

1 see: An R Package for Visualizing Statistical Models

2 **Daniel Lüdecke¹, Indrajeet Patil², Mattan S. Ben-Shachar³, Brenton**
3 **M. Wiernik⁴, Philip Waggoner⁵, and Dominique Makowski⁶**

4 **1** University Medical Center Hamburg-Eppendorf, Germany **2** Center for Humans and Machines,
5 Max Planck Institute for Human Development, Berlin, Germany **3** Ben-Gurion University of the
6 Negev, Israel **4** Department of Psychology, University of South Florida, USA **5** University of
7 Chicago, USA **6** Nanyang Technological University, Singapore

DOI: [10.21105/joss.03393](https://doi.org/10.21105/joss.03393)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Editor: [Frederick Boehm](#) ↗

Reviewers:

- [@MatthewSmith430](#)
- [@jakobbossek](#)

Submitted: 15 June 2021

Published: 22 July 2021

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

8 Summary

9 The *see* package is embedded in the *easystats* ecosystem, a collection of R packages that
10 operate in synergy to provide a consistent and intuitive syntax when working with statistical
11 models in the R programming language ([R Core Team, 2021](#)). Most *easystats* packages
12 return comprehensive numeric summaries of model parameters and performance. The *see*
13 package complements these numeric summaries with a host of functions and tools to produce
14 a range of publication-ready visualizations for model parameters, predictions, and performance
15 diagnostics. As a core pillar of *easystats*, the *see* package helps users to utilize visualization
16 for more informative, communicable, and well-rounded scientific reporting.

17 Statement of Need

18 The grammar of graphics ([Wilkinson, 2012](#)), largely due to its implementation in the *ggplot2*
19 package ([Wickham, 2016](#)), has become the dominant approach to visualization in R. Building
20 a model visualization with *ggplot2* is somewhat disconnected from the model fitting and
21 evaluation process. Generally, this process entails:

- 22 1. Fitting a model.
 - 23 2. Extracting desired results from the model (e.g., model parameters and intervals, model
24 predictions, diagnostic statistics) and arranging them into a dataframe.
 - 25 3. Passing the results dataframe to `ggplot()` and specifying the graphical parameters.
- 26 For example:

```
library(ggplot2)

# step-1
model <- lm(mpg ~ factor(cyl) * wt, data = mtcars)

# step-2
results <- fortify(model)

# step-3
ggplot(results) +
  geom_point(aes(x = wt, y = mpg, color = factor(cyl))) +
  geom_line(aes(x = wt, y = .fitted, color = `factor(cyl)`))
```

A number of packages have been developed to extend *ggplot2* and assist with model visualization.¹ Some of these packages provide functions for additional geoms, annotations, or common visualization types without linking them to a specific statistical analysis or fundamentally changing the *ggplot2* workflow (e.g., *ggrepel*, *ggalluvial*, *ggridges*, *ggdist*, *ggpubr*, etc.). Other *ggplot2* extensions provide functions to generate publication-ready visualizations for specific types of models (e.g., *metaviz*, *tidymv*, *sjPlot*, *survminer*). For example, the *ggstatsplot* package (Patil, 2021) offers visualizations for statistical analysis of one-way factorial designs, the *plotmm* package (Waggoner, 2020) supports specific types of mixture model objects. The *fortify()* function from *ggfortify* package (Horikoshi & Tang, 2018) does offer a unified plotting framework for a wide range of statistical models, although it is not as comprehensive as the *see* package because the *easystats* ecosystem covers a much larger collection of statistical models.

