Acta Infologica
acin 2023, 7 (2), 333–347
DOI: 10.26650/acin.1327370

Research Article / Araştırma Makalesi

Comparison of Outlier Detection Methods in Linear Regression: A Multiple-Criteria Decision-Making Approach

Doğrusal Regresyonda Uç Değer Tespit Yöntemlerinin Karşılaştırılması: Çok Kriterli Karar Verme Yaklaşımı

Mehmet Hakan Satman[1]

[1](Prof.Dr.), Istanbul University, Faculty of Economics, Department of Econometrics, Beyazit, Istanbul, Turkiye

Corresponding author : Mehmet Hakan SATMAN
E-mail : mhsatman@istanbul.edu.tr

ABSTRACT

This paper focuses on the application of a suite of simulation studies to assess well-known and contemporary outlier detection methods in linear regression. These simulations vary across different parameters, including the number of observations, parameters, levels, and direction of contamination. The recorded final parameter estimates are used to rank the methods using Multiple-criteria decision-making (MCDM) tools. The study reveals that method success varies based on simulation settings. MCDM analysis results indicate a limited set of applicable methods when the contamination structure and level are unknown. Additionally, the most successful methods demand increased computation time, while some alternatives exhibit applicability within shorter durations with median rankings. These findings offer valuable insights for researchers employing regression analysis in scenarios where the underlying model is known, and the possibility of potential outliers exists.

Keywords: outlier detection, robust regression, linear regression, decision analysis

ÖZ

Bu makale, doğrusal regresyonda bilinen ve çağdaş aykırı değer tespit yöntemlerini değerlendirmek için bir dizi simülasyon çalışmasının uygulanmasına odaklanmaktadır. Bu simülasyonlar, gözlem sayılarının, parametre sayılarının ve kirlenmenin yönü ve oranı dahil olmak üzere farklı parametreler için gerçekleştirilmiştir. Kaydedilen nihai parametre tahminleri ve Çok Kriterli Karar Verme (ÇKKV) araçları kullanılarak tahmincilerin sıralanması sağlanmıştır. Çalışma, tahmincilerin başarısının simülasyon ayarlarına bağlı olarak değiştiğini ortaya koymaktadır. ÇKKV analizi sonuçları, kirlenme yönünün ve oranının bilinmediği durumlarda uygulanabilecek tahmincilerin sınırlı sayıda olduğunu göstermektedir. Ayrıca, en başarılı yöntemler artan hesaplama zamanı gerektirirken, bazı alternatifler orta sıralamalarla kısa süreler içinde uygulanabilirlik göstermektedir. Bu bulgular, altta yatan modelin bilindiği ve potansiyel aykırı değerlerin olabileceği senaryolarda regresyon analizi kullanan araştırmacılar için değerli öngörüler sunmaktadır.

Anahtar Kelimeler: uçdeğer teşhisi, dayanıklı regresyon, doğrusal regresyon, karar analizi

Submitted : 14.07.2023
Revision Requested : 09.11.2023
Last Revision Received : 09.11.2023
Accepted : 10.11.2023
Published Online : 14.12.2023

1. INTRODUCTION

Suppose the linear regression model is

$$y = X\beta + \varepsilon$$

where y is the n-vector of the response variable, X is the design matrix, β is the unknown vector of regression parameters, ε is the i.i.d. error-term with zero mean, p is the number of parameters, and n is the number of observations. The Ordinary Least Squares (OLS) estimator

$$\hat{\beta} = (X'X)^{-1}X'y$$

is an unbiased and efficient estimator of β, that is,

$$E(\hat{\beta} - \beta) = \mathbf{0}$$

and variance of $\hat{\beta}$ is minimum among the other unbiased estimators when some conditions are held. This implies

$$MSE(\hat{\beta}) = [Bias(\hat{\beta})]^2 + Var(\hat{\beta})$$

is minimum where MSE is *Mean Square Error*.

When data includes unusual observations (a.k.a. outliers), properties of OLS may drastically change depending on the level of contamination. In the case of single outlier, one-leave-out techniques and regression diagnostics are successfully applied (Belsley et al., 1980; Hadi and Chatterjee, 2015). When the level of contamination is high and known, m-leave-out techniques can be used instead but these class of methods may not be applicable as the number of all possible subsets tend to be quite large where $m < n$ is the number of outliers. In addition to this, the number of outliers is generally unknown.

Outlier detection and robust regression methods seek a solution for the outlier problem in linear regression in different but similar ways. An outlier detection procedure simply performs computation iterations to reveal the outliers. Contrary, a robust regression estimator tries to estimate an outlier-free $\hat{\beta}$ without inherently labelling any observations as clean or contaminated. When an outlier detection algorithm reports a set of contaminated observations then a robust estimate of β can be obtained by removing the contaminated observations from the data. A robust regression estimate of β can also be used to delete observations using a predefined threshold.

In this paper, 17 outlier detection and robust regression methods are simulated using a suite of Monte Carlo studies. In the simulations, Mean Square Error of estimated parameters are evaluated. The methods are ranked in the context of a multi-criteria decision-making analysis (MCDM). It is shown that the algorithms fail in many situations depending on the number of observations, number of parameters, level of contamination, and the direction of outliers. The MCDM analysis shows that only a small subset of techniques are applicable when the properties of outliers are unknown.

In Section 1 the problem and the context of the paper is introduced. In Section 2 we introduce the methods and estimators simulated in this study. The MCDM methods used in the decision analysis are also introduced. In Section 3 simulation and MCDM analysis results are reported. Finally in Section 4, we discuss the results and conclude.

2. MATERIALS AND METHODS

2.1. Outlier Detection Methods

2.1.1. `hs93`, `bacon`, and `bch2006`

`hs93`, `bacon`, and `bch2006` are multi-stage outlier detection methods and they are introduced in the same place as they follow similar patterns by construct. `hs93` is a multi-stage method and starts with an initial subset with size of $p + 1$ in its first stage (Hadi and Simonoff, 1993). The observations with lowest `DFFITS` regression diagnostics are used to construct the initial basic subset. In the second step, the initial basic subset is used to construct a basic subset by enlarging the former by adding new observations. In the last stage the subset obtained from the former stages is enlarged until a test statistic exceeds a threshold. The threshold is selected as α-quantiles of Student's T Distribution with degrees of freedom $s - p$ where s is the number of observations held by the latest subset.

`bacon` (Blocked Adaptive Computationally efficient Outlier Nominators) is a multi-stage outlier detection method (Billor et al., 2000). In the first stage, an initial basic subset is created which is considered as free of outliers. In this

stage, a sample of $p+1$ observations is created and enlarged until the basic subset includes up to m observations. This $p+1$ sized sample is constructed through a multivariate outlier detection algorithm which is only applied on the design matrix. The method is iterated until a specific t-statistic reaches a predefined cut-off value. The method requires the parameter m to be set.

