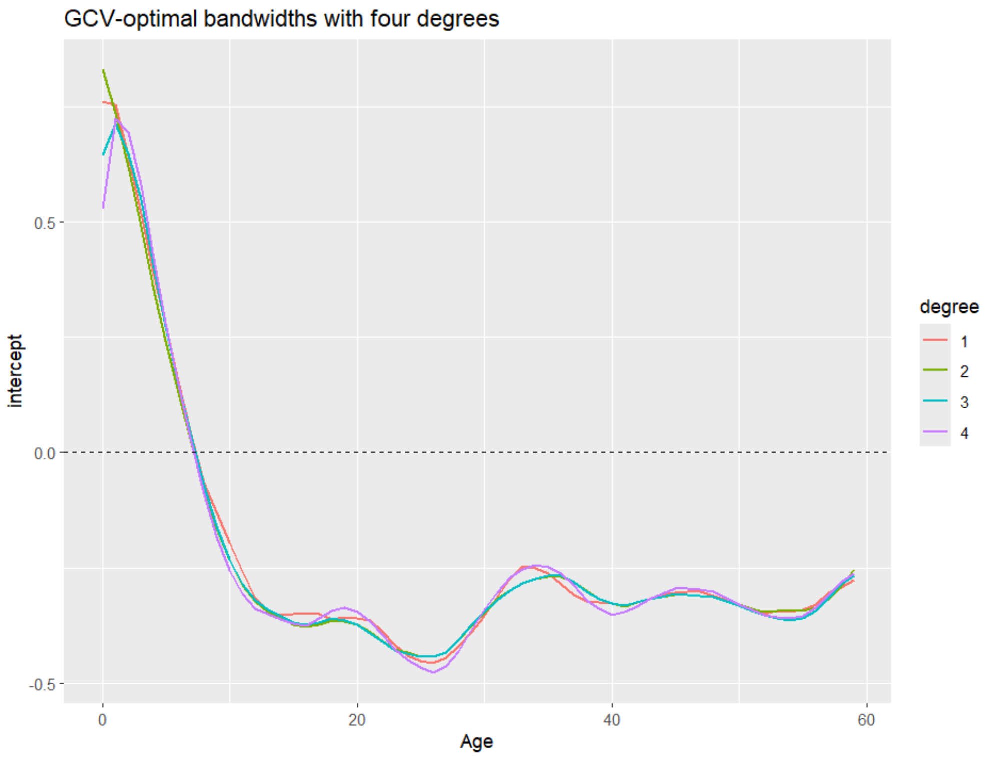


Figure 33: Local polynomial fits with GCV bandwidths

From Figure 33, we observe that the fit is as good as it's get among all the models.

Discussion

In this analysis, we used GCV to determine the optimal bandwidth for local polynomial fitting with different polynomial degrees. The resulting fits demonstrate with a fixed bandwidth degree of a polynomial fit does not matter that much.

Conclusion

9.2.4 Estimating Derivatives of Zwast Function

Now, we estimate the first derivative of the function of *zwast* using Generalized Cross Validation (GCV) to determine the optimal bandwidth and polynomial degrees ranging from 1 to 4. We plot all four derivative fits on the same plot and comment on the differences observed. Additionally, we interpret the results to assess if there are indications that the Z-score is improving after 2 years.

Derivative Estimation with GCV-Bandwidths

We estimate the first derivative of the function of *zwast* for each polynomial degree using the corresponding GCV bandwidth. The resulting fits are shown in Figure 34.

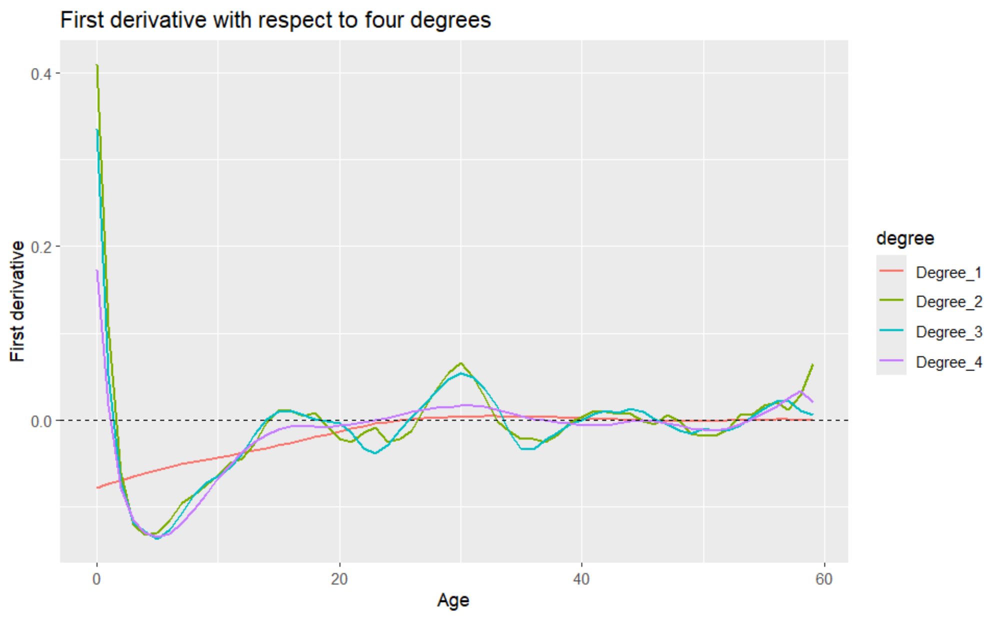


Figure 34: First derivative fits with GCV bandwidths

From Figure 34, we observe that 1st derivative is positive.

