National Technical University of Athens &
University of Seville

Master Thesis

Statistical techniques to identify and handle outliers in multivariate data

Christos Grentzelos

supervised by
Chrysseis Caroni
Inmaculada Barranco-Chamorro

examined by
Juan Luis Moreno-Rebollo

This work was carried out with the support of the Erasmus Programme under an Inter-Institutional Agreement of Higher Education Student and Staff Mobility between the University of Seville and the National Technical University of Athens.

June 18, 2020

Acknowledgements

I would like to express my sincere gratitude to my supervisors Professor C. Caroni of the School of Applied Mathematical and Physical Sciences at the National Technical University of Athens and Professor I. Barranco-Chamorro of the Department of Statistics and Operational Research at the University of Seville. Their contribution and guidance along with their advice regarding the analysis, were an integral part of this dissertation.

Then, I would like to acknowledge my family as well as my grandmother for their consistent support all these years.

Finally, I would like to thank all my friends that were by my side over the years and especially over the recent months.

Abstract

In this thesis, we focus on methods for detecting outliers in a multivariate setting. Outliers are also referred to as abnormalities, discordants, deviants, or anomalies in the data mining and statistics literature. It can be said that an outlier generally exhibits some abnormality or some kind of out of the way behavior. Understanding the nature of outliers gives us a better insight into the data generation process [2, 40]. Outlier detection is an integral part of the data analysis that sheds light on objects that do not conform with the rest of the data.

After a brief Introduction, in Chapter 2 we illustrate some of the various methods that were devised to deal with univariate samples. Moreover, we state the masking and swamping effect that, as we will discuss, can be difficult to handle even in univariate samples. Finally, we apply these methods to a normally distributed sample in order to demonstrate the masking effect and to compare their results.

In the following chapters, we present different methods for multivariate data based on various characteristics, which can be grouped into five big categories: Depth-based methods, Distance-based methods, Density-based methods, methods based on Mahalanobis distance and Distribution-based methods.

In Chapter 3, different notions of depth are presented and some of their corresponding detection methods. Throughout the chapter, we discuss the notion of depth originated by Tukey that the ISODEPTH and FDC algorithms were based on and the notion of Liu that led to the Modified Band Depth which we will apply to multivariate samples.

In Chapter 4, we present outlier detection methods based on the distance between objects. Moreover, we present the first notion of outliers based on their distance, the DB outliers as well as the one that is currently used based on the k nearest neighbor distance. Moreover, we present some of the basic pruning methods that distance-based methods use in order to handle bigger datasets. In addition, we present the definition of a reverse k-nearest neighbor by Hautamaki et al. Finally, we apply the method that is used to find DB outliers, two methods computing k nearest distances of the objects along with the method based on Hautamaki's definition.

In Chapter 5, density-based methods are listed, that take into account the local density of each observation. We present the Local Outlier Factor (LOF) method, that is the basis of the best-known density-based methods, along with a more robust extension of this notion known as the Robust Kernel Outlier Factor (RKOF) method. In the same chapter we apply these methods and compare their results.

In Chapter 6 we develop methods that are based on the Mahalanobis distance. The classical Mahalanobis distance is presented as well as a more robust version of it. Their main difference is that the first computes each observation's distance based on the estimators of the mean and scatter when all of the observations are taken into consideration while the second uses the the estimated mean and scatter from a specific subset of observations. Apart from the χ^2 quartile that is usually used as a cutoff for these methods, we present Filzmoser's extension, the adaptive quartile. Finally,

we test these methods and compare their results.

In Chapter 7, we present methods that detect outliers based on a distributional assumption. Most of these methods are more efficient when normally distributed datasets are under examination. Moreover, we compare the methods that we present in this chapter with the help of package “OutliersO3”.

Finally, in Chapters 8 and 9 we apply all these methods to the humus and dat datasets respectively. Testing these methods on different datasets gives us the opportunity to compare them and to build a more solid opinion about outlier detection, given in Chapter 10.

Resumen

En este TFM, nos centramos en el estudio de outliers en un marco multivariante. A los outliers también se les denomina anormalidades, discordancias, desviaciones o anomalías en Minería de Datos y otra literarura estadística. Puede decirse que un outlier exhibe alguna anormalidad o tipo de comportamiento diferente al resto. Comprender la naturaleza de los outliers nos ayuda a comprender el proceso de generación de los datos. La detección de outliers es así una parte integral del análisis de datos que nos ayuda a identificar aquellos objetos que no están en consonancia con el resto de los datos.

Tras una breve Introducción, en el Capítulo 2 ilustramos algunos de los métodos diseñados para tratar con muestras univariantes. Introducimos además los problemas de enmascaramiento y saturación que, como discutiremos, pueden ser difícil de solventar incluso en muestras univariantes. Finalmente, aplicamos estos datos a una muestra normalmente distribuida para dirigir el efecto de enmascaramiento y comparar los resultados que se obtienen.

En los siguientes capítulos, se presentan diferentes métodos basados en características multivariantes, y que pueden agruparse en cinco grandes categorías: métdos basados en la profundidad, métodos basados en distancias, métodos basados en densidad, métodos basados en la distancia de Mahalanobis, y métodos basados en la distribución.

En el Capítulo 3, se presentan diferentes definiciones de profundidad y sus correspondientes métodos de detección. A lo largo del capítulo, se estudian la noción de profundidad, debida originalmente a Tukey, y en la que se basan los algoritmos ISODEPTH y FDC, así como la noción de Liu, que condujo al concepto de profundidad de banda modificada, técnica que aplicaremos a muestras multivariantes.

En el Capítulo 4, se presentan métodos de detección de outliers basados en la distancia entre objetos. Además, se presenta la primera noción de outliers basada en su distancia, los DB outliers, así como la que se usa actualmente, basada en la distancia a los k-vecinos más cercanos. Se presentan, además, algunos de los métodos de poda básicos que utilizan los métodos basados en distancias para manejar conjuntos de datos más grandes. Se incluye también la definición de inverso de k-vecinos más cercanos debida a Hautamaki y otros. Para finalizar, se aplica el métodos para encontrar DB outliers, dos métodos para calcular las k distancias más cercanas de los objetos, junto con método basado en la definición de Hautamaki.

En el Capítulo 5, se estudian los métodos basados en la densidad, que tienen en cuenta la densidad local de cada observación. Presentamos el método Local Outlier Factor (LOF), que es la base de los métodos más conocidos basados en la densidad, junto con una extensión más robusta de esta noción, definida como método Robust Kernel Outlier Factor (RKOF). En el mismo capítulo se aplican estos métodos y se comparan sus resultados.

En el Capítulo 6 se desarrollan métodos basados en la distancia de Mahalanobis. Se trata la distancia de Mahalanobis clásica, así como una versión más robusta de ella. Su principal diferencia es que, la primera calcula la distancia de cada observación basándose en las estimaciones de la media

y la dispersión que se obtienen considerando todas las observaciones, mientras que la segunda utiliza las estimaciones de la media y dispersión basadas en un subconjunto específico de observaciones. Aparte de cuantiles χ^2, que son usualmente utilizados como punto de corte en estos métodos, se presenta la extensión de Filzmoser, denominada cuantil adaptativo. Finalmente, se aplican estos métodos y se comparan sus resultados.

En el Capítulo 7, se presentan métodos que detectan outliers basados en una hipótesis distribucional. La razón es que, la mayoría de estos métodos son más eficientes cuando los datos bajo consideración se distribuyen normalmente. Además, los métodos presentados en este capítulo se comparan con la ayuda del paquete OutliersO3.

Para finalizar, en los Capítulos 8 y 9, se aplican todos estos métodos a los conjuntos de datos humus y dat, respectivamente. Contrastar estos métodos sobre diferentes conjuntos de datos nos da la oportunidad de compararlos y construir una opinión más sólida sobre la detección de outliers, la cuál se recoge en el Capítulo 10.

Contents

1 Introduction

Over the years, many definitions have been made of outliers. Hawkins defines an outlier as "an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism".[26]. Barnett and Lewis define an outlier in a set of data as "an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data" [4]. Outliers are also referred to as abnormalities, discordants, deviants, or anomalies in the data mining and statistics literature [2].

So, an outlier generally exhibits some abnormality or some kind of out of the way behavior. More specifically, the creation of an outlier is because of unusual behaviour of the generating process underlying a given dataset. Understanding of the nature of the outliers gives us a better insight into the data generation process [40].

Outlier detection can be useful in many applications [38] such:

- **Medical Diagnosis:** In many medical applications the data is collected from a variety of devices, unusual patterns of which indicate disease conditions.

- **Credit Card Fraud:** Suppose that the number and password of one's card is compromised. This could lead to unreasonable expenses from the least likely places coming from this card i.e. an unusual signal that can be detected through the transaction data that had been collected over time by the bank.

- **Sensor Networks:** A sensor network is a communication system that monitors conditions like humidity, pressure, vibrations, intensity of sound, level of pollution etc. in different locations. Outlier detection is employed these networks in order to locate strange or false indices.

Throughout this thesis, we will present a variety of outlier detection methods that declare objects that are not part of the rest of the data generated from an underlying process (denoted as normal data, indicating that is the outlier-free subset of the data). For that purpose, we will make use of packages [54, 35, 20, 18, 59] that are implemented in the statistical environment of **R** and apply these methods on three datasets: Wood specific gravity, humus and dat in order to observe similarities and differences between them.

Some basic ideas will be introduced in Chapter 2, referring to univariate data. However, this thesis concentrates on outlier detection in multivariate data, for which a variety of different concepts and methods – presented from Chapter 3 onwards – are available [23].

2 Univariate Case

To understand the nature of an outlier and its definitions that were given above, one must be familiar with outlier detection in a univariate dataset.
Outlier detection methods can be categorized into many groups. The main two are: Tests of discordancy (formal tests) and outlier labeling methods (informal tests) [4, 51].

Most formal tests use test statistics for hypothesis testing. They are usually based on assuming some well-behaved distribution for the dataset, and for the extreme values a test is carried out on whether or not it deviates from the assumed distribution declaring it outlier or not. Some tests are for a single outlier and others for more than one. Selection of these tests mainly depends on the number of outliers, and type of data distribution [51].

Many discordancy tests have been built (Dixon test, Shapiro-Wilk etc.) mostly in the years when computer analysis was not at its peak and most of them are applicable to distributions such as Normal, Exponential and Gamma.

While formal tests are quite efficient when examining datasets modeled by known distributions (provided that tests based on these distributions are used), most distributions of real-world data may be unknown or may not follow specific distributions such as the ones we gave above. In such cases, data points may be incorrectly reported as outliers because of poor fit to the erroneous assumptions of the model. Another disadvantage is that discordancy tests are very sensitive to masking or swamping problems [51].

Masking Effect (an outlier is undetected as such): Masking can occur when a group of outlying points contaminates the dataset in such a way that the mean and standard deviation are affected by it resulting in misleading estimate. Thus, the distance of the outlying point from the mean can be small. Accordingly, an outlier can mask other outliers which can be detected only after omitting the first one.
Swamping Effect (a non-outlier is classified as an outlier): Swamping can occur when an observation can be declared as outlier only in the presence of an additional outlying object. That can be caused also from a contamination of the dataset in which the estimate of mean and standard deviation can be defined far away from uncontaminated objects [1].

Informal tests are based on a criterion formed by various estimates of scale and location of the underlying dataset and declare the observations that do not comply with this criterion as outliers. The main difference between formal and informal tests is that the latter ones do not require test statistics based on a certain distribution or a hypothesis to determine whether an extreme value is an outlier or not according to this distribution.

We will now present some of the methods that can be used for outlier detection in the univariate case.

2.1 Z-Score

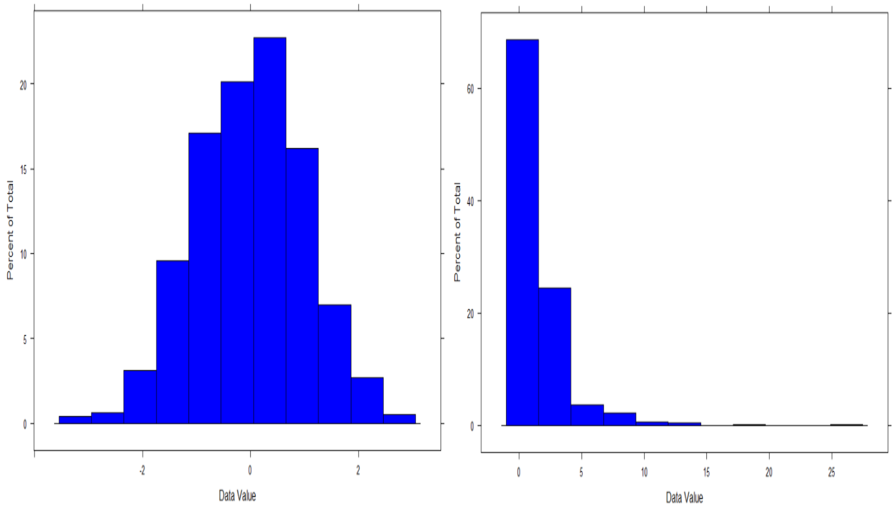


Figure 1: Histograms of Normal and LogNormal Distibution

The Z-score test for outliers is probably one of the most common tools for a rough evaluation of the discrepancies that may exist in the underlying dataset. Suppose we examine one-dimensional data observations, denoted by $x_1, x_2, x_3, ..x_n$ with mean μ and standard deviation σ.
The Z-value for the data point x_i is denoted as z_i and is defined as follows:

$$z_i = \frac{|x_i - \mu|}{\sigma}$$

The Z-value test computes the number of standard deviations by which the data varies from the mean. Assuming that the data is generated from the normal distribution and having accurately estimated the mean and standard deviation, one can have a very good picture of the outliers in the data defining them as the observations which exceed the rule: $z_i \geq 3$. We note that for efficient estimations of mean and standard deviation, datasets with few observations are not recommended.

Although this method does not need specific distribution assumptions, it may not work well in distributions with high skewness like the LogNormal Distribution (shown in Figure 1) [2].

Quite often, masking may occur when this method is applied, mainly due to the effect of outliers on mean and standard deviation estimators. An example is given next. We will refer to this dataset as contaminated normal dataset.

Application

We generate observations from the Normal Distribution with $\mu = 10$ and $\sigma = 2$ and we replace randomly the last two observations of that sample with the extreme values 18 and 20 which are obviously far enough from the mean with respect to its standard variation.

```
normal_data<-rnorm(30,10,2)
normal_data_con<-c(normal_data[-(1:2)],18,20)
sd_normal_data_con<-abs(normal_data_con-mean(normal_data_con))/sd(normal_data_con)
```

ID	x_i	z_i	ID	x_i	z_i	z_i'	z_i'
1	12.29	0.48	16	8.88	0.75	0.73	0.79
2	7.27	1.33	17	12.51	0.56	1.52	0.83
3	11.33	0.13	18	12.04	0.39	0.31	0.62
4	13.94	1.07	19	7.61	1.21	1.48	1.36
5	12.57	0.58	20	10.07	0.32	0.86	0.26
6	12.22	0.45	21	9.42	0.56	0.70	0.55
7	7.84	1.13	22	10.72	0.09	1.26	0.03
8	9.75	0.44	23	8.86	0.76	0.40	0.8
9	9.88	0.39	24	11.39	0.15	0.35	0.33
10	9.98	0.36	25	10.09	0.31	0.30	0.25
11	10.86	0.04	26	9.78	0.43	0.09	0.39
12	8.1	1.03	27	11.98	0.37	1.14	0.6
13	7.71	1.17	28	10.62	0.12	1.32	0.01
14	12.03	0.38	29	18	2.54	0.62	**3.3**
15	11.13	0.06	30	20	**3.26**	0.22	

Table 1: z-values for the contaminated normal sample before and after omitting observation 30

As we can see from Table 1, only after omitting observation 30, the z-value test (denoted as z_i') manages to declare observation 29 as outlier. That is, because observation 30 masks observation 29 and because of the sensitivity of z-score to the presence of more than one outlier, the latter observation fails to be detected.

2.2 Generalized extreme Studentized deviate test

The generalized extreme Studentized deviate (ESD) test [42] is a formal test used to detect one or more outliers in a univariate dataset $x_1, x_2, ...x_n$ that follows an approximately normal distribution and is an extension of Grubbs test [24].

Grubbs test is defined for the hypothesis:

$\mathbf{H_0}$: There are no outliers in the data set

$\mathbf{H_1}$: There is exactly one outlier in the data set

In this test, an observation x_j is declared as outlier when

$$R_j = \frac{max_i|x_i - \overline{x}|}{s} > \frac{t_{n-2,p} \cdot (n-1)}{\sqrt{(n-2+t^2_{n-2,p}) \cdot n}}$$

where $t_{\nu,p}$ is the $100 \cdot p$ percentage point from t distribution with ν degrees of freedom and $p = 1 - \frac{\alpha}{2 \cdot n}$ for a significance level α ($\overline{x}$ and s denotes the sample mean and standard deviation respectively).

With the use of the modified version of the extreme studentized deviate, one can test for up to a prespecified number r of outliers.

Given the upper bound, r (for $r = 1$ the modified method overlaps with the classical), the generalized ESD test essentially performs r separate ESD tests. The generalized ESD test is defined for the hypothesis:

$\mathbf{H_0}$: There are no outliers in the data set

$\mathbf{H_1}$: There are up to r outliers in the data set

The test-statistic for one outlier is defined as $R_1 = R_j$. In addition, we remove the observation that maximizes $|x_i - \tilde{x}|$ and we recompute the above statistic, denoted now as R_2, with $n-1$ observations. We repeat this process until r observations have been removed. This results in the r test statistics $R_1, R_2, ..., R_r$. Corresponding to the r test statistics, we compute the following r critical values

$$\lambda_i = \frac{t_{n-i-1} \cdot (n-i)}{\sqrt{(n-i-1+t^2_{n-i-1,p})(n-i+1)}}, \tag{1}$$

$i = 1, 2, ..., r$. Finally, the largest i that satisfies $R_i > \lambda_i$ determines the number of outliers as they are detected in the first i steps. Rosner showed that this approximation is very accurate when $n > 25$ [42].

Application

Applying this method to the contaminated normal dataset previously introduced and setting $r = 3$ we get the results above.

We note that we used rosner.test function from R package **EnvStats**

```
    > rosnerTest(normal_data_con,3)
$statistic
 R.1      R.2      R.3
3.257076 3.301117 2.026632
$alpha
[1] 0.05
$crit.value
lambda.1 lambda.2 lambda.3
2.908473 2.892705 2.876209
$all.stats
  i   Mean.i     SD.i    Value Obs.Num    R.i+1 lambda.i+1 Outlier
```

```
1 0 10.96163 2.774995 20.00000      30 3.257076   2.908473    TRUE
2 1 10.64996 2.226531 18.00000      29 3.301117   2.892705    TRUE
3 2 10.38746 1.751780 13.93767       4 2.026632   2.876209   FALSE
```

Given the above results, we observe that the generalized ESD test is more robust to outliers. The disadvantage is that like any formal test, the generalized ESD test assumes that the data is modeled by a particular distribution, in this case the normal distribution.

2.3 Tukey's Method (Boxplot)

Tukey's method [56], is an informal test for outliers that with the use of the median, lower quartile, upper quartile, lower extreme, and upper extreme of a data set, constructs the boxplot. Its original characteristics were the *hinges* and the *Hspread* which are similar to $Q1$, $Q3$ and IQR that are defined below. It is considered to be less sensitive to extreme values of the data than the previous methods because instead of the sample mean and the standard deviation which can be affected by outliers, it uses quartiles on the simple assumption that outliers are the minority of the data.

A classical Boxplot consists of:

- The IQR (Inter Quartile Range) that is the distance between the lower ($Q1$) and upper ($Q3$) quartiles which are the medians of the lower and upper half of the dataset respectively. A box is drawn to span IQR and the median is drawn inside this box.
- The whiskers that extend from the edges of the box to the maximum and minimum data point (adjacent values) inside the so-called **inner fences**. Inner fences are located at a distance $1.5 \times IQR$ below $Q1$ above $Q3$. Basically, the upper whisker is the minimum point that is larger than $Q1 - 1.5 \times IQR$ and the upper whisker is the maximum point that is smaller than $Q3 + 1.3 \times IQR$
- **Outer fences** are located at a distance $3 \times IQR$ below Q1 and above Q3.

Any value plotted outside the whiskers that is between the inner and outer fences is a possible outlier. There is no statistical basis for the reason that Tukey uses 1.5 and 3 regarding the IQR to make inner and outer fences [51].

Application

By applying this method to contaminated normal dataset, we get that $Q1 = 9.42$, $Q3 = 12.04$, $IQR = 2.62$, median is 10.67 (which is the mean of observations 22, 28), lower and upper whisker is 7.27 and 13.94 respectively. Finally, inner fences expresses the interval [5.49,15.97] and outer fences [1.56,19.9]. In Figure 2, boxplot detects observations 29, 30 as they lie outside the computed whiskers.

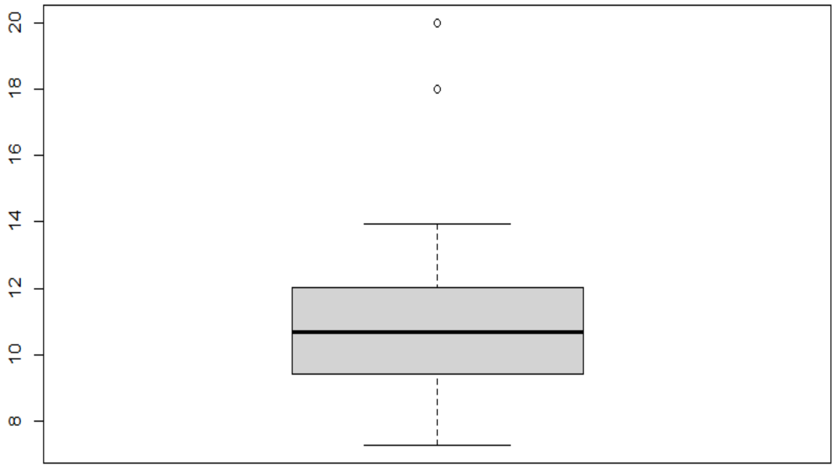


Figure 2: Boxplot of the contaminated normal sample

While the previous method is limited to unimodal and reasonably symmetric data such as the normal distribution, Tukey's method is applicable to any other data since it makes no distributional assumptions and it does not depend on a mean or standard deviation. Due to the fact that this method is based on robust measures such as lower and upper quartiles and the IQR without considering the skewness of the data, its disadvantage is that more skewness to the data may result in more wrongly defined outliers. That is why, in many cases the classical boxplot is used more as a graphical tool to detect skewness in an underlying dataset than a test for detecting outliers.

3 Depth-Based Methods

3.1 Introduction

Depth-based methods are integral to multivariate extreme value methods. Extreme value methods usually (especially when they are applied to univariate data) try to model the underlying distribution explicitly. Although this is not the case with depth-based methods, they were built to detect the same kind of outliers [2].

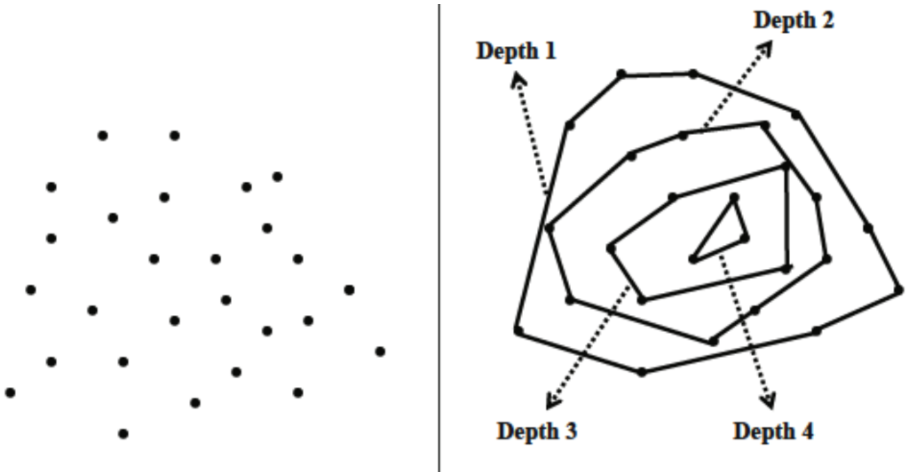


Figure 3: Depth-Based Outlier Detection [2]

3.2 Aim of depth-based methods

Johnson et al. [28] state that even though over one hundred discordancy/outlier tests have been developed [4], they are not suitable for proper outlier detection mainly for two reasons:

- Most of them are univariate so they are unsuitable for multidimensional data sets.
- All of them are distribution-based, which in many cases does not suit outlier analysis since in most data sets we have no knowledge about the exact distribution and one has to proceed with extensive testing in order to find a distribution that fits the attribute.

To avoid the aforementioned problems of distribution fitting and restriction to univariate data sets, depth-based approaches have been developed.

3.3 Analytical approach of depth-based methods

In depth-based methods, convex hull analysis is used in order to find outliers. The idea is shown in Figure 3 and Figure 4 where points in the outer boundaries of the data lie at the corners of the convex hull i.e. a polygon with vertices, the points in the shallow layers of the dataset. Accordingly, such points are considered to be extreme values and more likely to be outliers. A depth-based algorithm proceeds in an iterative fashion. In the k-th iteration, every point at the vertices of the convex hull is omitted from the initial dataset and assigned a depth of k. The algorithm keeps repeating this process until the dataset is empty. Points with depth at most r are declared as outliers [2]. This robust notion of depth, called depth contour was originally introduced by Tukey [56].

First, let us consider the univariate case. Suppose, a one-dimensional data set $P = \{p_1, p_2, p_3, ..., p_n\}$

Definition 3.1. The depth of a point z relative to the data set P, is the minimum of the number of data points to the left of z, and the number of data points to the right of z:

$$depth_1(z; P) = min(\#\{i : p_i \leq z\}, \#\{i, p_i \geq z\})$$

Remark. In the univariate case, rank [57, 56] and depth are related. Moreover, extreme values are considered to be the points of depth one as they lie far away from the rest of the dataset. In addition, second lowest and second highest values based on rank assigned with depth two, and so on. Consevely, the median is the point with the highest depth based on Definition 3.1 [49].

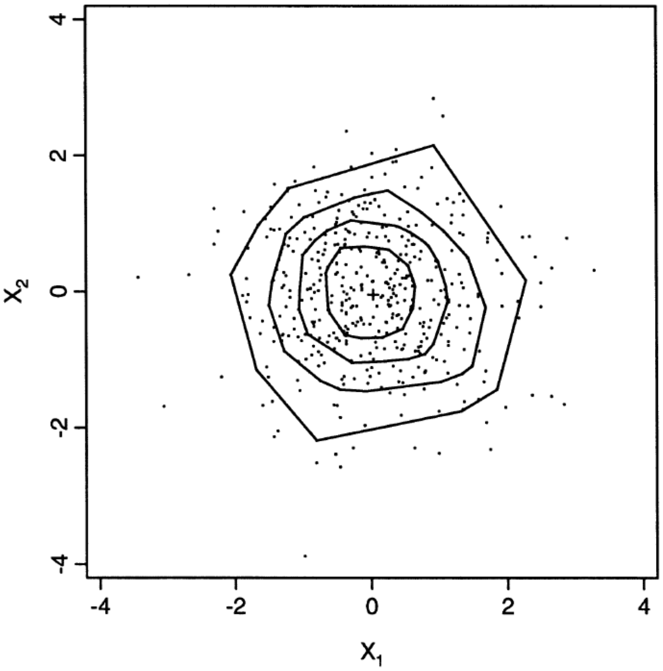


Figure 4: Example of depth contours. The deepest point in the sample is marked by a cross [32]

Considering the multivariate case we present the following definition [57, 49].

Definition 3.2. The half-space depth of a point $\mathbf{z} \in \mathbb{R}^d$ relative to a d-dimensional data cloud $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, ...\mathbf{p}_n\}$ is defined as the smallest depth of $\mathbf{z}$ in any one-dimensional projection of the data set.

Remark. The half-space depth is affine invariant because depth is invariant to linear trandormations of the data $\mathbf{p}$ [13, 14] making it independent of the underlying coordinate system.

Given these two definitions we define a set P_k as a contour of depth k of a data set $\mathbf{P} \subset \mathbb{R}^d$ if its interior points have depth at least k, and the boundary points have depth equal to k:

$$P_k = \{\mathbf{p} \in \mathbb{R}^d | depth(\mathbf{p}, \mathbf{P}) \geq k\}$$

Remark. $\mathbf{P}_k$ is the intersection of all half-spaces that contain at least $n + 1 - k$ points of the cloud, hence it is convex [49].

The maximal depth depends on the shape of $\mathbf{P}$ meaning that the the computational complexity of depth-based methods that map the underlying dataset into convex hulls increases exponentially with dimensionality. That is because, a convex-hull in a d-dimensional space consists of at least 2^d points so by moving to higher dimension the number of points lying at the vertices of the convex-hull is also increased [2]. As a result these heuristic and non-probabilistic methods, due to their high computational cost can only be trustworthy in low dimensional data sets.

3.4 Proposed depth-based methods

As we noted above, these methods do not scale up well because of their computational complexity. P_k is the intersection of a collection of half-planes, of which it can be assumed that their boundary line passes through two data points. Each edge of P_k is thus part of a line through two data points, and hence each vertex of the desired contour lies on the intersection of two such lines. So, naively thinking, we can consider all lines through two data points, and compute all their intersection points, of which there are $O(n^4)$. At each of these intersection points we then compute the half-space depth by means of an algorithm of Rousseeuw and Ruts [47]. The intersection points with depth equal to k are then stored, denoting their number by N_k. The depth contour P_k is the convex-hull consisting of these N_k points. The computation of the depth in all of the $O(n^4)$ points brings the total time complexity to $O(n^5 logn)$ [49].

Many algorithms have been proposed in order to reduce the high computational cost.

Ruts and Rousseeuw develop an algorithm called ISODEPTH, which computes 2-D depth contours [43]. The key idea of their algorithm is based on **dividers** and for $n < 1000$ the time of the proposed algorithm behaves as a multiple of $O(n^2 logn)$. Although the ISODEPTH algorithm managed to reduce the computational cost, it still remains high. Furthermore, ISODEPTH relies on the non-existence of collinear points and removing all of them can be very time consuming.

Johnson et al [28] propose the FDC algorithm which is an extension of ISODEPTH. FDC performs better than ISODEPTH when n is not too small and is robust against collinear points. The main difference between the two lies in the fact that ISODEPTH computes the dividers of all n data points while FDC restricts the computation to a selected, much smaller, subset of points, thanks to the construction of the appropriate convex hulls.

Definition 3.3. Given a point cloud P consisting of n points, a line L is an e-divider of P if there are e points in P to the left of L, and $(n-e)$ points in P to the right of L.

Remark. In the spirit of finding outliers, whenever $e \leq (n-e)$, we say that the e points are to the "outside" of L and the remaining points on the "inside" of L.

Definition 3.4. (a) We call the "inside region" and the "outside region", denoted as $IR(L)$ and $OR(L)$ respectively, the two subregions into which the line L divides the convex hull of P.

(b) Given a collection of e-dividers, we refer to the intersection of all their inside regions (i.e., $\cap IR(L)$) the e-intersected inside region.

The only interesting part of an e-divider is a finite segment of it. The line segments of the e-dividers construct the inside region of a given convex hull. Suppose that the convex hull (denoted as G) of the entire point cloud of a given data set P is a polygon with an arbitrary number of vertices.

FDC algorithm:

- Returns this polygon as the zero-th depth contour.
- Computes the convex hull (denoted as H; suppose a polygon) of the remaining points in the data cloud (i.e., all points except the ones on the outer boundaries)
- Computes all the one-dividers in the initial convex hull in the first iteration and
  - If the convex hull that is constructed from the line segment of a one-divider and the rest of the edges of the initial polygon G contains the polygon H, then the intersection of these convex hulls gives the contour of depth equal to 1.
  - Otherwise, the algorithm expands that convex hull in order for the polygon H to be inside it.
- The algorithm moves to the next iteration (computing the two-dividers) until it finds all k first depth contours.

3.5 Modified Band Depth (MBD)

The modified band depth measure introduced by Lopez-Pintado [33] also assigns a value indicating how "deep" an observation is inside the sample but in a different notion than the one discussed above. It is an extension of Liu's simplicial depth function [31] which measures each observation's depth by the number of convex hulls (made of a given number of points i.e. vertices) that contain it.

In order to visualize the idea, suppose a dataset $P = (\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n)$, where $\mathbf{p}_i \in \mathbb{R}^2$, for $i = 1, ..., n$. Now, consider the triangle generated by three arbitrary points p_i, p_j, p_k. The idea of simplicial depth function lies in the observation that a point's depth depends on how many triangles of the $\binom{n}{3}$ possible combinations this point falls inside. Normally, points of high depth tend to be inside a large proportion of triangles while points in the boundaries are not likely to fall in many triangles. Therefore, high values of the simplicial function for a point indicate that it is deep in the data cloud, while low values indicate the opposite.

In that context modified band depth is defined as follows

Definition 3.5. Let $P = (\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n)$, where $\mathbf{p}_i \in \mathbb{R}^d$ a d-dimensional point. The modified band depth of a point $\mathbf{p}$ is

$$MBD(\mathbf{p}) = \frac{1}{d} \sum_{k=1}^{d} \binom{n}{2}^{-1} \sum_{1 \leq i_1 < i_2 \leq n} \mathcal{I}\{min\{\mathbf{p}_{i_1}(k), \mathbf{p}_{i_2}(k)\} \leq \mathbf{p}(k) \leq max\{\mathbf{p}_{i_1}(k), \mathbf{p}_{i_2}(k)\}\}$$

where $\mathcal{I}(\cdot)$ is the indicator function and $\mathbf{p}(k)$ is the k-th coordinate of p.

Based on Definition 3.5. the depth of an object $\mathbf{p}$ can be interpreted as the mean, over all pairs of observations, of the proportions of coordinates whose values are between the minimum and maximum of two values of points from the dataset.

MBD can be computed over all combinations of objects higher than two but it is shown that choosing to test each object's depth with respect to each pair of observations, we can have efficient results and keep the computational complexity low at the same time. Moreover, MBD computational cost is shown to be $O(n^2 \cdot d)$ which is much smaller than the aforementioned methods [33].

3.6 Applying outlier detection using depth-based method to wood specific gravity data

To motivate the Least Median Squares method, Rousseeuw [43] contaminated a rather well behaved real data set by replacing four observations. The raw data came from Draper and Smith [15] and were used to determine the influence of anatomical factors on wood specific gravity, with five explanatory variables and an intercept. Observations 4, 6, 8, and 19 are identified using LMS. These observations do not appear to be obvious outliers from the Least Squares analysis [12].

More specifically, we go on applying outlier detection using the depth based method based on the modified band depth measure implemented in the R-package "OutlierDetection". We note that every function in this package also provides for bivariate data scatterplots showing the outlying observations.

The code is as follows:

```
 > depthout(my_data, rnames = FALSE, cutoff = 0.25, boottimes = 100)
```

The arguments of the above command are:

- The Wood Gravity data
- A logical value indicating whether the data set has rownames (default value is False)
- A percentile threshold used for depth; basically after computing the MBD of all the observations it declares that percentage of observations with the lowest MBD values (default is 0.05).
- A number of bootstrap samples to find the cutoff (default is 100 samples) based on which, the outlyingness of each object is computed in a sense of how frequent this observation was detected as outlier.

Using this command we get the following results

```
$`Outlier Observations`
      x1     x2    x3    x4    x5     y
4  0.437 0.1591 0.446 0.423 0.992 0.450
6  0.444 0.1628 0.429 0.411 0.984 0.431
8  0.413 0.1673 0.418 0.430 0.978 0.423
19 0.417 0.1687 0.405 0.415 0.981 0.401

$`Location of Outlier`
[1]  4  6  8 19
```

```
$`Outlier Probability`
[1] 0.82 0.97 1.00 1.00
```

We also apply the original function implemented in the R-package ”depthTools”. The code is below

```
    > MBD(my_data)
$ordering
 [1] 14 15  5 16  2 13 18  3 17  1  9 11  7 12 10 20  4  6  8 19

$MBD
    [,1]      [,2]       [,3]       [,4]       [,5]       [,6]       [,7]
 0.4333333 0.477193 0.4491228 0.2964912 0.4982456 0.2596491 0.4070175
  [,8]      [,9]      [,10]      [,11]      [,12]       [,13]       [,14]
0.2315789 0.4333333 0.3859649 0.4122807  0.4035088 0.4631579  0.5017544
  [,15]   [,16]      [,17]      [,18]      [,19]      [,20]
 0.5026316 0.4780702 0.4403509 0.4561404 0.1614035 0.3561404
```

As we see from the results, the aforementioned command correctly detects the outliers that intentionally were included. Moreover the results of the two functions overlap completely as they both have the same results. The advantage of the second functions is that we can have an ordering of the observations i.e. a further insight about which point based on that method is further away from the others. The first one, does not give any order even though it prints the same results but it is able to return a proportion of times an object is detected as outlier after bootsrtaping. Applying the command for different cutoffs we observe that **depthout**'s sensitivity increases with higher cutoff values. Furthermore, the highest “outlyingness” of these observations are obtained by giving the value 0.24 to the cutoff. The same results but with lower “outlyingness”, we obtain by giving values for the cutoff from the space [0.23,0,27].

3.7 Conclusions

Depth-based methods are different from the other extreme value methods. They are able to detect outliers from the outer boundaries of a data space without assuming any particular statistical distribution of the underlying data nor requiring the existence of a metric distance function (as distance-based methods do). Nevertheless, except for MBD they do not scale up well [2] in higher dimensions and based on their structure they are unable to detect isolated points in the inside regions of the convex-hull as they are suited for extreme value detection. In the underlying data, **deapthout** was able to detect the true outliers of the dataset letting us assume that the nature of outlierness of these observations is rather extreme than local.

4 Distance-Based Methods

4.1 Introduction

Distance-based methods are a popular class of outlier-detection algorithms across a wide variety of data domains, and define outlier scores on the basis of nearest neighbor distances. Distance-based outlier analysis methods work with the assumption that the k-nearest neighbor distances of outlier data points are much larger than normal data points. [2]

Knorr and Ng [29] propose the following distance-based definition for outliers that is consistent with Hawkin's definition and generalizes the one that Barnett and Lewis proposed :

Definition 4.1. An object O in a data set T is a $DB(p, D)$ outlier (DB stands for Distance Based), if at least fraction p of the objects in T lies greater than distance D from O.

In simple terms, in a data set T we define an object O as an outlier if no more than a certain number of objects (denoted as k) are at a distance less than or equal to D from O. Defining that fraction as $\frac{N-k}{N}$ (where N is the number of data points in T) and giving a rather small value for k, this fraction has to be close to unity in order to give proper results. Most distance-based algorithms therefore work with the parameter k, because it is simpler, and more intuitive to understand.

Based on the notion of the above definition, many methods have been proposed for outlier detection, some of which will be discussed in this section. Moreover, we present Ramaswamy's idea for dealing with high dimensional data.

4.2 Aim of the distance-based methods

Like depth-based methods, distance-based methods do not need any statistical assumption about the underlying data. Therefore, there are suitable for situations where the observed distribution does not fit any standard distribution.

More importantly, they are well defined for d-dimensional data sets for any value of d. Unlike the depth-based methods (except for MDB), DB-outliers are not restricted computationally to small values of d. While depth-based methods rely on the computation of layers in the data space, outliers detected by distance-based methods go beyond the data space and rely on the computation of distance values based on a metric distance function (Manhattan, Euclidean etc.). Computing all these distances for all of the objects in their k nearest neighborhoods can be computationally expensive. In the next chapter we will scratch the surface of how to deal with high dimensional data by a brief presentation of Ramaswamy's method.

4.3 Analytical approach of the distance-based methods

Ramaswamy et al. [39] proposed a slightly different definition for an outlier based on the fact that the main interest of a user is the identification of the top n outliers.

Definition 4.2. Given an input data set with N points, parameters n and k and denoting as $D^k(x)$ the distance of point x from its k-th nearest neighbor, a point x is a D^k_n outlier, if there are no more than $n-1$ other points x' such that $D_k(x') > D_k(x)$.

In other words, if we rank points according to the $D^k(x)$ distance, the top n points in this ranking are considered to be outliers. Based on this definition, the user is no longer required to specify the distance D (as in the previous one) to define the neighborhood of a point. Instead, the only thing that has to be specified is the number of outliers of interest.

The simplest approach to the problem uses a nested loop approach. In the nested loop approach, two arrays are maintained– the first array contains the candidates for outlier data points, and the other array contains the points to which these candidates are compared in distance based processing. Once more than k data points have been identified to lie within a distance of D^k from a point in the first array, that point is automatically marked as a non-outlier. Subsequently, no more time is spent on distance computations involving that data point. Such an approach may require $O(N^2)$ distance computations in the worst case. Since each distance computation may require $O(p)$ time, it follows that the overall running time is $O(N^2 \cdot p)$. Therefore, pruning methods are required in order to speed up the distance computations[2, 39]

The approximation of a set of points using their minimum bounding rectangle (MBR), is a key tool for these pruning methods. Upper and lower bounds on $D^k(x)$ for points in each MBR are then computed. These bounds provide useful information enabling us to prune entire MBRs that cannot possibly contain outliers.

Most researchers use as distance metric the Euclidean distance (Ramaswamy et al. used the squared Euclidean distance). Let us denote a point x in a d-dimensional space by $x = (x_1, x_2, ..., x_d)$ and a d-dimensional rectangle R by the two endpoints of its major diagonal, $r = (r_1, r_2, ..., r_d)$ and $r' = (r'_1, r'_2, ..., r'_d)$ such that $r_i < r'_i$ for $i = 1, 2, ..., d$. We now give the following definitions that describe the minimum and maximum distances between a point and a rectangle and between two rectangles.

