The Journal of Open Source Software

AlgebraOfGraphics.jl: A Makie-powered algebraic grammar of graphics for Julia

Julius Krumbiegel [1¶] **and Pietro Vertechi** [2]

1 PumasAI **2** Ottante, Milan, Italy ¶ Corresponding author

DOI: 10.21105/joss.10894

Software

- Review
- Repository
- Archive

Editor: Mehmet Hakan Satman

Reviewers:

- @akchurinda
- @goerz

Submitted: 01 May 2026
Published: 27 July 2026

Summary

`AlgebraOfGraphics.jl` is a Julia (Bezanson et al., 2017) package for declarative data visualization built on top of the `Makie.jl` plotting ecosystem (Danisch & Krumbiegel, 2021). It provides a high-level grammar in which datasets, variable mappings, analyses, and visual specifications are represented as first-class objects that can be combined with the algebraic operators + and *. The distributive property of these operators makes it possible to factor out information that is shared across layers of a figure, eliminating much of the repetition that users encounter when composing layered plots in lower-level plotting APIs. A plot specification is rendered into a full Makie figure by a single `draw` call, and because the result is an ordinary Makie figure, users retain full programmatic access to every axis, legend, and plot object for further customization.

Statement of need

Exploratory statistical visualization requires expressing, in as few lines as possible, the relationship between variables in a dataset and the visual attributes of a figure. Users typically want to map categorical variables to colors or markers, split a figure into small multiples, overlay raw data with summary statistics, attach legends and colorbars, and label axes meaningfully — all while iterating rapidly on the underlying question. Doing this directly with a general-purpose plotting library such as `Makie.jl` is technically possible but cumbersome: the user is forced to manage groupings, allocate palette entries, compute limits across facets, and assemble legends manually, and the resulting code obscures the analytic question behind a large amount of bookkeeping.

R's `ggplot2` (Wickham, 2016) popularized the grammar of graphics (Wilkinson, 2005) as a solution to this problem, and ports and successors exist in many languages. In Julia, however, there was no well-maintained grammar-of-graphics front-end for `Makie.jl`, which has become one of the most capable native Julia plotting libraries, with support for 2D and 3D graphics, interactivity, and multiple backends (Danisch & Krumbiegel, 2021). `AlgebraOfGraphics.jl` fills this gap, and does so with two design choices that distinguish it from existing grammar-of-graphics libraries.

First, `AlgebraOfGraphics.jl` represents plot specifications as elements of an algebraic structure. Users combine building blocks — `data(table)`, `mapping(:x, :y, color = :group)`, `visual(Scatter)`, analyses such as `linear()` or `density()` — with * (which merges specifications) and + (which superimposes layers). Because * distributes over +, information that is shared across layers can be written once and factored out. For example,

```
data(penguins) * mapping(:bill_length_mm, :bill_depth_mm, color = :species) *
    (visual(Scatter) + linear())
```

expresses a scatter plot with a linear fit per species, with the dataset, mapping, and color

grouping factored out of the superimposed layers. This algebraic factoring is rarely possible in imperative grammar-of-graphics libraries, where each layer typically re-specifies its own data and aesthetics. Figure 1 illustrates how a plot is built incrementally and how the individual building blocks combine into a single expression.

