The Journal of Open Source Software

statsExpressions: R Package for Tidy Dataframes and Expressions with Statistical Details

Indrajeet Patil[1]

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

DOI:

Software

- Review
- Repository
- Archive

Submitted:
Published:

Summary

The `statsExpressions` package has two key aims: to provide a consistent syntax to do statistical analysis with tidy data, and to provide statistical expressions (i.e., a pre-formatted in-text statistical result) for plotting functions. Currently, it supports common types of statistical approaches and tests: parametric, nonparametric, robust, and Bayesian *t*-test, one-way ANOVA, correlation analyses, contingency table analyses, and meta-analyses. The functions are pipe-friendly and compatible with tidy data.

Statement of need

Statistical packages exhibit substantial diversity in terms of their syntax and expected input and output data type. For example, some functions expect vectors as inputs, while others expect dataframes. Depending on whether it is a repeated measures design or not, functions from the same package might expect data to be in wide or tidy format. Some functions can internally omit missing values, while other functions error in their presence. Furthermore, the statistical test objects returned by the test functions might not have all required information (e.g., degrees of freedom, significance, Bayes factor, etc.) accessible in a consistent data type. Depending on the specific test object and statistic in question, details may be returned as a list, a matrix, an array, or a dataframe. This diversity can make it difficult to easily access all needed information for hypothesis testing and estimation, and to switch from one statistical approach to another.

This is where `statsExpressions` comes in: It can be thought of as a unified portal through which most of the functionality in these underlying packages can be accessed, with a simpler interface and no requirement to change tidy data format.

Comparison to Other Packages

Unlike `broom` (Robinson, Hayes, & Couch, 2021) or `parameters` (Lüdecke, Ben-Shachar, Patil, & Makowski, 2020), the goal of `statsExpressions` is not to convert model objects into tidy dataframes, but to provide a consistent and easy syntax to carry out statistical tests. Additionally, none of these packages return statistical expressions.

Tidy Dataframes from Statistical Analysis

The package offers six primary functions that let users choose a statistical approach without changing the syntax (i.e., by only specifying a single argument). The users are always expected to provide a dataframe in tidy format (Wickham et al., 2019), and all functions work with missing data. Moreover, they always return a dataframe that can be further utilized downstream in the workflow (for a visualization, e.g.).

Table 1: A summary table listing the primary functions in the package and the statistical approaches they support. For a more detailed description of the tests and outputs from these functions, the readers are encouraged to read vignettes on the package website: https://indrajeetpatil.github.io/statsExpressions/articles/.

Function	Parametric	Non-parametric	Robust	Bayesian
`one_sample_test`	✓	✓	✓	✓
`two_sample_test`	✓	✓	✓	✓
`oneway_anova`	✓	✓	✓	✓
`corr_test`	✓	✓	✓	✓
`contingency_table`	✓	✓	-	✓
`meta_analysis`	✓	-	✓	✓

`statsExpressions` internally relies on `stats` package for parametric and non-parametric (R Core Team, 2021), `WRS2` package for robust (Mair & Wilcox, 2020), and `BayesFactor` package for Bayesian statistics (Morey & Rouder, 2020). The random-effects meta-analysis is carried out using `metafor` (parametric) (Viechtbauer, 2010), `metaplus` (robust) (Beath, 2016), and `metaBMA` (Bayesian) (Heck et al., 2019) packages. Additionally, it relies on `easystats` packages (Ben-Shachar, Lüdecke, & Makowski, 2020; Lüdecke, Ben-Shachar, Patil, & Makowski, 2020; Lüdecke, Ben-Shachar, Patil, Waggoner, & Makowski, 2021; Lüdecke, Waggoner, & Makowski, 2019; Makowski, Ben-Shachar, & Lüdecke, 2019; Makowski, Ben-Shachar, Patil, & Lüdecke, 2020) to compute appropriate effect size/posterior estimates and their confidence/credible intervals.