Definition 4.3. The minimum distance between a point x and a rectangle R (denoted by MINDIST(x,R)) [48] is defined as MINDIST(x,R)=$\sum_{i=1}^{d} \delta_i^2$, where δ_i is defined as

$$\delta_i = \begin{cases} r_i - x_i & \text{, if } x_i < r_i \\ x_i - r'_i & \text{, if } r'_i < x_i \\ 0 & \text{, otherwise} \end{cases}$$

Definition 4.4. The maximum distance between a point x ad a rectangle R (denoted by MAXDIST(x,R)) is defined as MAXDIST(x,R)=$\sum_{i=1}^{d} \delta_i^2$, where δ_i is defined as

$$\delta_i = \begin{cases} r'_i - x_i & \text{, if } x_i < \frac{r_i + r'_i}{2} \\ x_i - r_i & \text{, otherwise} \end{cases}$$

Definition 4.5. The minimum distance between two rectangles R, S (denoted by MINDIST(R,S)) is defined as MINDIST(R,S)=$\sum_{i=1}^{p} \delta_i^2$, where δ_i is defined as

$$\delta_i = \begin{cases} r_i - s'_i & \text{, if } s'_i < r_i \\ s_i - r'_i & \text{, if } r'_i < s_i \\ 0 & \text{, otherwise} \end{cases}$$

Definition 4.6. The maximum distance between two rectangles R,S (denoted by MAXDIST(R, S)) is defined as MAXDIST(R, S)=$\sum_{i=1}^{p} \delta_i^2$, where $\delta_i = max\{|s'_i - r_i|, |r'_i - s_i|\}$.

Every point in R is at distance of at least MINDIST(x, R) from x and no point in R is at a distance that exceeds MAXDIST(x, R) from x. Similarly every point in R is at a distance of at least MINDIST(R, S) from every point in S (S is denoted similarly by the two endpoints s and s') and no point in R is at a distance that exceeds MAXDIST(R, S) from every point in S (and vice versa) [39].

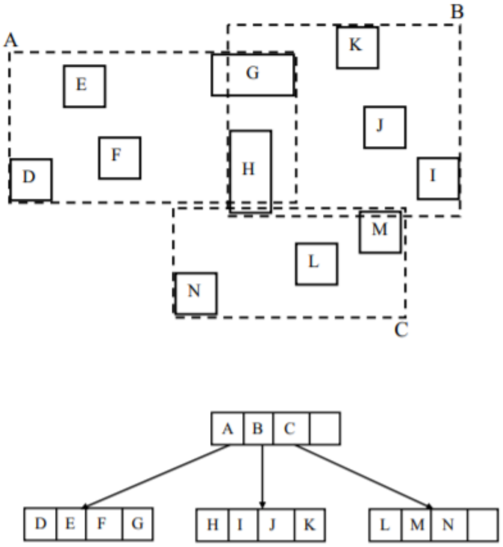


Figure 5: Example of minimum bounded rectangles and their corresponding R-tree: A set of the MBRs of some data geometric objects (not shown). These MBRs are D, E, F, G, H, I, J, K, L, M, and N, which will be stored at the leaf level of the R-tree. The same figure demonstrates the three MBRs (A, B, and C) that organize the aforementioned rectangles into an internal node of the R-tree.[36]

4.3.1 Index-based approach

The aforementioned bounds can be used in conjunction with index structures such as the R-tree [6] for estimating the k-nearest neighbor distance of data points. Of course any other spatial index structure can be used. R-tree (an example is given in Figure 5) is preferred because it is an index structure in which nearby objects are grouped and represented in rectangles in a way that no intersections between the rectangles and their content objects can be made. So an R-tree uses minimum bounding rectangles in order to represent the data at the nodes. In order to determine the outliers in the data set, the points are processed one by one in order to determine their k-nearest neighbor distances. The highest n such distances are maintained dynamically over the course of the algorithm. For the efficient estimation of the $D^k(x)$ a pruning method can be applied: Looking in a arbitrary subset of the input objects $D^k(x)$ for x is computed . If the aforementioned value is lower than the minimum distance of a MBR then none of the points in that rectangle can be defined as k-neighbors of x. Such subtrees of the R-tree can be completely pruned from consideration.

In addition, since we are interested in the top n outliers further pruning can be made in the computation of $D^k(p)$. In each step of this index-based algorithm the top n outliers can be sorted and the minimum distance among these outliers can be computed. If during the computation of $D^k(x)$ of a point x we find that its value is lower than the minimum distance of the top n outliers then point x cannot possibly be an outlier.[2, 39, 29].

Partition-Based speed-up: Since we are interested in detecting the top n outliers and typically the value of n is small, unnecessary computations of distances D^k of points cannot be avoided in spite of the aforementioned pruning methods.
The key idea to this pruning method (it is more of a preprocessing step) is to cluster the data into partitions and compute the lower and upper bounds on D^k for points in each partition. If the upper bound of this partition is lower than the minimum distance of the top n outliers that are provided in each step, then the whole partition can be avoided in further computations. An approximation of this minimum distance can be obtained by sorting a certain number of lower bounds of partitions that consist of at least n objects[2, 29, 39]

An extension to Ramaswamy's work is the proposed method of Hautamaki et al. [25]. As we discussed above, the algorithm needs a specific number of outliers as an input and that in most cases is unknown. For this extension, two different variants are considered: mean of k-nearest neighbor (kNN) distances (MeanDIST) and maximum of kNN distances (KDIST). Vectors with large average kNN distance are all marked as outliers. When scanning the ordered list from smaller to larger distances, we check if the difference between adjacent distances is larger than a given threshold. We then define vectors beyond the cut point as outliers. We define the threshold as:

$$T = max(L_i - L_{i-1}) \cdot t,$$

where L_i is the KDIST or the MeanDIST of the i^{th} vector and $t \in (0, 1)$ is an input parameter.

4.3.2 Reverse Nearest Neighbor approach

Most of the distance-based methods directly use the k-nearest neighbor distribution in order to define outliers. A different approach is to use the number of reverse k-nearest neighbors in order to define outliers [25]. Therefore, the concept of a reverse k-nearest neighbor is first defined.

Definition 4.7. A data point x is a reverse k-nearest neighbor of y, if and only if y is a k-nearest neighbor of x [2].

Data points which have large k-nearest neighbor distances, will also have few reverse neighbors, because the latter ones tend to have neighbors with smaller distances. Thus, an outlier is defined as a point for which the number of reverse k-nearest neighbors is less than a user-defined threshold [2].

The reverse nearest neighbor approach can also be easily understood in terms of the underlying k-nearest neighbor (kNN) graph.

We define the k-nearest neighbor graph as a weighted directed graph, in which every vertex represents a single vector, and the edges correspond to pointers to neighbor vectors. Every vertex has exactly k edges to the k nearest vectors according to a given distance function. The weight of the edge e_{ij} is the distance between vectors x_i and y_i. The graph can be constructed by exhaustive search considering all pairwise distances at the cost of $O(N^2)$ time.

Moreover, Hautamaki et al. [25] proposed a method in which an object (denoted by a vertex) is defined as an outlier on the basis of its indegree number in the kNN graph. For a vertex, the number of head ends (by the phrase head ends we mean the edges that point the underlying vertex; for example a k nearest neighborhood of an object in a $k$nn graph is shown by the edges that start from the target object and end to its $k$nn neighbor objects pointing their corresponding verices) adjacent to a vertex is called the indegree of the vertex and the number of tail ends adjacent to a vertex is its outdegree. In our case, the indegree number of an object can be interpret as the number of reverse k-nearest neighbors, due to the observation that the k nearest neighbor of a point is not a symmetric definition. The fact that a point is a k nearest neighbor of another point does not imply the opposite. That observation of Hautamaki creates a connection between the distance based methods and the density based methods that we will present in the next chapter. According to the above, an outlier can be defined as follows:

Definition 4.8. Given kNN graph G for data set S, an outlier is a vertex, whose indegree is less than or equal to T [25].

In other words, the proposed algorithm, after creating the kNN graph checks each vertex to see if its indegree number is lower than a given threshold T and if it is, it marks it as an outlier.

4.4 Applying outlier detection using distance-based methods to wood specific gravity data

In this section we will test four functions on Wood specific gravity data (we recall that the Wood specific gravity data is a data set of twenty observations on six variables, four of which (4, 6, 8, 19) are known to be outliers). The first one is based on the Definition 4.1 which is the original seed of these methods. The following two measure $k$nn distances and based on a cutoff declare the exceeding objects as outliers. Finally, the last function is based on Hautamaki's method and the indegree number of objects discussed above.

- **DB**: Calculates how many observations are within each object's neighborhood, which is formed by a specific distance given by the user. The objects that have proportion of neighbors lower than a user's fraction are declared as outliers.
- **nn**: Computes the average k-nearest neighbor distance of observation and based on the bootstrapped cutoff, labels an observation as outlier.
- **nnk**: Computes the k-th nearest neighbor distance of an observation and based on the bootstrapped cutoff, labels an observation as outlier.
- **KNN_IN**: Calculates the indegree number of each observation given a k-nearest neighbors graph.

We begin with the first function implemented in R package "DDoutlier" testing it with its default settings. The code is below

```
    > DB(my_data, d=1, fraction = 0.05)
$neighbors
 [1] 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19

$classification
 [1] "Inlier" "Inlier" "Inlier" "Inlier" "Inlier" "Inlier" "Inlier" "Inlier" "Inlier" "Inlier
[13] "Inlier" "Inlier" "Inlier" "Inlier" "Inlier" "Inlier" "Inlier" "Inlier"
```

As we see from the results, the function tuned in these settings is unable to find any outlier. That is because the arguments depend on the size of the dataset and on the distance between the objects. In this case, it seems that the given distance is too high, and since this method does not depend on a specific number of neighbors, unifies all objects in one neighborhood. In Table 2 we present some results of **DB** function. Input value d decides the number of objects within each observation and input value fraction represents a cutoff based on which we declare the outliers. Even though, the function eventually manages to find the correct outliers in the dataset, one might think that this

DB function		
d	fraction	Outliers
0.1	0.1	7 9 10 12 13 18 20
	0.2	1 2 4 5 6 7 8 9 10 11 12 13 16 17 18 19 20
	0.3	1 2 3 4 5 6 7 8 9 10 11 12 13 16 17 18 19 20
0.2	0.1	0
	0.2	19
	0.3	4 6 8 19

Table 2: Number of detected outliers for different *d* and franctions for **DB** function

way is more complicating than helpful. That is because, the information of an efficient distance *d* is strictly linked to the structure of the underlying data which is most of the times unknown.

On the contrary, functions **nn** and **nnk** (implemented in R package "OutlierDetection") search for outliers based on the distances of the objects within their neighborhoods which are equally sized determined by a specific *k* given by the user (default value for *k* is the number of the observations times 0.05). Moreover, the number of outliers detected in the dataset depends on a percentile cutoff value given by the user (default is 0.95). These algorithms compute all *k*nn distances and flag all observations whose distances exceed the latter threshold. Even though *k*nn distances are not given, in order to have an outlier score between the flagged outliers an outlier probability based on bootstraping is employed (like in the **depthout** function). The difference between the two functions is that the first one compute the average of the *k*nn distances of each observation while the second one computes the *k*-th nearest distance of each observation.

We begin testing the **nn** function and for that purpose we set the arguments according to the (small) size of the dataset. The code of **nn** and the obtained results are given below.

```
    nn(my_data, k = 5, cutoff = 0.75, boottimes = 100)
```

```
$`Outlier Observations`
      x1     x2    x3    x4    x5     y
7  0.489 0.1231 0.562 0.455 0.824 0.481
9  0.536 0.1182 0.592 0.464 0.854 0.475
10 0.685 0.1564 0.631 0.564 0.914 0.486
12 0.703 0.1335 0.519 0.484 0.812 0.519

$`Location of Outlier`
[1]  7  9 10 12
```

```
$`Outlier Probability`
[1] 0.96 0.82 0.99 0.97
```

The code of the **nnk** function as well as the obtained results are as follows

```
   nnk(my_data, k = 5, cutoff = 0.75, boottimes = 100)
$`Outlier Observations`
      x1     x2    x3    x4    x5     y
6  0.444 0.1628 0.429 0.411 0.984 0.431
8  0.413 0.1673 0.418 0.430 0.978 0.423
19 0.417 0.1687 0.405 0.415 0.981 0.401

$`Location of Outlier`
[1]  6  8 19

$`Outlier Probability`
[1] 0.96 0.96 1.00
```

As we can see from the above results and Table 3, **nnk** is able to find the true outliers of the dataset quicker than **nn**. That is because of the small size of the dataset. In small datasets the size of the k nearest neighborhoods is accordingly small. So the average of the $k$nn distances can not be representative. For $k = 10$ both functions show the same results. Consequently, these functions in larger datasets can provide in a high percentage the same outliers.

Outliers		
nn	**nnk**	k
7 10 12	4 6 8 19	6
7 10 12 19	4 6 8 19	7
7 8 10 19	4 6 8 19	8
6 8 10 19	4 6 8 19	9
4 6 8 19	4 6 8 19	10

Table 3: Number of detected outliers for different k for **nn** and **nnk** functions

Finally we present Hautamaki's method (implemented in R package DDoutlier). It works quite differently from the other methods as it has as outlier score the indegree number of each observation. Basically, it computes for a given k the number of times that an observation is a k nearest neighbor of the other observations.

The code for **KNN_IN** function for $k = 5$ and the obtained results are shown below

```
> KNN_IN(my_data,k=5)
 [1]  8  7  7  3 10  3  5  3  2  1  5  3  4 10  9  4  4  7  3  2
> order(KNN_IN(my_data,k=5))
 [1] 10  9 20  4  6  8 12 19 13 16 17  7 11  2  3 18  1 15  5 14
```

Based on the obtained results and Table 4, we observe a decrease in the number of observations that satisfy $T \leq 3$. Furthermore, we notice convergence (slower that the one we encountered in **nnk** function) to the true outliers as we increase the value of k.

KNN_IN	
k	outliers
4	10 12 20 7 9 4 6 8 17 19
6	10 9 4 6 8 19
7	10 9 4 6 8 19
8	4 6 8 19
9	4 6 8 19

Table 4: Detected outliers for different k based on their indegree number. Observations with $T \leq 3$ are presented

4.5 Conclusions

Distance-based methods are able to detect outliers in data sets in which the underlying distribution is unknown. Their main idea is that higher $k$nn distances tend to come from more isolated points of the dataset. Furthermore, these methods scale up better than the depth-based methods, because they are not so much restricted to lower dimensions due to a variety of data structures and pruning methods like the ones we discussed above. Applying some of these methods to a dataset known for its contamination, we note that **DB** while successful, needs more information about the data structure than the other three. Furthermore, it has no outlier score, thus giving no clue to the user of how much of an outlier an observation is. Consequently, **DB**, can be unsuitable for high dimensional raw datasets. On the other hand, **nn**, **nnk** and **KNN_IN** are much easier to handle as they search for outliers in equally sized neighborhoods. However, the choice of k in order to have efficient results is not clear. In the previous analysis we noticed a convergence to the true outliers increasing k. However in higher dimensional datasets in which the number of outliers is unknown and and the observations canot be represented in a graph the choice of k can be tricky as we will see in Chapter 8.

5 Density-Based Methods

5.1 Introduction

Density-based approaches for outlier detection have been developed to deal with the local density problems of the distance-based methods. In many applications there can be different regions and varying characteristics in a dataset. Therefore it is more meaningful to search for outlying objects based on other objects in each of their neighborhoods.

In a typical density-based framework, there are two parameters that define the notion of density of an object.

- A parameter **MinPts** specifying the minimum number of objects, and
- A parameter specifying the volume under consideration.

These two parameters determine a density threshold for the algorithm to operate. To detect density-based outliers, it is necessary to compare the densities of different sets of objects in the data.[40, 27]

In this section we will discuss the Local Outlying Factor (LOF) based on which many algorithms were built and its properties. Next, we will describe an extension of LOF called the Robust Kernel-based Outlier Factor (RKOF).

5.2 Aim of density-based methods

In the previous chapter, we presented the distance-based methods. These methods define as outliers the objects that are a threshold distance away from a partition of the data set. They detect outliers in a more global sense making them unable to find all kinds of outliers as they were proved to be sensitive to data locality (except for the reverse-nearest neighbor approach that can handle local data variations well).

As we can see in Figure 6, while both points p_1 and p_2 are outliers, a classical distance-based method would probably detect only p_2 because it is far away from both clusters C_1 and C_2. On the other hand, point p_1 is too close to them and so it is considered as normal. Moreover, if we choose to have a stricter threshold for the distance in order to reveal p_1, the method will probably denote more objects as outliers than the real ones.

In addition, all the aforementioned methods detect outliers in a binary concept declaring them outliers or not. Density-based methods proceed to a quantification of how much of an outlier an object is giving in that way more information about the outlying objects [40, 8]

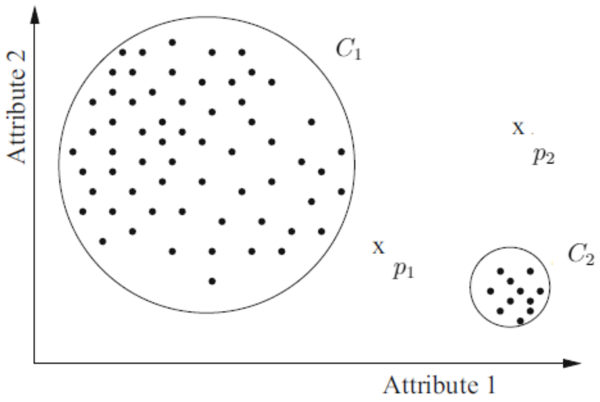


Figure 6: Illustration of a local outlier [40].

5.3 Analytical approach of the density-based methods

5.3.1 Local Outlier Factor (LOF)

Most of the density-based methods follow the concept of local outlier factor which is a measure of difference in density between an object and its neighborhood. That is why we begin our analysis with the following definitions giving the k-distance and the k-distance neighborhood of a point p [8]. As we have discussed in previous chapters a distance between two objects p and o is denoted by $d(p, o)$.

Definition 5.1. For any positive integer k, the k-distance of object p, denoted as $d_k(p)$, is defined as the distance $d(p, o)$ between p and an object $o \in D$ such that:

(a) for at least k objects $o' \in D \backslash \{p\}$ it holds that $d(p, o') \leq d(p, o)$, and

(b) for at most $k - 1$ objects $o' \in D \backslash \{p\}$ it holds that $d(p, o') < d(p, o)$.

Definition 5.2. Given the k-distance of p, the k-distance neighborhood of p (denoted by $N_k(p)$) contains every object whose distance from p is not greater than the k-distance, i.e.

$$N_k(p) = \{q \in D \backslash \{p\} | d(p, q) \leq d_k(p))\}$$

. These objects q are called the k-nearest neighbors of p.

Remark. The cardinality of $N_k(p)$ can be greater than k in case of ties in the k-distances of points.

Definition 5.3. Let k be a natural number. The reachability distance of object p with respect to object o is defined as $rd(p,o) = \max\{d_k(o), d(p,o)\}$.

Intuitively, if the two points are far away from each other then the reachability distance between them is their actual distance. If on the contrary, they are close to each other then their reachability distance equals the k-distance of point o.

While the above definitions work for any natural number k, we are only interested in the notion of **MinPts** and we use the values of the $rd(p,o)$ (for all points o in $N_k(p)$) as a measure of volume both of which determine the local density [8].

Definition 5.4. The local reachability density of p (denoted by $lrd_k(p)$) is defined as

$$lrd_{\mathrm{k}}(p) = \left(\frac{\sum_{o \in N_k(p)} rd_k(p,o)}{|N_k(p)|} \right)^{-1}$$

Intuitively, the local reachability density of an object p is the inverse of the average reachability distance based on the k-nearest neighbors of p. It works as an estimator of the density at point p by analyzing the k-distance of the objects in $N_k(p)$.

Definition 5.5. The (local) outlier factor of p is defined as

$$LOF_k(p) = \frac{\sum_{o \in N_k(p)} \frac{lrd(o)}{lrd(p)}}{|N_k(p)|}$$

The outlier factor of object p is the average of the ratio of the local reachability density of p and those of its k-nearest neighbors. Basically, it measures the outlierness of the point p. The higher the value of $LOF(p)$ the "more of an outlier" an observation is.

We continue the analysis by giving some properties of LOF.

Lemma 5.1. *Let C be a collection of objects. Let rd_{min} denote the minimum reachability distance of objects in C, i.e., $rd_{min} = min\{rd(p,q)|p,q \in C\}$. Similarly, let rd_{max} denote the maximum reachability distance of objects in C. Let ϵ be defined as $\frac{rd_{max}}{rd_{min}} - 1$.*
Then for all objects $p \in C$, such that:

(a) all the k-nearest neighbors q of p are in C, and

(b) all the k-nearest neighbors o of q are also in C,

it holds that $\frac{1}{1+\epsilon} \leq LOF(p) \leq 1+\epsilon$.

Proof. It holds that $rd(p,q) \geq rd_{min}$, $\forall q \in N_k(p)$. Therefore, based on Definition 5.4 it holds for the reachability distance of p that $lrd_k(p) \leq \frac{1}{rd_{min}}$. In contrast, it can be seen that $lrd_k(p) \geq \frac{1}{rd_{max}}$ because it holds that $rd(p,q) \leq rd_{max}$, $\forall q \in N_k(p)$.

Let q be a k-nearest neighbor of p. Working in the same way, we can prove that the local reachability density of q is between the same bounds.

Thus, taking into consideration Definition 1.5 we prove for the upper bound that

$$LOF_k(p) = \frac{\sum_{o \in N_k(p)} \frac{lrd(o)}{lrd(p)}}{|N_k(p)|} \leq \frac{\sum_{o \in N_k(p)} \frac{\frac{1}{rd_{min}}}{\frac{1}{rd_{max}}}}{|N_k(p)|} = \frac{\frac{1}{rd_{min}}}{\frac{1}{rd_{max}}} = 1 + \epsilon$$

. We work analogously for the lower bound. □

The interpretation of Lemma 5.1. is that if a point p is in a dense area of a cluster C then its k-nearest neighbors will also be in C along with their k nearest neighborhoods, leading to a value of ϵ close to zero and a $LOF(p)$ close to one. Thus for that kind of points Lemma 5.1. gives efficiently tight bounds denoting that they cannot be possible outliers [8]

The following theorem is an extension of Lemma 5.1. It corresponds to points that are on the borders of a cluster and gives the same bounds as Lemma 5.1. when a point deep in the cluster is considered.

First we present the following terminology [8].

Definition 5.6. For any object p let

- the $direct_{min}(p)$ denotes the minimum reachability distance between p and a k-nearest neighbor of p, i.e.,
$$direct_{min}(p) = min\{rd_k(p,q) | q \in N_k(p)\}$$

- the $direct_{max}(p)$ denotes the maximum reachability distance between p and a k-nearest neighbor of p, i.e.,
$$direct_{max}(p) = max\{rd_k(p,q) | q \in N_k(p)\}$$

- the $indirect_{min}(p)$ denotes the minimum reachability distance between q and a k-nearest neighbor of q, i.e.,
$$indirect_{min}(p) = min\{rd_k(q,o) | q \in N_k(p) \text{ and } o \in N_k(q)\}$$

- the $indirect_{max}(p)$ denotes the maximum reachability distance between q and a k-nearest neighbor of q, i.e.,
$$indirect_{max}(p) = max\{rd_k(q,o) | q \in N_k(p) \text{ and } o \in N_k(q)\}$$

Theorem 5.2. *Let p be an object from the database D, and $1 \leq k \leq |D|$. Then, it is the case that*

$$\frac{direct_{min}}{indirect_{max}} \leq LOF_k(p) \leq \frac{indirect_{max}}{direct_{min}}$$

Proof. It is similar to the previous proof. In this case we will deal with the lower bound and analogously we can get the upper bound. By definition of $direct_{max}(p)$ we obtain

$$rd_k(p,o) \leq direct_{max}(p) \ \forall o \in N_k(p)$$

From the above inequality we derive

$$\left(\frac{\sum_{o \in N_k(p)} rd_k(p,o)}{|N_k(p)|}\right)^{-1} \geq \frac{1}{direct_{max}(p)} \Rightarrow lrd_k(p) \geq \frac{1}{direct_{max}(p)} \tag{2}$$

Again by definition of $indirect_{min}(p)$ we find

$$rd_k(o,q) \geq indirect_{min}(p)$$

Similarly we find

$$\left(\frac{\sum_{q \in N_k(o)} rd_k(o,q)}{|N_k(o)|}\right)^{-1} \leq \frac{1}{indirect_{min}(p)} \Rightarrow lrd_k(o) \leq \frac{1}{direct_{min}(p)} \tag{3}$$

From the inequalities (1) and (2) we get the final step

$$\frac{\sum_{o \in N_k(p)} \frac{lrd(o)}{lrd(p)}}{|N_k(p)|} \leq \frac{\sum_{o \in N_k(p)} \frac{\frac{1}{direct_{min}(p)}}{\frac{1}{indirect_{max}(p)}}}{|N_k(p)|} \Leftrightarrow LOF_k(p) \geq \frac{indirect_{max}}{direct_{min}}$$

□

Remark. For points deep inside the cluster Theorem 5.2 is equivalent to Lemma 5.1 as the values of $indirect_{max}$ and $direct_{min}$ coincide with the values of rd_{max} and rd_{min} respectively.

Finally, we present (without its proof) Theorem 5.3 - an extension of Theorem 5.2 - which gives more efficient bounds for the points whose k-nearest neighborhood overlaps with more than one cluster [8].

Theorem 5.3. *Let p be an object from the database D, $1 \leq k \leq |D|$, and $C_1, C_2, ..., C_n$ be a partition of $N_k(p)$, i.e. $N_k(p) = C_1 \cup C_2 \cup ... \cup C_n \cup \{p\}$ with $C_i \cap C_j = \emptyset$, $C_i \neq \emptyset$, for $1 \leq i,j \leq n, i \neq j$. Furthermore, let $\xi_i = \frac{|C_i|}{|N_k(p)|}$ be the percentage of objects in p's neighborhood, which are also in C_i. Let the notions $direct^i_{min}(p)$ $direct^i_{max}(p)$ $indirect^i_{min}(p)$ $indirect^i_{max}(p)$ be defined analogously to $direct_{min}(p)$, $direct_{max}(p)$, $indirect_{min}(p)$, and $indirect_{max}(p)$ but restricted to the set C_i. Then, it holds that*

(a) $LOF(p) \geq \left(\sum_{i=1}^n \xi_i \cdot direct^i_{min}(p)\right) \cdot \left(\sum_{i=1}^n \frac{\xi_i}{direct^i_{max}(p)}\right)$

(b) $LOF(p) \leq \left(\sum_{i=1}^n \xi_i \cdot direct^i_{max}(p)\right) \cdot \left(\sum_{i=1}^n \frac{\xi_i}{direct^i_{min}(p)}\right)$

Breunig et al.[8] originally presented the LOF method as a density-based approach because of its ability to adjust to regions of varying density. For example, in the case of Figure 6, the LOF values of data points in clusters C_1 and C_2 will be quite close to one, even though the densities of the two clusters are different. On the other hand, the LOF values of both the outlying points p_1 and p_2 will be much higher since they will be computed in terms of the average of ratios of local reachability densities [40].

In general, the maximum value of $LOF_k(p)$ over a range of different values of k is used as the outlier score in order to rank the different objects. The computational complexity of the LOF method obviously depends on the size of the underlying database. For low dimensional data, distances of the k nearest neighbors are computed with complexity of $O(n)$ using a grid-based approach (n indicates the number of observations) while in data sets of a higher dimension they can be computed with complexity $O(n \log n)$ using an index-based approach. Finally the computational cost of the LOF values is $O(k \cdot n)$. Subsequently, many investigations [27, 30] into the LOF method were able to reduce the computational complexity and handle bigger data sets [8].

5.3.2 Robust Kernel-Based Local Outlier Factor(RKOF)

The RKOF method is an extension of the LOF method in which the same framework is used with more efficient density estimates. The LOF method employs the local reachability distance as a density estimate which in large data sets can be inaccurate especially in the presence of outlying points. In order to achieve more accurate density estimation the RKOF method employs a kernel density estimate for the computation of outlier factors [21].

In the univariate case, the kernel estimator [52] is defined by

$$\hat{f}(x) = \frac{1}{n \cdot h} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right),$$

where X_i denotes the i-th observation in a data set with cardinality n , h is the window width (also called the smoothing parameter) and $K(\cdot)$ is a kernel function that satisfies $\int_{-\infty}^{+\infty} K(x) = 1$

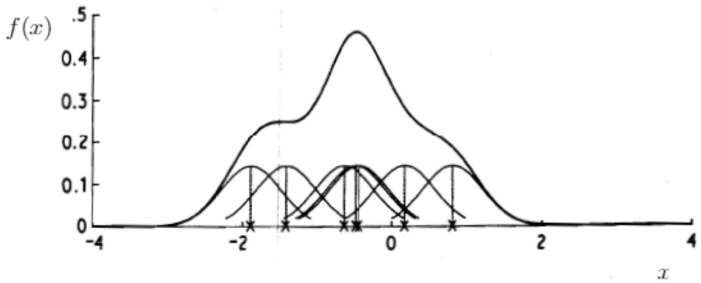


Figure 7: Kernel estimate by individual kernels with window width 0.4 [52]

Intuitively, the kernel estimator is expressed by the sum of "bumps" placed at the observations. The shape of the bumps and their width are determined by the kernel function $K(\cdot)$ and the smoothing parameter h respectively.

With the purpose of allowing the scale parameter of the bumps to vary among the observations to which they correspond, we define the variable kernel estimate [52] by

$$\hat{f}(x) = \frac{1}{n} \sum_{j=1}^{n} \frac{1}{h \cdot d_k(X_j)} \cdot K\left(\frac{x - X_j}{h \cdot d_{j,k}} \right),$$

where $d_k(X_j)$ denotes the distance from point X_j to its k- nearest neighbor. We note that the window width around X_j is proportional to $d_k(X_j)$ so that points in more sparse regions will have flatter kernels. For fixed k,only the parameter h is responsible for the degree of smoothing. The choice of k determines the degree of h's sensitivity to the data locality [52]

Extending to the multivariate case, the kernel function $K(\cdot)$ is a function defined in d-dimensional space that satisfies

$$\int_{\mathbb{R}^d} K(\mathbf{x}) = 1, \mathbf{x} \in \mathbb{R}^d$$

The most common multivariate kernel functions are the Gaussian

$$K(\mathbf{x}) = (2 \cdot \pi)^{-\frac{d}{2}} \cdot \exp(-\frac{1}{2}\mathbf{x}^T\mathbf{x})$$

and the Epanechnikov

$$K(\mathbf{x}) = \begin{cases} \frac{1}{2} \cdot c_d^{-1} \cdot (d-2) \cdot (1 - \mathbf{x}^T\mathbf{x}) & \text{, if } \mathbf{x}^T\mathbf{x} < 1 \\ 0 & \text{, otherwise} \end{cases}$$

where c_d is the volume of the unit d-dimensional sphere [52].

Returning to the RKOF method, we present an extension of the variable kernel estimator in the multivariate case.

Definition 5.7. The local kernel density estimate of p is defined as

$$kde(p) = \frac{\frac{1}{h^\gamma} \cdot \sum_{o \in N_k(p)} \frac{1}{\lambda_o^\gamma} \cdot K\left(\frac{p-o}{h^\gamma \cdot \lambda_o^\gamma} \right)}{N_k(p)},$$

$$\lambda_o = \left(\frac{f(o)}{g} \right)^{-\alpha}, \ \log g = \frac{\sum_{q \in D} \log f(q)}{|D|}$$

where h is the smoothing parameter, γ is the sensitivity parameter, $K(\cdot)$ is the multivariate kernel function and λ_o is the local bandwidth factor. $f(\cdot)$ is a pilot density estimate that satisfies $f(q) > 0$ for all the objects, α is the sensitivity parameter that satisfies $0 \leq \alpha \leq 1$, and g is the geometric mean of $f(q)$.

Remark. In the original (multivariate) variable kernel estimate parameter γ equals the dimension of the data set. [21, 52]

$kde(p)$ is computed locally in the k-distance neighborhood of object p while also retaining the adaptive kernel window width that, as we discussed, is allowed to vary from one object to another.

Basically, the pilot density estimate is used to get a rough idea about the density and yields a pattern of bandwidths (corresponding to the various observations) which are used to build the kernel local estimate [52].

Gao et al. [21] use as pilot density estimate the following function

$$f(o) = \frac{1}{d_k(o)}$$

Substituting this into the $kde(p)$ we obtain

$$kde(p) = \frac{\sum_{o \in N_k(p)} \frac{1}{(C \cdot d_k(o)^{\alpha})^{\gamma}} \cdot K\left(\frac{p-o}{C \cdot d_k(o)^{\alpha}}\right)}{|N_k(p)|}, \, C = h \cdot g^{\alpha}$$

where the default values of C and α equal one.

Definition 5.8. The weighted density estimate of p's neighborhood is defined as

$$wde(p) = \frac{\sum_{o \in N_k(p)} \omega_o \cdot kde(o)}{\sum_{o \in N_k(p)} \omega_o}, \, \omega_o = \exp\left\{ - \frac{\left(\frac{d_k(o)}{d_{k_{min}}} - 1\right)^2}{2 \cdot \sigma^2} \right\}$$

where ω_o is the weight of object o in the k-distance neighborhood of object p, σ is the variance with the default value 1, and $d_{k_{min}} = min\{d_n(o)|o \in N_k(p)\}$.

Based on the definition of LOF, the detection performance is sensitive to the parameter k, which must be sufficiently large in order for the k- distance neighborhood to consist of enough normal points so that outliers have the chance to be detected. In the weighted neighborhood density estimate, the weight of the neighbor object is a monotonically decreasing function of its k-distance with the largest weight equal to one for the nearest neighbor. Thus, the weighted neighborhood density estimate is able to detect outlying objects accurately even if the number of outliers in the neighborhood equals the number of normal objects making it more robust than the local reachability density used in the LOF method [21].

Definition 5.9. The robust kernel-based outlier factor of p is defined as

$$RKOF(p) = \frac{wde(p)}{kde(p)},$$

where $wde(p)$ is the density estimate of the k-distance neighborhood of p, and $kde(p)$ is the local density estimate of p.

As for LOF, RKOF measures the degree to which a point is outlying. The larger the RKOF value of a point the more probable it is to be an outlier. The computational complexity is the same as LOF's [52].

5.4 Applying outlier detection using LOF and RKOF algorithms to wood specific gravity data

In this section we will test three functions on Wood specific gravity data (we recall that the Wood specific gravity data is a data set of twenty observations on six variables, four of which (4, 6, 8, 19) are known to be outliers).

- **LOF**: Computes a local density for observations with a user-given k-nearest neighbors. The density is compared to the density of the respective nearest neighbors, resulting in the local outlier factor.
- **RKOF**: Computes a kernel density estimation by comparing density estimation to the density of neighboring observations resulting in the robust kernel outlier factor. A Gaussian kernel is used for density estimation, given a bandwidth with k-distance.
- **dens**: Computes the RKOF values based on **RKOF** function and based on a user-given percentile cutoff declares as outliers the observations whose RKOF values exceed the underlying cutoff. Moreover, it provides an outlier probability like in **depthout**, **nn** and **nnk**

The first two functions can be found in ”DDoutlier” package while the latter is implemented in ”OutlierDetection”

We begin with the first function. Its code is below

```
   LOF(dataset, k)
```

The only arguments needed are the underlying data set and the **MnPts** determining the local density. The function returns the LOF value of each observation.

We obtain the following results

```
   LOF(my_data, k = 6)
 [1] 0.9320898 1.0289675 0.9792408 1.3571371 0.9723653 1.3929624 1.3058637
 1.4051025 1.1767145 1.2400128 1.1262819 1.2063343 1.1032250
[14] 0.9712314 1.0205625 0.9908529 0.9442170 1.1100389 1.4282616 1.0432364
```

Sorting the above scores in ascending order with **order()** function we obtain the following vector

```
   > order(LOF(my_data, k = 6))
 [1]  1 17 14  5  3 16 15  2 20 13 18 11  9 12 10  7  4  6  8 19
```

We note that observations 4, 6, 8, 19 have the higher LOF values for k=6, 7, ..., 15 (see Table 5). However, for values higher than eight we observe lower LOF values converging to one. The LOF method was indeed able to direct higher scores to the true outliers of the data set. Due to the fact that normal data points have a LOF value close to one and taking into consideration the fourth observation, a question arises whether a value equal to 1.3571371 is high enough for this object to be denoted as an outlier.

The code of the second function is as follows

```
   RKOF(dataset, k , C = 1, alpha = 1, sigma2 = 1)
```

The first two arguments are the same as for the first function while the other ones have been introduced in the previous chapter. Moreover, **C** is a multiplication parameter for k-distance of neighboring observations acting like a bandwidth increaser. Default is one such that k-distance is used for the Gaussian kernel. Finally, **alpha** is a sensitivity parameter for k-distance (small **alpha** creates small variance in RKOF and vice versa) and **sigma2** is a variance parameter for weighting of neighboring observations. Both of them have also default value equal to one. The function returns the RKOF value of each observation.

We obtain the following results

```
     RKOF(my_data, k = 8, C = 1, alpha = 1, sigma2 = 1)
 [1] 1.0274242 0.9525171 0.9304886 2.4660712 0.9138639 2.4192465 2.2790186
 2.6386842 1.7495769 1.9928869 1.1746130 1.6597684 1.4778650
[14] 0.7717380 0.8477847 1.0204903 1.0984443 1.3634725 2.8206986 1.3187638

     order(RKOF(my_data, k = 8, C = 1, alpha = 1, sigma2 = 1))
 [1] 14 15  5  3  2 16  1 17 11 20 18 13 12  9 10  7  6  4  8 19
```

We note that observations 4, 6, 8, 9 have the higher RKOF values for k=8, ..., 19 (see in Table 5). Moreover, for values 10 or higher these observations are in the same order as in LOF method. The main difference apart from k is the fact that the method directs much higher RKOF values to the outlying points (by these results one could denote the seventh observation also as an outlier, although in varying values of k its score keeps reducing).

Furthermore, if we choose to change the value of **sigma2** to 0.1 we obtain the following results

```
   >RKOF(my_data, k=7, C=1, alpha = 1, sigma2 =0.1)
 [1] 1.0963566 1.1197494 1.0016784 3.7314614 1.0502284 3.8828330 2.7476913 3.2005587
 2.3847559 2.5486490 1.5015050 2.1789623 2.0767597 0.8336603 0.9017041 1.1594693
 1.2525511 1.8449604 3.4199065 1.5509820
```

```
    order(RKOF(my_data, k = 7, C = 1, alpha = 1, sigma2 = 0.11))
[1] 14 15  3  5  1  2 16 17 11 20 18 13 12  9 10  7  8 19  4  6
```

As we can see from the above results, giving a smaller value to the parameter **sigma2**, not only we detect the true outliers with the highest RKOF values faster (smaller k) but also we obtain a clearer idea of the outlying objects through their RKOF values as the latter ones have much bigger values than the rest of the data. Finally, we can safely speculate that the nature of these outliers having been already compared with three different kind of methods is that of an extreme value, because of their mutual sensitivity to that kind of outliers.

Outliers			
LOF	**RKOF** ($\sigma = 1$)	**RKOF** ($\sigma = 0.1$)	k
4 6 8 19 12 9 7 10	13 17 12 18 9 20 7 10	17 13 12 18 20 9 7 10	4
9 12 6 8 19 10 7 4	11 18 9 12 20 4 7 10	11 18 12 20 9 7 10 4	5
9 12 10 7 4 6 8 19	18 10 4 8 9 12 6 7	19 12 10 9 7 4 8 6	6
9 12 10 7 4 6 8 19	12 9 10 8 19 7 4 6	12 9 10 7 8 19 4 6	7
13 12 10 7 4 6 8 19	12 9 10 7 6 4 8 19	12 9 10 7 4 6 8 19	8
13 12 10 7 4 6 8 19	9 12 10 7 4 6 19 8	9 12 7 10 4 6 8 19	9
18 12 10 7 4 6 8 19	9 12 10 7 4 6 8 19	9 12 10 7 4 6 8 19	10

Table 5: Number of detected outliers for different k for **LOF** and **RKOF** functions

As we see in Table 5, LOF can be considered more efficient, in the sense of assigning the highest values to the true outliers at smaller values of k. That could be due to the small size of the dataset and the small percentage of contamination (20%). Either way, we observe in both methods a convergence in the ascending order of the observations while the methods are declaring the true outliers. Moreover, this phenomenon also appeared when testing KNN_IN function for different values of k indicating a similarity in the behaviour of these methods.

Finally, we present **dens** function. Its code and obtained results for $k = 4$ are below

```
 >dens(my_data, k = 4, C = 1, alpha = 1, sigma2 = 1,cutoff = 0.75, rnames = F,
 boottimes = 100)

$`Location of Outlier`
[1]  7  9 10 20

$`Outlier Probability`
[1] 1.00 0.89 1.00 0.94
```

According to the results, **dens** function is indeed consistent with the RKOF values presented above for $k = 4$ in Table 5. The only difference is that it declared only the ones that had values higher that the given percentage. This can be very helpful for datasets with more observations. Moreover, instead of the order given by the RKOF values, an outlier possibility is given as an outlier score indicating which observation is declared as an outlier the most. Accordingly, the ascending order of the outliers based on their outlier possibility matches the ascending order presented in Table 5.

5.5 Conclusions

Density-based methods work efficiently in data sets with varying local densities, while other methods may not recognise all the non normal objects. The LOF method is the basis for many density-based methods and is able to adjust to these local density variations and measure the "outlyingness" of the objects by their LOF values. The RKOF method extends the latter method. It is more robust to the choice of k and as we noticed earlier outlying observations receive higher scores than LOF's making much easier the decision to the user to denote them.

6 Mahalanobis distance

6.1 Introduction

The methods presented in Chapters 5 and 6 are based on the most fundamental idea in multivariate data analysis, that is to measure the similarity of the objects in target data by their distance from each other. As we have already discussed, objects with large distance show low similarity, while objects with small distance between them are considered to belong to the same category or to have similar properties (that part of the data is also called normal). The distance between objects depends on the selected distance metric function, the dimension, and on the scaling of each dimension. Distance measurements in high-dimensional space are extensions of distance measures in two dimensions [62].