`bch2006` is a multi-stage outlier detection method and it shares similar patterns to that used in the `bacon` procedure (Billor et al., 2006). The method initially calculates the Mahalanobis distances for all rows of the design matrix excluding the intercept using the coordinate-wise median instead of the sample mean for the location estimate. Best h observations are selected to build a vector of squared Mahalanobis distances. The generated basic-subset is then fed into an iteratively weighted least squares procedure, and this step is iterated until a maximum number of iterations is reached.

2.1.2. *cm97 and ccf*

`cm97` starts with construction of weights using the diagonal elements of the hat matrix using the formula

$$w_i = \frac{1}{\max(H_{ii}, \bar{p})}$$

where $H = X(X'X)^{-1}X'$ and $\bar{p} = p/n$ (Chatterjee and Mächler, 1997). A weighted least squares regression is applied using the weights w_i. The weights are updated using the formula

$$w_i = \frac{(1 - H_{ii})^2}{\max(|r_i|, m)}$$

until the estimated regression coefficients are stabilized where r_i is the ith residual and m is the sample median of absolute residuals.

`ccf` is a fast regression method that is robust to outliers and shares a similar logic with the `cm97` method. The method starts with a weighted least squares estimation with ith weight is set to $w_i = n/2$ for all observations by default (Barratt et al., 2020). The weights are updated using the formula

$$w_i = \Gamma \mathbf{sign}(e_i^2 - \alpha)$$

where $\alpha = p \times \sum_i^n e_i^2$, and e_i is the ith residual. The authors suggest selecting the Γ parameter as 0.1. The iterations of weight updating are repeated until a predefined maximum number of iterations is reached.

2.1.3. *imon2005*

`imon2005` implements a robust version of the well-known regression diagnostics `DFFITS`, namely `GDFFITS` (Rahmatullah Imon, 2005). The method starts with constructing an outlier-free h-subset through a robust fit estimator. The authors suggest using `lms` but any other robust fitting algorithm can be used instead. It is also suggested that the observations with `GDFFITS` statistic that exceed $3\sqrt{\frac{p}{h}}$ are labelled as outliers.

2.1.4. *ks89*

`ks89` method starts with calculating Studendized residuals and considers the first p observations regarding the corresponding smallest values (Kianifard and Swallow, 1989). The initial subset is enlarged using the recursive residuals. The recursive residuals are calculated using the formula

$$w_k = (y_k - X_k'\hat{\beta})/\sqrt{1 + X_k'(X^{*\prime}X^*)^{-1}X_k}$$

where w_k is the kth recursive residual, X^* is the subset of the design matrix with elements corresponding to first $k-1$ smallest recursive residuals. The iterations are repeated until $k = n$. The observations that have standardized recursive residuals greater than a specific threshold are labelled as outliers. The threshold can be selected as α-quantiles of a Student's T distribution with degrees of freedom $n - p - 1$.

2.1.5. `lad` and `quantilereg`

The `lad` (Least Absolute Deviations) estimator minimizes the sum of absolute residuals and has a unique solution obtained by a goal programming context (Narula et al., 1999). Supposing e_i^- and e_i^+ denote the ith residual, $e_i^- > 0$ if the ith residual is negative, $e_i^+ > 0$ if the ith residual is positive, otherwise it fits the regression equation. The linear objective function

$$\min z = \sum \left(e^- + e^+ \right)$$

is minimized subject to the constraints

$$X\beta + e^- - e^+ = y$$

where $e_i^- \geq 0$, $e_i^+ \geq 0$, $\beta_j \in \mathcal{R}$, $i = 1, 2, \ldots, n$, and $j = 1, \ldots, p$. Similarly, `quantilereg` (Quantile Regression) estimates a predefined conditional quantile of the response variable y (Yu et al., 2003). `quantilereg` regression parameters minimizes the linear objective function

$$\min z = \sum \left[(1-\tau)e^- + \tau e^+ \right]$$

under the same constraints of `lad` where $0 \leq \tau \leq 1$. When τ is set to 0.25, 0.50 or 0.75, well-known conditional quartiles are estimated. Note that any other percentile value can be selected, instead. When τ is set to 0.50, the conditional median of the response variable is estimated given a set of exploratory variables.

2.1.6. `lms`, `lts`, and `lta`

`lms` (Least Median of Squares) estimator

$$\min \text{median } e^2$$

minimizes the sample median of squared residuals (Rousseeuw, 1984), whereas, `lts` (Least Trimmed Squares) estimator

$$\min \sum_{i=1}^{h} e_i^2$$

minimizes the sum of first h ordered squared residuals where h is at least half of the data (Rousseeuw and Van Driessen, 2006). Similarly `lta` estimator

$$\min \sum_{i=1}^{h} |e_i|$$

minimizes the sum of the first h ordered absolute residuals (Hawkins and Olive, 1999). Since the objective function of these estimators are not in closed-form, the estimation process requires comprehensive iterations. Rousseeuw (1984) proposed a random sampling based algorithm for `lms`. Rousseeuw and Van Driessen (2006) devised a fast algorithm for `lts` in which a couple of samples of size p are randomly drawn and enlarged to size h using *concentration steps* (c-steps).

2.1.7. `py95`

`py95` is a method in which the eigen structure of

$$M = \frac{1}{ps^2} EDHDE$$

matrix is investigated where $s^2 = \sum e^2/(n-p)$, H is hat matrix, D is $n \times n$ diagonal matrix with elements $1/(1-H_{ii})$, E is $n \times n$ diagonal matrix with elements e_i, e_i is the ith residual (Peña and Yohai, 1995). Differently, `py95` reports

suspicious observations rather than absolute outliers. Suppose that the v is one of the eigenvectors of M. Let $a_i = v_i/v_{i-1}$ for $i = n, n-1, \ldots, c_1$, $b_j = v_j/v_{j+1}$ for $j = 1, 2, \ldots, c_2$, $c_1 = c_2 = \lfloor n/4 \rfloor$, and oc is vector of ordered coordinates. If none of $a_i > k$ for $i \in oc$ and $b_j > k$ for $j \in oc$, then there is no any suspicious observations where k can be selected as 2.5. Otherwise, the method returns the set of suspicious outliers.

2.1.8. `satman2013` and `satman2015`

`satman2013` is a two-stage method for detecting outliers in linear regression (Satman, 2013). In the first stage of the method, a subset of outlier-free observations is created using a robust covariance matrix estimation inspired by the *Comediance* statistic (Huo et al., 2012). This covariance matrix is calculated in reasonably small times when it is compared to the MVE and MCD (Van Aelst and Rousseeuw, 2009; Rousseeuw and Driessen, 1999) but lacks a couple of nice statistical properties such as rotation invariance. The method continues with a weighted least squares estimation using the weights obtained by the former stage. Finally, the method iterates *c-steps* defined in (Rousseeuw and Van Driessen, 2006) using the clean subset of observations obtained.

