Robustness properties of multivariate S-estimators

Unveiling the resilience and reliability in a multivariate statistical analysis

C. Marrone

Robustness properties of multivariate S-estimators

Unveiling the resilience and reliability in a multivariate statistical analysis

by

C. Marrone

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Monday July 17, 2023 at 3:00 PM.

Student number: 4816749
Project duration: April 24, 2023 – June 20, 2023
Thesis committee: Dr. H. P. Lopuhaä, TU Delft, supervisor
Dr. C. Kraaikamp, TU Delft

TU Delft

Preface

This thesis report marks the culmination of an extensive journey into the realm of robust Statistics. Undertaking this research endeavor has been both challenging and rewarding, and it is with great satisfaction that I present this work to the academic community.

The purpose of this thesis is to investigate the behavior of S-estimators under various circumstances. Through rigorous analysis, experimentation, and critical thinking, I have aimed to contribute to the existing body of knowledge in Statistics and shed light on the robustness and breakdown point of S-estimators.

The research conducted for this thesis would not have been possible without the guidance and support of my supervisor who has contributed his expertise and time. I would like to express my deepest gratitude to Dr. H. P. Lopuhaä, whose invaluable guidance, encouragement, and constructive feedback have shaped the direction and quality of this research. I would also like to extend my appreciation to the second member of my thesis committee, Dr. C. Kraaikamp.

Lastly, I would like to express my heartfelt gratitude to my family and friends for their unwavering support, understanding, and encouragement throughout this demanding process. Their belief in my abilities has been a constant source of inspiration, motivating me to persevere during the challenging phases of this research.

It is my sincere hope that this thesis report will serve as a valuable resource for researchers, academics, and professionals in the field of Statistics. May it contribute to further advancements and stimulate future investigations in the pursuit of knowledge.

C. Marrone
Delft, June 2023

Contents

Variables and parameters notation

Variables and parameter notation	**Description**
$\mathbf{t} = (t_1, ..., t_p)^T$	$p-$dimensional vector
$\mathbf{M} = (m_{ij})$	$(p \times p)$-matrices
$PDS(p)$	class of positive definite symmetric matrices
$\Theta = \mathbb{R}^p \times PDS(p)$	set of pairs θ = (**t,C**)
$\mathbf{x_1}, \mathbf{x_2}, ...$	vectors in $\mathbb{R}^p$
$X_1, X_2, ...$	if underlying distribution is assumed
$\boldsymbol{\mu} \in \mathbb{R}^p$	multivariate location parameter of an estimator
$\lfloor x \rfloor$	nearest integer less than or equal to x.
$d^2(\mathbf{x}; \mathbf{t}, \mathbf{C}) = (\mathbf{x} - \mathbf{t})\mathbf{C}^{-1}(\mathbf{x} - \mathbf{t})^T$	mahalanobis distance

Table 1: Variables and parameters notation

1 Introduction

Outliers are observations that deviate from the bulk of the data and they can distort the results of statistical analysis leading to incorrect conclusions if not addressed appropriately. In real-world data, outliers occur frequently enough to require our attention. The reasons for outliers are various, for example, instrument failure, non-representative sampling, formatting errors, and items that originate from different populations. Robust methods in statistics are very useful because they can provide more reliable statistical analysis in the presence of outliers.

The aim of any robust method is to reduce or remove the effect of atypical observations and allow the remainder to primarily determine the results. Robust methods are particularly useful in multivariate analysis: in high dimensions, no graphical aid is available to help with the outliers detection. For this reason, in such a setting a robust method does not only assist with a more accurate analysis, not being affected by discordant observations, but it can also facilitate the detection of outliers.

S-estimators are an example of a robust method. S-estimators of multivariate location and covariance are statistical methods used to estimate the location and covariance parameters of a multivariate distribution. The aim of this research is to investigate the behavior of S-estimators under different kinds of contamination by means of simulations with the statistical package R. Contamination refers to the deliberate introduction of outliers or abnormal values into a dataset. The primary objective of the simulations is to assess the efficacy of S-estimators under diverse contamination scenarios, thereby evaluating their performance.

In Section 2 the breakdown point, a measure of robustness, will be discussed. Section 3 introduces S-estimators starting with their original definition in a linear regression context, whereafter a direct generalization to S-estimators of multivariate location and covariance will be provided. Section 4 contains simulations and their results. Finally, section 5 presents a discussion of the obtained results with a particular focus on the performance of S-estimators compared to other robust and non-robust methods.

2

Breakdown point

With the aim of explaining the robustness of S-estimators, it is necessary to introduce the concept of the breakdown point. The breakdown point is a concept in statistics that refers to the proportion of outliers or anomalous observations that a statistical estimator can tolerate before it fails to provide useful results. Specifically, the breakdown point is the largest proportion of outliers that a statistical estimator can handle before it breaks down completely or produces grossly inaccurate results.

The breakdown point was invented by Hampel (1985), who gave a rigorous asymptotic definition. Donoho and Huber (1983) introduced a simplified version that works on finite samples. In this paper, we shall use the latter, namely:

Definition 2.1 (Breakdown Point for a location estimator)**.** Let X be a sample of n data points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ and $\mathbf{t}_n \in \mathbb{R}^p$ a location estimator of the parameter vector θ.
Let $\beta(m, \mathbf{t}_n, X)$ be the supremum of $||\mathbf{t}_n(X) - \mathbf{t}_n(X')||$ over all corrupted samples X' where m of the original points of X are replaced by arbitrary values. Then the breakdown point of $\mathbf{t}_n$ at X is given by:

$$\varepsilon_n^*(\mathbf{t}_n, X) = \min_{1 \le m \le n} \left\{ \frac{m}{n}; \beta(m, \mathbf{t}_n, X) \text{ is infinite} \right\}. \tag{2.1}$$

To get a better understanding of this concept here follow two examples of low and high breakdown point estimators for location, respectively.

Example 2.1 (Sample Mean)**.** Let X be a sample of n data points $x_1, \ldots, x_n \in \mathbb{R}$ and denote the sample mean by $t_n \in \mathbb{R}$. Substitute x_n, with an arbitrary value, say $y \in \mathbb{R}$. Then the contaminated sample X' becomes $(x_1, x_2, \ldots, x_{n-1}, y)$. Let $y \to \infty$, then the sample mean given by

$$t_n(X') = \frac{1}{n} \left[\left(\sum_{k=1}^{n-1} x_k \right) + y \right] \to \infty.$$

Hence within the univariate setting, for the sample mean even one outlying observation can cause breakdown, thus $\varepsilon_n^*(t_n, X) = 1/n$ which tends to 0 for large values of n.

Example 2.2 (Sample Median)**.** Let $X = (3, 5, 9, 12, 23)$ be a sample of 5 data points in $\mathbb{R}$ and denote the sample median by $t_n \in \mathbb{R}$. Since the observations are sorted in ascending order, the sample median, defined as the middle point of the data, is given by: $t_n(X) = 9$. Now, suppose we arbitrarily alter or replace 2 of the 5 observations. We do this by selecting any 2

observations and replacing them with an arbitrary value, say $y \in \mathbb{R}$. Without loss of generality, assume we replace the first 2 observations. The sample becomes $X' = (y, y, 9, 12, 23)$. Now let $y \to \infty$, then the sample in ascending order is given by $X' = (9, 12, 23, y, y)$, and so the sample median will be $t_n(X') = 23$. Note that in this case, the contamination may alter the estimated value (that depends on the choice of the two observations that we replace). However, it still gives a value close to the center of the data without becoming infinitely large. If we were to substitute 3 of the 5 observations with an arbitrary value, $y \in \mathbb{R}$, and let $y \to \infty$, then the sample median would tend to infinity and break. This shows that the sample median in this example has breakdown point given by: $\varepsilon_n^*(t_n, X) = 3/5$. Since the sample median lies always in between $x_{(n/2)}$ and $x_{(n/2+1)}$ for a sample of n observations in ascending order, the breakdown point of the sample median is given by:

$$\varepsilon_n^*(t_n, X) = \frac{\lfloor (n+1)/2 \rfloor}{n},$$

which tends to 0.5 for large values of n.

It is important to take into consideration that the breakdown point for a location estimator is reached by taking the estimator to the boundary of the parameter space, which for the location is infinity. For a scale estimator, there are two boundaries, namely 0 and infinity, hence the following definition:

Definition 2.2 (Breakdown Point for a covariance estimator)**.** Let X be a sample of n data points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ and $\mathbf{C}_n \in \mathrm{PDS}(p)$ a covariance estimator. Let $\gamma(m, \mathbf{C}_n, X)$ be the supremum of $D(\mathbf{C}_n(X), \mathbf{C}_n(X'))$ over all corrupted samples X' where m of the original points of X are replaced by arbitrary values, where

$$D(\mathbf{A}, \mathbf{B}) = \max\left\{ |\lambda_1(\mathbf{A}) - \lambda_1(\mathbf{B})|, |\lambda_p(\mathbf{A}^{-1}) - \lambda_p(\mathbf{B}^{-1})| \right\},$$

for λ_1 and λ_p being the smallest and largest eigenvalues, respectively. Then the breakdown point of $\mathbf{C}_n$ at X is given by:

$$\varepsilon_n^*(\mathbf{C}_n, X) = \min_{1 \le m \le n} \left\{ \frac{m}{n}; \gamma(m, \mathbf{C}_n, X) \text{ is infinite} \right\}. \tag{2.2}$$

Once again here follow two examples of low and high breakdown point estimators for covariance, respectively.

Example 2.3 (Sample Variance)**.** With the same setting as in Example 2.1 denote the sample variance by $\mathbf{C}_n \in \mathrm{PDS}(1)$. Let $y \to \infty$, then the sample variance given by

$$\begin{aligned}
\mathbf{C}_n(X') &= \frac{1}{n-1}\left[\sum_{x \in X'} (x - \bar{x})^2 \right] \\
&= \frac{1}{n-1}\left[\sum_{x \in X'} (x^2 - 2x\bar{x} + \bar{x}^2) \right] \\
&= \frac{1}{n-1}\left[\left(\sum_{x \in X'} x^2 \right) - 2\bar{x} \sum_{x \in X'} x + n\bar{x} \right] \\
&= \frac{1}{n-1}\left[\left(\sum_{x \in X'} x^2 \right) - n\bar{x} \right] \to \infty
\end{aligned} \tag{2.3}$$

as by Example 2.1, where $\bar{x}$ is the sample mean of the contaminated sample X'. Once again one altered observation causes breakdown of the estimator. As a result, $\varepsilon_n^*(\mathbf{C}_n, X) = 1/n$, which tends to 0 for large values of n.

Example 2.4 (Median of absolute deviation)**.** Let $X = (1,2,3,4,5)$ be a sample of 5 datapoints in $\mathbb{R}$ and denote the median of absolute deviation by $C_n \in \text{PDS}(1)$. Then C_n is defined by

$$C_n(X) = \text{median}(|x_i - \tilde{x}|),$$

where $\tilde{x}$ is the median of the original sample and x_i is the i-th observation of the sample X. So $C_n(X) = \text{median}(2,1,0,1,2) = 1$. Now, suppose we alter the last of the 5 observations by an arbitrary value $y \in \mathbb{R}$ then the sample becomes $X' = (1,2,3,4,y)$ and $C_n(X') = \text{median}(2,1,0,1,|y-3|) = 1$ fon any $y \in \mathbb{R}$, hence no breakdown is caused. But suppose we alter the last 2 of the 5 observations by an arbitrary value $y \in \mathbb{R}$ then the sample becomes $X' = (1,2,3,y,y)$ and $C_n(X') = \text{median}(2,1,0,|y-3|,|y-3|) = 0$ for $y = 3$, hence breakdown is caused. We conclude that the breakdown point is $2/5$. In a general setting:

$$\varepsilon_n^*(C_n, X) = \frac{\lfloor n/2 \rfloor}{n},$$

which tends to 0.5 for large values of n.

2.1. Maximum breakdown point

After observing the breakdown point for two different location estimators, the natural question is: is there an upper bound for location estimators? And if so can this upper bound actually be attained? Well, the answer in general is no. Consider a location estimator given by $t_n(X) = c \in \mathbb{R}$ for all possible samples X of n observations $x_1, \ldots, x_n \in \mathbb{R}$, such an estimator has breakdown point 1 as we can replace the whole sample X with arbitrary values and still have location estimate given by $t_n(X) = c$. Clearly, this estimator is not a sensible choice as in most cases it will fail to provide significant results. It is therefore crucial to pursue alternative estimators that are more likely to produce meaningful outcomes.

The natural option for such a choice is an equivariant estimator. An estimator is said to be equivariant with respect to a transformation T if applying T to the data and then estimating the parameter with the transformed data is equivalent to estimating the parameter with the original data and then applying the transormation T to the estimate. The kind of transformation determines the type of equivariance. Most relevant equivariances to this study are the translation equivariance and the affine equivariance.

Definition 2.1.3 (Translation equivariance for a location estimator)**.** Let X be a sample of n data points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ and denote by $\mathbf{t}_n(X) \in \mathbb{R}^p$ a location estimate based on X. Then $\mathbf{t}_n$ is translation equivariant if $\mathbf{t}_n(X+\mathbf{v}) = \mathbf{t}_n(X)+\mathbf{v}$ for all $\mathbf{v} \in \mathbb{R}^p$ where $X+\mathbf{v} = \{\mathbf{x}_1+\mathbf{v}, \ldots, \mathbf{x}_n+\mathbf{v}\}$.

Definition 2.1.4 (Affine equivariance for both a location estimator and covariance)**.** Let X be a sample of n data points $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^p$ and denote by $\mathbf{t}_n(X) \in \mathbb{R}^p$ a location estimate based on X and by $\mathbf{C}_n(X) \in \text{PDS}(p)$ a covariance estimate based on the same sample, then $\mathbf{t}_n(X)$ and $\mathbf{C}_n(X)$ are said to be affine equivariant if $\mathbf{t}_n(\mathbf{A}X + \mathbf{v}) = \mathbf{A}\mathbf{t}_n(X) + \mathbf{v}$ and $\mathbf{C}_n(\mathbf{A}X + \mathbf{v}) = \mathbf{A}\mathbf{C}_n(X)\mathbf{A}^T$ for all non-singular $p \times p$ matrices $\mathbf{A}$ and $\mathbf{v} \in \mathbb{R}^p$, where $\mathbf{A}X + \mathbf{v} = \{\mathbf{A}\mathbf{x}_1 + \mathbf{v}, \ldots, \mathbf{A}\mathbf{x}_n + \mathbf{v}\}$. If this property holds for an orthogonal matrix $\mathbf{A}$, then we talk about orthogonal equivariance.