The aim of the *see* package is to produce visualizations for a wide variety of models and statistical analyses in a way that is tightly linked with the model fitting process and requires minimal interruption of users' workflow. *see* accomplishes this aim by providing a single *plot()* method for objects created by the other *easystats* packages, such as *parameters* tables, *modelbased* predictions, *performance* diagnostic tests, *correlation* matrices, and so on. The *easystats* packages compute numeric results for a wide range of statistical models, and the *see* package acts as a visual support to the entire *easystats* ecosystem. As such, visualizations corresponding to all stages of statistical analysis, from model fitting to diagnostics to reporting, can be easily created using *see*. *see* plots are compatible with other *ggplot2* functions for further customization (e.g., *labs()* for a plot title). In addition, *see* provides several aesthetic utilities to embellish both *easystats* plots and other *ggplot2* plots. The result is a package that minimizes the barrier to producing high-quality statistical visualizations in R.

The central goal of *easystats* is to make the task of doing statistics in R as easy as possible. This goal is realized through intuitive and consistent syntax, consistent and transparent argument names, comprehensive documentation, informative warnings and error messages, and smart functions with sensible default parameter values. The *see* package follows this philosophy by using a single access point—the generic *plot()* method—for visualization of all manner of statistical results supported by *easystats*.

Features

Below we present one or two plotting methods for each *easystats* package, but many other methods are available. Interested readers are encouraged to explore the range of examples on the package website, <https://easystats.github.io/see/>.

Themes and Palettes

The package includes different *ggplot2* themes that one can set for each plot, or generally as shown below:

```
ggplot2::theme_set(see::theme_modern())
```

The package provides also color palettes, such as *scale_color_material* or *scale_color_flat* for material and flat design colors (<https://www.materialui.co/colors>), respectively.

¹For a sampling of these packages, visit <https://exts.ggplot2.tidyverse.org/gallery/>

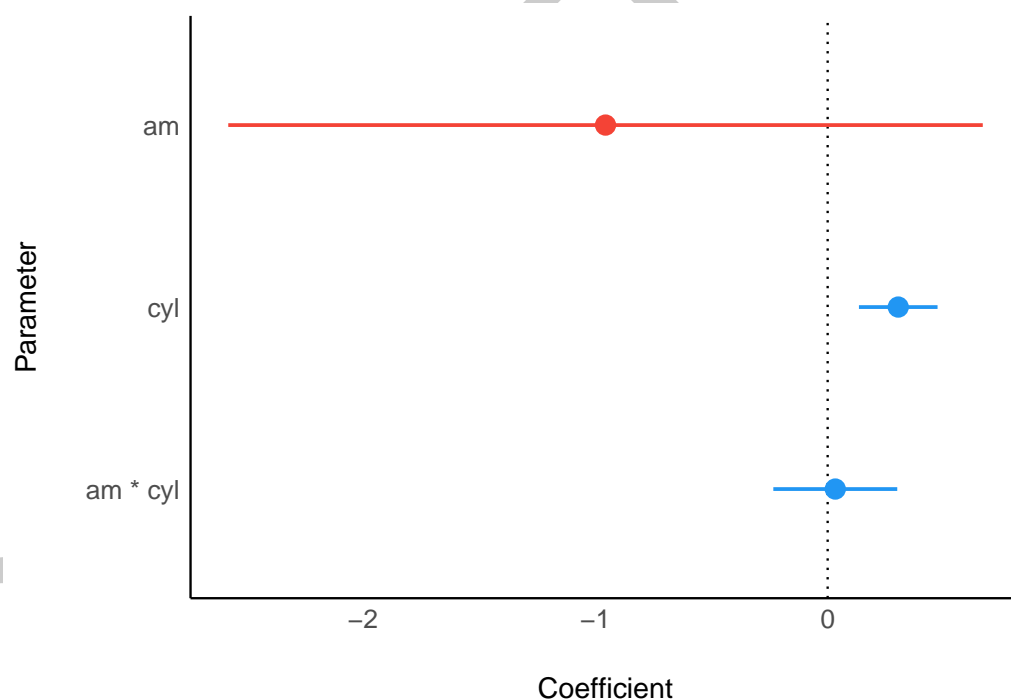
66 Visualizing Model Parameters

67 The *parameters* package converts summaries of regression model objects into dataframes
68 (Lüdecke et al., 2020). The *see* package can take this transformed object and, for example,
69 create a dot-and-whisker plot for the extracted regression estimates simply by passing the
70 *parameters* class object to `plot()`.

```
library(parameters)
library(see)
library(ggplot2)

model <- lm(wt ~ am * cyl, data = mtcars)

plot(parameters(model))
```