Similarly, `satman2015` (Satman, 2015) is also a two-stage method but it differs in constructing the basic subset. Instead using the *Comediance* measure, the method constructs an initial subset using the design matrix by applying a multi-dimensional sorting algorithm, e.g. non-dominated sorting algorithm defined in (Deb, 2015). A $p + 1$ subset of initial subset of observations are selected from the most-middle of the data. This selection method is not invariant to affine transformations.

2.1.9. `smr98` and `asm2000`

`smr98` algorithm starts with an OLS estimation (Sebert et al., 1998). A single-linkage clustering is then applied on the standardized pairs of $\hat{y}$ and $\hat{e}$. The cluster tree is cut using the Mojana criterion

$$\bar{h} + 1.25\sigma_h$$

where h is the vector of heights of dendrogram branches. Clusters with the majority of observations are labeled as clean. The standardized pairs of $(\hat{y}, \hat{e})$ play a role of dimension reduction, so the algorithm works perfectly when the number of regressors is small, e.g. $p = 2$. The performance of the algorithm drastically reduces in higher dimensions. `asm2000` solves this problem by applying a robust fit at the very early steps of the `smr98` algorithm. The clustering stage is based on the robust estimates of $\hat{y}$ and $\hat{e}$ (Adnan et al., 2000).

2.2. SIMULATION STUDY

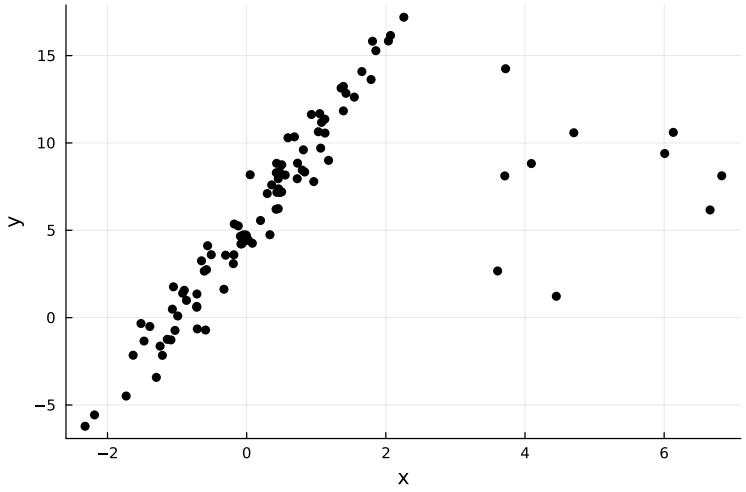
In the simulation study, regression data is created using the following data generating process: The number of observations and the number of regression parameters selected as $n = 100, 500, 1000$ and $p = 5, 10, 25$, respectively. Each single design matrix has 1s in the first column, that is, the models include an intercept term. Exploratory variables and the error term are drawn from independent Normal distributions with zero mean and unit variance. Regression parameters are set to $[5, 5, \ldots, 5]$. Regression data is then contaminated either in $x-$ and $y-$ directions with the ratios of $c = 0.10, 0.20, 0.30$. Variables are contaminated using the formula

$$V_i = \max(V) + r_i$$

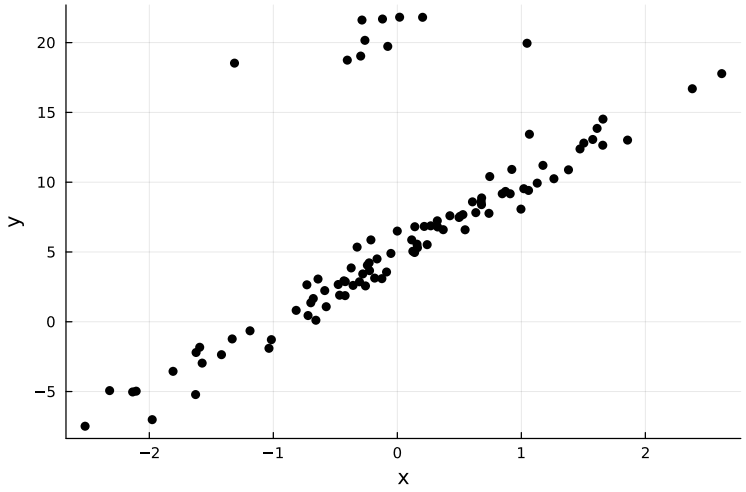
where r_i is a random value drawn from a Uniform(0, 5) distribution, V is either the response variable or columns of the design matrix excluding the intercept, and $\max(V)$ is the maximum value of V including the V_i. $x-$ outlier observations are contaminated in all dimensions.

Figure 1(a) represents a random data contaminated in $x-$ direction. As the used contamination formula indicates, outlier values are at least distant as the maximum value of the majority of observations. Similarly, Figure 1(b) represents a random data with outliers in $y-$ direction. Note that the configuration of $p = 2$ is never used in simulations but the same logic is applied in greater dimensions of spaces.

If the method $\mathcal{M}$ is a robust regression estimator, then the reported $\hat{\beta}_i$ is used to calculate the MSE values. If the method $\mathcal{M}$ is an outlier detection method and the reported outlier set is $\mathcal{S}$ then OLS parameters are estimated using the complement set of $\mathcal{S}$. By this setting, all of the methods are considered as regression estimators and low MSE values are the signals and indicators of the low masking and swamping effects of method $\mathcal{M}$. Since the data generating process differs in the number of parameters, *mean of mean square errors* (mmse) are calculated and presented for each single setting in the simulation results.



(a) x-outliers



(b) y-outliers

Figure 1: Simulation data with $n = 100$, $p = 2$, and $c = 0.10$

2.3. Multiple-criteria Decision-Making Tools

Suppose a multiple-criteria decision problem is presented as in the Table 1 where $C_1, \ldots, C_m$ are criteria, $w_1, \ldots, w_m$ are weights of criteria, $g_1(A_k), \ldots, g_m(A_k)$ are functions that takes a cost or a gain value for the alternative A_k, and $A_1, \ldots, A_n$ are alternatives. Since a sorting operation requires the $\leq$ operator is defined for elements of vector A, Table 1 is not called to be *sortable* because of the operator $\leq$ is not defined in $\mathcal{R}^m$ (or at least it doesn't have an exact and unique definition), $A_1, A_2, \ldots, A_n$ are not sortable.