Lopuhaä and Rousseeuw (1991) discuss upper bounds for location estimators satisfying various equivariance properties. In particular, a significant result is that the upper bound for a translation equivariant estimator is given by $\lfloor (n+1)/2 \rfloor / n$. Obviously, since the class of affine equivariant estimators is contained in the class of translation equivariant estimators, the upper bound $\lfloor (n+1)/2 \rfloor / n$ also holds for them. Considering high breakdown is often associated with reliability and accuracy of estimation, it is interesting to look at the case when this bound can be obtained.

Example 2.5 (Coordinate-wise median)**.** Let $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ be a sample of n observations in $\mathbb{R}^p$, $p > 1$. Let x_{ij} be the j-th coordinate of the observation $\mathbf{x}_i$, $1 \leq i \leq n$, $1 \leq j \leq p$, $i, j \in \mathbb{Z}$. Denote the coordinate-wise median of the sample X by CWM(X). Let T be a transformation given by $T(X) = X + \mathbf{v}$ for some $\mathbf{v} \in \mathbb{R}^p$, then the sample after the transformation becomes $T(X) = (\mathbf{x}_1 + \mathbf{v}, \ldots, \mathbf{x_n} + \mathbf{v})$. Now we can observe that the coordinate-wise median operation independently considers each coordinate of the vectors. In particular the CWM(X) = (median$(x_{11}, \ldots, x_{n1}), \ldots,$ median$(x_{1p}, \ldots, x_{np})$. Let $1 \leq j \leq p$ be arbitrary, denote the j-th coordinate of the coordinate-wise median of the transformed dataset by CWM$(T(X))_j$ then:

$$\begin{aligned} \text{CWM}(T(X))_j &= \text{median}(x_{1j} + v_j, \ldots, x_{nj} + v_j) \\ &= \text{median}(x_{ij}, \ldots, x_{nj}) + v_j \\ &= \text{CWM}(X)_j + v_j, \end{aligned} \tag{2.4}$$

where v_j is the j-th coordinate of the vector $\mathbf{v}$. Since j was arbitrarily chosen (2.4) holds that for all $1 \leq j \leq p$, therefore the coordinate-wise median is translation equivariant. Since the sample median as defined in Example 2.2 is also translation equariant and the CWM can be computed coordinate-wise as the sample median, it holds that:

$$\varepsilon_n^*(\text{CWM}, X) = \frac{\lfloor (n+1)/2 \rfloor}{n}.$$

In order to present results about upper bounds for affine equivariant estimators, it is essential to introduce the following concept.

Definition 2.1.5 (General position)**.** A collection X is said to be in general position if there are no $p+1$ points contained in some hyperplane of dimension smaller than p. For example, a set of points in general position implies that no three points are collinear (lie on the same line) and no four points are coplanar (lie in the same plane).

Davies (1987) showed that for covariance estimators the bound $\lfloor (n+1)/2 \rfloor / n$ is not sharp. When the collection X is in general position, if $n \geq p+1$, the breakdown point of any affine equivariant covariance estimator $\mathbf{C}_n$, is at most $\lfloor (n-p+1)/2 \rfloor / n$. Also in this case it is interesting to find estimators actually attaining this bound. As we have seen in Example (2.4) the median of absolute deviation attains the above-mentioned upper bound. The class of multivariate S-estimators contains members that also attain this bound in higher dimensions.

3

S-estimators

The estimator we will analyze in detail in terms of its robustness is the multivariate S-estimator of location and covariance. S-estimators were introduced by Rousseeuw and Yohai (1984) in a regression context, and defined as the solution to the problem of minimizing σ subject to

$$\frac{1}{n}\sum_{i=1}^{n}\rho\left(\frac{y_i-\boldsymbol{\theta}^T\mathbf{x}_i}{\sigma}\right)\leq b_0 \tag{3.1}$$

among all $(\boldsymbol{\theta},\sigma)\in\mathbb{R}^p\times(0,\infty)$, where $0<b_0<\sup\rho$. It is important to highlight that for ρ defined by $\rho(x)=x^2$, $x\in\mathbb{R}$, (3.1) becomes:

$$\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i-\boldsymbol{\theta}^T\mathbf{x}_i}{\sigma}\right)^2=b_0\Leftrightarrow$$

$$\sigma^2(\boldsymbol{\theta})=\frac{1}{b_0}\frac{1}{n}\sum_{i=1}^{n}\left(y_i-\boldsymbol{\theta}^T\mathbf{x}_i\right)^2 \tag{3.2}$$

Hence minimizing σ^2 of (3.2) over $\boldsymbol{\theta}$ corresponds to the least square estimator by definition. The problem with such estimators though is that for any outlier, meaning a single point diverging significantly from the true mean, the estimator fails to deliver reliable results as the distance between the points and the true mean is quadratically taken into account with the minimization problem.

In order to obtain more robust estimates the function ρ was assumed to satisfy the following properties:

(R1) ρ is symmetric, has a continuous derivative ψ and $\rho(0)=0$;

(R2) There exists a finite constant $c_0>0$ such that ρ is non increasing on $[0,c_0]$ and constant on $[c_0,\infty)$.

The property (R2) ensures that points far from the true mean no longer contribute quadratically but only with a fixed amount. A specific choice of ρ, still satisfying (R2) will ensure some accuracy. Here follows an example of a ρ-function satisfying those properties:

- a

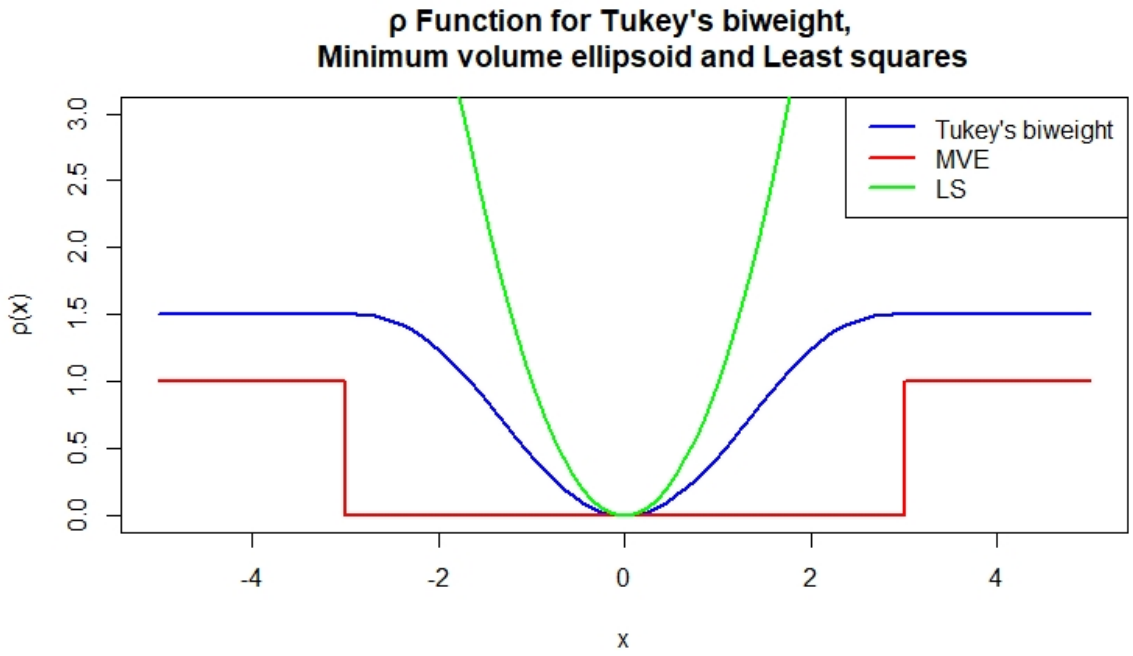

Figure 3.1: Graph of ρ for $c_0 = 3$ corresponding to least squares, minimum volume ellipsoid, and Tukey's biweight

- c_0 tuned for low asymptotic variance → low breakdown point

Example 3.1 (Tukey's biweight)**.**

$$\rho_B(y, c_0) = \begin{cases} \frac{y^2}{2} - \frac{y^4}{2c_0^2} + \frac{y^6}{6c_0^4} & \text{if } |y| \le c_0, \\ \frac{c_0^2}{6} & \text{if } |y| \ge c_0 \end{cases} \tag{3.3}$$

Its derivative is known as Tukey's biweight function $\psi_B(y, c_0) = y(1-(y/c_0)^2)^2 \mathbb{1}_{[-c_0,c_0]}(y)$.

The biweight rho-function makes sure that a good level of accuracy is maintained up to a certain distance from the mean by choice of the constant c_0. Afterward, we will see how such a constant can be chosen to acquire the largest possible breakdown point. It is important to notice that Tukey's biweight function is a good compromise between the least squares estimator ($\rho(x) = x^2$) and the one corresponding to the minimum volume ellipsoid one as to be seen in Figure 3.1. The minimum volume ellipsoid estimator for multivariate location and scatter (also known as MVE) is defined as follows:

Definition 3.1 (Minimum volume ellipsoid)**.** Let $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, $n \ge p+1$, then the minimum volume ellipsoid is given by the center and covariance structure of the ellipsoid with minimal volume that covers at least h points of X, where h can be chosen between $\lfloor n/2+1 \rfloor$ and n. Because the size of an ellipsoid is determined by $\det(\mathbf{C})$ this definition can be reformulate as follows: the minimum volume ellipsoid location estimator $\mathbf{t}_n \in \mathbb{R}^p$ and scatter estimator $\mathbf{C}_n \in \text{PDS}(p)$ minimize the determinant of $\mathbf{C}$ subject to:

$$\#\{i : (\mathbf{x}_i - \mathbf{t})^T \mathbf{C}^{-1} (\mathbf{x}_i - \mathbf{t}) \le c_0\} \ge h. \tag{3.4}$$

Since Rousseeuw (1985) showed that the MVE is affine equivariant with breakdown point $(\lfloor n/2 \rfloor - p + 1)/n$, which is smaller than the covariance upper bound $\lfloor (n-p+1)/2 \rfloor / n$, then (3.4) is readjusted by Lopuhaä and Rousseeuw (1991) so that the MVE does attain this upper bound:

$$\#\{i : (\mathbf{x}_i - \mathbf{t})^T \mathbf{C}^{-1} (\mathbf{x}_i - \mathbf{t}) \leq c_0\} \geq \left\lfloor \frac{n+p+1}{2} \right\rfloor \tag{3.5}$$

In this case the ρ function is defined by $\rho(x) = \mathbb{1}_{d_i > c_0}$, where $d_i(\mathbf{x}_i; \mathbf{t}, \mathbf{C})$ is as in Table 1. Note that in this case the constant b_0 is given by:

$$b_0 = 1 - \frac{\lfloor \frac{n+p+1}{2} \rfloor}{n}$$

which can be found by solving:

$$\#\{i : (\mathbf{x}_i - \mathbf{t})^T \mathbf{C}^{-1} (\mathbf{x}_i - \mathbf{t}) > c_0\} \leq n - \left\lfloor \frac{n+p+1}{2} \right\rfloor = b_0 n$$

A direct generalization to S-estimators of multivariate location and covariance is obtained by adjustment of (3.1).

Definition 3.2 (S-estimate)**.** Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ and let $\rho : \mathbb{R} \to [0, \infty)$ be a function satisfying (R1) and (R2). Then the S-estimate of multivariate location and covariance is defined as the solution $\boldsymbol{\theta}_n = (\mathbf{t}_n, \mathbf{C}_n)$ to the problem of minimizing $\det(\mathbf{C})$ subject to:

$$\frac{1}{n} \sum_{i=1}^{n} \rho \left[\left\{ (\mathbf{x}_i - \mathbf{t})^T \mathbf{C}^{-1} (\mathbf{x}_i - \mathbf{t}) \right\}^{1/2} \right] \leq b_0 \tag{3.6}$$

where the constant $0 < b_0 < \sup \rho$ can be chosen in agreement with an assumed underlying distribution.

In particular, if $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ are assumed to be a sample $X_1, X_2, \ldots, X_n$ with an underlying multivariate normal distribution with mean $\boldsymbol{\mu} \in \mathbb{R}^p$ and covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{p \times p}$ then since we aim to have $\mathbf{t}_n \approx \boldsymbol{\mu}$ and $\mathbf{C}_n \approx \boldsymbol{\Sigma}$ the constant b_0 is chosen by:

$$\begin{aligned} b_0 &\geq \frac{1}{n} \sum_{i=1}^{n} \rho \left[\left\{ (\mathbf{x}_i - \mathbf{t}_n)^T \mathbf{C}_n^{-1} (\mathbf{x}_i - \mathbf{t}_n)^{1/2} \right\} \right] && (3.7) \\ &\approx \mathbb{E} \left[\rho \left[\left\{ (\mathbf{x}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} \left(\mathbf{x}_i - \boldsymbol{\mu} \right)^{1/2} \right\} \right] \right] && (3.8) \\ &\approx \mathbb{E} \left[\rho(\|Z\|) \right] && (3.9) \end{aligned}$$

where (3.8) is due to the Law of Large Numbers (for large values of n) and Z in (3.9) is a random vector with a p-variate normal distribution, with mean given by $\mathbf{0} \in \mathbb{R}^p$ and covariance given by the identity matrix $\mathbf{I} \in \mathbb{R}^{p \times p}$. Note, however, that because of the property (R2) b_0 does not only depend on Z but also on the constant c_0. Moreover, take into consideration that for continuous ρ functions the minimization problem (3.1) is the same as the one where the equal sign substitutes the inequality sign. An example of a ρ-function for (3.6) is Tukey's biweight function as in Example 3.1.

3.1. Breakdown point of S-estimators

A significant outcome of the paper by Lopuhaä and Rousseeuw (1991) is the upper-bound for the breakdown point of S-estimators which is given by:

$$\varepsilon_n^*(\mathbf{t}_n, \mathbf{X}) = \varepsilon_n^*(\mathbf{C}_n, \mathbf{X}) = \frac{\lceil nr \rceil}{n} \tag{3.10}$$

where $\mathbf{X}$ is a set on $n \geq p+1$ points in general position in $\mathbb{R}^p$ and $r = b_0 / \sup \rho$ under the assumption that $r \leq (n-p)/(2n)$ and ρ satisfies properties (R1) and (R2).

This result implies that the constant c_0 of the property (R2) can be chosen such that the breakdown point is maximized. In particular, since both b_0 and ρ depend on c_0, the latter can be computed by solving:

$$\frac{n-p}{2n} = \frac{b_0(c_0)}{\rho(c_0)} \tag{3.11}$$

for large values of n. In this respect, it is very interesting to look at the asymptotic variance of the S-estimators, a measure of the variability of the estimator as the sample size approaches infinity. In Lopuhaä (1989) (Section 6) the relation between asymptotic variance and breakdown point of the S-estimators is analyzed: it is given that in general for any S-estimator the asymptotic variance will depend on the ρ-function. As a consequence, the asymptotic variance depends on c_0. Moreover, from (3.11) clearly, the constant c_0 can be tuned so that the breakdown point is maximized. To be noted as well from Lopuhaä (1989) is that a constant c_0 calibrated to have a high breakdown would correspond to a high asymptotic variance, and analogously c_0 tuned for a low asymptotic variance would correspond to a low break down point. This shows that a drawback of the S-estimators is that it is not possible to achieve small asymptotic variance and high breakdown point at the same time.