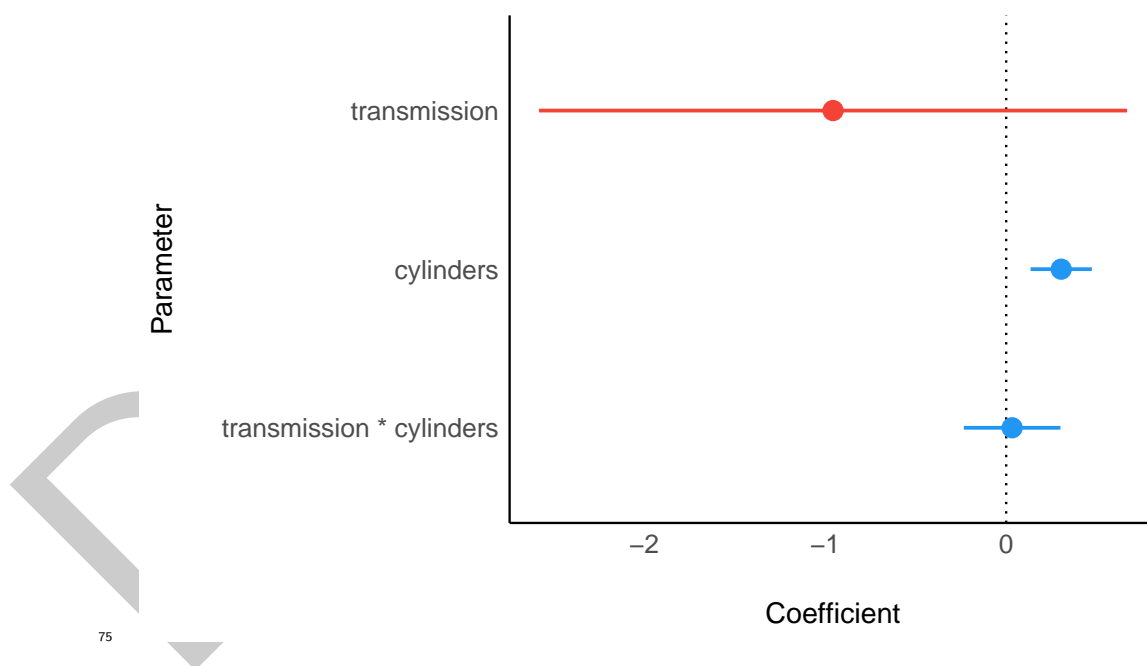
72 As see outputs objects of class `ggplot`, *ggplot2* functions can be added as layers to the plot
73 the same as with all other *ggplot2* visualizations. For example, we might add a title using
74 `labs()` from *ggplot2*.

```
library(parameters)
library(see)

model <- lm(wt ~ am * cyl, data = mtcars)

# changing title and axis labels using ggplot2 functions
plot(parameters(model)) +
  labs(title = "A Dot-and-Whisker Plot") +
  scale_y_discrete(labels = c(
    "transmission * cylinders",
    "cylinders",
    "transmission"
  ))
))
```