Let two objects be defined by the vectors $\mathbf{x}_A = (x_{A_1}, x_{A_2}, ..., x_{A_d})$, $\mathbf{x}_B = (x_{B_1}, x_{B_2}, ..., x_{B_d})$. Most used is the **Euclidean Distance** (denoted by $d_{Euclidean}()$) which is equivalent to that used in daily life. Applying Pythagoras' rule gives

$$d_{Euclidean}(\mathbf{x}_A, \mathbf{x}_B) = \left(\sum_{j=1}^{d} (x_{B_j} - x_{A_j})^2 \right)^{\frac{1}{2}} = (\mathbf{x}_B - \mathbf{x}_A)^T (\mathbf{x}_B - \mathbf{x}_A)$$

Over the years, different kinds of measures have been introduced such us the Minkowski distance

$$d_{Minkowski} = \left(\sum_{j=1}^{d} (x_{B_j} - x_{A_j})^d \right)^{\frac{1}{d}}$$

and the Manhattan distance

$$d_{Manhattan} = ||\mathbf{x}_B - \mathbf{x}_A||_1 = \sum_{j=1}^{d} |x_{B_j} - x_{A_j}|$$

. The **Mahalanobis distance** analyzed below considers the distribution of the object points in the variable space (as characterized by the covariance matrix) and more importantly is not affected by the scaling of the variables [62].

6.2 Aim of the Mahalanobis distance

The concept of the Mahalanobis distance can also be used in the context of detecting multivariate outliers. In Figure 8, we show how more concentrated to the data are the ellipses formed from the Mahalanobis distances of all the objects from the center than the circles formed by the Euclidean distances of the objects to the center. Unlike Euclidean distace, Mahalanobis distance takes into account the correlation between the variables via the covarience matrix so that it can provide safer results concerning the outliers and the geometrical structure of the dataset in general. In that context, multivariate outliers can be interpreted as points that are far away from the ellipse or ellipsoid (if we are dealing with high dimensional data) i.e. those noted with exceptionally high Mahalanobis distances [62].

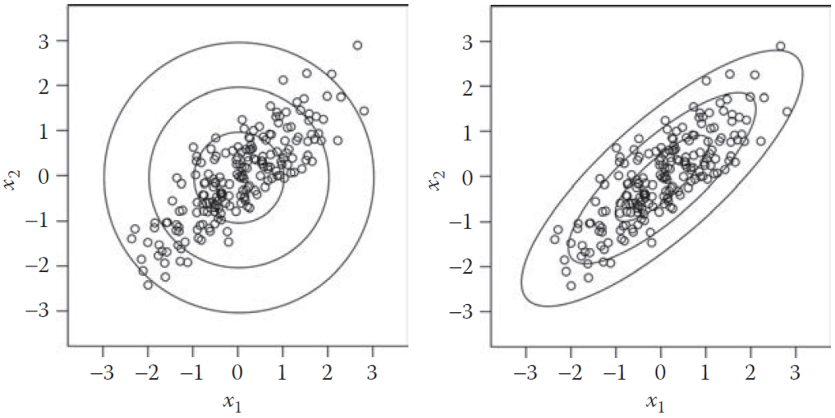


Figure 8: Euclidean distance (left) and Mahalanobis distance (right) [62]

6.3 Analytical approach of the Mahalanobis distance

In general, the Mahalanobis distance between two d-dimensional objects x_A and x_B without any distributional assumption is defined by

$$MD(\mathbf{x}_A, \mathbf{x}_B) = \sqrt{[(\mathbf{x}_B - \mathbf{x}_A)^T \cdot \mathbf{S}^{-1\cdot}(\mathbf{x}_B - \mathbf{x}_A)]}$$

where $\mathbf{S}$ is the sample covariance of the data matrix. The computation of the Mahalanobis distance requires the inversion of the covariance matrix $\mathbf{S}$ [62].

Even though Mahalanobis distance can be assigned to all kinds of datasets, its values correspond to coherent results when the entire dataset is assumed to be normally distributed about its mean in the form of a multivariate Gaussian distribution. Let μ be the d-dimensional mean vector of a d-dimensional data set, and Σ be its $d \times d$ covariance matrix. In this case, the (i, j)th entry of the covariance matrix is equal to the covariance between the dimensions i and j. The covariance matrix Σ $(d \times d)$ is a quadratic, symmetric matrix which consist of the covariances σ_{jk} between every two variables x_j and x_k . The cases $j = k$ (main diagonal) are “covariances” between one and the same variable, which are in fact the variances σ_{jj} of the variables x_j for $j = 1, ..., d$.

The probability distribution f($\mathbf{x}$) for a d-dimensional data point $\mathbf{x}$ can be defined as follows:

$$f(\mathbf{x}) = \frac{1}{\sqrt{|\Sigma|} \cdot (2 \cdot \pi)^{\frac{d}{2}}} \cdot e^{-\frac{1}{2} \cdot (\mathbf{x} - \mu) \cdot \Sigma^{-1} \cdot (\mathbf{x} - \mu)^T}$$

The value of $|\Sigma|$ denotes the determinant of the covariance matrix. We note that the term in the exponent is (half) the squared Mahalanobis distance between the data point $\mathbf{x}$ and the mean μ of the data. The value in the exponent of the normal distribution above is used as the outlier score [2].

Suppose a $n \times d$ data matrix $\mathbf{X}$ (n observations d variables). The classical measure of covariance between two variables $\mathbf{x}_j$ and $\mathbf{x}_k$ is the sample covariance, s_{jk}, defined by

$$\mathbf{s}_{jk} = \frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \bar{\mathbf{x}}_j) \cdot (x_{ik} - \bar{\mathbf{x}}_k)$$

For mean-centered variables and written in vector notation

$$\mathbf{s}_{jk} = \frac{1}{n-1} \sum_{i=1}^{n} x_{ij} \cdot x_{ik} = \mathbf{x}_j^T \cdot \mathbf{x}_k$$

Based on the above definition, the sample covariance matrix $\mathbf{S}$ can be calculated for mean-centered $\mathbf{X}$ by

$$\mathbf{S} = \frac{1}{n-1} \mathbf{X}^T \cdot \mathbf{X}$$

In general, $\mathbf{S}$ is given by

$$\mathbf{S} = \frac{1}{n-1} (\mathbf{X} - \mathbf{1}\bar{\mathbf{x}})^T \cdot (\mathbf{X} - \mathbf{1}\bar{\mathbf{x}}),$$

where $\mathbf{1}$ is a vector of ones of length n and $\bar{\mathbf{x}} = (\bar{x}_1, \bar{x}_2, ..., \bar{x}_d)$ is the mean vector.

Another measure for the relation of variables $\mathbf{x}_j$ and $\mathbf{x}_k$ is their correlation originally proposed by Pearson defined by

$$r_{j,k} = \frac{\sum_{i=1}^{n} (x_{ij} - \bar{\mathbf{x}}_j) \cdot (x_{ik} - \bar{\mathbf{x}}_k)}{\sqrt{\sum_{i=1}^{n} (x_{ij} - \bar{\mathbf{x}}_j)^2} \cdot \sqrt{\sum_{i=1}^{n} (x_{ik} - \bar{\mathbf{x}}_k)^2}}$$

The range of $r_{j,k}$ is -1 to +1; a value of +1 indicates a perfect linear relationship, a value of -1 indicates a perfect inverse linear relationship; absolute values of approximately < 0.3 indicate a poor or no linear relationship [62]

The Mahalanobis distance is similar to the Euclidean distance, except that it normalizes the data on the basis of the inter-attribute correlations. For example, if the axis system of the data were to be rotated to the principal directions (using principal components analysis) then the data would have no correlations between the target variables and so the Mahalanobis distance would not be affected by them. The Mahalanobis distance is simply equal to the Euclidean distance in such a transformed (axes-rotated) data set after dividing each of the transformed coordinate values by the standard deviation of that direction. This approach recognizes the fact that the different directions of correlation have different variance, and the data should be treated in a statistically normalized way along these directions [2].

Thus, the information about the distribution of the object points in the variable space is contained in the covariance matrix and its estimation as well as the mean's can be affected by a possible contamination of the dataset. As we will discuss more specifically in Chapter 7, a sufficient number of anomalous points can damage the estimations of scale and location of a target dataset, leading to erroneous Mahalanobis distances, that is, outlier scores.

A more robust measure, can be derived from a robust covariance estimator such as the **minimum covariance determinant** (MCD) estimator [43, 46, 62]. The MCD estimator searches for a subset of h observations having the smallest determinant of their classical sample covariance matrix. The robust location estimator $\mathbf{T}$ is then defined as the arithmetic mean of these h observations, and the robust covariance estimator is given by the sample covariance matrix of the h observations as well as the correlation for each pair of variables.

The Mahalanobis distance based on the assumption that the dataset is Normally distributed approximately follows a chi-square distribution χ^2_d with d degrees of freedom (d is the number of variables). If an object has a larger squared Mahalanobis distance than the cutoff it is exceptionally high and can therefore be considered as a potential multivariate outlier. One can then take a quantile of the chi-square distribution, like the 97.5% quantile $x^2_{d,0.975}$ as cutoff value.

Even though for higher dimensions we cannot visualize the ellipsoid formed by the Mahalanobis distances of the d-dimensional points, due to the fact that their values are one-dimensional, a index plot can be made showing each observation by its Mahalanobis distance value and a straight line indicating the cutoff value. Observations that exceed this line are considered to be outliers.

Moreover, Garrett [22] proposed the χ^2 plot in order to detect the outliers. This plots the empirical distribution function of the robust Mahalanobis distances (denoted by RD that is the Mahalanobis distances computed using the MCD estimations of location and scatter) against the χ^2 distribution. The cutoff value can also be used as a break in the tails and observations whose MD exceed the break are considered to be outliers.

On that basis, Filzmoser [16] introduced the adaptive quartile. The adaptive quartile is basically a threshold that takes into account the difference between the empirical and a theoretical distribution of the RD's in the tails. If we denote by $G_n(u)$ the empirical distribution of the RD^2_i and by $G_n(u)$ the theoretical function of χ^2_d then we can compare their tails in order to detect outliers, since for multivariate normally distributed samples G_n converges to G. Defining the tails by $\delta = \chi^2_{p,0.975}$ we obtain

$$p_n(\delta) = \underset{u \geq \delta}{sup} \big(G(u) - G_n(u) \big)^+$$

indicating the positive differences beyond the tails (in case of multivariate normally distributed data a critical value $p_{crit}(\delta, n, d)$ is employed for more exact value of $p_n(\delta)$, denoted as $\alpha_n(\delta)$ [16]). Finally, the target cutoff is defined by

$$c_n(\delta) = G^{-1}\big(1 - p_n(\delta)\big)$$

6.4 Applying outlier detection using Mahalanobis distance to wood specific gravity data

In this section we will apply two detection methods using the Mahalanobis distance. The first, is the classical one as discussed above and the second one is based on the MCD estimator.

We recall that the Wood specific is a data set of twenty observations on six variables, four of which (4, 6, 8, 19) are known to be outliers.

We use four different functions that were built in the software enviroment of **R**:

- **maha**: Computes the Mahalanobis distance of an observation and based on the Chi square cutoff, labels an observation as outlier. Outlyingness of the labelled "outlier" is also reported based on its p values (That is a vector of $(1 - p)$ values of the detected observations).

- **aq.plot**: Plots the ordered squared robust Mahalanobis distances of the observations against the empirical distribution of the RD_i^2. In addition the distribution function of χ_d^2 is plotted as well as two vertical lines corresponding to the x_d^2-quantile specified in the argument list (default is 0.975) and the adjusted quantile. Three additional graphics are created (the first showing the data, the second showing the outliers detected by the specified quantile of the χ_d^2 distribution and the third showing these detected outliers by the adjusted quantile). In case of $d > 2$ the data is projected on the first two robust principal components (i.e. principal component analysis based on a covariance matrix estimated by the MCD method).

- **Moutlier**: Plots the classical and the robust (based on the MCD estimator) Mahalanobis distance.

We begin with the first function. Its code and obtained results are below

```
  > maha(my_data,cutoff=0.8)
```

```
 \begin{verbatim}
     $`Outlier Observations`
       x1     x2    x3    x4    x5     y
7  0.489 0.1231 0.562 0.455 0.824 0.481
11 0.664 0.1588 0.506 0.481 0.867 0.554
12 0.703 0.1335 0.519 0.484 0.812 0.519
16 0.523 0.1320 0.505 0.612 0.919 0.508

$`Location of Outlier`
[1]  7 11 12 16

$`Outlier Probability`
```

```
[1] 0.8850952 0.8922186 0.8136708 0.8487555
```

As we see from the above results, none of the true outliers were detected. Furthermore, if we increase the percentile threshold to 0.9, the method fails to detect any outlier. We speculate that the presence of these contaminated outliers caused a swamping effect, with normal points like 7, 11, 12 and 16 detected as outliers.

For identifying outliers, it is crucial how center mean and covariance are estimated from the data. Since the classical estimators, arithmetic mean vector $\bar{\mathbf{x}}$ and sample covariance matrix $\mathbf{S}$ are sensitive to outliers, they can be inefficient for the purpose of outlier detection.

The code of the second function is as follows

```
 >   aq.plot(my_data, delta=qchisq(0.975, df=ncol(x)), quan=0.6, alpha=0.05)
```

The arguments of the function are

- **x**: A given data set
- **delta**: A quantile of the χ^2 distribution with degrees of freedom equal to the numbers of variables of the data. This quantile appears as cyan-colored vertical line in the plot.
- **quan**: A proportion of observations which are used for MCD estimations.
- **alpha**: Maximum thresholding proportion of outlying observations (optional scalar, default: alpha = 0.05). For $\alpha = 0.025$ overlap with the quartile cutoff of $\chi^2_{p,0.975}$

In Figure 9, the upper left plot shows the RD_i of the objects projected on the first two principal components. The two plots on the bottom marks outliers, the observations that exceed (on the left) χ_d quantile and (on the right) the adjusted quantile. Finally, the upper right plot, shows the empirical distribution of the RD_i as well as the chi^2_6 distribution and the aforementioned cutoff values labeled by cyan and blue colors.

Based on the following results, the method managed to detect the true outliers in the data set. Moreover, all three plots detected the same outliers along with the results in Boolean form (TRUE-FALSE)

```
    Projection to the first and second robust principal components.
Proportion of total variation (explained variance): 0.6152118
$outliers
 [1] FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE
 FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
```

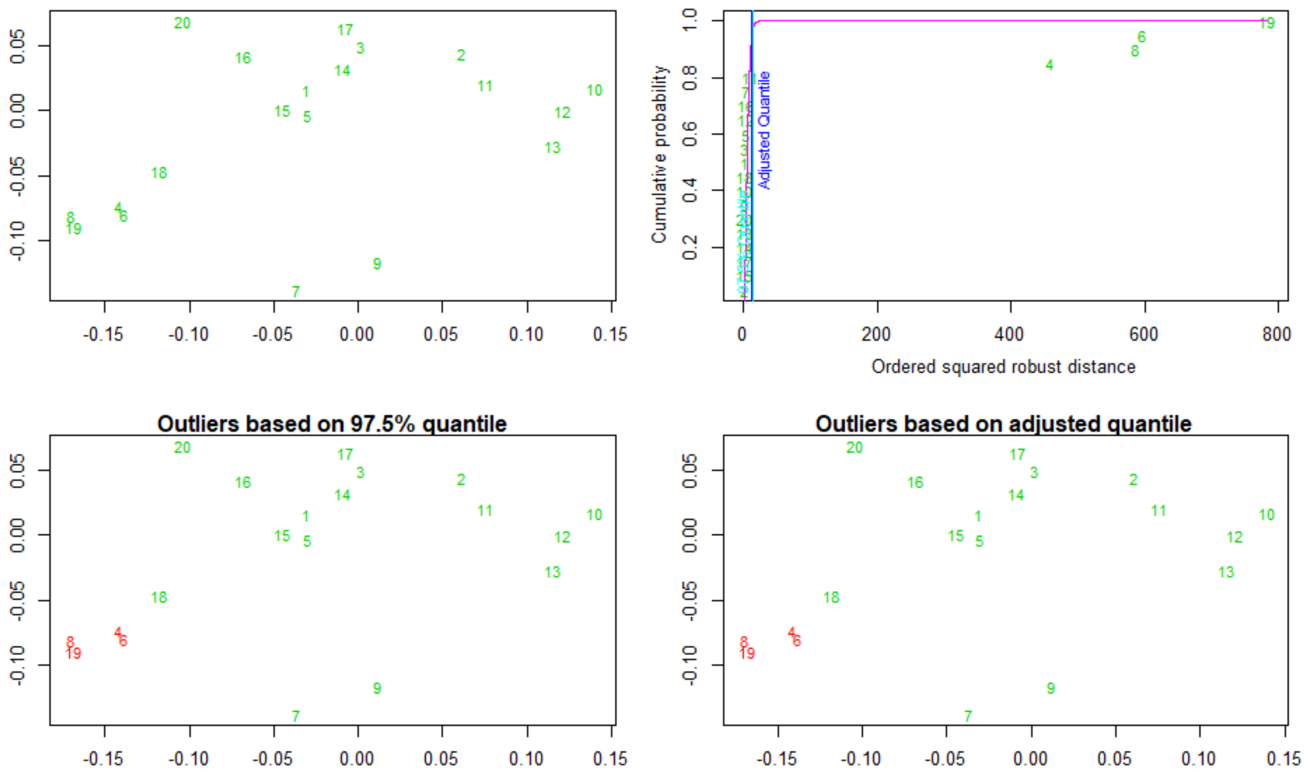


Figure 9: Plots produced by the **aq.plot** function (wood specific data)

Finally, we give another presentation of the above results with the plots below. The code of the function **Moutlier** is

```
 >  Moutlier(my_data, quantile = 0.975, plot = TRUE)
```

The values of the function are the data set and a quantile (cutoff value) for the Mahalanobis distance. These plots are given along with the scores for both methods (Classical vs Robust) plus the value with the outlier cutoff. The horizontal line indicates the cutoff value: $\sqrt{x^2_{20,0.975}} = 3.801233$

```
    $md
 [1] 2.245038 1.246945 2.306514 2.032232 2.154265 2.061767 3.199989 2.130094 2.424357 2.79667
[15] 1.553139 3.069379 2.153202 2.178547 2.488627 2.342370


$rd
 [1]  1.3006229  0.7936416  1.2717753 17.3993341  1.4017126 19.8568695  3.6723871 19.8399457
[13]  1.0419987  1.3601641  1.0569889  3.3563387  1.1397842  1.2200950 22.8174234  1.0946777


$cutoff
[1] 3.801233
```

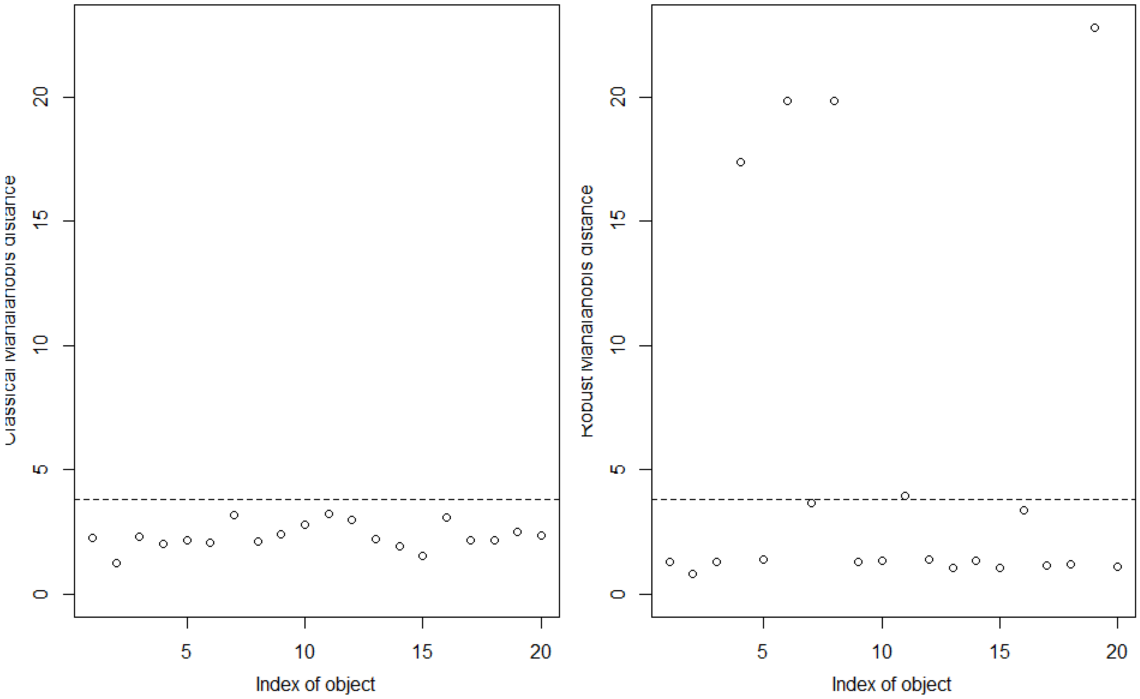


Figure 10: Classical Mahalanobis distance scores (left) Robust Mahalanobis distance scores (right)

Based on Figure 10, the first method does not detect any outlier while the other one was able to detect the known outliers plus the 11th observation with a value of 3.93 which is a little above the line.

6.5 Conclusions

Mahalanobis distance is a distance-based method for outlier detection. It is a distance measure that accounts for the covariance structure, estimated by the sample covariance matrix **S**. The problem of swamping arise due to the influence of outliers on classical location and scatter estimates (sample mean and sample covariance matrix), which implies that the estimated distance will not be robust to outliers. That is why we also choose robust versions of the Mahalanobis distance such as the one that is based on the MCD estimator. A problem with the Mahalanobis distance is the need to invert the covariance matrix which may be difficult with highly correlating variables. Moreover, in cases which the data may have many different clusters with different orientations, this kind of distance-based method may not be effective.

7 Distribution-Based Methods

7.1 Introduction

In this section we present some methods that aim to declare objects in a data set as outliers based on a certain assumption. The most common is that the objects have been generated from a multivariate Gaussian distribution. The squared distances of the objects from their estimated center are χ^2 distributed with p numbers of freedom and the distances that exceed a certain cutoff value based on the χ^2 distribution declared as outliers.

Generally, in order to provide efficient outlier analysis, it is desirable for these methods to have some properties, that is, to be affine equivariant, robust and computationally not too expensive.

An estimator T is affine equivariant if it yields

$$T(\mathbf{X} \cdot \mathbf{A} + \mathbf{b}) = T(\mathbf{X}) \cdot \mathbf{A} + \mathbf{b},$$

where X is the data matrix, A is any non singular matrix and b a random vector, meaning that it does not depend on any orientation of the data. A method in order to be robust, has to have high breakdown point, that is handling properly the data even if the percentage of the contamination is high [11, 7]. The combination of these two properties forces the computational cost to increase. In this section we will discuss how these methods handle these requirements in order to provide efficient results.

7.2 FastMCD algorithm

The minimum covariance determinant (MCD) estimator was originally proposed by Rousseeuw [43] and its objective is to find in a data set (of n observations) h observations whose classical covariance matrix has the lowest determinant. Basically, it corresponds to finding the h points for which the classical tolerance ellipsoid (for a given level) has minimum volume, and then taking its center. It is considered to be one of the robust tools that were built in order to deal with the masking and swamping effect that prevents the Mahanalobis distance from being able to handle the presence of many outliers in the underlying data set [45, 43].

The main idea of the MCD method lies in the following theorem [45]

Theorem 7.1. *Consider a data set $N = \{\mathbf{x}_1, ., \mathbf{x}_n\}$, where $\mathbf{x}_i = (x_{i1}, ..., x_{ip})$ for $i = 1, ..., n$ is a p-variate observation. Let $H_1 \subset \{1, .., n\}$ with $|H_1| = h$, and put $\mathbf{T}_1 := \frac{1}{h} \cdot \sum_{i \in H_1} \mathbf{x}_i$ and $\mathbf{S}_1 := \frac{1}{h} \cdot \sum_{i \in H_i} (\mathbf{x}_i - \mathbf{T}_1) \cdot (\mathbf{x}_i - \mathbf{T}_1)'$. If $det(\mathbf{S}_1) \neq 0$, define the relative distances*

$$d_1(i) := \sqrt{(\mathbf{x}_i - \mathbf{T}_1)' \cdot \mathbf{S}_1^{-1} \cdot (\mathbf{x}_i - \mathbf{T}_1)} \qquad \textit{for } i = 1, ..., n.$$

Now take H_2 such that $\{d_1(i) | i \in H_2\} := \{(d_1)_{1:n}, (d_1)_{2:n}, ..., (d_1)_{h:n}\}$, where $(d_1)_{1:n} \leq (d_1)_{2:n} \leq ... \leq (d_1)_{h:n}$ are the ordered distances , and compute $\mathbf{T}_2$ and $\mathbf{S}_2$ based on H_2. Then

$$det(\mathbf{S}_2) \leq det(\mathbf{S}_1)$$

with equality if and only if $\mathbf{T}_2 = \mathbf{T}_1$ and $\mathbf{S}_2 = \mathbf{S}_1$.

Theorem 1.1 generates an iterative process in which we move from pair $(\mathbf{T}_k, \mathbf{S}_k)$ to pair $(\mathbf{T}_{k+1}, \mathbf{S}_{k+1})$, by producing H_{k+1}, computing each time the n distances (with respect to $(\mathbf{T}_k, \mathbf{S}_k)$) and sorting them in an ascending order. Since the determinants are non-negative and it holds that $det(\mathbf{S}_{k+1}) \leq det(\mathbf{S}_k)$ for all $k \in \mathbb{N}$ the procedure converges (in practice the number of iterations is lower than ten) [45].

With that being said, we note that the MCD algorithm basically applies Theorem 1.1 to initial choices of H_1 subsets until convergence and keeps the one with the lowest determinant.

Corollary 7.1.1. *The MCD subset H of N is separated from $N \setminus H$ by an ellipsoid.*

Proof. Suppose any H subset of the iteration process generated by Theorem 1.1. By definition of H it holds that

$$d^2(i) \leq \lambda = \left\{ (\mathbf{x} - \mathbf{T})' \cdot S^{-1} \cdot (\mathbf{x} - \mathbf{T}) \right\}_{h:n} \qquad \forall x_i \in H,$$

where λ is the the maximum squared distance of the observations $1, ..., h$ (and the least distance of the observations $h, ..., n$). On the other hand for the remaining points it holds that

$$d^2(i) \geq \lambda \quad \forall x_i \notin H$$

Defining the ellipsoid $E = \left\{ \mathbf{x} \mid (\mathbf{x} - \mathbf{T})' \cdot S^{-1} \cdot (\mathbf{x} - \mathbf{T}) \leq \lambda \right\}$, we get that $H \subset E$ and $N \setminus H \subset \overline{\mathrm{E}}$.

□

Remark. There is at least one point in H subset in the borders of the ellipsoid while it is not necessary for $x_j \notin H$ to be.

Based on Corollary 7.1.1. [45] we have to be careful in the choice of the initial H_1. It can be arbitrary, but in that way the method will be inefficient in data sets with many outliers since the ellipsoid that will be constructed may consist of outlying objects. From a non-robust perspective, one could choose h to be equal with n. In that way the MCD location estimate $\mathbf{T_1}$ would be the arithmetic mean and MCD scatter estimate $\mathbf{S}_1$ would be the covariance matrix of the whole data set. The best initial choice is an H subset of $p+1$ objects to which we will keep adding objects until it yields a non-negative determinant (having a subset with fewer observations than variables would be pointless since it will always give zero determinant). In that way we note that $h = \frac{n+p+1}{2}$ gives MCD its highest possible breakdown value [34]. Alternatively, due to the fact that most cases of data sets are no more than 25% contaminated, a good choice for the initial cardinality of the H subset would be $0.75 \cdot n$ [45].

Having enough proper initial choices, the algorithm begins with the iterations. It has been proved that robust solutions are provided in the first two or three steps so a pruning can be made in the remaining iterations.

Finally, when the method deals with high dimensional data sets, it provides a pre-processing step in which a partition is made of the data set. Then, after the algorithm is applied to each partition it stores the ten best solutions for each partition. Then the subsets are pooled, yielding a merged set to which we apply the algorithm with initial subsets the ones that the algorithm stored in the first place. The ten best solutions after the iteration steps are stored and used as initial subsets for the whole data set to which the method is applied one final time giving the pair $(\mathbf{T}_{full}, \mathbf{S}_{full})$ with the lowest determinant among these ten.

The latter pair is used for the robust distances of the observations defined as

$$RD_i = \sqrt{(\mathbf{x}_i - \mathbf{T}_{full})' \cdot \mathbf{S}_{full}^{-1} \cdot (\mathbf{x}_i - \mathbf{T}_{full})} \quad i = 1, ..., n$$

If the data is generated from the multivariate Gaussian distribution we denote

$$\mathbf{T}_{MCD} = \mathbf{T}_{full} \quad \text{and} \quad \mathbf{S}_{MCD} = \frac{med_i RD_i}{\chi_{p,0.5}} \cdot \mathbf{S}_{full}$$

The method flags the observations that exceed a cutoff value (in most cases that value is $\sqrt{\chi^2_{p,0.975}}$, where p is the number of the variables).

The MCD method is affine equivariant, that is, when the data is translated or subjected to a linear transformation, the resulting $(\mathbf{T}_{full}, \mathbf{S}_{full})$ will transform accordingly. The computational cost of each iteration is $O(n)$ [45].

7.3 BACON algorithm

The blocked adaptive computationally efficient outlier nominators (BACON) algorithm was introduced by Billor et al. [7]. The method like FastMCD searches through an iterative process for a subset with certain properties. Moreover, this subset is considered to be free of outlying objects, thus declaring every object that is not in that subset as an outlier.

Consider an $n \times p$ data matrix.

BACON algorithm starts by choosing a basic subset of $m > p$ observations using one of the following methods:

(a) It computes the Mahalanobis distances of the observations

$$MD_i = \sqrt{(\mathbf{x}_i - \mathbf{T})' \cdot \mathbf{S}^{-1} \cdot (\mathbf{x}_i - \mathbf{T})} \quad i = 1, ..., n$$

where $\mathbf{T}$ and $\mathbf{S}$ are the mean and covariance matrix respectively of all n observations. The $m = c \cdot p$ (where c is an integer given by the user) observations with the smallest distances form the initial basic subset.

(b) It computes the distances of the observations from the median i.e. $||\mathbf{x}_i - \mathbf{m}||$, where $\mathbf{m} = med(\mathbf{x}'_1, .., \mathbf{x}'_n)$ and $|| \cdot ||$ is the vector norm. The m observations with the smallest distances form the initial basic subset.

We note that method (a) is affine equivariant but it is not robust which means that it gives poor results when the rate of the contamination is high. In contrast, method (b) is not affine equivariant as the median depends on the linear transformation of the data, but is robust being able to handle up to 40% contamination in the data.

After computing the initial basic subset with either of these methods, the BACON algorithm computes for all observations their distances (discrepancies) respectively from this subset characterized by the pair $(\mathbf{T}_b, \mathbf{S}_b)$ where $\mathbf{T}_b$ is its mean and $\mathbf{S}_b$ its covariance matrix

$$d_i = \sqrt{(\mathbf{x}_i - \mathbf{T}_b)' \cdot \mathbf{S}_b^{-1} \cdot (\mathbf{x}_i - \mathbf{T}_b)} \quad i = 1, ..., n$$

The observations whose discrepancies satisfy $d_i < c_{npr} \cdot \chi^2_{p,\frac{\alpha}{n}}$ form a new basic subject based on which the distances of all the observations are again computed and tested by the latter inequality leading to a more concentrated subset. This iteration process stops when the subset no longer changes.

Once the iteration process is terminated the objects not located in the returned subset are declared as outliers.

Suppose that r is the number of objects in the i_{th} iteration. c_{npr}, plays the role of a correction factor defined by $c_{npr} = c_{np} + c_{hr}$ where

$$c_{hr} = max\{0, \frac{h-r}{h+r}\}, \quad h = \frac{n+p+1}{2}$$

$$c_{np} = 1 + \frac{p+1}{n-p} + \frac{1}{n-h-p}$$

Based on its definition, c_{hr} is considered to be an inflation factor in the case of a significantly lower value of r than the value of h

7.4 Skewness-adjusted Outlyingness

The robust distances of the observations in a data set based on the MCD estimator provide efficient conclusions about the outlying observations when the uncontaminated objects are generated from the multivariate Gaussian distribution. However, the method may be inefficient when it deals with data sets of high skewness. Brys et al. [9] presented a modification of the outlyingness measure that can handle skewness (We note that this method does not need an distributional assumption, but we listed it in this section for illustrating the OutliersO3 package).

The Stahel–Donoho outlyingness measure [53, 13] is defined as:

$$SD_i = \max_{\mathbf{v} \in H} \frac{|\mathbf{x}_i' \cdot \mathbf{v} - med_j(\mathbf{x}_j' \cdot \mathbf{v})|}{mad_k(\mathbf{x}_k' \cdot \mathbf{v})}$$

where $mad_k(\mathbf{x}_k' \cdot \mathbf{v})$ is the the median absolute deviation defined as

$$mad_k(\mathbf{x}_k' \cdot \mathbf{v}) = med_k|\mathbf{x}_k' \cdot \mathbf{v} - med_j(\mathbf{x}_j' \cdot \mathbf{v})|$$

We denote as H the space of the unit length vectors $\mathbf{v}$ (directions) that are perpendicular to the hyperplanes that are spanned by an arbitrary number of points in the data set. Like the robust distances this measure is expressed as a rejection rule for all the objects whose SD exceeds the value of $\sqrt{\chi^2_{p,0.975}}$.

The skewness-adjusted outlyingness measure [9] is defined as

$$AO_i = \max_{\mathbf{v} \in H} \frac{|\mathbf{x}_i' \cdot \mathbf{v} - med_j(\mathbf{x}_j' \cdot \mathbf{v})|}{\big(c_2(\mathbf{v}) - med_j(\mathbf{x}_j' \cdot \mathbf{v})\big) \cdot \mathbb{1}_{\mathbf{x}_i' \cdot \mathbf{v} > med_j(\mathbf{x}_i' \cdot \mathbf{v})} + \big(med_j(\mathbf{x}_j' \cdot \mathbf{v}) - c_1(\mathbf{v})\big) \cdot \mathbb{1}_{\mathbf{x}_i' \cdot \mathbf{v} < med_j(\mathbf{x}_j' \cdot \mathbf{v})}}$$

where $c_1(\mathbf{v})$ and $c_2(\mathbf{v})$ are the end points of the skewness adjusted boxplot [61] of the projected data. It is based on the classical boxplot (discussed in Chapter 2) with the only difference that it takes into account the skewness of the data.

It is defined as

$$[c_1, c_2] = \begin{cases} [\mathcal{Q}_1 - 1.5 \cdot e^{-3.5 \cdot MC} \cdot \mathcal{IQR}, \mathcal{Q}_3 + 1.5 \cdot e^{4 \cdot MC} \cdot \mathcal{IQR}] & \text{, if} \quad MC \geq 0 \\ [\mathcal{Q}_1 - 1.5 \cdot e^{-4 \cdot MC} \cdot \mathcal{IQR}, \mathcal{Q}_3 + 1.5 \cdot e^{3.5 \cdot MC} \cdot \mathcal{IQR}] & \text{, if} \quad MC \leq 0 \end{cases}$$

where MC is a robust measure of skewness called medcouple [10] and is defined as

$$MC(\mathbf{x}_j) = \underset{x_i \leq med_j(\mathbf{x}_j) \leq x_k}{med} \frac{\big(x_k - med_j(\mathbf{x}_j)\big) - \big(med_j(\mathbf{x}_j) - x_i\big)}{x_k - x_i}$$

Remark. When the data is symmetric we obtain $MC = 0$ in every direction and the modified boxplot is equivalent to the classical. However, when the data is right skewed then $MC > 0$ enabling c_1 and c_2 to exceed the endpoints of the classical boxplot (specially for c_2). In contrast, when the data is left skewed, $MC < 0$ resulting again in a wider space [9].

Like the Stahel-Donoho outlyingness measure, its modification measures distances between the projected data and their median with the difference that AO_i takes into account the skewness of the projected data. Moreover, since it is fatal computationally to project all observations on every univariate direction $\mathbf{v}$ the method generates 250 of them since it has been proved that exceeding this number does not have significantly better results.

The cutoff value of the skewness-adjusted outlyingness measure in most cases is the upper bound of the modified boxplot e.g. $cutoff = 1.5 \cdot e^{4 \cdot MC} \cdot \mathcal{IQR}$ where right skewness is taken into consideration. We recall that the squared SD_i's are approximately χ^2 distributed and χ^2 considered to be a right skewed distribution. Although for the SD_i's there is no assumption about the distribution it is typically right skewed.

Once the AO_i's are computed, the method denotes as outliers the observations whose AO exceeds the above cutoff. Like the MCD method, the aforementioned method is affine equivariant [9].

7.5 FastPCS algorithm

The Projection Congruent Subset (PCS) method was proposed by Vakili et al. [60] and like the FastMCD and AO provides an outlyingness index measuring how much each observation departs from the pattern set by the majority of the data. It adopts some steps of the FastMCD method as it strives to select among many possible H-subsets of observations one devoid of outliers. Then, the outlyingness index is simply the distance of each observation to this subset. The main difference from the FastMCD method is the choice of the H subset rendering it less sensitive to outliers. More specifically, in case of fastMCD, the presence of a large enough number of outliers can prevent the method from finding an efficient H subset of normal observations for the estimation of location and scatter.

PCS looks for the H-subset of multivariate observations that is most congruent along many univariate projections. For that purpose, FastPCS starts with m random choices of subsets consisting of p observations, $\mathbf{x}_i = (x_{i1}, ..., x_{ip})$. It then considers K random directions α_{mk} such that $\{\mathbf{x} \mid \mathbf{x}_i' \cdot \alpha_{mk} = 1\}$ is a hyperplane for $k = 1, 2, ..., K$.

In addition, it computes the squared orthogonal distances from each object to α_{mk} defined by

$$d^2_{p,i}(\alpha_{mk}) = \frac{(\mathbf{x}_i' \cdot \alpha_{mk} - 1)^2}{||\alpha_{mk}||^2} \quad i = p+1, ..., n$$

It denotes the set of h observations with the lowest squared distances denoted as H_{mk}.

Furthermore, it follows an iterative process similar to the one we discussed in the FastMCD method for the computation of an optimal subset H in which the algorithm:

- commences with M subsets H^0_m of $p+1$ observations and for each of them it computes the following measure among K directions α_{mk}

$$D_i(H^0_m) = \underset{k=1}{\overset{K}{ave}} \frac{d^2_{p,i}(\alpha_{mk})}{\underset{j \in H^0_m}{ave}\, d^2_{p,j}(\alpha_{mk})} \quad i = 1, ..., n$$

- Then for a given point q [60] it computes H^1_m, a set of points that satisfy

$$D_i(H_m) \leq D_q(H_m)$$

- It continues until convergence. Empirically, after three steps H^3_m consist of $[\frac{n+p+1}{2}]$ objects.

- Returns the optimal H_m

Once the H_m subset is given, the method proceeds to the computation of the incongruence index of H_m along a_{mk} defined as

$$I(H_m, \alpha_{mk}) = \log \frac{\underset{i \in H_m}{ave}\, d_i^2(\alpha_{mk})}{\underset{i \in H_{mk}}{ave}\, d_i^2(\alpha_{mk})}$$

The value of this index is always positive and measures the relation of the subsets. Smaller values indicate more exact overlapping between the projection of the objects of H_m along a_{mk} with the objects of H_{mk}.

The key point that makes this algorithm more insensitive to the presence of many outliers lies in the observation that a disjoint form of H_{mk} tends to be less congruent with an optimal H_m than a cohesive one. It has been proven [60] that a cohesive subset H_{mk} when joined with the optimal H_m produces a set $H_{mk} \cup H_m$ bigger than the one that would have been produced with a disjoint one. That results in a denominator in $I(H_m, a_{mk})$ bigger in the latter case while the numerator remains the same in both cases ending up with a higher value of $I(H_m, \alpha_{mk})$ in case of a disjoint or contaminated subset [60].

Finally, the algorithm takes into consideration all the K projections removing the dependence among $I(H_m, \alpha_{mk})$ by taking the average among them

$$I(H_m) = ave_{k=1}^{K} I(H_m, \alpha_{mk})$$

Among the M different values of $I(H_m)$ that are computed FastPCS chooses the smallest. Identically to the FastMCD method, the pair of $(\mathbf{T}_{PCS}, \mathbf{S}_{PCS})$ is computed which gives to the user the final outlyingness index of each observation defined by

$$d^2_{PCS}(i) = \sqrt{(\mathbf{x}_i - \mathbf{T}_{SPC})' \cdot S_{PCS}^{-1} \cdot (\mathbf{x}_i - \mathbf{T}_{PCS})} \quad i = 1, ..., n$$

Any observation whose d^2_{PCS} exceeds the $cutoff = \sqrt{\chi^2_{p,0.975}}$ is considered to be an outlier.

We note that the initial number M of the H_m subsets must be large enough in order to find an uncontaminated subset. As for the number of projections, it has been proven that no more than 25 are needed for efficient results due to the more concentrated work on the computation of the H_m subset, while for the Skewness -adjusted Outlyingness 250 are generated. Each computation of H_m has a complexity of $O(p^3 + n \cdot p)$ [60].

Due to the fact that FastPCS in order to run, needs an input data set of $25 \cdot p$ objects we will not be able to run it for the Wood specific gravity data to illustrate the O3plot.