4

Simulations

In statistical analysis, it is sometimes assumed that data follow a normal distribution. However, in real-world applications, data may be contaminated with outliers, which can affect the accuracy of the statistical analysis.

Multivariate normal data refers to data that are distributed according to a multivariate normal distribution. In the presence of contamination, the performance of S-estimators can be evaluated using simulations. Simulation studies involve generating data from a known distribution and adding contamination to the data to evaluate the performance of statistical methods. Simulations can be used to evaluate the robustness of S-estimators to various levels of contamination and to compare their performance to other statistical methods. The simulations can also be used to assess the impact of different factors such as sample size, number of dimensions, and type of contamination on the performance of the S-estimators.

In this research, we will simulate multivariate standard normal data with different types of contamination, to see how the S-estimators perform in different circumstances and compare it to other robust and non-robust estimators, in particular, the sample mean and covariance, also known as classic estimator, as non-robust, and the minimum covariance determinant (MCD), formally defined in Rousseeuw (1985), and the M-estimator, formally defined in Maronna (1976), as robust. The MCD and M-estimator are chosen for comparison, as by Maronna et al. (2006) the first has maximum breakdown point similar to the S-estimator whereas the second has maximum breakdown point given by $1/(p+1)$ where p is the number of dimensions. Since in multivariate analysis $p > 1$ it should theoretically perform less well than the S-estimator in higher dimensions. All simulations will be performed with the statistical package R. In particular, the estimators will be computed by using the rrcov library.

In order to understand how the estimators perform, for each contamination setting 500 repetitions are conducted. At each iteration, values of the estimates for the location and covariance matrix are stored. To assess the performance of the estimators three indicators will be used:

- **Location indicator**: boxplot of a vector where the averages of all coordinates of the estimated location vectors are stored;
- **Covariance indicator (smallest eigenvalue)**: boxplot of a vector where the smallest eigenvalues of the estimated covariance matrices are stored;

- **Covariance indicator (largest eigenvalue)**: boxplot of a vector where the largest eigenvalues of the estimated covariance matrices are stored.

The uncontaminated data is generated from the multivariate standard normal distribution, with mean vector $\mathbf{0}$ in $\mathbb{R}^p$ and covariance matrix equal to the identity matrix in $\mathbb{R}^{p\times p}$. Since the average of the coordinates of the true mean of the uncontaminated data is 0, the location estimator will be assessed as performing well if the location indicator is centered around 0. Although the covariance of the uncontaminated data is the identity matrix, to determine how the estimator for the covariance performs it is necessary to look at both largest and smallest eigenvalue of the estimated covariance matrix, as the estimator breaks if either the smallest eigenvalue tends to zero, or if the largest eigenvalue tends to infinity.

4.1. Contamination by alteration of the mean vector

The original dataset is contaminated by replacing part of the data points with points generated by a contaminated distribution. Firstly, the latter is constructed by shifting all coordinates of the mean zero vector over a fixed distance that varies over 25, 10, and 5. Also, variation in contamination size will be performed for each level of mean shift. The simulations are carried out on the basis of 5 different contamination levels, namely: 10%, 20%, 30%, 40%, and 45%. For proper comparison, the estimators have been computed for the original dataset without contamination (0%) as well. Moreover, the dimension of the data points has been taken as yet another factor. It will vary between 2, 5, and 10. The sample size was fixed to 100. The sample mean and covariance estimator was denoted by CL, the S-estimator by S, the M-estimator by M, and the minimum covariance determinant by MCD.

4.1.1. Mean shift of 25

The 1st simulation was carried out with contamination given by a mean shift of 25.

The plot in Figure 4.1.1.1 shows the location indicator for different levels of contamination and different dimensions by a mean shift of 25. At dimension 2 all non-classic estimators perform well. At dimension 5 a notable result is that from 40% contamination the S-estimator breaks, as opposed to all non-classic robust estimators. At dimension 10 the S-estimator presents the same behavior but already from 30% contamination. From 40% contamination, all estimators fail to provide accurate estimates.

The plot in Figure 4.1.1.2 shows the covariance indicator (smallest eigenvalue) for different levels of contamination and different dimensions by a mean shift of 25. All estimators estimate the smallest eigenvalue lower than 1, and the negative bias gets larger as the dimension increases. From dimension 5 the S-estimator moves downwards again with a very high percentage of contamination. The classic estimator is quite steady throughout all levels of contamination and dimension size.

The plot in Figure 4.1.1.3 shows the covariance indicator (largest eigenvalue) for different levels of contamination and different dimensions by a mean shift of 25. At dimension 2 all estimators but the classic one are reliable. At dimension 5 with 40% and 45% the S-estimator presents very high values as opposed to MCD and M which perform well. At dimension 10 the S-estimator performs poorly from 30% contamination onwards. MCD and M-estimator are inaccurate as well from 40% contamination onwards.

