

Article

Check your outliers! An accessible introduction to identifying statistical outliers in R with *easystats*

Rémi Thériault^{1,*} , Mattan S. Ben-Shachar² , Indrajeet Patil³ , Daniel Lüdtke⁴ , Brenton M. Wiernik⁵ , Dominique Makowski⁶ 

¹ Department of Psychology, Université du Québec à Montréal, Montréal, Québec, Canada;

² Independent Researcher;

³ Center for Humans and Machines, Max Planck Institute for Human Development, Berlin, Germany;

⁴ Institute of Medical Sociology, University Medical Center Hamburg-Eppendorf, Germany;

⁵ Independent Researcher, Tampa, FL, USA;

⁶ School of Psychology, University of Sussex, Brighton, UK;

* Correspondence: theriault.remi@courrier.uqam.ca.

Version February 20, 2023 submitted to Mathematics



Simple Summary: The *{performance}* package from the *easystats* ecosystem makes it easy to diagnose outliers in R and according to current best practices thanks to the `check_outliers()` function.

Abstract: Beyond the challenge of keeping up-to-date with current best practices regarding the diagnosis and treatment of outliers, an additional difficulty arises concerning the mathematical implementation of the recommended methods. In this paper, we provide an overview of current recommendations and best practices and demonstrate how they can easily and conveniently be implemented in the R statistical computing software, using the *{performance}* package of the *easystats* ecosystem. We cover univariate, multivariate, and model-based statistical outlier detection methods, their recommended threshold, standard output, and plotting methods. We conclude with recommendations on the handling of outliers: the different theoretical types of outliers, whether to exclude or winsorize them, and the importance of transparency.

Keywords: univariate outliers; multivariate outliers; robust detection methods; R; *easystats*

1. Introduction

Real-life data often contain observations that can be considered *abnormal* when compared to the main population. The cause of it—be it because they belong to a different distribution (originating from a different generative process) or simply being extreme cases, statistically rare but not impossible—can be hard to assess, and the boundaries of “abnormal” are hard to define.

Nonetheless, the improper handling of these outliers can substantially affect statistical model estimations, biasing effect estimations and weakening the models’ predictive performance. It is thus essential to address this problem in a thoughtful manner. Yet, despite the existence of established recommendations and guidelines, many researchers still do not treat outliers in a consistent manner, or do so using inappropriate strategies [1,2].

One possible reason is that researchers are not aware of the existing recommendations, or do not know how to implement them using their analysis software. In this paper, we show how to follow current best practices for automatic and reproducible statistical outlier detection (SOD) using R and the *{performance}* package [3], which is part of the *easystats* ecosystem of packages that build an R framework for easy statistical modeling, visualization, and reporting [4].

2. Identifying Outliers

Although many researchers attempt to identify outliers with measures based on the mean (e.g., z scores), those methods are problematic because the mean and standard deviation themselves are not robust to the influence of outliers and they assume a normal distribution. Therefore, current guidelines recommend using robust methods to identify outliers, such as those relying on the median as opposed to the mean [2,5,6].

Nonetheless, which exact outlier method to use depends on many factors. In some cases, eye-gauging odd observations can be an appropriate solution, though many researchers will favour algorithmic solutions to detect potential outliers, for example, based on a mathematical score.

One of the factors to consider when selecting an algorithmic outlier detection method is the statistical test of interest. When using a regression model, relevant information can be found by identifying observations that do not fit well with the model. This approach, known as model-based outliers detection (as outliers are extracted after the statistical model has been fit), can be contrasted with distribution-based outliers detection, which is based on the distance between an observation and the “center” of its population. Various quantification strategies of this distance exist for the latter, both univariate (involving only one variable at a time) or multivariate (involving multiple variables).

When no method is readily available to detect model-based outliers, such as for structural equation modelling (SEM), looking for multivariate outliers may be of relevance. For simple tests (t tests or correlations) that compare values of the same variable, it can be appropriate to check for univariate outliers. However, univariate methods can give false positives since t tests and correlations, ultimately, are also models/multivariable statistics. They are in this sense more limited, but we show them nonetheless for educational purposes.

Importantly, whatever approach researchers choose remains a subjective decision, which usage (and rationale) must be transparently documented and reproducible [5]. Researchers should commit (ideally in a preregistration) to an outlier treatment method before collecting the data. They should report in the paper their decisions and details of their methods, as well as any deviation from their original plan. These transparency practices can help reduce false positives due to excessive researchers’ degrees of freedom (i.e., choice flexibility throughout the analysis). In the following section, we will go through each of the mentioned methods and provide examples on how to implement them with R.