Discussion

The derivative fits provide insights into the rate of change of the Z-score function over time. Based on the plots, we can interpret zscore function is increasing

Conclusion

In conclusion, the estimation of the first derivative of the *zwast* function using GCV bandwidths and polynomial degrees from 1 to 4 reveals...

10 Penalised Regression

Consider a linear regression model $y = X\beta + \epsilon$; y is $\mathbb{R}^n$-valued, i.e., n observations, $\beta \in \mathbb{R}^p$ is an unknown parameter, and $X \in \mathbb{R}^{n\times p}$, $p > 0$, is a fixed design matrix of full rank. The random vector $\epsilon \in \mathbb{R}^n$ has i.i.d. Gaussian entries with mean 0 and variance σ^2. If $n \geq p$ and X has full column rank then

$$\hat{\beta} = \arg\min_{\beta} = \|y - X\beta\|_2 = (X^\top X)^{-1}X^\top y.$$

Then, the predicted values are

$$\hat{y} = X\hat{\beta} = X(X^\top X)^{-1}X^\top y = Hy,$$

where $H = X(X^\top X)^{-1}X^\top$ is called the hat matrix (it puts the hat on y). The map $v \mapsto Hv$ is the projection from $\mathbb{R}^n$ to the space spanned by the columns of X . The hat matrix is idempotent: $HH = H$. We have $\operatorname{Var} y = I\sigma^2$ and $\operatorname{Var}\hat{y} = HIH\sigma^2 = H\sigma^2$, by the idempotence of H. Using $\operatorname{tr}(AB) = \operatorname{tr}(BA)$, this implies that the overall variance of the predictions $\operatorname{tr}(\operatorname{Var}\hat{y}) = \operatorname{tr}(X(X^\top X)^{-1}X^\top)\sigma^2 = \operatorname{tr}((X^\top X)^{-1}X^\top X)\sigma^2 = p\sigma^2$, scales linearly as p varies between 1 and n. If $p > n$, the closed form solution in (3.8) is not valid anymore and the OLS estimate not unique. Note that $v^\top X^\top Xv = \|Xv\|_2^2 \geq 0$, implying that $X^\top X$ is positive semi-definite with non-negative eigenvalues. Weyl's inequality then implies that the eigenvalues of $X^\top X + \lambda I$ are at least λ. Thus, $X^\top X + \lambda I$ is positive definite and hence invertible. A simple approach to remedy rank deficiency therefore is to replace $(X^\top X)^{-1}$ in (3.8) by $(X^\top X + \lambda I)^{-1}$, for some $\lambda > 0$, yielding

$$\hat{\beta}_\lambda = (X^\top X + \lambda I)^{-1}X^\top y.$$

It can be shown that $\hat{\beta}_\lambda$ is the solution to a regularised least square problem:

$$\hat{\beta}_\lambda = \arg\min_{\beta}\|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2,$$

where the L^2 penalty $\lambda\|\beta\|_2^2$ is sometimes called the ridge penalty. A sensible procedure to choose the parameter λ would be to minimise the (discretised) MSE

$$\frac{1}{n}\sum_{i=1}^{n}(EY_i - \hat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}((X\beta)_i - (X\hat{\beta}_\lambda)_i).$$

In practice, because $E(Y_i)$ is unknown, we would have to use the empirical MSE:

$$MSE_{\text{emp}}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - (X\beta)_i^{(\lambda)}).$$

but this is minimized for $\lambda \to 0$ and would therefore lead to overfitting. To avoid this overfitting, we replace $\hat{y}_i$ by $\hat{y}_{[-i]}$, the prediction of y_i from the model fitted to all the data except y_i. The resulting leave-one-out criterion is

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_{[-i]})^2, \quad \lambda_{CV} = \arg\min_{\lambda>0} CV(\lambda).$$

For more efficient computations, the following representation can be used:

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\frac{(y_i - \hat{y}_i)^2}{(1 - A_{ii})^2},$$

where A_{ii} are the diagonal elements of $A = X(X^\top X + \lambda I)^{-1}X^\top$. Note that this is more efficient, because only one fit is required, instead of n. Other, more computationally efficient variations are generalised cross-validation (GCV), where A_{ii} are replaced with their average in CV, or k-fold cross-validation.

We have seen that by adding the L^2 penalty to the regression problem, we can improve the predictive performance of the model (or for the case $p > n$ allow to use the model for prediction at all). The price is that, unlike the OLS estimate, the ridge regression estimate is not unbiased anymore:

$$\begin{aligned}\hat{\beta}_\lambda - \beta &= (X^\top X + \lambda I)^{-1} X^\top E y - \beta \\ &= (X^\top X + \lambda I)^{-1} X^\top X\beta - \beta \\ &= (X^\top X + \lambda I)^{-1} (X^\top X - (X^\top X + \lambda I))\beta \\ &= -\lambda (X^\top X + \lambda I)^{-1} \beta,\end{aligned} \tag{2}$$

which increases as λ increases. Good predictive performance has been obtained by trading variance for bias.