7.6 HDoutliers algorithm

One could say that the HD algorithm presented by Wilkinson [64] is a mixture of methods. It is able to handle high dimensional data of categorical and continuous variables. HDoutliers differs from the previously mentioned methods because it is based on the exponential distribution. The basic idea for the distributional assumption lies in the following theorem [63]

Theorem 7.2. *Let $\{\mathbf{x}_i\}$, for $i = 1, ..., n$, be a sample of independent identically distributed random variables. If the upper order statistics are defined as*

$$\mathbf{x}_{1:n} \quad \textit{is the largest element of } \{\mathbf{x}_i\}$$

$$\mathbf{x}_{2:n} \quad \textit{is the second largest element and so on..}$$

and their successive differences

$$D_i = \mathbf{x}_{i:n} - \mathbf{x}_{i+1:n}$$

then if the extreme value index is 0, then the differences D_i are asymptotically independent and exponentially distributed with means proportional to $1/i$, for $i = 1, 2, ...$ as long as $i \ll n$

We note that, the extreme value index is a parameter that controls shape in the generalized extreme value family defined by

$$G(z) = \exp\left[-\left(1 + \xi \cdot \frac{z-\mu}{\sigma}\right)^{-\frac{1}{\xi}} \right], \quad 1 + \xi \cdot \frac{z-\mu}{\sigma} \geq 0$$

where μ and σ control location and scale respectively [50]. So basically, HDoutliers after partitioning the data into exemplars [64], takes from each exemplar its representative observations fitting to them an exponential distribution.

More specifically the algorithm commences with normalizing the data if the variables are not on a common scale (we omit its transformation step to the data set in case of the presence of categorical values and its reduction step when a data consist of more that 10000 observations-briefly it uses random projections reducing the dimensionality to a default p').

Next, it creates a list of exemplars by adding the first row of the data matrix declaring it to be its center. For all the remaining rows their distances from the first row are computed. If a distance exceeds $\delta = \frac{0.1}{(\log n)^{\frac{1}{p}}}$ then the observation responsible is added to the exemplar list representing its center. If not, it is added to the exemplar centered in the first row. The value of δ is designed to be below the expected value of the distances between $\frac{n\cdot(n-1)}{2}$ pairs of the normalized points. In that way the data set is divided into balls of the same radius and centered on actual points. Furthermore, it computes all nearest neighbor distances between objects in each exemplar and fits an exponential distribution to their upper tail.

Any exemplar that is significantly far from the other exemplars based on the upper $1-\alpha$ (default is 0.05) point of the fitted cumulative distribution function is considered to be outlying. We note that an exemplar can be a single observation or a bunch of observations.

7.7 DDC algorithm

This particular method is quite different from any other method that we have discussed so far. That is because it does not handle outliers in a $n \times d$ data matrix as d-dimensional observations but as single deviating data cells. Generally, outlying methods detect cases that do not belong in the dataset, because they are members of a different population, handling them as points in d-dimensional space making them in that way indivisible (and not looking in each dimension seperately). In [44] Rousseeuw et al. consider each case as a row in the $n \times d$ data matrix that can be divided into cells. Matrix-wise, the motivation for this method was the simple assumption that some rows may be outlying due to some of their columns' values while other may be normal. It has been proved [3] that given a fraction ϵ of contaminated cells at random positions, the expected fraction of contaminated rows is

$$1 - (1 - \epsilon)^d$$

which quickly exceeds 50% for increasing ϵ and d. Many methods under that percentage of contamination are unable to reach sufficient conclusions about the outlying objects.

The DDC algorithm runs with any underlying distribution for the data set. However, it works ideally under the assumption that the $\mathbf{x}_i$ rows of the underlying data set are generated from the multivariate Gaussian with unknown d-variate mean μ and positive semi definite covariance matrix Σ.

Suppose a $n \times d$ data matrix $\mathbf{X}$.

The DDC algorithm firstly standardizes $\mathbf{X}$ to $\mathbf{Z}$

$$z_{ij} = \frac{x_{ij} - m_j}{s_j} \tag{4}$$

where m_j is a robust estimator of location for each column j and is computed as follows

$$m_j = robLoc(\mathbf{x}_j) = \frac{\sum_i^n w_{i,j} \cdot x_{ij}}{\sum_{i=1}^n w_{i,j}}$$

where the weights are given by

$$w_{i,j} = W\left(\frac{x_{ij} - med_j(x_{ij})}{med_j|x_{ij} - med_j(x_{ij})|}\right) \quad \text{and} \quad W(t) = \left(1 - \left(\frac{t}{c}\right)^2\right)^2, \quad |t| \leq c$$

where c is a positive constant (default value is 3).

Furthermore, s_j is a robust estimator of scale for each column j and is computed as follows

$$s_j = robScale(\mathbf{x}_j) = med_j|x_{ij} - m_j| \cdot \sqrt{\frac{1}{\delta} \cdot ave_i^n \rho\left(\frac{x_{ij} - m_j}{med_j|x_{ij} - m_j|}\right)}$$

where $\rho(t) = min(t^2, b^2)$, $b^2 = 2.5$ and $\delta = 0.845$ ensuring consistency for Gaussian data.

We note that this method of estimation constitutes the first step of the M-estimators algorithm [44, 37].

After the columnwise standarization in (1), the DDC algorithm creates a new matrix denoted as U defined as

$$u_{ij} = \begin{cases} z_{ij} & \text{, if } |z_{ij}| \leq \sqrt{\chi^2_{1,p}} \\ \text{NA} & \text{, otherwise} \end{cases} \tag{5}$$

where $\chi^2_{1,p}$ is the p_{th} quantile of the χ^2 distribution with 1 degree of freedom. Basically U matrix is an outlier detector in the univariate case of each column flagging the values that exceed a certain cut off value.

In addition, the DDC algorithm takes into consideration the bivariate correlations between variables for each observation.

For columns j, h with $i \neq h$ DDC employs a robust correlation measure dented as $cor_j h$, that is the plain product-moment correlation of the data points (u_{ij}, u_{ih}) inside the ellipse around (0,0) with coverage probability p as in (2) implied by the initial correlation estimate

$$\hat{\rho}_{jh} = \frac{\Bigg(robScale(u_{ij} + u_{ih})\Bigg)^2 - \Bigg(robScale(u_{ij} - u_{ih})\Bigg)^2}{4} \tag{6}$$

DDC algorithm uses only the pairs (j, h) that satisfy $cor_{jh} \geq 0.5$ computing a robust slope b_{jh} of a robust regression line without intercept that predicts variable j from variable h. That line is computed from the points that satisfy $|r_{ihj}| \leq \chi^2_{1,p} \cdot m^n_i(r_{ijh})$ where $r_{ihj} = u_{ij} - b_{jh} \cdot u_{ih}$ for $i = 1, ..., n$. DDC denotes λ_{jh} as the initial slope of the (j, h) pair defined as $\lambda_{jh} = med_i\Bigg(\frac{u_{ij}}{u_{ih}}\Bigg)$.

Furthermore, DDC algorithm computes the predicted values of $\hat{u}_{ij}$ considering only the variables that their correlation measure with the j_{th} is above 0.5 (constructing along with h an index space denoted as H_j) as follows

$$\hat{u}_{ij} = G\Bigg(\{b_{jh} \cdot u_{ij} \mid h \in H_j\}\Bigg) \tag{7}$$

where G is a combination rule that handles the NA values.

Finally, DDC algorithm computes the standardized cell residuals, defined as

$$r_{ij} = \frac{u_{ij} - \hat{u}_{ij}}{m^n_i(\hat{u}_{ij} - u_{ij})} \tag{8}$$

and flags in each column j the cells that satisfy $|r_{ij}| \geq \sqrt{\chi^2_{1,p}}$

7.8 Applying outlier detection using distribution-based methods to wood specific gravity data

In this section we will illustrate the plot O3. Plot 03 was designed for outlier detection. What makes it so intuitive is the fact that it is able to detect outliers in lower dimensions of the underlying data set through varying combinations of the respective variables. Unwin [58] originally presented this plot concerning outlier detection. The O3 plots are able to detect outliers employing the six different methods that were presented above.

We illustrate the Plot O3, with the use of the well known by now Wood specific gravity data.

We employ the following functions from the R package **Outliers03**

- **O3prep**: Identifies outliers for the variable combinations and tolerance levels specified. It is used as an input for the following functions.

- **O3plotT**: Draws O3 plots for one method and up to three tolerance levels and supporting parallel coordinate plots.

- **O3plotM**: Draws O3 plots for more than one methods enabling the comparison between them and supporting parallel coordinate plots.

We begin our analysis testing the FastMCD method to the data inputting 0.5 as a tolerance value

```
> MCD <- O3prep(my_data, k1=1, method=c("MCD"),tols = 0.05)
> MCD_plot <- O3plotT(MCD)
> MCD_plot$gO3
```

The O3 Plot is divided into two blocks by the two white columns in the middle. In the left block each column represents a different variable and each row a different combination of them. Gray cells show which variable participates in each combination while the blue dashed lines separate each group of different number of variables participating in the combination. The variables are sorted by their frequency of participation in the combinations. In the right block each column represents an observation. Red cells in each row indicate the outlying objects flagged by the underlying method for the combination of the variables in that row.

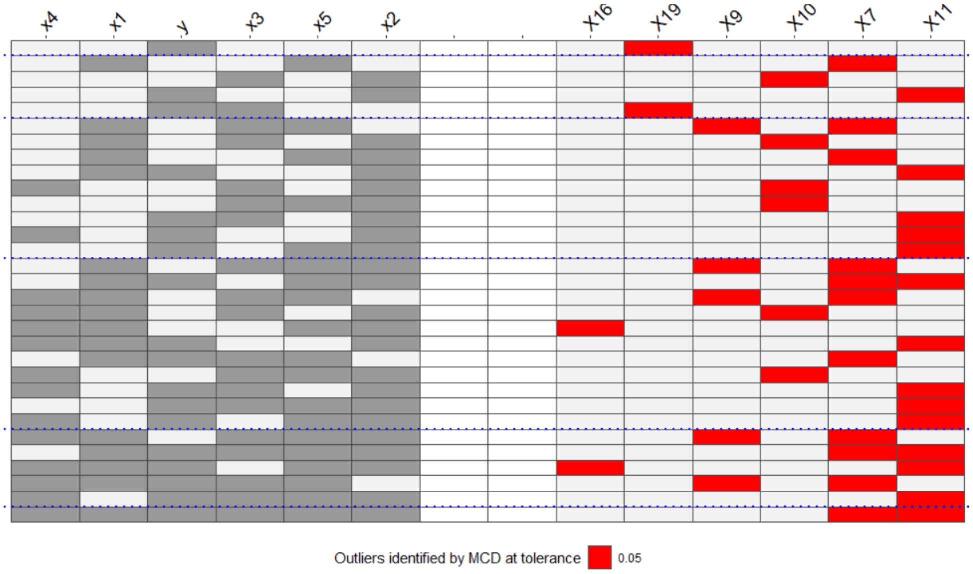


Figure 11: O3 plot from the FastMCD algorithm

As we can see from Figure 11, the MCD method with a tolerance level of 0.05 detects six observations outlying. We note that the 11th observation was flagged by the most combinations of variables along with the 7th. Also these were the only objects detected to be outlying when all of the variables were combined. The MCD method seems to be affected by a masking effect maybe due to the low number of observations in the data set.

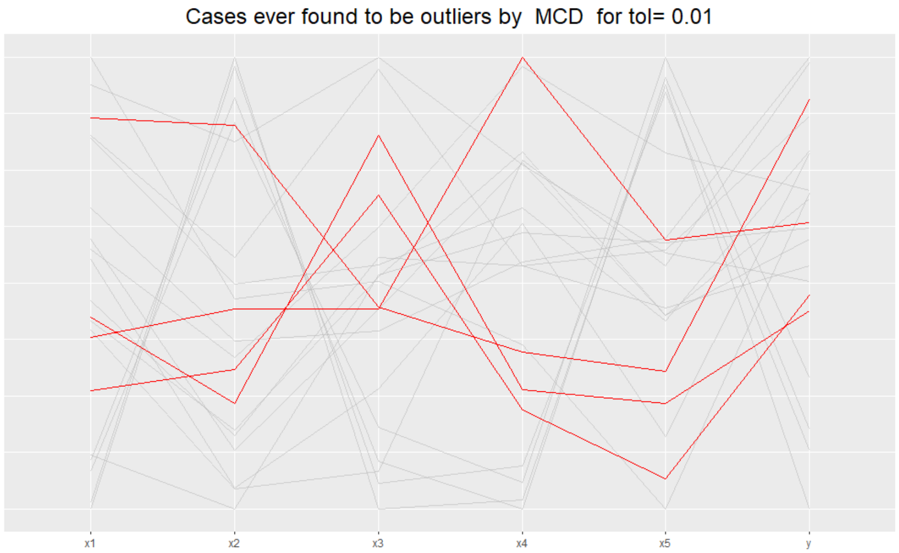


Figure 12: A parallel coordinate plot of all the data. The cases ever found to be outliers are coloured red

Relaxing the tolerance FastMCD detects three more objects as outliers (shown in Figure 13) although only in univariate and bivariate combinations. In contrast, a stricter tolerance level of value equal to 0.01 leads to the detection of four outliers, 16,9,7 and 11. In figure 12 we see these objects through a parallel coordinate plot.

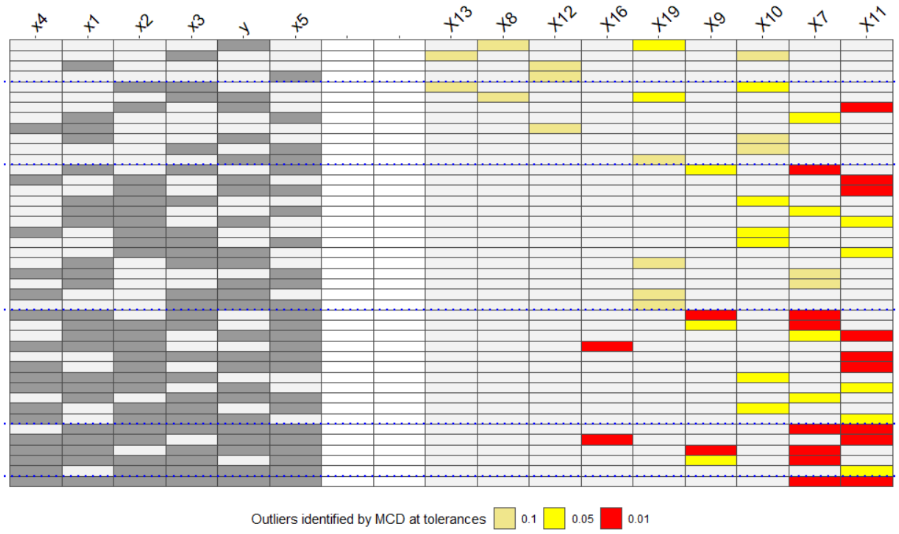


Figure 13: O3 plot from the FastMCD algorithm for different tolerance values

We continue our analysis by giving the code for the BACON algorithm

```
>BAC <- O3prep(my_data, k1=1, method=c("BAC"), tols = c(0.1,0.05,0.01))
>BAC_plot <- O3plotT(bac)
>BAC_plot$gO3
```

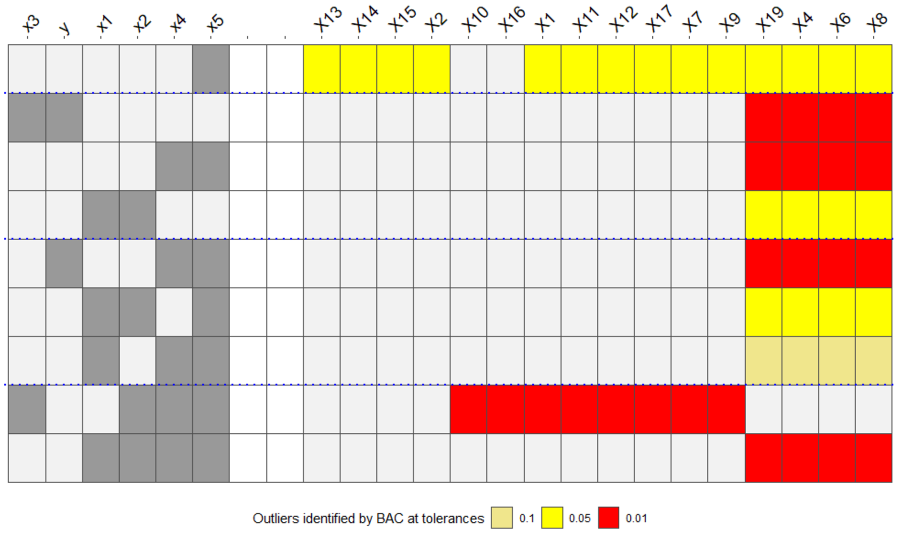


Figure 14: O3 plot from the BACON algorithm for different tolerance values

In Figure 14 the BACON method is illustrated. From the 03 plot we see that the algorithm detected in almost all the combinations the true outliers. We observe, though, that it did not flag any object when all variables were combined. As we see in Figure 15, omitting variables x3 and y from the data set, observations 4, 6, 8, 19 yield very low values in variables x1 and x4 and very high values in x2 and x5. Furthermore, an increase of the tolerance value led to a significant increase of the outlying objects in the univariate case of variable x5. Finally we note that the BACON algorithm was indeed successful in finding the correct outliers in the underlying data set of 20% contamination [7, 11].

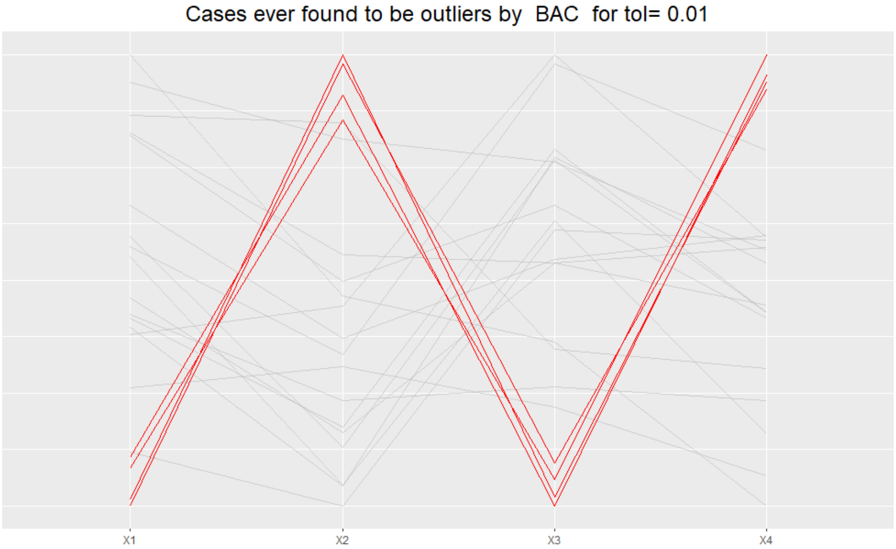


Figure 15: A parallel coordinate plot of the data considering only variables x1, x2, x4, x5

The code for the skewness-adjusted outlyingness measure is given below

```
> adjout <-  O3prep(my_data, k1=1, method=c("adjOut"), tols = c(0.1,0.05,0.01))
> adjout_plot <- O3plotT(adjout)
>adjout_plot$gO3
```

The tolerance level in this case tends to be less sensitive in this method as we observe in Figure 16 that it identifies only three observations that have been also declared in the FastMCD method. Although it did not give proper results it tends to give fewer outliers than the FastMCD.

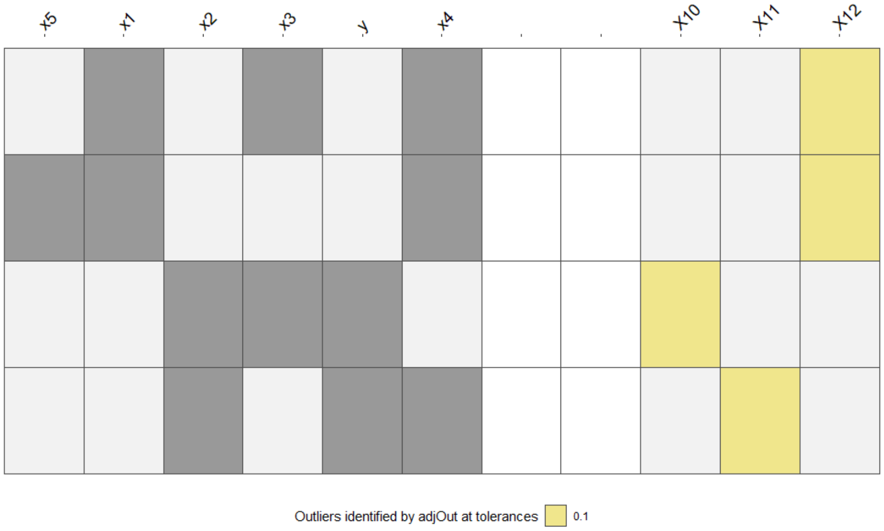


Figure 16: O3 plot from the AO measure for different tolerance values

We examine the results of the HDoutliers algorithm by the following commands

```
> HDo <-  O3prep(my_data, k1=1, method=c("HDo"), tols = c(0.1,0.05,0.01))
 > HDo_plot <- O3plotT(HDo)
 >HDo_plot$gO3
```

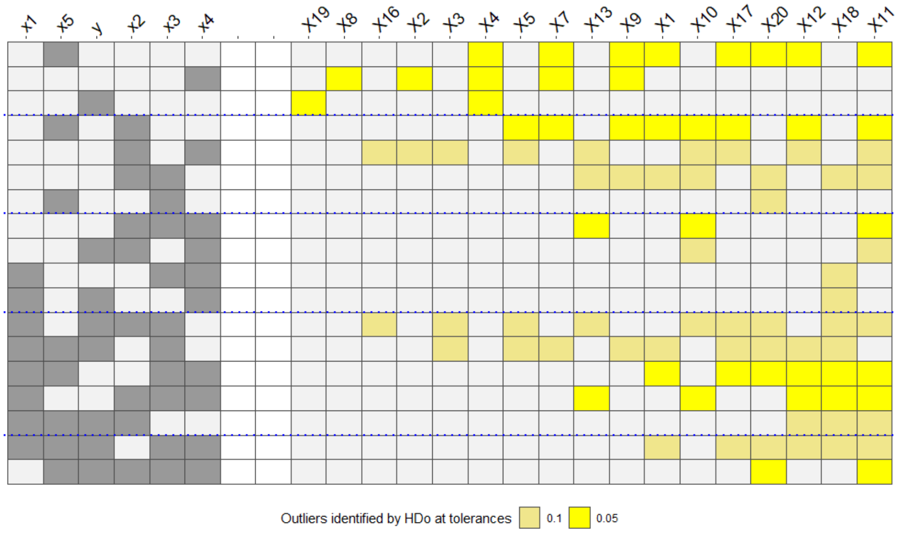


Figure 17: O3 plot from the HDoutliers algorithm for different tolerance values

As we see in Figure 17, HDoutliers is more sensitive than the aforementioned alogirithms cellwise declaring observation 7 as an outlier in the most variable combinations. Like BACON and AO, it fails to detect outliers when all variables are combined .

Finally we present the code for the DDC algorithm below

```
 > DDC <-  O3prep(my_data, k1=1, method=c("DDC"), tols = c(0.1,0.05,0.01))
> DDC_plot <- O3plotT(DDC)
>DDC_plot$gO3
```

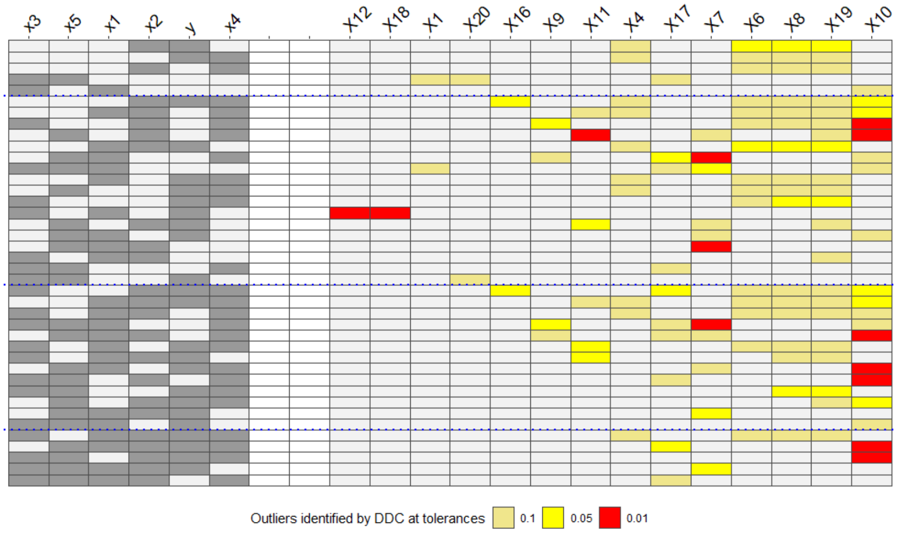


Figure 18: O3 plot from the DDC algorithm for different tolerance values

In Figure 18 we observe efficient results as the DDC algorithm was able to detect the true outlying objects. More specifically observations 4, 6, 8, 19 are the only ones that were detected in the combination of variables x1, x2, x3, x4, y. However the observation found as an outlier in most cases was the 10th. Finally we observe that relaxing the tolerance value leads to more outlying objects in lower dimensions and less in fewer dimensions.

7.9 Conclusions

Distribution-based methods have the advantage that they respond well to exact fit situations, that is datasest that are generated from the multivariate normal distribution. Moreover, most of the methods we discussed are robust so they can handle heavily contaminated data sets. Plot03 unifies these methods and modifies them in order to detect outliers in every possible combination of the data variables. This new tool enables the user to see the discrepancies in lower dimensions having in that way more information about the data. This in fact is very useful when we want to look at the behaviour of a data set with many variables in a smaller subset without any reduction method. The poor results that most of the methods produced is probably due to the small number of observations in the underlying data set.

8 Humus Layer (O-horizon) of the Kola Data

The humus dataset is implemented in the **mvoutlier** package. The data originates from the Kola Project, which was a geological survey and data collection effort conducted from 1993-1998. Samples from the humus ground layer were taken from Norway, Finland, and Russia and elemental ground makeup was recorded for 617 unique observations. The humus ground layer is formed from decomposing organic litter to include leaves, roots, branches, etc [55, 41, 17]. Humus layer is a data frame with 617 observations on 44 different variables.

Before we begin our analysis we omit the first 3 variables because they represent the ID and the coordinates of each observation.We then measure the skewness of the data using the function **skewness()** from the R package **moments**.

```
Ag    6.1136720     Ni    9.3199522
Al    3.0464415     P    11.7156569
As   10.3667974     Pb   18.9472707
B     2.7078609     Rb    1.7673577
Ba    1.6708973     S     0.7695066
Be    6.3772582     Sb    1.8081578
Bi    2.7778763     Sc    2.5496159
Ca    6.9433935     Si    0.6158160
Cd    2.2540816     Sr   13.3922063
Co    8.3331150     Th    8.6649278
Cr    9.3505637     Tl    1.8492946
Cu   12.2560644     U    20.0555647
Fe    6.2160905     V     3.5954228
Hg    4.0259441     Y    14.8257164
K     5.1160085     Zn    1.7124205
La    9.8782389     C    -1.9757673
Mg    3.4694265     H    -1.8453518
Mn    9.4020969     N    -0.1644502
Mo    6.4442158     LOI  -2.0947662
Na   10.7304293     pH    1.1165646
                    Cond  0.2414859
```

As we can see from the above results, we deal with a high skewed data set that we have to log transform in order to reduce the skewness.

```
log_humus<-cbind(humus[1:3],log(humus[4:27]),humus[28],log(humus[29:30]),humus[31],
log(humus[32:38]),log(1+50.8-humus[39]),log(1+7.1-humus[40]),humus[41],
log(1+98.8-humus[42]),log(humus[43]),humus[44])
```

We note that variables like **Cond** did not need log transformation because of its very low value of skewness. We only transformed when the absolute value of skewness was above 1. Furthermore, for variables with negative skewness we log transform the reflected variables. As we can see below, the log transformation of the dataset reduced by a lot the skewness of the variables.

```
Ag       0.07686216     Ni      1.21638725
Al       0.36323030     P       1.07475529
As       1.80696684     Pb      1.73450733
B        0.67454365     Rb     -0.42419502
Ba      -0.09421180     S       0.76950659
Be       0.37050354     Sb     -1.04440349
Bi       0.23926369     Sc      0.01473079
Ca       0.01180195     Si      0.61581603
Cd       0.14538457     Sr      1.52768415
Co       0.94174909     Th      0.73218060
Cr       0.73792750     Tl      0.10688968
Cu       2.15778988     U       1.03944466
Fe       0.60438594     V       0.59391029
Hg       0.39483989     Y       0.74417940
K        0.41581384     Zn     -0.26501122
La       0.50176916     C       0.09347919
Mg       0.72110704     H       0.86359353
Mn       0.13273250     N      -0.16445024
Mo       1.65680056     LOI     0.18002052
Na      -0.25920409     pH      0.68674777
                        Cond    0.24148593
```

8.1 Outlier analysis

We begin our analysis by running all the methods that we presented in the previous chapters based on **OutlierDetection** package with their default settings. The advantage of these functions is that while they use the outlier methods they employ a cutoff produced by bootstrapping. Therefore, each observation which is denoted as an outlier, comes with a bootstrapped estimated probability of its outlyingness.

```
  > depthout_log_humus_dflt <- depthout(log_humus, rnames = FALSE, cutoff = 0.05,
  boottimes = 100)
```

As we discussed, this method based on depth, computes the Modified Band Depth of each observation. Those that have the lowest values are considered to be outliers. For that purpose we tuned a cutoff of 5%.

The results are given below

```
  depthout_log_humus_dflt$`Location of Outlier`
 [1]  28  35  60  75 133 160 173 201 232 238 256 258 275 297 301 383 390 399 409 451
 478 487 615
> depthout_log_humus_dflt$`Outlier Probability`
 [1] 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 0.99 1.00 1.00 1.00 1.00 1.00
 1.00 0.99 1.00 1.00 1.00 1.00 1.00
```

We find that 23 observations have the least depth in the data set and they are lying in the shallowest layers of the data set.

We now present the distance based functions **nn** and **nnk** and their results

```
> nn_log_humus_dflt <- nn(log_humus, k = dflt, cutoff = 0.95,   Method = "euclidean",
  rnames = FALSE, boottimes = 100)
> nn_log_humus_dflt$`Location of Outlier`
 [1] 17  22  27  28  34  54  75  79 106 125 133 160 168 232 235 258 266 390 449 569
  612 615
> nn_log_humus_dflt$`Outlier Probability`
 [1] 1.00 0.99 1.00 1.00 1.00 0.99 1.00 0.99 1.00 1.00 1.00 1.00 1.00
 1.00 1.00 1.00 1.00 1.00 1.00 1.00 0.99 1.00
>
 > nnk_log_humus_dflt <-nnk(log_humus, k = dflt, cutoff = 0.95, Method = "euclidean",
 rnames = FALSE, boottimes = 100)
> nnk_log_humus_dflt$`Location of Outlier`
 [1]  17  22  27  28  75  79 106 125 133 160 161 168 232 235 240 258 266 358 390 449
  569 612 615
```

```
> nnk_log_humus_dflt$`Outlier Probability`
 [1] 1.00 0.95 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 0.99 1.00
 1.00 1.00 1.00 1.00 1.00 0.99 1.00 1.00 1.00 0.99 1.00
```

In these methods we use for k nearest neighbors the default setting which equals 30 as it is the integer part of the product 0.05*nrow(log_humus). We note that the cutoff in this case is 0.95 because the methods declare outliers with the highest distances exceeding this value. We recall that **nn** computes the average of the k nearest neighbor distances while **nnk** computes the actual k nearest neighbors distances. **nnk** detects a total of 23 observations as outliers out of which 8 points were also detected in the depth-based method. **nn** method returns almost identical results to **nnk** due to the large number of observations in the dataset.

In the context of distance we also present Hautamaki's method concerning the in-degree number of each observations. We recall that lower in-degree indicates higher outlying probability.

We present the 24 lower in-degree values with $T = 8$ as a cutoff

```
> which(8>=KNN_IN(log_humus, k= dflt))
>order(KNN_IN(log_humus, k= dflt))[1:24]
  77 615  28  34  75 125 168 355 449 258 385 569  68 232 266 311 399 582
  27  41  66 160 201 515
```

We find that 8 observations match with the results of **depthout** while 12 match those of **nnk**'s.

Continuing our analysis we perform the function that detects outlying objects taking into account their RKOF values. We recall that higher RKOF values express higher probability of the corresponding observations to be outliers.

The code along with the results are below

```
>dflt=0.05*nrow(log_humus)
> dens_log_humus_dflt <- dens(log_humus, k = dflt, C = 1, alpha = 1, sigma2 = 1,
cutoff = 0.95, rnames = F, boottimes = 100)
> dens_log_humus_dflt$`Location of Outlier`
 [1]  17  22  27  28  34  41  75  77  79 106 125 133 160 168 201 232 258
 266 390 399 449 565 569 615
> dens_log_humus_dflt$`Outlier Probability`
 [1] 1.00 1.00 1.00 1.00 1.00 0.95 1.00 1.00 1.00 1.00 1.00 0.95 1.00
 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00
```

We note that this method detects 24 ouliers, 10 of which are also detected by the depth-based method.

For the purpose of finding outliers based on local density and the number of outliers detected by the **dens** function we present the 24 highest LOF and RKOF values that were built in the **DDoutliers** package with the default settings.

```
   > order(LOF(log_humus,k=dflt))[594:617]
 [1] 41 565 133  22 399  79 106 201 160  34 390  27 232 266  17 125
 77 258 569 168  75 449  28 615
> order(RKOF(log_humus, k=dflt, C=1, alpha = 1, sigma2 = 1))[594:617]
 [1] 133  41 565 399  22  79 106 201 160 390  34  27 266  17 232 125 258
 77 569 168  75 449  28 615
```

First of all, we note that the **dens** function is consistent with the RKOF values produced by the **RKOF** function. Furthermore, compering results between LOF and RKOF there is only a small difference in the order of the outliers due to the different weighting of the two methods. The advantage of these methods is that they measure each observation's outlyingness. We also notice that observations 615 and 28 with the highest LOF and RKOF values also have the second and third lowest in-degree values, respectively showing some kind of consistency between these methods.

Consequently we expect that observations 168, 75, 449, 28, and 615 that were detected with high LOF and RKOF values and also detected in most methods are indeed outlying.

Finally, we present the outliers detected by the Mahalanobis distance. Objects whose Mahalanobis distance exceeds the cutoff $\sqrt{\chi^2_{41,0.975}} \approx 7.78$ are declared outliers.

```
   > maha_log_humus <- maha(log_humus,0.975)
> maha_log_humus$`Location of Outlier`
  9  27  28  31  34  35  53  60  61  63  65  67  70  75  87  90  97 116 125 133
  135 139 168 173 194 200 219 232 235 237 238 240 258 261 265 275 290 299 311
  332 341 352 359 362 376 378 383 390 395 399 418 420 427 441 449 459 469 478
  480 487 488 501 518 521 557 558 569 573 574 576 580 605 613 614 615
```

We note a tremendous increase in the number of detected outliers as in this case 75 observations are considered to be outlying. The underlying size increases as we can see in Figure 19 when we compute robust distances instead. That is because the ellipsoid constructed from all observations is affected by the contaminated objects. In contrast, robust distances are based on the MCD estimators of center and covariance which are computed with fewer observations which are considered to be uncontaminated, thus being able to capture all the outliers that classical Mahalanobis distance could not with the same cutoff.

Different methods have different cutoffs. For example squared Mahalanobis distances with the assumption that the objects are generated from a p-variate Gaussian distribution, are χ_p^2 distributed and thus a percentile of χ_p^2 is used as a cutoff. The previous methods do not need any distribution assumptions but their sensitivity in most cases depends on k and of course on the percentile cutoff that we choose to use.

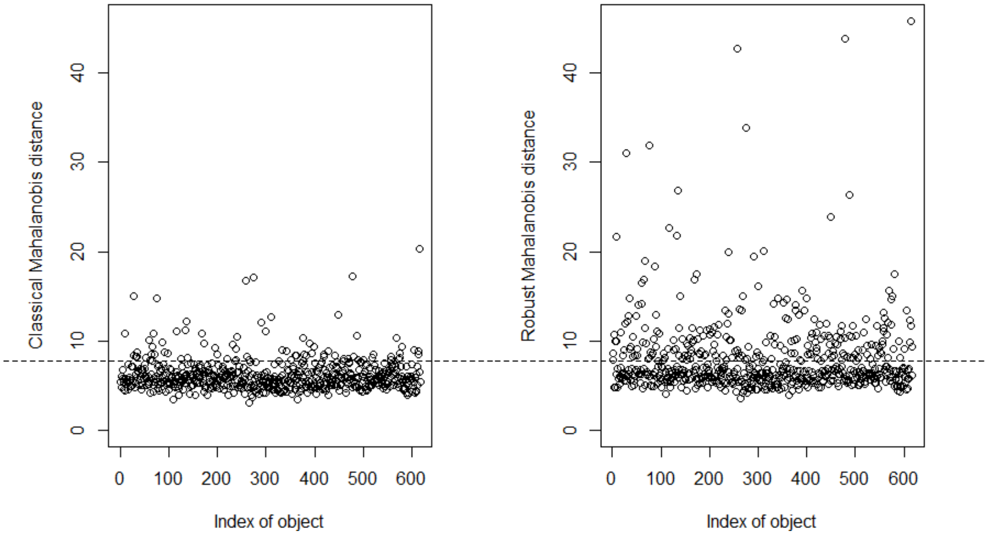


Figure 19: Mahalanobis distances versus the object number. On the left, distances are computed using classical estimates for for location and covariance. On the right, they are computed based on the MCD estimator

Number of outliers			
dens	**nnk**	k	Percentage
25	23	5	
26	24	10	
24	25	15	0.95
26	22	20	
23	21	25	
83	83	5	
85	83	10	
88	84	15	0.85
87	84	20	
84	85	25	

Table 6: Number of detected outliers for different k and percentages for **den** and **nnk**

	k	Observations
KNN_IN	5	**T=0**: 34 41 77 126 171 224 232 237 311 355 428 458 605 **T=1**:12 25 61 66 97 135 140 166 201 211 243 258 262 299 326 379 381 393 399 406 452 466 491 519 532 535 546 565 580 582 610 614 615
	10	**T=0**: 77 311 355 **T=1**: 34 126 582 605 615 **T=2**: 41 66 68 111 139 171 201 232 258 379 428
	15	**T=0**: 311 355 **T=1**: 77 **T=2**: 34 68 139 201 258 379 615
	20	**T=0**:0 **T=1**:77 311 355 **T=2**: 34 68 615
	25	**T=0**:0 **T=1**: 0 **T=2**:77 355 615

Table 7: Number of detected outliers by KNN_IN for different cutoffs T

In Table 7 we observe that increasing the number of k we detect fewer outliers with these indegrees values. For example when $k = 25$ we detect only 77, 355, 615 with indegree value equal to 2. That means that in neighborhoods of 25 points three observations participate in only two.

In Table 8 we observe different LOF and RKOF values for different k. When applying these methods there is no clear answer about the optimal choice of k. Bigger k indicates bigger scale for the neighborhood. Increasing k we detect a weak convergence of the methods. For example for $k = 15, 20, 25$ observations 75, 449, 28, 615 based on their LOF values are outlying with the same ascending order. In case of RKOF we detect a slightly weaker convergence as we detect for the same k, observations only 449, 28, 615 outlying in the same order. We speculate that the log transformation had a smoothing effect to the distances of the data objects making the neighborhoods less contaminated so that the robustness of the RKOF provided the same results as LOF.

Methods	k	Observations
LOF	5	515 61 399 458 171 610 54 385 126 582 66 311 201 258 34 390 355 41 77 28 232 615 569 75 449
	10	61 554 582 458 201 390 311 355 17 27 385 34 160 41 258 266 125 232 168 569 77 75 449 28 615
	15	22 311 355 399 79 41 201 106 385 27 160 390 34 17 266 232 125 258 168 77 569 75 449 28 615
	20	117 565 41 22 399 385 79 106 201 160 17 390 34 27 266 232 125 77 258 168 569 75 449 28 615
	25	117 41 385 565 22 399 79 106 201 27 34 160 390 232 17 266 125 77 258 569 168 75 449 28 615
RKOF	5	135 160 201 54 61 399 385 610 582 28 311 171 126 66 258 34 615 77 355 390 41 569 232 75 449
	10	201 458 554 582 390 61 17 311 27 385 34 355 258 160 266 125 168 41 232 569 77 75 449 28 615
	15	22 79 399 106 311 201 355 390 41 385 17 27 160 34 266 258 125 168 232 77 569 449 75 28 615
	20	565 355 22 399 79 41 385 106 201 390 17 160 34 27 266 125 232 258 168 77 569 75 449 28 615
	25	117 399 41 22 565 385 79 106 201 390 34 160 27 17 266 232 125 258 168 77 569 75 449 28 615

Table 8: LOF and RKOF values for different k. The 25 highest values are listed.

So far we have a rough idea about the outlying objects, but due to the high dimensionality we do not have a picture of the objects. For that purpose, we will use the 03 plots that are can be very useful when we want to see the nature of the objects in smaller dimensions.

We choose randomly four variables (Al, Co, Cu, Ni) for examination. Our goal is to figure out if the previous results have any consistency in lower dimensions.