2.1. Univariate Outliers

Researchers frequently attempt to identify outliers using measures of deviation from the center of a variable’s distribution. One of the most popular such procedure is the z score transformation, which computes the distance in standard deviation (SD) from the mean. However, as mentioned earlier, this popular method is not robust. Therefore, for univariate outliers, it is recommended to use the median along with the Median Absolute Deviation (MAD), which are more robust than the interquartile range or the mean and its standard deviation [2,5].

Researchers can identify outliers based on robust (i.e., MAD-based) z scores using the `check_outliers()` function of the `{performance}` package, by specifying `method = "zscore_robust"`. Although Leys *et al.* [2] suggest a default threshold of 2.5 and Leys *et al.* [5] a threshold of 3, `{performance}` uses by default a less conservative threshold of ~ 3.29 .¹ That is, data points will be flagged as outliers if they go beyond $\pm \sim 3.29$ MAD. Users can adjust this threshold using the `threshold` argument, as demonstrated below.

```
library(performance)
```

¹ 3.29 is an approximation of the two-tailed critical value for $p < .001$, obtained through `qnorm(p = 1 - 0.001 / 2)`. We chose this threshold for consistency with the thresholds of all our other methods.

```
# Create some artificial outliers and an ID column
```

```
data <- rbind(mtcars[1:4], 42, 55)
```

```
data <- cbind(car = row.names(data), data)
```

```
outliers <- check_outliers(data, method = "zscore_robust", ID = "car")
```

```
outliers
```

```
70 #> 2 outliers detected: cases 33, 34.
```

```
71 #> - Based on the following method and threshold: zscore_robust (3.09).
```

```
72 #> - For variables: mpg, cyl, disp, hp.
```

```
73 #>
```

```
74 #> -----
```

```
75 #>
```

```
76 #> The following observations were considered outliers for two or more
```

```
77 #> variables by at least one of the selected methods:
```

```
78 #>
```

```
79 #> Row car n_Zscore_robust
```

```
80 #> 1 33 33 2
```

```
81 #> 2 34 34 2
```

```
82 #>
```

```
83 #> -----
```

```
84 #> Outliers per variable (zscore_robust):
```

```
85 #>
```

```
86 #> $mpg
```

```
87 #> Row car Distance_Zscore_robust
```

```
88 #> 33 33 33 3.709699
```

```
89 #> 34 34 34 5.848328
```

```
90 #>
```

```
91 #> $cyl
```

```
92 #> Row car Distance_Zscore_robust
```

```
93 #> 33 33 33 12.14083
```

```
94 #> 34 34 34 16.52502
```

95 The row numbers of the detected outliers can be obtained by using `which()` on the output object,
96 which can be used for exclusions for example:

```
which(outliers)
```

```
97 #> [1] 33 34
```

```
data_clean <- data[-which(outliers), ]
```

98 All `check_outliers()` output objects possess a `plot()` method, meaning it is also possible to
99 visualize the outliers:

```
library(see)
```

```
plot(outliers)
```

100 Other univariate methods are available, such as using the interquartile range (IQR), or based on
101 different intervals, such as the Highest Density Interval (HDI) or the Bias Corrected and Accelerated
102 Interval (BCI). These methods are documented and described in the function's [help page](#).

