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

(Alt title:) Detecting Statistical Outliers: Univariate, Multivariate, and Model-Based

## Abstract

xyz

## Introduction

The improper handling of outliers can substantially affect statistical model estimations, thus contributing to false positives (Simmons et al., 2011) but almost certainly to false negatives as well. It is thus essential to address this problem in a thoughtful manner. Fortunately, guidelines exist in this regard. Yet, especially in the field of psychology, many researchers still do not treat outliers in a consistent manner or do so using inappropriate strategies (Leys et al., 2013; Simmons et al., 2011).

One possible reason is that researchers do not know of existing recommendations or currently available software options for their implementation. In this paper, we show how to follow current recommendations for the detection of outliers using R and {performance} package (Lüdecke et al., 2021) from *easystats* ecosystem.

## 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 influenced by outliers and they assume a normal distribution (Leys et al., 2013, 2019). Therefore, current guidelines [ref?] recommend using robust methods to identify outliers, such as those relying on the median as opposed to the mean.

Nonetheless, which exact outlier method to use depends on several factors, like the statistical test of interest. When using a regression model, for example, the most relevant information will be regarding observations that do not fit well with the model, known as model-based outliers. When no method is readily available to detect model-based outliers, such as for structural equation modelling (SEM), looking for multivariate outliers maybe relevant. Finally, for simple tests (t-tests or correlations) that compare values of the same variable, it makes sense to check for univariate outliers. In the following section, we will go through each of the mentioned methods.

### Univariate Outliers

For univariate outliers, it is recommended to use the median along the Median Absolute Deviation (MAD), which is more robust than the interquartile range or the mean and its standard deviation (Leys et al., 2013, 2019). The MAD can be calculated as follow:

$$MAD = bM_i(|x_i - M_j(x_j)|)$$

In {performance}'s `check_outliers()`, one can use this approach with `method = "zscore_robust"`. Although Leys et al. (2013) suggests a default threshold of 2.5 and Leys et al. (2019) a threshold of 3, for consistency with other outlier detection methods, {performance} uses a less conservative threshold of 3.09 by default. That is, data points will be flagged as outliers if they go beyond +/- 3.09 MAD. Users can adjust this threshold using `threshold` argument.

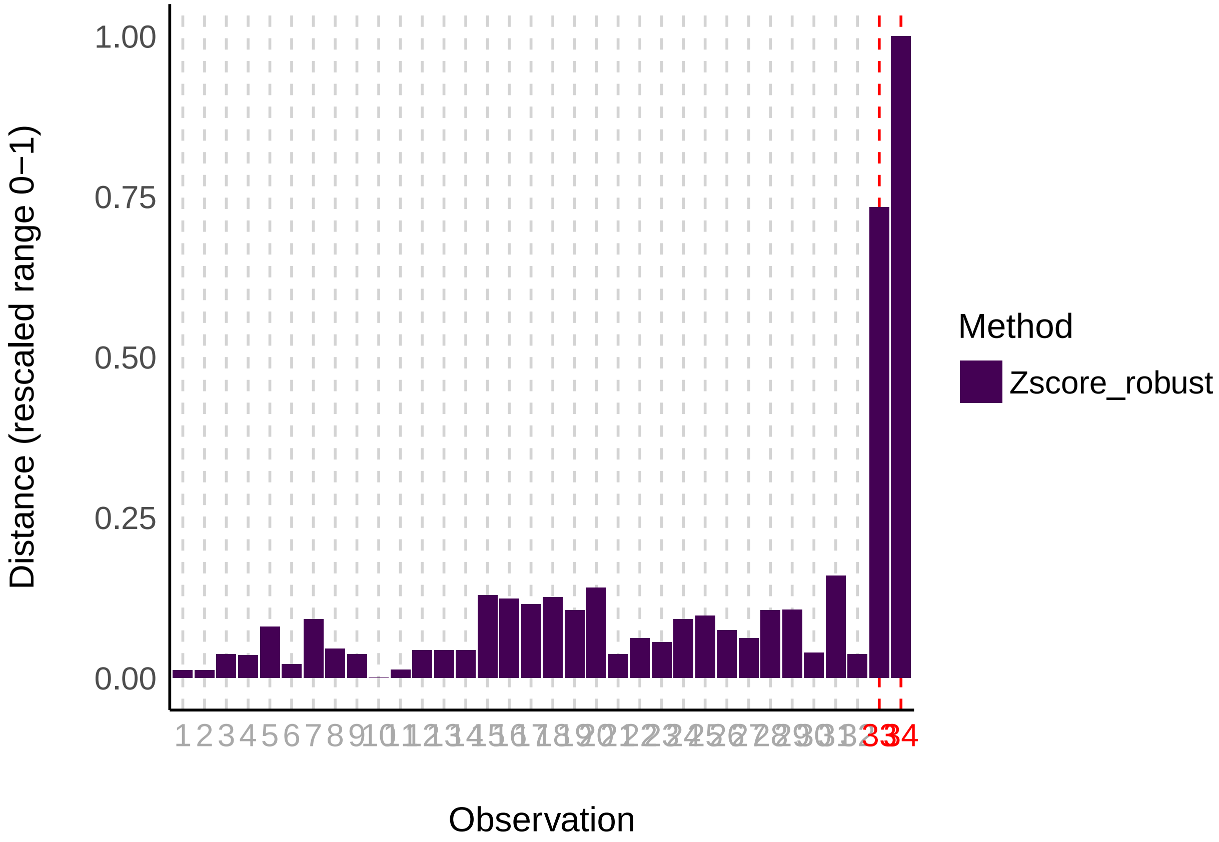
Example:

```
library(performance)

# create some fake outliers and an ID column
data <- rbind(mtcars[1:4], 42, 55)
data <- cbind(car = row.names(data), data)

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