We begin our analysis by comparing **HDoutliers** and **FastPCS**. For these methods to be comparable, we choose different tolerance levels. We recall that HDoutliers labels observations as outliers if they fall in the $(1-\alpha)$ tail of the fitted exponential distribution. On the other hand the tolerance level of FastPCS equals $1-\alpha$ where α in this case is the size of the subsets used in the algorithm. The algorithm after computing an optimal subset declares as outliers all the observations whose distances exceed a χ^2_4 cutoff. For illustrative reasons we choose a low tolerance level for FastPCS while in HDoutliers we use the default setting.

```
>HPmethods  <- O3prep(sd_log_humus, method=c("HDo", "PCS"),tolHDo = 0.05,
tolPCS = 0.01)
>HPmethods_O3 <- O3plotM(HPmethods)
>HPmethods_O3$gO3
```

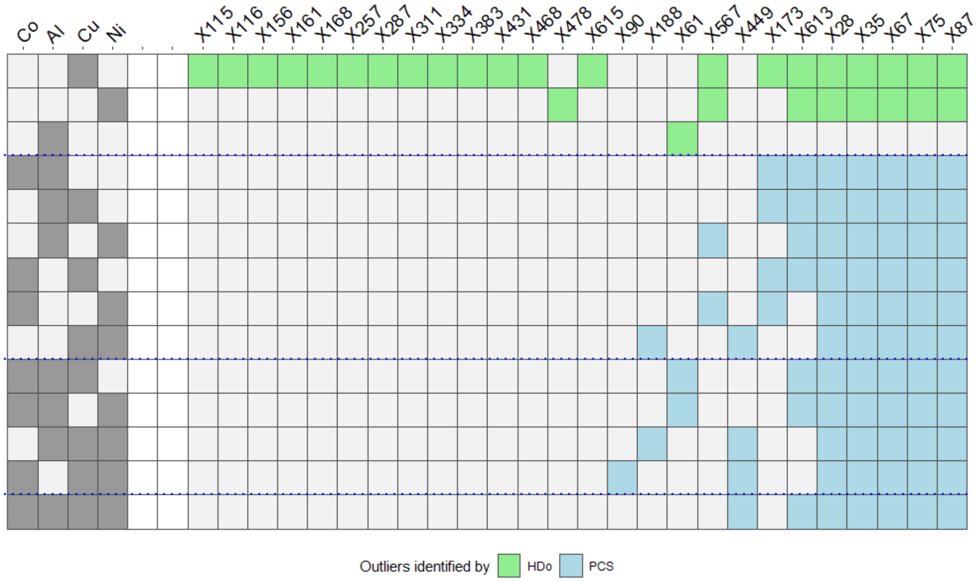


Figure 20: O3 Plot comparing HDoutliers and FastPCS

In Figure 20, we notice that for the given tolerance levels the two methods do not overlap. That is partly because FastPCS does not detect ouliers in the univariate case. On the other hand we observe that HDoutliers while it detects a lot of outlying objects (especially in variables Cu and Ni), fails to detect outlier in higher dimensions. The idea of the method is, as we discussed, to group cases that are very close together in exemplars before calculating nearest neighbor distances. If there are too many cases with the same or close values, this can result in the algorithm identifying far too many cases as outliers. For example all observations until 615 are not detected by FastPCS even though they are considered to be outliers by HDOutliers.

Finally increasing the tolerance level for the FastPCS to 0.25 (indicating initial subsets of $0.75 \cdot 617 \approx 461$ objects) the method detects 268 observations.

Next, we present the comparison of **FastMCD** and **adjOut**. The tolerance level for the first method corresponds to the quantile (default is 0.975) of χ^2_4 set by the user. In this case it is set to that small value (1-0.999) in order for the plot to be representative. As for the latter method, its tolerance method corresponds to the quantiles that we want to use in order to build the skewness adjusted boxplot.

```
> Mamethods  <- O3prep(sd_log_humus, method=c("MCD", "adjOut"),
tolMCD = 0.001,toladj = 0.2)
> Mamethods_humus_O3 <- O3plotT(Mamethods)
>Mamethods_humus_O3$gO3
```

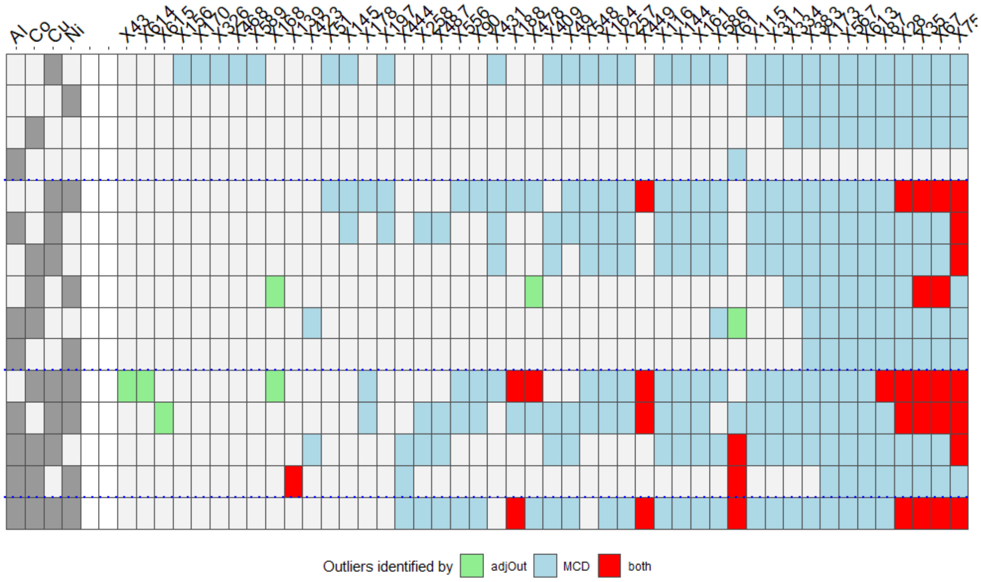


Figure 21: O3 Plot comparing FastMCD and adjOut

The FastMCD method results in many more outliers than adjOut as we notice in Figure 21. That is because of the different perspective of their cutoffs. Moreover, unlike FastMCD, adjOut does not have any distributional assumption and its cutoff takes into account the skewness of the data. We recall that we have already log transformed the data before the analysis and due to the fact that this modified boxplot is wider than the classical one, we find this insensitivity of the method reasonable. That is probably why adjOut does not detect any outliers in the univariate case. Furthermore, we notice that in most cases the results of adjOut overlap with FastMcd's. Of interest is observation

615 which is detected by adjOut only when Al, Cu, Ni are combined and based on our analysis so far it is probably outlying.

We note that most observations are detected by the combination of Co, Cu, Ni. As we see in Figure 22 outliers (labelled as red) correspond to high values in these three variables in comparison with the non outlying points.

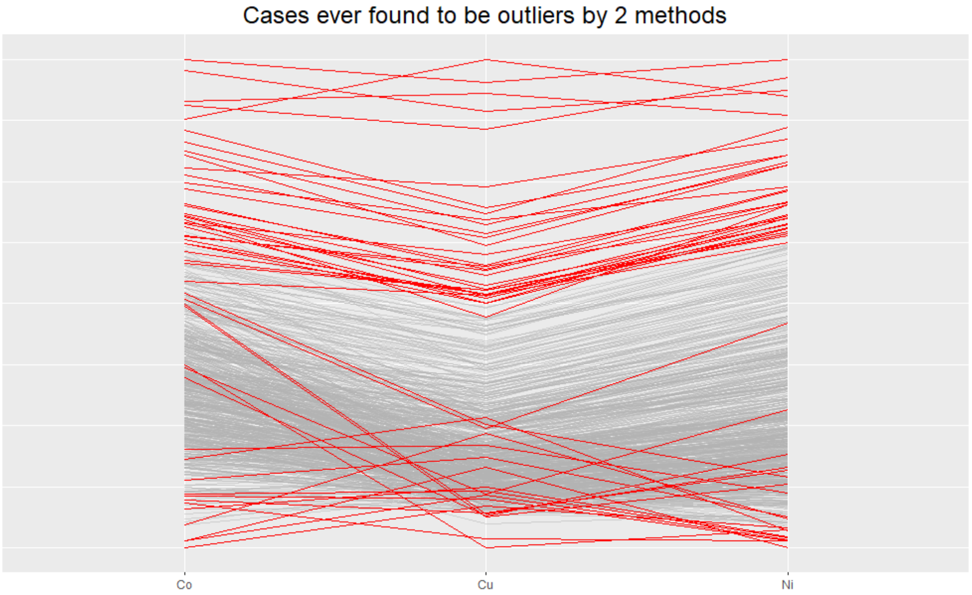


Figure 22: Parallel Coordinate Plot for FastMCD and adjOut

Finally we present the O3 plot of **DDC** and **BACON**. The tolerance level for DDC is α which indicates the $1-\alpha$ percentile of χ^2_1 (it is a cellwise detection). As for BACON we choose a tolerance level of 0.95. We recall that the cutoff value for the latter method is the value of $\chi^2_{4,\frac{0.95}{617}}$

```
> DBmethods  <- O3prep(sd_log_humus, method=c("BAC", "DDC"),tolBAC = 0.95,tolDDC = 0.03)
> DBmethods_humus_O3 <- O3plotM(DBmethods)
> DBmethods_humus_O3$gO3
```

In Figure 23 we note that like FastPCS, the DDC algorithm does not detect outliers in the univariate case. We further note that BACON like FastMCD detects a large number of outliers. However BACON is not considered to be a robust tool as its breakdown point is 20%. In addition we notice, in rows c0110 (indicating that Co and Ni are the variables combined) and c0111 an overlapping of the two methods in the most detected observations. We also note many differences in the flagged cells due to the fact that DDC takes into account the robust correlations between variables while BACON does not.

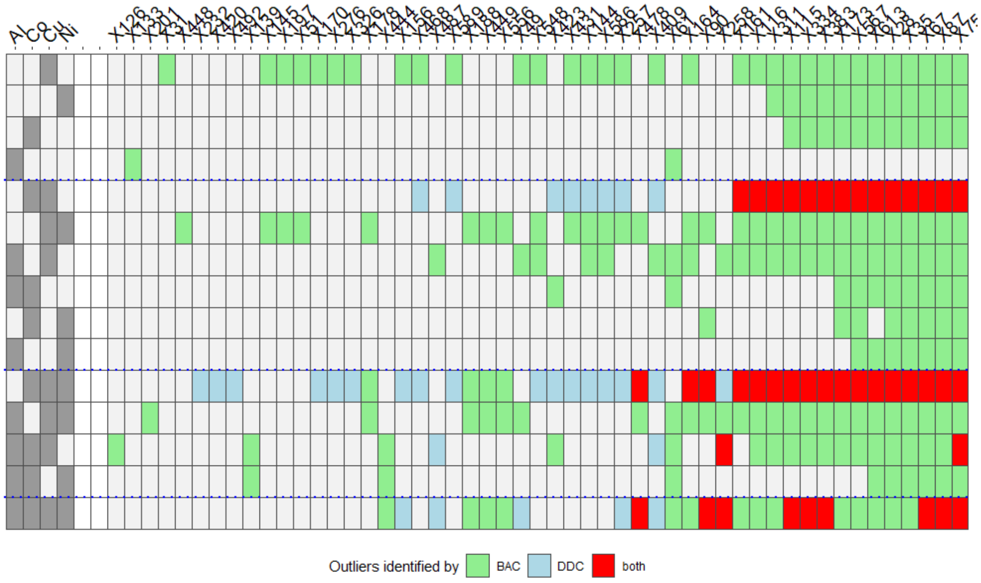


Figure 23: Plot comparing DDC and BACON

As we discussed FastMCD, BACON and FastPCS have a similar perspective on detecting outliers. Their common key idea is that all of these methods search for an “outlier free” subset and based on that flag the bigger distances. Figure 24 shows the consistency of these methods. Moreover, we observe that the BACON method detects the most cases followed by FastMCD and finally the fewest cases are detected by FastPCS.

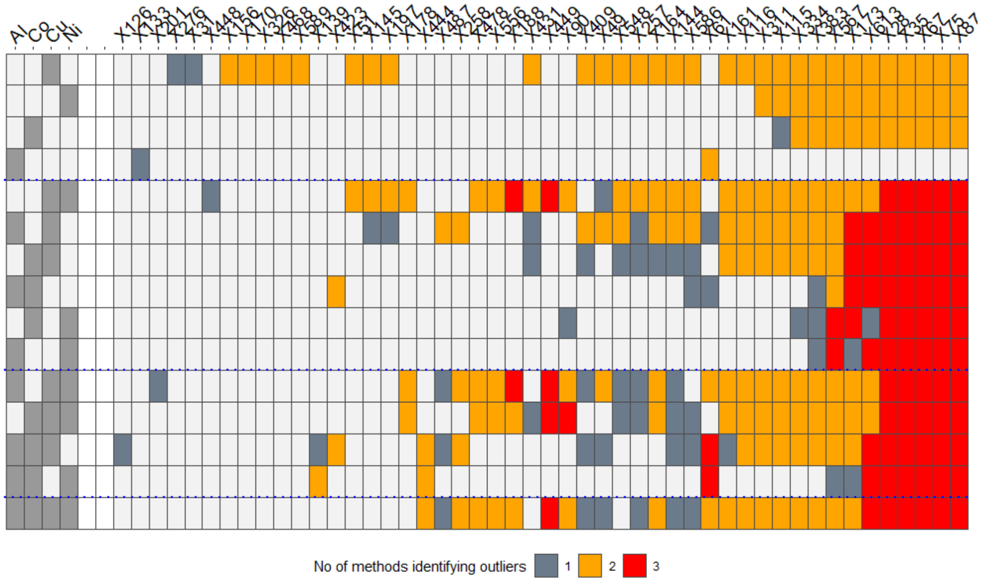


Figure 24: Plot comparing FastMCD FastPCS and BACON

We present the code for Figure 6 along with the outliers flagged by each method below

```
PBMmethods  <- O3prep(sd_log_humus, method=c("BAC", "MCD","PCS"),tolBAC = 0.95,
tolMCD = 0.001,tolPCS = 0.01)
PBMmethods_humus_O3 <- O3plotM(DBmethods)
PBMmethods_humus_O3$gO3
PBMmethods_humus_O3$nOut
    PCS BAC MCD
    12  48  42
```

Looking at the above plots we observe that the combination of variables Cu and Ni results in the most observations detected as outliers. We recall that we have already log transformed our data in the prepocessing step. Looking at Figure 25 we see a linear relationship between the log transformed variables. Moreover, we have labelled the observations that were most detected by the methods in the Outliers03 package. We observe, that observations 87, 35, 28, 67, 75 were flagged by all the methods the most times by most combinations. Scatterplot shows that these observations are indeed far away from the rest of the data. While observations 28 and 75 seem to be strong outliers when the whole dataset is tested, observations 87, 35, 67 do not share the same role. Moreover, observations like 449 and 615 based on this scatterplot do not seem to be outlying so much from the rest of the data even though they were noted by the most LOF and RKOF values when the whole dataset was tested. Therefore, it can be seen that a subset of variables can interpret the outlyingness of some objects but it also shows that possible strong outliers can be presented as normal points and vice versa.

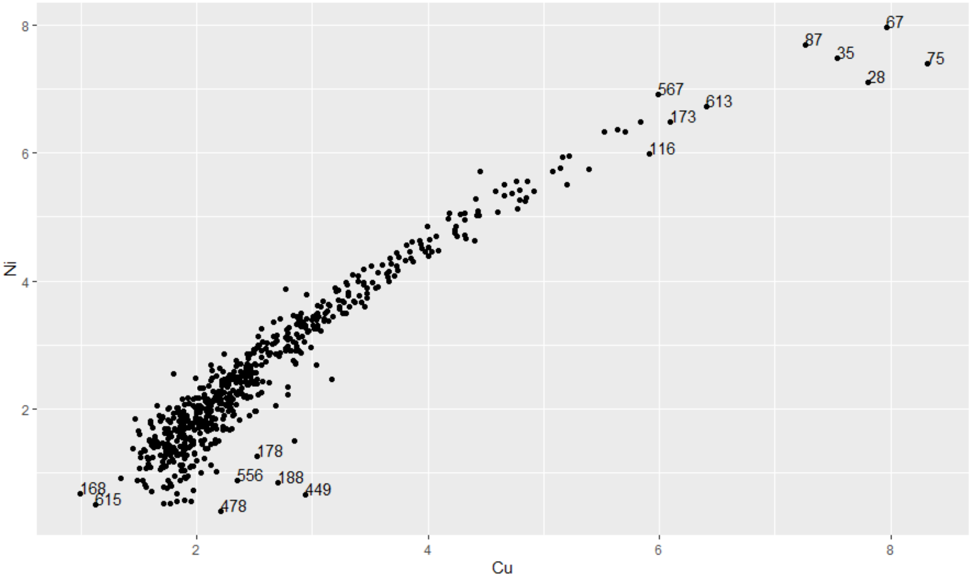


Figure 25: Scatterplot of Cu and Ni

Finally, we search for outliers in the dataset after being reduced to lower dimension. For that purpose we employ the method of Principal Components Analysis (PCA). PCA is the most famous procedure for dimension reduction. It produces the Principal Components that are fewer than the original variables when the latter are highly correlated (like the log transformations of Cu and Ni presented in Figure 25). Briefly, Principal components are the eigenvectors of the covariance data matrix with the highest eigenvalues multiplied by the underlying data matrix. Consequently they are linear transformations of the original variables, orthogonal and uncorrelated. Thus, in this section with the use of PCA we represent the distances between objects in 41-dimensional space in the directions with the maximum variance i.e. information of the dataset.

We present the code we used for the computation of the principal components and the first 5 components based on their proportion of variance.

```
PCA_log_humus<-princomp(log_humus)
```

Components	1	2	3	4	5
Proportion of variance	0.93	0.067	0.0065	4.04×10^{-5}	1.86×10^{-5}

Table 9: Components ordered by their proportion of variance

Based on Table 9 keeping only the first two components we can retain approximately 99% of the original variance. That means that we can actually visualize the distances of the observations without losing information.

With the scores computed above, we run again the methods setting as k (when required) the values in Table 10. We compare the results of the methods before and after the PCA giving a proportion of their intersecting values. The R code we used to obtain the scores is below.

```
PC12_log_humus<-PCA_log_humus$scores[,1:2]
```

	Percentage				
k Methods	5	10	15	20	30
dens	56%	76%	91.97%	95.83%	92%
LOF	76%	76%	92%	100%	96%
KNN_IN	36%	44%	60%	72%	80%
nnk	92%	84%	84%	88%	92%
maha	16%				
depthout	37.5%				

Table 10: Percentages of consistency before and after PCA

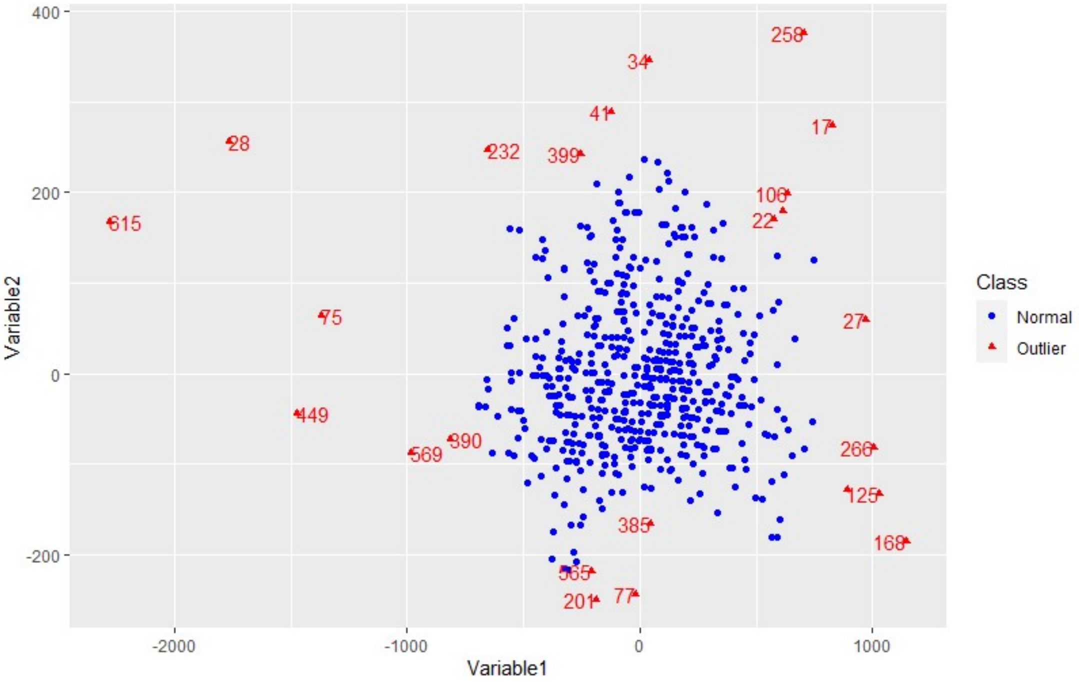


Figure 26: Scatterplot with the principal components for the dens method with $k = 20$. The observations labeled red are the detected outliers

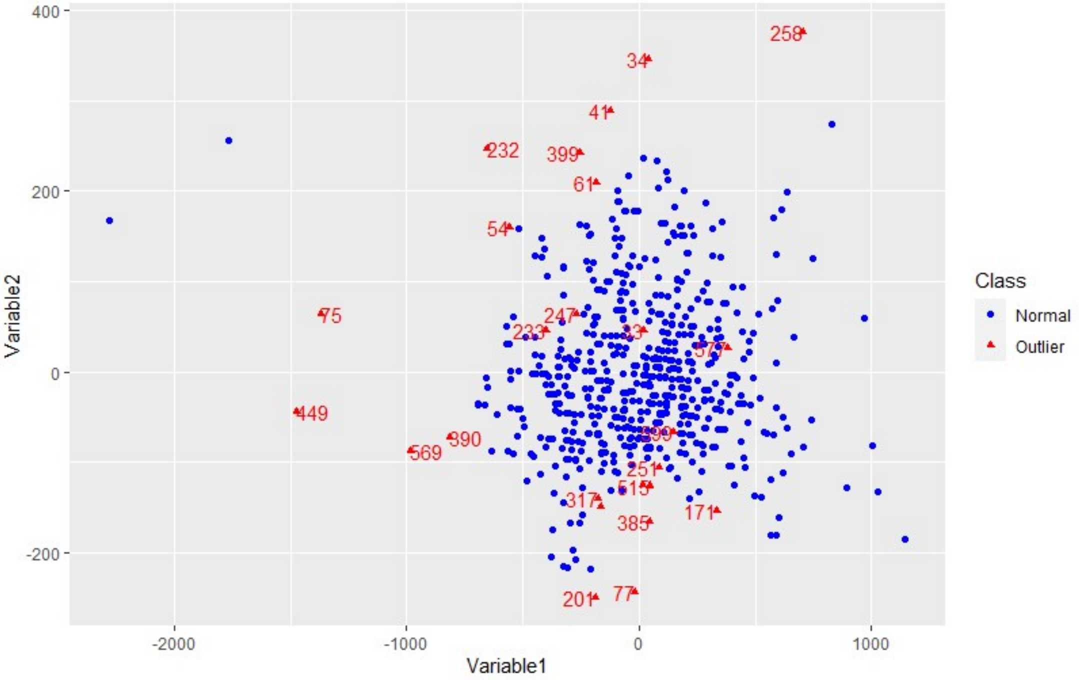


Figure 27: Scatterplot with the principal components for the dens method with $k = 5$. The observations labeled red are the detected outliers

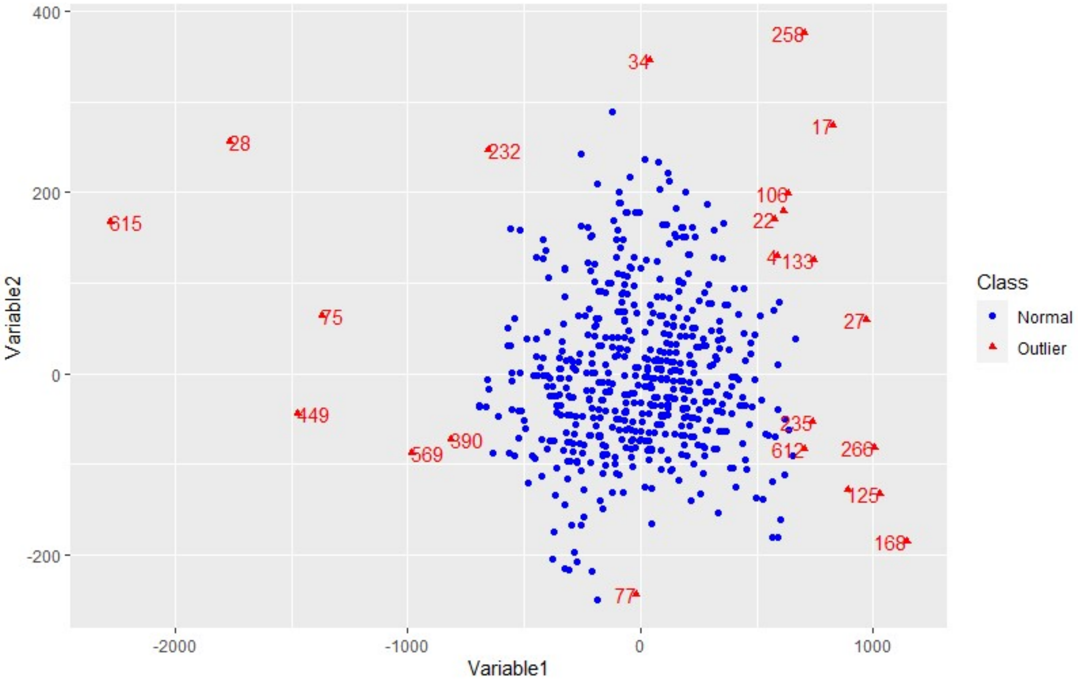


Figure 28: Scatterplot with the principal components for the nnk method with $k = 15$. The observations labeled red are the detected outliers

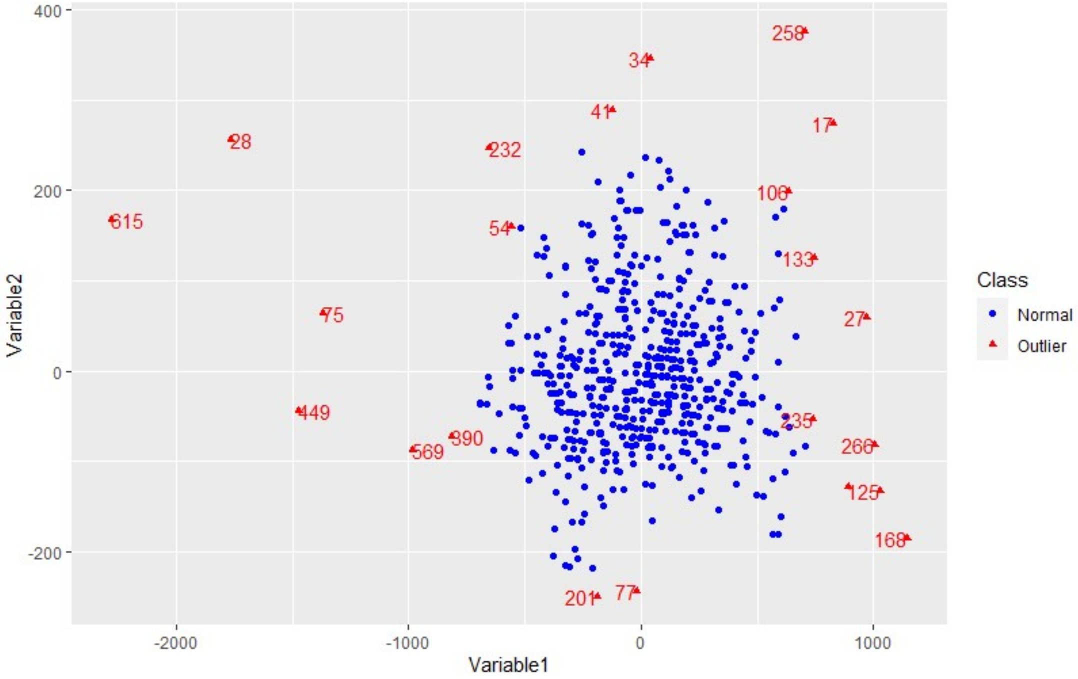


Figure 29: Scatterplot with the principal components for the nnk method with $k = 5$. The observations labeled red are the detected outliers

For LOF and KNN_IN, we compared the 25 highest values while in the remaining methods, the percentage is computed taking as denominator the larger number number of detected outliers before and after PCA of each method (we omitted nn and RKOF due to their consistency with nnk and dens respectively). We note that these proportions of common detected outliers do not imply the same weighting order.

Since we found from principal components analysis that we can project the whole dataset in two dimensions without losing almost any information, we are able to visualize it in a scatterplot. As we can see in the next figures, the objects form one cluster and the outliers seem to be in the form of extreme values.

We observe that increasing k in dens and LOF, the number of mutually detected ouliers also increases in each method before and after applying PCA to the data. Moreover, we observe the same thing for KNN_IN but with smaller rate. If we set $k = 60$ we get a percentage of consistency equal to 96% . On the contrary, the nnk method shows consistency in lower values of k.

In Figures 26 and 27 we see exactly why there is such a difference in the consistency between the results before and after applying PCA to the dataset. Choosing $k = 5$ makes the algorithm search for outliers in an undesirably local prospect. Based in Figure 27, the method detects observations that they do not seem to be outlying, just because the $k$nn neighborhood we chose for local detection is too small. Furthermore, even though PCA managed to retain almost all the information of the variables, some of it is inevitably missing and it is more likely to be seen in objects that are close to each other than the ones that were already sufficiently far. Consequently, the method tuned in $k = 5$, was set to detect outliers even inside the underlying region in which the distance of the objects after PCA was slightly changed. In contrast, **nnk** function marks as outliers those observations that have the highest $k$nn distances, that is, the points that are away from the region not depending so much on the value of k. That is why, its results we obtained before and after applying PCA were similar.

For the purpose of comparing these two methods, we note that due to the fact that the objects formed one cluster, they detected almost the same objects. **nnk** function detected observation 133 while **dens** did not even though both of the methods detected observation 106. However, **dens** function managed to detect observations 399 and 565 which can be (based on the plots) declared as outliers. Furthermore interesting is the case of observation 385 which even though it may not be considered as extreme, is definitely isolated from the region and was detected only by **dens**.

Finally, **maha** and **depthout** values do not seem to be as consistent as the aforementioned methods. **depthout** takes into account every pair of points in computing the modified band width of each observation without having a fluctuation parameter like distance or density based methods.

The case of the **maha** function is interesting. We present the results after running the **maha** function on the scores of the principal components.

```
>maha_PC12_log_humus<-maha(PC12_log_humus,0.975)
>maha_PC12_log_humus$`Location of Outlier`
>[1] 17  27  28  34  41  75  79 106 125 160 168 201 232 258 266 399 449 569 615
```

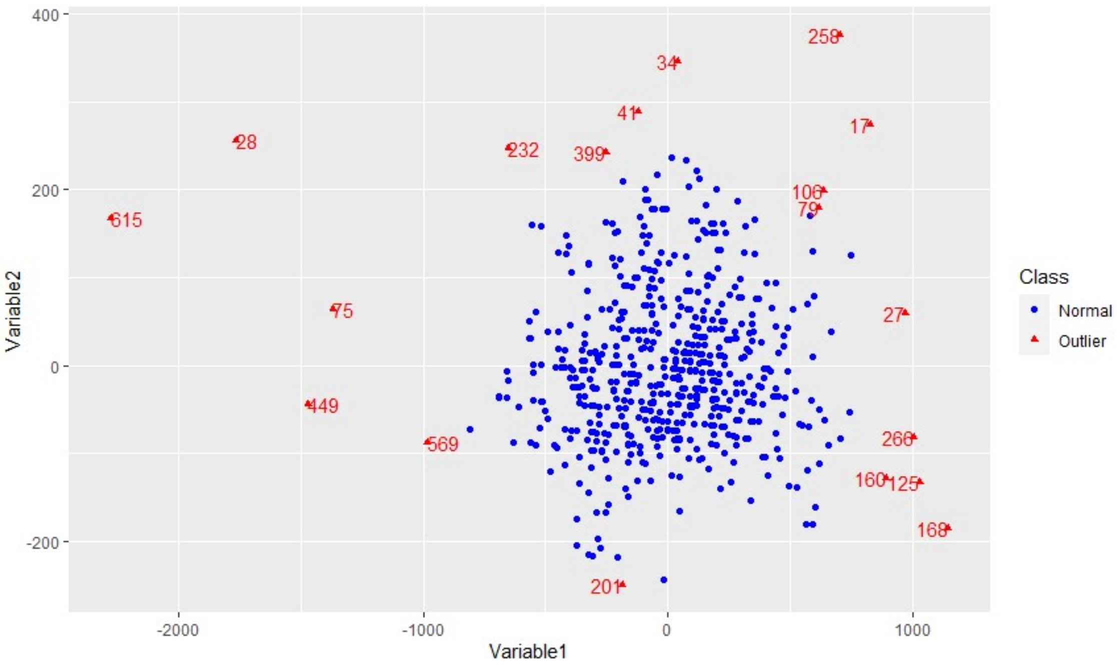


Figure 30: Scatterplot with the principal components for the maha method with $\sqrt{\chi^2_{2,0975}} \approx 2.72$ cutoff. The observations labeled red are the detected outliers

We recall that before applying PCA, the **maha** function detected 75 outlying observations. It is known that by employing the PCA method the underlying data matrix is projected into the directions that best explain the data set. With that being said, we can assume that objects that were not explained by the principal components are probably the outlying ones. Consequently, applying PCA to the data is very useful in the case of methods that declare outliers by their Mahalanobis distances.

Finally, we compare the results from the classical Mahalanobis distances with the robust distances.

For that purpose, we employ the following codes

```
   > Moutlier_PC12_log_humus<-Moutlier(PC12_log_humus, quantile=0.975,plot=TRUE)
> which(Moutlier_PC12_log_humus$cutoff<Moutlier_PC12_log_humus$rd)
12  17  27  28  34  36  40  41  49  54  61  73  75  77  79 106 125 160 168 201 221
232 244 258 266 268 275 375 399 449 569 615
```

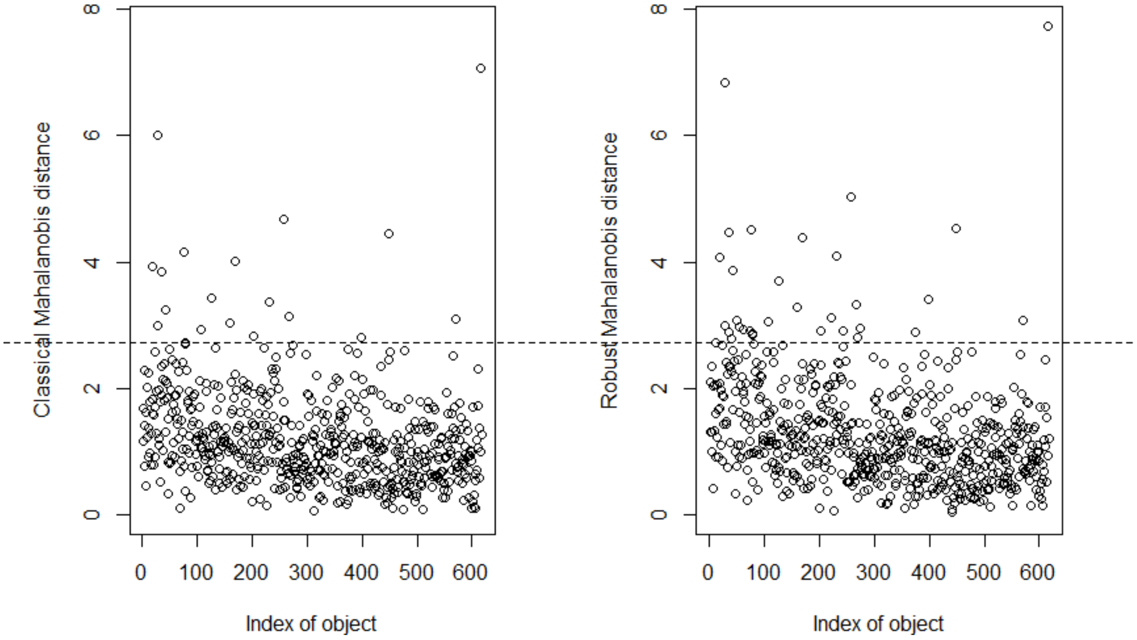


Figure 31: Plots showing the classical and robust Mahalanobis distances for $\sqrt{\chi^2_{2,0.975}} \approx 2.72$ cutoff.

Observing Figure 31 and the results we notice an increase in the number of outliers in the robust case, but considering that most of the outlying objects were only slightly above the cutoff line, both of the methods share the same set of outliers. Moreover, we employ the **aq.plot** function which compares the exceeding robust distances using the cutoff from χ^2 and an adjusted quantile depending on the empirical distribution of the robust distances. As we notice from the results, the exceeding robust distances based on the adjusted cutoff and taking into account the scatterplots presented in Figure 32, are more consistent with **dens** and **maha** than the ones exceeding the $\sqrt{\chi^2_{2,0.975}}$ value. However, interesting is the case of observation 77 that was only detected with the classical cutoff and not the adjusted.

```
> aq.plot_log_humus <- aq.plot(PC12_log_humus, delta=qchisq(0.975,
df=ncol(PC12_log_humus)), quan=1/2, alpha=0.05)
 which(TRUE==aq.plot_log_humus$outliers)
>[1]  17  27  28  34  41  49  75 106 125 160 168 201 221 232 244 258 266 399 449 569 615
```

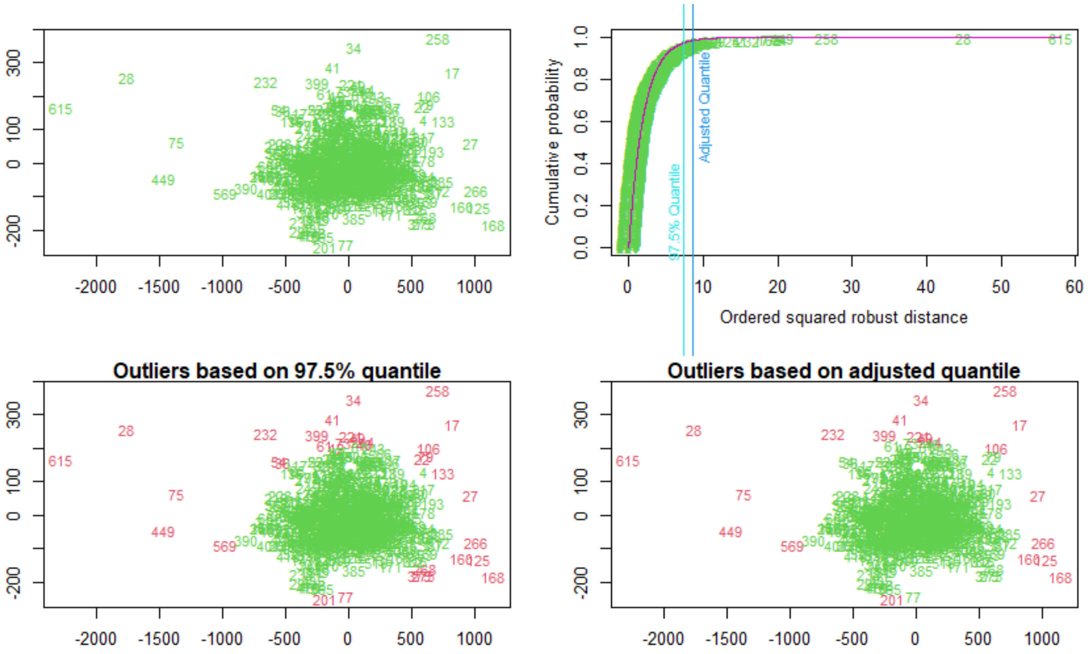


Figure 32: Upper left plot shows the robust Mahalanobis distances. Upper right plot shows the empirical distribution of the robust distances. The red line labels the χ^2 distribution and the two vertical lines label the 97.5% cutoff and the adjusted quantile cutoff respectively. The two lower plots labels in red the outliers that exceed each cutoff.

8.2 Conclusions

In this section, we presented all the methods that we have introduced, applying them to the humus dataset. First, we did a search for outliers applying all methods tuned to their default settings in the whole data. Even though we found common outlying objects in all of the methods, different methods detect different kind of outliers. Moreover, there is no straight answer to which are the right parameters for the optimal selection, due to high dimensionality of the raw data. In that context, we examined a subset of variables, to see if some of the strong outliers that were detected in the whole data frame could have originated from it with the use of the methods employed by the R package Outliers03. We noticed overlapping between most of the methods, specifically FastMCD, BACON and FastPCS. Finally we applied PCA to the data, which probably because of the high correlation of the variables could be projected in only two directions that preserved the variance of the objects. Moreover we observed a consistency of the percentages of the common outliers being detected for each method before and after PCA. Based on that last observation and the fact that we are able to visualize the data in a scatterplot, we compared the methods with k best suited for each method. We note that the lowest consistency percentage was yielded by the Mahalanobis distance method which before PCA declared far more observations than the other methods.

9 dat dataset

In this section we will apply the methods presented above in the dat dataset where the majority of the data came from a bivariate Normal distribution [19]. The dat dataset is included in the **mvoutlier** package. It is an illustrative data example consisting of 100 observations on two variables. It is constructed to have outliers.

In Figure 33, we mark the observations that can be considered as observations far away from the bulk of the dataset.

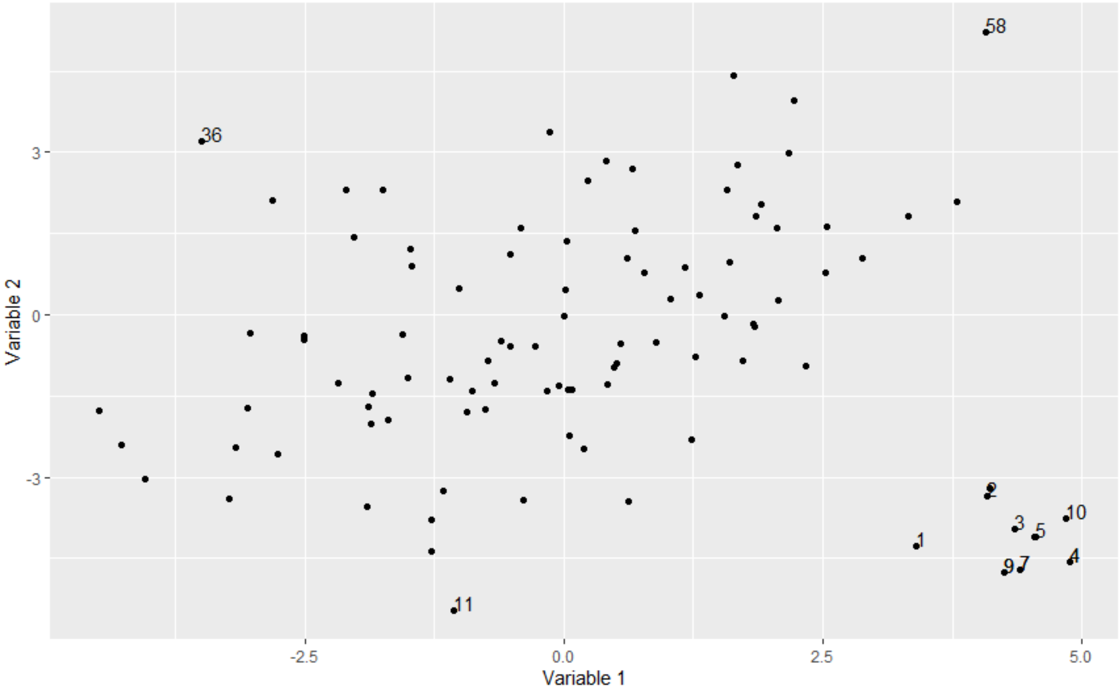


Figure 33: Scatterplot showing possible outlying points

9.1 Outlier analysis

We begin our analysis by applying the **maha** function to the data. Its code and the obtained results are shown below

```
maha_dat$`Location of Outlier`
[1] 58
> maha_dat$`Outlier Probability`
[1] 0.9887983
```

As we can see from the results the **maha** function detected only one observation. We speculate that the contamination in the dataset by extreme values, prevented the appropriate estimation of scale and location, leading to a masking effect for the rest of the outliying objects.