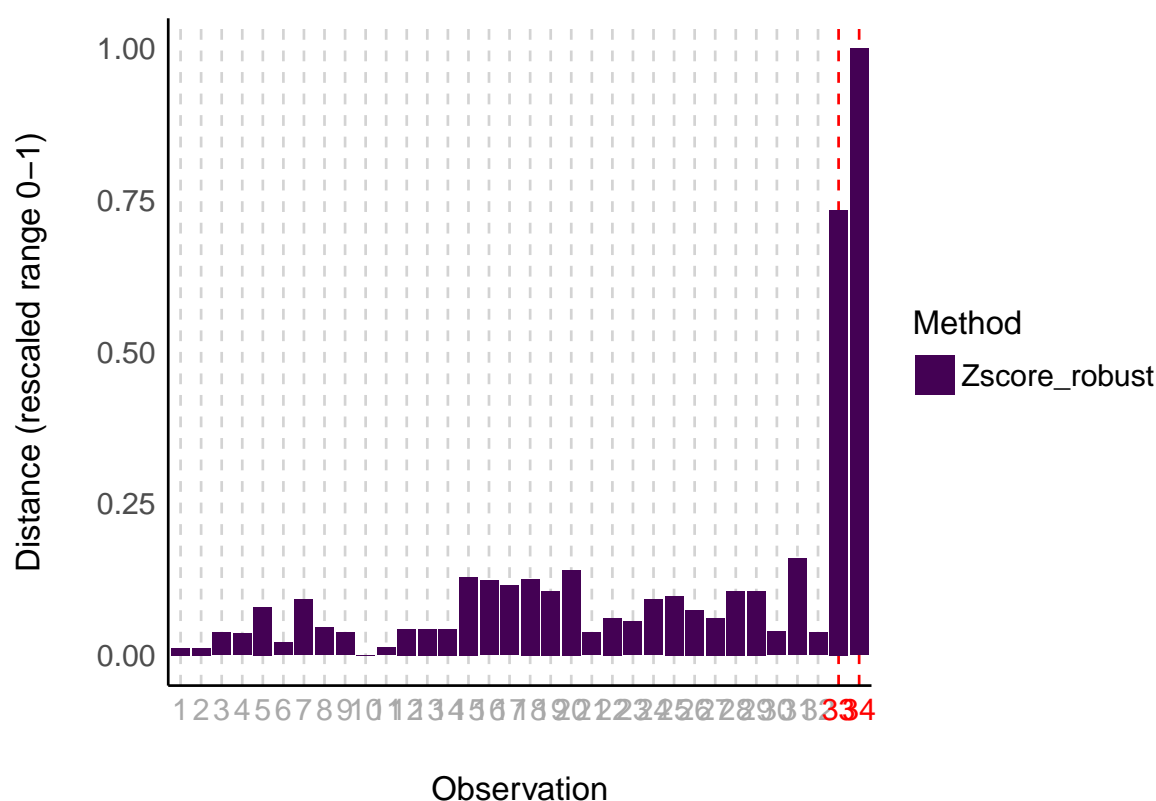


Figure 1. Visual depiction of outliers using the robust z-score method.

2.2. Multivariate Outliers

Univariate outliers can be useful when the focus is on a particular variable, for instance the reaction time, as extreme values might be indicative of inattention or non-task-related behavior².

However, in many scenarios, variables of a data set are not independent, and an abnormal observation will impact multiple dimensions. For instance, a participant giving random answers to a questionnaire. In this case, computing the z score for each of the questions might not lead to satisfactory results. Instead, one might want to look at these variables together.

One common approach for this is to compute multivariate distance metrics such as the Mahalanobis distance. Although the Mahalanobis distance is very popular, just like the regular z scores method, it is not robust and is heavily influenced by the outliers themselves. Therefore, for multivariate outliers, it is recommended to use the Minimum Covariance Determinant, a robust version of the Mahalanobis distance [MCD, 5,6].

In *performance*'s `check_outliers()`, one can use this approach with `method = "mcd"`.³

```
results <- check_outliers(data, method = "mcd")
results
```

```
#> 9 outliers detected: cases 7, 15, 16, 17, 24, 29, 31, 33, 34.
#> - Based on the following method and threshold: mcd (20).
#> - For variables: mpg, cyl, disp, hp.
```

² Note that they might not be the optimal way of treating reaction time outliers [7,8]

³ Our default threshold for the MCD method is defined by `stats::qchisq(p = 1 - 0.001, df = ncol(x))`, which again is an approximation of the critical value for $p < .001$ consistent with the thresholds of our other methods.