To illustrate the simplicity of this syntax, let's say we want to run a two-sample *t*-test. If we first run a parametric *t*-test and then decide to run a robust *t*-test instead, the syntax remains the same and the statistical approach can be modified by changing a single argument:

```r
mtcars %>% two_sample_test(am, wt, type = "parametric") # Welch's t-test
#> # A tibble: 1 x 14
#>   term  group mean.group1 mean.group2 statistic df.error    p.value
#>   <chr> <chr>       <dbl>       <dbl>     <dbl>    <dbl>      <dbl>
#> 1 wt    am           3.77        2.41      5.49     29.2 0.00000627
#>   method                  estimate conf.level conf.low conf.high effectsize
#>   <chr>                      <dbl>      <dbl>    <dbl>     <dbl> <chr>
#> 1 Welch Two Sample t-test     1.84       0.95     1.00      2.66 Hedges' g
#>   expression
#>   <list>
#> 1 <language>

mtcars %>% two_sample_test(am, wt, type = "robust")     # Yuen's t-test
#> # A tibble: 1 x 10
#>   statistic df.error   p.value
#>       <dbl>    <dbl>     <dbl>
#> 1      5.84     13.6 0.0000485
#>   method                                               estimate conf.low
#>   <chr>                                                   <dbl>    <dbl>
#> 1 Yuen's test on trimmed means for independent samples    0.915    0.754
#>   conf.high conf.level effectsize                         expression
#>       <dbl>      <dbl> <chr>                              <list>
#> 1     0.977       0.95 Explanatory measure of effect size <language>
```

These functions are also compatible with other popular data manipulation packages (see Appendix for an example).

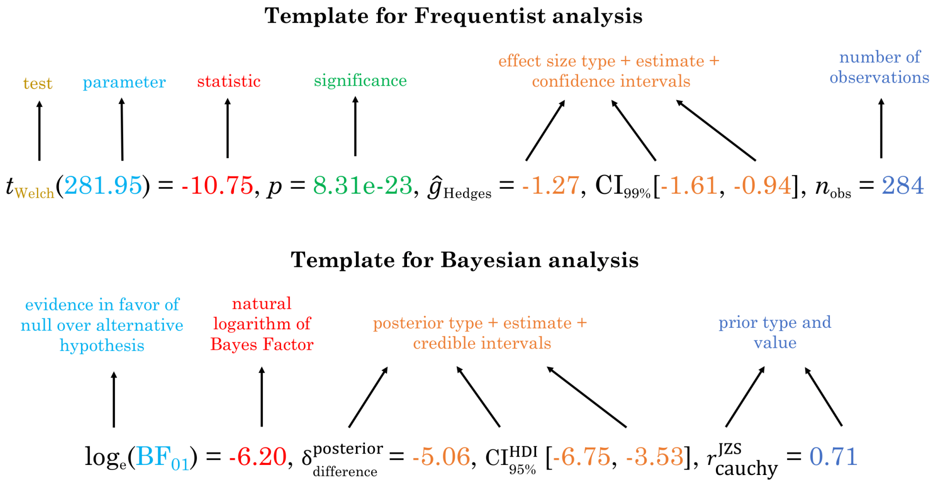
Template for Frequentist analysis
test
parameter
statistic
significance
effect size type + estimate + confidence intervals
number of observations
$t_{\text{Welch}}(281.95) = -10.75,\ p = 8.31\text{e-}23,\ \hat{g}_{\text{Hedges}} = -1.27,\ \text{CI}_{99\%}[-1.61, -0.94],\ n_{\text{obs}} = 284$
Template for Bayesian analysis
evidence in favor of null over alternative hypothesis
natural logarithm of Bayes Factor
posterior type + estimate + credible intervals
prior type and value
$\log_{e}(\text{BF}_{01}) = -6.20,\ \delta^{\text{posterior}}_{\text{difference}} = -5.06,\ \text{CI}^{\text{HDI}}_{95\%}\ [-6.75, -3.53],\ r^{\text{JZS}}_{\text{cauchy}} = 0.71$

Figure 1: The templates used in ‘statsExpressions‘ to display statistical details in a plot.

Expressions for Plots

In addition to other details contained in the dataframe, there is also a column titled `expression`, which contains a pre-formatted text with statistical details. These expressions (Figure 1) attempt to follow the gold standard in statistical reporting for both Bayesian (Doorn et al., 2020) and Frequentist (American Psychological Association and others, 2019) frameworks.

This expression be easily displayed in a plot (Figure 2). Displaying statistical results in the context of a visualization is indeed a philosophy adopted by the `ggstatsplot` package (Patil, 2021), and `statsExpressions` functions as its statistical processing backend.