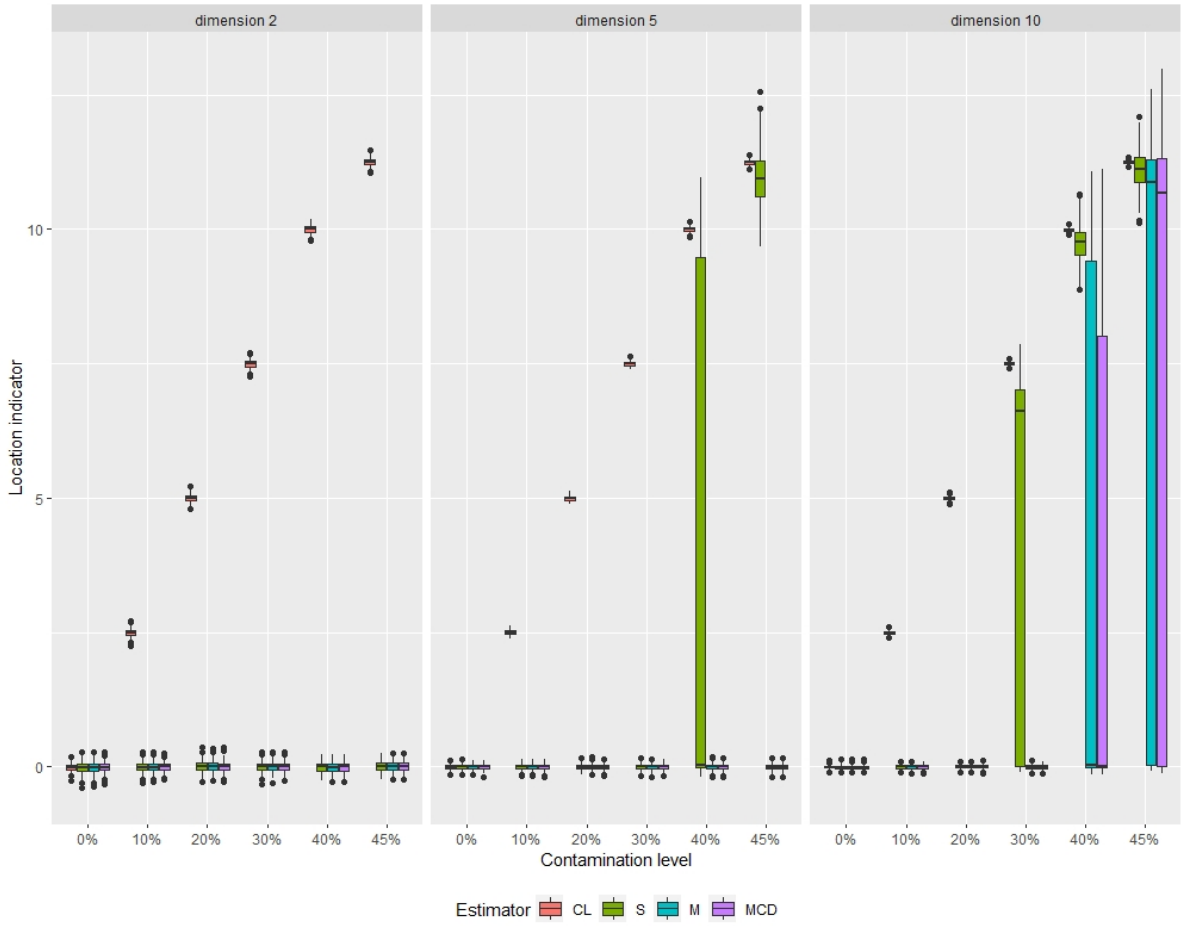

Figure 4.1.1.1: Location indicator by mean shift of 25

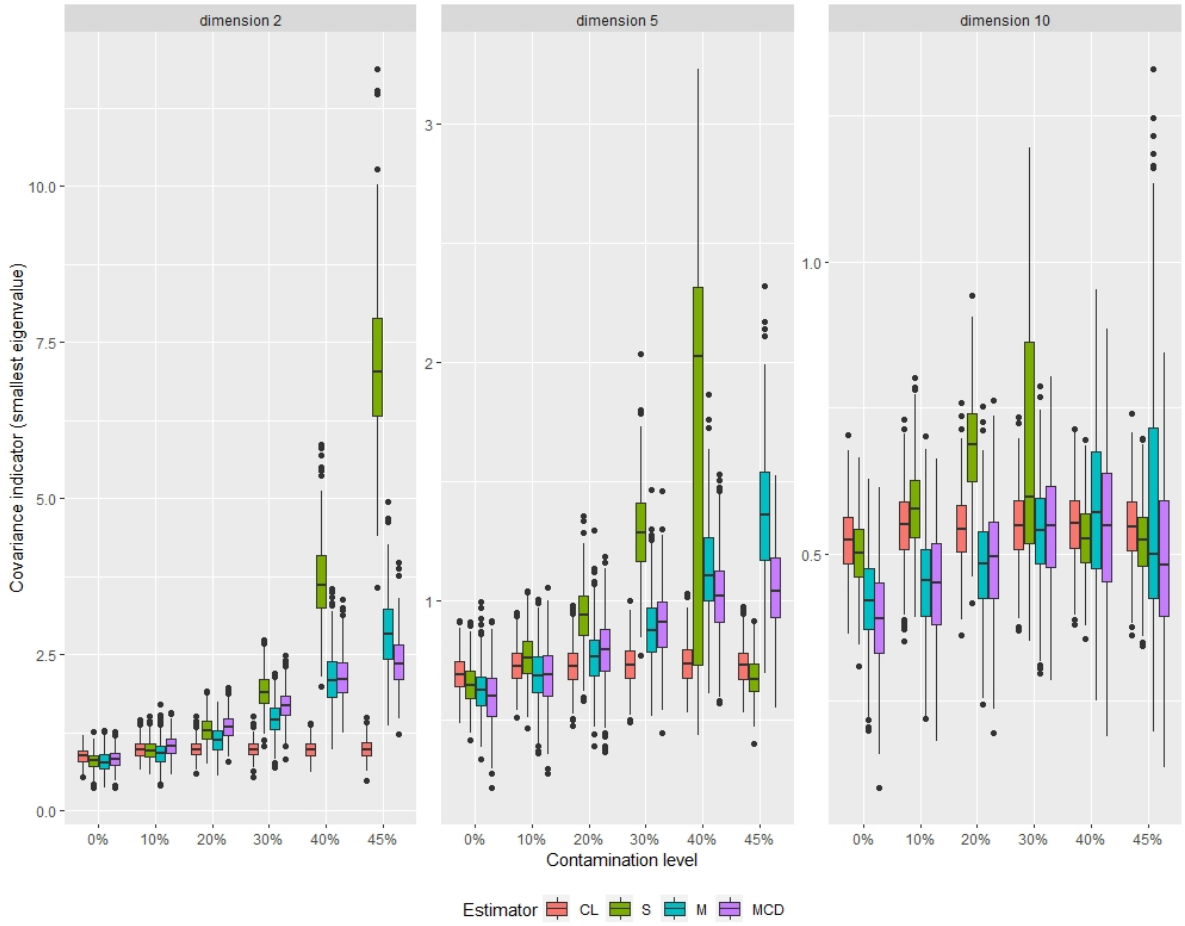

Figure 4.1.1.2: Covariance indicator (smallest eigenvalue) by mean shift of 25

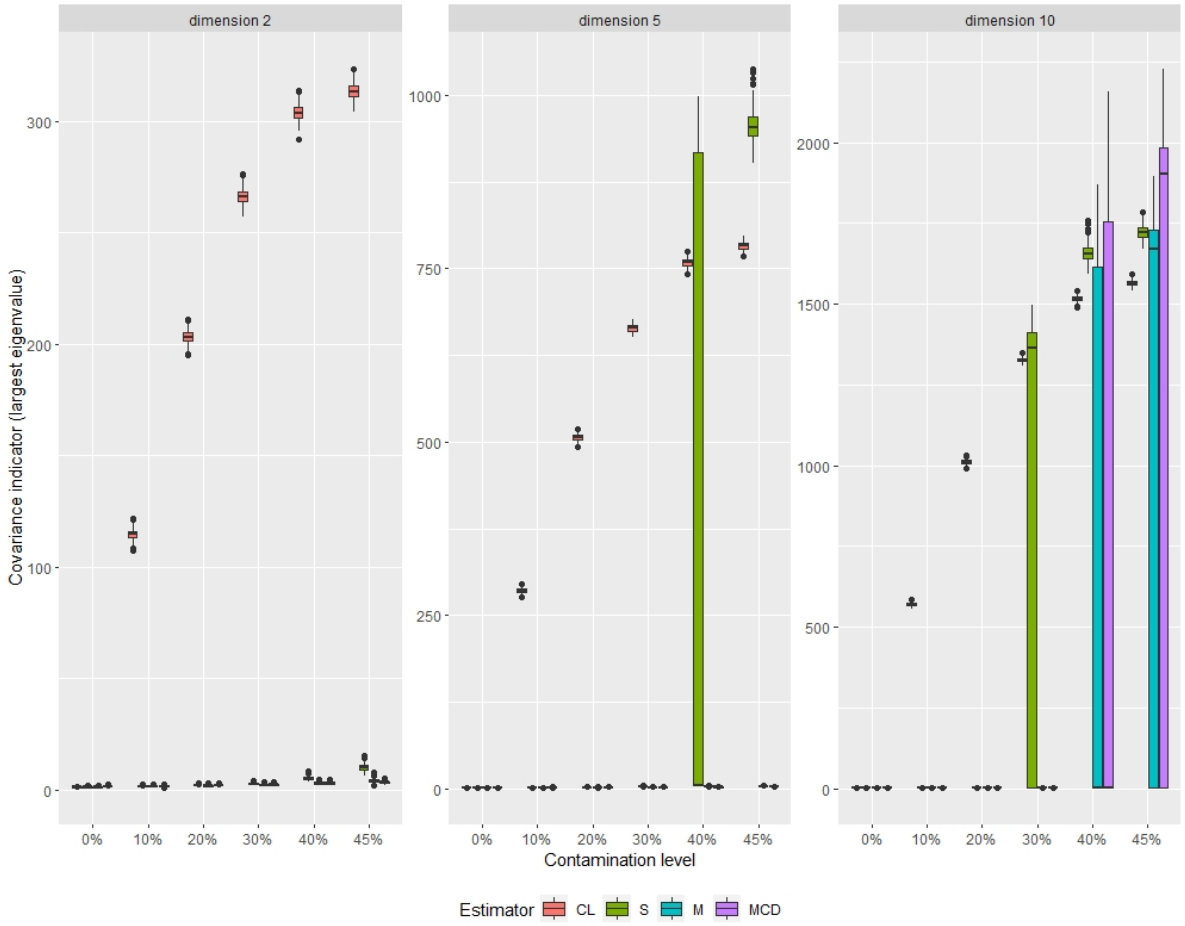

Figure 4.1.1.3: Covariance indicator (largest eigenvalue) by mean shift of 25

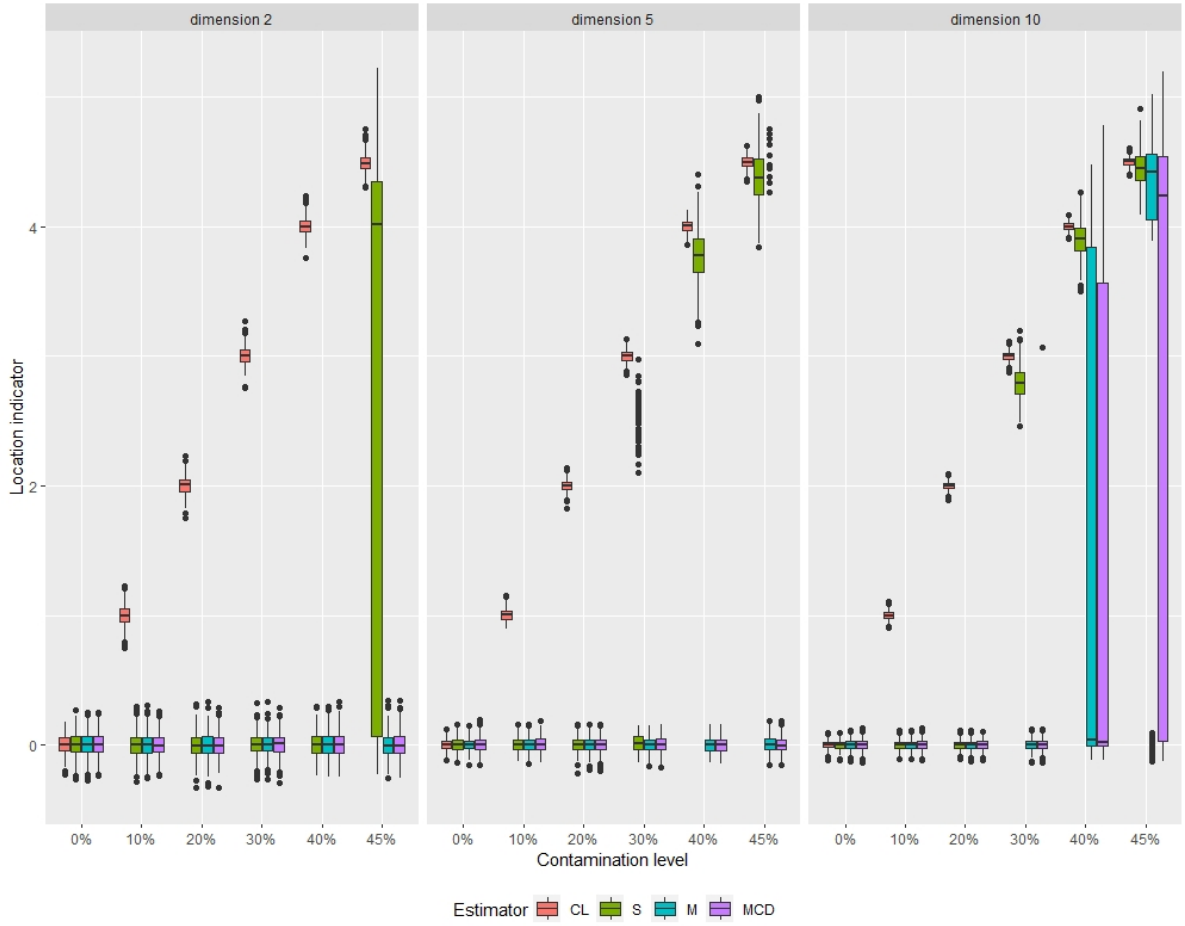

Figure 4.1.2.1: Location indicator by mean shift of 10

4.1.2. Mean shift of 10

The 2nd simulation was carried out with contamination given by a mean shift of 10.

The plot in Figure 4.1.2.1 shows the location indicator for different levels of contamination and different dimensions by a mean shift of 10. At dimension 2 the S-estimator present a stretched boxplot with 45% contamination, so the estimator breaks, as opposed to the other non-classic estimators. At dimension 5 the S-estimator breaks already with 40% contamination whereas M and MCD still perform well. At dimension 10 the S-estimator breaks with 30% contamination, MCD and M break from 40% contamination onwards.

The plot in Figure 4.1.2.2 shows the covariance indicator (smallest eigenvalue) for different levels of contamination and different dimensions by a mean shift of 10. The behaviors of the estimators are very similar to the ones registered previously. In this case, the estimated value for the smallest eigenvalue starts from around 1 and decreases as the dimension size increases. At dimensions 5 and 10, the S-estimator moves downwards from contamination levels of 40% and 30%, respectively.

The plot in Figure 4.1.2.3 shows the covariance indicator (largest eigenvalue) for different levels of contamination and different dimensions by a mean shift of 10. At dimension 2 the S-estimator does not perform well with 45% contamination, as the center of the data leans towards high values. Other non-classic estimators perform well. At dimension 5 the S-estimator breaks from a contamination level of 40%, whereas MCD and M estimates are