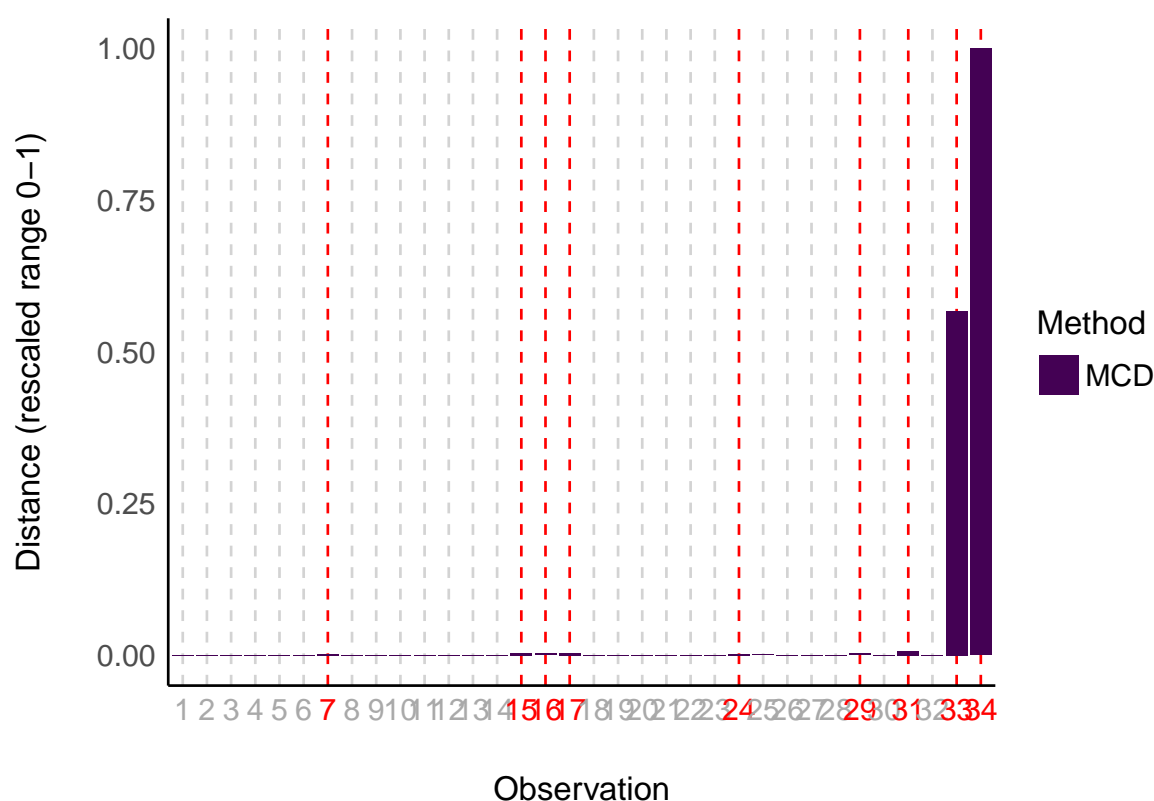


Figure 2. Visual depiction of outliers using the Minimum Covariance Determinant (MCD) method, a robust version of the Mahalanobis distance.

```
plot(results)
```

Other multivariate methods are available, such as another type of robust Mahalanobis distance that in this case relies on an orthogonalized Gnanadesikan-Kettenring pairwise estimator [9]. These methods are documented and described in the function's [help page](#).

2.3. Model-Based Outliers

Working with regression models creates the possibility of using model-based SOD methods. These methods rely on the concept of *leverage*, that is, how much influence a given observation can have on the model estimates. If few observations have a relatively strong leverage/influence on the model, one can suspect that the model's estimates are biased by these observations, in which case flagging them as outliers could prove helpful (see next section, "Handling Outliers").

In {performance}, two such model-based SOD methods are currently available: Cook's distance, for regular regression models, and Pareto, for Bayesian models. As such, `check_outliers()` can be applied directly on regression model objects, by simply specifying `method = "cook"` (or `method = "pareto"` for Bayesian models).⁴

```
model <- lm(disp ~ mpg * hp, data = data)
outliers <- check_outliers(model, method = "cook")
outliers
```

⁴ Our default threshold for the Cook method is defined by `stats::qf(0.5, ncol(x), nrow(x) - ncol(x))`, which again is an approximation of the critical value for $p < .001$ consistent with the thresholds of our other methods.