Multiple-criteria decision-making tools are methods which are defined for sorting, a.k.a. ranking, alternatives $A_1, \ldots, A_n$ by using different kinds of comparing operators. The TOPSIS method (Hwang and Yoon, 1981) scores the alternatives using the Euclidean distance of weighted normalized A_i vectors to best-ideal and worst-ideal vectors. VIKOR (Opricovic, 1998; Opricovic and Tzeng, 2002) scores the alternatives using the formula

$$v\frac{s_i - \min s}{\max s - \min s} + (1 - v)\frac{r_i - \min r}{\max r - \min r}$$

where s_i and r_i are the sum and maximum of the ith row of weighted normalized decision matrix, respectively, and v can be selected as 0.5. ARAS (Zavadskas and Turskis, 2010) creates an extended decision matrix by adding an additional row that contains ideal values of all alternatives. A vector of *Utility degress* is then formed to score alternatives. WASPAS (Zavadskas et al., 2012) utilities scores by using the product of normalized decision matrix and row sums. COPRAS (Zavadskas et al., 1994) scores the alternatives using the formula

Criteria	C_1	C_2	...	C_m
Weights	w_1	w_2	...	w_m
Functions	f_1	f_2	...	f_m
A_1	$g_1(A_1)$	$g_2(A_1)$	...	$g_m(A_1)$
A_2	$g_1(A_2)$	$g_2(A_2)$	...	$g_m(A_2)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
A_n	$g_1(A_n)$	$g_2(A_n)$	...	$g_m(A_n)$

Table 1: A generic multiple-criteria decision problem

$$S_i = \frac{Q_i}{\max Q}$$

where $i = 1, 2, \ldots, n$, $Q_i = s_i^+ + \sum_{i=1}^{n} \frac{s_i^-}{s_i^+ Z}$, $Z = \sum_{i=1}^{n} 1/s_i^-$, s_i^+ and s_i^- are sums of rows of the normalized decision matrix regarding to the direction of optimization, e.g. either maximization or minimization, respectively.

Selection of criteria weights depends on the researcher and it is generally subjective. CRITIC (Diakoulaki et al., 1995) is an automatic method for selecting the importance level of criteria, a.k.a. weights. CRITIC weights are calculated using

$$w_j = s_j / \sum s$$

where s_j is the score of the jth criterion defined as

$$s_j = \mathcal{N}_j \sum \mathcal{F}_j$$

and $\mathcal{N}_j$ is standard deviation of jth column of the normalized decision matrix, $\mathcal{F}_j$ is jth column of the matrix $\mathcal{F}$, $\mathcal{F} = \mathbf{1} - \hat{\Sigma}$, $\hat{\Sigma}$ is the sample correlation matrix of the normalized decision matrix.

2.4. The Software

Simulation study and the multiple-criteria decision-making tools are applied with Julia (Bezanson et al., 2017). Julia is a fast, dynamic and compiled programming language that is mostly used in scientific computing. Selection of the programming language is mostly pragmatic as the simulation study requires 54000 iterations for each single estimator and the required functionality is packed compactly in a single environment. The Julia package *LinRegOutliers* is used in simulations (Satman et al., 2021a). This package implements all of the estimators used in this study purely in Julia. The multiple-criteria decision-making analysis is applied using the Julia package *JMcDM* (Satman et al., 2021b).

The methods of the *LinRegOutliers* package are implemented in a unified way and they are called in a scheme of

```
method(X, y)
```

where `X` is the design matrix, `y` is the response vector, and `method` is either `lms`, `lts`, `hs93`, etc. The *JMcDM* package is implemented in a similar way and a single MCDM method is called like

```
method(decisionMat, weights, directions)
```

where `decisionMat` is the decision matrix, `weights` is the vector of weights of criteria, and `directions` is the vector of directions of optimizations which can be either `minimum` or `maximum`. `method` is the function name and it can take values `topsis`, `waspas`, `copras`, etc. Use of the methods are explained in a great detail in papers Satman et al. (2021a) and Satman et al. (2021b), respectively.

3. RESULTS

Tables 2 - 4 summarize the simulation results. In these tables, average MSE values of estimates ($\hat{\beta}$) are reported for different contamination ratios ($c = 0.10, 0.20, 0.30$), outlier direction (either in x-space or y-space), and number of parameters ($p = 5, 10, 25$).

Table 2 summarizes the simulation results for $n = 100$. When the contamination in x-space is low ($c = 0.10$) and $p = 5, 10$; `bacon`, `asm2000`, `lts`, and `lta` have relatively smaller mmse values. `hs93` comes into scenes in higher dimensions ($p = 25$). This situation is also current for higher contamination rates ($c = 0.20, 0.30$). When $p = 25$ and contamination rates are higher, `hs93`, `bacon`, and `lta` have better performance.

In the case of y-outliers and $n = 100$, most of the methods are fine except `bacon`, `imon2005`, and `ccf` for small contamination rates ($c = 0.10$). When the contamination rate is increased to 0.20, `smr98`, `py95`, `ks89`, `lms`, `satman2015` tend to have larger mmse values. When the contamination rate is maximum, the winners are `lts`, `asm2000`, and `hs93` with distant mmse values compared to the remaining ones.

Table 3 summarizes the results for $n = 500$. When the dimensionality and the contamination is low ($p = 5$, $c = 0.10$); `bacon`, `asm2000`, `imon2005`, `satman2013`, `lts`, `lms`, and `lta` have better performance in the presence of x-outliers. The list remains the same when $p = 10$. In higher dimensions ($p = 25$) `satman2013` is replaced by `hs93` by their corresponding mmse values. A small subset of the list survives in higher dimensions and higher contamination rates. `bacon`, `hs93`, and `lta` are successors for $c = 0.20$. When $c = 0.30$, only `bacon` and `hs93` have relatively smaller mmse values.

In the case of y-outliers and $n = 100$, $p = 5$, and $c = 0.10$, the methods have similar performance by means of mmse. This situation remains the same for higher dimensions and contamination rate. In the worst case of $p = 25$ and $c = 0.30$ `hs93`, `bch2006`, `satman2013`, `lts`, `lad`, `quantilereg`, and `cm97` have relatively smaller mmse values and can be considered as applicable.

Table 4 summarizes the results for $n = 1000$. `bacon`, `asm2000`, `imon2005`, `smr98`, `satman2013`, `lts`, `lta`, and `lms` have better performance for $p = 5$ and $c = 0.10$ in the presence of x-outliers. In the case of high contamination rates only `asm2000`, `lts`, and `lta` have distant mmse values to the remaining elements of the list. In the worst case of $p = 25$ and $c = 0.30$, `hs93` and `bacon` are well ahead regarding their low mmse values.

In the case of y-outliers and $n = 1000$, $p = 5$, and $c = 0.10$, all of the methods are applicable. When $c = 20$ and $p = 10$ `imon2005` and `ccf` exit the list. In the worst case of $p = 25$ and $c = 0.30$ most of the methods are applicable except `ks89`, `py95`, `lta`, `lms`, `imon2005`, `ccf`, and `satman2015`.

