The Journal of Open Source Software

Visualizations with statistical details: The 'ggstatsplot' approach

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

Graphical displays can reveal problems in a statistical model that might not be apparent from purely numerical summaries. Such visualizations can also be helpful for the reader to evaluate the validity of a model if it is reported in a scholarly publication/report. But, given the onerous costs involved, researchers can avoid preparing information-rich graphics and exploring several statistical approaches/tests available. The `ggstatsplot` package in R programming language (R Core Team, 2021) provides a one-line syntax to enrich `ggplot2`-based visualizations with the results from statistical analysis embedded in the visualization itself. In doing so, the package helps researchers adopt a rigorous, reliable, and robust data exploratory and reporting workflow.

Statement of Need

In a typical data analysis workflow, data visualization and statistical modeling are two different phases: visualization informs modeling, and in turn, modeling can suggest a different visualization method, and so on and so forth (Wickham & Grolemund, 2016). The central idea of `ggstatsplot` is simple: combine these two phases into one in the form of an informative graphic with statistical details.

Before discussing benefits of this approach, we will see one example (Figure 1).

```
set.seed(123) # for reproducibility
library(palmerpenguins) # for 'penguins' dataset
library(ggstatsplot)

ggbetweenstats(penguins, species, body_mass_g)
```

As can be seen, with a single line of code, the function produces details about descriptive statistics, inferential statistics, effect size estimate and its uncertainty, pairwise comparisons, Bayesian hypothesis testing, Bayesian posterior estimate and its uncertainty. Moreover, these details are juxtaposed with informative and well-labeled visualizations. The defaults are designed to follow best practices in both data visualization (Cleveland, 1985; Grant, 2018; Healy, 2018; Tufte, 2001; Wilke, 2019) and (Frequentist/Bayesian) statistical reporting (Association, 2019; Doorn et al., 2020). Without `ggstatsplot`, getting these statistical details and customizing a plot would require significant amount of time and effort In other words, this package removes the trade-off often faced by researchers between ease and thoroughness of data exploration and further cements good data exploration habits.

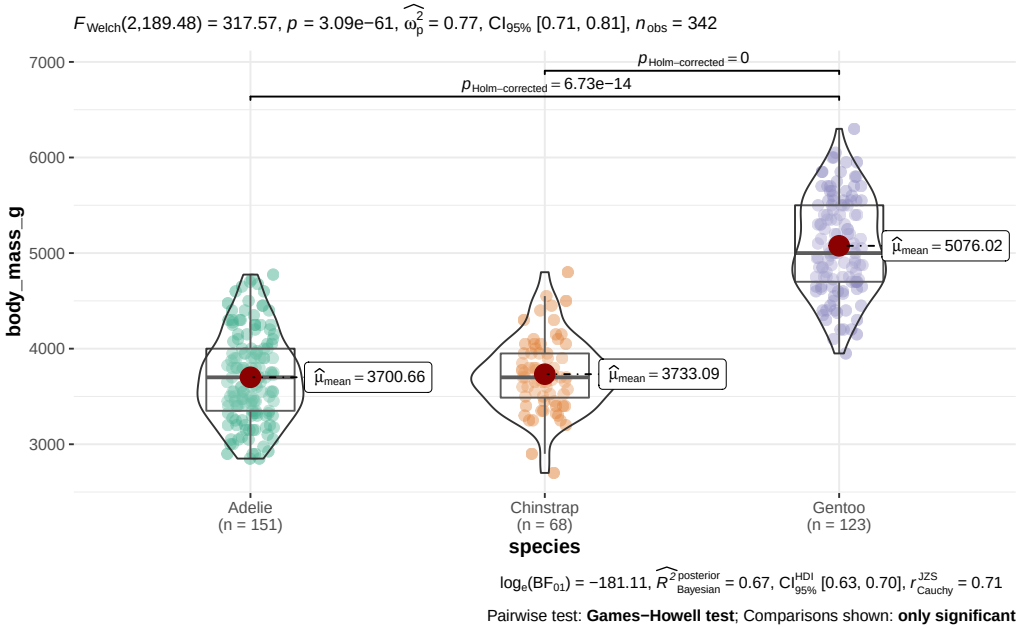

Figure 1: Example plot from the 'ggstatsplot' package illustrates its philosophy of juxtaposing informative visualizations with details from statistical analysis. To see all supported plots and statistical analyses, see the package website: https://indrajeetpatil.github.io/ggstatsplot/

Internally, data cleaning is carried out using `tidyverse` (Wickham et al., 2019), while statistical analysis is carried out via `statsExpressions` (Patil, 2021) and `easystats` (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) packages. All visualizations are constructed using the grammar of graphics framework (Wilkinson, 2012), as implemented in the `ggplot2` package (Wickham, 2016).

Benefits

In summary, the benefits of `ggstatsplot`'s approach are the following. It-

a. produces charts displaying both raw data, and numerical plus graphical summary indices,

b. avoids errors in and increases reproducibility of statistical reporting,

c. highlights the importance of the effect by providing effect size measures by default,

d. provides an easy way to evaluate *absence* of an effect using Bayes factors,

e. encourages researchers and readers to evaluate statistical assumptions of a model in the context of the underlying data (Figure 2),

f. is easy and simple enough that someone with little-to-no coding experience can use it without making an error and may even encourage beginners to programmatically analyze data, instead of using GUI software.