```
#> 2 outliers detected: cases 33, 34.
#> - Based on the following method and threshold: zscore_robust (3.09).
#> - For variables: mpg, cyl, disp, hp.
#>
#> -----------------------------------------------------------------------------
#> The following observations were considered outliers for two or more variables
#> by at least one of the selected methods:
#>
#>   Row car n_Zscore_robust
#> 1  33  33               2
#> 2  34  34               2
#>
#> -----------------------------------------------------------------------------
#> Outliers per variable (zscore_robust):
#>
#> $mpg
#>    Row car Distance_Zscore_robust
#> 33  33  33               3.709699
#> 34  34  34               5.848328
#>
#> $cyl
#>    Row car Distance_Zscore_robust
#> 33  33  33               12.14083
#> 34  34  34               16.52502
```

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

```
library(see)
plot(x)
```



### Multivariate Outliers

For multivariate outliers, it is recommended to use the Minimum Covariance Determinant, a robust version of the Mahalanobis distance (MCD, Leys et al., 2019). In {performance}'s `check_outliers()`, one can use this approach with `method = "mcd"`. The MCD can be calculated as follow (Hubert et al., 2018):

$$MATTHS^{andB}_{gohere}$$

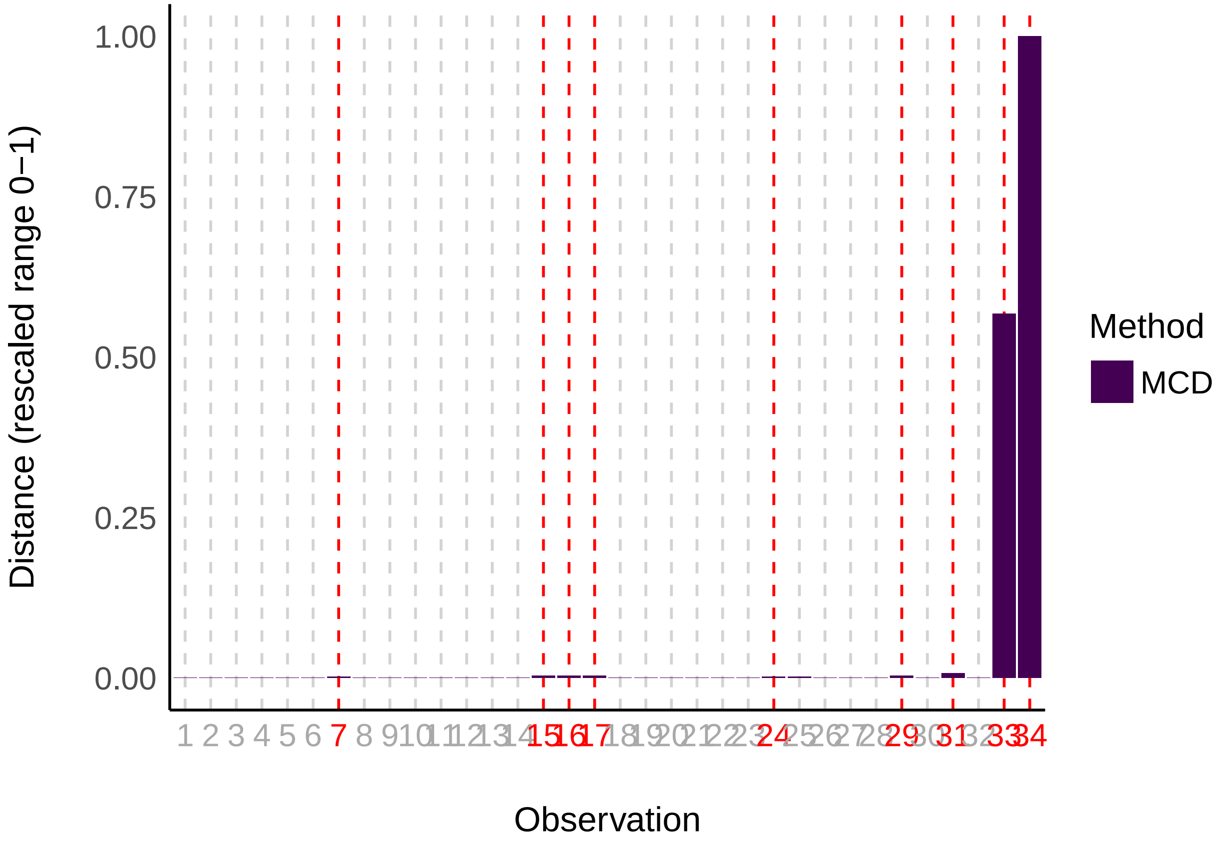
**(mattansb? can you add some maths here; perhaps from Hubert et al., 2018? https://doi.org/10.1002/wics.1421)**

Example:

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

```
#> 9 outliers detected: cases 7, 15, 16, 17, 24, 29, 31, 33, 34.
#> - Based on the following method and threshold: mcd (18.47).
#> - For variables: mpg, cyl, disp, hp.
```