Influential Observations
Points should be inside the contour lines

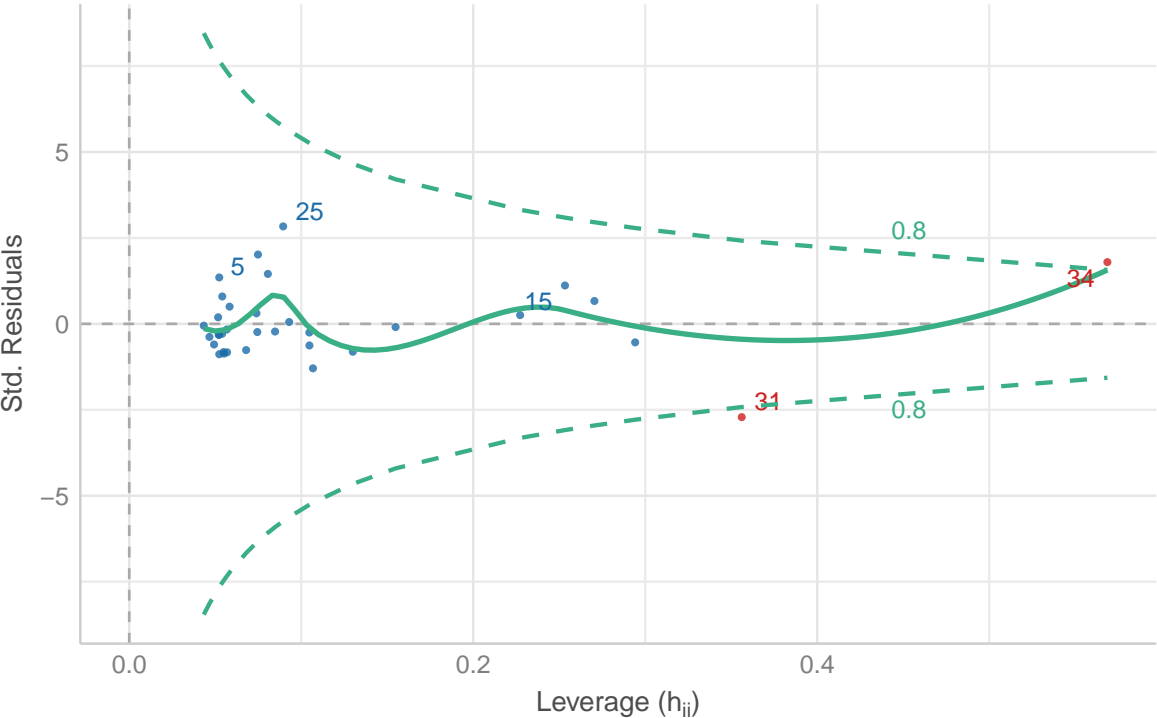


Figure 3. Visual depiction of outliers based on Cook’s distance (leverage and standardized residuals).

```
132 #> 2 outliers detected: cases 31, 34.  
133 #> - Based on the following method and threshold: cook (0.9).  
134 #> - For variable: (Whole model).
```

```
plot(outliers)
```

135 Table 1 below summarizes which methods to use in which cases, and with what threshold.

Table 1. Summary of Statistical Outlier Detection Methods Recommendations.

Statistical Test	Diagnosis Method	Recommended Threshold
Supported regression model	Model-based: Cook (or Pareto for Bayesian models)	$qf(0.5, ncol(x), nrow(x) - ncol(x))$ (or 0.7 for Pareto)
Structural Equation Modeling (or other unsupported model)	Multivariate: Minimum Covariance Determinant (MCD)	$qchisq(p = 1 - 0.001, df = ncol(x))$
Simple test with few variables (t test, correlation, etc.)	Univariate: robust z scores (MAD)	$qnorm(p = 1 - 0.001 / 2), \sim 3.29$

136 2.3.1. Cook’s Distance vs. MCD

137 Leys *et al.* [6] report a preference for the MCD method over Cook’s distance. This is because
138 Cook’s distance removes one observation at a time and checks its corresponding influence on the
139 model each time [10], and flags any observation that has a large influence. In the view of these authors,
140 when there are several outliers, the process of removing a single outlier at a time is problematic as the
141 model remains “contaminated” or influenced by other possible outliers in the model, rendering this
142 method suboptimal in the presence of multiple outliers.

However, distribution-based approaches are not a silver bullet either, and there are cases where the usage of methods agnostic to theoretical and statistical models of interest might be problematic. For example, a very tall person would be expected to also be much heavier than average, but that would still fit with the expected relationship between height and weight (i.e., it would be in line with a model such as $\text{weight} \sim \text{height}$). In contrast, using multivariate outlier detection methods there may flag this person as being an outlier—being unusual on two variables, height and weight—even though the pattern fits perfectly with our predictions.

Furthermore, unusual observations happen naturally: extreme observations are expected even when taken from a normal distribution. While statistical models can integrate this “expectation”, multivariate outlier methods might be too conservative, flagging too many observations despite belonging to the right generative process. For these reasons, we believe that model-based methods are still preferable to the MCD when using compatible regression models. Additionally, if the presence of multiple outliers is a significant concern, regression methods that are more robust to outliers should be considered—like t regression or quantile regression—as they render their precise identification less critical.

2.4. Multiple Methods

An alternative approach that is possible is to combine several methods, based on the assumption that different methods provide different angles of looking at the problem. By applying a variety of methods, one can hope to “triangulate” the true outliers (those consistently flagged by multiple methods) and thus attempt to minimize false positives.

In practice, this approach computes a composite outlier score, formed of the average of the binary (0 or 1) classification results of each method. It represents the probability that each observation is classified as an outlier by at least one method. The default decision rule classifies rows with composite outlier scores superior or equal to 0.5 as outlier observations (i.e., that were classified as outliers by at least half of the methods). In *performance*'s `check_outliers()`, one can use this approach by including all desired methods in the corresponding argument.

```
# Combine multiple methods
outliers <- check_outliers(
  data[1:5],
  method = c("zscore_robust", "iqr", "mcd", "ics")
)

outliers
```

```
#> 3 outliers detected: cases 31, 33, 34.
#> - Based on the following methods and thresholds: zscore_robust (3.09),
#>   iqr (2), mcd (20), ics (0.001).
#> - For variables: mpg, cyl, disp, hp.
#>
#> Note: Outliers were classified as such by at least half of the selected methods.
#>
#> -----
#>
#> The following observations were considered outliers for two or more
#>   variables by at least one of the selected methods:
#>
#>   Row n_Zscore_robust n_IQR          n_MCD          n_ICS
#> 1   33                2      2 (Multivariate) (Multivariate)
#> 2   34                2      2 (Multivariate) (Multivariate)
```