We continue our analysis by computing the robust Mahalanobis distances of the objects employing the MCD method.

```
 > Moutlier_dat<-Moutlier(dat,quantile=0.975, plot=TRUE)
> which(Moutlier_dat$cutoff<Moutlier_dat$rd)
 [1]  1  2  3  4  5  6  7  8  9 10 36
```

As we can see from the results and Figure 34 the Mahalanobis distances based on the MCD method exceeding the cutoff correspond to the objects forming the cluster to the bottom right plus observation 36. That means that the ellipse constructed by the MCD estimators of location and scale separates these points from the rest of the data in which observation 58 is included.

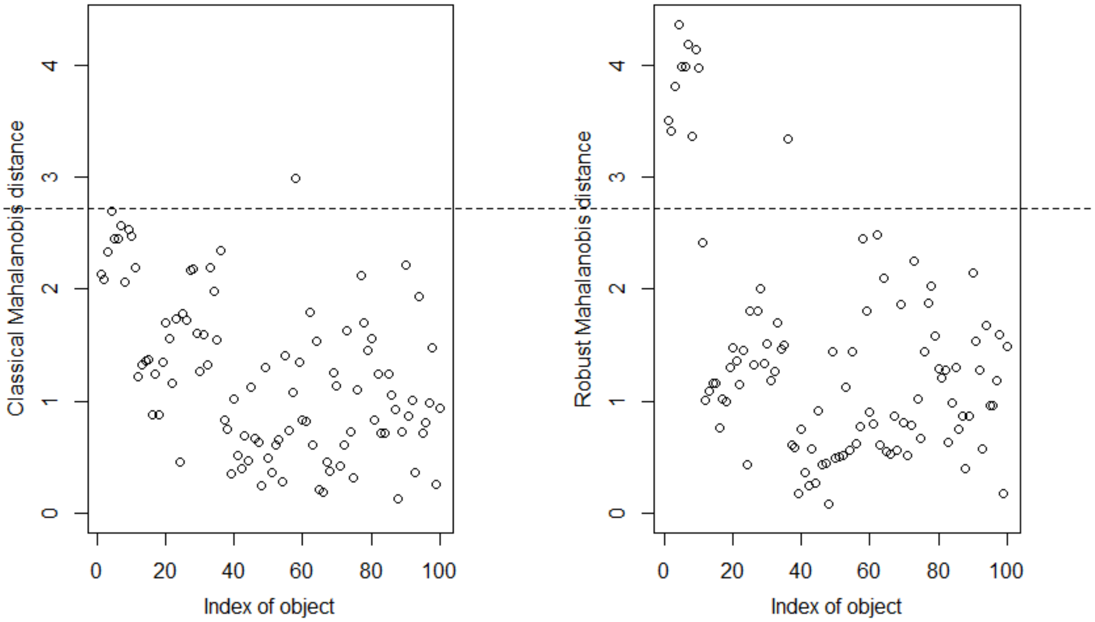


Figure 34: Plots showing (left) classical Mahalanobis distances versus the robust distances for cutoff $\sqrt{\chi^2_{2,0.975}} = 2.72$

We obtain the same results using as cutoff for the robust Mahalanobis distances the adjusted quartile, shown below in Figure 35.

```
aq.plot_dat <- aq.plot(dat, delta=qchisq(0.975, df=ncol(dat)), quan=1/2, alpha=0.05)
>which(TRUE==aq.plot_dat$outliers)
 [1]  1  2  3  4  5  6  7  8  9 10 36
```

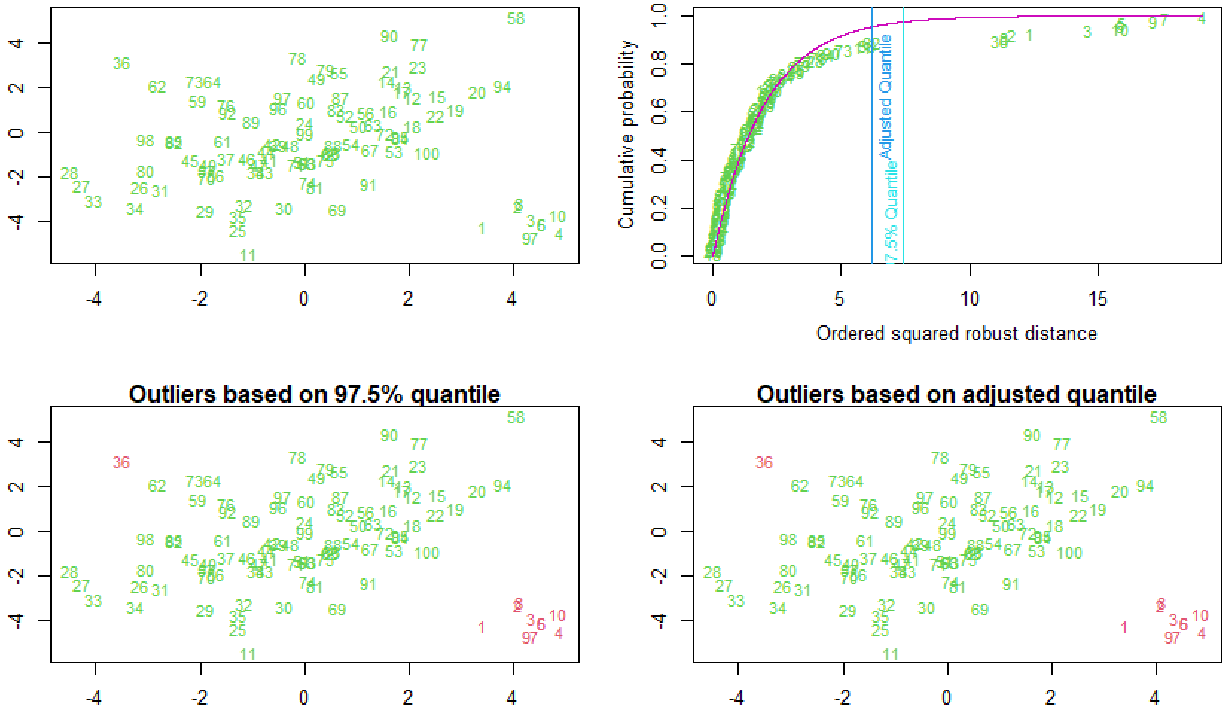


Figure 35: Upper left plot shows the robust Mahalanobis distances. Upper right plot shows the empirical distribution of the robust distances. The red line labels the χ^2 distribution and the two vertical lines label the 97.5% cutoff and the adjusted quantile cutoff respectively. The two lower plots labels in red the outliers that exceed each cutoff.

We continue our analysis by applying three methods implemented in the **OutliersO3** package.

```
PBMMethods_dat  <- O3prep(dat, method=c("PCS", "BAC", "MCD"),tolPCS=0.1,
tolBAC=0.95,tolMCD  =0.01 )
```

As we see in Figure 36, FastMCD, FastPCS and BACON overlap completely except for observation 36 which is only detected by BACON. Moreover, we notice that these results also overlaps with the results obtained by the Mahalanobis distances. This is quite reasonable, one might think, because all four methods form an ellipse based on the identification of an optimal subset of h observations and based on a $\chi^2_{p,a}$ cutoff declare the outliers.

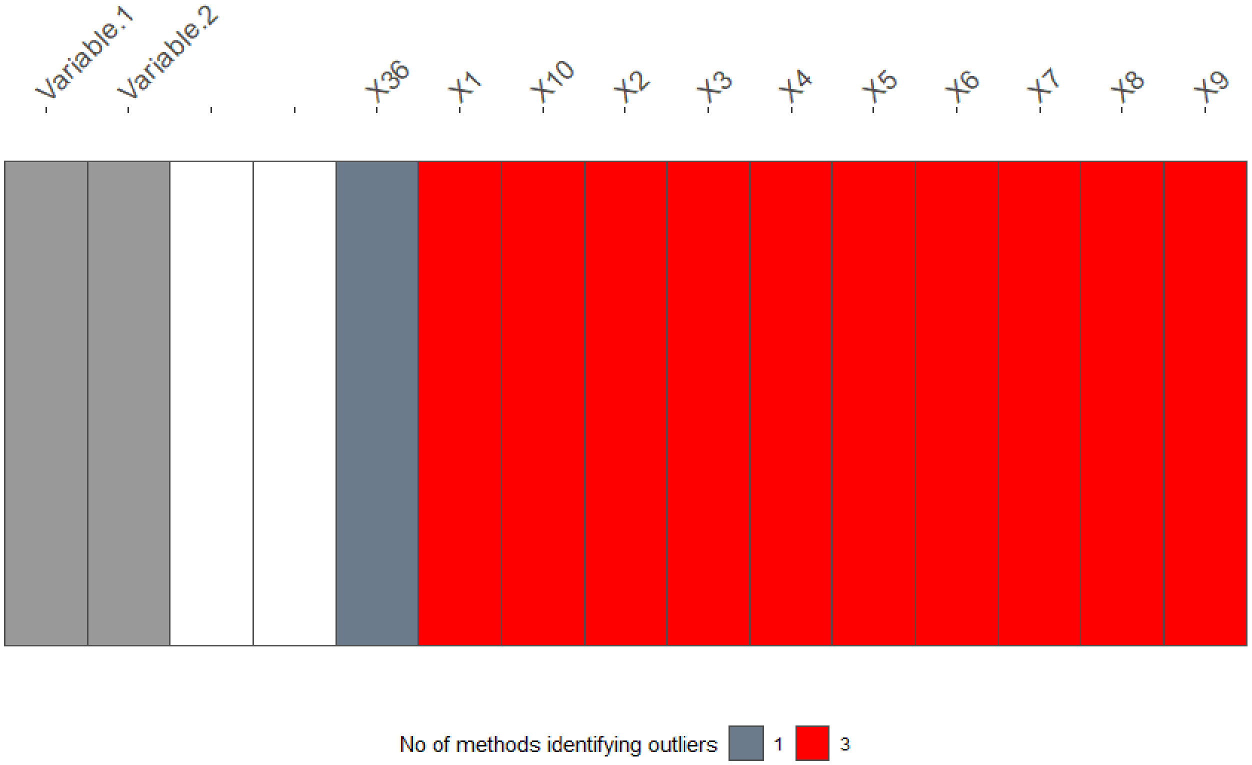


Figure 36: Plot comparing FastMCD, FastPCS and BACON

Next we apply the depth-based method to the data.

```
    > depthout_dat <- depthout(dat, rnames = FALSE, cutoff = 0.1, boottimes = 100)
> depthout_dat$`Location of Outlier`
[1]  4  5  6  7  9 10 36 58
> depthout_dat$`Outlier Probability`
[1] 1.00 0.95 0.95 1.00 0.99 0.90 0.99 0.96

> depthout_dat$`Location of Outlier`
 [1]  1  2  3  4  5  6  7  8  9 10 11 23 25 26 27
 28 29 31 33 34 36 58 62 73 77 90 94
> depthout_dat$`Outlier Probability`
 [1] 1.00 0.98 1.00 1.00 1.00 1.00 1.00 0.97 1.00
 1.00 0.97 1.00 0.90 0.95 1.00 0.98 0.90 0.74 1.00
 1.00 1.00 1.00 1.00 0.92 1.00 0.97 0.96
```

In Figures 37 and 38 we see that increasing the percentile threshold the observations that are only in the border of the data cloud are detected. For the underlying dataset we conclude that this way of searching for outliers is less focused than the other methods above.

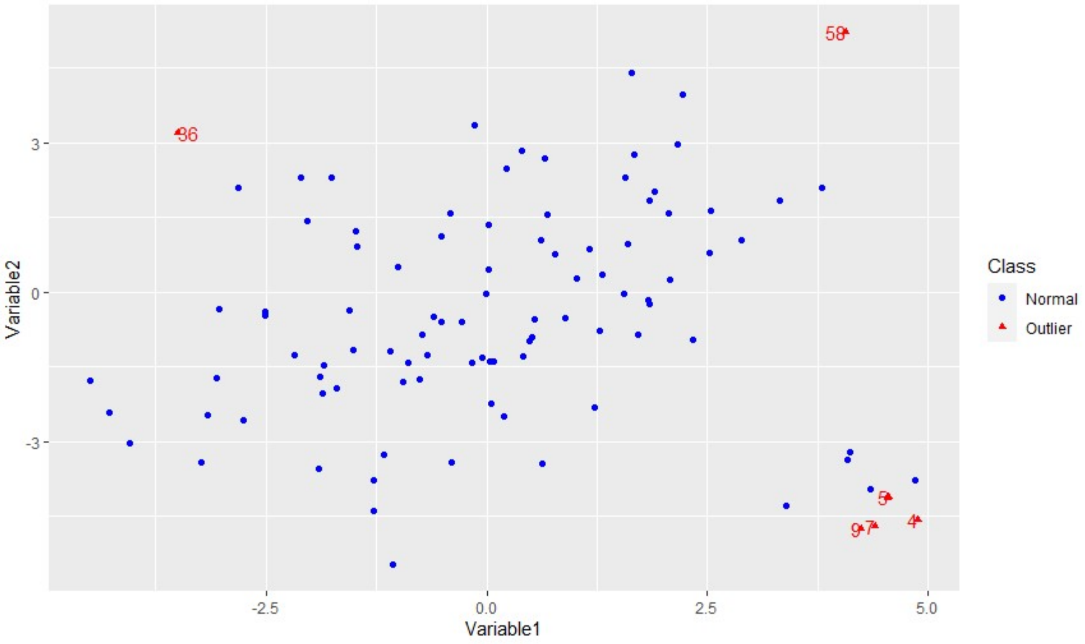


Figure 37: Scatterplot of observations detected by **depthout** for percentile threshold 0.1

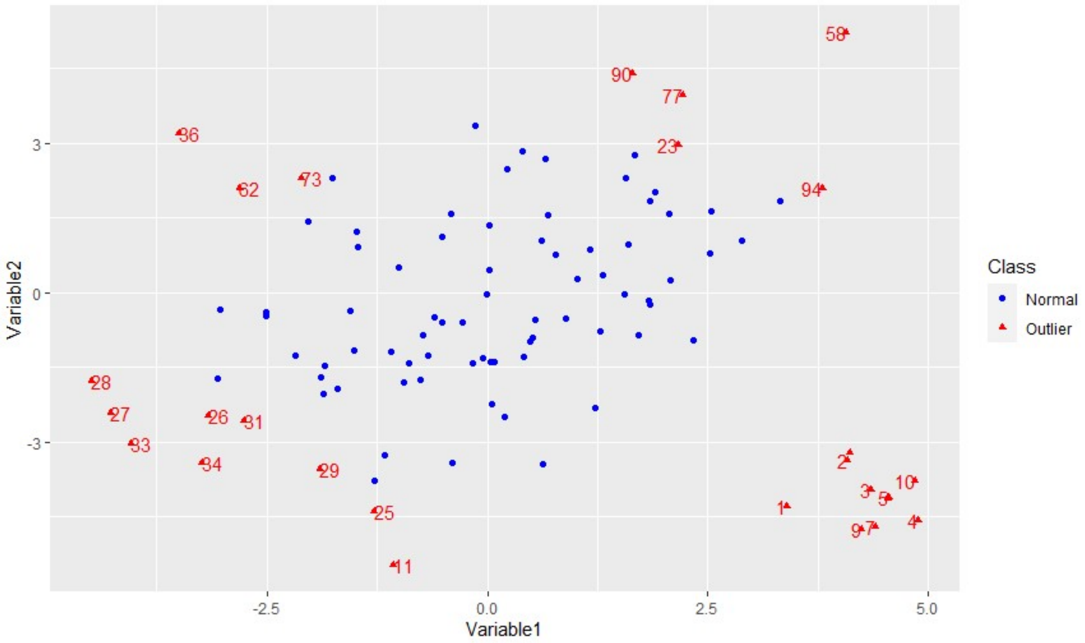


Figure 38: Scatterplot of observations detected by **depthout** for percentile threshold 0.3

nnk function	
number of outliers	*k*
11 27 28 34 36 58 77 78 90 94 98	5
11 28 35 36 58 62 69 73 77 78 90 94	6
11 25 28 35 36 58 62 69 77 78 90 94	7
11 25 27 28 33 34 36 58 62 69 77 78 90	8
11 25 27 28 33 36 58 62 69 77 78 90	9
11 25 27 28 33 36 58 62 69 77 90 94	10
1 2 3 4 5 6 7 8 9 10 11 36 58	11
1 2 3 4 5 6 7 8 9 10 36 58	15
1 2 3 4 5 6 7 8 9 10 36 58	30
1 2 3 4 5 6 7 8 9 10 36 58	40

Table 11: Number of detected outliers for different *k* for **nnk** for percentile equal to 0.85

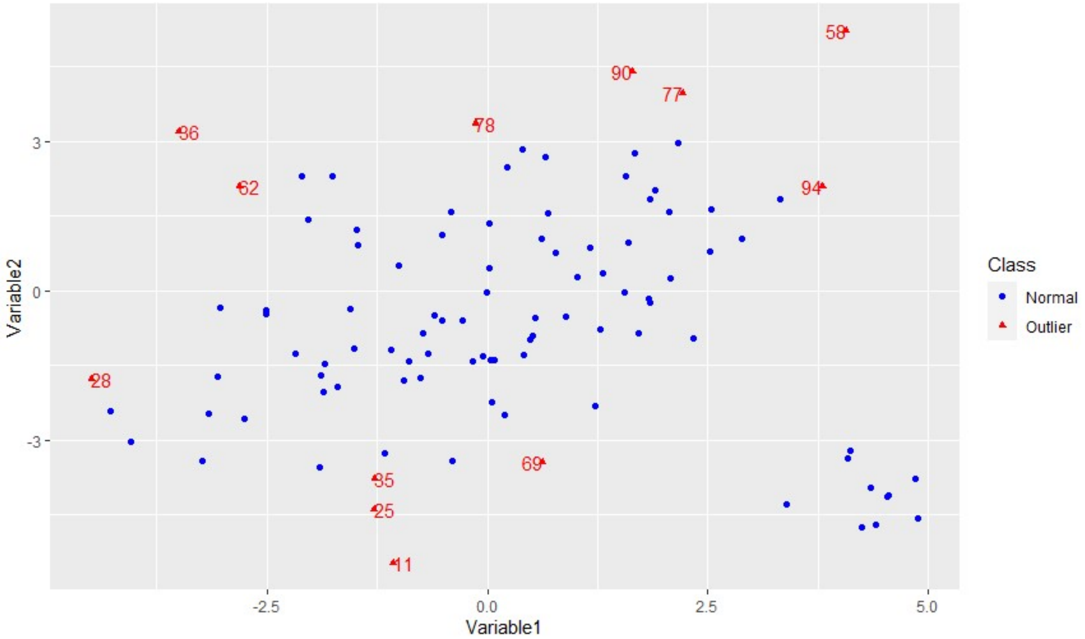


Figure 39: Scatterplot of observations detected by **nnk** for percentile threshold 0.85

In Table 11 we present the outliers detected by the **nnk** function for different values of k. We observe an instability in the first values and then a convergence to the observations that were also detected by the robust Mahalanobis distances plus observation 58 that in the latter method was considered to be extreme but not as outlier. Moreover, in Figure 39, choosing $k = 7$ sets the algorithm to search for outliers only in the bulk of data to the left of the scatterplot where it detects the observations that are on its border while the small cluster to the right does not get detected for any outlying action due to the small value of k.

KNN_IN function	
number of outliers	k
1 58 8 36 100 11 28 85 90 91 94 98 9	5
11 58 36 28 77 90 94 27 33 62 20 25 73	10
58 11 36 62 27 33 90 94 25 28 73 77 1	15
58 11 36 27 62 90 25 28 73 94 33 77 1	16
58 11 36 27 62 90 25 28 73 94 33 77 1	17
58 11 36 27 62 28 90 94 25 33 73 1 2	18
58 11 36 27 62 28 90 25 73 94 1 2 3	19
58 36 11 27 62 28 25 73 90 1 2 3 4	20
36 58 2 3 4 5 6 7 8 9 10 27 1	30
36 58 3 4 5 6 7 9 10 8 1 2 62	40

Table 12: Number of detected outliers for different k for **NNK_IN**. The thirteen observations with the lowest indegree number are listed

Applying **KNN_IN** to the dataset we obtain the same results as **nnk** for high values of k. The difference between these methods lies in $k = 15$. Setting $k = 15$ the **KNN_IN** method detects values that are indeed far away from the right data cloud, marked with red colour as we see in Figure 40. Moreover, it detects observation 1 (marked with black colour) which is isolated from both clusters. Consequently, for $k = 15$ **KNN_IN** handles both of the clusters as outlier-free subsets points and detects as outliers the points that are far away from each cluster. That is why, this method even if it uses $k$nn distances, also shares some of the properties of a density based method as it searches for observations that participate in the fewest k nearest neighborhoods.

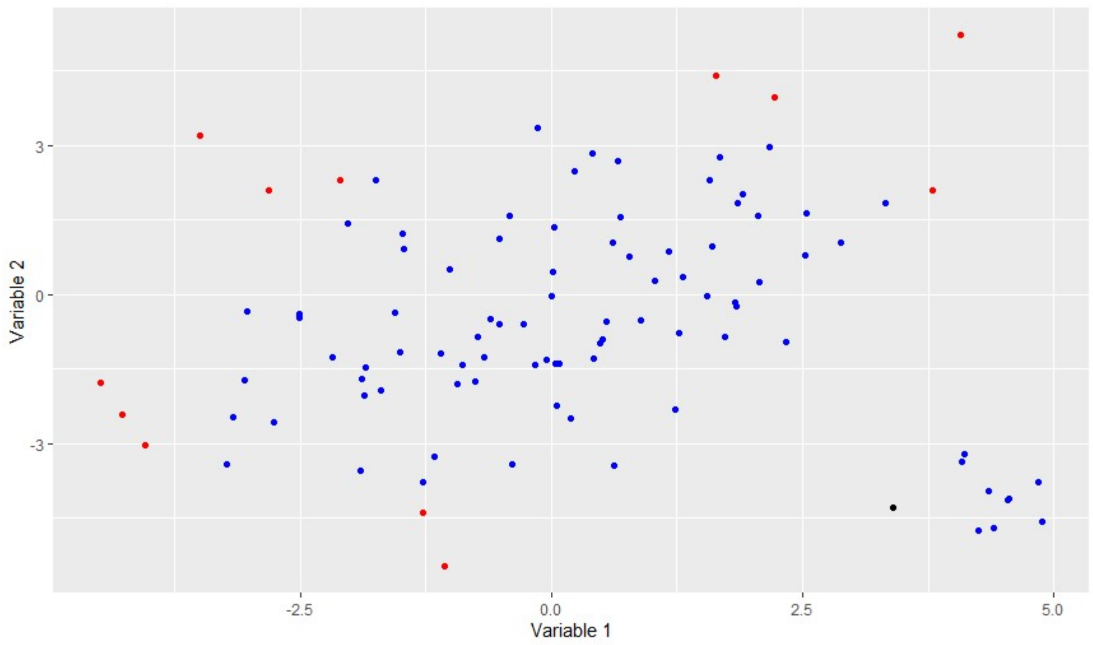


Figure 40: Scatterplot of observations detected by **KNN_IN**

detected outliers		
LOF	**RKOF**	k
11 80 81 98 61 1 74 90 77 36 94 91 58	81 100 11 80 94 61 77 90 36 74 1 91 58	5
81 35 32 90 62 77 33 27 91 69 11 36 58	98 33 32 81 27 77 94 90 11 69 91 36 58	10
32 28 62 25 77 90 27 91 33 69 11 36 58	81 94 32 28 33 27 77 90 11 91 69 36 58	11
7 90 4 25 8 2 91 27 33 69 11 36 58	77 28 30 33 27 90 2 8 91 11 36 69 58	12
8 2 3 9 6 5 69 10 7 4 11 36 58	91 3 6 5 8 2 7 10 4 11 36 69 58	13
1 8 2 3 6 5 10 9 7 4 11 36 58	1 6 5 7 9 4 36 69 10 11 2 8 58	14
1 8 2 9 7 3 36 6 5 10 11 4 58	7 9 69 36 3 6 5 2 11 8 4 10 58	15
11 36 8 2 1 3 6 5 10 58 9 7 4	36 11 8 2 1 10 3 6 5 9 7 4 58	20
11 8 2 36 1 3 58 10 6 5 9 7 4	11 36 8 2 1 58 3 10 6 5 9 7 4	30
11 36 8 1 2 3 6 5 10 9 7 58 4	11 36 8 1 2 3 58 6 5 10 9 7 4	40

Table 13: Detected outliers for different k for **LOF** and **RKOF**. The thirteen observations with the highest LOF and RKOF values are listed

Table 13 shows the thirteen most outlying observations based on their LOF and RKOF values (the computed RKOF values are based on the default settings of **RKOF** function). We notice that both methods follow the same pattern, noting an instability in small values of k while for higher values both methods detect the objects marked in Figure 33 .

Table 14 shows the outliers based on the highest RKOF values, the computations of which were made setting $C = 0.1$. Parameter C influences the $k$nn distances enabling RKOF to be more sensitive for objects in the spatial regions of the dataset as we see in Figure 41. We note, observations 99 and

24 to be detected for the first time as outliers lying inside the data cloud. Such isolated observations were not detected by any of the previous methods, because most of the previous methods are able to detect outliers on a more global scale. Finally, we note that by setting $k = 11$ and $C = 0.1$ the method handles the cluster in the right-hand side of the scatterplot as uncontaminated and searches for outliers in the borders and in spatial regions of the left cluster.

RKOF function	
detected outliers	k
80 45 18 4 10 100 91 69 11 61 36 58 1	5
78 11 24 18 99 30 80 100 36 69 61 91 58	10
78 99 18 24 11 30 80 100 36 69 61 91 58	11
18 78 99 24 11 30 80 100 36 69 61 91 58	12
18 80 78 24 99 11 30 100 36 69 61 91 58	13
37 78 80 24 99 100 11 30 36 69 61 91 58	14
78 37 24 80 100 99 11 30 36 61 69 91 58	15
89 80 99 78 24 100 11 30 36 69 61 91 58	20
80 78 89 99 24 100 30 11 36 61 69 91 58	30
78 34 80 24 100 89 30 11 61 36 69 91 58	40

Table 14: Detected outliers for different k for **RKOF** with C=0.1

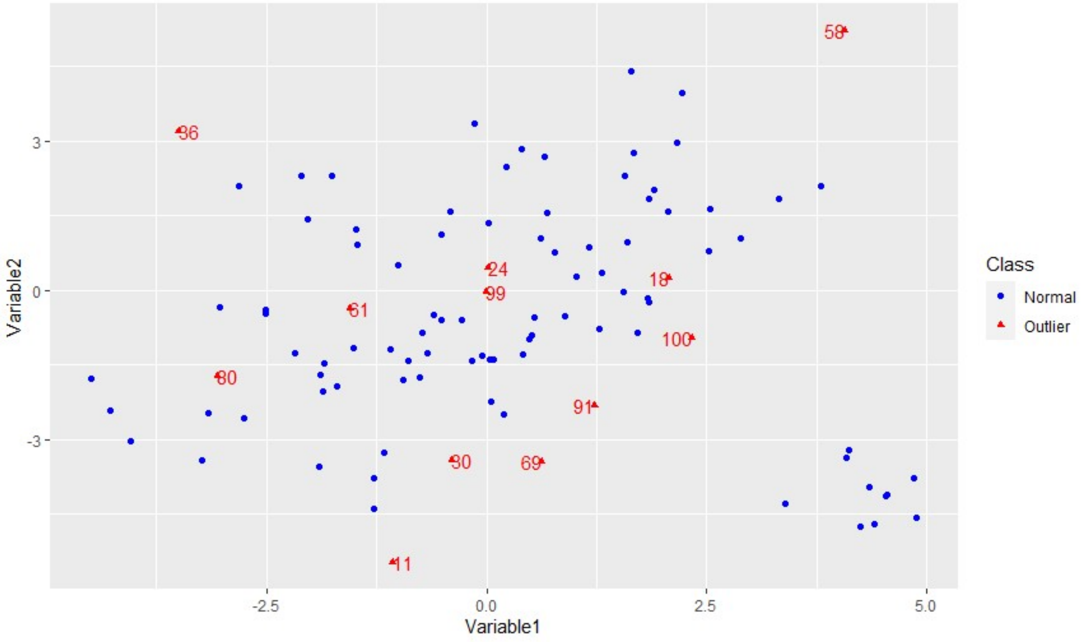


Figure 41: Scatterplot of observations detected by **RKOF** for $k = 11$

9.2 Conclusions

In this chapter we searched for outliers in the dat dataset. As we saw in the scatterplots, the data points form one big spatial cluster and a small cluster with a lot fewer objects. Distribution-based methods, like FastMCD, BACON FastPCS and robust Mahalanobis distances, were able to detect the bulk of the normal data and separate it from the outlying objects. We observe that these methods did not declare observation 58 as an outlier. Furthermore, the depth-based method based on the modified band depth, assigned lower depths to the observations that were on the global borders of the data cloud. Distance-based method **nnk** for high values of k recognized as outliers the small cluster that is far away from the rest of the data as well as some objects of extreme values lying in the bounds of the big cluster while for lower values of k it did not search for outlying objects in the small cluster. **KNN_IN** behaved like **nnk** for high values of k but for lower values k handled both clusters as normal and searched for outliers with respect to each cluster. Finally, density-based methods for high values of k gave the same results as **nnk** and **KNN_IN** while for small values (especially the **RKOF**) they searched also for isolated points within the big cluster.

10 General Conclusions

Outlier detection has always been a hot topic in data analysis because there is no general method to search for points out of a pattern. Moreover, outlier analysis helps the user to make a more focussed interpretation for the target dataset and at the same time to unlock possible prospects for further investigation based on the detected abnormalities.

Consequently, this inability to construct a generally effective method led to a variety of different outlier detection methods, some of which were discussed and examined above. Briefly, in Chapters 3, 5 and 6 we presented methods that unlike the methods introduced in the later chapters do not depend on the distribution generating the data. However, they are also based on a certain assumption whether this is depth, distance or density. Inevitably, all of the aforementioned methods differ in the kind of outliers they are searching for.

Testing these methods on the humus and dat datasets we conclude that no method outperforms all the others.

In case of dat, we noticed that even though the objects are normally distributed, widespread contamination can mask the true outlying objects when applying classical methods like the Mahalanobis distance so robust tools have to be employed. Moreover, we encountered a significant difference between distributional and non-distributional methods. While the latter methods declared some of the points as outliers, the former methods labeled them as normal points of extreme values. Finally, methods based on the data's density are able to detect points that for the rest of the methods could be considered normal as they are designed to be more sensitive searching for isolated points in the inside regions.

In case of humus, we saw how these methods work in higher dimensions. Moreover, we searched for outliers in a subset of variables to see the behaviour of the outlying objects detected on the global scale. Finally, we analyzed how the obtained results change after using the PCA method showing that Mahalanobis distance can be affected greatly by a dimension reduction even if most of the information is retained. In contrast, methods based on distance were consistent after variable reduction while methods based on density showed that the local regions of each observation can be modified after using PCA even if we are able to retain most of the variation.

References

[1] E. Acuna and C. Rodriguez (2004). *A Meta analysis study of outlier detection methods in classification.* Technical paper, Dept. of Mathematics, University of Puerto Rico at Mayaguez

[2] C. Aggarwal (2013). *Outlier Analysis*, Springer, New York

[3] F. Alqallaf, S. Van Aelst, V. Yohai and R. H. Zamar (2009). Propagation of Outliers in Multivariate Data, *The Annals of Statistics*, **37**, pp. 311–331

[4] V. Barnett and T. Lewis (1994). *Outliers in Statistical Data*, Wiley, Chichester

[5] I. Barranco-Chamorro, C. Grentzelos (2020). Some Uses of Orthogonal Polynomials in Statistical Inference. Submitted to *Computational and Mathematical Methods* (it passed first revision).

[6] N. Beckmann, H. P. Kriegel, R. Schneider, and B. Seeger (1990). The R-tree: An Efficient and Robust Access method for Points and Rectangles. *In Proc. of ACM SIGMOD*, pp. 322–331

[7] N. Billor, A. S. Hadib and P. F. Velleman (2000). BACON: blocked adaptive computationally eficient outlier nominators, *Computational Statistics & Data Analysis*, **34**, pp. 279-298

[8] M. Breunig, H. P. Kriegel, R. Ng, and J. Sander (2000). LOF: Identifying Density-based Local Outliers, *In Int. Conf. On Management of Data*, pp. 93-104

[9] G.Brys, M.Hubert and P.J. Rousseeuw (2005). A robustification of independent component analysis, *Chemometrics*, **19**, pp. 364–375

[10] G. Brys, M. Hubert and A. Struyf (2004). A robust measure of skewness, *Journal of Computational and Graphical Statistics*, **13**, pp. 996–1017

[11] C. Caroni and N. Billor (2007). Robust Detection of Multiple Outliers in Grouped Multivariate Data, *Journal of Applied Statistics*, **34**, 1241-1250

[12] B. R. Clarke (2018). *Robustness theory and application*, Wiley, Hoboken

[13] D.L. Donoho (1982). *Breakdown Properties of Multivariate Location Estimators*, Ph.D. Qualifying Paper, Harvard University

[14] D.L. Donoho, M. Gasko (1992). Breakdown properties of location estimates based on halfspace depth and projected outlyingness, *Ann. Statist.*, **20**, pp. 1803-1827

[15] N. R. Draper and H. Smith (1966). *Applied Regression Analysis*, Wiley, New York

[16] P. Filzmoser (2004). A Multivariate Outlier Detection Method,
http://file.statistik.tuwien.ac.at/filz/papers/minsk04.pdf.

[17] P. Filzmoser, R. G. Garrett and C. Reimann (2005). Multivariate outlier detection in exploration geochemistry, *Computers & Geosciences*, **31**, pp.579–587

[18] P. Filzmoser and M. Gschwandtner (2018). mvoutlier: Multivariate Outlier Detection Based on Robust Methods. R package version 2.0.9. https://CRAN.R-project.org/package=mvoutlier

[19] P. Filzmoser, A. Ruiz-Gazen and C. Thomas-Agnan (2014). Identification of local multivariate outliers, *Statistical Papers*, **55**, pp. 29-47

[20] P. Filzmoser and K. Varmuza (2017). chemometrics: Multivariate Statistical Analysis in Chemometrics. R package version 1.4.2. https://CRAN.R-project.org/package=chemometrics

[21] J. Gao, W. Hu, Z. Zhang, X. Zhang, and O. Wu (2011). RKOF: Robust Kernel-Based Local Outlier Detection. *Pacific-Asia Conference on Knowledge Discovery and Data Mining: Advances in Knowledge Discovery and Data Mining*, pp. 270-283

[22] R. G. Garrett (1989). The chi-square plot: A tool for multivariate outlier recognition, *Journal of Geochemical Exploration*, **32**, pp. 319-341

[23] C. Grentzelos, C. Caroni and I. Barranco-Chamorro (2020). Statistical techniques to handle outliers in multivariate data. *20th International Conference Computational and Mathematical Methods in Science and Engineering*, Rota, Spain.

[24] F. E. Grubbs (1969). Procedures for Detecting Outlying Observations in Samples, *Technometrics*, **11**(1), pp. 1-21

[25] V. Hautamaki, I. Karkkainen, and P. Franti (2004). Outlier Detection using k-nearest neighbor graph, *International Conference on Pattern Recognition*, pp. 430-433

[26] D. M. Hawkins (1980). *Identification of outliers*, Chapman and Hall, London

[27] W. Jin, A. Tung, and J. Han (2001). Mining Top-n Local Outliers in Large Databases. *ACM KDD Conference*, pp. 293-298

[28] T. Johnson, I. Kwok, and R. T. Ng (1998). Fast computation of 2-dimensional depth contours. *ACM KDD Conference*, pp. 224–228

[29] E. Knorr and R. Ng (1998). Algorithms for mining distance-based outliers in large datasets, *In Proc. of the VLDB Conference*, pp. 392–403

[30] A. Lazarevic and V. Kumar (2005). Feature bagging for outlier detection, *In: KDD*, pp. 157–166

[31] R. Liu (1990). On a Notion of Data Depth Based on Random simplices, *Ann. Statist.* **18**, pp. 405-414

[32] R. Y. Liu, J. M. Parelius and K. Singh (1999). Multivariate Analysis By Data Depth: Descriptive Statistics, Graphics And Inference. In Technical Report, *Dept. of Statistics, Rutgers University*

[33] S. Lopez-Pintado, Y. Romo and A. Torrente (2010). Robust depth-based tools for the analysis of gene expression data, *Biostatistics*, **11**, pp. 254-264

[34] H. P. Lopuhaa, and P. J. Rousseeuw (1991). Breakdown Points of Affine Equivariant Estimators of Multivariate Location and Covariance Matrices, *The Annals of Statistics*, **19**, pp. 229-248

[35] J. H. Madsen (2018). DDoutlier: Distance & Density-Based Outlier Detection. R package version 0.1.0. https://CRAN.R-project.org/package=DDoutlier

[36] Y. Manolopoulos, A. Nanopoulos, A. N. Papadopoulos and Y. Theodoridis (2005).*R-Trees: Theory and applications*, Springer, London

[37] R. A. Maronna, R. D. Martin and V. J. Yohai (2006). *Robust Statistics: Theory and Methods*, Wiley, New York

[38] D. Pahuja and R. Yadav (2013). Outlier Detection for Different Applications: Review, *International Journal of Engineering Research & Technology*, **2**

[39] S. Ramaswamy, R. Rastogi, and K. Shim (2000). Efficient algorithms for mining outliers from large data sets, *ACM SIGMOD Conference*, pp. 427–438

[40] N. N. R. Ranga Suri, Narasimha Murty M, G. Athithan (2019). *Outlier Detection Techniques and Applications: A Data Mining Perspective*, Springer, Switzerland

[41] C. Reimann, M. Äyräs, V. Chekushin,I. Bogatyrev, R. Boyd, P. D. Caritat, R. Dutter, T.E. Finne, J.H. Halleraker, Ø. Jæger, G. Kashulina, O. Lehto, H. Niskavaara, V. Pavlov, M.L. Räisänen, T. Strand, T. Volden (1998). Environmental Geochemical Atlas of the Central Barents Region. NGU-GTK-CKE Special Publication, *Geological Survey of Norway*, Trondheim, Norway, pp. 745

[42] B. Rosner (1983). Percentage Points for a Generalized ESD Many-Outlier Procedure, *Technometrics*, **25**(2), pp. 165-172

[43] P. J. Rousseeuw (1984). Least Median of Squares Regression, *Journal of the American Statistical Association*, **79**, 871-880

[44] P. J. Rousseeuw and W. V. D. Bossche (2018). Detecting Deviating Data Cells, *Technometrics*, **60**, pp. 135-145

[45] P. J. Rousseeuw and K. V. Driessen (1999). A Fast Algorithm for the Minimum Covariance Determinant Estimator, *Technometrics*, **41**, pp.212-223

[46] P.J. Rousseeuw and A. Leroy (1987). *Robust regression and outlier detection*, Wiley, New York

[47] P. J. Rousseeuw and I. Ruts (1996). Bivariate location depth, *Applied Statistics*, **45**, pp. 516-526

[48] N. Roussopoulos, S. Kelley, and F. Vincent (1995). Nearest neighbor queries. *In Proc. of ACM SIGMOD*, pp. 71–79

[49] I. Ruts, P. J. Rousseeuw (1996). Computing depth contours of bivariate point clouds, *Computational Statistics & Data Analysis* **23** pp. 153-168

[50] K. T. Schwartz (2008). *Wind Dispersion of Carbon Dioxide Leaking from Underground Sequestration, and Outlier Detection in Eddy Covariance Data using Extreme Value Theory*, PhD thesis, University of California Berkeley

[51] S. Seo (2002). *A Review and Comparison of Methods for Detecting Outliers in Univariate Data Sets*, Master's Thesis, University of Pittsburgh, http://d-scholarship-dev.library.pitt.edu/id/eprint/7948

[52] B. W. Silverman (1986). *Density Estimation for Statistics and Data Analysis*, Chapman and Hall, London

[53] W. Stahel (1981). *Robust Estimation: Infinitesimal Optimality and Covariance Matrix Estimators*, PhD thesis, ETH Zurich

[54] V. Tiwari and A. Kashikar (2019). OutlierDetection: Outlier Detection. R package version 0.1.1. https://CRAN.R-project.org/package=OutlierDetection

[55] A. Trigo and A. Kallhof (2017). Anomaly Detection,
https://rstudio-pubs-static.s3.amazonaws.com/301942_b3071cae53e64017a5f48550dbf1539a.html

[56] J. Tukey (1977). *Exploratory Data Analysis*, Addison-Wesley, London

[57] J. W. Tukey (1975). Mathematics and the picturing of data, *In Proc. International Congress on Mathmatics*, pp. 523-531

[58] A. Unwin (2019). Multivariate Outliers and the O3 Plot, *Journal of Computational and Graphical Statistics*, **28**, pp. 635-643

[59] A. Unwin (2020). OutliersO3: Draws Overview of Outliers (O3) Plots. R package version 0.6.3. https://CRAN.R-project.org/package=OutliersO3

[60] K. Vakili and E. Schmitt (2014). Finding Multivariate outliers with FastPCS, *Computational Statistics & Data Analysis*, **69**, pp. 54–66

[61] E. Vandervieren, M. Hubert (2004). An adjusted boxplot for skewed distributions, *In COMPSTAT, Proceedings in Computational Statistics*, pp. 1933–1940

[62] K. Varmuza and P. Filzmoser (2009). *Introduction to Multivariate Statistical Analysis in Chemometrics*, CRC Press, Boca Raton, FL

[63] I. Weissman (1978). Estimation of parameters and larger quantiles based on the k largest observations, *Journal of the American Statistical Association*, **73**, pp. 812–815

[64] L. Wilkinson (2018). Visualizing Big Data Outliers through Distributed Aggregation, *IEEE Transactions on Visualization and Computer Graphics*, **24**, pp. 256-266

Εθνικο Μετσοβιο Πολυτεχνειο&
Πανεπιστημιο Σεβιλλης

Ελληνικη περιληψη διπλωματικης εργασιας

Στατιστικές Μέθοδοι Ανίχνευσης Έκτροπων Παρατηρήσεων σε Πολυδιάστατα Δεδομένα

Χρήστος Γρέντζελος

Επιβλέπουσες Καθηγήτριες
Χρυσηίς Καρωνη
Inmaculada Barranco-Chamorro

Εξεταστής
Juan Luis Moreno-Rebollo

Η εργασία αυτή πραγματοποιήθηκε με την υποστήριξη του προγράμματος Εράσμους στο πλαίσιο Κινητικότητας Φοιτητών και Προσωπικού μέσω της Διμερούς Συμφωνιας μεταξύ του Πανεπιστημίου της Σεβίλλης και του Εθνικου Μετσοβίου Πολυτεχνείου.

