International Journal of Advanced Statistics and Probability, 3 (1) (2015) 53-63
www.sciencepubco.com/index.php/IJASP
©Science Publishing Corporation
doi: 10.14419/ijasp.v3i1.4439
Research paper

Fast online detection of outliers using least-trimmed squares regression with non-dominated sorting based initial subsets

Mehmet Hakan Satman

Istanbul University, Department of Econometrics, Turkey
Corresponding author E-mail: mhsatman@istanbul.edu.tr

Abstract

In this paper, a new algorithm is devised for calculating the Least Trimmed of Squares (LTS) estimator. The algorithm consists of two steps. In the first step, the non-dominated sorting algorithm is applied on the design matrix of regression data for selecting a clean subset of observations. In the second step, *C-step*s are iterated to adjust the LTS estimators. The algorithm is fast and precise for small sample sizes, however, the sorting algorithm is computationally inefficient in large datasets. A fast update mechanism can be used in online data with a linear increase in computation time. Some properties of the sorting algorithm are also investigated under some transformations. Results of applying the algorithm on some well-known datasets and Monte Carlo simulations show that the proposed algorithm is suitable to use in many cases when the computation time is the major objective and a moderate level of precision is enough.

Keywords: *Data streams; Initial subset selection; Non-dominated sorting; Outlier detection; Robust regression*

1. Introduction

Outliers in regression analysis are aberrant observations that do not fit the unknown regression surface. Since the regression coefficients are estimated using the actual data, outliers can dramatically change the partial coefficients and the intercept. This may result a big difference between the unknown parameters and their estimated counterparts, that is, an outlier may be located near the regression surface whereas a clean observation may get a high residual as it gets distant from the regression surface. The former situation is called *masking* which is amount or probability of labelling a true outlier as a clean observation whereas the latter is called *swamping* which is amount or probability of labelling a clean observation as an outlier [4].

Main reason underlying of an observation being an outlier is not only having considerably large or small values. An observation may not be well fitted by unknown regression surface either in X-space or y dimension. The former is called *leverage* which is a higher impact on estimated coefficients whereas the latter is called *vertical* or *regression* outlier [26]. If an outlier is distant from the both dimensions, it may lie on the regression surface and it is generally considered as a *good observation* as it reduces the standard errors of estimated coefficients.

Ordinary Least Squares (OLS) based outlier detection methods fail in most situations because of *masking* and

swamping effects [15, 29, 2, 6, 8, 22]. High breakdown estimators can be used instead. Suppose the model is

$$y = X\beta + \epsilon$$

where y is an n-vector of response variable, X is an $n \times p$ matrix of response variables, β is the $p-$vector of unknown parameters, ϵ is an $n-$vector of stochastic error term with zero mean and constant variance, n is the number of observations and p is the number of parameters. One of the high-breakdown estimators, *Least Trimmed Squares (LTS)*, is resistant up to $n-h$ of n outliers where h is at least 50% of data [19]. *LTS* estimator is defined as

$$\underset{\hat{\beta}}{Arg\min} \sum_{i=1}^{h} |\hat{e_i}^2|$$

where $|\hat{e_i}^2| = |\{y_i - X_{i,}\hat{\beta}\}^2|$ is the ith ordered squared residual, $X_{i,}$ is the ith row of matrix X, h is generally selected as $\lfloor \frac{n}{2} \rfloor + \lfloor \frac{p+1}{2} \rfloor$ [33] or $\lfloor \frac{n+p+1}{2} \rfloor$ [24] , $\lfloor a \rfloor$ is the nearest integer that is equal to or smaller than a.

[21] used an evolutionary algorithm, [3] used the forward search algorithm, [1] used a brunch and bound algorithm, [14] used mathematical programming for calculating the *LTS* estimators. Since optimizing the goal function requires high amount of computations when n and p are large, new algorithms are developed for calculating *LTS* estimators. [25] devised an algorithm for estimating *LTS* for large datasets. [27] extended this algorithm using genetic algorithms based initial subset selection.