```
plot(x)
```



## Model-Based Outliers

[FIND REFERENCES TO SUPPORT MODEL-BASED OUTLIER DETECTION ABOVE ALL OTHER METHODS!!]

When working with regression models, model-based outliers can be detected using `check_outliers()` by specifying `method = "cook"` (or `method = "pareto"` for Bayesian models). The Cook's distance can be calculated as follow:

$$MATTHS^{andB}_{gohere}$$

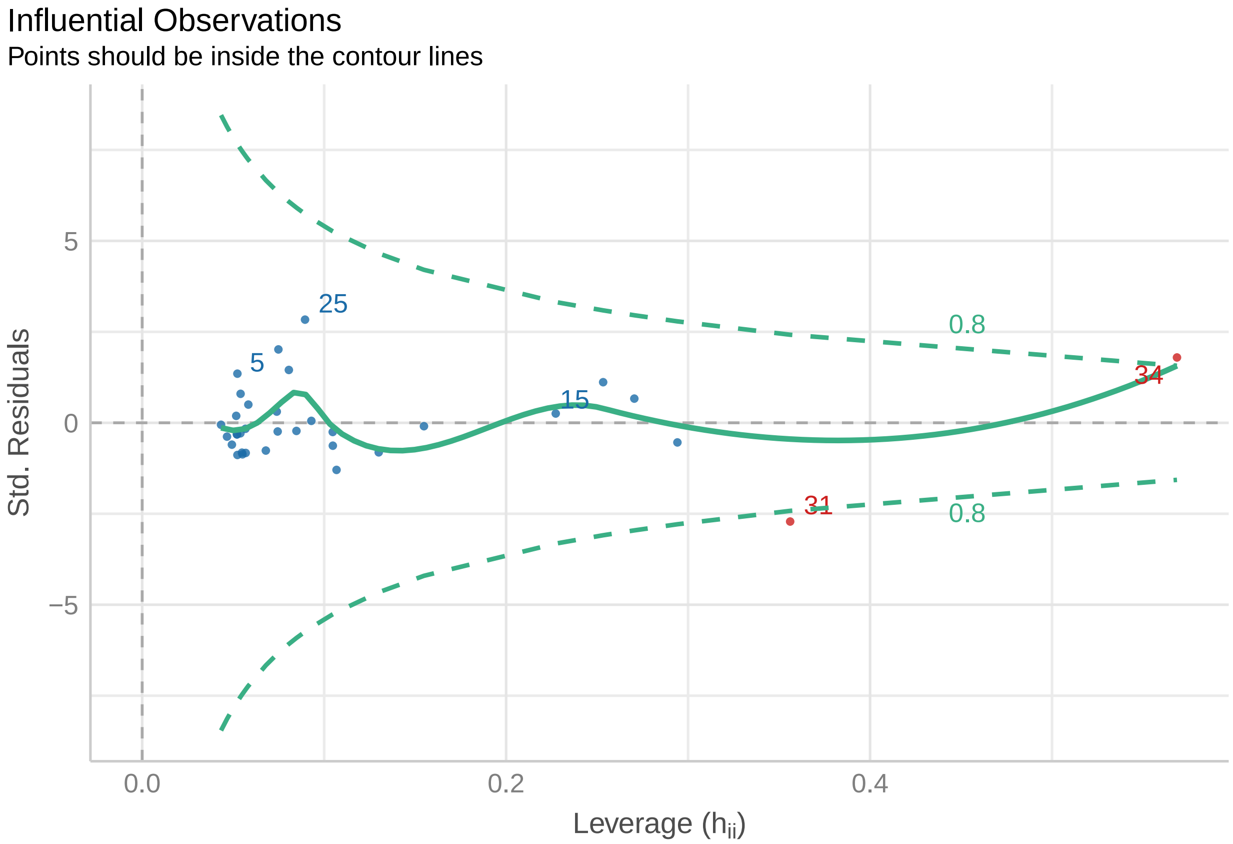
**(mattansb? can you add some maths here?)**

Example:

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

```
#> 2 outliers detected: cases 31, 34.
#> - Based on the following method and threshold: cook (0.86).
#> - For variable: (Whole model).
```

