Robust Estimators are Hard to Compute

Thorsten Bernholt

Lehrstuhl Informatik 2
Universität Dortmund, Germany

January 12, 2006

Abstract

In modern statistics, the robust estimation of parameters of a regression hyperplane is a central problem. Robustness means that the estimation is not or only slightly affected by outliers in the data. In this paper, it is shown that the following robust estimators are hard to compute: LMS, LQS, LTS, LTA, MCD, MVE, Constrained M estimator, Projection Depth (PD) and Stahel-Donoho. In addition, a data set is presented such that the ltsReg-procedure of R has probability less than 0.0001 of finding a correct answer. Furthermore, it is described, how to design new robust estimators.

Keywords: Computational statistics, complexity theory, robust statistics, algorithms, search heuristics

1 Introduction

Robust statistics [6, 9] offers a variety of estimators that are algorithmically interesting. As a drawback, the proved statistical properties of the estimators are only valid if the estimator is computed correctly. For that, exact computation is necessary. In this paper, we will show that the exact computation of the following robust estimators is NP-hard:

- Least Median of Squares (LMS) [8]
- Least Quantile of Squares (LQS) [9]
- Least Trimmed Squares (LTS) [11]
- Least Trimmed Absolute Deviation (LTA) [5]
- Minimum Covariance Determinant (MCD) [10]
- Minimum Volume Ellipsoid (MVE) [15]
- Subset Estimators in general
- Constrained M estimator (CM) [1]
- Projection Depth (PD) [16]
- Stahel-Donoho (SD) [4]

This means, that under the assumption P $\neq$ NP, there is no hope of finding exact algorithms that work in polynomial time. In Section 2, we give a short introduction to the theory of NP-completeness. We show in Section 3 that the popular ltsReg-procedure from R has a bad behavior on special high dimensional data sets. In Section 4, we argue that there is a need for new robust estimators and present a design pattern. Finally in Section 5, the NP-hardness proofs are presented. Here are some notations, which we use in the paper:

- $r_i(\beta)$: Consider the points $P_1, \ldots, P_n$. $r_i(\beta)$ is the residual of the i-th point with respect to a hyperplane with parameter vector β. In this paper, we will use it as a measure of the vertical distance of the point to the hyperplane.
- $r_{(i)}(\beta)$: The i-th residual in the sorted order of all residuals.

- $\lceil 0.4 \rceil = 1$ and $\lfloor 0.6 \rfloor = 0$: Round to the next integer above and below respectively.
- $\begin{pmatrix} a \\ b \end{pmatrix}^\top = \begin{pmatrix} a & b \end{pmatrix}$: Transpose of a vector or matrix.

2 Some notes on NP-hardness

We will show that the computation of the mentioned estimators is NP-hard. If a problem Π is NP-hard, we need more than polynomial time to solve Π, under the assumption P $\neq$ NP. This assumption is widely believed, cf. the keyword "Millennium Problems" [7]. P is the class of problems that can be solved by a deterministic Turing machine ([12]) in a runtime, that is polynomial in the input length. NP is the class of problems that can be solved by a non-deterministic Turing machine with the same time bound. Explanations of P, NP and NP-hardness can be found in [14].

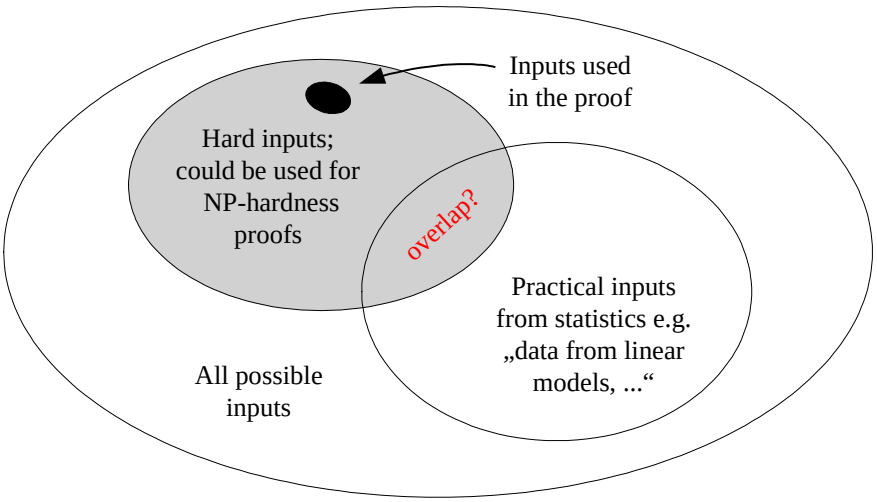

Figure 1: The big ellipse symbolizes the set of all d-dimensional point sets – the inputs for robust estimators. Also sketched is the subset of inputs that arise in practice and that are analyzed by the statistical community. Then there is a grey subset that contains inputs that could be used in NP-hardness proofs. The question is whether both subsets overlap? Or asked differently: Has the NP-hardness consequences for the runtime of algorithms on practical data sets? The black circle contains the inputs used in the NP-hardness proof.

An optimization problem B is proved to be NP-hard with the technique of Turing reductions. They work as follows: We construct a mapping — the Turing reduction — that maps each input of an NP-hard problem A to an input of B. Note, that in most cases not all inputs of B are used by this mapping, the set of used inputs is displayed in Figure 1 as the black circle. The reduction needs to have the property that we can construct from the solution of B in polynomial time a correct solution for A. If both problems are decision problems then they only answer YES or NO. Then the reduction needs to have the property that

$$A \text{ answers YES } \Leftrightarrow B \text{ answers YES}$$

There are two directions "$\Leftarrow$" and "$\Rightarrow$", that are considered in all NP-hardness proofs. Further, the mapping needs to have the property that it can be computed in polynomial time.