Success of methods differ regarding the number of observations, the number of parameters, the contamination rate, and the direction of contamination. However, these factors are generally unknown by the researcher, that is, the multivariate data is not visible to plots even a dimension reduction tool is applied to data[1]. As a consequence, the researcher is almost blind to direction of outliers and the contamination ratio.

Table 5 represents the scores calculated by TOPSIS, VIKOR, ARAS, WASPAS, and COPRAS methods to the decision matrix of simulation results. In the decision matrix, rows (the alternatives) are the methods. The criteria are formed by the simulation settings. The ith row and the jth column of the decision matrix represents the mmse of the method $\mathcal{M}_i$ for regression setting $\int_j$. In Table 5 it is shown that the `ols` has the lowest rank by all of the methods since the simulation data is always contaminated. The other methods have higher scores as expected. `asm2000` is in the top three for all MCDM methods. `hs93` is in the top three for 4 out of 5 methods whereas `lta` takes a place for 3 out of 5 methods. `lta`, `hs93`, `asm2000`, `lts`, `bacon` take a place for at least one MCDM method.

The success of the methods is compared in terms of computation time as well as mmse values. Table 6 represents the average absolute times and relative times elapsed by the methods[2].

Table 6 shows that the statistical properties and the consumed times of methods are related as the most successful methods `hs93` and `lta` consume more time than the others. `ccf` is also consistent as it has lower ranks by the MCDM and lower computation times. `satman2013` is an interesting method as it takes 8th or 9th row in the rankings with its relatively small computation times. `cm97` has similar speed properties with lower rankings. The cheapest-success method is `bacon` as it takes higher rankings with median computation times.

[1] Classical covariance matrix based methods are not robust to outliers and, for instance, *Principal Component Analysis* requires a robust covariance matrix to be estimated in the presence of outliers. Performance of covariance estimators (by means of 1. Detecting the true outliers, 2. Rejecting the false outliers, 3. Unbiased estimate of the location vector, 4. Efficient estimation of variance, etc.) is another issue and this subject is out of scope of this paper

[2] The absolute calculation times are average of all computation times in all simulation settings. These elapsed times are measured using a MacBook Pro with 8GB of memory and 2Ghz 8-core CPU. Since the elapsed times differ in many hardware configurations, relative average times are reported. The time consumed by `ols` is set to $1x$. Other methods' average times are divided by absolute elapsed time of `ols`. By this representation, the elapsed times are directly comparable.

4. DISCUSSIONS AND CONCLUSIONS

Robust regression methods take the model and a dataset as input and return the estimate of regression parameters whereas outlier detection methods take the same input and return a set of indices of outliers. When the reported outlier set is omitted from the data, the estimated OLS parameters are considered robust. In this study the well-known and modern outlier detection methods and robust regression methods are simulated for different number of observations, the number of parameters, levels of contamination, and direction of contamination. MSE of estimated parameters are recorded during the simulations. Simulation results show that the success of a method differs regarding the simulation setting. However, the researchers are generally not aware of the underlying data generating process and the contamination structure and selection of the proper method is a decision problem.

Multiple-criteria decision-making (MCDM) tools are generally used by ranking the alternatives using a given set of criteria and importance levels of these criteria. TOPSIS, Vikor, ARAS, WASPAS, and COPRAS are some well-known MCDM tools applied in the decision making literature. In this paper, these MCDM tools are used to rank outlier detection methods by their average MSE (mmse) of parameter estimates. The criteria are formed by each single setting of the data generating process. Since the importance level is unknown or subjective, the CRITIC method is used to determine a set of weights.

The results of the MCDM analysis show that the `ols` estimator has the lowest rank as expected just because the simulation study is performed on the contaminated data. All of the MCDM tools scored `asm2000` in the top three whereas `hs93` is ranked in top three for four out of five listings. `lta` takes the place of three out of five MCDM methods in the top 3. `lta`, `hs93`, `asm2000`, `lts`, `bacon` are top-ranked for at least one MCDM method. If the researcher has no idea of the underlying data generating process, results of these methods can be considered.

The computation times consumed by the methods are also reported. It is shown that the more successful methods take more computation time. `satman2013` is an interesting method as it takes 8th or 9th row in the rankings with its relatively small computation times. `cm97` has similar speed properties with lower rankings. The cheapest-success method is `bacon` as it takes higher rankings with median computation times. If the consumed time is an issue, `bacon` can be used with reasonably small MSE of estimates in many settings.

If the direction and level of contamination is known, results of the simulations are directly comparable. When $n = 1000$, $p = 25$ and the contamination is at the maximum level, `hs93` is the most performant method by means of lower MSE. If the direction of contamination is known and the presence of y-outliers is the case, `hs93`, `bch2006` are the absolute winners with a small time difference.

The simulation results is a confirmation of the previous simulation studies reported in Billor and Kiral (2008) and Wisnowski et al. (2001) in some sights. Instead of reporting the masking and swamping ratios, this study is original as it reports MSE of estimated parameters. The former studies utilize a comprehensive study with a wider range of contamination levels and extra contamination directions and structures. Our study differs as it tests the methods in larger data sets including the ones with $n = 1000$, $p = 25$ and covers a wider and novel set of methods to compare.

Combining the building blocks of successful methods for a faster and more robust outlier detection procedure and developing new methods would be the subject of future works.

Algorithm	$c = 0.10$, d =x			$c = 0.10$, d =y		
	$p = 5$	$p = 10$	$p = 25$	$p = 5$	$p = 10$	$p = 25$
asm2000	**0.012**	**0.012**	**0.016**	0.022	0.030	0.048
bacon	**0.012**	0.015	0.090	**0.012**	0.015	3.205
bch2006	16.400	30.081	43.537	0.041	0.040	0.055
ccf	17.682	23.629	32.232	0.135	1.402	7.934
cm97	17.134	24.072	35.438	0.021	0.021	0.028
hs93	0.216	0.470	0.073	**0.012**	**0.012**	**0.016**
imon2005	2.089	23.562	32.354	0.857	1.621	6.242
ks89	17.596	24.594	36.397	0.040	0.074	0.334
lad	17.944	24.881	37.132	0.025	0.026	0.032
lms	0.048	0.046	0.078	0.048	0.048	0.077
lta	0.079	0.083	0.109	0.076	0.084	0.112
lts	0.066	0.060	0.148	0.068	0.061	0.052
ols	18.256	23.713	32.329	2.505	3.776	7.696
py95	13.307	24.140	38.775	0.044	0.068	0.271
quantilereg	18.003	24.949	36.807	0.025	0.025	0.031
satman2013	0.350	13.780	38.723	0.061	0.051	0.046
satman2015	20.823	29.247	48.788	0.059	0.050	0.089
smr98	5.139	21.288	36.532	0.014	0.026	0.139