Licensing and Availability

`statsExpressions` is licensed under the GNU General Public License (v3.0), with all source code stored at GitHub. In the spirit of honest and open science, requests and suggestions for fixes, feature updates, as well as general questions and concerns are encouraged via direct interaction with contributors and developers by filing an issue while respecting *Contribution Guidelines*.

Acknowledgements

I would like to acknowledge the support of Mina Cikara, Fiery Cushman, and Iyad Rahwan during the development of this project. `statsExpressions` relies heavily on the `easystats` ecosystem, a collaborative project created to facilitate the usage of `R` for statistical analyses. Thus, I would like to thank the members of easystats as well as the users.

References

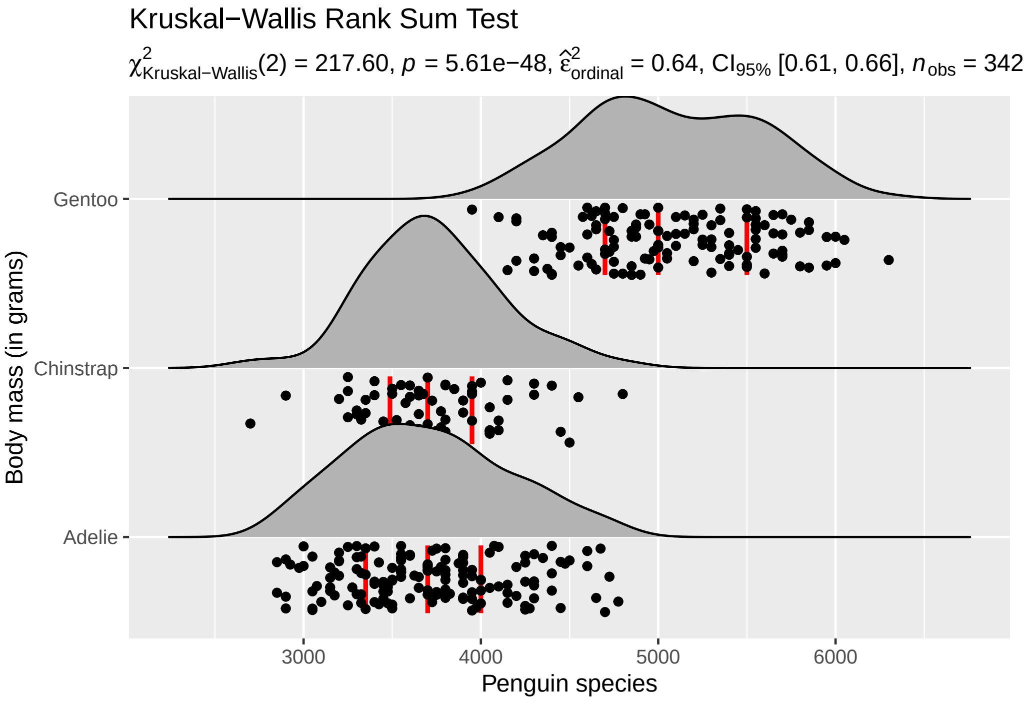

Figure 2: Example illustrating how ‘statsExpressions‘ functions can be used to display results from a statistical test in a plot. Code to create this figure is reported in Appendix.

American Psychological Association and others. (2019). *Publication Manual of the American Psychological Association* (7th Edition.). American Psychological Association.

Beath, K. J. (2016). metaplus: An R package for the analysis of robust meta-analysis and meta-regression. *R Journal*, *8*(1), 5–16. doi:10.32614/RJ-2016-001

Ben-Shachar, M. S., Lüdecke, D., & Makowski, D. (2020). effectsize: Estimation of effect size indices and standardized parameters. *Journal of Open Source Software*, *5*(56), 2815. doi:10.21105/joss.02815

Doorn, J. van, Bergh, D. van den, Böhm, U., Dablander, F., Derks, K., Draws, T., Etz, A., et al. (2020). The JASP guidelines for conducting and reporting a bayesian analysis. *Psychonomic Bulletin & Review*, 1–14. doi:10.3758/s13423-020-01798-5