In this paper, we devise a new algorithm for calculating the *LTS* estimators. A non-dominated sorting algorithm is applied on the independent variables to obtain an initial subset. This initial subset is then used to iterate *C-steps* as defined in [25]. Since the sorting algorithm is slow for large and static data, a fast update rule is developed for online data for re-selecting a new initial subset after receiving new cases. In section 2, we redefine and investigate some properties and usability of non-dominated sorting on initial subsets. In section 3, we introduce the proposed algorithm. In section 4, use of proposed algorithm in online data is discussed. In section 5, results of applying the proposed algorithm on some well-known datasets and simulated online data are presented. Finally, in section 6 we conclude.

2. Non-dominated Sorting and Its Properties

Non-dominated solutions [32], non-dominated sorting [30] and the related terms non-domination ranks and order are subjects of multi-objective optimization. The sorting algorithm is used to determine the solutions that are not comparable each other in the terms of superiority but superior to the remaining solutions. In a maximization problem with two objective functions, if elements of $s_1 = \{f(x), g(x)\}$ are not less than corresponding elements of $s_2 = \{f(y), g(y)\}$ and at least one element of s_1 is bigger than the corresponding element of s_2 then x dominates y, by symbols, $x \succ y$.

Definition: Suppose that $x_1 = (a_1, a_2, ..., a_p)$ and $x_2 = (b_1, b_2, ..., b_p)$ are points in p dimensional Cartesian space. If each b_i in x_2 is not smaller than a_i in x_1 and at least one b_i is bigger than a_i for $i = 1, 2, .., p$ then $x_2 \succ x_1$ (x_2 dominates x_1).

Definition: If a point x_i for $i = 1, 2, ..., n$ dominates $0 \leq c < n$ points then non-domination rank of x_i is c and the set of ordered points is $\mathcal{O}(x)$.

Non-domination ranks give an opportunity to sort points in higher dimensional spaces. Given the points $x_1 = (0, 0)$, $x_2 = (0, 2)$, $x_3 = (2, 0)$ and $x_4 = (3, 3)$; x_4 has the biggest values in all dimensions and it dominates the other points. x_2 and x_3 dominates x_1 and they share the same order. Finally, x_1 is dominated by all of the other points and the non-dominated order of these points is $x_4 \succ \{x_2, x_3\} \succ x_1$, or similarly, $\mathcal{O}(x) = \{x_1, \{x_2, x_3\}, x_4\}$ with ranks $R(x) = \{0, 1, 1, 3\}$.

In this paper, possibility of using the sorting algorithm in selecting observations in the middle of a dataset as a basic or elemental subset and in this section properties of the sorting algorithm are investigated. *Coordinatewise median* [7, 5], *comediance* [28], *multivariate median* [9, 20], *Minimum Covariance Determinant-MCD* [16] and *Minimum Volume Ellipsoid-MVE* [11], etc. estimators are used in robust regression literature for selecting a basic or clean subset of observations. Some of these algorithms are invariant under some transformations. These properties of non-dominated sorting algorithm are investigated in next subsections.

2.1. Location Invariance

Suppose that x_i and x_j are points in p-dimensional space for $i \neq j$. Choose a vector such that $k = [k_1 \quad k_2 \quad ... \quad k_p]^T$ where $k_u \in \mathbb{R}$ and $u = 1, 2, ..., p$. If $x_i \succ x_j$ then $x_i + k \succ x_j + k$, or similarly, $\mathcal{O}(x + k) = \mathcal{O}(x)$. This property

makes non-dominated sorting *location invariant* or invariant under shifting.

2.2. Scale Invariance

Suppose that x_i and x_j are points in p-dimensional space for $i \neq j$. Choose a vector such that $k = [k_1 \quad k_2 \quad ... \quad k_p]^T$ where each single element of k is positive. If $x_i \succ x_j$ than $x_i \circ k \succ x_j \circ k$, or similarly, $\mathcal{O}(x \circ k) = \mathcal{O}(x)$ where $a \circ b$ is *Hadamard product* [18] of vectors a and b. This property makes non-dominated sorting *scale invariant* for positive scalers. Note that when $k_1 = k_2 = ... = k_p$, $\mathcal{O}(x \circ k) = \mathcal{O}(k_1 x) = \mathcal{O}(x)$.