Algorithm	$c = 0.20$, d =x			$c = 0.20$, d =y		
	$p = 5$	$p = 10$	$p = 25$	$p = 5$	$p = 10$	$p = 25$
asm2000	**0.013**	**0.014**	37.395	0.090	0.084	0.202
bacon	0.055	0.017	0.086	0.135	0.491	41.722
bch2006	22.286	30.565	46.163	0.038	0.045	0.105
ccf	19.788	24.718	34.325	2.418	10.052	17.402
cm97	19.884	25.439	37.679	0.054	0.047	0.076
hs93	0.870	0.443	**0.019**	**0.014**	**0.015**	**0.030**
imon2005	15.372	24.660	34.233	5.648	8.908	17.886
ks89	20.084	26.077	39.141	0.508	1.064	6.031
lad	20.150	26.220	39.578	0.051	0.048	0.075
lms	0.059	0.084	57.019	0.062	0.084	7.460
lta	0.075	0.094	0.229	0.076	0.092	0.225
lts	0.059	0.053	49.205	0.058	0.056	0.053
ols	19.719	24.674	34.308	8.494	10.297	18.027
py95	20.043	26.880	43.595	1.100	1.291	4.736
quantilereg	20.157	26.284	40.096	0.050	0.050	0.075
satman2013	18.126	27.329	42.747	0.049	0.048	0.159
satman2015	22.551	31.344	58.984	0.150	0.728	20.408
smr98	16.419	26.095	40.112	0.043	0.331	3.717

Algorithm	$c = 0.30$, d =x			$c = 0.30$, d =y		
	$p = 5$	$p = 10$	$p = 25$	$p = 5$	$p = 10$	$p = 25$
asm2000	**0.016**	**0.017**	51.756	0.307	0.367	0.394
bacon	9.324	7.070	7.990	1.146	4.071	102.800
bch2006	22.895	32.088	48.924	0.038	0.041	5.497
ccf	20.510	25.398	36.658	12.891	21.031	30.651
cm97	20.558	26.297	41.145	0.190	0.268	10.264
hs93	1.205	0.045	**0.023**	**0.021**	**0.020**	1.563
imon2005	19.499	25.498	36.469	14.851	20.330	30.489
ks89	20.942	26.957	41.661	3.172	6.749	24.486
lad	21.029	27.221	43.547	0.126	0.153	7.418
lms	0.096	4.699	82.574	0.089	0.785	39.316
lta	0.071	0.114	44.034	0.072	0.115	28.762
lts	0.049	1.423	71.372	0.049	0.047	**0.120**
ols	20.249	25.424	36.646	17.996	21.026	29.899
py95	20.998	27.834	54.573	8.208	9.528	24.205
quantilereg	20.993	27.183	43.366	0.128	0.134	8.077
satman2013	21.549	29.471	48.733	0.045	0.055	10.647
satman2015	23.830	35.256	89.485	63.767	63.865	88.561
smr98	20.285	27.583	45.360	1.422	4.714	26.167

Table 2: Average MSE for $n = 100$

Algorithm	$c = 0.10$, d =x			$c = 0.10$, d =y		
	$p = 5$	$p = 10$	$p = 25$	$p = 5$	$p = 10$	$p = 25$
asm2000	**0.002**	**0.002**	**0.002**	0.004	0.004	0.010
bacon	**0.002**	**0.002**	**0.002**	**0.002**	**0.002**	**0.002**
bch2006	7.936	22.874	29.937	0.007	0.009	0.010
ccf	17.371	21.816	24.900	0.090	0.813	3.769
cm97	17.423	21.799	25.203	0.008	0.006	0.004
hs93	0.130	0.327	0.252	**0.002**	**0.002**	**0.002**
imon2005	0.004	0.004	9.546	0.803	0.881	1.254
ks89	17.270	21.878	25.619	0.013	0.018	0.032
lad	17.510	21.991	25.626	0.008	0.006	0.005
lms	0.021	0.027	0.579	0.021	0.027	0.125
lta	0.029	0.042	0.070	0.029	0.043	0.072
lts	0.018	0.016	0.012	0.018	0.017	0.013
ols	17.827	21.803	24.907	2.629	3.022	3.821
py95	14.599	21.698	25.768	0.016	0.021	0.034
quantilereg	17.455	22.000	25.650	0.008	0.006	0.005
satman2013	0.016	3.474	27.198	0.016	0.015	0.011
satman2015	18.312	23.765	28.833	0.016	0.014	0.011
smr98	0.161	8.767	25.019	0.003	0.003	0.003

Algorithm	$c = 0.20$, d =x			$c = 0.20$, d =y		
	$p = 5$	$p = 10$	$p = 25$	$p = 5$	$p = 10$	$p = 25$
asm2000	**0.002**	**0.003**	14.801	0.027	0.044	0.028
bacon	0.003	**0.003**	**0.003**	0.003	0.011	0.018
bch2006	19.446	24.112	30.776	0.006	0.008	0.009
ccf	19.037	22.466	25.357	2.324	10.111	12.729
cm97	19.029	22.552	25.664	0.032	0.019	0.011
hs93	1.517	0.295	0.106	**0.002**	**0.003**	**0.003**
imon2005	0.524	3.177	25.279	6.378	7.638	9.953
ks89	19.055	22.757	26.167	0.390	0.500	0.803
lad	19.105	22.717	26.133	0.026	0.016	0.011
lms	0.072	1.764	24.534	0.067	0.321	6.204
lta	0.031	0.055	0.148	0.030	0.054	0.146
lts	0.015	0.013	18.859	0.014	0.014	0.012
ols	19.050	22.445	25.345	9.879	11.094	12.625
py95	19.067	22.907	26.260	0.558	0.716	0.777
quantilereg	19.072	22.716	26.229	0.026	0.016	0.011
satman2013	17.174	23.631	28.475	0.013	0.012	0.010
satman2015	20.096	24.512	29.793	0.014	0.012	0.010
smr98	7.287	21.515	25.999	0.003	**0.003**	**0.003**

Algorithm	$c = 0.30$, d =x			$c = 0.30$, d =y		
	$p = 5$	$p = 10$	$p = 25$	$p = 5$	$p = 10$	$p = 25$
asm2000	**0.003**	0.327	26.057	0.188	0.098	0.163
bacon	10.414	7.387	1.362	1.601	0.675	0.585
bch2006	20.182	24.834	31.424	0.006	0.005	0.008
ccf	19.508	22.750	25.615	15.693	23.772	26.795
cm97	19.524	22.869	26.008	0.122	0.070	0.041
hs93	3.272	0.593	**0.003**	**0.003**	**0.003**	**0.003**
imon2005	5.178	16.416	25.854	18.310	20.939	25.607
ks89	19.591	23.045	26.509	2.876	3.510	5.491
lad	19.606	23.067	26.615	0.075	0.044	0.027
lms	1.799	14.530	27.409	0.539	4.349	16.216
lta	0.033	**0.077**	8.549	0.034	0.078	7.022
lts	0.011	0.682	34.699	0.011	0.011	0.010
ols	19.496	22.740	25.621	22.363	24.151	26.671
py95	19.591	23.210	26.576	7.666	7.348	6.340
quantilereg	19.619	23.059	26.656	0.075	0.044	0.027
satman2013	19.955	24.459	29.300	0.011	0.010	0.009
satman2015	20.577	25.150	31.301	153.840	111.669	63.315
smr98	14.966	22.961	26.526	0.004	0.300	0.927