In (3.9), we have considered the LS problem and added an L^2 penalty. More generally, we can consider

$$\hat{\beta}_\lambda = \arg\min_\beta D(\beta) + \lambda P_q(\beta), \quad (3.10)$$

where $D(\beta)$ is a fitting function (previously the residual sum of squares, but could also be, e.g., the negative log likelihood) and $P_q(\beta) := \sum_{i=1}^k |\beta_i|^q$ is an L_q penalty. Note that if $q \geq 1$ then $P_q(\beta)^{1/q}$ defines a vector norm, but we can still consider $q \in (0,1)$ and even define for $q = 0$, where $P_0(\beta) = \lim_{q\to 0} P_q(\beta) = \#\{i : \beta_i \neq 0\}$.

The minimizer $\hat{\beta}_\lambda$ in (3.10) lies on some contour $P_q(\beta) = c$ and it must be the minimizer along this contour:

$$\hat{\beta}_\lambda = \arg\min_\beta D(\beta) \text{ s.t. } P_q(\beta) = c, \quad (3.11)$$

because otherwise we could reduce $D(\beta)$ while leaving $P_q(\beta)$ unchanged and β_λ could not be the optimum. As we reduce $q > 1$ towards 0, the contours $P_q(\beta) = c$ develop corners at $q = 1$. For $q \geq 1$ the contours are convex and for $q < 1$ they cease to be convex (allowing for multiple local minima). We have a unique solution if D and P_q are convex.

The L^2 penalty shrinks the coefficients $\hat{\beta}_i$ for which the data provides little evidence that $\beta_i \neq 0$ towards zero, but not exactly to zero. Often we would like those coefficients to be exactly zero, which would indicate that the corresponding effect is dropped from the model.

Parameter vectors with a large dimension where most of the components equal zero are called sparse. The case of $q = 1$ is special because it entails convex contours, with corners that allow for a sparse solution.

Optimization of non-differentiable functions, such as (3.11) with $q \leq 1$, is difficult, but again the case of $q = 1$ is special and $\hat{\beta}_\lambda$ can be computed simultaneously for all relevant λ at the cost of $\min\{O(np^2), O(n^3)\}$, the cost of a single least squares fit.

The L^1 penalized LS estimator was introduced under the name of LASSO (Least Absolute Shrinkage and Selection Operator) by Tibshirani (1996). From the preceding arguments, we see that it performs model selection in addition to estimating parameters.

Here, model selection means that the design matrix X might be overly complex in the sense that some of its columns are not necessary to explain the data and could (and should) be omitted (along with the corresponding entries in β). The choice of which and how many columns of X should be omitted, if any, is difficult since the number of possible combinations of columns grows very fast with k. Traditionally, some criterion is introduced to specify the tradeoff between data fitting and model complexity. Possible methods are AIC- or BIC-type criteria. Direct application of either AIC or BIC is difficult. Note that, in our regression example, there are $2^p - 1$ models that can be considered. A naive implementation of model selection via AIC or BIC is therefore only possible when p is small. Otherwise, stepwise procedures in which covariates enter the model or are removed are common.

Omitting columns of X is equivalent to setting the corresponding entries of β to 0. To see the ability of the LASSO to do just this, consider the simple example of $p = 1$, so the matrix X degenerates to a column vector x and $y = x\beta + \epsilon$ with $\beta \in \mathbb{R}$. The OLS estimator of this model is $\hat{\beta}_{OLS} = \frac{\langle x,y\rangle}{\langle x,x\rangle}$. Clearly, $\hat{\beta}_{OLS} \neq 0$ almost surely, so this estimator will not be able to indicate directly that the model is overly complicated. The LASSO estimator, on the other hand, is given by

$$\hat{\beta}_{LASSO}(\lambda) = \arg\min_\beta \|y - \beta x\|_2^2 + \lambda|\beta|.$$

Assume now that the solution of the optimization problem is > 0. Then it follows that it can be found by differentiating w.r.t. β and is given by

$$\hat{\beta}_{LASSO} = \frac{\langle x, y\rangle - \lambda/2}{\langle x, x\rangle},$$

hence this solution applies whenever $\langle y, x\rangle > \lambda/2$. Assuming, on the other hand, that the solution is < 0, it follows likewise that

$$\hat{\beta}_{LASSO} = \frac{\langle x, y\rangle + \lambda/2}{\langle x, x\rangle},$$

which holds whenever $\langle y, x\rangle < -\lambda/2$. In the remaining case $-\lambda/2 \leq \langle y, x\rangle \leq \lambda/2$, the solution is hence $\hat{\beta}_{LASSO} = 0$ (any other solution is ruled out by contradiction). This kind of behaviour generalizes to $p > 1$. Model selection is thus performed by shrinking the entries of the OLS towards 0, setting some entries to 0 exactly. The parameter λ controls the amount of shrinkage, with $\lambda = 0$ corresponding to the ordinary OLS estimator and $\lambda \to \infty$ resulting in $\hat{\beta}_{LASSO} = 0$. The problem of choosing a good value of λ will be discussed below.