A Dot-and-Whisker Plot



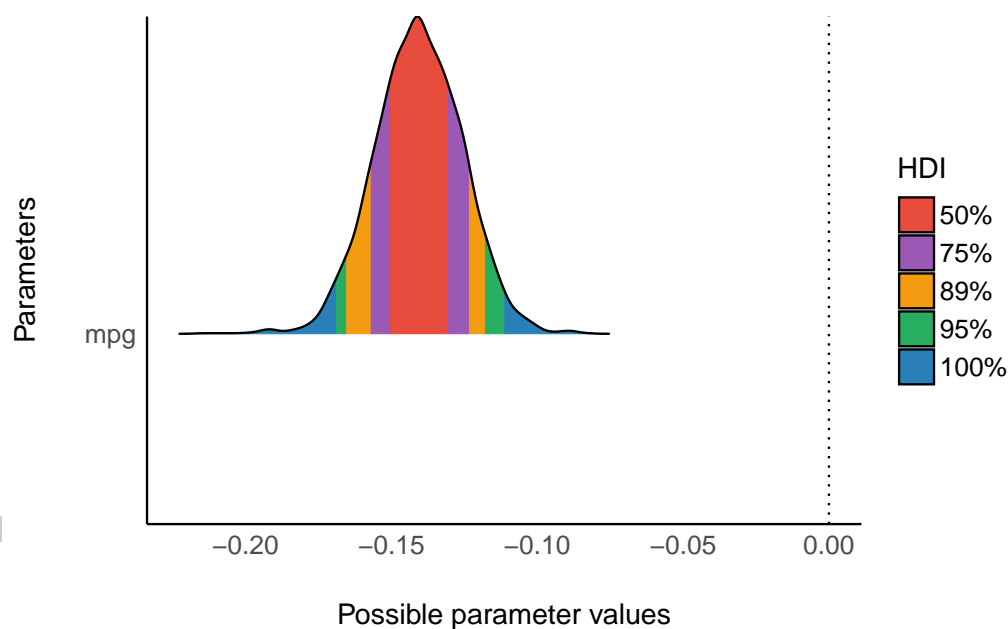
76 Similarly, for Bayesian regression model objects, which are handled by the *bayestestR* pack-
77 age (Makowski et al., 2019), the *see* package provides special plotting methods relevant for
78 Bayesian models (e.g., Highest Density Interval, or *HDI*). Users can fit the model and pass
79 the model results, extracted via *bayestestR*, to *plot()*.

```
library(bayestestR)
library(rstanarm)
library(see)

model <- stan_glm(wt ~ mpg, data = mtcars, refresh = 0)
result <- hdi(model, ci = c(0.5, 0.75, 0.89, 0.95))

plot(result)
```

Highest Density Interval (HDI)



81 Visualizing Model Performance and Diagnostic Checks

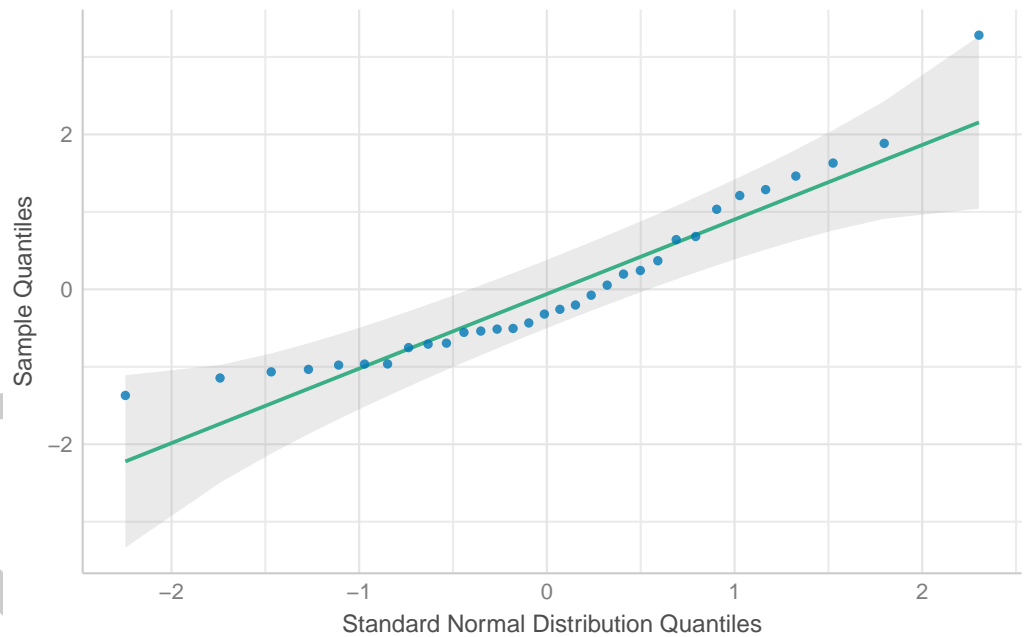
82 The *performance* package is primarily concerned with checking regression model assumptions
83 (Lüdtke et al., 2021). The *see* package offers tools to visualize these assumption checks,
84 such as the normality of residuals. Users simply pass the fit model object to the relevant
85 *performance* function (`check_normality()` in the example below). Then, this result can
86 be passed to `plot()` to produce a *ggplot2* visualization of the check on normality of the
87 residuals.

```
library(performance)
library(see)

model <- lm(wt ~ mpg, data = mtcars)
check <- check_normality(model)
#> Warning: Non-normality of residuals detected (p = 0.016).

plot(check, type = "qq")
```