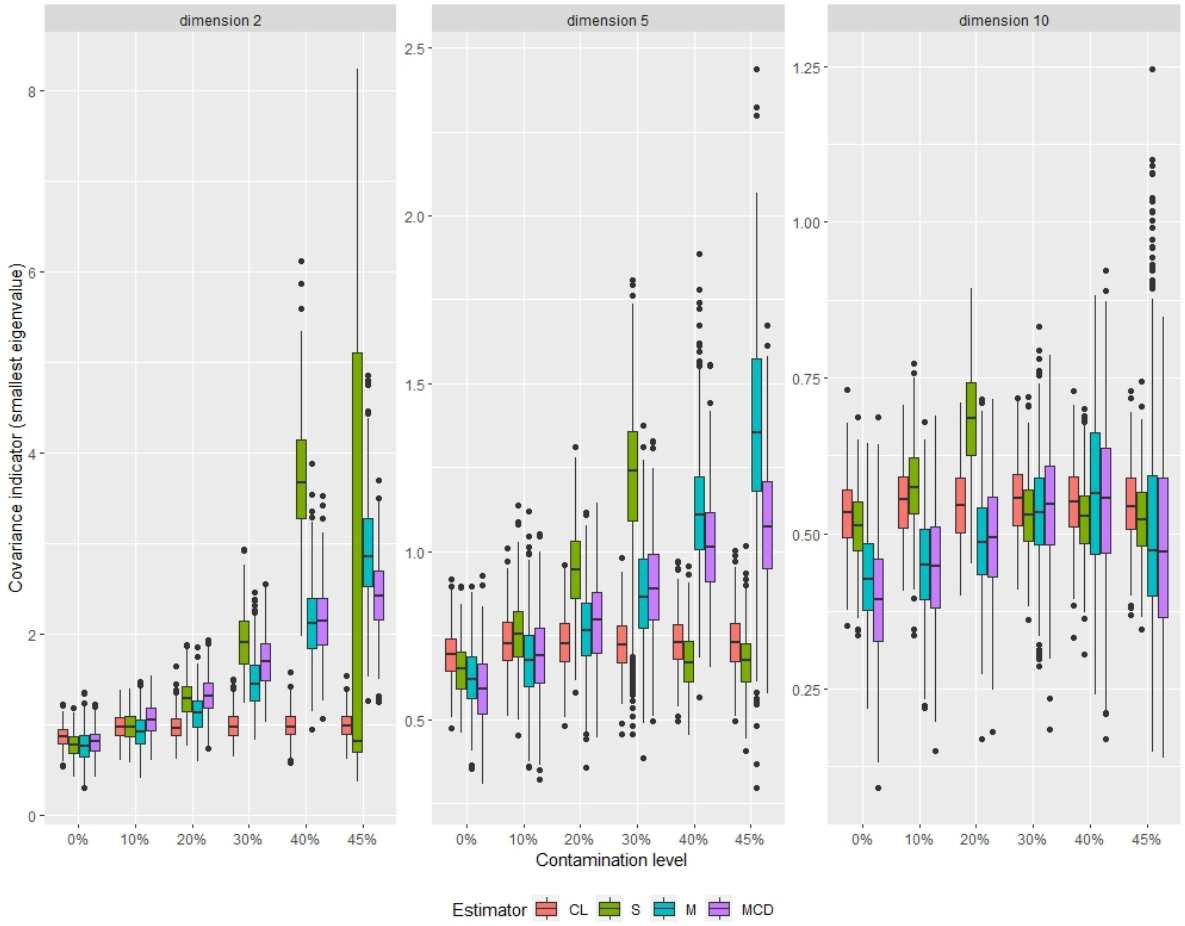

Figure 4.1.2.2: Covariance indicator (smallest eigenvalue) by mean shift of 10

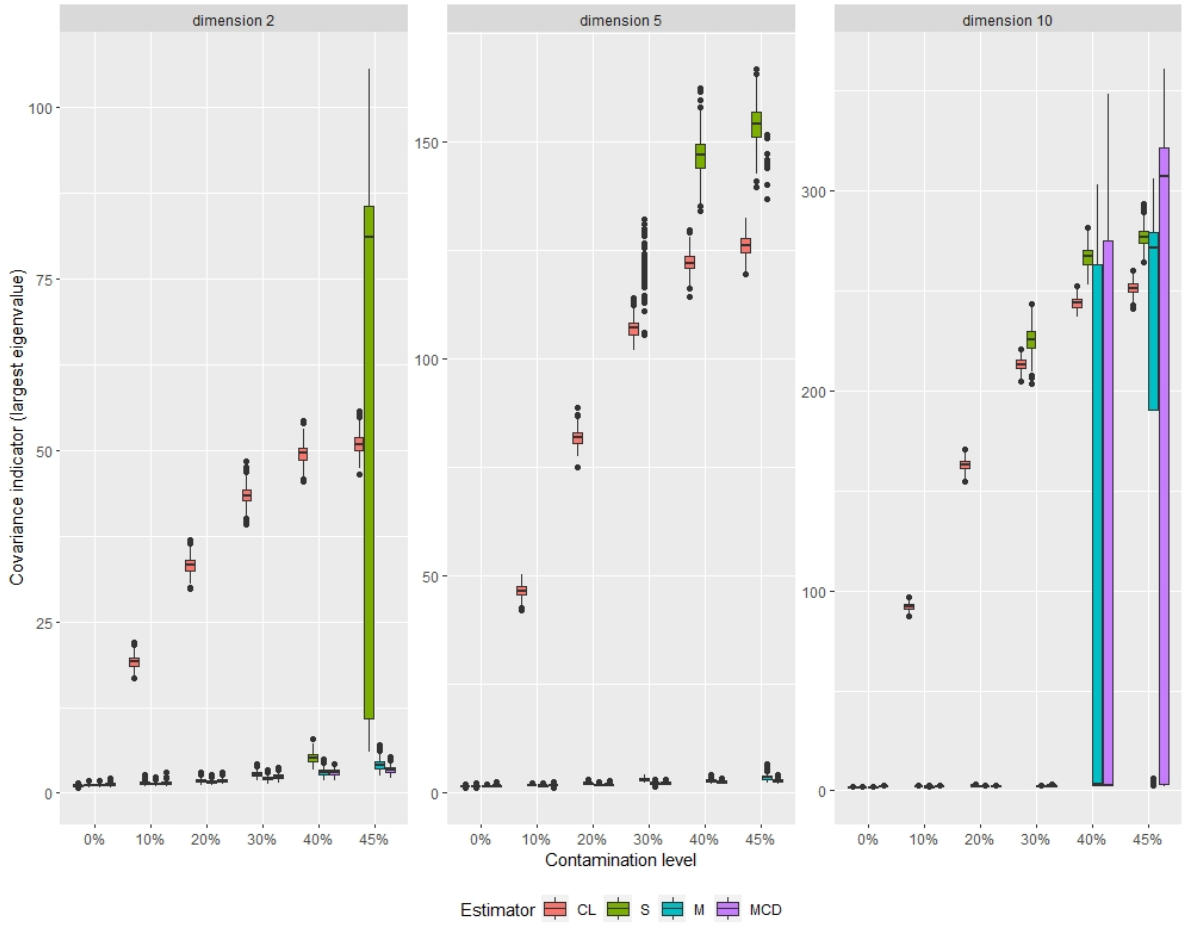

Figure 4.1.2.3: Covariance indicator (largest eigenvalue) by mean shift of 10

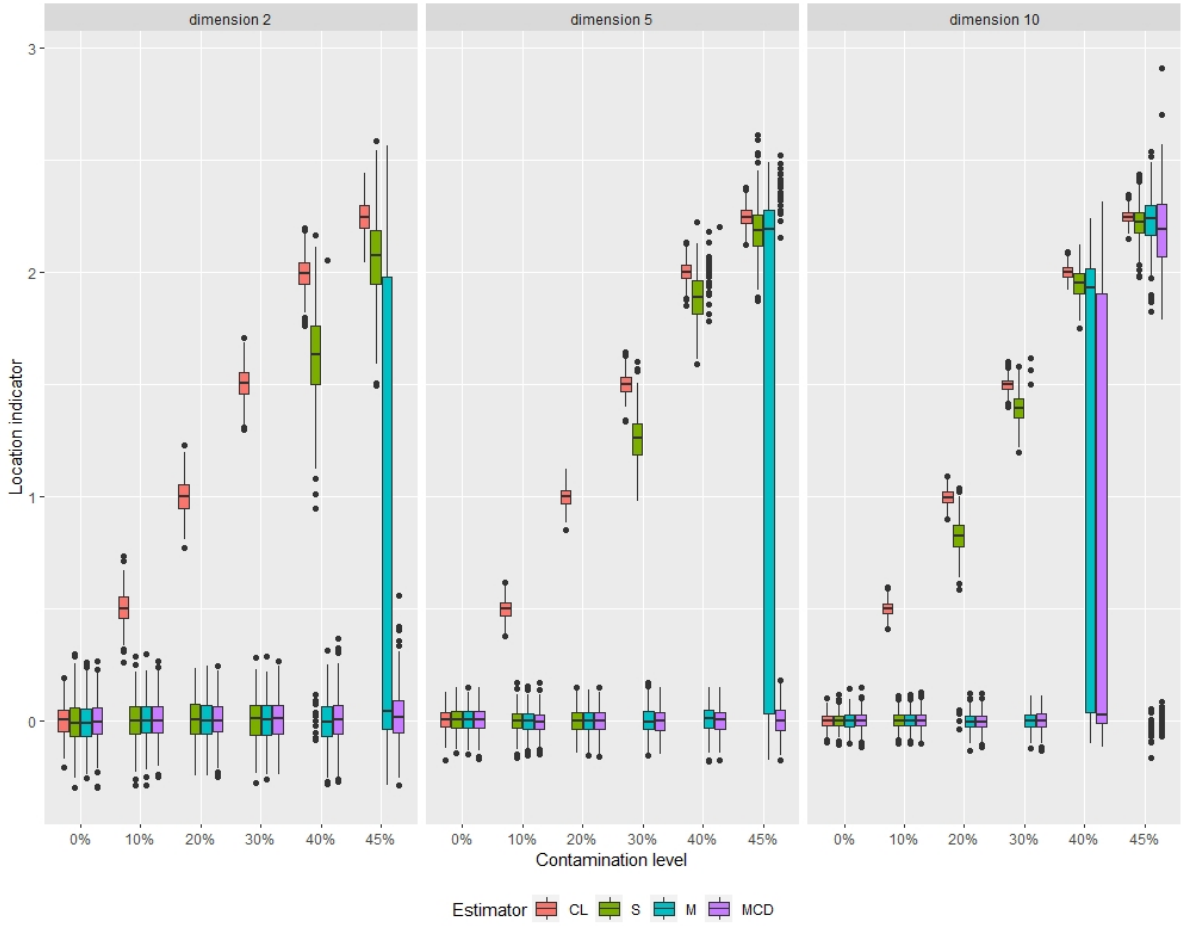

Figure 4.1.3.1: Location indicator by mean shift of 5

still centered around the true value. At dimension 10 the S-estimator break from 30% contamination, MCD and M estimators are inaccurate from 40% contamination onward.

4.1.3. Mean shift of 5

The 2nd simulation is carried out with contamination given by a mean shift of 5.

The plot in Figure 4.1.3.1 shows the location indicator for different levels of contamination and different dimensions by a mean shift of 5. In this case, already at dimension 2 the S-estimator breaks from 40% contamination and the M-estimator at 45% contamination whereas the MCD withstands all contamination levels. At dimension 5 the S-estimator starts providing non-accurate estimates already with 30% contamination, other non-classic estimators present the same results as for the previous dimension. At dimension 10 the S-estimator breaks already with 20% contamination, and in this case, both MCD and M estimators fail to provide accurate results from 40% contamination.

The plot in Figure 4.1.3.2 shows the covariance indicator (smallest eigenvalue) for different levels of contamination and different dimensions by a mean shift of 5. Given this scenario, the estimated smallest eigenvalue starts below 1 and it decreases with an increase in dimension size. The estimate by S-estimator moves downwards at dimension 2 from 40% contamination, and at dimension 5 from 30%. At dimension 10 all non-classic estimators record a small fluctuation in estimate values. Again the classic estimator is the steadiest.

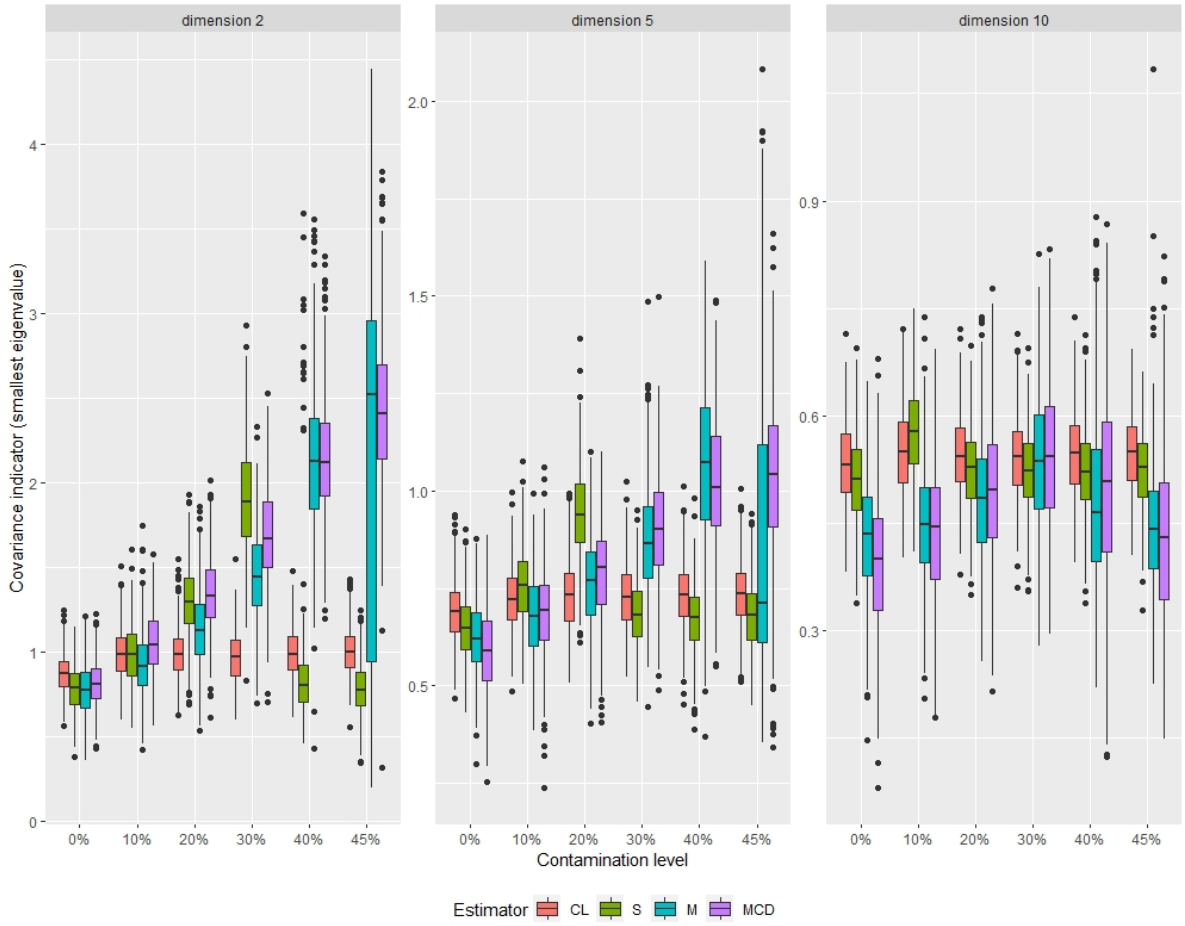

Figure 4.1.3.2: Covariance indicator (smallest eigenvalue) by mean shift of 5

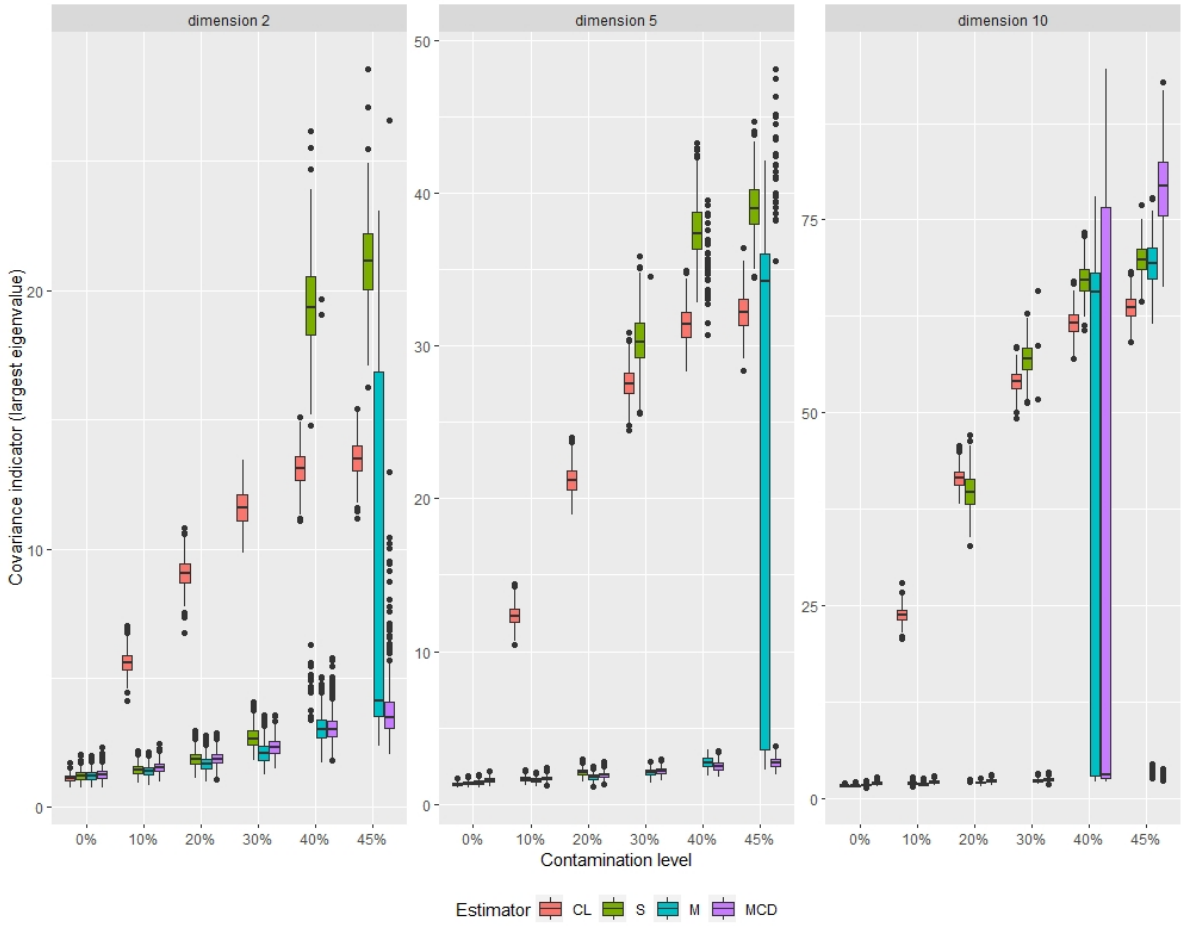

Figure 4.1.3.3: Covariance indicator (largest eigenvalue) by mean shift of 5

The plot in Figure 4.1.3.3 shows the covariance indicator (largest eigenvalue) for different levels of contamination and different dimensions by a mean shift of 5. At dimension 2 the S-estimator breaks from 40% contamination, the M-estimator with 45%. At dimension 5 the S-estimator breaks from 30% contamination and the M-estimator still with 45%. In both cases, the MCD estimator performs well. At dimension 10 the S-estimator does not perform well from 20% contamination onwards, M and MCD estimators fail to deliver accurate estimates from 40% contamination onwards.

4.1.4. Observations of contamination by mean shift

By putting the contamination far from the true mean the behavior of the location indicator for the classic estimator is exactly as the one described in Example 2.1. In fact, the estimator tries to adapt to the whole contaminated dataset. Analogously, the covariance indicator (largest eigenvalue) for the classic estimator behaves as in Example 2.3. To be observed as well is that the S-estimator's ability to distinguish between uncontaminated points and contaminated ones is less effective compared to the MCD in situations where the contamination level gets higher or when the data dimensionality increases. This phenomenon could be attributed to the subsampling algorithm used to compute the S-estimator. Moreover, the M-estimator exhibits significantly better performance than what was initially anticipated. This might be due to the kind of contamination used, namely a shift from the true mean. It could be possible that with a different contamination that causes implosion, such as an extreme data point as in Example 2.4, the M-estimator would break at lower levels of contamination. These particular behaviors of the S-estimator, the M-estimator, and the MCD are observed for both location indicator and covariance indicator (largest eigenvalue).

Furthermore, the performance and behaviors described above are to be observed in all mean shifts, although they are amplified as the distance from the true mean gets smaller. In particular, all non-classic estimators seem to be less capable to identify the contaminated points the closer they get to the uncontaminated ones.

4.2. Simulations' anomalies

Theoretically, the S-estimator should have a higher breakdown point than the M-estimator and a similar one to the MCD. Although we see that the S-estimator generally performs worst. Specifically, this fact is observed by the following settings:

1. **Mean shift of 25**:
    - at dimension 5 with 40% and 45% contamination;
    - at dimension 10 with 30% contamination.
2. **Mean shift of 10**:
    - at dimension 2 with 45% contamination;
    - at dimension 5 from 40% contamination;
    - at dimension 10 with 30% contamination.
3. **Mean shift of 5**:

- at dimension 2 from 40% contamination;
- at dimension 5 with 30% and 40% contamination;
- at dimension 10 with 20% and 30% contamination.

This finding leads to a deep analysis of the algorithms that the function CovSest() in the rrcov library uses to compute the S-estimator. We do so by graphically comparing the behavior of three methods available to compute the S-estimator, namely the default method, called sfast, and other two methods, called bisquare and rocke.

To understand the difference in estimation between the different algorithms it is essential to look deeper into how the algorithms are built in the first place.

4.2.1. Sfast method

The algorithm corresponding to the method sfast of the function CovSest() in the rrcov library is a subsampling algorithm. It is based on the FAST-LTS algorithm of Rousseeuw and Van Driessen (2006) and the SURREAL algorithm of Ruppert (1992). The algorithm of the sfast method for multivariate location and scatter is based on modifying each candidate to improve the S-optimality criterion as in (3.6) thus reducing the number of the necessary sub-samples required to achieve desired high breakdown point with high probability.