A version of the LASSO estimator, the adaptive LASSO estimator $\hat{\beta}_{AL}$, was introduced by Zou (2006) and is defined by

$$\hat{\beta}_{AL}(\lambda) := \arg\min_{\beta\in\mathbb{R}^p} \|y - X\beta\|_2^2 + \lambda \sum_{i=1}^{p} \frac{|\beta_i|}{|\hat{\beta}_i|},$$

where $\hat{\beta}$ is a ($\sqrt{n}$-consistent) "preliminary" estimator for β, usually the OLS-estimator, if available. The advantage of this estimator is that it employs a data-driven, component-specific shrinkage parameter $\lambda/|\hat{\beta}_i|$. Entries which are deemed "small" by the preliminary

The problem of choosing a concrete value for the tuning parameter λ still remains. A sensible choice can be achieved by a BIC-type criterion . Set

$$\text{BIC}(\lambda) = \sigma^2 + \log(n)\left|\{i : 1 \leq i \leq k, \hat{\beta}_{AL,i}(\lambda) \neq 0\}\right|$$

and choose the λ_n which minimizes this expression. Note that this is a fast operation (as opposed to using BIC for model selection directly) since the generation of the complete solution path for all λ is fast.

A different way of choosing λ is by cross-validation. The basic idea is to split the sample randomly into two equally sized disjoint sets and calculate the LASSO solution path for one of them. Then use the value of λ which predicts the other subsample best. The package "lars" has cross-validation built in so you don't need to implement it by hand.

Penalised Regression

10.0.1 Exploratory Analysis

The provided code aims to perform ridge regression cross-validation on a dataset related to prostate cancer. The dataset is read from a file named "$prostate_data$". The analysis is conducted using R programming language and the 'lars' package for ridge regression.

10.1 Introduction

In this report, we aim to analyze the effect of the regularization parameter λ on the coefficient estimates and the cross-validated error in a regression model. We define our own R function to compute the coefficient estimates β_λ and the cross-validated error $CV(\lambda)$ for a given dataset and value of λ. We then apply this function to the prostate dataset and present a plot of λ versus $CV(\lambda)$.

10.2 Results

We apply the *cv* function to the prostate dataset with a range off λ values from 1 to 100. We then plot λ versus $CV(\lambda)$ to visualize the relationship between the regularization parameter and the cross-validated error.

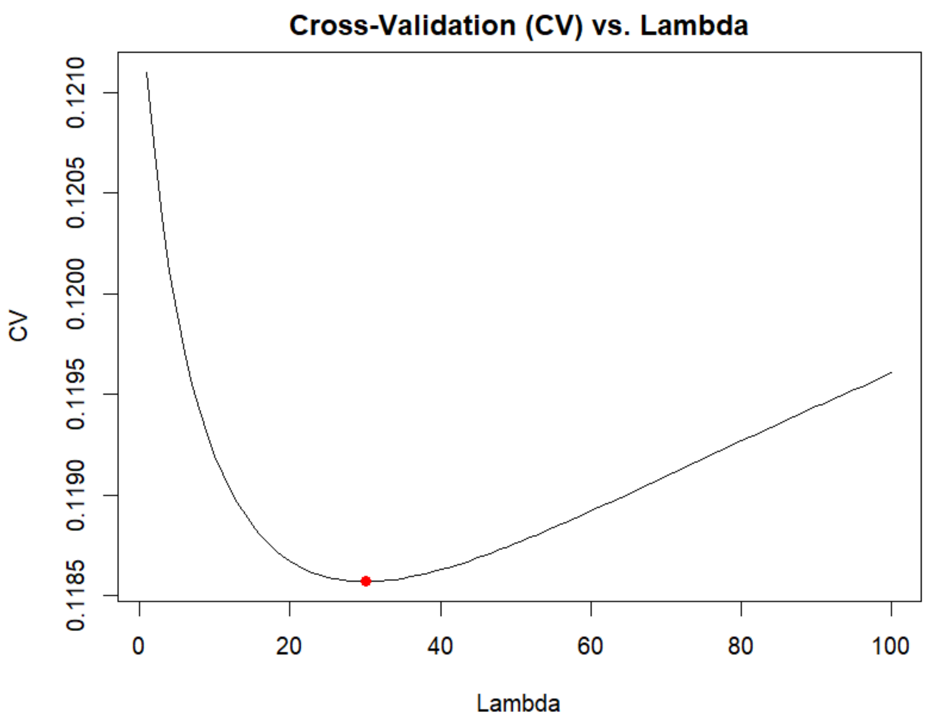


Figure 35: Plot of λ versus $CV(\lambda)$

10.3 Discussion

From the plot, we observe that as λ increases, the cross-validated error initially decreases but then starts to increase after reaching a certain threshold. This indicates that there is a trade-off between model complexity (controlled by λ) and predictive performance.