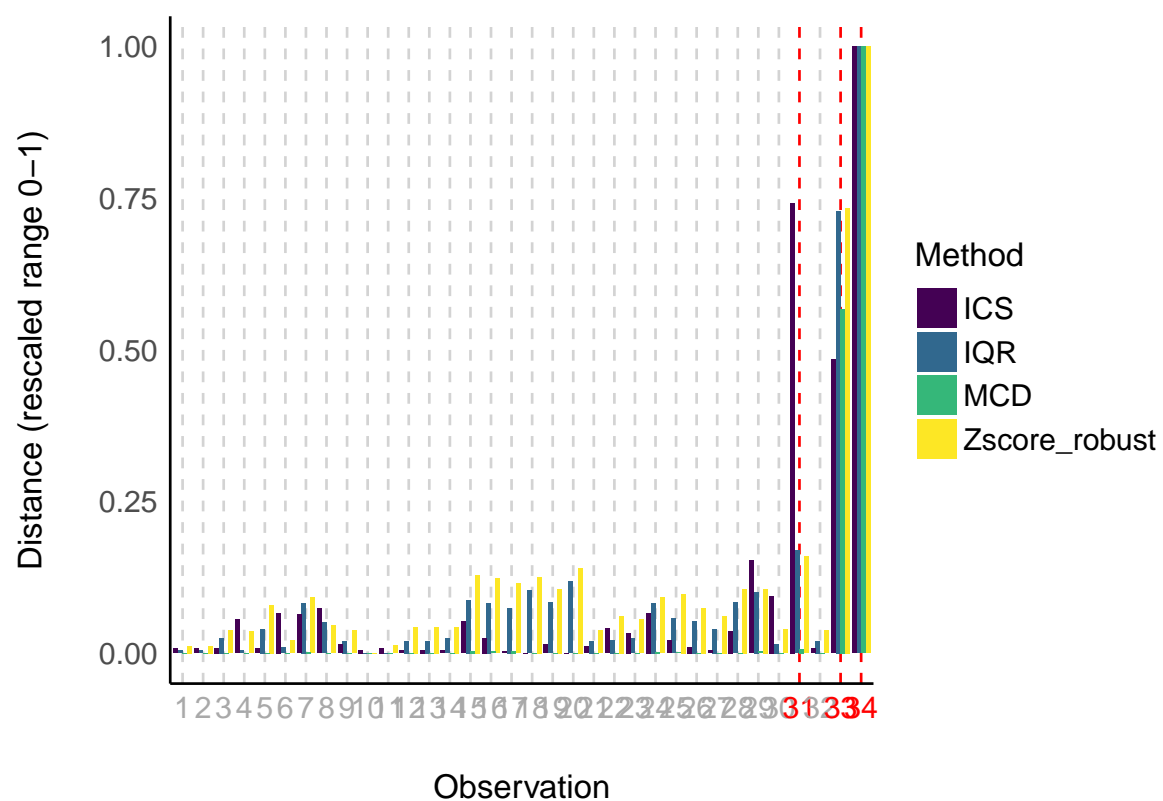


Figure 4. Visual depiction of outliers using several different statistical outlier detection methods.

```

184 #> 3  31          0      1 (Multivariate) (Multivariate)
185 #> 4   7          0      0 (Multivariate)              0
186 #> 5  15          0      0 (Multivariate)              0
187 #> 6  16          0      0 (Multivariate)              0
188 #> 7  17          0      0 (Multivariate)              0
189 #> 8  24          0      0 (Multivariate)              0
190 #> 9  29          0      0 (Multivariate)              0

```

```
plot(outliers)
```

Outliers (counts or per variables) for individual methods can then be obtained through attributes.
For example:

```
attributes(outliers)$outlier_var$iqr
```

```

193 #> $mpg
194 #>   Row Distance_IQR
195 #> 33  33      1.550881
196 #> 34  34      2.458544
197 #>
198 #> $cyl
199 #>   Row Distance_IQR
200 #> 33  33      5.294118
201 #> 34  34      7.205882
202 #>
203 #> $hp

```



```

204 #>      Row Distance_IQR
205 #> 31   31         1.348181

```

206 An example sentence for reporting the usage of the composite method could be:

207 Based on a composite outlier score [see the ‘`check_outliers()`’ function in the ‘`performance`’ R
 208 package, 3] obtained via the joint application of multiple outliers detection algorithms [(a) median
 209 absolute deviation (MAD)-based robust z scores, 2; (b) interquartile range (IQR), (c) Mahalanobis
 210 minimum covariance determinant (MCD), 5; and (d) invariant coordinate selection (ICS), 11], we
 211 excluded three participants that were classified as outliers by at least half of the methods used.