If a problem A is NP-hard, then there is no algorithm working in polynomial time under the assumption P $\neq$ NP. This can be seen as follows: Under the assumption that B has a polynomial time algorithm we can construct with the Turing reduction an algorithm for A that also works in polynomial time, in the following way: We take the input of A, run the Turing reduction in polynomial time and obtain an input for B. We run the polynomial time algorithm of B and obtain a solution, that leads to a solution for A. Therefore, we can compute a solution for A in polynomial time. So we know that one of the two assumptions must be false. But P $\neq$ NP seems to be the more plausible one.

3 An experiment on a bad data set

In this section, we describe a small experiment with the LTS estimator. We design a data set such that the related search space contains one global optimum and one local optimum (not counting the local optima that are caused by noise). The data set is displayed in Figure 2 and can be extended to d dimensions. We set the parameters of the model as follows:

$$\begin{array}{lll} 550 \text{ points:} & y = \varepsilon_{\text{noise}} - 1 \cdot x_1 + 1.2 & \text{with } x_1 \sim \text{uniform}(0, 1.2) \\ 450 \text{ points:} & y = \varepsilon_{\text{noise}} - \frac{1}{10} \cdot x_1 + 1.5 & \text{with } x_1 \sim \text{uniform}(1.2, 2) \end{array}$$

The variable $\varepsilon_{\text{noise}} \sim N(0, 0.001^2)$ and the other variables are $x_2 \sim \ldots x_d \sim \text{uniform}(-1, 1)$. We expect that the LTS estimator outputs a hyperplane with $\beta_1 = -1$ and $\beta_2 = 0, \ldots, \beta_d = 0, \beta_{d+1} = 1.2$ since the majority of the points lie close to this hyperplane.

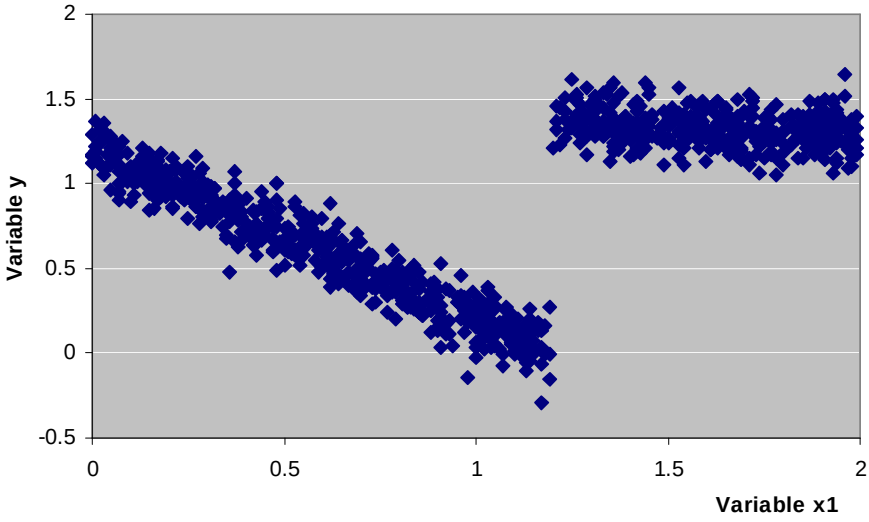

Figure 2: The bad data set, 550 points on the left and 450 points on the right. In the figure, a standard deviation of 0.1 is displayed, but in the experiments we used a value of 0.001.

We use the robust estimator

`ltsReg()` from the `rrcov`-package

of the R-software [11, 13] and set the parameter "alpha" = 0.51, to get a high breakdown point. The LTS estimator is defined as follows:

Definition 3.1 (LTS problem) *Given n points and an integer h ($0.5n \leq h < 1$), find a hyperplane with parameter vector β, such that the scale $\frac{1}{h}\sum_{i=1..h} r_{(i)}(\beta)$ is minimal.*

We run the ltsReg 10000 times for different dimensions. (In 18 dimensions we run the algorithm 200000 times). The solutions found by the algorithm are displayed in Figure 3 for the 12-dimensional case. As we expected, the solutions with $\beta_1 \approx -1, \beta_2 \approx 0, \ldots, \beta_d \approx 0, \beta_{d+1} \approx 1.2$ have a small scale of ≈ 0.0025. In these cases, the correct optimum was found. The second, local optimum contains solutions with a scale of ≈ 0.15 and its slope is $\beta_1 \approx 1.05$ and its y-axis is $\beta_{d+1} \approx 0.17$.

As we are interested in the number of times the algorithm finds the correct optimum, we count the number of times the estimate of the scale is smaller

than 0.01. The count is divided by 10000 and we obtain the estimated probability for the event that the algorithm has found a good solution. The probability is displayed in Figure 4. The error bars indicate the 1% quantile, e.g., if the true probability was at the end of the error bar then the measured value would occur with probability less than 1%. The bounds were estimated with Chernoff bounds.

For the 18 dimensional data set, the algorithm finds a global optimum with a probability of 0.00006. The success probability is plotted logarithmical on the vertical axis in Figure 4. One gets the impression that the success probability decreases exponentially as the dimension rises. Therefore, the algorithm is not practical for higher dimensions.

4 The design of robust estimators

Search heuristics are often used to compute the mentioned estimators. In Figure 5, the components of a search heuristic are displayed. A search heuristic maintains a set of individuals. Each individual represents a solution, e.g., a hyperplane for the LTS problem. It uses search operators to create new individuals and it explores the search space by asking questions about individuals to a black box. The black box gives fitness values for each individual, e.g., the fitness value of the LTS problem is the sum of the h smallest residuals (divided by h). In this way, the black box guides the search heuristic and it hopefully finds a minimal solution, but this is not guaranteed.

The LTS estimate is a solution of the LTS problem with minimal fitness. If one applies a search heuristic to a robust estimator problem, the difficulty is that with a certain probability a suboptimal solution is found, as we have demonstrated in Section 3. Such a solution could be affected by outliers. Hence, the robust estimators are not implemented appropriately and the approach lacks reliability.