2.3. Rotation Invariance

Suppose that x_i and x_j are points in 2-dimensional space for $i \neq j$. Since the rotation matrix is $R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$, it can be found at least one θ that implies $Rx_i \succ Rx_j$ where $x_i \prec x_j$ due to negative scalers mentioned in Section 2.2.

Suppose that the points $z_i = (x_i, y_i)$ are generated using a function $y = f(x) = x$ for $i = 1, 2, ..., n$. Non-domination ranks are then $\mathcal{O}(z) = \{0, 1, 2, ..., n-1\}$ and the data shares the same multivariate order as the order calculated by each single dimension. After rotating the data using the angle $\theta = \pi/2$, all of the pairs take the rank of $n - 1$.

This property implies that the non-dominated sorting is not invariant under rotations.

3. Proposed Algorithm

The devised algorithm consists on two stages. In the first stage a clean subset of p or $p+1$ observations is determined using *non-dominated sorting* on the X-space. If the number of observations is odd, $p + 1$ observations are selected as basic subset whereas p observations are selected if the number of observations is even. In robust regression literature the terms basic subset or clean subset are also called *elemental subsets* and they include p observations [17] as well as $p + 1$ observations [31]. In the second stage, an *LTS* estimation is obtained using the basic subset returned from the first stage. The estimated coefficients are then used to iterate *C-steps*. A C-step can be defined as a function that takes a subset of observations as input and returns a subset of observations with size of h as output [25]. The pseudo-code of algorithm is given in Program 1.

Program 1 Pseudo-code of proposed algorithm

```
select_clean_subset <- function(xmat){
   calculate_non-domination_ranks(xmat)
   if (n-p) is even then
        return p observations in the middle of data
   else
        return (p+1) observations in the middle of data
   end if
}

estimate_lts <- function (model, data){
    select_clean_subset(xmat)
    estimate_lts()
    enlarge_clean_subset(h)
    while(iterations < max.number.of.csteps or
              a better solution is not obtained){
        iterate_c_steps()
    }
    return(results)
}
```

As mentioned in subsections 2.1, 2.2 and 2.3, selected subset of observations does not change by shifting or rescaling observations, however, rotations will change the order.

Table 1: Quantiles of computation times of updating ranks with online data

Quantiles	0%	5%	25%	50%	75%	95%	100%
Seconds	0.003	0.005	0.005	0.006	0.006	0.007	0.008

Table 2: Non-domination ranks of independent variables of Stackloss data

Observations	1	2	3	4	5	6	7	8	9	10	11
Ranks	15	12	14	10	7	9	15	15	7	0	5
Observations	12	13	14	15	16	17	18	19	20	21	
Ranks	0	1	8	1	0	0	1	2	3	10	

In first *C-step*, h subset of observations I_1 is determined using the initial p or $p+1$ subset I_0. In the following *C-steps*, a new set of observations I_i is generated. If subsets I_i and I_{i+1} are equal sets, algorithm terminates. In some cases, yielding a case that implies $I_i = I_{i+1}$ takes too many times so a **maximum number of C-Steps** can be defined as a stopping criterion.

Since the first stage algorithm requires the ranks of observations are calculated, obtaining an initial subset may be infeasible in large datasets. However, calculating the rank of $(N+1)th$ observation with updating remaining N observations increases the computation time linearly. This property of the algorithm is discussed in next section.

4. Fast Update for Online Detection of Outliers

Online detection of outliers mainly differs from the static analysis in terms of data source and large sample size. If the time interval of receiving Y_t after Y_{t-1} is less than the time required for calculating some statistics such as $\hat{T}(Y)$ then computationally more efficient algorithms are required.

It can be seen in [12] that sorting a data matrix with N rows and p columns requires N^2p computations because each single row should be compared with the other rows along the p dimensions. In the case of online data, non-domination ranks can be updated by performing $(N+1)^2p - N^2p = (2N+1)p$ additional computations after appending the new observation Y_{N+1} to dataset. Since $(2N+1)p$ is linear whereas $(N+1)^2p$ is quadratic, non-domination ranks are computationally expensive for large and static datasets and convenient to use for online data because of its fast update availability.