Normality of Residuals
Dots should fall along the line



88

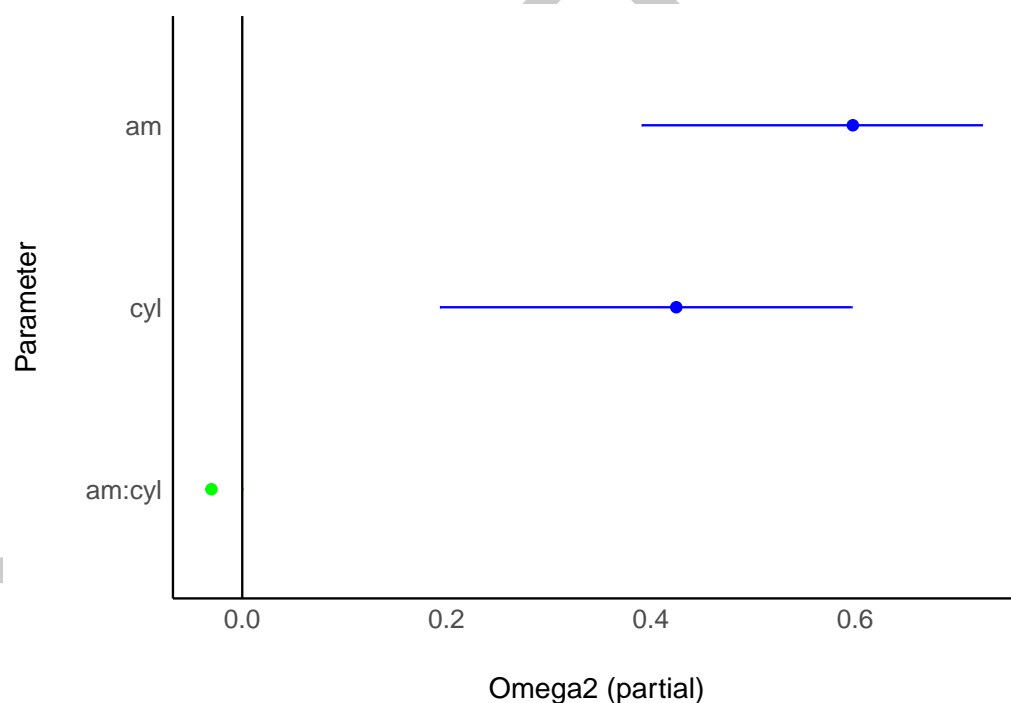
89 Visualizing Effect Sizes

90 The *effectsize* package computes a variety of effect size metrics for fitted models to assesses
91 the practical importance of observed effects (Ben-Shachar et al., 2020). In conjunction with
92 *see*, users are able to visualize the magnitude and uncertainty of effect sizes by passing the
93 model object to the relevant *effectsize* function (`omega_squared()` in the following example),
94 and then to `plot()`.

```
library(effectsize)
library(see)

model <- aov(wt ~ am * cyl, data = mtcars)

plot(omega_squared(model))
```