Ιούλιος 2020

Περίληψη

Στην παρούσα εργασία, επικεντρωνόμαστε σε μεθόδους ανίχνευσης έκτροπων παρατηρήσεων σε πολυδιάστατο χώρο. Πιο αναλυτικά, μία έκτροπη παρατήρηση εκφράζει κάποιου είδους ανώμαλης συμπεριφοράς στο συνολικό σύνολο δεδομένων. Η κατανόηση αυτής της συμπεριφοράς και γενικότερα της φύσης των έκτροπων παρατηρήσεων δίνει μία καλύτερη εικόνα για την διαδικασία παραγωγής των δεδομένων. Επομένως, η ανίχνευση έκτροπων παρατηρήσεων, αποτελεί αναπόσπαστο κομμάτι στην στατιστική ανάλυση δεδομένων, το οποίο ρίχνει φως σε εκείνα τα σημεία που δεν συμμορφώνονται με τα υπόλοιπα δεδομένα.

Μετά από μία σύντομη εισαγωγή, στο κεφάλαιο 2 παρουσιάζουμε κάποιες από τις ποικίλες μεθόδους που σχηματίστηκαν για την εύρεση μονοδιάστατων έκτροπων παρατηρήσεων. Επιπλέον αναδεικνύουμε πιθανά προβλήματα που μπορούν να προκληθούν κατά την εύρεση των έκτροπων παρατηρήσεων ακόμη και στο μονοδιάστατο χώρο. Τέλος, εφαρμόζουμε αυτές τις μεθόδους σε ένα σύνολο δεδομένων που έχει παραχθεί από την μονοδιάστατη κανονική κατανομή, προκειμένου να υπογραμμίσουμε αυτά τα προβλήματα και να συγκρίνουμε τα αποτελέσματα.

Στα επόμενα κεφάλαια, παρουσιάζουμε διαφορετικές μεθόδους για πολυδιάστατα δεδομένα, βασισμένες σε διάφορα χαρακτηριστικά, τα βασικότερα από τα οποία είναι: Το βάθος, η απόσταση, η πυκνότητα, η κατανομή και η απόσταση Mahalanobis.

Στο κεφάλαιο 3, γίνεται η παρουσίαση διαφορετικών εννοιών του βάθους και κάποιες από τις μεθόδους που βασίζονται στις αντίστοιχες έννοιες. Πιο αναλυτικά, συζητάμε την έννοια του βάθους όπως ορίστηκε από τον Tukey και τους αλγορίθμους IDODEPTH και FDC που βασίστηκαν σε αυτήν και την έννοια του βάθους όπως ορίστηκε από το Liu που αποτέλεσε την βάση για την Τροποποιημένη Ζώνη Βάθους (Modified Band Depth), την οποία εφαρμόζουμε σε πολυδιάστατα δείγματα.

Στο κεφάλαιο 4, παρουσιάζουμε μεθόδους ανίχνευσης έκτροπων παρατηρήσεων, που βασίζονται στην απόσταση μεταξύ των δεδομένων. Επιπροσθέτως, παρουσιάζουμε τον πρώτο χρονικά ορισμό της έκτροπης παρατήρησης που προκύπτει εξαιτίας της απόστασης της από τα υπόλοιπα δεδομένα, όπως και τον ορισμό που χρησιμοποιείται πιο πολύ από τους σύγχρονους αλγορίθμους και βασίζεται στην k-οστή κοντινότερη γειτονιά. Στην συνέχεια, παρουσιάζουμε κάποιες μεθόδους που μειώνουν σημαντικά τον χρόνο ανίχνευσης εκείνων των έκτροπων παρατηρήσεων, προκειμένου οι αντίστοιχοι αλγόριθμοι να είναι αποτελεσματικοί και σε μεγαλύτερα σύνολα δεδομένων. Επιπλέον παρουσιάζουμε τον ορισμό του αντίστροφου k-οστού κοντινότερου γείτονα από τον Hautamaki. Τέλος, εφαρμόζουμε την μέθοδο που χρησιμοποιεί τον πρώτο χρονικά ορισμό, καθώς και δύο μεθόδους βασισμένες στον υπολογισμό των k κοντινότερων αποστάσεων, όπως και την μέθοδο που βασίζεται στον ορισμό του Hautamaki.

Στο κεφάλαιο 5, παραθέτουμε μεθόδους ανίχνευσης έκτροπων παρατηρήσεων που λαμβάνουν υπόψιν την τοπική πυκνότητα κάθε παρατήρησης. Παρουσιάζουμε την μέθοδο του τοπικού παράγοντα έκτροπης παρατήρησης (Local Outlier Factor), που χαρακτηρίζεται ως η βάση των πιο γνωστών μεθόδων πυκνότητας, μαζί με την εύρωστη επέκταση της εν ονόματι Robust Kernel Outlier Factor (RKOF). Στο ίδιο κεφάλαιο εφαρμόζουμε αυτές τις μεθόδους και συγκρίνουμε τα αποτελέσματα μεταξύ τους.

Στο κεφάλαιο 6, αναπτύσσουμε μεθόδους οι οποίες αναδεικνύουν έκτροπες παρατηρήσεις με βάση την απόσταση Mahalanobis. Πιο συγκεκριμένα παρουσιάζουμε την κλασσική απόσταση Mahalanobis καθώς και μία πιο εύρωστη εκδοχή της. Η κύρια διαφορά τους βρίσκεται στο γεγονός πως η πρώτη υπολογίζεται για κάθε σημείο χρησιμοποιώντας τις εκτιμήσεις του μέσου και της διασποράς όταν όλες οι παρατηρήσεις έχουν ληφθεί υπόψιν, ενώ η δεύτερη χρησιμοποιεί εκείνες τις εκτιμήσεις που έχουν προέλθει από ένα «βέλτιστο» υπόσυνολο παρατηρήσεων. Εκτός από το ποσοστιμόριο της χ^2 κατανομής που συνήθως χρησιμοποιείται σαν φράγμα (απόστασεις που ξεπερνούν το οποίο, θεωρείται πως αντιστοιχούν σε έκτροπες παρατηρήσεις), παρουσιάζουμε την επέκταση του Filzmoser. Τέλος, δοκιμάζουμε αυτές τις μεθόδους και συγκρίνουμε τα αποτελέσματα.

Στο 7ο κεφάλαιο, παρουσιάζουμε μεθόδους που βασίζονται στην υπόθεση πως τα εκάστοτε δεδομένα προέχονται από γνωστή κατανομή. Οι περισσότερες από αυτές τις μεθόδους είναι αποτελεσματικές σε δεδομένα τα οποία προσεγγίζουν την κανονική κατανομή. Επιπροσθέτως, συγκρίνουμε αυτές τις μεθόδους με την βοήθεια του πακέτου 'OutliersO3'.

Τέλος, στα κεφάλαια 8 και 9 εφαρμόζουμε όλες αυτές τις μεθόδους στα δεδομένα 'humus' και 'dat' αντίστοιχα. Δοκιμάζοντας όλες τις προαναφερθέντες μεθόδους σε διαφορετικά δεδομένα, λαμβάνουμε μια πληρέστερη εικόνα σχετικά με την ανίχνευση έκτροπων παρατηρήσεων, την οποία παραθέτουμε στο κεφάλαιο 10.

Η παρακάτω περίληψη χωρίζεται σε διαφορετικά κεφάλαια, με σκοπό να συνοψίσει την αρχική μελέτη τονίζοντας τα σημαντικότερα σημεία.

Περιεχόμενα

1 Εισαγωγή

Με το πέρασμα των χρόνων, πολλοί ορισμοί έχουν δοθεί σχετικά με το τι είναι μία έκτροπη παρατήρηση. Ο Hawkins ορίζει ως έκτροπη, 'εκείνη την παρατήρηση που αποκλείνει τόσο πολύ από τις υπόλοιπες παρατηρήσεις ώστε να δημιουργεί την υπόνοια πως έχει προέλθει από διαφορετικό μηχανισμό'. Ο Barnett και ο Lewis ορίζουν ως έκτροπη παρατήρηση σε μία συλλογή δεδομένων, 'εκείνη τη παρατήρηση (ή το σύνολο παρατηρήσεων) που φαίνεται πως είναι αντιφατική με τις υπόλοιπες παρατηρήσεις της συλλογής'.

Συμπερασματικά, μία εκτροπη παρατήρηση παρουσιάζει, εν γένει, κάποιου είδους αλλοπρόσαλλη συμπεριφορά. Πιο συγκεκριμένα, η δημιουργία μιας έκτροπης παρατήρησης προκύπτει από την ασυνήθιστη συμπεριφορά της αναπαραγωγικής διαδικασίας των δεδομένων, οπότε, κατανοώντας την φύση των δεδομένων λαμβάνουμε γνώση για την διαδικασία αυτή.

Η ανίχνευση έκτροπων παρατηρήσεων, μπορεί να φανεί χρήσιμη σε πολλές εφαρμογές όπως:

- **Ιατρική Διάγνωση:** Σε πολλές ιατρικές εφαρμογές τα δεδομένα συλλέγονται από μία ποικιλία μηχανημάτων, ασυνήθιστα μοτίβα των οποίων, υποδεικνύουν κάποιου είδους νοσολογικής κατάστασης.

- **Απάτη Πιστωτικών Καρτών:** Έστω πως έχει υποκλαπεί ο αριθμός και ο αντίστοιχος κωδικός πιστωτικής κάρτας ενός χρήστη. Το γεγονός αυτό, μπορεί να οδηγήσει σε ανεξήγητες αγορές, από τοποθεσίες στις οποίες ο ίδιος ο χρήστης δεν θα ήταν πιθανό να είχε επισκεφτεί! Τα έξοδα αυτά καθιστώνται ως ασυνήθιστα δεδομένα, αναφορικά με την υπόλοιπη συλλογή αντίστοιχων εξόδων (τα οποία έχουν συλλεχθεί από την εκάστοτε τράπεζα) του χρήστη.

- **Δίκτυα Αισθητήρων:** Σε ένα δίκτυο αισθητήρων, καταγράφονται συνθήκες όπως η υγρασία, η πίεση, δονήσεις, ποσοστά μόλυνσης κλπ σε διαφορετικές τοποθεσίες. Η ανίχνευση έκτροπων παρατηρήσεων σε αυτή την περίπτωση χρησιμοποιείται προκειμένου να εντοπιστούν περίεργες ή λάθος ενδείξεις.

2 Μονοδιάστατη Περίπτωση

Η ανίχνευση έκτροπων παρατηρήσεων μπορεί να χωριστεί στις εξής κατηγορίες:

- Ελέγχους, οι οποίοι βασίζονται σε μία μηδενική υπόθεση H_0 (**formal tests**)
- Ελέγχους, οι οποίοι βασίζονται στις εκτιμήσεις του μέσου και της διασποράς (**informal tests**)

Βασικά μειονεκτήματα της πρώτης κατηγορίας είναι:

- **Masking Effect** (αδυναμία χαρακτηρισμού μίας παρατήρησης ως έκτροπης)
- **Swamping Effect** (ψευδής χαρακτηρισμός μίας παρατήρησης ως έκτροπης).

2.1 Μέθοδοι στην Μονοδιάστατη Περίπτωση

Βασικές μέθοδοι στην μονοδιάστατη περίπτωση αποτελούν:

- Z-score
- Ο γενικευμένος έλεγχος του Grubb (Generalized extreme Studentized deviate Test)
- Η μέθοδος του Tukey (Boxplot)

2.1.1 Z-score

Έστω πως εξετάζουμε μονοδιάστατες παρατηρήσεις, $x_1, x_2, x_3, ..x_n$ με μέσο μ και τυπική απόκλιση σ. Η Z-τιμή για το σημείο x_i συμβολίζεται με z_i και ορίζεται ως εξής:

$$z_i = \frac{|x_i - \mu|}{\sigma}$$

Ο Z-score έλεγχος, υπολογίζει τον αριθμό των τυπικών αποκλίσεων από τον οποίο κάθε σημείο διαφέρει από τον μέσο.

Έκτροπες ορίζονται οι παρατηρήσεις, οι οποίες υπερβαίνουν τον κανόνα:

$$z_i \geq 3$$

.

2.1.2 Η μέθοδος του Tukey (Boxplot)

Ένα τυπικό Boxplot περιέχει:

- Το εσωτερικό εύρος τεταρτημορίων (Inter Quartile Range-IQR) που είναι η απόσταση μεταξύ του κατώτερου Q_1 και του ανώτερου Q_3 τεταρτημορίου. Ένα «κουτί» διαγράφει αυτό το εύρος αυτό και η διάμεσος παρουσιάζεται μέσα του με ένα ευθύγραμμο τμήμα.

- Τα λεγόμενα «μουστάκια» (whiskers) τα οποία επεκτείνονται από τις δύο απέναντι πλευρές του κουτιού μέχρι την ελάχιστη και μέγιστη τιμή της συλλογής αντίστοιχα, στο πλαίσιο των λεγόμενων inner fences (εσωτερικοί φράχτες), τα οποία βρίσκονται σε απόσταση $1.5 \times IQR$ κάτω από το Q_1 και πάνω από το Q_3

- Τα λεγόμενα outer fences (εξωτερικοί φράχτες) τα οποία βρίσκονται σε απόσταση $3 \times IQR$ πάνω από το Q_1 και κάτω από το Q_1.

Οποιαδήποτε τιμή στο γράφημα αυτό, που παρουσιάζεται εκτός των whiskers και είναι ανάμεσα στα inner και outer fences χαρακτηρίζεται ως έκτροπη.

2.2 Εφαρμογή

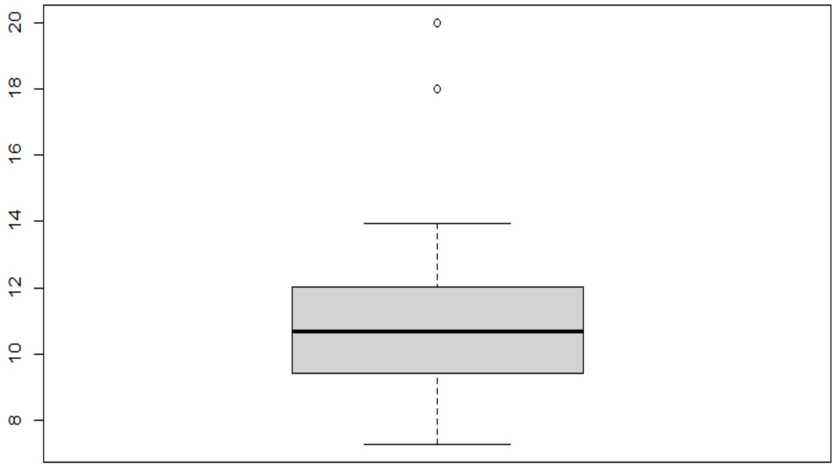
Παράγουμε παρατηρήσεις από την Κανονική κατανομή με μέσο $\mu = 10$ και διασπορά $\sigma = 2$ και αντικαθιστούμε, τυχαία, τις δύο τελευταίες παρατηρήσεις του δείγματος με τις ακραίες τιμές 18 και 20, που προφανώς απέχουν αρκετά πολύ από τον μέσο ως προς και την τυπική τους απόκλιση.

Όπως φαίνεται από τον Πίνακα 1, μόνο αφού αποκόψουμε την παρατήρηση 30, ο z-score έλεγχος (οι τιμές του οποίου, συμβολίζονται ως z_i') επιτυγχάνει να αναδείξει την παρατήρηση 29 ως έκτροπη.

ID	x_i	z_i	ID	x_i	z_i	z_i'	z_i'
1	12.29	0.48	16	8.88	0.75	0.73	0.79
2	7.27	1.33	17	12.51	0.56	1.52	0.83
3	11.33	0.13	18	12.04	0.39	0.31	0.62
4	13.94	1.07	19	7.61	1.21	1.48	1.36
5	12.57	0.58	20	10.07	0.32	0.86	0.26
6	12.22	0.45	21	9.42	0.56	0.70	0.55
7	7.84	1.13	22	10.72	0.09	1.26	0.03
8	9.75	0.44	23	8.86	0.76	0.40	0.8
9	9.88	0.39	24	11.39	0.15	0.35	0.33
10	9.98	0.36	25	10.09	0.31	0.30	0.25
11	10.86	0.04	26	9.78	0.43	0.09	0.39
12	8.1	1.03	27	11.98	0.37	1.14	0.6
13	7.71	1.17	28	10.62	0.12	1.32	0.01
14	12.03	0.38	29	18	2.54	0.62	**3.3**
15	11.13	0.06	30	20	**3.26**	0.22	

Πίνακας 1: z-τιμές του «μολυσμένου» κανονικού δείγματος πριν και μετά την αποκοπή της παρατήρησης 30.

Αντιθέτως, στην Εικόνα 1, η μέθοδος του Tukey αναδεικνύει τις παρατηρήσεις 29 και 30 χωρίς κάποια επιπλέον ενέργεια.



Εικόνα 1: Το Boxplot του «μολυσμένου» κανονικού δείγματος.

Παρατηρούμε ότι:

- Στο Z-score έλεγχο, η παρατήρηση 30 «κρύβει» (masking effect) την παρατήρηση 29, λόγω της ευαισθησίας του ελέγχου, στην παρουσία περισσότερων από μίας έκτροπης παρατήρησης, με αποτέλεσμα η τελευταία να μην μπορεί να ανιχνευθεί.
- Το Boxplot καταφέρνει να ανιχνεύσει και τις δύο έκτροπες παρατηρήσεις, επειδή βασίζεται σε εύρωστα μέτρα όπως το κατώτερο και ανώτερο τεταρτημόριο και το IQR.

3 Πολυδιάστατη Περίπτωση

Παρουσιάζουμε διάφορες μεθόδους για την πολυδιάστατη περίπτωση, βασισμένες σε διάφορα χαρακτηριστικά, οι οποιές μπορούν να χωριστούν στις εξής μεγάλες κατηγορίες:

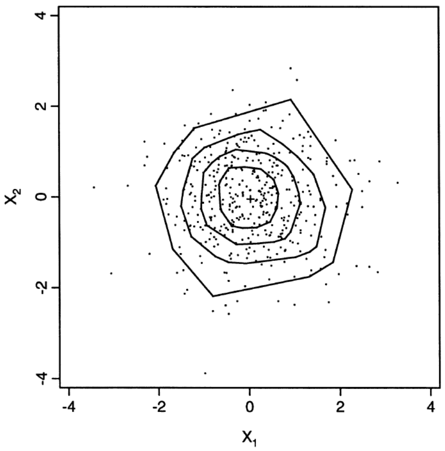
- Μέθοδοι που βασίζονται στο βάθος.
- Μέθοδοι που βασίζονται στην απόσταση.
- Μέθοδοι που βασίζονται στην πυκνότητα.
- Μέθοδοι που βασίζονται στην απόσταση. Mahalanobis.
- Μέθοδοι που βασίζονται σε μία κατανομή.

3.1 Μέθοδοι που βασίζονται στο βάθος

Οι μέθοδοι που βασίζονται στο βάθος, αποτελούν αναπόσπαστο κομμάτι μεθόδων που ειδικεύονται στις ακραίες τιμές.

Σκοπός των μεθόδων του βάθους

- Οι περισσότεροι μέθοδοι ανίχνευσης έκτροπων παρατηρήσεων αφορούσαν την μονοδιάστατη περίπτωση, καθιστώντας τες απαγορευτικές για πολυδιάστατη ανάλυση δεδομένων.
- Όσες μέθοδοι ήταν πολυδιάστατες, βασίζονταν σε μία μηδενική υπόθεση, κάτι το οποίο δεν ανταποκρίνεται σε δεδομένα που δεν έχουν προέλθει από κάποια κατανομή.



Εικόνα 2: Ανίχνευση έκτροπων παρατηρήσεων, με βάση το βάθος.

3.1.1 Τροποποιημένη Ζώνη Βάθους

- Το μέτρο της τροποποιημένης ζώνης βάθους MBD (modified band depth measure) προτάθηκε από την Lopez-Pintado το 2010.

- Είναι μία επέκταση της απλής συνάρτησης βάθους (simplicial depth function) του Liu, που μετράει το βάθος κάθε παρατήρησης από τον αριθμό των κυρτών πολυγώνων (τα οποία έχουν σαν κορυφές τα ίδια τα σημεία) που την περικλείουν αντίστοιχα.

Ορισμός 3.1. Έστω $P = (\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n)$, όπου $\mathbf{p}_i \in \mathbb{R}^d$ ένα d-διάστατο σημείο. Η τροποποιημένη ζώνη βάθους του σημείου $\mathbf{p}$ είναι

$$MBD(\mathbf{p}) =$$

$$\frac{1}{d}\sum_{k=1}^{d}\binom{n}{2}^{-1}\sum_{1\leq i_1<i_2\leq n}\mathcal{I}\{min\{\mathbf{p}_{i_1}(k), \mathbf{p}_{i_2}(k)\} \leq \mathbf{p}(k) \leq max\{\mathbf{p}_{i_1}(k), \mathbf{p}_{i_2}(k)\}\}$$

όπου $\mathcal{I}(\cdot)$ είναι η δείκτρια συνάρτηση και $\mathbf{p}(k)$ είναι η k-οστή συντεταγμένη του p.

3.2 Μέθοδοι που βασίζονται στην απόσταση

- Οι μέθοδοι του βάθους είναι υπολογιστικά ακριβοί.

- Σημεία του συνόλου, με μεγάλες αποστάσεις k-κοντινότερων γειτόνων ορίζονται ως έκτροπα.

- Παρουσιάζουμε τρεις διαφορετικούς ορισμόυς, βάσει των οποίων αναπτύχθηκαν τρεις διαφορετικές μέθοδοι που προτάθηκαν από τον Knor και τον Ng (1998), τον Ramaswamy (2000), και τον Hautamaki (2004).

Ορισμός 3.2 (Προσέγγιση του Knor)**.** Ένα σημείο O σε ένα σύνολο T είναι $DB(p, D)$ έκτροπη παρατήρηση (DB προέρχεται από τη φράση Distance Based), αν τουλάχιστον ένα ποσοστό p σημείων στην T απέχει απόσταση μεγαλύτερη από D από το σημείο O.

Ορισμός 3.3 (Προσέγγιση του Ramaswamy)**.** Έστω σύνολο δεδομένων με N σημεία, παραμέτρους n και k και θέτοντας με $D^k(x)$ την απόσταση ενός σημείου x από τον k-οστό κοντινότερο γείτονα του, ένα σημείο x είναι D_n^k έκτροπο, αν δεν υπάρχουν παραπάνω από $n-1$ άλλα σημεία x' που να ικανοποιούν την σχέση $D_k(x') > D_k(x)$.

Ορισμός 3.4 (Προσέγγιση του Hautamaki)**.** Ένα σημείο x ορίζεται ως αντίστροφος k-οστός κοντινότερος γείτονας του y, αν και μόνο αν το y είναι ένας k-οστος κοντινότερος γείτονας του x.

Παρατηρήσεις

- Η προσέγγιση του Knor επιστρέφει μία διαδική ένδειξη (αν είναι ή οχι μία παρατήρηση, έκτροπη) παρά ένα μέτρο που να δείχνει πόσο έκτροπη μία παρατήρηση μπορεί να είναι.

- Η απόλυτη απόσταση από τον k-οστό κοντινότερο γείτονα ή η μέση απόσταση όλων των k-οστών κοντινότερων γειτόνων μπορεί να χρησιμοποιηθεί ως ένα τέτοιο μέτρο.

- Η προσέγγιση του αντίστροφου k-οστού κοντινότερου γείτονα θεωρείται μία μίξη των μεθόδων που βασίζονται στην απόσταση και στις μεθόδους που βασίζονται στην πυκνότητα.

- Σημεία που έχουν μεγάλες αποστάσεις από τους k-οστούς κοντινότερους γείτονες, θα έχουν επίσης λίγους αντίστροφους k-οστούς κοντινότερους γείτονες.

- Ως έκτροπο, χαρακτηρίζεται το σημείο όπου ο αριθμός των αντίστροφων k-οστών κοντινότερων γειτόνων (indegree number) του οποίου, είναι μικρότερος από ένα ορισμένο από τον χρήστη όριο.

- Μπορεί να γίνει εύκολα κατανοητή η προσέγγιση του Hautamaki στο πλαίσιο του γραφήματος k-οστών κοντινότερων γειτόνων (kNN graph).

3.3 Μέθοδοι που βασίζονται στην πυκνότητα

Στόχος των μεθόδων πυκνότητας

Σύμφωνα με την Εικόνα 3 παρατηρούμε ότι:

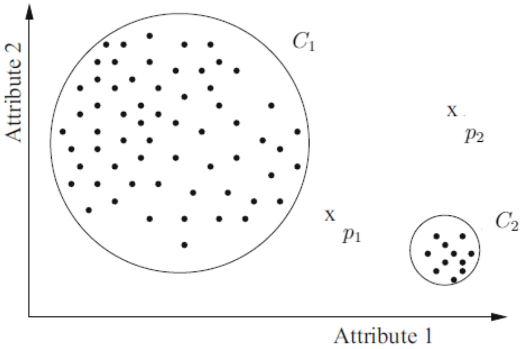
- Και τα δύο σημεία p_1 και p_2 είναι έκτροπα.

- Μία κλασσική μέθοδος απόστασης πιθανότατα να ανίχνευε το σημείο p_2.

- Το σημείο p_1 είναι αρκετά κοντα στα κανονικά σημεία του συμπλέγματος (cluster) C_1 ώστε να θεωρηθεί και εκείνο κανονικό.

- Αν επιλέξουμε ένα πιο αυστηρό όριο όσον αφορά την απόσταση, προκειμένου να αναδείξουμε το p_1, είναι πολύ πιθανό, η μέθοδος (του Knor) να ανίχνευε παραπάνω σημεία εκτός του p_1.

- Αυτές οι μέθοδοι αναπτύχθηκαν για να αντιμετωπίσουν τα προβλημάτα τοπικής πυκνότητας.



Εικόνα 3: Απεικόνιση τοπικά έκτροπης παρατήρησης.

- Σε ένα τυπικό πλαίσιο μεθόδου πυκνότητας, υπάρχουν δύο παράμετροι που χαρακτηρίζουν την έννοια της πυκνότητας ενός σημείου.

  (α΄) Η παράμετρος **MinPts** που αναφέρεται στον ελάχιστο αριθμό σημείων και

  (β΄) μία παράμετρος που να αναφέρεται στην ένταση κάτω από την οποία διερευνάται η τοπική πυκνότητα.

- Παρουσιάζουμε δύο μεθόδους:

  (α΄) Το LOF (Local Outlier Factor), από τον Breunig (2000).

  (β΄) Το RKOF (Robust Kernel Outlier Factor), από τον Gao (2013).

- Το LOF είναι ένα μέτρο διαφοράς πυκνότητας μεταξύ ενός σημείου και της αντίστοιχης γειτονιάς του. Η μέθοδος LOF χρησιμοποιεί σαν εκτίμηση πυκνότητας την απόσταση τοπικής προσιτότητας (local reachability distance).

- Το RKOF αποτελεί μία επέκταση του LOF. Η μέθοδος του RKOF χρησιμοποιεί μία εκτίμηση πυκνότητας πυρήνα (kernel) για τον υπολογισμό των παραγόντων των έκτροπων παρατηρήσεων.

- Και οι δύο μέθοδοι μετρούν τον βαθμό εκτροπής κάθε παρατήρησης. Όσο μεγαλύτερη είναι αυτή η τιμή τόσο πιο πιθανό είναι οι αντίστοιχες παρατηρήσεις να είναι έκτροπες.

3.3.1 Παράγοντας Τοπικά Έκτροπης Παρατήρησης

Ορισμός 3.5. Έστω k ένας φυσικός αριθμός. Η απόσταση προσιτότητας (reachability distance) ενός σημείου p ως προς το σημείο o ορίζεται ως $rd(p, o) = \max\{d_k(o), d(p, o)\}$.

Ορισμός 3.6. Η απόσταση τοπικής προσιτότητας (local reachability distance) του p (συμβολίζοντας την ως $lrd_k(p)$) ορίζεται ως

$$lrd_\varkappa(p) = \left(\frac{\sum_{o \in N_k(p)} rd_k(p, o)}{|N_k(p)|}\right)^{-1}$$

Ορισμός 3.7. Ο παράγοντας (τοπικής) έκτροπης παρατήρησης του p ορίζεται ως

$$LOF_k(p) = \frac{\sum_{o \in N_k(p)} \frac{lrd(o)}{lrd(p)}}{|N_k(p)|}$$

Παρατηρήσεις

- Η απόσταση τοπικής προσιτότητας λειτουργεί σαν μία εκτίμηση πυκνότητας στο σημείο p αναλύοντας τις k-οστές κοντινότερες αποστάσεις των σημείων στην $N_k(p)$ γειτονιά.

- Τα σημεία που βρίσκονται στην πιο πυκνή περιοχή του συμπλέγματος τείνουν να έχουν τιμές LOF ίσες με την **μονάδα**.

- Όσο υψηλότερες τιμές $LOF(p)$ τόσο «περισσότερο έκτροπες» είναι οι παρατηρήσεις που τις σημειώνουν.

3.3.2 Robust Kernel Outler Factor

Στην μέθοδο RKOF χρησιμοποιείται το ίδιο πλαίσιο, με πιο αποτελεσματικές εκτιμήσεις πυκνότητας.

Στη μονοδιάστατη περίπτωση, η εκτίμηση πυρήνα ορίζεται ως:

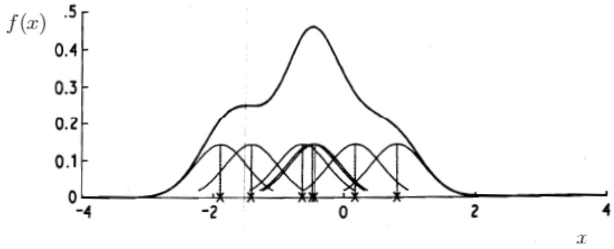
$$\hat{f}(x) = \frac{1}{n \cdot h} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right),$$

όπου

- το X_i συμβολίζει την i-οστή παρατήρηση.

- το h συμβολίζει το πλάτος (ορίζεται επίσης ως παράγοντας ομαλοποίησης).

- Το $K(\cdot)$ είναι η συνάρτηση πυρήνα που ικανοποιεί την σχέση $\int_{-\infty}^{+\infty} K(x)dx = 1$.



Εικόνα 4: Εκτίμηση πυρήνα από μεμονωμένους πυρήνες πλάτους 0.4.

- Επιτρέποντας στην παράμετρο h των «εξογκομάτων» να διαφέρουν ανάλογα με τις παρατηρήσεις για τις οποίες σχηματίζονται, ορίζουμε την εκτίμηση πυρήνα ως:

$$\hat{f}(x) = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{h \cdot d_k(X_j)} \cdot K\left(\frac{x - X_j}{h \cdot d_{j,k}}\right),$$

- Όταν το k είναι σταθερό, μόνο η παράμετρος h είναι υπεύθυνη για το βαθμό ομαλοποίησης.

- Η επιλογή του k καθορίζει το βαθμό της ευαισθησίας του h **τοπικά**.

Στην πολυδιάστατη περίπτωση, η συνάρτηση πυρήνα $K(\cdot)$ είναι μία συνάρτηση ορισμένη στον d-διάστατο χώρο που ικανοποιεί την σχέση:

$$\int_{\mathbb{R}^d} K(\mathbf{x})dx = 1, \mathbf{x} \in \mathbb{R}^d$$

Οι πιο συνήθεις συναρτήσεις πυρήνα, είναι η Γκαουσιανή

$$K(\mathbf{x}) = (2 \cdot \pi)^{-\frac{d}{2}} \cdot \exp(-\frac{1}{2}\mathbf{x}^T\mathbf{x})$$

και η Επανέτσικοβ

$$K(\mathbf{x}) = \begin{cases} \frac{1}{2} \cdot c_d^{-1} \cdot (d-2) \cdot (1 - \mathbf{x}^T\mathbf{x}) & , \text{ιφ } \mathbf{x}^T\mathbf{x} < 1 \\ 0 & , \text{οτηερωισε} \end{cases}$$

όπου c_d είναι ο όγκος της μονοδιαίας d-διάστατης σφαίρας.

Ορισμός 3.8. Η εκτίμηση της τοπικής πυκνότητας πυτήνα (local kernel density) του p ορίζεται ως:

$$kde(p) = \frac{\sum_{o\in N_k(p)} \frac{1}{(C\cdot d_k(o)^\alpha)^\gamma} \cdot K\Bigg(\frac{p-o}{C\cdot d_k(o)^\alpha}\Bigg)}{|N_k(p)|}, \, C = h\cdot g^\alpha,$$

$$\log g = \frac{\sum_{q\in D} \log f(q)}{|D|}, \, f(o) = \frac{1}{d_k(o)}$$

όπου το h είναι ο παράγοντας ομαλοποίησης, το γ είναι παράμετρος ευαισθησίας, το $K(\cdot)$ είναι η πολυδιάστατη συνάρτηση πυρήνα και το α είναι παράμετρος ευαισθησίας που ικανοποιεί τη σχέση $0 \leq \alpha \leq 1$, όπου το g είναι ο γεωμετρικός μέσος του $f(q)$ και το $f(\cdot)$ είναι η εκτίμηση της πυκνότητας του κοντινότερου γείτονα.

Ορισμός 3.9. Ο σταθμισμένος εκτιμητής πυκνότητας (weighted density estimate) της γειτονιάς του p ορίζεται ως:

$$wde(p) = \frac{\sum_{o\in N_k(p)} \omega_o \cdot kde(o)}{\sum_{o\in N_k(p)} \omega_o}, \, \omega_o = \exp\Bigg\{ - \frac{\big(\frac{d_k(o)}{d_{k_{min}}} - 1\big)^2}{2\cdot\sigma^2}\Bigg\}$$

όπου το ω_o είναι το βάρος του σημείου o στην γειτονιά των k-οστών κοντινότερων σημείων του p, το σ είναι η διακύμανση με προδιαγραμμένη τιμή την μονάδα και $d_{k_{min}} = min\{d_n(o)|o \in N_k(p)\}$.

Ορισμός 3.10. Ο παράγοντας έκτροπης παρατήρησης RKOF του p ορίζεται ως:

$$RKOF(p) = \frac{wde(p)}{kde(p)},$$

όπου $wde(p)$ είναι η εκτίμηση πυκνότητας της γειτονιάς k-οστών κοντινότερων σημείων του p και $kde(p)$ είναι η εκτίμηση της τοπικής πυκνότητας του p.

- **Παράμετροι** : k, C, α, σ^2.
- Προδιαγραμμένες τιμές C, α και σ^2 είναι ίσες με την **μονάδα**.
- Όσο μεγαλύτερη η τιμή RKOF, τόσο πιο πιθανό μία παρατήρηση να είναι έκτροπη.

3.4 Μέθοδοι που βασίζονται στην απόσταση Mahalanobis

Έστω δύο σημεία ορισμένα από τα διανύσματα $\mathbf{x}_A = (x_{A_1}, x_{A_2}, ..., x_{A_d})$, $\mathbf{x}_B = (x_{B_1}, x_{B_2}, ..., x_{B_d})$.

Ορισμός 3.11. Η **Ευκλείδεια απόσταση** ορίζεται ως εξής:

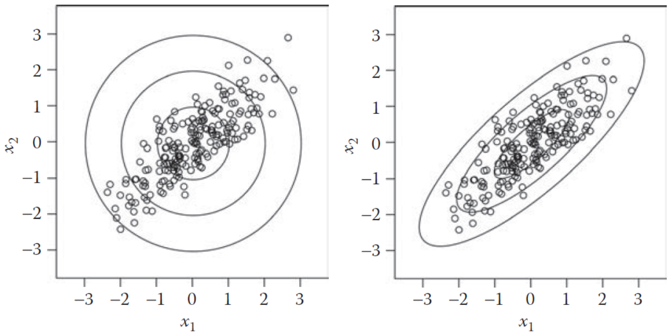
$$d_{Euclidean}(\mathbf{x}_A, \mathbf{x}_B) = \left(\sum_{j=1}^{d} (x_{B_j} - x_{A_j})^2 \right)^{\frac{1}{2}} = (\mathbf{x}_B - \mathbf{x}_A)^T (\mathbf{x}_B - \mathbf{x}_A)$$

Ορισμός 3.12. Η απόσταση Mahalanobis μεταξύ δύο d-διάστατων σημείων x_A και x_B ορίζονται ως εξής:

$$MD(\mathbf{x}_A, \mathbf{x}_B) = \sqrt{[(\mathbf{x}_B - \mathbf{x}_A)^T \cdot \mathbf{S}^{-1\cdot} (\mathbf{x}_B - \mathbf{x}_A)]}$$

όπου $\mathbf{S}$ είναι ο πίνακας συνδιακύμανσης. Ο υπολογισμός της απόστασης Mahalanobis απαιτεί την αντιστοφή του πίνακα συνδιακύμανσης $\mathbf{S}$.

- Οι αποστάσεις Mahalanobis όλων των σημείων από το κέντρο σχηματίζουν ελλείψεις οι οποίες είναι πιο συγκεντρωμένες, ως προς το δείγμα, από τους κύκλους οι οποίοι σχηματίζονται από από τις Ευκλείδειες αποστάσεις των σημείων από το κέντρο.
- Η απόσταση Mahalanobis λαμβάνει υπόψιν την συσχέτηση μεταξύ των μεταβλητών, μέσω του πίνακα συνδιακύμανσης.



Εικόνα 5: Η Ευκλείδεια απόσταση (αριστερά) και η απόσταση Mahalanobis (δεξία).

Ορισμός 3.13. Η συνάρτηση πυκνότητας πιθανότητας της Κανονικής κατανομής $\varphi(\boldsymbol{\xi})$ σε ένα d-διάστατο σημείο $\boldsymbol{\xi}$ ορίζεται ως εξής:

$$f(\boldsymbol{\xi}) = \frac{1}{\sqrt{|\Sigma|} \cdot (2 \cdot \pi)^{\frac{d}{2}}} \cdot e^{-\frac{1}{2} \cdot (\boldsymbol{\xi} - \mu) \cdot \Sigma^{-1} \cdot (\boldsymbol{\xi} - \mu)^T}$$

Η τιμή στο εκθετικό της Κανονικής κατανομής παραπάνω, χρησιμοποιείται σαν μέτρο εκτροπής.

- Οι τετραγωνισμένες αποστάσεις Mahalanobis ακολουθούν περίπου την χ^2_d κατανομή με d βαθμούς ελευθερίας.

- Αν μία παρατήρηση σημειώσει απόσταση Mahalanobis μεγαλύτερη από ένα προκαθορισμένο όριο, τότε μπορεί να θεωρηθεί **πιθανή πολυδιάστατη έκτροπη παρατήρηση**.

- Συνηθισμένο ποσοστημόριο της χ^2_d κατανομής, θεωρείται αυτό της τάξης του 97.5% με $x^2_{d,0.975}$ σαν τιμή του ορίου.

- Ένα πιο εύρωστο μέτρο, μπορεί να αποκτηθεί από έναν εύρωστο εκτιμητή διακύμανσης όπως τον εκτιμητή της ελάχιστης ορίζουσας του πίνακα διακύμανσης (**minimum covariance determinant (MCD) estimator**).

Γραφική Προσέγγιση

- Σε χαμηλότερες διαστάσεις, μία έλλειψη/ένα ελλειψοειδές σχηματίζεται από τις αποστάσεις Mahalanobis. Τα σημεία εκτός αυτού του ελλειψοειδούς χαρακτηρίζονται ως έκτροπες παρατηρήσεις.

- Ένα γράφημα (index plot) παρουσιάζοντας τις τιμές των αποστάσεων Mahalanobis κάθε παρατήρησης και μία γραμμή, υποδηλώνοντας το προκαθορισμένο όριο.

- χ^2 γράφημα, από τον Garrett (1989).

- Adaptive χ^2 γράφημα, από τον Filzmoser (2004).

3.5 Μέθοδοι που βασίζονται σε μία κατανομή

- Σε αυτήν την κατηγορία, οι μέθοδοι στοχεύουν στην ανάδειξη έκτροπων παρατηρήσεων, βασισμένοι σε μία μηδενική υπόθεση.

- Η πιο συνηθισμένη είναι, πως όλες οι παρατηρήσεις έχουν προέλθει από μία πολυδιάστατη Γκαουσιανή κατανομή.

Παρουσιάζουμε έξι αλγορίθμους:

- Τον αλγόριθμο FastMCD, από τους Rousseeuw και Driessen (1999).