To add reliability, we provide the black box with the ability to output whether a solution is optimal (minimal for LTS) or not. Can this work for robust estimators like LTS? The answer is NO due to the following reasons: To decide whether a solution is minimal, means, to decide whether the solution is smaller than or equal to all other possible solutions. Such decision problems contain a universal quantifier "$\forall$" and belong to the class of co-NP.

Definition 4.1 (co-LTS Problem) *Given n points, an integer h ($0.5n \leq h < 1$) and a bound B, decide whether $\frac{1}{h}\sum_{i=1..h} r_{(i)}(\beta) \geq B$ for all possible hyperplanes with parameter vector β.*

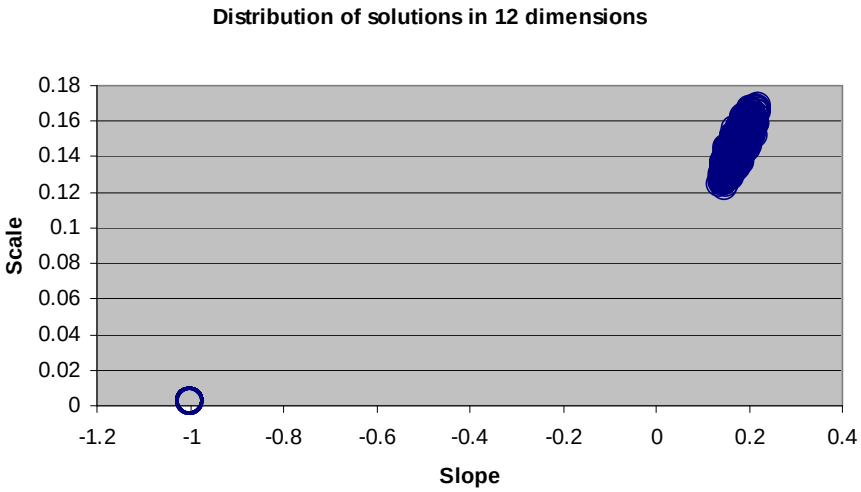

Figure 3: The solutions of the 12-dimensional data sets are displayed. Most of the solutions are located in the right point cloud, which is only a local optimum. Only a few solutions are located in the left cloud, which is the global optimum.

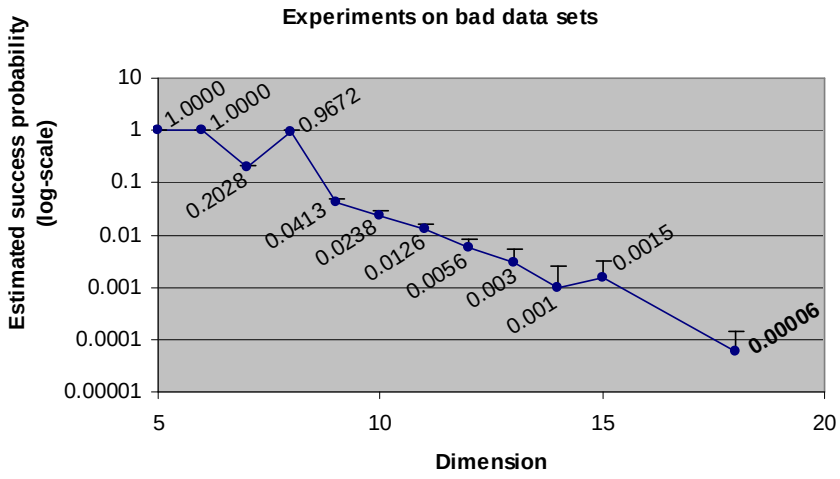

Figure 4: The logarithm of the success probability of the ltsReg() function from the rrcov-package is displayed. It was run 10000 times (200000 times in 18 dimensions) on difficult data sets in various dimensions.

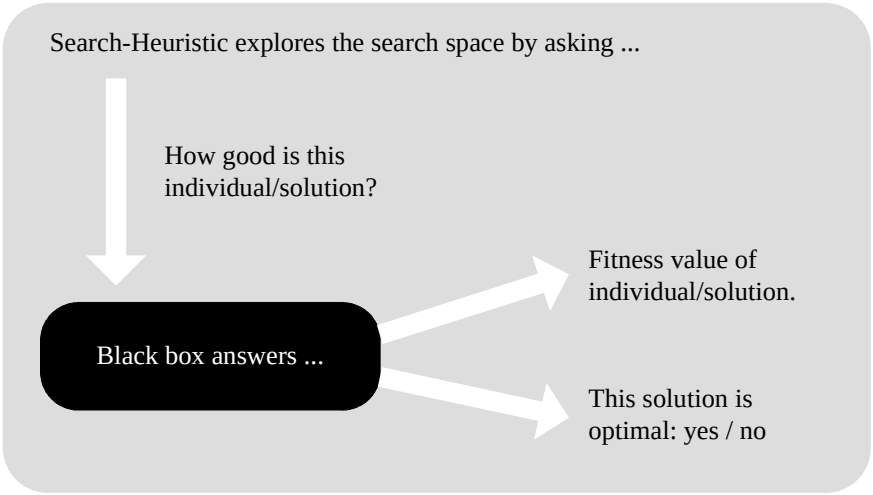

Figure 5: The components of a search heuristic.

We show in Section 5 that the LTS problem is NP-hard. Hence, the co-LTS problem is co-NP-hard. With the assumption P $\neq$ NP, it follows that there is no black box for co-LTS that works in the described way in polynomial time. If one wants to use search heuristics in robust statistics, the LTS estimator is not suitable. The other robust estimators listed in Section 1 are not suitable either.