Procedures for calculating and updating non-domination ranks are written in *R* [23] and *C++* using the package *Rcpp* [13]. A random data matrix is created with 100000 rows and 10 columns. After calculating the non-domination ranks, a single random row is appended to data matrix in each iterations. Times consumed for sequential adding and updating are recorded for 20000 times. Some quantiles of recorded times in seconds are reported in Table 1 [1].

Table 1 shows that re-selecting basic subsets using non-domination ranks are not computationally expensive once a dataset is ordered. This fast update rule makes non-dominated sorting based initial subsets convenient for a computationally expensive robust estimator such as LTS.

5. Computational Results

5.1. Well-known Datasets

In this section OLS, Exact LTS, random LTS and non-dominated sorting based LTS (*Nds-Lts*) are applied on some well-known datasets which contain outliers. Results on Stack loss data [10], Hawkins-Bradu-Kass data [17] and Animals2 data [33] are reported in next subsections.

5.1.1. Stack Loss Data

Stack Loss data [10] has 3 independent variables, 5 outliers in 21 observations. Non-domination ranks that are calculated using the independent variables are reported in Table 2.

Observations $11, 5$ and 9 are selected as clean subset which are free from the outlying observations $1, 2, 3, 4$ and 21. In Table 3, results after applying the whole algorithm are given.

[1] Computations are performed in an Intel i8 computer with 8GBs Ram and Ubuntu Linux installed.

Table 3: Results of estimators on Stackloss data

	(Intercept)	Air.Flow	Water.Temp	Acid.Conc.	Time
OLS	-39.919	0.715	1.295	-0.152	1x
Nds-LTS	-35.210	0.746	0.337	-0.005	1.5x
Random-LTS	-37.154	0.750	0.321	0.019	4x
Exact-LTS	-35.810	0.750	0.333	0.000	8.5x

Table 4: Non-dominated ranks of independent variables of Hawkins,Bradu and Kass data

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-15	**61**	**61**	**64**	**65**	**66**	**63**	**64**	**61**	**63**	**61**	**71**	**71**	**71**	**71**	31
16-30	4	0	13	9	37	17	5	11	3	0	14	36	5	3	6
32-45	9	1	15	0	36	12	3	0	3	4	20	2	1	6	2
46-60	5	4	20	25	28	4	8	2	4	4	13	3	25	14	4
60-75	11	0	1	36	8	1	9	0	1	12	22	5	2	1	1

In Table 3 it is shown that the estimates of *Exact-LTS*, *Random-Lts* and *Nds-LTS* are similar, whereas, computation time required for *Nds-LTS* is considerably small and near to time required for OLS.

5.1.2. Hawkins, Bradu and Kass Data

Hawkins-Bradu-Kass data [17] includes 3 independent variables and a linear multiple regression model fits the data. It is known that the first 10 observations are bad leverage points whereas observations $11-14$ are good leverages. Non-domination ranks that are calculated on the independent variables are reported in Table 4.

It is shown in Table 4 that first 10 observations take considerable large ranks which are known to be outliers. Observations $65, 19$ and 31 are selected as the clean subset as they stay in the middle of the data by means of non-domination ranks. Table 5 summarizes the results of different algorithms.

It is shown in Table 5 that OLS estimators have different values when it is compared to LTS estimators, however, values of LTS estimators are different in some precision. Figure 1 reveals the effects of these differences by presenting residuals obtained by three LTS algorithms. In Figure 1 it is clear to see that the three LTS estimators yield large residuals and remarks the outliers in positions $1-10$. By considering the computation times reported in Table 5, it can be easily seen that, *Nds-LTS* has many benefits to use instead of others with losing some precision versus gaining time.

5.1.3. Animals2 Data

Animals2 data [33] contains *body* and *brain* weights of 65 species and a $\log-\log$ simple linear model fits the data. Since the dataset contains single independent variable, ordering and visualizing the data is easy to suspect outlyingness of observations and we show the results of this example for comparison issues only. Non-domination ranks obtained from the design matrix results that the observations $32, 33$ and 34 are selected as clean subset which belongs to species *Artic Fox*, *Water Opossum* and *Nine-banded Armadillo* which are free of outliers *Rhesus Monkey*, *Human*, *Triceratops*, *Dipliodocus* and *Brachiosaurus*. Results of estimators are reported in Table 6.