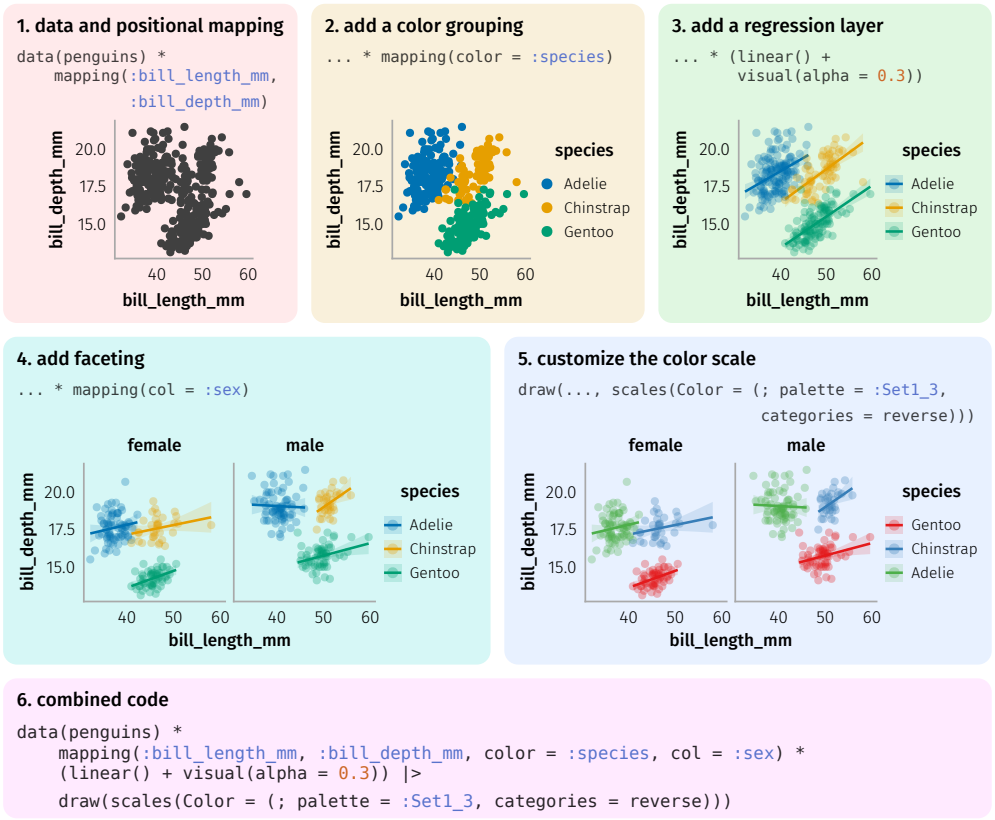

Figure 1: Building a plot incrementally with `*` and `+`. Each panel shows a step in the construction of a visualization of the Palmer penguins dataset: starting from `data` and a positional `mapping`, we add a color grouping, a regression layer, a facet by sex, and finally a customized color scale with reversed category order. The bottom panel shows the equivalent expression as a single pipeline.

Second, `AlgebraOfGraphics.jl` is a thin grammar layer over a native Julia plotting library rather than a self-contained rendering engine. The result of `draw` is a Makie figure, which means users can freely mix `AlgebraOfGraphics` layers with hand-authored Makie plots, add or modify axes after the fact, and leverage Makie's interactivity and multiple backends without leaving Julia. This matters in practice because published figures are rarely identical to their first draft: authors routinely tweak labels, add annotations, adjust legends, and rearrange panels. Doing this programmatically keeps the figure fully reproducible, in contrast to workflows that finish a figure in Illustrator or Inkscape.

State of the field

Several existing tools occupy adjacent parts of the space:

- `ggplot2` (Wickham, 2016) is the canonical grammar-of-graphics implementation. Its underlying representation is an R `grob` tree that is comparatively opaque once rendered, making it difficult to adjust properties of the figure which are not directly exposed by ggplot's interface.

- `plotnine` (Kibirige, 2024) and `seaborn` (Waskom, 2021) occupy similar ground in Python. They can be called from Julia via interop, but cannot be extended from Julia, so native Julia data types and plotting recipes do not integrate with them. Tweaking the resulting figures further requires familiarity with `matplotlib`, which adds another layer of tooling that Julia users may not be familiar with.
- `Gadfly.jl` (Jones et al., 2014) was an earlier grammar-of-graphics library for Julia with an elegant design, but its output options and interactivity story remained limited and its development has largely stalled.
- `VegaLite.jl` (Anthoff & VegaLite.jl contributors, 2021) wraps the Vega-Lite declarative specification. The generated artefact is a JSON spec, which means figures are portable across renderers but cannot be freely manipulated with native Julia plotting code.
- `TidierPlots.jl` (Boyes, 2024) is a recent effort to port `ggplot2`'s syntax onto `Makie.jl`. It shares `AlgebraOfGraphics.jl`'s Makie substrate and overlaps substantially in the set of plots that can be produced. The key difference is `AlgebraOfGraphics.jl`'s algebraic model: `+` and `*` combined with distributivity offer a compositional style that differs from `ggplot2`'s imperative +-chains.
- `StatsPlots.jl` (Breloff & StatsPlots.jl contributors, 2016) extends the recipe-based `Plots.jl` ecosystem (Christ et al., 2023) with statistical plots, but it is not a grammar-of-graphics library and does not separate data, mapping, and visual specification.

Software design

A plot specification is represented internally as a `Layer` or sum of layers. Each layer carries a dataset, a mapping from column references to aesthetic roles, a visual specification (which Makie recipe to use and with which fixed attributes), and optionally an analysis that transforms the data before plotting (e.g., kernel density estimation, a linear fit, histogramming). During `draw`, layers are grouped, analyses are applied, and the resulting data are handed to a scales system that resolves categorical and continuous variables into concrete Makie attributes across all axes, legends, and colorbars of the figure.

The scales system is a central piece of the current architecture and is the result of a major refactor of the original codebase. In the first versions of `AlgebraOfGraphics.jl`, the intent was to be a thin dispatcher that forwarded positional and keyword arguments almost directly to Makie recipes. Experience showed that keeping an implicit interface between AlgebraOfGraphics and Makie plotting functions was not sustainable, not all functions subscribe to the pattern that X and Y are the first two positional arguments, not all functions have only a single color attribute called `color`. Introducing an explicit scales system resolved a large class of latent bugs and unlocked features that are otherwise difficult to express, including multiple independent color scales with their own colorbars within a single figure, multiple independent X or Y scales across neighboring facets, and correct label and tick management for empty or partially populated facets.