Therefore, we need new estimators that are robust and that can be computed reliably by search heuristics. The approach is to design a black box that decides in polynomial time whether an individual is an optimal solution or not. It is essential to drop the universal quantifier " "$\forall$" and replace it by an existential quantifier "$\exists$". A suitable definition of a computationally feasible robust estimator should have the following structure:

Design Pattern: Does there exist a bit string J of polynomial length (the individual), such that a polynomial-time algorithm (the black box), running on J and the data set, computes YES? (This is the characterization of the problem class NP.)

To use the design pattern, one has to specify the semantics of the bit string as well as the black box. To give an example of such an approach, consider the usual decomposition of data into signal and noise:

$$\text{data} = \text{signal} + \text{noise}.$$

The search heuristic proposes an individual – a signal – and the black box has to decide whether the remaining part of the data looks like noise. If

so, the individual describes a signal that we wanted to find. A promising approach in two dimensions is the use of the multiresolution criteria, see e.g. [2]. It is able detect Gaussian noise. The question is: Can this approach be generalized to d dimensions such that the multiresolution criteria problem remains computable in polynomial time?

5 NP-hardness of robust estimators

Many robust estimators can be used to find subsets of points, such that all points of the subset are located on a common hyperplane. Therefore, we define the following problem:

Definition 5.1 (Degenerate Point Subset Problem (c-DPS)) *Given n points in d dimensions and a fixed number c with $0 < c \leq 0.5$, are there parameters $\beta_1, \ldots, \beta_d$ such that $h := \lceil (1-c) \cdot n \rceil$ points are located on the hyperplane $y = \sum_{i=1..d-1} \beta_i x_i + \beta_d$?*

This definition is related to the exact-fit-property of a robust estimator in statistics. We use the c-DPS problem to prove the NP-hardness of some robust estimators. The proof of the NP-hardness of c-DPS is based on the NP-complete Vertex Cover problem, see e.g. [3, 14]. It is defined as follows:

Definition 5.2 (Vertex Cover Problem) *Given a graph G=(V,E) and a number $k \in \mathbb{N}$, is there a subset $V' \subseteq V$ of size $|V'| = k$, such that for all edges $e \in E$, at least one vertex of e is contained in V'?*

The Vertex Cover problem is illustrated in Figure 6. In the following theorems, we will fix the value c. What does this mean? Well, one could prove, that c-DPS is NP-hard only for a certain value of c. But then there is no statement about whether this problem is NP-hard for other values of c. Therefore, we will carry out the proofs for all values of c fixed in advance to $0 < c \leq 0.5$. We start with the following theorem:

Theorem 5.3 *The c-DPS problem is NP-hard for all fixed c with $0 < c \leq 0.5$.*

Proof. In the following we construct a Turing reduction where we use the c-DPS problem to find an answer to the Vertex Cover problem. Therefore, we have to transform the given graph G and the parameter k into a point set.

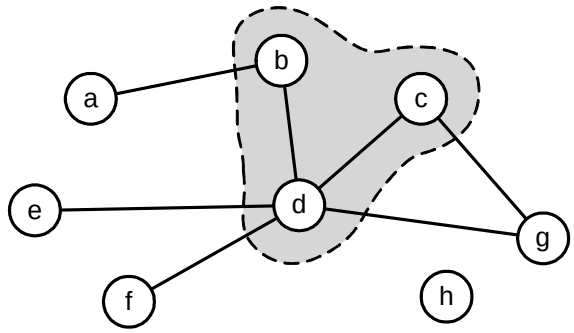

Figure 6: An undirected graph with 8 vertices and 7 edges. A minimal vertex cover $\{b, c, d\}$ is marked by the grey area. Either an edge is crossing the border of the area or an edge is completely contained in the area. Therefore, the vertices b, c and d cover all edges of the graph.

If the c-DPS problem outputs YES, then there is a hyperplane

$$y = \sum_{i=1..d-1} \beta_i x_i + \beta_d,$$

that contains h points. From that, we can obtain the vertex cover of size k as follows: The information which vertices are contained in the subset V', is coded into the parameters $\beta_1, \ldots, \beta_d$ of the hyperplane. A value of $\beta_i = 2$ for $i = 1, \ldots, |V|$ means that the vertex i belongs to the vertex cover. A value of $\beta_i = 1$ means that the vertex does not belong to the vertex cover. We will later show that the construction ensures that only these two values of β_i are meaningful. We first give an informal description of the used points:

- v_i and $\overline{v_i}$:
  For each vertex i, we construct two points v_i and $\overline{v_i}$. Later in the proof, they are used to force β_i to take the values 1 or 2.

- e'_{ij} and e''_{ij}:
  For each edge (i, j), we construct these two points. Later they are used to check whether each edge of the graph is covered. If for the edge (i, j) both vertices i and j are not in the vertex cover, then both points e'_{ij} and e''_{ij} are not located on the hyperplane.

- K^* and $\overline{K^*}$:
  The point K^* ensures that $\sum_{i=1..|V|} \beta_i = |V| + k$. If K^* is located on the hyperplane then exactly k vertices belong to the vertex cover. For symmetry reasons, the point $\overline{K^*}$ is added to the point set.

- p^* and $\overline{p^*}$:
  The point p^* forces the parameter β_d of the hyperplane to 0. The point $\overline{p^*}$ is added due to symmetry.

- $p_1, \ldots, p_\zeta$:
  Without these points, the proof would only work for $c = 0.5$. By adding enough points, every value of $0 < c \leq 0.5$ is achievable.