It is shown in Table 6 that the slope estimates are same for all LTS estimators whereas estimates of intercept parameter differ in some precision. All of the LTS estimators reports the true outliers. It is also shown in Table 6, computation time required for *Nds-LTS* is relatively small and it is preferable when the time is more important then the precision.

Table 5: Results of estimators on Hawkins-Bradu-Kass data

	(Intercept)	X_1	X_2	X_3	Time
OLS	-0.388	0.239	-0.335	0.383	1x
Nds-LTS	-0.614	0.245	0.106	-0.136	3x
Random-Lts	-0.657	0.243	0.122	-0.138	17x
Exact-LTS	-0.556	0.244	0.043	-0.100	40x

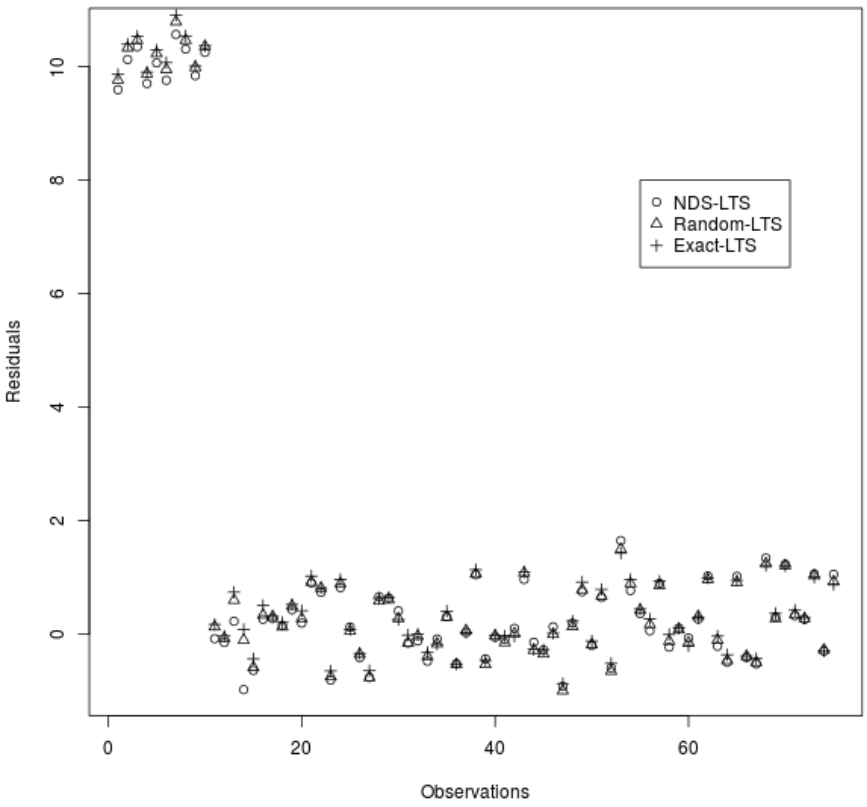

Figure 1: Residuals of estimators on Hawkins, Bradu and Kass Data

Table 6: Nds-LTS and Exact-LTS result on Animals data

	(Intercept)	$log(Body)$	Time
OLS	2.172	0.592	1x
Nds-LTS	2.015	0.771	2x
Random-Lts	1.952	0.771	22x
Exact-LTS	1.952	0.771	25x

Table 7: n=1000 + 1000, p=4, X-Space outliers

Contamination: 15%				
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$
bias	-0.120	0.003	0.000	0.000
variance	0.010	0.000	0.000	0.000
mse	0.024	0.000	0.000	0.000
median	4.872	5.003	5.000	5.000
mad	0.092	0.001	0.002	0.001
Contamination: 25%				
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$
bias	1.362	-0.009	-0.011	-0.007
variance	901.127	0.055	0.053	0.022
mse	902.982	0.055	0.053	0.022
median	5.019	5.002	4.999	5.000
mad	0.043	0.001	0.001	0.001
Contamination: 35%				
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$
bias	51.155	-0.408	-0.338	-0.323
variance	31222.410	2.008	1.361	1.221
mse	33839.261	2.174	1.476	1.325
median	4.854	5.004	5.001	4.998
mad	0.078	0.001	0.001	0.001