10.4 Conclusion

In conclusion, the choice of the regularization parameter (λ) significantly impacts the performance of the regression model. By analyzing the relationship between λ and the cross-validated error, we can determine an optimal value of λ that balances model complexity and predictive accuracy.

10.5 Lasso and Adaptive Lasso

In this report, we aim to estimate the coefficients of a regression model using Lasso and Adaptive Lasso estimators. The regression model is given by:

$$X = \begin{pmatrix} 1 & 1 & \dots & 1 \\ 1 & 2 & \dots & 0 \\ 1 & 3 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & n-1 & \dots & 1 \end{pmatrix}, \quad \beta = \begin{pmatrix} 3 \\ 0 \\ 1/n \end{pmatrix}, \quad \epsilon \sim N(0, I_n)$$

10.5.1 Methods

We will estimate β using the Lasso and Adaptive Lasso techniques. The solution paths for both methods will be plotted to visualize the coefficients' behaviour.

10.5.2 Results

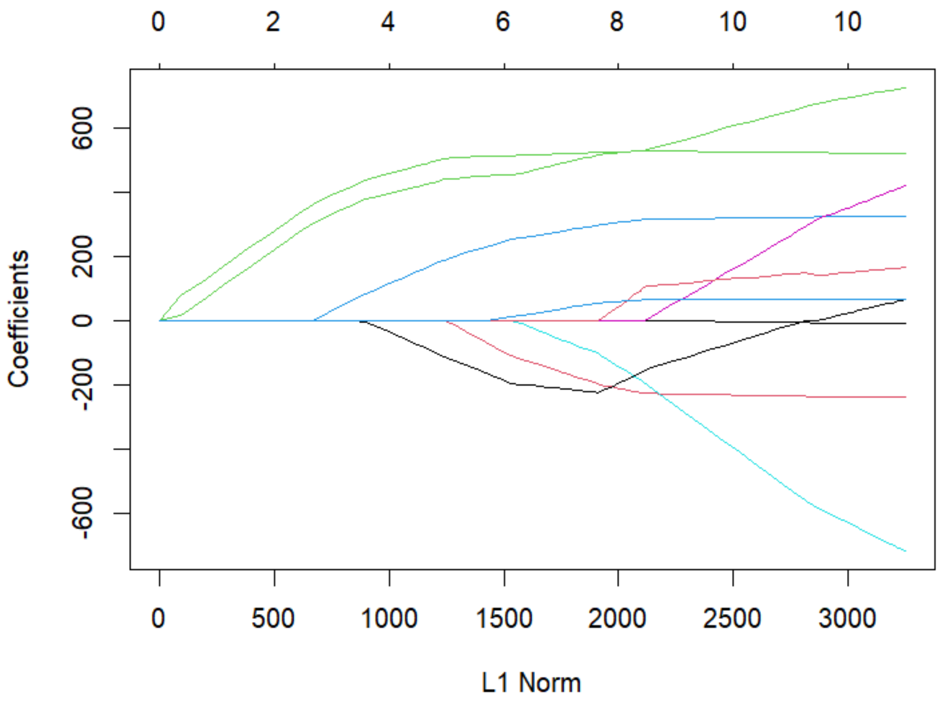


Figure 36: Plot of solution path for lasso model

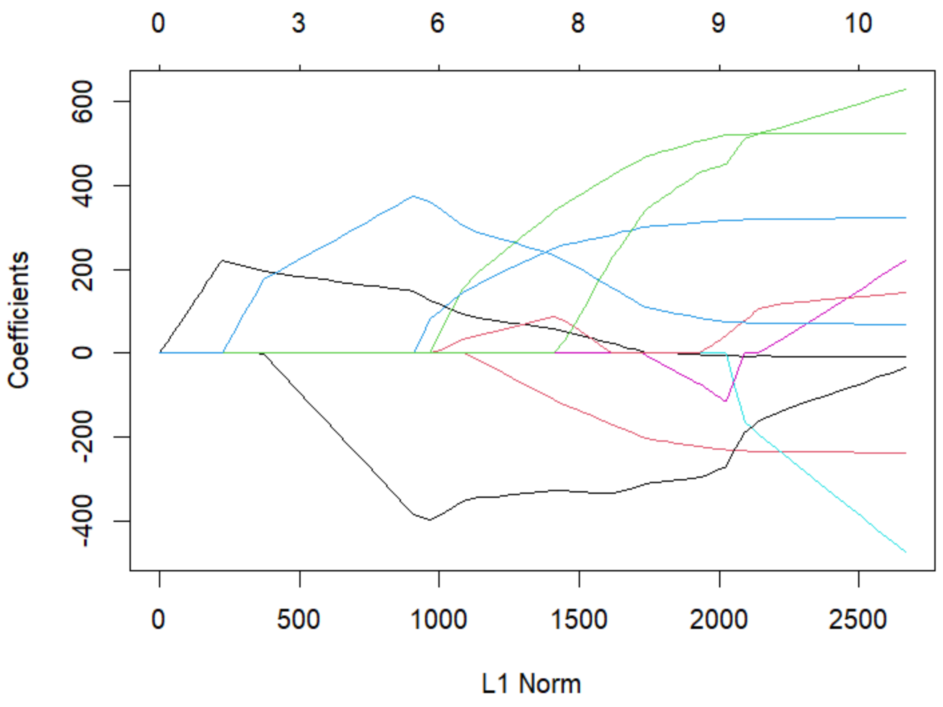


Figure 37: Plot of solution path for adaptive lasso model

Table 9: Results of Penalized Regression

$\lambda_{\text{CV}}^{avg}$	$\lambda_{\text{BIC}}^{avg}$	Prediction Error (BIC)	Prediction Error (CV)
5.69	2.61	252.28	257.72