96 Visualizing Model Predictions and Marginal Effects

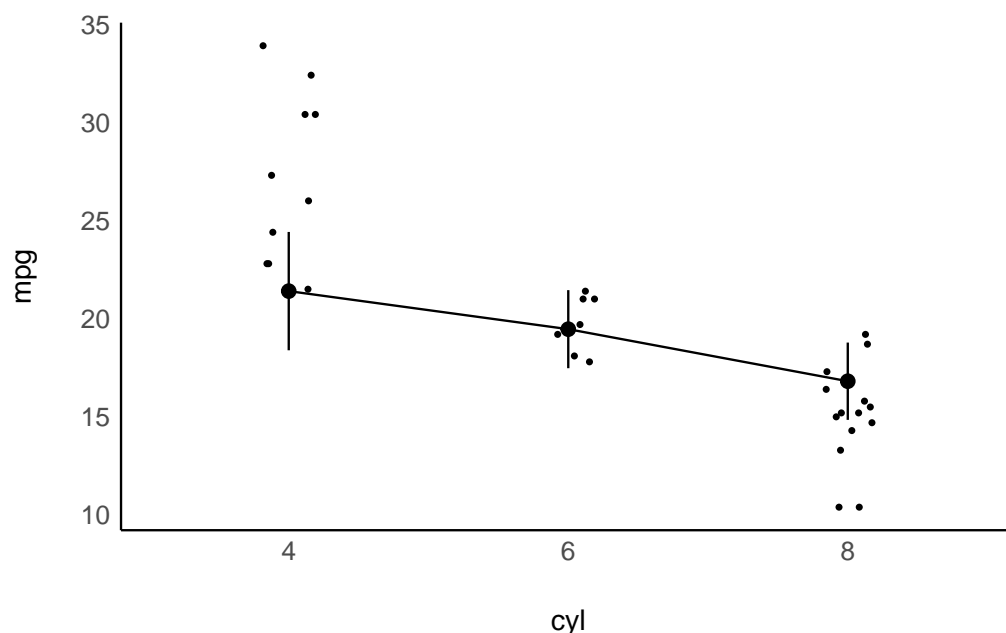
97 The *modelbased* package computes model-based estimates and predictions from fitted models
98 ([Makowski et al., 2020a](#)). see provides methods to quickly visualize these model predictions
99 using `estimate_prediction()`. `estimate_means()` computes *marginal means*, i.e. the
100 mean at each factor level averaged over other predictors.

```
library(modelbased)
library(see)

model <- lm(mpg ~ wt * as.factor(cyl), data = mtcars)
means <- estimate_means(model)

plot(means)
```

Estimated Means (mpg ~ wt * as.factor(cyl))