```
plot(x)
```



## Multiple methods

An alternative approach suggested by easystats is to combine several methods. This approach computes a composite outlier score, formed of the average of the binary (0 or 1) 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.

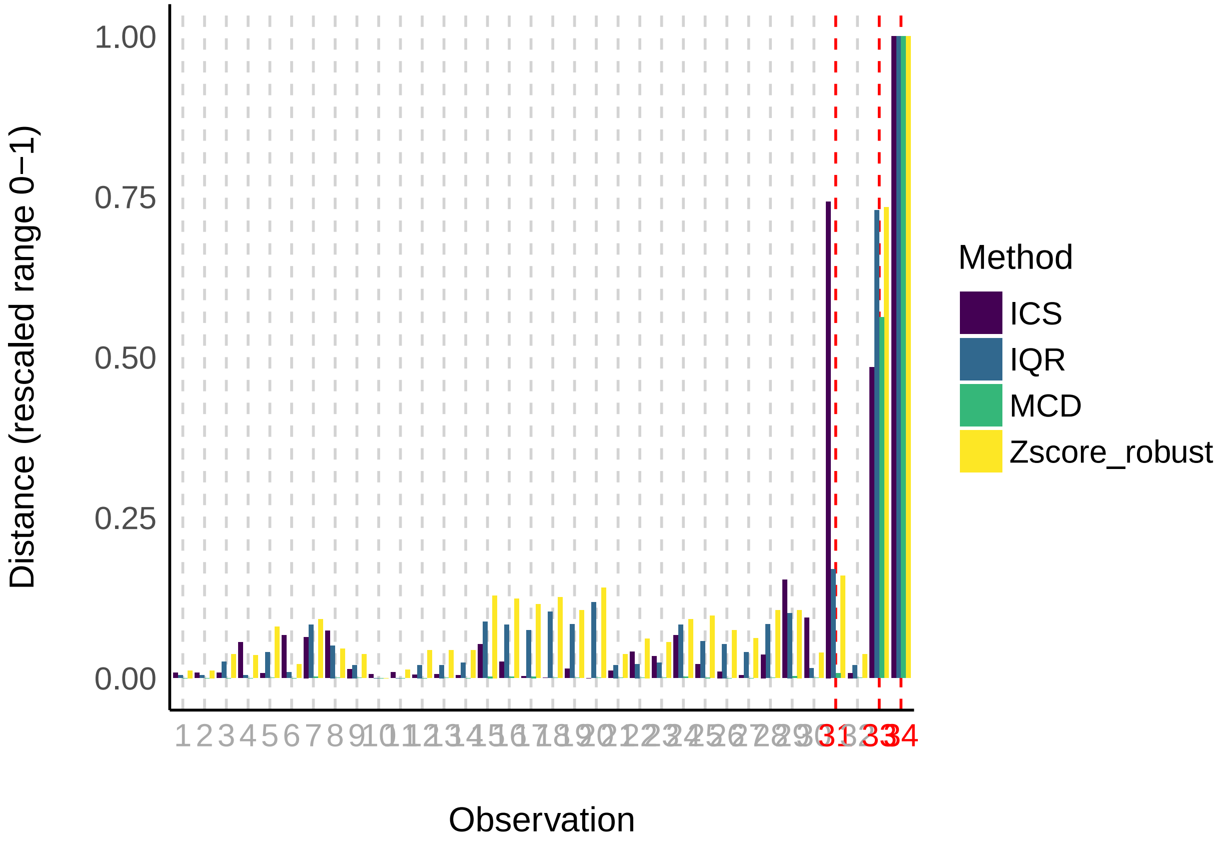
Example:

```
x <- check_outliers(
  data,
  method = c("zscore_robust", "iqr", "mcd", "ics"),
  ID = "car"
)
x
```

```
#> 3 outliers detected: cases 31, 33, 34.
#> - Based on the following methods and thresholds: zscore_robust (3.09),
#>   iqr (1.7), mcd (18.47), ics (0).
#> - 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                car n_Zscore_robust n_IQR          n_MCD          n_ICS
#> 1  33                 33               2     2 (Multivariate) (Multivariate)
#> 2  34                 34               2     2 (Multivariate) (Multivariate)
#> 3  31      Maserati Bora               0     1 (Multivariate) (Multivariate)
#> 4   7         Duster 360               0     0 (Multivariate)              0
#> 5  15  Cadillac Fleetwood              0     0 (Multivariate)              0
#> 6  16 Lincoln Continental              0     0 (Multivariate)              0
#> 7  17   Chrysler Imperial              0     0 (Multivariate)              0
#> 8  24         Camaro Z28               0     0 (Multivariate)              0
#> 9  29      Ford Pantera L              0     0 (Multivariate)              0
```