The solution paths for Lasso and Adaptive Lasso are shown in Figure **??** and **??**. These plot represent how the coefficient becomes sparse(some of its component becomes 0) Based on cross-validation (CV) or Bayesian Information Criterion (BIC), we select the best estimator.

Here we run cv.glmnet for (CV) function and our own BIC function for a interval of λ ´s and get a minimising λ called λ_{CV} and λ_{BIC} respectively. Now since usually CV and BIC function are not convex, we usually arrive at local minimiser so we need take repeat this optimisation technique for several times and get several local minimiser and and find a prediction error minimising λ called λ_{CV} and λ_{BIC} respectively as an average model. The Table 9 consists of such λ_{CV} and λ_{BIC} and their prediction error.

10.5.3 Conclusion

In conclusion, we have successfully estimated the coefficients of the regression model using Lasso and Adaptive Lasso techniques. The selected estimator provides a reliable set of coefficients for prediction and inference.

10.6 Methods

Now as asked we repeat the Adaptive Lasso estimation procedure N times with n = 100 for both cross-validation (CV) and Bayesian Information Criterion (BIC) for model selection. We also estimate the prediction error using the formula:

$$E[(y - X\hat{\beta})^T(y - X\hat{\beta})]$$

for a simpler $\beta = (3, 0, 5)^T$

10.6.1 Results

The model corresponding to $\lambda_{\text{BIC\_new\_Beta}}$ and $\lambda_{\text{CV\_new\_Beta}}$ chosen by the Adaptive Lasso estimator for both CV and BIC criteria. Table 10 displays the prediction error for the simpler parameter vector $\beta = (3, 0, 5)^T$.

Table 10: Results of Penalized Regression with New Beta

$\lambda_{\text{BIC\_new\_Beta}}$	$\lambda_{\text{CV\_new\_Beta}}$	Prediction Error (BIC_new_Beta)	Prediction Error (CV_new_Beta)
8.7074652359494	11.0897762531817	284.125943267007	284.13071596045

10.6.2 Conclusion

In conclusion, the Adaptive Lasso estimator demonstrates its utility in model selection and prediction error estimation. Further investigation could explore its performance under different sample sizes and parameter vectors.

10.6.3 Findings

The analysis demonstrates the effectiveness of Lasso and Adaptive Lasso regression in variable selection and model regularization. Lasso regression automatically selects important predictors by shrinking coefficients, while Adaptive Lasso further penalizes less important predictors based on their OLS coefficients. The following Figures Figure 38 and 39 represents how the the CV and BIC changes against λ