212 3. Handling Outliers

213 The above section demonstrated how to identify outliers using the `check_outliers()` function
 214 in the *{performance}* package. But what should we do with these outliers once identified? Although
 215 it is common to automatically discard any observation that has been marked as “an outlier” as if it
 216 might infect the rest of the data with its statistical ailment, we believe that the use of SOD methods is
 217 but one step in the get-to-know-your-data pipeline; a researcher or analyst’s *domain knowledge* must
 218 be involved in the decision of how to deal with observations marked as outliers by means of SOD.
 219 Indeed, automatic tools can help detect outliers, but they are nowhere near perfect. Although they can
 220 be useful to flag suspect data, they can have misses and false alarms, and they cannot replace human
 221 eyes and proper vigilance from the researcher. If you do end up manually inspecting your data for
 222 outliers, it can be helpful to think of outliers as belonging to different types of outliers, or categories,
 223 which can help decide what to do with a given outlier.

224 3.1. Error, Interesting, and Random Outliers

225 Leys *et al.* [5] distinguish between error outliers, interesting outliers, and random outliers. *Error*
 226 *outliers* are likely due to human error and should be corrected before data analysis or outright removed
 227 since they are invalid observations. *Interesting outliers* are not due to technical error and may be of
 228 theoretical interest; it might thus be relevant to investigate them further even though they should be
 229 removed from the current analysis of interest. *Random outliers* are assumed to be due to chance alone
 230 and to belong to the correct distribution and, therefore, should be retained.

231 It is recommended to *keep* observations which are expected to be part of the distribution of interest,
 232 even if they are outliers [5]. However, if it is suspected that the outliers belong to an alternative
 233 distribution, then those observations could have a large impact on the results and call into question
 234 their robustness, especially if significance is conditional on their inclusion.

235 On the other hand, there are also outliers that cannot be detected by statistical tools, but should
 236 be found and removed. For example, if we are studying the effects of X on Y among teenagers and we
 237 have one observation from a 20-year-old, this observation might not be a *statistical outlier*, but it is an
 238 outlier in the *context* of our research, and should be discarded to allow for valid inferences.

239 3.2. Winsorization

240 *Removing* outliers can in this case be a valid strategy, and ideally one would report results with
 241 and without outliers to see the extent of their impact on results. This approach however can reduce
 242 statistical power. Therefore, some propose a *recoding* approach, namely, winsorization: bringing
 243 outliers back within acceptable limits [e.g., 3 MADs, 12]. However, if possible, it is recommended
 244 to collect enough data so that even after removing outliers, there is still sufficient statistical power
 245 without having to resort to winsorization [5].

246 The *easystats* ecosystem makes it easy to incorporate this step into your workflow through the
 247 `winsorize()` function of the *{datawizard}* package [13]. This procedure will bring back univariate
 248 outliers within the limits of ‘acceptable’ values, based either on the percentile, the z score, or its robust
 249 alternative based on the MAD.

```
data[33:34, ] # See outliers rows
```

```
250 #>      car mpg cyl disp hp
251 #> 33   33  42  42   42 42
252 #> 34   34  55  55   55 55
```

```
# Winsorizing using the MAD
library(datawizard)
winsorized_data <- winsorize(data, method = "zscore", robust = TRUE, threshold = 3)

# Values > +/- MAD have been winsorized
winsorized_data[33:34, ]
```

```
253 #>      car      mpg      cyl disp hp
254 #> 33   33 37.68598 14.8956   42 42
255 #> 34   34 37.68598 14.8956   55 55
```

256 3.3. The Importance of Transparency

257 Once again, it is a critical part of a sound outlier treatment that regardless of which SOD method
 258 used, it should be reported in a reproducible manner. Ideally, the handling of outliers should be
 259 specified *a priori* with as much detail as possible, and preregistered, to limit researchers' degrees
 260 of freedom and therefore risks of false positives [5]. This is especially true given that interesting
 261 outliers and random outliers are often times hard to distinguish in practice. Thus, researchers should
 262 always prioritize transparency and report all of the following information: (a) how many outliers
 263 were identified; (b) according to which method and criteria, (c) using which function of which R
 264 package (if applicable), and (d) how they were handled (excluded or winsorized, if the latter, using
 265 what threshold). If at all possible, (e) the corresponding code script along with the data should be
 266 shared on a public repository like the Open Science Framework (OSF), so that the exclusion criteria
 267 can be reproduced precisely.