```
plot(x)
```



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

> Based on a composite outlier score (see the 'check_outliers' function in the 'performance' R package, Lüdecke et al., 2021) obtained via the joint application of multiple outliers detection algorithms ((a) median absolute deviation (MAD)-based robust z-scores, Leys et al., 2013; (b) interquartile range (IQR), (c) Mahalanobis minimum covariance determinant (MCD), Leys et al.,

2019; and (d) invariant coordinate selection (ICS), Archimbaud et al., 2018), we excluded three participants that were classified as outliers by at least half of the methods used.

# Handling Outliers

We have at this point demonstrated how to identify outliers. But what should we do with these outliers once identified? There is no one-size-fits-all rule, as this depends on several factors. For example, Leys et al. (2019) distinguishes between error outliers, interesting outliers, and random outliers.

*Error outliers* are likely due to human error and should be corrected before data analysis or outright removed since they are invalid observations. *Interesting outliers* are not due to technical error and may be of theoretical interest; it might thus be relevant to investigate them further even though they should be removed from the current analysis of interest. *Random outliers* are assumed to be due to chance alone and to belong to the correct distribution and, therefore, should be retained.

It is recommended to *keep* observations which are expected to be part of the distribution of interest, even if they are outliers (Leys et al., 2019). However, if it is suspected that the outliers belong to an alternative distribution, then those observations could have a large impact on the results and call into question their robustness, especially if significance is conditional on their inclusion.

*Removing* outliers can in this case be a valid strategy, and ideally one would report results with and without outliers to see the extent of their impact on results (assuming the study was not preregistered with a prespecified outlier treatment plan). This approach however can reduce statistical power. Therefore, some propose a *recoding* approach; i.e., winsorizing (Tukey & McLaughlin, 1963) outliers to bring them back within acceptable limits (e.g., 3 MAD). However, if possible, it is recommended to collect enough data so that even after removing outliers, there is still sufficient statistical power without having to resort to winsorization (Leys et al., 2019).

We note that no matter which outlier method you use, the handling of outliers should be specified *a priori* with as much detail as possible, and ideally preregistered, to limit researchers' degrees of freedom and therefore risks of false positives (Leys et al., 2019). This is especially true given that interesting outliers and random outliers are oftentimes hard to distinguish in practice. Thus, researchers should always prioritize transparency and report exactly how many and which outliers were handled, and share code so that the exclusion criteria can be reproduced.

## Winsorization

(Should we add a section on winsorization since we talk about this in the intro/recommendations?)

```
data[33:34, ]
```

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

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

winsorized.data[33:34, ]
```

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

# Conclusion

In this paper, we have showed how to investigate outliers using the `check_outliers()` function of the {performance} package while following current good practices. We note that in addition to using the current functions and respecting existing recommendations, it is also important to pre-specify your plans to manage outliers, such as with preregistration, or at the very least justify your choices and stay consistent. Ideally, one would additionally also report the package, function, and threshold used (ideally linking to the full code). We hope that this paper will help more researchers engage in good research practices while providing a smooth experience.

# Acknowledgments

{performance} is part of the collaborative *easystats* ecosystem. Thus, we thank the members of easystats as well as the users.

# References

Archimbaud, A., Nordhausen, K., & Ruiz-Gazen, A. (2018). ICS for multivariate outlier detection with application to quality control. *Computational Statistics and Data Analysis*, *128*, 184–199. https://doi.org/10.1016/j.csda.2018.06.011

Hubert, M., Debruyne, M., & Rousseeuw, P. J. (2018). Minimum covariance determinant and extensions. *Wiley Interdisciplinary Reviews: Computational Statistics*, *10*(3), e1421. https://doi.org/10.1002/wics.1421

Leys, C., Delacre, M., Mora, Y. L., Lakens, D., & Ley, C. (2019). How to classify, detect, and manage univariate and multivariate outliers, with emphasis on pre-registration. *International Review of Social Psychology*. https://doi.org/10.5334/irsp.289

Leys, C., Ley, C., Klein, O., Bernard, P., & Licata, L. (2013). Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology*, *49*(4), 764–766. https://doi.org/10.1016/j.jesp.2013.03.013

Lüdecke, D., Ben-Shachar, M. S., Patil, I., Waggoner, P., & Makowski, D. (2021). performance: An r package for assessment, comparison and testing of statistical models. *Journal of Open Source Software*, *6*(60), 3139. https://doi.org/10.21105/joss.03139

Simmons, J. P., Nelson, L. D., & Simonsohn, U. (2011). False-positive psychology: Undisclosed flexibility in data collection and analysis allows presenting anything as significant. *Psychological Science*, *22*(11), 1359–1366. https://doi.org/10.1177/0956797611417632

Tukey, J. W., & McLaughlin, D. H. (1963). Less vulnerable confidence and significance procedures for location based on a single sample: Trimming/winsorization 1. *Sankhyā: The Indian Journal of Statistics, Series A*, 331–352.