Table 3: Average MSE for $n = 500$

Algorithm	$c = 0.10, d =$x			$c = 0.10, d =$y		
	$p = 5$	$p = 10$	$p = 25$	$p = 5$	$p = 10$	$p = 25$
asm2000	**0.0010**	**0.0010**	**0.0010**	**0.0010**	**0.0010**	**0.0010**
bacon	**0.0010**	**0.0010**	**0.0010**	**0.0010**	**0.0010**	**0.0010**
bch2006	5.7756	22.0429	26.7652	0.0036	0.0039	0.0045
ccf	17.5024	21.6381	24.2338	0.0804	0.7822	3.4871
cm97	17.5442	21.6073	24.3706	0.0066	0.0043	0.0029
hs93	1.7994	4.0458	0.9715	**0.0010**	**0.0010**	**0.0010**
imon2005	**0.0010**	**0.0010**	**0.0010**	0.8566	0.8655	1.0107
ks89	17.3462	21.6626	24.6064	0.0088	0.0117	0.0185
lad	17.5348	21.6808	24.5792	0.0060	0.0040	0.0028
lms	0.0230	0.0423	3.5922	0.0224	0.0347	0.3284
lta	0.0214	0.0385	0.0674	0.0220	0.0377	0.0664
lts	0.0096	0.0091	0.0334	0.0094	0.0093	0.0078
ols	17.8770	21.6403	24.2406	2.7914	3.1037	3.6352
py95	16.1162	21.4521	24.6933	0.0190	0.0161	0.0191
quantilereg	17.5272	21.6874	24.5982	0.0060	0.0041	0.0028
satman2013	0.0086	1.2815	25.9011	0.0088	0.0083	0.0064
satman2015	16.4922	22.8783	26.8538	0.0090	0.0077	0.0062
smr98	0.0024	2.7959	23.6374	0.0014	**0.0010**	**0.0010**

Algorithm	$c = 0.20, d =$x			$c = 0.20, d =$y		
	$p = 5$	$p = 10$	$p = 25$	$p = 5$	$p = 10$	$p = 25$
asm2000	**0.0010**	**0.0010**	12.8159	0.0322	**0.0010**	0.0295
bacon	**0.0010**	**0.0010**	**0.0010**	**0.0010**	**0.0010**	**0.0010**
bch2006	19.1146	23.0278	27.3760	0.0030	0.0035	0.0040
ccf	18.9962	22.2326	24.5423	2.5844	11.0908	13.1101
cm97	18.9782	22.2848	24.7040	0.0294	0.0156	0.0081
hs93	11.3026	2.2483	0.0011	**0.0010**	**0.0010**	0.0013
imon2005	0.0688	0.0441	16.8873	7.0404	8.0550	9.6137
ks89	19.0040	22.3559	24.9464	0.3832	0.4586	0.6018
lad	19.0112	22.3573	24.9449	0.0234	0.0132	0.0073
lms	0.4672	6.1905	24.1882	0.1580	0.9109	7.0917
lta	0.0236	0.0499	0.1384	0.0252	0.0499	0.1381
lts	0.0078	0.0074	15.2878	0.0080	0.0078	0.0070
ols	19.0194	22.2258	24.5409	10.9012	11.8859	12.8084
py95	19.0350	22.5864	25.0380	1.2078	0.6776	0.5984
quantilereg	19.0048	22.3663	24.9542	0.0232	0.0133	0.0073
satman2013	17.3134	23.0274	26.8841	0.0074	0.0063	0.0054
satman2015	19.7148	23.5378	27.3970	0.0072	0.0070	0.0057
smr98	2.5292	19.7529	24.8598	0.0018	0.0019	0.0018

Algorithm	$c = 0.30, d =$x			$c = 0.30, d =$y		
	$p = 5$	$p = 10$	$p = 25$	$p = 5$	$p = 10$	$p = 25$
asm2000	**0.001**	0.498	24.525	0.094	0.159	0.048
bacon	10.133	7.521	1.123	2.403	0.949	0.345
bch2006	19.733	23.402	27.918	0.003	0.003	0.003
ccf	19.436	22.464	24.730	17.917	26.366	28.234
cm97	19.425	22.520	24.894	0.116	0.062	0.030
hs93	11.553	1.125	**0.001**	0.045	**0.002**	**0.002**
imon2005	1.780	3.854	24.839	20.245	22.903	26.694
ks89	19.472	22.623	25.143	3.101	3.522	4.466
lad	19.480	22.639	25.228	0.070	0.038	0.019
lms	5.995	18.192	25.641	1.536	6.981	17.702
lta	0.028	**0.073**	7.585	0.028	0.072	6.289
lts	0.006	0.751	30.136	0.006	0.006	0.006
ols	19.425	22.463	24.716	24.620	26.749	28.140
py95	19.484	22.770	25.270	11.084	7.890	5.468
quantilereg	19.447	22.621	25.217	0.069	0.038	0.019
satman2013	19.719	23.656	27.324	0.006	0.005	0.005
satman2015	20.138	23.930	28.055	183.544	153.479	61.311
smr98	10.018	22.276	25.141	**0.002**	**0.002**	0.195

Table 4: Average MSE for $n = 1000$

TOPSIS		VIKOR		ARAS		WASPAS		COPRAS	
Algorithm	Score	Algorithm	Score	Algorithm	Score	Algorithm	Score	Algorithm	Score
lta	0.922	lta	0.998	asm2000	0.528	asm2000	0.344	hs93	1.000
hs93	0.922	asm2000	0.974	bacon	0.454	bacon	0.305	lta	0.835
asm2000	0.885	lts	0.960	hs93	0.451	hs93	0.297	asm2000	0.694
lts	0.855	bacon	0.953	lts	0.183	lts	0.134	lts	0.529
lms	0.782	hs93	0.884	smr98	0.159	smr98	0.123	bacon	0.234
bacon	0.756	lms	0.867	lta	0.122	bch2006	0.078	lms	0.169
smr98	0.745	smr98	0.770	bch2006	0.112	lta	0.075	smr98	0.136
satman2013	0.720	imon2005	0.763	imon2005	0.081	satman2013	0.058	satman2013	0.126
cm97	0.682	satman2013	0.741	satman2013	0.081	cm97	0.051	cm97	0.097
lad	0.680	cm97	0.544	quantilereg	0.063	quantilereg	0.051	bch2006	0.097
quantilereg	0.680	bch2006	0.542	lad	0.063	lad	0.051	lad	0.096
bch2006	0.679	lad	0.540	cm97	0.062	imon2005	0.049	quantilereg	0.096
ks89	0.669	quantilereg	0.540	lms	0.050	lms	0.033	ks89	0.084
py95	0.666	py95	0.484	satman2015	0.030	satman2015	0.025	py95	0.079
imon2005	0.584	ks89	0.477	ks89	0.010	ks89	0.010	imon2005	0.047
satman2015	0.505	satman2015	0.415	py95	0.009	py95	0.009	satman2015	0.039
ccf	0.486	ccf	0.074	ccf	0.002	ccf	0.002	ccf	0.036
ols	0.337	ols	0.010	ols	0.001	ols	0.000	ols	0.024