The coordinates of the mentioned points are listed in the following table, the first coordinate of the points is y, the other coordinates are named $x_1, \ldots, x_{d-1}$:

		y	$x_1, \ldots, x_i, \ldots, x_j, \ldots, x_{\lvert V\rvert}$	$x_{\lvert V\rvert+1}, \ldots, x_{d-1}$	
$\overline{v_i}$	$= ($	1	$0, \ldots, 0, 1, 0, \ldots, 0$	$0, \ldots, 0$	$)$
v_i	$= ($	2	$0, \ldots, 0, 1, 0, \ldots, 0$	$0, \ldots, 0$	$)$
e'_{ij}	$= ($	3	$0, \ldots, 0, 1, 0, \ldots, 0, 1, 0, \ldots, 0$	$0, \ldots, 0$	$)$
e''_{ij}	$= ($	4	$0, \ldots, 0, 1, 0, \ldots, 0, 1, 0, \ldots, 0$	$0, \ldots, 0$	$)$
K^*	$= ($	$\lvert V\rvert + k$	$1, \ldots, 1$	$0, \ldots, 0$	$)$
$\overline{K^*}$	$= ($	$\lvert V\rvert + k - 1$	$1, \ldots, 1$	$0, \ldots, 0$	$)$
p^*	$= ($	0	$0, \ldots, 0$	$0, \ldots, 0$	$)$
$\overline{p^*}$	$= ($	$1,$	$0, \ldots, 0$	$0, \ldots, 0$	$)$
p_ℓ	$= ($	1	$0, \ldots, 0$	$0, \ldots, 0, 1, 0, \ldots, 0$	$)$

Each point consists of three blocks, the second block has a length of $|V|$, the third one has a length of ζ. The value of ζ and other parameters of c-DPS, the dimension, the number of points and the parameter h are listed below:

$$\begin{aligned} \zeta &= 2 \cdot \left\lceil \tfrac{1-2c}{2c} \cdot (|V| + |E| + 2) \right\rceil \\ d &= 2 + |V| + \zeta \\ n &= 2 \cdot (|V| + |E| + 2) + \zeta \\ h &= 1 \cdot (|V| + |E| + 2) + \zeta \end{aligned}$$

For each vertex i in the graph, we construct the points v_i and $\overline{v_i}$, the "1" is placed at the i-th position within the second block. For each edge $(i, j) \in E$, we construct the points e'_{ij} and e''_{ij}, the "1"s in the second block are placed at the positions i and j. We construct the points p_ℓ for $\ell = 1, \ldots, \zeta$ to adjust the point set to each fixed value of c. In addition, we construct the points K^*, $\overline{K^*}$, p^*, and $\overline{p^*}$. Note that the point set can be computed in polynomial time as c is a fixed value.

To show that this Turing reduction is correct, we have to prove two claims:

Claim 5.4 *If there is a vertex cover of size k, then there is a hyperplane containing h points.*

Proof. Consider a vertex cover V' of size k. The parameters of the hyperplane are chosen as follows: If $i \in V'$ then we set $\beta_i = 2$ and $\beta_i = 1$ otherwise. Furthermore, set $\beta_{|V|+1}, \ldots, \beta_{d-1} = 1$ and $\beta_d = 0$.

It is easy to see that either v_i or $\overline{v_i}$ for all $i = 1, \ldots, |V|$ is located on the hyperplane. This gives $|V|$ points. The subset V' is a vertex cover, hence, each edge is adjacent to at least one vertex in V'. Therefore, it holds that for each point tuple (e'_{ij}, e''_{ij}) the parameter $\beta_i = 2$ or $\beta_j = 2$ and, therefore, either e'_{ij} or e''_{ij} is located on the hyperplane. This gives $|V| + |E|$ points. The point K^* is on the hyperplane, as k values of $\beta_i = 2$ and the remaining $|V| - k$ ones are equal to 1. The choice of $\beta_d = 0$ ensures that p^* is on the hyperplane. This gives $|V| + |E| + 2$ points. As $\beta_{|V|+1}, \ldots, \beta_{d-2} = 1$, all points $p_1, \ldots, p_\zeta$ are located on the hyperplane. This gives $|V| + |E| + 2 + \zeta$ points.

Therefore, this hyperplane contains h points. □

Claim 5.5 *If the size of all vertex covers is larger than k, then all hyperplanes contain less than h points.*

Proof. The vertices v_i and $\overline{v_i}$ cannot be on the same hyperplane, since vertical hyperplanes are not possible with the used hyperplane representation. The same holds for the tuples (e'_{ij}, e''_{ij}), $(p^*, \overline{p^*})$ and $(K^*, \overline{K^*})$. Therefore, from the points mentioned above, at most $|V| + |E| + 2$ points can be located on the same hyperplane. Taking the points $p_1, \ldots, p_\zeta$ into account, we obtain that at most $h = |V| + |E| + 2 + \zeta$ points can be located on the same hyperplane. The following proof is based on the fact, that we are not able to reach the required number of points h, if one point is missing.

At first, we consider hyperplanes with parameters $\beta_i = 1$ or $\beta_i = 2$ for $i = 1, \ldots, |V|$ and $\beta_{|V|+1}, \ldots, \beta_{d-1} = 1$ and $\beta_d = 0$. We argue that these hyperplanes contain less than h points, To enumerate all considered hyperplanes, we construct the following mapping: Each subset of the vertices corresponds to a parameter-vector β: A vertex v_i is contained in the subset V' if and only if $\beta_i = 2$. Otherwise $\beta_i = 1$.

From the assumption of the claim we know, that each subset of $\leq k$ vertices is not a vertex cover. Therefore, there is an edge (i, j) that is not covered and the vertices v_i and v_j are not in the considered subset. This implies that $\beta_i = 1$ and $\beta_j = 1$ and neither the point e'_{ij} nor e''_{ij} is located on the hyperplane.

Hence, the number of points is at most $|V| + |E\backslash\{(i,j)\}| + 2 + \zeta < h$ and, therefore, no hyperplane with parameters considered above contains h points. For subsets with more than k vertices, it follows that $\sum_{i=1...|V|} \beta_i > |V| + k$ and, therefore, the points K^* and $\overline{K^*}$ are not located on the hyperplane.