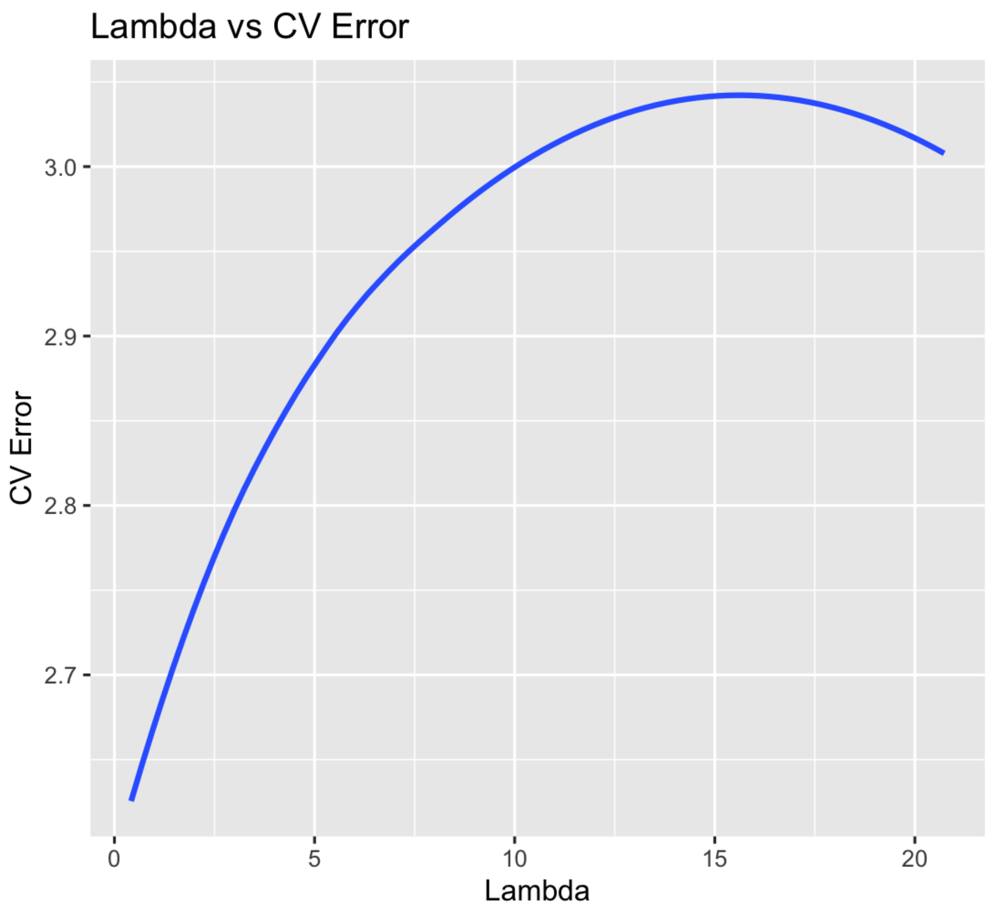


Figure 38: Plot of CV(λ) against λ

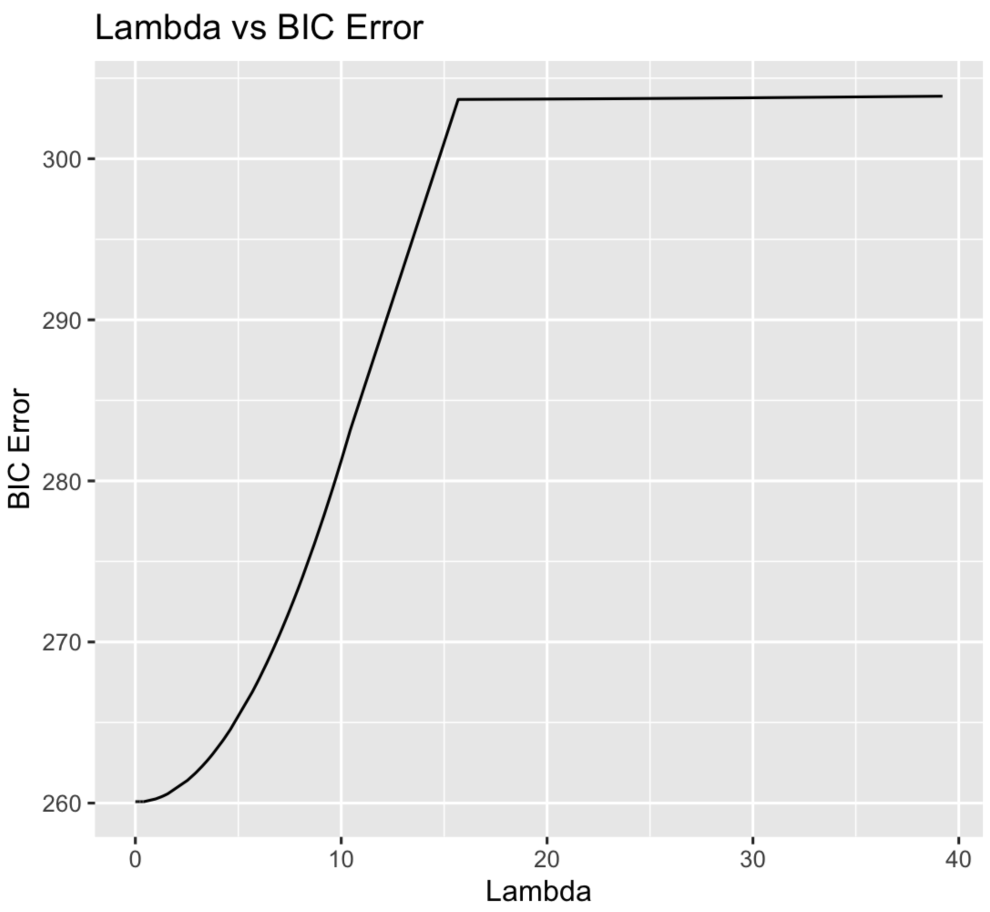


Figure 39: Plot of BIC(λ) against λ

10.6.4 Comments

Both Lasso and Adaptive Lasso regression techniques offer advantages in handling high-dimensional data and multicollinearity. Lasso regression provides a simple and interpretable model, while Adaptive Lasso enhances feature selection by incorporating prior knowledge from OLS coefficients.

10.7 Adaptive Lasso Estimator for Diabetes Dataset

In this report, we explore the use of the adaptive Lasso estimator for the diabetes dataset from the `lars` package. The adaptive Lasso is a variant of the Lasso regression that adapts the penalty weights to the data, allowing for better variable selection in high-dimensional settings.

10.8 Methodology

We fit the adaptive Lasso model to the diabetes dataset using the `lars` package in R. The adaptive Lasso imposes penalties on the regression coefficients, with the penalty weights determined by the absolute values of the estimated coefficients from an initial model.