5.2. Monte Carlo Simulations

In previous subsection, several algorithms are applied on some datasets which are static in terms of constant sample size. In order to reveal properties of the fast update availability, we perform some Monte Carlo simulations. Data is generated using two multiple linear regression models with 4 and 10 parameters including an intercept. Independent variables are generated using a Uniform Distribution with parameters 0 and 100. Stochastic error term is generated using a standard Normal Distribution. Regression parameters are selected as $\beta = [5 \quad 5 \quad ... \quad 5]^T$. Sample size is initially determined as 1000. The dataset generated in this step is free of any outliers. After generating the initial data, a new observation is generated and appended to the initial data. The new case is an outlier with probability $P_c x = \{0.15, 0.25, 0.35\}$ and $P_c y = \{0.25, 0.35, 0.45\}$ for X-space and y-space outliers, respectively. When a newly received case is a y-space outlier, non-domination ordering has no effect and *C-step*s play the important role. That is why a higher set of contamination levels is selected for y-space outliers.

Independent variables of a contaminated observation are generated using the formula $x_{i,N+1} = Uniform(0, 100) + Uniform(\max(x_i), \max(x_i) + 3\hat{\sigma}_{x_i})$ where $\hat{\sigma}_{x_i}$ is sample standard deviation of x_i. A contaminated y value is generated using the formula $y_{N+1} = X\beta + \epsilon + Uniform(max(y), max(y) + 3\hat{\sigma}_y)$, where $\hat{\sigma}_y$ is sample standard deviation of y. In simulations, after receiving a new case, non-domination ranks are updated and a new basic subset is selected for calculating next step estimations. The process of appending new observations is repeated 1000 times for each single scenario. Tables 7 - 11 summarize the results.

In Table 7, it is shown that bias and variance of estimators increase when the contamination level increases. As a result of this, higher *MSE* values are obtained. It is also shown that the intercept estimator has higher bias and variance. This result is an indication of reduced robustness of the *Nds-LTS* estimator. Since medians and *mad*s of estimators are almost equal to 5 and 0, respectively; higher bias and variance values indicate existence of extremely large values. In Table 8, some quantiles of intercept estimator is reported. It is shown in Table 8 that, 99.8% and 91.5% of intercepts are estimated correctly by means of bias and variance for contamination levels 25% and 35%, respectively.

In Table 9, simulation results for y-space outliers in 4 parameters model are reported. As it is mentioned before, non-domination ranks have no effect on detecting this kind of outliers because the sorting algorithm is only applied on the X-space. Results reported in Table 9 indicate that *C-step*s based *LTS* algorithm yields considerable small bias and variance of estimators that are free of outliers as clearly seen in median and *mad* values.

Table 10 and Table 11 show results of the simulations performed on 10 variables model for X-space and y-space outliers, respectively. Results are surprisingly different when they are compared to the results of 4-variables model. *MSE*s of slope estimators as well as the intercept are reduced as a result of lower values of bias and variance.

Table 8: n=1000 + 1000, p=4, X-Space outliers, Some Quantiles for intercept

	25% Contamination						
Quantiles	90%	91%	92%	93%	94%	99.8%	99.9%
$\hat{\beta}_0$	5.09	5.10	5.10	5.10	5.10	5.15	676.11
	35% Contamination						
Quantiles	75%	85%	90%	91%	91.5%	92%	95%
$\hat{\beta}_0$	5.17	5.22	5.23	5.24	5.25	616.21	659.65

Table 9: n=1000 + 1000, p=4, Y-Space outliers

Contamination: 25%				
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$
bias	-0.076	0.001	-0.001	0.002
variance	0.007	0.000	0.000	0.000
mse	0.012	0.000	0.000	0.000
median	4.933	5.000	4.998	5.002
mad	0.088	0.001	0.002	0.001
Contamination: 35%				
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$
bias	-0.267	0.004	0.002	0.001
variance	0.019	0.000	0.000	0.000
mse	0.090	0.000	0.000	0.000
median	4.699	5.004	5.002	5.001
mad	0.050	0.002	0.002	0.002
Contamination: 45%				
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$
bias	-0.109	-0.000	0.001	0.001
variance	0.044	0.000	0.000	0.000
mse	0.056	0.000	0.000	0.000
median	4.851	5.000	5.002	5.001
mad	0.224	0.001	0.001	0.001