Second, we inspect hyperplanes with "wrong" parameters, and we also argue that the parameters lead to hyperplanes that contain less than h points.

A hyperplane with $\beta_d \neq 0$ and $\beta_d \neq 1$ does not contain the points p^* and $\overline{p^*}$. For $\beta_d = 1$, we have to set $\beta_i = 0$ or 1 for all $i = 1...|V|$ to achieve that one of the points v_i or $\overline{v_i}$ is located on the hyperplane. But then the points K^* and $\overline{K^*}$ are not located on the hyperplane. Therefore, $\beta_d = 0$ is the only meaningful choice.

For a hyperplane with $\beta_d = 0$ and $\beta_i \neq 1$ and $\beta_i \neq 2$ for $i = 1, \ldots, |V|$ or $\beta_j \neq 1$ for $j = |V| + 1, \ldots, d - 1$ it can easily be checked that the points v_i and $\overline{v_i}$ or p_j are not located on the hyperplane. □

This completes the proof of the NP-hardness of c-DPS. □

Corollary 5.6 *For $c = 0$, the point set used in the proof of the NP-hardness of c-DPS, is organized in tuples $(P_i\ ,\ \overline{P_i})$. The residuals of the points P_i and $\overline{P_i}$ have the property that for all hyperplanes L*

$$|r_i(L) - \overline{r_i(L)}| = 1 \quad .$$

5.1 Subset estimators

In the statistical literature, $h = \lceil (1-\varepsilon) \cdot n \rceil + \lceil \varepsilon \cdot (d+1) \rceil$ is used. All proofs in this paper hold with this alternative definition. But we do not consider both cases and we focus on the simpler case $h = \lceil (1-\varepsilon) \cdot n \rceil$, which is also used in the literature for the CM estimator.

Definition 5.7 (subset estimator problem) *Given a set $\mathcal{P}$ of n points in d dimensional space and a fixed parameter ε with $0 < \varepsilon \leq 0.5$, the subset estimator problem is defined as follows: Find a subset $S \subseteq \mathcal{P}$ of size $h = \lceil (1-\varepsilon) \cdot n \rceil$, that minimizes a function $f_{subset} : \mathbb{P}(\mathcal{P}) \longrightarrow \mathbb{R}^+$. The function f_{subset} has the property that $f_{subset}(S) = 0$ if the points of S are located on a hyperplane and $f_{subset}(S) > 0$ otherwise.*

Theorem 5.8 *For every fixed ε, the computation of the subset estimator is NP-hard.*

Proof. We prove this theorem by a reduction from the c-DPS problem to the subset estimator problem with $c = \varepsilon$. The input is directly passed to the subset estimator. If the subset estimator finds a subset S with $f_{\text{subset}}(S) = 0$, then there exists a hyperplane with h points on it and we output YES. Otherwise, the function value is larger than zero, a hyperplane does not exist, and we output NO. □

Definitions of the following estimators can be found in [8],[9],[11],[5]. Note that $r_{(i)}$ indicates the i-th residual in the sorted order of all residuals.

Definition 5.9 (ε-LXX estimator problems) *For a given point set of size n in d dimensions and a fixed parameter ε with $0 < \varepsilon \leq 0.5$, the ε-LXX estimator problems are defined as follows. Let $h = \lceil (1-\varepsilon) \cdot n \rceil$. Find a hyperplane that is described by the parameter vector β, such that*

$$\begin{array}{rrcl} \textit{for} & \textit{LMS:} & f_{LMS} & = r_{(\lceil n/2 \rceil + \lceil (d+1)/2 \rceil)}(\beta)^2 \\ \textit{for} & \varepsilon\textit{-LQS:} & f_{\varepsilon\text{-}LQS} & = r_{(h)}(\beta)^2 \\ \textit{for} & \varepsilon\textit{-LTS:} & f_{\varepsilon\text{-}LTS} & = \frac{1}{h} \sum_{k=1}^{h} r_{(k)}(\beta)^2 \\ \textit{for} & \varepsilon\textit{-LTA:} & f_{\varepsilon\text{-}LTA} & = \frac{1}{h} \sum_{k=1}^{h} |r_{(k)}(\beta)| \end{array}$$

is minimized.

We discuss the MCD [10] and MVE [15] estimator problems only briefly to avoid technical details.

Definition 5.10 (ε-MCD and ε-MVE estimator problem) *For a given point set of size n in d dimensions and a fixed parameter $0 < \varepsilon \leq 0.5$, the problems are defined as follows. Find a subset of the points of size $\lceil (1-\varepsilon) \cdot n \rceil$, such that*

- *for ε-MCD: the determinant of the covariance matrix of this subset is minimized.*
- *for ε-MVE: the volume of the ellipsoid covering this subset is minimized.*

Theorem 5.11 *For every fixed ε ($0 < \varepsilon \leq 0.5$), the computation of the following estimators is NP-hard: LMS, ε-LQS, ε-LTS, ε-LTA, ε-MCD, and ε-MVE.*

Proof. The mentioned estimators match the definition of the subset estimator:

- **LMS, LQS, LTS and LTA:**
  If a point is located on a hyperplane, its residual is zero. If there are h points that are located on a common hyperplane, then their residuals are all zero and the fitness value of the LXX estimators become zero, as only the smallest h residuals are involved in the objective function. Otherwise, the fitness value is larger than zero.

- **MCD and MVE:** Consider a subset of the points of size h. There is an ellipsoid with minimal volume that contains these points. If these points are located on a hyperplane, the ellipsoid is flat and therefore one eigenvalue of this ellipsoid is zero and therefore the determinant and the volume are zero. Otherwise, the functional values are larger than zero.