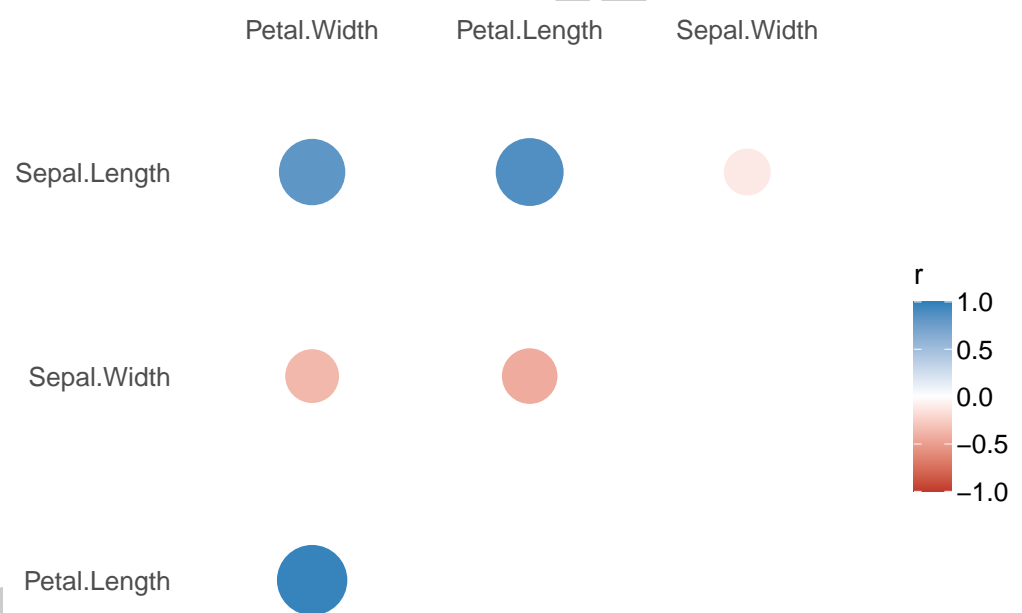
102 Visualizing Correlation Matrices

103 The *correlation* package provides a unified syntax and human-readable code to carry out many
104 types of correlation analysis (Makowski et al., 2020b). A user can run `summary(correlation`
105 `on(data))` to create a construct a correlation matrix for the variables in a dataframe. With
106 `see`, this matrix can be passed to `plot()` to visualize these correlations in a correlation matrix.

```
library(correlation)
library(see)

results <- summary(correlation(iris))

plot(results)
```



108 Licensing and Availability

109 *see* is licensed under the GNU General Public License (v3.0), with all source code openly devel-
110 oped and stored at GitHub (<https://github.com/easystats/see>), along with a corresponding
111 issue tracker for bug reporting and feature enhancements. In the spirit of honest and open
112 science, we encourage requests, tips for fixes, feature updates, as well as general questions
113 and concerns via direct interaction with contributors and developers.

114 Acknowledgments

115 *see* is part of the collaborative *easystats* ecosystem. Thus, we thank the [members of easystats](#)
116 as well as the users.

References

- Ben-Shachar, M. S., Lüdtke, D., & Makowski, D. (2020). effectsize: Estimation of effect size indices and standardized parameters. *Journal of Open Source Software*, 5(56), 2815. <https://doi.org/10.21105/joss.02815>
- Horikoshi, M., & Tang, Y. (2018). Ggfortify: Data visualization tools for statistical analysis results. <https://CRAN.R-project.org/package=ggfortify>
- Lüdtke, D., Ben-Shachar, M. S., Patil, I., & Makowski, D. (2020). Extracting, computing and exploring the parameters of statistical models using R. *Journal of Open Source Software*, 5(53), 2445. <https://doi.org/10.21105/joss.02445>
- Lüdtke, 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>
- Makowski, D., Ben-Shachar, M. S., & Lüdtke, D. (2019). bayestestR: Describing effects and their uncertainty, existence and significance within the Bayesian framework. *Journal of Open Source Software*, 4(40), 1541. <https://doi.org/10.21105/joss.01541>
- Makowski, D., Ben-Shachar, M. S., Patil, I., & Lüdtke, D. (2020a). Estimation of model-based predictions, contrasts and means. CRAN. <https://github.com/easystats/modelbased>
- Makowski, D., Ben-Shachar, M. S., Patil, I., & Lüdtke, D. (2020b). Methods and algorithms for correlation analysis in R. *Journal of Open Source Software*, 5(51), 2306. <https://doi.org/10.21105/joss.02306>
- Patil, I. (2021). Visualizations with statistical details: The 'ggstatsplot' approach. *Journal of Open Source Software*, 6(61), 3167. <https://doi.org/10.21105/joss.03167>
- R Core Team. (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. <https://www.R-project.org/>
- Waggoner, P. D. (2020). plotmm: Tidy tools for visualizing mixture models. <https://CRAN.R-project.org/package=plotmm>
- Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York. ISBN: 978-3-319-24277-4
- Wilkinson, L. (2012). The Grammar of Graphics. In *Handbook of computational statistics* (pp. 375–414). Springer. ISBN: 978-3540404644