Table 10: n=1000 + 1000, p=10, X-Space outliers

Contamination: 15%										
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$	$\hat{\beta}_9$
bias	-0.24	0.00	-0.00	0.00	-0.00	0.00	0.00	0.00	0.00	0.00
var	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mse	0.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
med	4.75	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
mad	0.22	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Contamination: 25%										
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$	$\hat{\beta}_9$
bias	0.06	0.00	0.00	-0.00	-0.00	-0.00	0.00	-0.00	0.00	-0.00
var	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mse	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
med	5.10	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
mad	0.17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Contamination: 35%										
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$	$\hat{\beta}_9$
bias	-0.79	0.00	0.00	0.00	0.00	0.00	0.00	-0.00	0.00	0.00
var	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mse	0.71	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
med	4.11	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.01
mad	0.14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 11: n=1000 + 1000, p=10, y-Space outliers

Contamination: 25%										
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$	$\hat{\beta}_9$
bias	0.04	-0.00	0.00	-0.00	-0.00	0.00	0.00	-0.00	-0.00	-0.00
var	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mse	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
med	5.02	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
mad	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Contamination: 35%										
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$	$\hat{\beta}_9$
bias	0.01	0.00	-0.00	0.00	0.00	-0.00	0.00	0.00	0.00	-0.00
var	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mse	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
med	4.98	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
mad	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Contamination: 45%										
	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$	$\hat{\beta}_9$
bias	0.15	0.00	-0.00	-0.01	-0.00	0.00	-0.00	0.00	0.00	-0.00
var	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mse	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
med	5.10	5.00	5.00	4.99	5.00	5.00	5.00	5.00	5.00	5.00
mad	0.23	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Addition to this, when the contamination level increases; bias and variance do not systematically change. These results imply that *Nds-LTS* does not lose its robustness property when the model contains more variables. Note that bias and variance values of intercept estimator in clean data are 0.025 and 0.180, respectively.

6. Conclusion

Many algorithms in robust regression literature are based on selecting a small subset of observations which are free of outliers. The selecting procedure itself is generally important by means of determining masking and swamping ratios, properties under some transformations and the computation time of algorithms. Trying all possible subsets results gaining the most precise estimates in return of maximum computation time. In this paper, a two-step algorithm is devised to calculate *LTS* estimators. In the first step, the non-dominated sorting algorithm is applied on the design matrix of regression data for determining a small subset of observations in the middle of the dataset. The non-dominated sorting algorithm is used to determine solutions on the pareto frontier in the optimization theory. Its use in statistics literature is new. It is also shown in this paper that the non-dominated ordering of observations is invariant under shifting in $\mathbb{R}$ and scaling in $\mathbb{R}^+$. By a mandatory result of second property, it is not invariant under rotations. In the second step, *C-step*s are iterated to adjust *LTS* estimators. Since the process of calculating non-domination ranks is computationally expensive, it is shown that a fast update rule is available to apply this algorithm in data streams or online data with a linear order of magnitude. Algorithm is applied on some well-known datasets and the results show that the algorithm yields considerable precise estimates in a small time interval. Simulation results also show that the algorithm yields estimates with low bias and variance when new cases are appended to datasets in real time.

References

[1] J. Agulló. New algorithms for computing the least trimmed squares regression estimator. *Computational Statistics & Data Analysis*, 36(4):425–439, 2001.

[2] A. Atkinson. Fast very robust methods for the detection of multiple outliers. *Journal of the American Statistical Association*, 89(428):1329–1339, 1994.

[3] A. C. Atkinson and T.-C. Cheng. Computing least trimmed squares regression with the forward search. *Statistics and Computing*, 9(4):251–263, 1999.

[4] V. Barnett and T. Lewis. Outliers in statistical data. *John Wiley & Sons*, 286:293, 1978.

[5] C. Béguin and B. Hulliger. The bacon-eem algorithm for multivariate outlier detection in incomplete survey data. *Survey Methodology*, 34(1):91, 2008.