Therefore, the computation of the mentioned estimators is NP-hard. □

5.2 CM estimator

Let ρ be a symmetric and continuously differentiable function, which is bounded and nondecreasing on $[0, \infty[$ and $\rho(0) = 0$. For a given set of n points in $\mathbb{R}^d$ and a $\beta \in \mathbb{R}^d$, let $r_i(\beta)$ be the residual of the i-th point.

Definition 5.12 (ε-CM problem [1]) *For a given point set and a fixed parameter $0 < \varepsilon < 1$, the CM problem is defined as follows: Find a scale parameter σ and a hyperplane that is described by the parameter vector β, such that*

$$\frac{1}{n} \sum_{i=1,\ldots,n} \rho\left(\frac{r_i(\beta)}{\sigma}\right) \leq \varepsilon \cdot \rho(\infty)$$

and such that the function

$$f_{CM}(\beta, \sigma) = \log \sigma + \frac{1}{n} \sum_{i=1,\ldots,n} \rho\left(\frac{r_i(\beta)}{\sigma}\right)$$

is minimized.

Theorem 5.13 *For every fixed ε ($0 < \varepsilon \leq 0.5$), the ε-CM problem is NP-hard.*

Proof. We solve the ε-DPS problem using the ε-CM problem as a subroutine. Therefore, we pass the point set to the ε-CM problem. The estimator outputs

a vector β and a value σ. If $\sigma = 0$, then we output YES. If $\sigma > 0$, then we output NO. We show now that the given answer is correct.

If there is a degenerate point set of size $\lceil (1-\varepsilon) \cdot n \rceil$, then there exists a hyperplane H such that the $\lceil (1-\varepsilon) \cdot n \rceil$ residuals are zero even for $\sigma \to 0$. Therefore, only $\lfloor \varepsilon \cdot n \rfloor$ residuals r_i are larger than 0 and it follows that the constraint of CM is smaller than $\varepsilon \cdot \rho(\infty)$ even if $\sigma = 0$. For this hyperplane H, it is possible to choose $\sigma = 0$, with the result that the objective function $f_{\mathrm{CM}}(\beta, \sigma)$ goes to $-\infty$ for $\sigma \to 0$.

If there is no degenerate point set of this size, then for all hyperplanes H, there are more than $\lfloor \varepsilon \cdot n \rfloor$ residuals with r_i larger than 0. The constraint ensures that $\sigma > 0$, since $\sigma = 0$ forces the constraint to become larger than $\varepsilon \cdot \rho(\infty)$. □

5.3 Projection Depth and Stahel-Donoho estimator

Definition 5.14 (DULS) *A pair (μ, σ) of estimators belongs to the class of degenerate univariate location and scale estimators (DULS), if there is a number $c^* \in \mathbb{R}$ with $0 < c^* \leq 0.5$, such that the following properties are true:*

- *If S is a sequence of numbers of length n, where $h^* = \lceil (1-c^*) \cdot n \rceil$ numbers have the same value w, then $\mu(S) = w$ and $\sigma(S) = 0$.*
- *If S is a sequence of numbers of length n, where less than h^* numbers have the same value, then $\sigma(S) \neq 0$.*

Popular examples for univariate estimators of location μ and scale σ are the median and the Median Absolute Deviation (MAD):

$$\mu(p_1, \ldots, p_n) = \operatorname*{med}_i(p_i) \text{ with } p_i \in \mathbb{R}$$
$$\sigma(p_1, \ldots, p_n) = \operatorname*{med}_i |p_i - \mu(p_1, \ldots, p_n)|.$$

It is easy to see that (med, MAD) $\in$ DULS, whereas (mean, variance) are not in DULS.

We now define two robust estimators that use DULS estimators. Consider the projection $\beta^\top x$ of a point x onto a vector β. Indeed, this is an orthogonal residual of a point to a hyperplane with the normal vector β. We define the following abbreviation: $\beta^\top X := (\beta^\top x_1, \ldots, \beta^\top x_n)$. The following two definitions can be found in [16]:

Definition 5.15 (Outlyingness) *Given n points in d dimensions, the outlyingness of a data point i is defined as*

$$OL_i(\sigma, \mu) = \max_{|\beta|=1} \frac{\left|\beta^\top x_i - \mu(\beta^\top X)\right|}{\sigma(\beta^\top X)},$$

where μ and σ are estimators of location and scale. If the numerator and the denominator of OL_i are both equal to zero, then we define $OL_i = 0$. If only the denominator is equal to zero, then we define $OL_i = \infty$.

Definition 5.16 ((μ, σ)-Projection Depth problem ((μ, σ)-PD)) *Given n points in d dimensions, for each point i the following value is called the projection depth:*

$$PD_i(\sigma, \mu) = \frac{1}{1 + OL_i(\sigma, \mu)} = \min_{\beta} \frac{\sigma(\beta^\top X)}{\sigma(\beta^\top X) + |\beta^\top x_i - \mu(\beta^\top X)|}$$

Theorem 5.17 *The (μ, σ)-Projection Depth problem with fixed $(\mu, \sigma) \in DULS$ is NP-hard.*

Proof. As the estimators μ and σ are fixed, there is a constant c^*, such that (σ, μ) fulfill the properties of Definition 5.14. Let $h^* = \lceil c^* \cdot n \rceil$. We use the Projection Depth problem to compute a solution of the c^*-DPS problem in the following way: We check in polynomial time whether all points are located on the same hyperplane. If this is true, we output YES. If $h^* = n$, we output NO. Otherwise, we pass the point set to the Projection Depth Problem. It reports the projection depth of each point. If there is at least one point with projection depth zero, we output YES, otherwise NO.

The correctness can be seen as follows: In the first case, we assume that there is a degenerate point set of size $h^* < n$. Consider the normal vector β of a hyperplane that contains all these h^* points. Then there are h^* residuals with the same value w. Due to the definition of DULS we derive that $\mu(\beta^\top X) = w$ and $\sigma(\beta^\top X) = 0$. Therefore, there is a point x_i with $\beta^\top x_i = w$ and it follows that $PD_i = 0$.