10.9 Results

After fitting the adaptive Lasso model to the diabetes dataset, we observed the following findings:

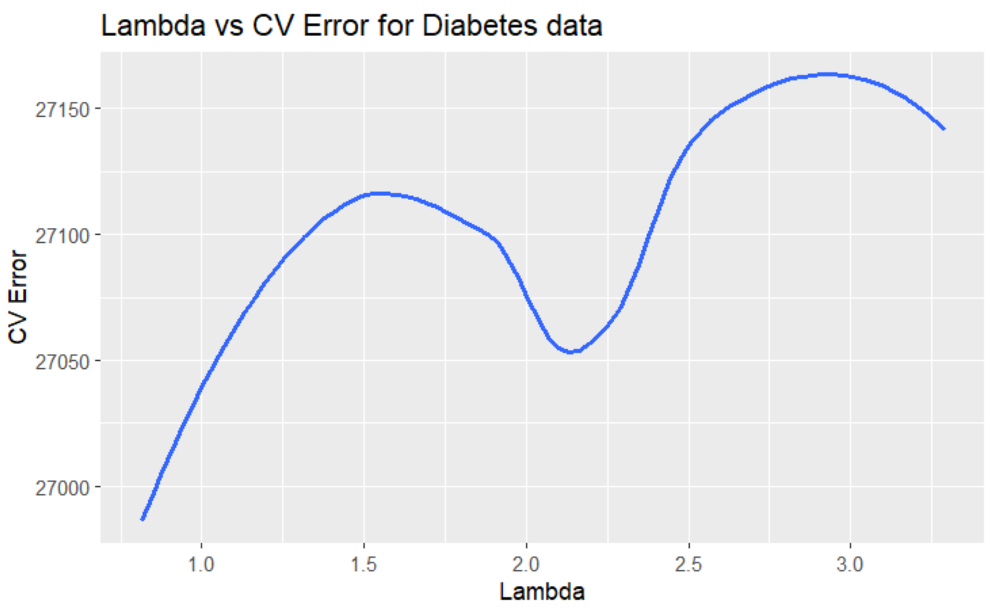


Figure 40: Plot of CV(λ) against λ

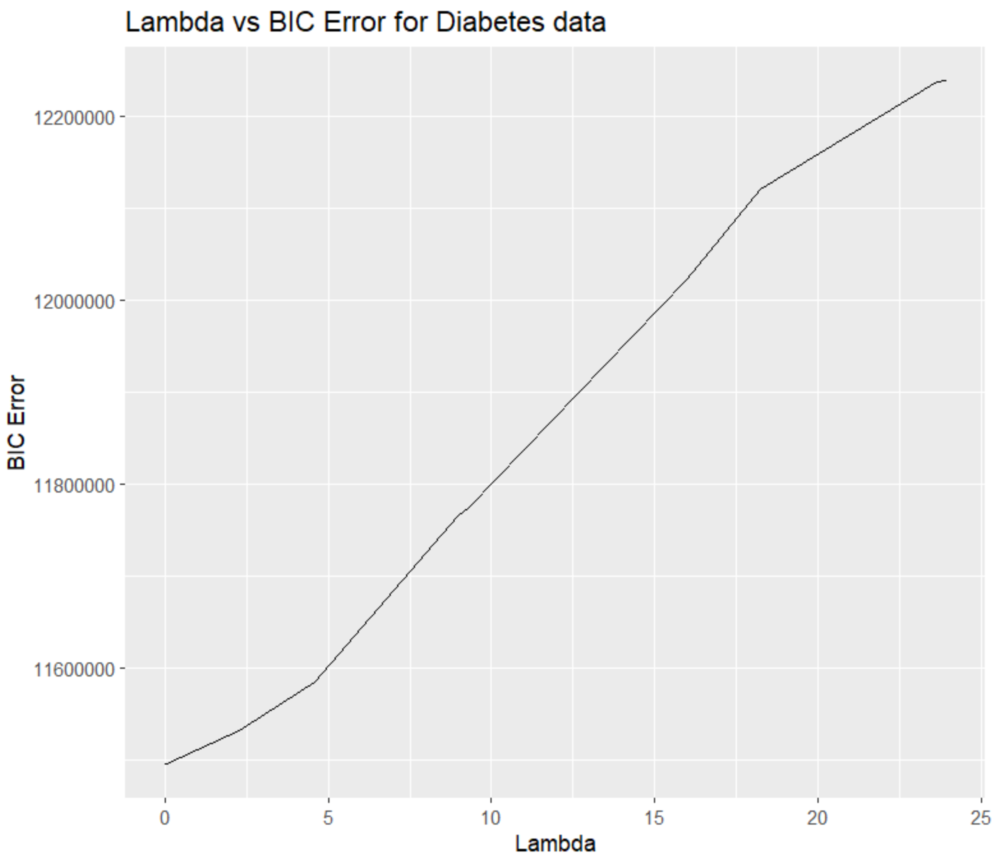


Figure 41: Plot of BIC(λ) against λ

Table 11: Results of Penalized Regression

λ_{CV}	λ_{BIC}	Prediction Error (BIC)	Prediction Error (CV)
2.27461119518531	2.75302365292618	11540382.3038135	11531792.2040814

10.10 Comments

As usual the Figures Fig 40 and 41 represents how the the CV and BIC changes against λ. It helps us finding minimizing λ. Also notice that the is not convex, so forget about global minimizer. We need to find several local minimizers and choose best among them.