4.2.2. Bisquare method

The algorithm corresponding to the method bisquare of the function CovSest() in the rrcov library is an iterative process that uses the relation between S and M-estimators. In particular by Huber (1981):

Definition 4.1 (M-estimators)**.** The M-estimate based upon $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ is defined as solutions of the simultaneous equations:

$$\begin{aligned} \frac{1}{n}\sum_{i=1}^{n} v_1(d_i)(\mathbf{x}_i - \mathbf{t}) &= \mathbf{0} \\ \frac{1}{n}\sum_{i=1}^{n} \left\{ v_2(d_i^2)(\mathbf{x}_i - \mathbf{t})(\mathbf{x}_i - \mathbf{t})^T - v_3(d_i)\mathbf{C} \right\} &= \mathbf{0}, \end{aligned} \tag{4.1}$$

where $d_i = d(\mathbf{x}_i, \mathbf{t}, \mathbf{C})$ and v_1, v_2 and v_3 are real valued functions on $[0, \infty)$.

Lopuhaä (1989) (Section 2.3) shows that the solution of the minimization problem (3.6), which corresponds to the S-estimator, is also a solution of the simultaneous equations (4.1), for $v_1(d) = w(d)$, $v_2(d) = pw(d)$, $v_3(d) = v(d)$ where $\psi(d) = \rho'(d)$, $w(d) = \psi(d)/d$, $v(d) = \psi(d)d$, with constraint (3.6). (4.1) can be rewritten as fixed points equations, namely:

$$\begin{aligned} \mathbf{t} &= \frac{\sum_{i=1}^{n} v_1(d_i) x_i}{\sum_{i=1}^{n} v_1(d_i)} \\ \mathbf{C} &= \frac{\sum_{i=1}^{n} v_2(d_i)(x_i - \mathbf{t})(x_i - \mathbf{t})^T}{\sum_{i=1}^{n} v_3(d_i)} \end{aligned} \tag{4.2}$$

Note that a solution of the minimization problem of (3.6) must satisfy (4.2). A solution to the fixed point equations (4.2) can thus be found iteratively by correcting in every iteration the covariance estimator to satisfy the S-constraint (3.6).

To understand how the iterative algorithm of the bisquare method works denote by $\hat{\mathbf{t}}$ and $\hat{\mathbf{C}}$ the S-estimate for location and covariance, respectively, then since the local minima of $\hat{\mathbf{C}}$ are solutions of (4.1), a natural procedure to minimize $\hat{\mathbf{C}}$ is as follows: assume we have initial values $\mathbf{t}^{(0)}$ and $\mathbf{C}^{(0)}$ given by a high breakdown point estimate (generally minimum volume ellipsoid estimate is chosen). Set iteration j to 0, then:

1. compute indexes $j+1$ from j;
2. compute Mahalanobis distances $d_i^{(j)} = \left[(x_i - \mathbf{t}^{(j-1)})^T (\mathbf{C}^{(j-1)})^{-1} (x_i - \mathbf{t}^{(j-1)})\right]^{1/2}$
3. compute $k^{(j)}$ as the solution of the constraint equation $(1/n)\sum_{i=1}^{n} \rho(d_i^{(j)}/k^{(j)}) = b_0$
4. replace $d_i^{(j)}$ with adjusted distances $\bar{d}_i^{(j)} = d_i^{(j)}/k^{(j)}$
5. compute the new location vector

$$\mathbf{t}_i^{(j)} = \frac{\sum_{i=1}^{n} \nu_1(\bar{d}_i^{(j)}) x_i}{\sum_{i=1}^{n} \nu_1(\bar{d}_i^{(j)})}$$

6. compute the new shape matrix

$$\mathbf{C}^{(j)} = \frac{\sum_{i=1}^{n} \nu_2(\bar{d}_i^{(j)})(x_i - \mathbf{t}^{(j)})(x_i - \mathbf{t}^{(j)})^T}{\sum_{i=1}^{n} \nu_3(\bar{d}_i^{(j)})}$$

7. if the change in $\mathbf{C}^{(j)}$ and $\mathbf{t}^{(j)}$ is below a given tolerance, return $\mathbf{C}^{(j)}$ and $\mathbf{t}^{(j)}$

where $\nu = \psi$. Moreover, the ρ function that this algorithm uses is the Tukey's biweight function of Example 3.1.

4.2.3. Rocke method

The algorithm corresponding to the method rocke of the function CovSest() in the rrcov library is an iterative algorithm similar to the one of the bisquare method. As starting values for the iteration the minimum volume ellipsoid is chosen as default as a high breakdown point. Because in Rocke (1996) it is shown that S-estimators in high dimensions can be sensitive to outliers even if the breakdown point is set to 50%, the algorithm is designed to down weight the influence of contaminated values by iteratively adjusting the weights assigned to observations. It does so by choosing a particular ρ function in (3.6) called translated biweight (or t-biweight) and by replacing the standardization step given in (3.6) with a standardization step consisting of equating the median of $\rho(d_i)$ with the median under normality, where d_i is defined as in Table 1. The specifics of the iteration are given in Rocke and Woodruff (1996). Figure 4.2.4.2 gives confirmation of the better working of the rocke algorithm in higher dimensions.

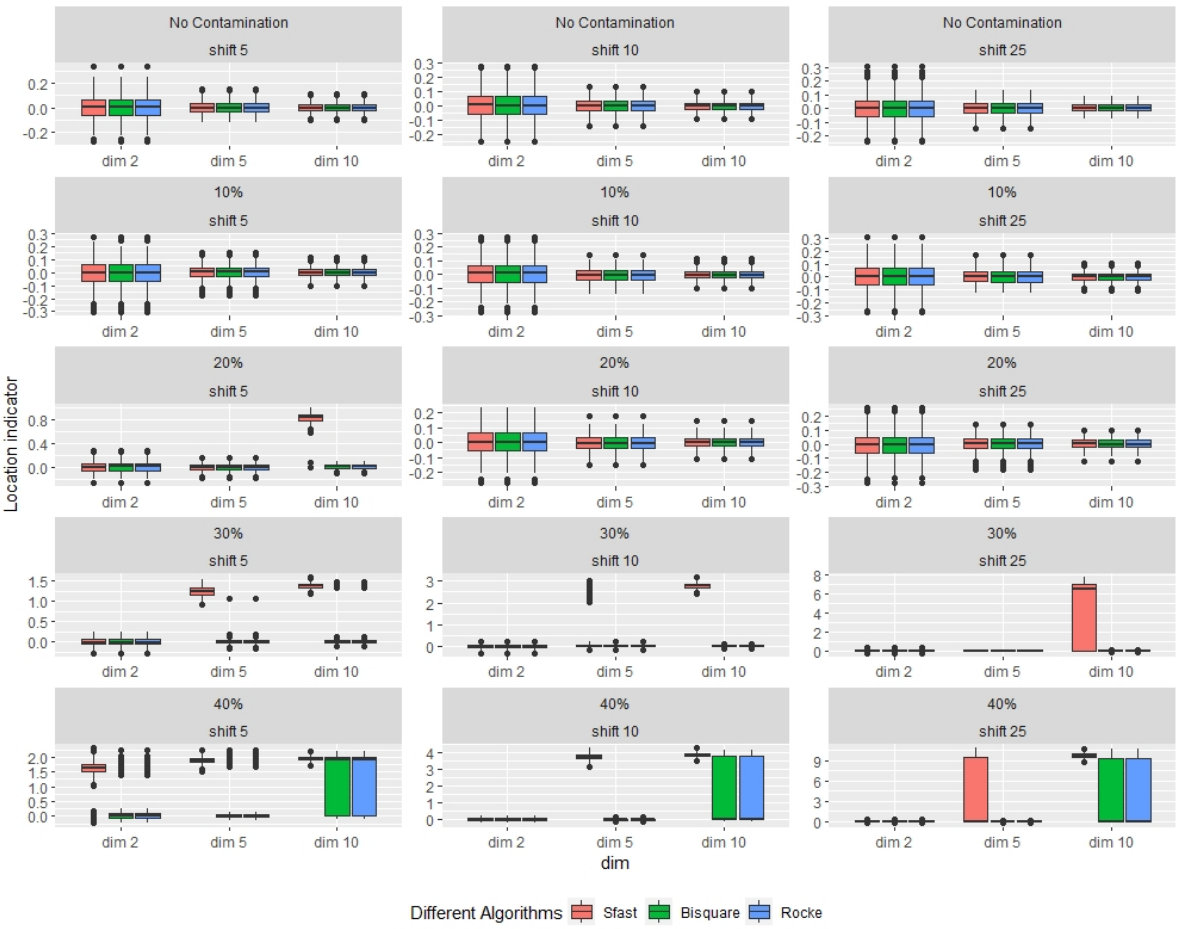

Figure 4.2.4.1: Location indicator for sfast, bisquare, and rocke algorithm

4.2.4. Algorithms' comparison

In order to assess the performance of the S-estimator computed with the different methods, the location indicator as in the previous section is used as well as a new indicator given by the boxplot of a vector in which the determinant values of the estimated covariance matrix is stored at each iteration. With the latter, the performance will be assessed as good if the indicator is centered around 1 as 1 would be the value of the true determinant. Note that theoretically, the S-estimator has a maximal finite sample breakdown point given by $(n-p)/(2n)$ where n is the sample size, so with $n = 100$ and $p = 10$, this is exactly 45%. Therefore we will omit the contamination level of 45%.

In Figure 4.2.4.1 the location indicator is computed among all different settings of the mean shift, contamination level, and dimension. The contamination settings in which the S-estimator by sfast method performs worse than MCD and M-estimator correspond exactly to the ones in which the S-estimator by sfast method performs worst than the one by bisquare and rocke methods, namely by a mean shift of 25 at dimension 5 with 40% contamination and at dimension 10 with 30% contamination, by a mean shift of 10 at dimension 5 with 40% contamination, and at dimension 10 with 30% contamination, and by a mean shift of 5 at dimension 2 with 30% contamination, at dimension 5 with 30% and 40% contamination, and at dimension 10 with 20% and 30% contamination.

In Figure 4.2.4.2 the covariance indicator (determinant) is computed among all differ-

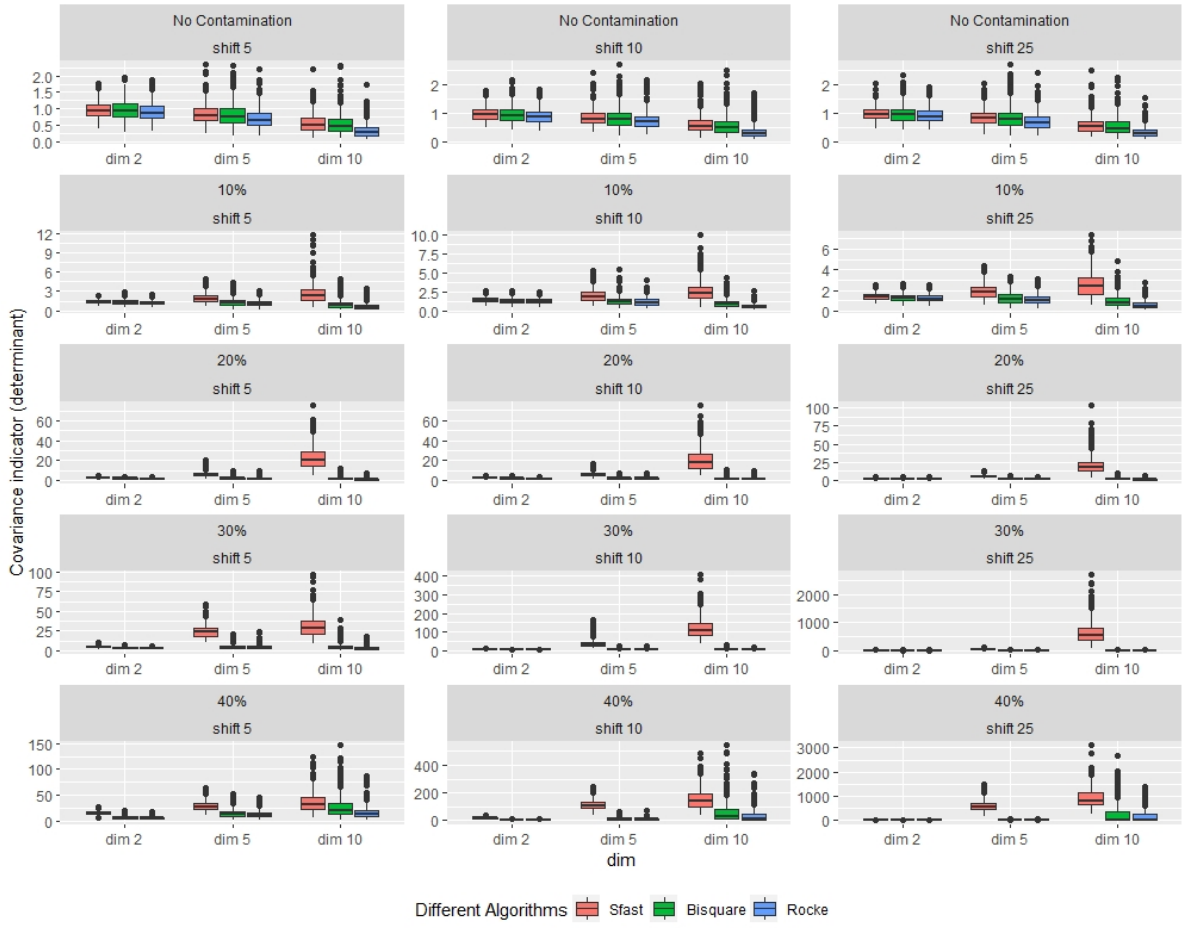

Figure 4.2.4.2: Covariance indicator (determinant) for sfast, bisquare, and rocke algorithm

ent settings of mean shift, contamination level, and dimension. Also in this case we see the same behaviors as with the previous indicator. To be noted, however, is that the S-estimator computed with the rocke method performs better in higher dimensions than the one computed with both the sfast and bisquare algorithms. This particular result can be seen very clearly at dimension 10 with 40% contamination and by all mean shifts, and it is indeed what we expected due to translated biweight function mentioned in Section 4.2.3.

4.3. Anomalous indicators' behaviors

Throughout the simulations, it is to be seen that the S-estimator presents, in some specific contamination settings, indicators (both for location and covariance) that are not symmetric around the median but stretched. For example, by mean shift of 25 at dimension 10 with contamination 30% we can observe stretched boxplots for the S-estimator in all indicators.