The case that all points are located on the same hyperplane is handled separately. Therefore, we can assume that there is at least one point with projection $\beta^\top x_i \neq w$. As we have to minimize the projection depth, we get that $\mathrm{PD}_i = \sigma/(\sigma + \beta^\top x_i) = 0$ as $\sigma = 0$.

In the second case, we assume that all degenerate point sets have a size less than h^*. Then, for all β, there are less than h^* projections $\beta^\top x_i$ with the same value w . From Definition 5.14, it follows that $\sigma(\beta^\top X) > 0$ for all β and therefore the Projection Depth of all points is larger than zero. □

Definition 5.18 ((μ,σ)-Stahel-Donoho problem ((μ,σ)-SD) [4]) *Given n points $x_1,\ldots,x_n$ in d dimensions, compute the weighted mean T and the covariance matrix V of the points with weights $PD_i(\mu,\sigma)$ as follows:*

$$w = \sum_{i=1}^{n} PD_i(\mu,\sigma)$$

$$T = \frac{1}{w} \cdot \sum_{i=1}^{n} PD_i(\mu,\sigma) \cdot x_i$$

$$V = \frac{1}{w} \cdot \sum_{i=1}^{n} PD_i(\mu,\sigma) \cdot (x_i - T) \cdot (x_i - T)^\top.$$

If all $PD_i = 0$, we define as output $T = (\infty,\ldots,\infty)^\top$. Analogously for V.

Theorem 5.19 *The (μ,σ)-Stahel-Donoho estimator problem with $(\mu,\sigma) \in$ DULS is NP-hard.*

Proof. The estimators μ and σ are fixed. There is a constant c^*, such that (σ,μ) fulfills the properties of Definition 5.14. We use the SD Problem to compute a solution of the c^*-DPS problem in the following way: We check in polynomial time whether all points are located on the same hyperplane. If this is true, we output YES. If $h^* = n$, we output NO. Otherwise, we call SD and obtain a tuple (T,V). If $T = (\infty,\ldots,\infty)$ we output YES. Otherwise, we call SD n times with a modified input. In the i-th call we use the data set $(x_1,\ldots,x_n)$ without x_i and get the output (T_i,V_i). If there is an i with $x_i \neq T$ and $T = T_i$ we output YES. If there is an i with $x_i = T$ and $V = V_i$ we output YES. Otherwise we output NO.

That the c^*-DPS problem is solved correctly can be seen as follows: We focus on the case, that not all n points are on a common hyperplane and $h^* < n$, as we have handled this cases separately.

First, consider that there is a hyperplane such that $h^* = \lceil (1-c^*) \cdot n \rceil$ points are located on it. We consider this common hyperplane and we know from the proof of Theorem 5.17 that there is at least one point x_i with $PD_i = 0$. If $x_i \neq T$ then the deletion of the point does not alter w or T, as $PD_i = 0$, and the answer is correct. If $x_i = T$, then $(x_i - T) = 0$ and, therefore, the deletion of the point does not alter w or V and also the answer is correct.

Second, let us assume that less than h^* points are located on the same hyperplane. Then we know from the proof of Theorem 5.17 that all $PD_i \neq 0$. If $x_i \neq T$, the deletion of x_i will alter T, as $PD_i \neq 0$. In the case of $x_i = T$, the deletion of x_i will alter w and therefore will alter V. The output is NO and, therefore, the answer is correct. □

Acknowledgement

Many thanks to Karen Schettlinger for programming the R-script used in the experiments. I want to thank Roland Fried, Thomas Hofmeister and Ingo Wegener for helpful comments on the paper. The financial support of the Deutsche Forschungsgemeinschaft (SFB 475, "Reduction of complexity in multivariate data structures") is gratefully acknowledged.

References

[1] O. Arslan, O. Edlund, and H. Ekblom. Algorithms to compute CM- and S-estimates for regression. *Metrika*, 55:37–51, 2002.

[2] P.L. Davies and A. Kovac. Local extremes, runs, strings and multiresolution. *The Annals of Statistics*, 29:1–65, 2001.

[3] M.R. Garey and D.S. Johnson. *Computers and Intractability — A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, San Francisco, 1979.

[4] D. Gervini. The influence function of the stahel-donoho estimator of multivariate location and scatter. *Statistics & Probability Letters*, 60:425–435, 2002.

[5] D.M. Hawkins and D. Olive. Applications and algorithms for least trimmed sum of absolute deviations regression. *Computational Statistics and Data Analysis*, 32:119–134, 1999.

[6] P. Huber. *Robust Statistics*. John Wiley & Sons, 1981.

[7] Clay Mathematics Institute. *http://www.claymath.org/millennium*.

[8] P.J. Rousseeuw. Least median of squares regression. *Journal of the American Statistical Association*, 79:871–880, 1984.

[9] P.J. Rousseeuw and A.M. Leroy. *Robust Regression and Outlier Detection*. Wiley, 1987.

[10] P.J. Rousseeuw and K. Van Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41:212–223, 1999.

[11] P.J. Rousseeuw and K. Van Driessen. Computing lts regression for large data sets. *Estadistica*, 54:163–190, 2002.

[12] Alan Turing. On computable numbers, with an application to the entscheidungsproblem. 1936.

[13] R-Project Website. *http://cran.r-project.org/doc/packages/rrcov.pdf*.

[14] I. Wegener. *Complexity Theory: Exploring the Limits of Efficient Algorithms*. Springer, 2005.

[15] D. Woodruff and D. Rocke. Computable robust estimation of multivariate location and shape in high dimension using compound estimators. *Journal of the American Statistical Association*, 89:888–896, 1994.

[16] Y. Zoo. Projection-based depth functions and associated medians. *The Annals of Statistics*, 31:1460–1490, 2003.