268 4. Conclusion

269 In this paper, we have showed how to investigate outliers using the `check_outliers()` function
 270 of the *{performance}* package while following current good practices. However, best practice for outlier
 271 treatment does not stop at using appropriate statistical algorithms, but entails respecting existing
 272 recommendations, such as preregistration, reproducibility, consistency, transparency, and justification.
 273 Ideally, one would additionally also report the package, function, and threshold used (linking to the
 274 full code when possible). We hope that this paper and the accompanying `check_outlier()` function
 275 of *easystats* will help researchers engage in good research practices while providing a smooth outlier
 276 detection experience.

277 **Acknowledgments:** *{performance}* is part of the collaborative *easystats* ecosystem [4]. Thus, we thank all [members](#)
 278 [of easystats](#), contributors, and users alike.

279 **Author Contributions:** R.T. drafted the paper; all authors contributed to both the writing of the paper and the
 280 conception of the software.

281 **Conflicts of Interest:** The authors declare no conflict of interest.

282 Abbreviations

283 The following abbreviations are used in this manuscript:
 284

SOD	Statistical outlier detection
SEM	Structural equation modelling
SD	Standard deviation
MAD	Median absolute deviation
IQR	Interquartile range
HDI	Highest density interval
BCI	Bias corrected and accelerated interval
MCD	Minimum covariance determinant
ICS	invariant coordinate selection
OSF	Open Science Framework

References

- Simmons, J.P.; Nelson, L.D.; Simonsohn, U. False-Positive Psychology: Undisclosed Flexibility in Data Collection and Analysis Allows Presenting Anything as Significant. *Psychological Science* **2011**, *22*, 1359–1366. <https://doi.org/10.1177/0956797611417632>.
- Leys, C.; Ley, C.; Klein, O.; Bernard, P.; Licata, L. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology* **2013**, *49*, 764–766. <https://doi.org/10.1016/j.jesp.2013.03.013>.
- Lüdtke, D.; Ben-Shachar, M.S.; Patil, I.; Waggoner, P.; Makowski, D. performance: An R package for assessment, comparison and testing of statistical models. *Journal of Open Source Software* **2021**, *6*, 3139. <https://doi.org/10.21105/joss.03139>.
- Lüdtke, D.; Ben-Shachar, M.S.; Patil, I.; Wiernik, B.M.; Bacher, E.; Thériault, R.; Makowski, D. easystats: Framework for Easy Statistical Modeling, Visualization, and Reporting. *CRAN* **2022**. R package.
- Leys, C.; Delacre, M.; Mora, Y.L.; Lakens, D.; Ley, C. How to Classify, Detect, and Manage Univariate and Multivariate Outliers, With Emphasis on Pre-Registration. *International Review of Social Psychology* **2019**. <https://doi.org/10.5334/irsp.289>.
- Leys, C.; Klein, O.; Dominicy, Y.; Ley, C. Detecting multivariate outliers: Use a robust variant of the Mahalanobis distance. *Journal of Experimental Social Psychology* **2018**, *74*, 150–156. <https://doi.org/10.1016/j.jesp.2017.09.011>.
- Ratcliff, R. Methods for dealing with reaction time outliers. *Psychological bulletin* **1993**, *114*, 510. <https://doi.org/10.1037/0033-2909.114.3.510>.
- Van Zandt, T.; Ratcliff, R. Statistical mimicking of reaction time data: Single-process models, parameter variability, and mixtures. *Psychonomic Bulletin & Review* **1995**, *2*, 20–54. <https://doi.org/10.3758/BF03214411>.
- Gnanadesikan, R.; Kettenring, J.R. Robust estimates, residuals, and outlier detection with multiresponse data. *Biometrics* **1972**, pp. 81–124.
- Cook, R.D. Detection of Influential Observation in Linear Regression. *Technometrics* **1977**, *19*, 15–18. <https://doi.org/10.1080/00401706.1977.10489493>.
- Archimbaud, A.; Nordhausen, K.; Ruiz-Gazen, A. ICS for multivariate outlier detection with application to quality control. *Computational Statistics and Data Analysis* **2018**, *128*, 184–199. <https://doi.org/10.1016/j.csda.2018.06.011>.
- Tukey, J.W.; McLaughlin, D.H. Less vulnerable confidence and significance procedures for location based on a single sample: Trimming/Winsorization 1. *Sankhyā: The Indian Journal of Statistics, Series A* **1963**, pp. 331–352.
- Patil, I.; Makowski, D.; Ben-Shachar, M.S.; Wiernik, B.M.; Bacher, E.; Lüdtke, D. datawizard: An R package for easy data preparation and statistical transformations. *Journal of Open Source Software* **2022**, *7*, 4684. <https://doi.org/10.21105/joss.04684>.