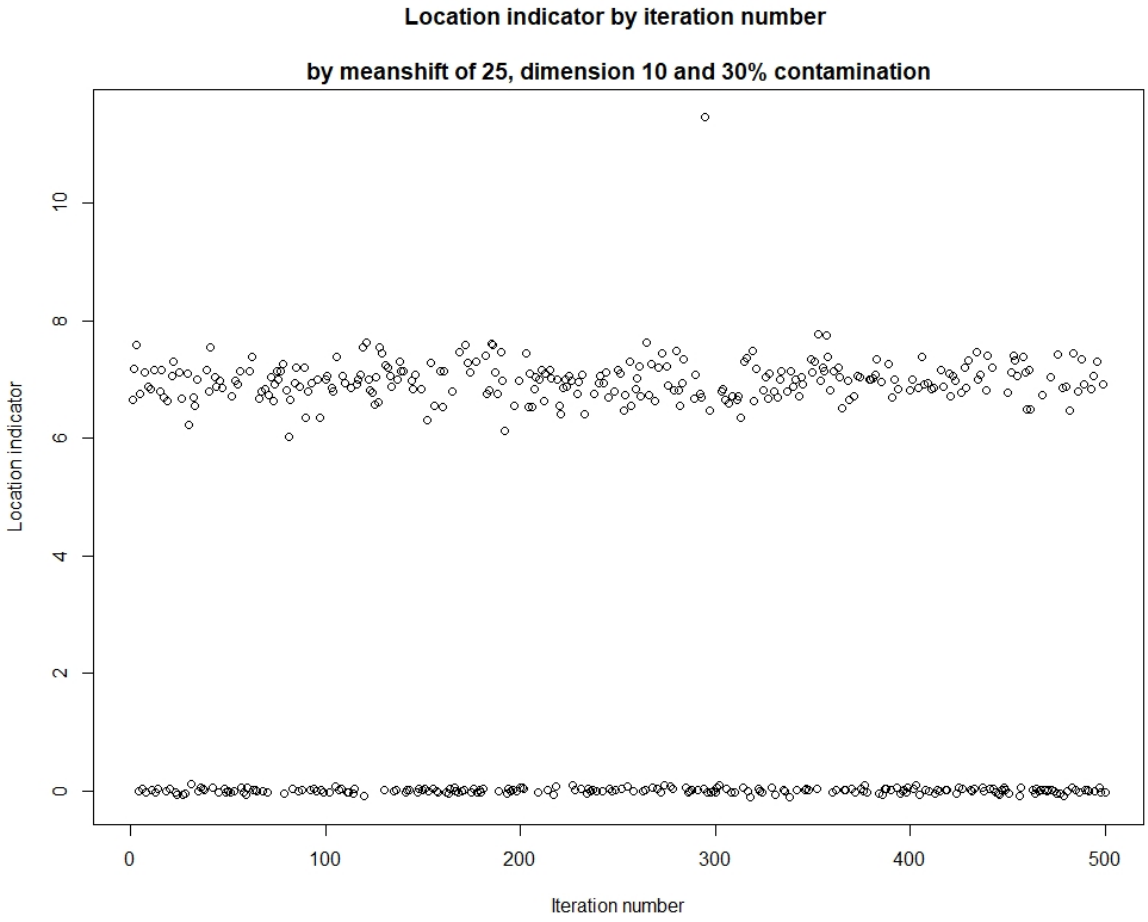

Figure 4.3.1: Location indicator for S-estimator by iteration number

In Figure 4.3.1 the location indicator by S-estimator is plotted against the iteration number of the simulation by mean shift of 25 at dimension 10 with contamination 30%. The stretched behavior of the location indicator of Figure 4.1.1.1 is due to the fact that the S-estimator estimates roughly half of the time a value close to the true mean and half not. The S-estimator was computed by means of the sfast algorithm, which is a subsampling algorithm.

Conjecture 4.1 (Sfast algorithm)**.** The sfast algorithm is not capable of recognizing the contaminated points which explains why the optimal subsample returned has roughly half of its points within the contaminated data and half within the uncontaminated data.

Unfortunately, the rrcov library does not offer the possibility of viewing the optimal subset to be able to confirm the previously mentioned hypothesis. In order to be able to have a confirmation we looked at the behavior of the MCD estimator, as it is also computed by means of a subsampling algorithm and the library rrcov offers the possibility to extract the optimal subsample. As we can see from Figure 4.1.1.1, Figure 4.1.1.2 and Figure 4.1.1.3 at dimension 10 and 40% contamination the indicators corresponding to the MCD estimator are stretched. So we fixed that specific contamination setting and plotted it against the whole set.

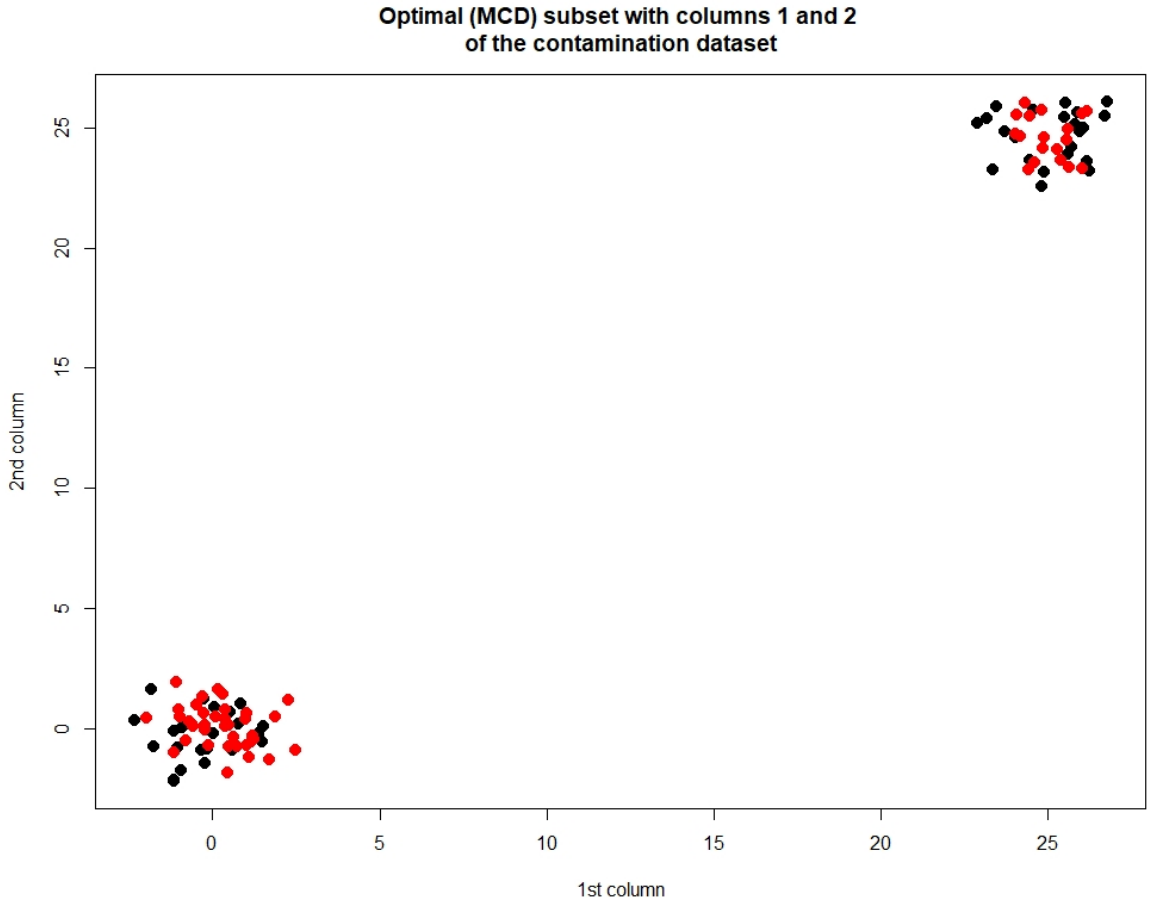

Figure 4.3.2: Optimal MCD subsample

Figure 4.3.2 shows in red the points of the optimal MCD subsample and in black the contaminated dataset as computed in Section 4.1. It is very clear that the subsampling algorithm is not capable of recognizing the contaminated points returning a subsample that picks, as expected, roughly half of the points from the uncontaminated data (left bottom of the plot), and roughly half from the contaminated one (right top of the plot).

5

Conclusions

This study focused on the robustness of S-estimators and explored their behavior in different scenarios by means of a simulation study. We have made several key observations that shed light on the performance of S-estimators and their dependence on the choice of the algorithm used to compute them.

Firstly, while the S-estimator possesses a high theoretical breakdown point, up to 50% for the right choice of the constant c_0 as in Equation (3.11), we found that it tends to break earlier than expected in practice. This indicates that the robustness of the S estimator may be more limited in real-world situations, highlighting the need for caution when relying solely on its breakdown point as a measure of robustness. In particular, we have seen that the S-estimator performs generally worse than the minimum covariance determinant (MCD), which also has a maximum breakdown point equal to 50% (see Maronna et al. (2006)).

Secondly, we observed that the behavior of the S-estimator is closely linked to the choice of the algorithm employed. Our analysis revealed that S-estimators computed by different algorithms exhibit varying degrees of robustness. This underscores the importance of carefully considering the algorithm selection when applying S-estimators in practice, as it can significantly impact their performance and resilience to outliers. More specifically we have seen that the subsampling algorithm corresponding to the sfast method, which happens to be the default choice for S-estimators computation in the library rrcov, fails to deliver accurate results from relatively low contamination levels already, as opposed to the iterative algorithms, corresponding to the bisquare and rocke methods, with the latter being more accurate in higher dimensions.

Surprisingly, we found that the M-estimator, another commonly used estimator, exhibited higher resistance to breakdown than initially anticipated. By Maronna et al. (2006) its breakdown point is given by $1/(p+1)$ which means that theoretically, it should break with very low contamination levels by higher dimensions. Instead, throughout the simulation study, it delivers good estimates of up to 30% contamination even in higher dimensions. This unforeseen result could be explained by the specific kind of contamination used. It might very well be that a different contamination would cause breakdown in lower contamination levels.

Additionally, our simulation study led us to develop Conjecture 4.1 regarding the behavior of the subsampling algorithm sfast used to compute S-estimators. We observed that

in situations where the collected estimated data are distributed asymmetrically around its center, the optimal subsample returned by the algorithm picks roughly half of its points from the contaminated data, and half from the non-contaminated one.

In summary, our study on the robustness of S-estimators highlighted the complexities involved in assessing their performance in specific scenarios. The choice of the algorithm, the behavior of the M-estimator, and the potential utility of conjecture 4.1 all emerged as critical factors to consider. These findings contribute to the ongoing understanding of robust estimation techniques and provide valuable insights for practitioners in choosing appropriate methods for robust data analysis.

Further research in this area is recommended to refine our understanding and explore additional aspects of robustness in estimation methods.

6

Appendix

6.1. R-code for estimators' comparison

```
library(rrcov)
library(robustbase)
library(MASS)
library(dplyr)
library(stats4)
library(ggplot2)

sample_size = 100
nrep=500


contamination_size = c(0, 0.10, 0.20, 0.30, 0.40, 0.45)
dimension_size = c(2, 5, 10)
shift_vec = c(5,10,25)

dat=NULL

for (k in 1:3){
  for (i in 1:6){
    for (j in 1:3){
      shift=shift_vec[k]
      c_n=contamination_size[i]
      eps=1-c_n
      p=dimension_size[j]
      n=sample_size

      Sigma = diag(p)
      Mu = rep(0, p)

      #creating vectors to store estimated data
      mu_vector_sample_mean = vector('numeric', nrep)
```

```
mu_vector_s_est = vector('numeric', nrep)
mu_vector_m_est = vector('numeric', nrep)
mu_vector_MCD_est = vector('numeric', nrep)

largest_eig_vector_sample_var = vector('numeric', nrep)
largest_eig_vector_s_est = vector('numeric', nrep)
largest_eig_vector_m_est = vector('numeric', nrep)
largest_eig_vector_MCD_est = vector('numeric', nrep)

smallest_eig_vector_sample_var = vector('numeric', nrep)
smallest_eig_vector_s_est = vector('numeric', nrep)
smallest_eig_vector_m_est = vector('numeric', nrep)
smallest_eig_vector_MCD_est = vector('numeric', nrep)

for (m in 1:nrep){
  #creating contaminated data at every iteration
  # treat c_n=0 separately
  if (eps==1){contamination = mvrnorm(n * eps, mu = Mu,
                                      Sigma = Sigma)}
  else{
    #original data
    sim_data_clean = mvrnorm(n * (eps), mu = Mu,
                             Sigma = Sigma)
    #contaminated data
    sim_data_cont = mvrnorm(n * c_n, mu = rep(shift, p),
                            Sigma = Sigma)
    #dataset with conatmination
    contamination = rbind(sim_data_clean, sim_data_cont)
    } # END ifelse


  #sample mean
  mu_vector_sample_mean[m] = mean(CovClassic(contamination)$center)

  #computing estimators for contaminated data
  s_est = CovSest(contamination)
  m_est = CovMest(contamination)
  MCD_est = CovMcd(contamination)

  #assign mean values of estimators to their vectors
  mu_vector_s_est[m] = mean(s_est@center)
  mu_vector_m_est[m] = mean(m_est@center)
  mu_vector_MCD_est[m] = mean(MCD_est@center)

  # getting largest and smallest eigenvalues
  #of estimated covariance matrices:
```

```
    largest_eig_vector_sample_var[m] =
      max(eigen(cov(contamination))$values)
    largest_eig_vector_s_est[m] = max(eigen(s_est@cov)$values)
    largest_eig_vector_m_est[m] = max(eigen(m_est@cov)$values)
    largest_eig_vector_MCD_est[m] = max(eigen(MCD_est@cov)$values)

    smallest_eig_vector_sample_var[m] =
      min(eigen(cov(contamination))$values)
    smallest_eig_vector_s_est[m] = min(eigen(s_est@cov)$values)
    smallest_eig_vector_m_est[m] = min(eigen(m_est@cov)$values)
    smallest_eig_vector_MCD_est[m] = min(eigen(MCD_est@cov)$values)

  } # END iteration loop

# storing results
CL=cbind(rep(shift,times=nrep),
         rep(c_n,times=nrep),
         rep(p,times=nrep),
         rep(1,times=nrep),
         mu_vector_sample_mean,
         smallest_eig_vector_sample_var,
         largest_eig_vector_sample_var)

S=cbind(rep(shift,times=nrep),
        rep(c_n,times=nrep),
        rep(p,times=nrep),
        rep(2,times=nrep),
        mu_vector_s_est,
        smallest_eig_vector_s_est,
        largest_eig_vector_s_est)

M=cbind(rep(shift,times=nrep),
        rep(c_n,times=nrep),
        rep(p,times=nrep),
        rep(3,times=nrep),
        mu_vector_m_est,
        smallest_eig_vector_m_est,
        largest_eig_vector_m_est)

MCD=cbind(rep(shift,times=nrep),
          rep(c_n,times=nrep),
        rep(p,times=nrep),
        rep(4,times=nrep),
        mu_vector_MCD_est,
        smallest_eig_vector_MCD_est,
        largest_eig_vector_MCD_est)
```

```
    dat=rbind(dat,CL,S,M,MCD)
    } # END j-loop
  }# END i-loop
}# END k-loop

dat=data.frame(dat)
colnames(dat)=c("shift","Contamination level","dim","Estimator",
                "Location indicator",
                "Covariance indicator (smallest eigenvalue)",
                "Covariance indicator (largest eigenvalue)")
View(dat)


#assigning factor class to factor columns
dat[,1]=factor(dat[,1],levels=shift_vec,labels=c("shift 5","shift 10",
                                                 "shift 25"))
dat[,2]=factor(dat[,2], levels = contamination_size,
               labels = c("0%", "10%", "20%", "30%", "40%", "45%"))
dat[,3]=factor(dat[,3],levels=dimension_size,
               labels=c("dimension 2","dimension 5","dimension 10"))
dat[,4]=factor(dat[,4],levels=1:4,labels = c("CL","S","M","MCD"))


#grouped ggplots
ggplot(dat[dat$shift == 'shift 25', ],
       aes(x=`Contamination level`, y=`Location indicator`,
           fill=Estimator)) +
  geom_boxplot()+
  facet_wrap(~dim) +
  theme(legend.position = "bottom")

ggplot(dat[dat$shift == 'shift 5', ],
       aes(x=`Contamination level`,
           y=`Covariance indicator (smallest eigenvalue)`,
                                        fill=Estimator)) +
  geom_boxplot()+
  facet_wrap(~dim, scales = 'free') +
  theme(legend.position = "bottom")

ggplot(dat[dat$shift == 'shift 5', ],
       aes(x=`Contamination level`,
           y=`Covariance indicator (largest eigenvalue)`,
           fill=Estimator)) +
  geom_boxplot()+
  facet_wrap(~dim, scales = 'free') +
```

```
  theme(legend.position = "bottom")
```