Heck, W., D., Gronau, F., Q., Wagenmakers, &, & E.-J. (2019). *metaBMA: Bayesian model averaging for random and fixed effects meta-analysis.* Retrieved from https://CRAN.R-project.org/package=metaBMA

Lüdecke, D., Ben-Shachar, M. S., Patil, I., & Makowski, D. (2020). parameters: Extracting, computing and exploring the parameters of statistical models using R. *Journal of Open Source Software*, *5*(53), 2445. doi:10.21105/joss.02445

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. doi:10.21105/joss.03139

Lüdecke, D., Waggoner, P., & Makowski, D. (2019). insight: A unified interface to access information from model objects in R. *Journal of Open Source Software*, *4*(38), 1412. doi:10.21105/joss.01412

Mair, P., & Wilcox, R. (2020). Robust Statistical Methods in R Using the WRS2 Package. *Behavior Research Methods*, *52*, 464–488. doi:10.3758/s13428-019-01246-w

Makowski, D., Ben-Shachar, M. S., & Lüdecke, D. (2019). bayestestR: Describing effects and their uncertainty, existence and significance within the bayesian framework. *Journal of Open Source Software*, *4*(40), 1541. doi:10.21105/joss.01541

Makowski, D., Ben-Shachar, M. S., Patil, I., & Lüdecke, D. (2020). Methods and algorithms for correlation analysis in r. *Journal of Open Source Software*, *5*(51), 2306. doi:10.21105/joss.02306

Morey, R. D., & Rouder, J. N. (2020). *BayesFactor: Computation of bayes factors for common designs*. Retrieved from https://richarddmorey.github.io/BayesFactor/

Patil, I. (2021). Visualizations with statistical details: The 'ggstatsplot' approach. *PsyArxiv*. doi:10.31234/osf.io/p7mku

R Core Team. (2021). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Robinson, D., Hayes, A., & Couch, S. (2021). *Broom: Convert statistical objects into tidy tibbles*. Retrieved from https://CRAN.R-project.org/package=broom

Viechtbauer, W. (2010). Conducting meta-analyses in R with the metafor package. *Journal of Statistical Software*, *36*(3), 1–48. Retrieved from https://www.jstatsoft.org/v36/i03/

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., et al. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. doi:10.21105/joss.01686

Appendix

Example with dplyr

We can use combination of dplyr and statsExpressions to repeat the same statistical analysis across grouping variables.

```
# running one-sample proportion test for all levels of `cyl`
mtcars %>%
  group_by(cyl) %>%
  group_modify(~ contingency_table(.x, am), .keep = TRUE) %>%
  ungroup()
#> # A tibble: 3 x 11
#>     cyl statistic    df p.value method
#>   <dbl>     <dbl> <dbl>   <dbl> <chr>
#> 1     4     2.27      1 0.132   Chi-squared test for given probabilities
#> 2     6     0.143     1 0.705   Chi-squared test for given probabilities
#> 3     8     7.14      1 0.00753 Chi-squared test for given probabilities
#>   estimate conf.level conf.low conf.high effectsize        expression
#>      <dbl>      <dbl>    <dbl>     <dbl> <chr>             <list>
#> 1    0.344       0.95    0         0.917 Cramer's V (adj.) <language>
#> 2    0           0.95    0         0     Cramer's V (adj.) <language>
#> 3    0.685       0.95    0.127     1.18  Cramer's V (adj.) <language>
```

Code to reproduce for Figure 2

```
# needed libraries
library(statsExpressions)
library(ggplot2)
library(ggridges)
library(palmerpenguins) # `penguins` dataset is from this package

# creating a dataframe
res <- oneway_anova(penguins, species, body_mass_g, type = "nonparametric")

# create a ridgeplot using `ggridges` package
ggplot(penguins, aes(x = body_mass_g, y = species)) +
  geom_density_ridges(
    jittered_points = TRUE,
    quantile_lines = TRUE,
    scale = 0.9,
    vline_size = 1,
    vline_color = "red",
    position = position_raincloud(adjust_vlines = TRUE)
  ) + # use 'expression' column to display results in the subtitle
  labs(
    x = "Penguin species",
    y = "Body mass (in grams)",
    title = "Kruskal-Wallis Rank Sum Test",
    subtitle = res$expression[[1]]
  )
```