- Τον αλγόριθμο BACON, από τον Billor (2000).

- Τον αλγόριθμο FastPCS, από τους Vakili και Schmitt (2014).

- Τον αλγόριθμο HDoutliers, από τον Wilkinson (2018).

- Τον αλγόριθμο Skewness-adjusted Outlierness, από τον Brys (2005).

- Τον αλγόρθμο DDC, από τους Rousseeuw και Bosche (2018).

Αλγόριθμος MCD

- Ο εκτιμητής ελάχιστης ορίζουσας πίνακα διακύμανσης (MCD estimator) προτάθηκε για πρώτη φορά από τον Rousseeuw (1984).

- Θεωρείται ένα από τα μεγαλύτερα εύρωστα εργαλεία που μπορούν να αντιμετωπίσουν προβλήματα masking και swamping.

- Η ιδέα της μεθόδου MCD βασίζεται στο παρακάτω θεώρημα:

Θεώρημα 3.1. *Έστω δείγμα* $N = \{\mathbf{x}_1, ., \mathbf{x}_n\}$*, όπου* $\mathbf{x}_i = (x_{i1}, ..., x_{ip})$ *με* $i = 1, ..., n$ *είναι μία* p*-διάστατη παρατήρηση. Έστω* $H_1 \subset \{1, .., n\}$ *με* $|H_1| = h$ *και* $\mathbf{T}_1 := \frac{1}{h} \cdot \sum_{i \in H_1} \mathbf{x}_i$ *και* $\mathbf{S}_1 := \frac{1}{h} \cdot \sum_{i \in H_i} (\mathbf{x}_i - \mathbf{T}_1) \cdot (\mathbf{x}_i - \mathbf{T}_1)'$*. Αν* $det(\mathbf{S}_1) \neq 0$*, ορίζουμε τις σχετικές αποστάσεις*

$$d_1(i) := \sqrt{(\mathbf{x}_i - \mathbf{T}_1)' \cdot \mathbf{S}_1^{-1} \cdot (\mathbf{x}_i - \mathbf{T}_1)} \quad \varphi o\rho \; i = 1, ..., n.$$

Έστω τώρα, H_2 *τέτοιο ώστε* $\{d_1(i) | i \in H_2\} := \{(d_1)_{1:n}, (d_1)_{2:n}, ..., (d_1)_{h:n}\}$*, όπου* $(d_1)_{1:n} \leq (d_1)_{2:n} \leq ... \leq (d_1)_{h:n}$ *είναι οι διατεταγμένες αποστάσεις και υπολογίζουμε τα μέτρα* $\mathbf{T}_2$ *και* $\mathbf{S}_2$ *βάσει του* H_2*. Τότε*

$$det(\mathbf{S}_2) \leq det(\mathbf{S}_1)$$

με την ισότητα να ισχύει αν και μόνο αν $\mathbf{T}_2 = \mathbf{T}_1$ *και* $\mathbf{S}_2 = \mathbf{S}_1$*.*

- Το Θεώρημα 3.1 δημιουργεί μία επαναληπτική διαδικασία, με την οποία μετακινούμαστε από ζεύγος $(\mathbf{T}_k, \mathbf{S}_k)$ σε ζεύγος $(\mathbf{T}_{k+1}, \mathbf{S}_{k+1})$, παράγοντας υποσύνολα H_{k+1}.

- Οι ορίζουσες είναι μη αρνητικές και ισχύει ότι $det(\mathbf{S}_{k+1}) \leq det(\mathbf{S}_k)$ για κάθε $k \in \mathbb{N}$, οπότε η διαδικασία συγλίνει.

- Ο αλγορθμος MCD εφαρμόζει το Θεώρημα 3.1 σε αρχικές επιλογές υποσυνόλων H_1 μέχρι να επιτευχθεί σύγκλιση και επιλέγει εκείνο το υποσύνολο με την μικρότερη ορίζουσα.

- Τα περισσότερα δείγματα δεν είναι παραπάνω από 25% μολυσμένα (με έκτροπες παρατηρήσεις), οπότε μία καλή επιλογή πληθαρίθμου για το υποσύνολο H είναι η $0.75 \cdot n$.

- Το τελευταίο ζευγάρι χρησιμοποιείται για τον υπολογισμό των εύρωστων αποστάσεων των παρατηρήσεων που οριζονται ως εξής:

$$RD_i = \sqrt{(\mathbf{x}_i - \mathbf{T}_{full})' \cdot \mathbf{S}_{full}^{-1} \cdot (\mathbf{x}_i - \mathbf{T}_{full})} \quad i = 1, ..., n$$

- Η μέθοδος αναδεικνύει εκείνες τις παρατηρήσεις που ξεπερνάνει μία προκαθορισμένη τιμή ορίου (συνήθως η τιμή αυτή είναι $\sqrt{\chi^2_{p,0.975}}$, όπου p ειναι ο αριθμός των μεταβλητών), ως έκτροπες.

Αλγόριθμος BACON

- Ο αλγόριθμος BACON προτάθηκε για πρώτη φορά από τον Billor (2000).
- Όπως και στην περίπτωση του FastMCD, ο αλγόριθμος BACON αναζητά ένα βέλτιστο υποσύνολο παρατηρήσεων και υπολογίζει τις αποστάσεις Mahalanobis ως προς τις εκτιμήσεις του μέσου και της διασποράς εκείνου του υποσυνόλου.
- Οι παρατηρήσεις, όπου ικανοποιούν τη σχέση $d_i < c_{npr} \cdot \chi^2_{p,\frac{\alpha}{n}}$ σχηματίζουν το αρχικό υποσύνολο.
- Η επαναληπτική διαδικασία τερματίζεται, όταν το υποσύνολο δεν αλλαζει περαιτέρω.
- Αφού τερματιστεί η διαδικασία, τα σημεία τα οποία δεν βρίσκονται μέσα σε αυτό το βέλτιστο υποσύνολο, χαρακτηρίζονται ως έκτροπα.

Αλγόριθμος FastPCS

- Η μέθοδος PCS προτάθηκε από τον Vakili (2014).
- Υιοθετεί κάποια βήματα από την μέθοδο FastMCD καθώς στοχεύει να επιλέξει μεταξύ πολλών υποσυνόλων H, εκείνο το υποσύνολο παρατηρήσεων το οποίο είναι κενό από έκτροπες παρατηρήσεις.
- Ο αλγόριθμος PCS αναζητά εκείνο το υποσύνολο H πολυδιάστατων παρατηρήσεων, που ταιριάζει περισσότερο (congruent) υπολογίζοντας τις τετραγωνισμένες ορθογώνιες αποστάσεις από κάθε σημείο προβάλλοντας τα στο μονοδιάστατο χώρο.
- Αφού υπολογιστεί το ζεύγος $(\mathbf{T}_{PCS}, \mathbf{S}_{PCS})$ από το υποσύνολο H_m, η μέθοδος χρησιμοποιεί την ίδια τακτική με τον αλγόριμο FastMCD.

Αλγόριθμοι AO, DDC, HDoutliers

- **AO** (skewness-adjusted outlingness): Το μέτρο αυτό, αποτελεί μία τροποποίηση του μέτρου Stahel-Donoho outlyingness (1981) το οποίο μπορεί να διαχειριστεί το πρόβλημα της λοξότητας (skewness).
- **HDoutliers**: Ο αλγόριθμος αυτός, μπορεί να ανιχνεύσει έκτροπες παρατηρήσεις σε δείγματα τα οποία αποτελούνται από σημεία τεράστιων διαστάσεων (high dimentional data sets) χωρίζοντας τις παρατηρήσεις σε ομάδες (exemplars).
- **DDC**: Ο αλγόριθμος DDC αναζητά έκτροπα σημεία σε ένα $n \times d$ πίνακα , όχι σαν d-διάστατες παρατηρήσεις, αλλά σαν ξεχωριστά στοιχεία πίνακα.

4 Πακέτα στο R

Στην παρούσα μελέτη, έγινε χρήση των πακέτων:

- OutlierDetection, των Tiwary και Kashikar (2019)
- DDoutlier, του Madsen (2018)
- mvoutlier, των Filzmoser και Gschwandtner (2018)
- chemometrics, των Filzmoser και Varmuza (2017)
- OutliersO3, του Unwin (2020)
- depthTools, των Lopez-Pintado και Torrente (2013)

4.1 Συναρτήσεις Πακέτων

Από το πακέτο **DDoutlier**

- DB
- KNN_IN
- LOF
- RKOF

Από το πακέτο **depthTools**

- MBD

Από το πακέτο **chemometrics**

- Moutlier

Από το πακέτο **mvoutlier**

- aq.plot

Από το πακέτο **OutlierDetection**

- depthout
- nn
- nnk
- dens
- maha

Από το πακέτο **OutliersO3**

- O3prep
- O3plotT
- O3plotT

5 Εφαρμογές

Τα δείγματα που εξετάστηκαν είναι:

- Wood specific gravity
- dat
- Humus

5.1 Wood specific gravity

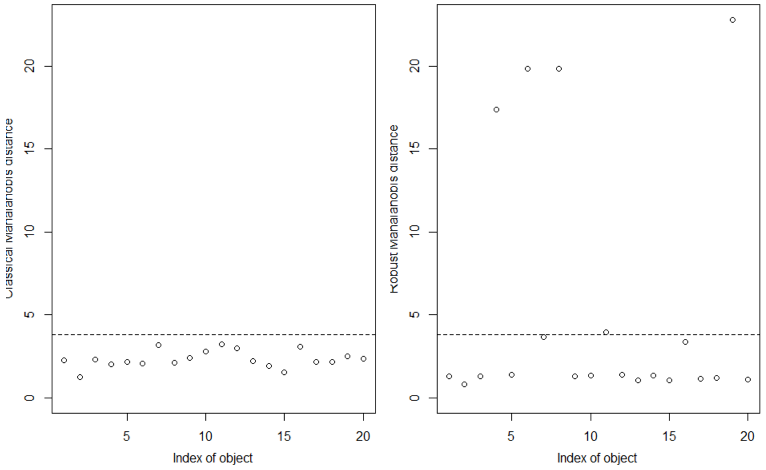
Ο Rousseeuw (1980) «μόλυνε» ένα δείγμα **αντικαθιστώντας** τέσσερις παρατηρήσεις. Οι παρατηρήσεις προέχονται από μετρήσεις που καθορίζουν την επιρροή ανατομικών παραγόντων στη βαρύτητα του ξύλου, με πέντε επεξηγηματικές μεταβλητές και μία σταθερή. Οι παρατηρήσεις **4**, **6**, **8**, και **19** ανιχνεύτηκαν με την χρήση της μεθόδου LMS. Αυτές οι παρατηρήσεις δεν είναι προφανείς έκτροπες από την ανάλυση ελαχίστων τετραγώνων.

Πακέτα	Συναρτήσεις	k	Ποσοστά	Έκτροπες
Outlier Detection	depthout	-	0.25	4, 6, 8, 19
	nn	6	0.75	7, 10, 12
	nnk	6	0.75	4, 6, 8, 19
	dens	10	0.75	4, 6, 8, 19
	maha	-	0.8	7, 11 ,12 ,16
DDoutlier	KNN_IN	9	-	4, 6, 8, 19
	LOF	6	-	4, 6, 8, 19
	RKOF	10	-	4, 6, 8, 19

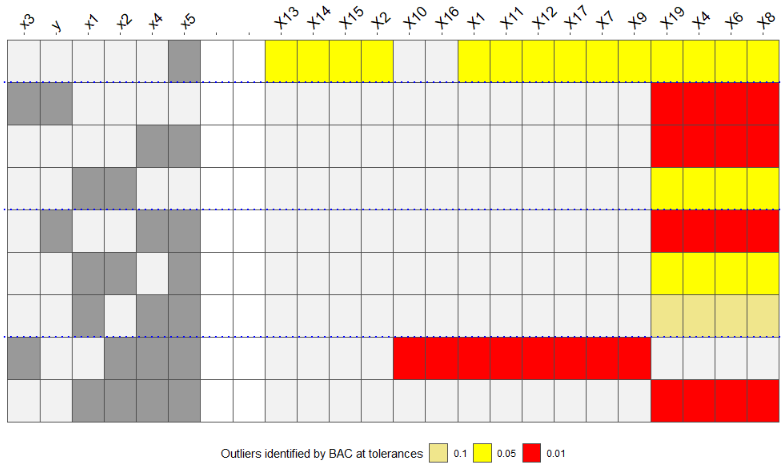
Πίνακας 2: Αριθμός έκτροπων παρατηρήσεων, για διαφορετικές τιμές του k και ποσοστά για τις συναρτήσεις του πακέτου OutlierDetection.

Παρατηρήσεις:

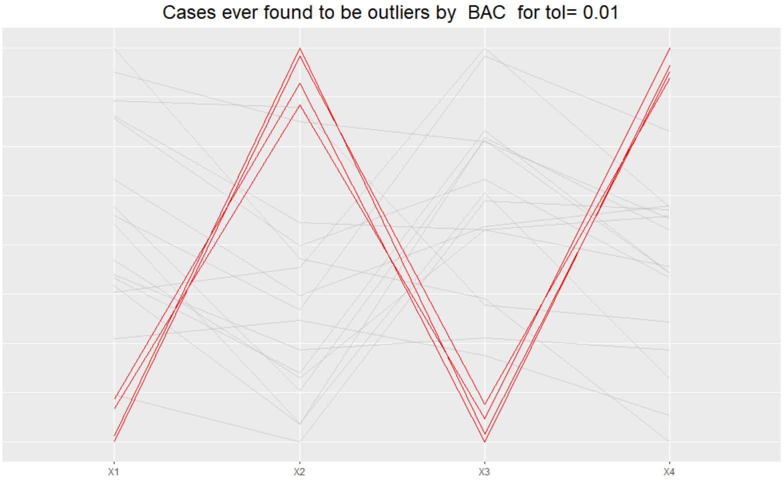
- Για μεγαλύτερες τιμές του k, η nn ανιχνεύει τις ίδιες παρατηρήσεις σαν έκτροπες, όπως η nnk.
- Οι κλασσικές αποστάσεις Mahalanobis επιρρεάστηκαν από την αντικατάσταση των παρατηρήσεων.



Εικόνα 6: Τιμές κλασσικών αποστάσεων Mahalanobis (αριστερά) - Τιμές εύρωστων αποστάσεων Mahalanobis (δεξιά).



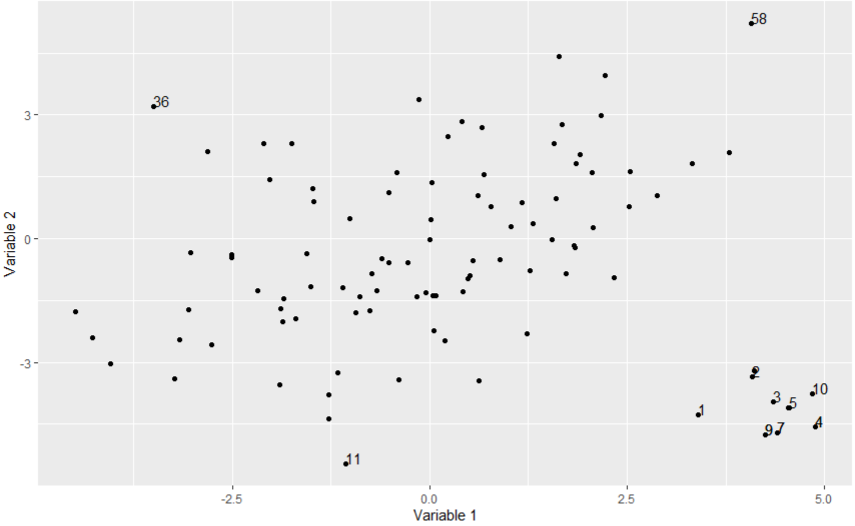
Εικόνα 7: O3 plot από τον αλγόριθμο του BACON για διάφορες τιμές ευαισθησίας.



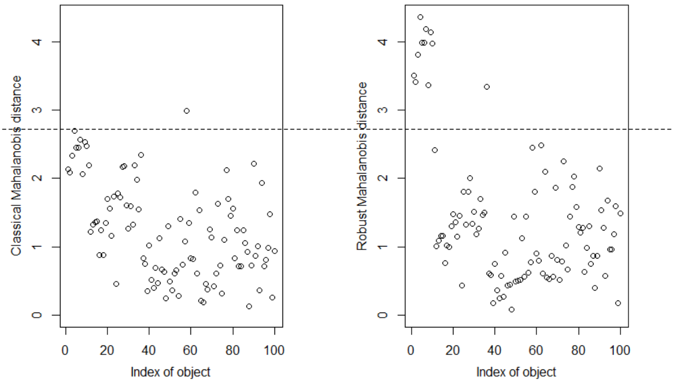
Εικόνα 8: Γράφημα parallel coordinate του δείγματος μόνο με τις μεταβλητές x1, x2, x4, x5.

5.2 Dat

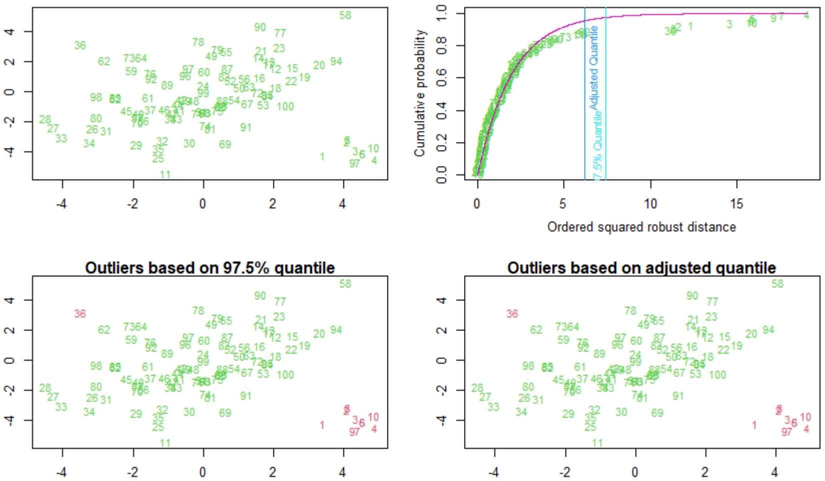
- Το σύνολο δεδομένων dat είναι τεχνητό, αποτελούμενο από παρατηρήσεις που προέρχονται από την δισδιάστατη Κανονική κατανομή.
- Το σύνολο δεδομένων dat βρίσκεται στο πακέτο **mvoutlier**.
- Στο δείγμα αυτό περιέχονται 100 δισδιάστατες παρατηρήσεις.



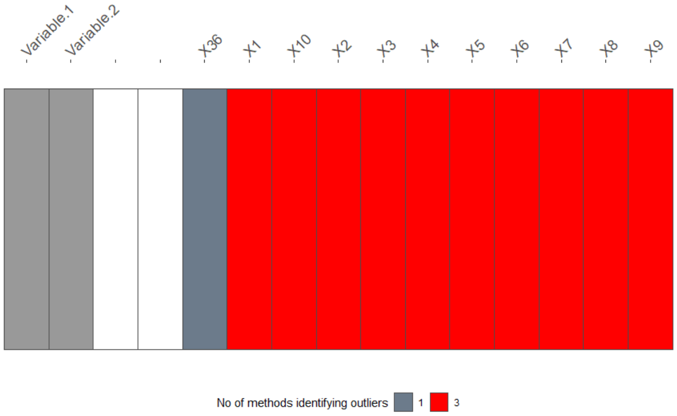
Εικόνα 9: Διάγραμμα διασποράς, με αριθμημένες τις πιθανές έκτροπες παρατηρήσεις.



Εικόνα 10: Γραφήματα που δείχνουν (αριστερά) τις κλασσικές αποστάσεις Mahalanobis και τις εύρωστες αποστάσεις (δεξία) για όριο $\sqrt{\chi^2_{2,0.975}} = 2.72$.



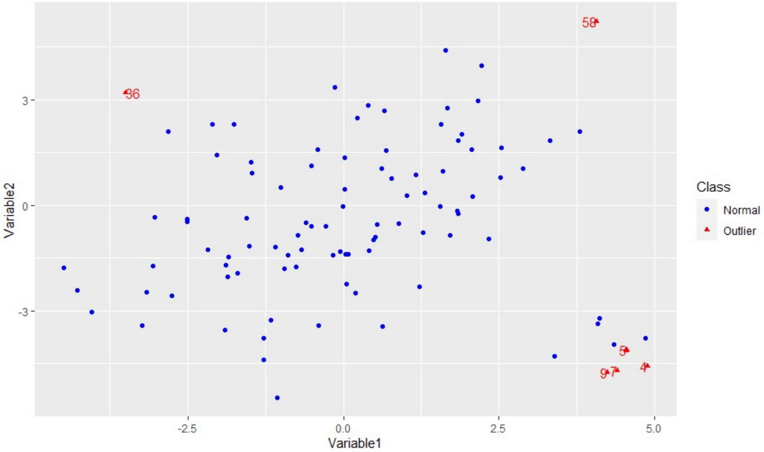
Εικόνα 11: Διάγραμμα πάνω αριστερά: Εύρωστες αποστάσεις Mahalanobis. Διάγραμμα πάνω δεξιά: Εμπειρική κατανομή των εύρωστων αποστάσεων. Κόκκινη γραμμή: Η χ^2 κατανομή και οι δύο κάθετες γραμμές υποδηλώνουν το 97.5% όριο και το προσαρμοστικό (adjusted) όριο αντίστοιχα. Τα δύο κάτω διαγράμματα έχουν με κόκκινο όλα τα έκτροπα σημεία που έχουν ξεπεράσει κάθε όριο αντίστοιχα.



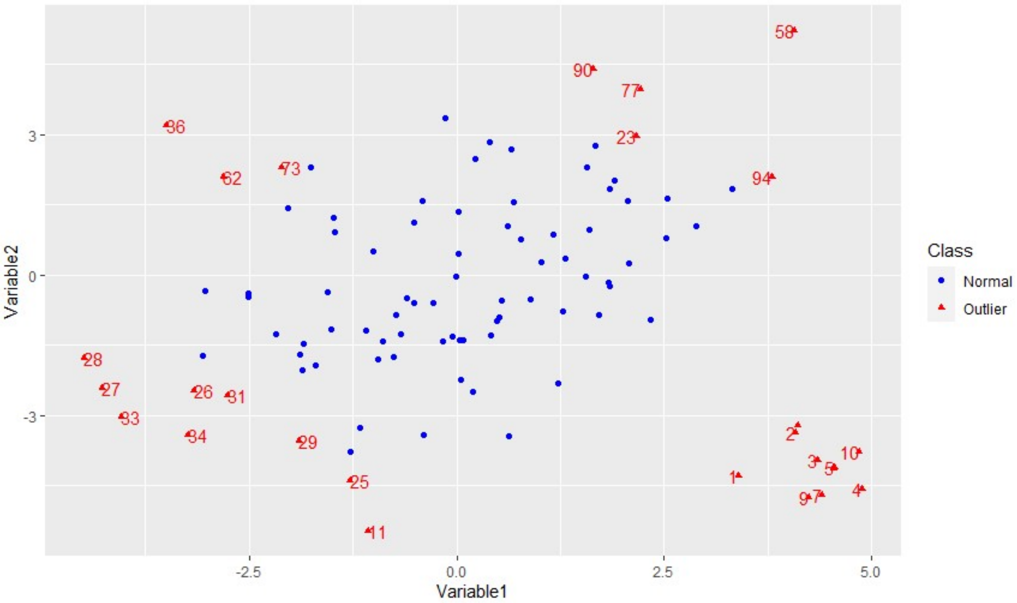
Εικόνα 12: Διάγραμμα σύγκρισης μεθόδων FastMCD, FastPCS και BACON.

Παρατηρήσεις

- Η κλασσική απόσταση Mahalanobis ανέδειξε μόνο την παρατήρηση 58 σαν έκτροπη.
- Οι εύρωστες αποστάσεις Mahalanobis που ξεπερνούν το όριο αντιστοιχούν στα σημεια που σχηματίζουν το μικρό σύμπλεγμα συν την παρατήρηση 36. **Η παρατήρηση 58 δεν ανιχνεύθηκε.**
- Οι μέθοδοι FastMCD, FastPCS και BACON συμπίπτουν πλήρως μεταξύ τους εκτός της παρατήρησης 36 που ανιχνεύθηκε μόνο από την BACON. Επιπλέον συμπίπτουν επίσης και με τις εύρωστες αποστάσεις Mahalanobis.
- Όσον αφορά τη συνάρτηση depthout, αυξάνοντας το ποσοστιαίο όριο, έχει ως αποτέλεσμα την ανίχνευση περισσότερων παρατηρήσεων αλλά μόνο **στην συνοριακή περιοχή** των συμπλεγμάτων.



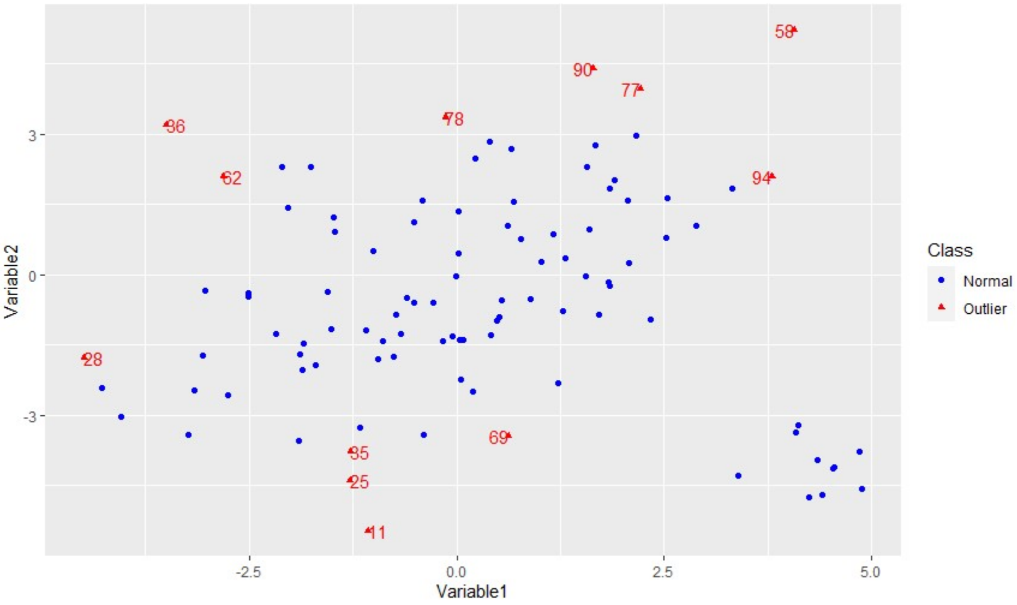
Εικόνα 13: Διάγραμμα διασποράς που δείχνει τις έκτροπες παρατηρήσεις που ανιχνεύθηκαν απο την **depthout** με ποσοστιαίο όριο **0.1**.



Εικόνα 14: Διάγραμμα διασποράς που δείχνει τις έκτροπες παρατηρήσεις που ανιχνεύθηκαν απο την **depthout** με ποσοστιαίο όριο **0.3**.

nnk συνάρτηση	
Έκτροπες παρατηρήσεις	k
11 27 28 34 36 58 77 78 90 94 98	5
11 28 35 36 58 62 69 73 77 78 90 94	6
11 25 28 35 36 58 62 69 77 78 90 94	7
11 25 27 28 33 34 36 58 62 69 77 78 90	8
11 25 27 28 33 36 58 62 69 77 78 90	9
11 25 27 28 33 36 58 62 69 77 90 94	10
1 2 3 4 5 6 7 8 9 10 11 36 58	11
1 2 3 4 5 6 7 8 9 10 36 58	15
1 2 3 4 5 6 7 8 9 10 36 58	30
1 2 3 4 5 6 7 8 9 10 36 58	40

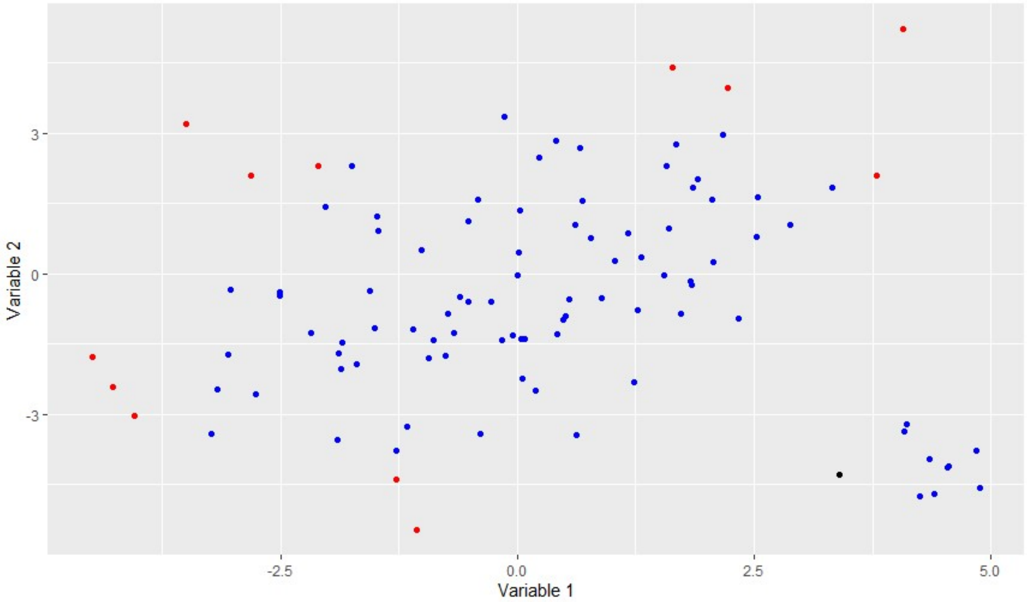
Πίνακας 3: Έκτροπες παρατηρήσεις για διάφορες τιμές του k για την **nnk** με ποσοστιαίο όριο 0.85.



Εικόνα 15: Διάγραμμα διασποράς που δείχνει τις έκτροπες παρατηρήσεις που ανιχνεύθηκαν απο την **nnk** για $k = 7$ και ποσοστιαίο όριο 0.85.

KNN_IN συνάρτηση	
Έκτροπες παρατηρήσεις	k
1 58 8 36 100 11 28 85 90 91 94 98 9	5
11 58 36 28 77 90 94 27 33 62 20 25 73	10
58 11 36 62 27 33 90 94 25 28 73 77 1	15
58 11 36 27 62 90 25 28 73 94 33 77 1	16
58 11 36 27 62 90 25 28 73 94 33 77 1	17
58 11 36 27 62 28 90 94 25 33 73 1 2	18
58 11 36 27 62 28 90 25 73 94 1 2 3	19
58 36 11 27 62 28 25 73 90 1 2 3 4	20
36 58 2 3 4 5 6 7 8 9 10 27 1	30
36 58 3 4 5 6 7 9 10 8 1 2 62	40

Πίνακας 4: Έκτροπες παρατηρήσεις για διάφορες τιμές του k για την **NNK_IN**. Καταγράφτηκαν οι δεκατρείς παρατηρήσεις με τον χαμηλότερο δείκτη (indegree number).



Εικόνα 16: Διάγραμμα διασποράς που δείχνει τις έκτροπες παρατηρήσεις που ανιχνεύθηκαν απο την **KNN_IN** για $k = 15$.

Παρατηρήσεις

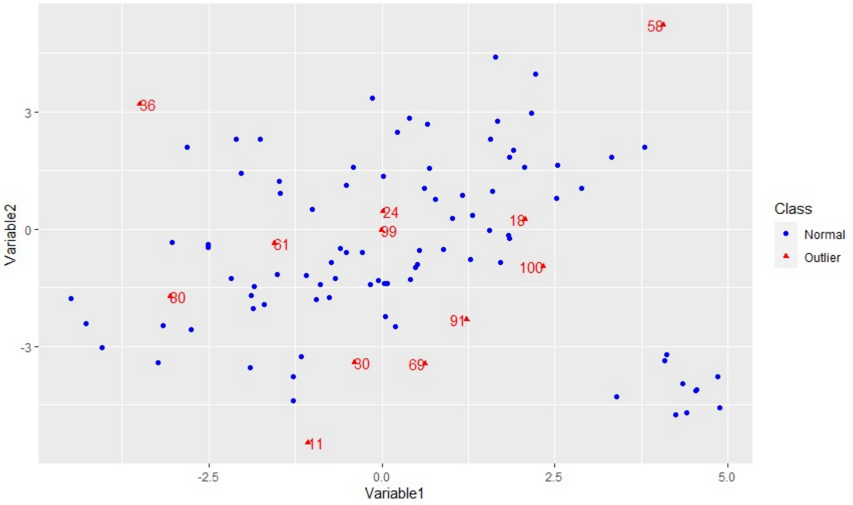
- Όσον αφορά την nnk, υπάρχει μία αστάθεια στις αρχικές τιμές του k και μετά σύγκλιση σε παρατηρήσεις, που ανιχνεύθηκαν επίσης από τις εύρωστες αποστάσεις Mahalanobis **συν την παρατήρηση 58**.
- Επιλέγοντας $k = 7$, ο αλγόριθμος ρυθμίζεται έτσι ώστε να ανιχνεύσει **μόνο** παρατηρήσεις από το μεγαλύτερο σύμπλεγμα παρατηρήσεων στα αριστερά του διαγράμματος διασποράς.
- Εφαρμόζοντας την KNN_IN παίρνουμε τα ίδια αποτελέσματα με αυτά της nnk για μεγάλες τιμές του k.
- Για $k = 15$ η KNN_IN αντιμετωπίζει **και τα δύο συμπλέγματα** σαν φυσιολογικά υποσύνολα και ανιχνεύει, για καθένα από αυτά, σημεία τα οποία απέχουν από κάθε σύμπλεγμα αντίστοιχα.
- Η μέθοδος KNN_IN, εκτος των $k$nn αποστάσεων που χρησιμοποιεί, υιοθετεί επίσης κάποιες από τις ιδιότητες των μεθόδων της πυκνότητας, καθώς αναζητά παρατηρήσεις που συμμετέχουν στις λιγότερες k-οστές κοντινότερες γειτονιές.

Έκτροπες παρατηρήσεις		
LOF	**RKOF**	k
11 80 81 98 61 1 74 90 77 36 94 91 58	81 100 11 80 94 61 77 90 36 74 1 91 58	5
81 35 32 90 62 77 33 27 91 69 11 36 58	98 33 32 81 27 77 94 90 11 69 91 36 58	10
32 28 62 25 77 90 27 91 33 69 11 36 58	81 94 32 28 33 27 77 90 11 91 69 36 58	11
7 90 4 25 8 2 91 27 33 69 11 36 58	77 28 30 33 27 90 2 8 91 11 36 69 58	12
8 2 3 9 6 5 69 10 7 4 11 36 58	91 3 6 5 8 2 7 10 4 11 36 69 58	13
1 8 2 3 6 5 10 9 7 4 11 36 58	1 6 5 7 9 4 36 69 10 11 2 8 58	14
1 8 2 9 7 3 36 6 5 10 11 4 58	7 9 69 36 3 6 5 2 11 8 4 10 58	15
11 36 8 2 1 3 6 5 10 58 9 7 4	36 11 8 2 1 10 3 6 5 9 7 4 58	20
11 8 2 36 1 3 58 10 6 5 9 7 4	11 36 8 2 1 58 3 10 6 5 9 7 4	30
11 36 8 1 2 3 6 5 10 9 7 58 4	11 36 8 1 2 3 58 6 5 10 9 7 4	40

Πίνακας 5: Έκτροπες παρατηρήσεις για διάφορες τιμές του k για τις **LOF** και **RKOF**. Καταγράφτηκαν οι δεκατρείς παρατηρήσεις με τις υψηλότερες τιμές LOF και RKOF.

RKOF συνάρτηση	
Έκτροπες παρατηρήσεις	k
80 45 18 4 10 100 91 69 11 61 36 58 1	5
78 11 24 18 99 30 80 100 36 69 61 91 58	10
78 99 18 24 11 30 80 100 36 69 61 91 58	11
18 78 99 24 11 30 80 100 36 69 61 91 58	12
18 80 78 24 99 11 30 100 36 69 61 91 58	13
37 78 80 24 99 100 11 30 36 69 61 91 58	14
78 37 24 80 100 99 11 30 36 61 69 91 58	15
89 80 99 78 24 100 11 30 36 69 61 91 58	20
80 78 89 99 24 100 30 11 36 61 69 91 58	30
78 34 80 24 100 89 30 11 61 36 69 91 58	40

Πίνακας 6: Έκτροπες παρατηρήσεις για διάφορες τιμές του k για την **RKOF** με $C = 0.1$.



Εικόνα 17: Διάγραμμα διασποράς που δείχνει τις έκτροπες παρατηρήσεις που ανιχνεύθηκαν απο την **RKOF** με $k = 11$.

Παρατηρήσεις

- Οι μέθοδοι LOF και RKOF ακολουθούν το ίδιο μοτίβο, σημειώνοντας μία αστάθεια σε μικρές τιμές του k ενώ σε μεγαλύτερες τιμές του και οι δύο μέθοδοι ανιχνεύουν τις ίδιες παρατηρήσεις με την nnk, εκτός της παρατήρησης 11.

- Η παράμετρος C επιρρεάζει τις $k$nn κάνοντας την RKOF πιο ευαίσθητη σε παρατηρήσεις που βρίσκονται στις λιγότερο πυκνές περιοχές του δείγματος.

- Οι παρατηρήσεις 99 και 24 ανιχνεύθηκαν για πρώτη φορά σαν έκτροπες παρόλο που **βρίσκονται στο εσωτερικό** του συμπλέγματος.

- Απομονωμένες παρατηρήσεις όπως αυτές, δεν ανιχνεύθηκαν από τις υπόλοιπες μεθόδους, καθώς οι περισσότερες από αυτές, αναζητούν έκτροπες παρατηρήσεις σε μία πιο καθολική κλίμακα.

5.3 Humus

- Το δείγμα humus βρίσκεται στο πακέτο **mvoutlier**.

- Τα δεδομένα προέρχονται από το Kola Project, που είναι μία γεολογική έρευνα και συλλέχθηκαν την περίοδο 1993-1998.

- Η συλλογή humus αποτελείται από 617 παρατηρήσεις με 44 διαφορετικές μεταβλητές.

- Προκειμένου να διαχειριστούμε τη λοξότητα από την οποία πάσχει το δείγμα, επιλέγουμε να τα μεταμορφώσουμε, λογαριθμίζοντας τα.

Συνοπτικά βήματα αναλύσης

- Αρχικά, εφαρμόζουμε όλες τις συναρτήσεις που χρησιμοποιήθηκαν ως τώρα από τα R πακέτα "OutlierDetection" και "DDoutlier" σε όλο το εύρος του δείγματος.
- Στην συνέχεια χρησιμοποιούμε τα "03 plots" από το R πακέτο "OutliersO3" το οποίο είναι εξαιρετικά χρήσιμο, αν θελήσουμε να δούμε την φύση των δεδομένων σε μικρότερες διαστάσεις.
- Επιλέγουμε τυχαία τέσσερις μεταβλητές (Al, Co, Cu, Ni) προς εξέταση.
- Επιπλέον, εφαρμόζουμε την μέδοδο των κύριων συνιστωσών (PCA).
- Τελικά, εφαρμόζουμε όλες τις μεθόδους ανίχνευσης έκτροπων παρατηρήσεων στο δείγμα μετά την εφαρμογή της PCA και συγκρίνουμε τα αποτελέσματα.

6 Συμπεράσματα

Εφαρμόζοντας όλες τις μεθόδους στα δεδομένα wood specific gravity, dat και humus καταλήγουμε στο γεγονός, πως καμία μέθοδος δεν υπερτερεί ολοκληρωτικά της άλλης.

Στην περίπτωση του dat, παρατηρούμε πως παρόλο που τα σημεία είναι κανονικά κατανεμημένα, ένα ικανό πλήθος έκτροπων παρατηρήσεων μπορεί να μολύνει το δείγμα σε τέτοιο βαθμό, ώστε κλασσικές μέθοδοι όπως αυτή της απόστασης Mahalanobis να μην δίνουν ορθά αποτελέσματα, δημιουργώντας την ανάγκη για χρήση πιο εύρωστων εργαλείων. Επιπλεόν, συναντήσαμε μία σημαντική διαφορά μεταξύ μεθόδων που χρησιμοποιούν μηδενική υπόθεση και αυτών που δεν χρησιμοποιούν. Οι τελευταίες ανέδειξαν σαν έκτροπες, παρατηρήσεις όπου οι πρώτες θεώρησαν απλώς ακραίες. Τέλος, μέθοδοι που βασίζονται στην πυκνότητα κατάφεραν να ανιχνεύσουν παρατηρήσεις, όπου οι υπόλοιπες μέθοδοι θεώρησαν ως φυσιολογικά σημεία καθώς είναι ικανές να αναζητήσουν απομονωμένα σημεία ακόμη και στο εσωτερικό ενός συμπλέγματος του δείγματος.

Στην περίπτωση του humus, καταγράψαμε πως λειτουργούν όλες αυτές οι μέθοδοι, σε μεγαλύτερες διαστάσεις. Επιπλέον, αναζητήσαμε έκτροπες παρατηρήσεις σε υποσύνολα παρατηρήσεων για να καταγράψουμε την συμπεριφορά των παρατηρήσεων που χαρακτηρίστηκαν ως έκτροπες, όταν έγινε εξέταση του δείγματος σε όλο το εύρος του. Τέλος, αναλύσαμε τις αλλαγές στα αποτελέσματα που πήραμε, αφού εφαρμόσαμε την μέθοδο της PCA, υπογραμμίζοντας πως η απόσταση Mahalanobis μπορεί να επιρρεαστεί από την μείωση διάστασης, ακόμη κι αν διατηρηθεί το μεγαλύτερο ποσοστό της πληροφορίας. Σε αντίθεση με αυτήν, οι μέθοδοι που βασίζονται στην απόσταση, ήταν συνεπείς στα αποτελέσματα τους και μετά την εφαρμογή της PCA, ενώ οι μέθοδοι που βασίζονται στην πυκνότητα, έδειξαν πως οι τοπικές γειτονιές κάθε παρατήρησης τροποποιούνται μετά την μείωση διάστασης.