6.2. R-code for algorithms' comparison

```
library(rrcov)
library(robustbase)
library(MASS)
library(dplyr)
library(stats4)
library(ggplot2)

sample_size = 100
nrep=500


contamination_size = c(0, 0.10, 0.20, 0.30, 0.40)
dimension_size = c(2, 5, 10)
shift_vec = c(5,10,25)

dat_rocke=NULL

for (k in 1:3){
  for (i in 1:5){
    for (j in 1:3){
      shift=shift_vec[k]
      c_n=contamination_size[i]
      eps=1-c_n
      p=dimension_size[j]
      n=sample_size

      Sigma = diag(p)
      Mu = rep(0, p)


      mu_vector_s_est_sfast = vector('numeric', nrep)
      mu_vector_s_est_bisquare = vector('numeric', nrep)
      mu_vector_s_est_rocke = vector('numeric', nrep)

      cov_s_est_sfast_smallest = vector('numeric', nrep)
      cov_s_est_bisquare_smallest = vector('numeric', nrep)
      cov_s_est_rocke_smallest = vector('numeric', nrep)

      cov_s_est_sfast_largest = vector('numeric', nrep)
      cov_s_est_bisquare_largest = vector('numeric', nrep)
      cov_s_est_rocke_largest = vector('numeric', nrep)
```

```
determinant_vector_sfast = vector('numeric', nrep)
determinant_vector_bisquare = vector('numeric', nrep)
determinant_vector_rocke = vector('numeric', nrep)

for (m in 1:nrep){
  #creating contaminated data at every iteration
  # treat c_n=0 separately
  if (eps==1){contamination = mvrnorm(n * eps,
                                      mu = Mu, Sigma = Sigma)}
  else{
    sim_data_clean = mvrnorm(n * (eps), mu = Mu, Sigma = Sigma)
    sim_data_cont = mvrnorm(n * c_n,
                            mu = rep(shift, p), Sigma = Sigma)
    contamination = rbind(sim_data_clean, sim_data_cont)
  } # END ifelse




  #computing estimators for contaminated data
  s_est_sfast = CovSest(contamination)
  s_est_bisquare = CovSest(contamination, method = 'bisquare')
  s_est_rocke = CovSest(contamination, method = 'rocke')


  #assign mean values of estimators to their vectors
  mu_vector_s_est_sfast[m] = mean(s_est_sfast$center)
  mu_vector_s_est_bisquare[m] = mean(s_est_bisquare$center)
  mu_vector_s_est_rocke[m] = mean(s_est_rocke$center)




  #assign  to their vector:
  cov_s_est_sfast_smallest[m] = min(eigen(s_est_sfast$cov)
                                    $values)
  cov_s_est_bisquare_smallest[m] = min(eigen(s_est_bisquare$cov)
                                       $values)
  cov_s_est_rocke_smallest[m] = min(eigen(s_est_rocke$cov)
                                    $values)

  cov_s_est_sfast_largest[m] = max(eigen(s_est_sfast$cov)
                                   $values)
  cov_s_est_bisquare_largest[m] = max(eigen(s_est_bisquare$cov)
                                      $values)
  cov_s_est_rocke_largest[m] = max(eigen(s_est_rocke$cov)
```

```
                                        $values)


    #getting determinants of estimated covariance:
    determinant_sfast = det(s_est_sfast$cov)
    determinant_bisquare = det(s_est_bisquare$cov)
    determinant_rocke = det(s_est_rocke$cov)
    #assign to their vector:
    determinant_vector_sfast[m] = determinant_sfast
    determinant_vector_bisquare[m] = determinant_bisquare
    determinant_vector_rocke[m] = determinant_rocke
  } # END iteration loop

  # storing results


  S_sfast=cbind(rep(shift,times=nrep),
                rep(c_n,times=nrep),
                rep(p,times=nrep),
                rep(1,times=nrep),
                mu_vector_s_est_sfast,
                cov_s_est_sfast_smallest,
                cov_s_est_sfast_largest,
                determinant_vector_sfast)

  S_bisquare=cbind(rep(shift,times=nrep),
                   rep(c_n,times=nrep),
                   rep(p,times=nrep),
                   rep(2,times=nrep),
                   mu_vector_s_est_bisquare,
                   cov_s_est_bisquare_smallest,
                   cov_s_est_bisquare_largest,
                   determinant_vector_bisquare)
  S_rocke=cbind(rep(shift,times=nrep),
                rep(c_n,times=nrep),
                rep(p,times=nrep),
                rep(3,times=nrep),
                mu_vector_s_est_bisquare,
                cov_s_est_rocke_smallest,
                cov_s_est_rocke_largest,
                determinant_vector_rocke)



  dat_rocke = rbind(dat_rocke, S_sfast, S_bisquare, S_rocke)
}
```

```
  }
}
dat_rocke = data.frame(dat_rocke)
colnames(dat_rocke)=c("shift","Contamination level","dim",
                      "Different Algorithms",
                      "Location indicator",
                      "Covariance indicator (smallest eigenvalue)",
                      "Covariance indicator(largest eigenvalue)" ,
                      "Covariance indicator (determinant)")
View(dat_rocke)

dat_rocke[,1]=factor(dat_rocke[,1],levels=shift_vec,
                     labels=c("shift 5","shift 10","shift 25"))
dat_rocke[,2]=factor(dat_rocke[,2],levels=contamination_size,
               labels=c("No Contamination","10%","20%","30%","40%"))
dat_rocke[,3]=factor(dat_rocke[,3],levels=dimension_size,
                     labels=c("dim 2","dim 5","dim 10"))
dat_rocke[,4]=factor(dat_rocke[,4],levels=1:3,
                     labels = c("Sfast","Bisquare","Rocke"))



ggplot(dat_rocke, aes(x=dim,
                      y=`Covariance indicator (smallest eigenvalue)`,
                      fill=`Different Algorithms`)) +
  geom_boxplot()+
  facet_wrap(~`Contamination level`+shift,scale="free",
             nrow = 5, ncol = 3) +
  theme(legend.position = "bottom")

ggplot(dat_rocke, aes(x=dim,
                      y=`Location indicator`,
                      fill=`Different Algorithms`)) +
  geom_boxplot()+
  facet_wrap(~`Contamination level`+shift,scale="free",
             nrow = 5, ncol = 3) +
  theme(legend.position = "bottom")

ggplot(dat_rocke, aes(x=dim,
                      y=`Covariance indicator(largest eigenvalue)`,
                      fill=`Different Algorithms`)) +
  geom_boxplot()+
  facet_wrap(~`Contamination level`+shift,scale="free",
             nrow = 5, ncol = 3) +
  theme(legend.position = "bottom")
```

```
ggplot(dat_rocke, aes(x=dim, y=‘Covariance indicator (determinant)‘,
                      fill=‘Different Algorithms‘)) +
  geom_boxplot()+
  facet_wrap(~‘Contamination level‘+shift,scale="free",
             nrow = 5, ncol = 3) +
  theme(legend.position = "bottom")
```

6.3. R-code for Figure 4.3.1

```
library(rrcov)
library(robustbase)
library(MASS)
library(dplyr)
library(stats4)

n = 100
p = 10
c_n_30 = 0.30
eps_30 = 1 - c_n_30
Sigma = diag(p)
Mu = rep(0, p)

#creating 500x10 matrix to contain mu estimate in each row with contamination 30%
mu_sm_30 = matrix(0,ncol=p,nrow=500) #for sample mean
mu_s_30 = matrix(0,ncol=p,nrow=500) # for S-estimator
mu_MCD_30 = matrix(0,ncol=p,nrow=500) # for MCD estimator
mu_m_30 = matrix(0,ncol=p,nrow=500) #for M-estimator


#simulating 500 times 30% contamination with mu shift
#from 0 to 25 in every coordinate
for (i in 1:500){
  #creating contaminated data at every iteration
  sim_data_clean_30 = mvrnorm(n * (eps_30),
                              mu = Mu, Sigma = Sigma)
  sim_data_cont_30 = mvrnorm(n * c_n_30,
                             mu = rep(shift, p), Sigma = Sigma)
  contamination_30 = rbind(sim_data_clean_30, sim_data_cont_30)
  #generating estimates
  s_est_30 = CovSest(contamination_30)
  MCD_est_30 = CovMcd(contamination_30)
  m_est_30 = CovMest(contamination_30)
  #assigning 500 values with coordinate wise
```

```
  #average of the mu estimation
  mu_sm_30[i,] = colMeans(contamination_30)
  mu_s_30[i,] = s_est_30@center
  mu_MCD_30[i,] = MCD_est_30@center
  mu_m_30[i,] = m_est_30@center

}

plot(rowMeans(mu_s_30),
main = 'Location indicator by iteration number
     \nby meanshift of 25, dimension 10 and 30% contamination',
     xlab = 'Iteration number', ylab = 'Location indicator')
```

6.4. R-code for Figure 4.3.2

```
library(rrcov)
library(robustbase)
library(MASS)
library(dplyr)
library(stats4)

n = 100
p = 10
c_n_30 = 0.30
eps_30 = 1 - c_n_30
Sigma = diag(p)
Mu = rep(0, p)

#creating 500x10 matrix to contain mu estimate in each row with contamination 30%
mu_sm_30 = matrix(0,ncol=p,nrow=500) #for sample mean
mu_s_30 = matrix(0,ncol=p,nrow=500) # for S-estimator
mu_MCD_30 = matrix(0,ncol=p,nrow=500) # for MCD estimator
```

```
mu_m_30 = matrix(0,ncol=p,nrow=500) #for M-estimator


#simulating 500 times 30% contamination with mu shift
#from 0 to 25 in every coordinate
for (i in 1:500){
  #creating contaminated data at every iteration
  sim_data_clean_30 = mvrnorm(n * (eps_30),
                              mu = Mu, Sigma = Sigma)
  sim_data_cont_30 = mvrnorm(n * c_n_30,
                             mu = rep(shift, p), Sigma = Sigma)
  contamination_30 = rbind(sim_data_clean_30, sim_data_cont_30)
  #generating estimates
  s_est_30 = CovSest(contamination_30)
  MCD_est_30 = CovMcd(contamination_30)
  m_est_30 = CovMest(contamination_30)
  #assigning 500 values with coordinate wise
  #average of the mu estimation
  mu_sm_30[i,] = colMeans(contamination_30)
  mu_s_30[i,] = s_est_30@center
  mu_MCD_30[i,] = MCD_est_30@center
  mu_m_30[i,] = m_est_30@center

}

plot(rowMeans(mu_s_30),
main = 'Location indicator by iteration number
     \nby meanshift of 25, dimension 10 and 30% contamination',
     xlab = 'Iteration number', ylab = 'Location indicator')
```

Bibliography

P Laurie Davies. Asymptotic behaviour of s-estimates of multivariate location parameters and dispersion matrices. The Annals of Statistics, pages 1269–1292, 1987.

David L Donoho and Peter J Huber. The notion of breakdown point. A festschrift for Erich L. Lehmann, 157184, 1983.

Frank R Hampel. The breakdown points of the mean combined with some rejection rules. Technometrics, 27(2):95–107, 1985.

PJ Huber. Robust statistics wiley 308p. 1981.

Hendrik P Lopuhaä. On the relation between s-estimators and m-estimators of multivariate location and covariance. The Annals of Statistics, pages 1662–1683, 1989.

Hendrik P Lopuhaä and Peter J Rousseeuw. Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. The Annals of Statistics, pages 229–248, 1991.

Ricardo A Maronna, R Douglas Martin, Victor J Yohai, and Matías Salibián-Barrera. Robust statistics: theory and methods (with R). John Wiley & Sons, 2006.

Ricardo Antonio Maronna. Robust m-estimators of multivariate location and scatter. The annals of statistics, pages 51–67, 1976.

David M Rocke. Robustness properties of s-estimators of multivariate location and shape in high dimension. The Annals of statistics, pages 1327–1345, 1996.

David M Rocke and David L Woodruff. Identification of outliers in multivariate data. Journal of the American Statistical Association, 91(435):1047–1061, 1996.

Peter Rousseeuw and Victor Yohai. Robust regression by means of s-estimators. In Robust and Nonlinear Time Series Analysis: Proceedings of a Workshop Organized by the Sonderforschungsbereich 123 “Stochastische Mathematische Modelle”, Heidelberg 1983, pages 256–272. Springer, 1984.

Peter J Rousseeuw. Multivariate estimation with high breakdown point. Mathematical statistics and applications, 8(283-297):37, 1985.

Peter J Rousseeuw and Katrien Van Driessen. Computing lts regression for large data sets. Data mining and knowledge discovery, 12:29–45, 2006.

David Ruppert. Computing s estimators for regression and multivariate location/dispersion. Journal of Computational and Graphical Statistics, 1(3):253–270, 1992.