Table 5: Ranking and scores

Algorithm	Absolute time	Relative time
ols	0.00013	1.000
ccf	0.00155	12.116
cm97	0.00774	60.596
satman2013	0.01599	125.211
quantilereg	0.06122	479.287
lad	0.06172	483.183
satman2015	0.06690	523.747
smr98	0.15469	1211.039
ks89	0.24363	1907.391
bacon	0.25505	1996.776
lms	0.42403	3319.777
lts	0.48258	3778.131
imon2005	0.53484	4187.311
asm2000	0.54336	4254.016
py95	0.68249	5343.253
bch2006	1.55553	12178.298
lta	4.65866	36472.915
hs93	6.57872	51505.193

Table 6: Absolute and relative elapsed times by algorithms. Relative average times are calculated due to the `ols` by setting its time to $1x$.

Peer Review: Externally peer-reviewed.
Conflict of Interest: The author has no conflict of interest to declare.
Grant Support: The author declared that this study has received no financial support.

ORCID ID of the author / Yazarın ORCID ID'si
Mehmet Hakan Satman 0000-0002-9402-1982

REFERENCES

R. Adnan, H. Setan, and M. N. Mohamad. Identifying multiple outliers in linear regression: Robust fit and clustering approach. In *The 10th FIG International Symposium on Deformation Measurements*, SESSION X : THEORY OF DEFORMATION ANALYSIS II, pages 380–389, Orange, California, USA, 2000.

S. Barratt, G. Angeris, and S. Boyd. Minimizing a sum of clipped convex functions. *Optimization Letters*, 14:2443–2459, 2020.

D. A. Belsley, E. Kuh, and R. E. Welsch. *Regression diagnostics: Identifying influential data and sources of collinearity*. 1980. ISBN 0-471-05856-4.

J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017. doi:10.1137/141000671.

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

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.

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.

S. Chatterjee and M. Mächler. Robust regression: A weighted least squares approach. *Communications in Statistics-Theory and Methods*, 26(6): 1381–1394, 1997.

K. Deb. Multi-objective evolutionary algorithms. *Springer handbook of computational intelligence*, pages 995–1015, 2015.

D. Diakoulaki, G. Mavrotas, and L. Papayannakis. Determining objective weights in multiple criteria problems: The critic method. *Computers & Operations Research*, 22(7):763–770, 1995. doi:10.1016/0305-0548(94)00059-h.

A. S. Hadi and S. Chatterjee. *Regression analysis by example*. John Wiley & Sons, 2015.

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.

D. M. Hawkins and D. Olive. Applications and algorithms for least trimmed sum of absolute deviations regression. *Computational Statistics & Data Analysis*, 32(2):119–134, 1999.

L. Huo, T.-H. Kim, and Y. Kim. Robust estimation of covariance and its application to portfolio optimization. *Finance Research Letters*, 9(3): 121–134, 2012.

C.-L. Hwang and K. Yoon. *Methods for Multiple Attribute Decision Making*. Springer Berlin Heidelberg, 1981.

F. Kianifard and W. H. Swallow. Using recursive residuals, calculated on adaptively-ordered observations, to identify outliers in linear regression. *Biometrics*, pages 571–585, 1989.

S. C. Narula, P. H. Saldiva, C. D. Andre, S. N. Elian, A. F. Ferreira, and V. Capelozzi. The minimum sum of absolute errors regression: a robust alternative to the least squares regression. *Statistics in medicine*, 18(11):1401–1417, 1999.

S. Opricovic. Multicriteria optimization of civil engineering systems, 1998.

S. Opricovic and G.-H. Tzeng. Multicriteria planning of post-earthquake sustainable reconstruction. *Computer-Aided Civil and Infrastructure Engineering*, 17(3):211–220, may 2002. doi:10.1111/1467-8667.00269.

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)*, 57(1):145–156, 1995.

A. Rahmatullah Imon. Identifying multiple influential observations in linear regression. *Journal of Applied statistics*, 32(9):929–946, 2005.

P. J. Rousseeuw. Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880, 1984.

P. J. Rousseeuw and K. V. Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223, 1999.

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

M. H. Satman. A new algorithm for detecting outliers in linear regression. *International Journal of statistics and Probability*, 2(3):101, 2013.

M. H. Satman. Fast online detection of outliers using least-trimmed squares regression with non-dominated sorting based initial subsets. *International Journal of Advanced Statistics and Probability*, 3(1):53, 2015.

M. H. Satman, S. Adiga, G. Angeris, and E. Akadal. Linregoutliers: A julia package for detecting outliers in linear regression. *Journal of Open Source Software*, 6(57):2892, 2021a. doi:10.21105/joss.02892.

M. H. Satman, B. F. Yıldırım, and E. Kuruca. Jmcdm: A julia package for multiple-criteria decision-making tools. *Journal of Open Source Software*, 6(65):3430, 2021b. doi:10.21105/joss.03430.

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.

S. Van Aelst and P. Rousseeuw. Minimum volume ellipsoid. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(1):71–82, 2009.

J. W. Wisnowski, D. C. Montgomery, and J. R. Simpson. A comparative analysis of multiple outlier detection procedures in the linear regression model. *Computational statistics & data analysis*, 36(3):351–382, 2001.

K. Yu, Z. Lu, and J. Stander. Quantile regression: applications and current research areas. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 52(3):331–350, 2003.

E. K. Zavadskas and Z. Turskis. A new additive ratio assessment (aras) method in multicriteria decision-making, 2010.

E. K. Zavadskas, A. Kaklauskas, and V. Sarka. The new method of multicriteria complex proportional assessment of projects, 1994.

E. K. Zavadskas, Z. Turskis, and J. Antucheviciene. Optimization of weighted aggregated sum product assessment. *Electronics and Electrical Engineering*, 122(6), jun 2012. doi:10.5755/j01.eee.122.6.1810.

How cite this article

Satman, M.H. (2023). Comparison of outlier detection methods in linear regression: A multiple-criteria decision-making approach. *Acta Infologica, 7*(2), 333-347. https://doi.org/10.26650/acin.1327370