[6] N. Billor, S. Chatterjee, and A. S. Hadi. A re-weighted least squares method for robust regression estimation. *American journal of mathematical and management sciences*, 26(3-4):229–252, 2006.

[7] N. Billor, A. S. Hadi, and P. F. Velleman. Bacon: blocked adaptive computationally efficient outlier nominators. *Computational Statistics & Data Analysis*, 34(3):279–298, 2000.

[8] N. Billor and G. Kiral. A comparison of multiple outlier detection methods for regression data. *Communications in StatisticsSimulation and Computation®*, 37(3):521–545, 2008.

[9] B. Brown. Statistical uses of the spatial median. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 25–30, 1983.

[10] K. A. Brownlee and K. A. Brownlee. *Statistical theory and methodology in science and engineering*, volume 150. Wiley New York, 1965.

[11] L. Davies. The asymptotics of rousseeuw's minimum volume ellipsoid estimator. *The Annals of Statistics*, pages 1828–1843, 1992.

[12] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. *Lecture notes in computer science*, 1917:849–858, 2000.

[13] D. Eddelbuettel and R. François. Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, 40(8):1–18, 2011.

[14] A. Giloni and M. Padberg. Least trimmed squares regression, least median squares regression, and mathematical programming. *Mathematical and Computer Modelling*, 35(9):1043–1060, 2002.

[15] A. S. Hadi and J. S. Simonoff. Procedures for the identification of multiple outliers in linear models. *Journal of the American Statistical Association*, 88(424):1264–1272, 1993.

[16] J. Hardin and D. M. Rocke. Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator. *Computational Statistics & Data Analysis*, 44(4):625–638, 2004.

[17] D. M. Hawkins, D. Bradu, and G. V. Kass. Location of several outliers in multiple-regression data using elemental sets. *Technometrics*, 26(3):197–208, 1984.

[18] R. A. Horn. The hadamard product. In *Proc. Symp. Appl. Math*, volume 40, pages 87–169, 1990.

[19] A. M. Leroy and P. J. Rousseeuw. *Robust regression and outlier detection*. 1987.

[20] R. A. Maronna and V. J. Yohai. Robust estimation of multivariate location and scatter. *Encyclopedia of Statistical Sciences*, 1998.

[21] R. Nunkesser and O. Morell. An evolutionary algorithm for robust regression. *Computational Statistics & Data Analysis*, 54(12):3242–3248, 2010.

[22] D. Peña and V. J. Yohai. The detection of influential subsets in linear regression by using an influence matrix. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 145–156, 1995.

[23] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013.

[24] P. J. Rousseeuw and K. Van Driessen. An algorithm for positive-breakdown regression based on concentration steps. In *Data Analysis*, pages 335–346. Springer, 2000.

[25] P. J. Rousseeuw and K. Van Driessen. Computing lts regression for large data sets. *Data mining and knowledge discovery*, 12(1):29–45, 2006.

[26] P. J. Rousseeuw and B. C. Van Zomeren. Unmasking multivariate outliers and leverage points. *Journal of the American Statistical Association*, 85(411):633–639, 1990.

[27] M. H. Satman. A genetic algorithm based modification on the lts algorithm for large data sets. *Communications in Statistics-Simulation and Computation*, 41(5):644–652, 2012.

[28] M. H. Satman. A new algorithm for detecting outliers in linear regression. *International Journal of Statistics and Probability*, 2(3):p101, 2013.

[29] D. M. Sebert, D. C. Montgomery, and D. A. Rollier. A clustering algorithm for identifying multiple outliers in linear regression. *Computational statistics & data analysis*, 27(4):461–484, 1998.

[30] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3):221–248, 1994.

[31] A. J. Stromberg. Computing the exact least median of squares estimate and stability diagnostics in multiple linear regression. *SIAM Journal on Scientific Computing*, 14(6):1289–1299, 1993.

[32] K. Tamura and S. Miura. Necessary and sufficient conditions for local and global nondominated solutions in decision problems with multi-objectives. *Journal of Optimization Theory and Applications*, 28(4):501–523, 1979.

[33] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. ISBN 0-387-95457-0.