Because the connection between a Makie recipe and the AoG scale machinery is established through multiple dispatch on the recipe type, third-party Makie plotting recipes can register themselves with `AlgebraOfGraphics.jl` from outside the package, making the system fully extensible: any package that defines a custom Makie recipe can opt it into the grammar without modifications to `AlgebraOfGraphics.jl`. In combination with Julia's package extension mechanism, available since Julia 1.9, this support can be added conditionally — a downstream package can ship AoG integration that activates only when both `AlgebraOfGraphics` and the recipe-providing package are loaded, without taking on a hard dependency on either side.

Other notable design choices include: support for multiple tabular formats via the `Tables.jl` interface (Quinn & Tables.jl contributors, 2021) as well as "pre-grouped" collections of arrays for data that is not naturally tabular; wide-data mappings via `dims`, avoiding the need to reshape data before plotting; element-wise column transformations inside `mapping`, which interact correctly with grouping and faceting; a flexible pagination system for plots that would

otherwise contain too many facets to display at once; and first-class support for units through `Unitful.jl` (Keller & Unitful.jl contributors, 2016) and `DynamicQuantities.jl` (Cranmer & DynamicQuantities.jl contributors, 2023).

Research impact

`AlgebraOfGraphics.jl` is used across several communities. In pharmacometrics, it is the recommended visualization tool in the Pumas software stack (Rackauckas et al., 2020). Other software like `UnfoldMakie.jl` (Mikheev & Ehinger, 2025), a toolbox for visualization of event-related EEG analyses, has been built on top of it. It is covered in open educational resources including the *Julia Data Science* book (Storopoli et al., 2021) and John Verzani's *Using Julia for Introductory Statistics* (Verzani, 2025). The package has accumulated over 500 stars on GitHub and is regularly discussed on the Julia Discourse. Because no canonical citation has existed until now, adoption in the scientific literature has been under-reported; this paper is intended to provide a stable reference for use in future publications.

AI usage disclosure

The manuscript was drafted with the assistance of a large language model (Anthropic's Claude) acting as a writing collaborator. All claims, structure, and wording were reviewed and edited by the authors. Within the software itself, no AI-generated code was merged during the first five years of development; AI coding assistance has been used only for a small fraction of recent contributions and is always reviewed by a human maintainer before merging.

Acknowledgements

`AlgebraOfGraphics.jl` builds on `Makie.jl`, and we thank Simon Danisch, Frederic Freyer and all Makie contributors for their continued work on the underlying plotting framework. We thank Fabian Greimel for substantial early contributions to the faceting and legend subsystems, and all other community contributors whose fixes, bug reports, and feature requests have shaped the package over the years. JK's work on `AlgebraOfGraphics.jl` has been supported by PumasAI.

References

Anthoff, D., & VegaLite.jl contributors. (2021). *VegaLite.jl: Julia bindings to Vega-Lite*. https://github.com/queryverse/VegaLite.jl

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Boyes, R. (2024). *TidierPlots.jl: Tidier data visualization in Julia, modeled after the ggplot2 R package*. https://github.com/TidierOrg/TidierPlots.jl

Breloff, T., & StatsPlots.jl contributors. (2016). *StatsPlots.jl: Statistical plotting recipes for Plots.jl*. https://github.com/JuliaPlots/StatsPlots.jl

Christ, S., Schwabeneder, D., Rackauckas, C., Borregaard, M. K., & Breloff, T. (2023). Plots.jl – a user extendable plotting API for the Julia programming language. *Journal of Open Research Software*, *11*(1), 5. https://doi.org/10.5334/jors.431

Cranmer, M., & DynamicQuantities.jl contributors. (2023). *DynamicQuantities.jl: Efficient and type-stable physical quantities in Julia*. https://github.com/JuliaPhysics/DynamicQuantities.jl

Danisch, S., & Krumbiegel, J. (2021). Makie.jl: Flexible high-performance data visualization for Julia. *Journal of Open Source Software*, *6*(65), 3349. https://doi.org/10.21105/joss.03349

Jones, D. C., Nagy, T., Leblanc, S., & Gadfly.jl contributors. (2014). *Gadfly.jl: Crafty statistical graphics for Julia*. https://doi.org/10.5281/zenodo.593105

Keller, A. J., & Unitful.jl contributors. (2016). *Unitful.jl: Physical quantities with arbitrary units in Julia*. https://github.com/JuliaPhysics/Unitful.jl

Kibirige, H. (2024). *plotnine: A grammar of graphics for Python*. https://plotnine.org

Mikheev, V., & Ehinger, B. V. (2025). UnfoldMakie.jl: EEG/ERP visualization package. *Journal of Open Source Software*, *10*(105), 7560. https://doi.org/10.21105/joss.07560

Quinn, J., & Tables.jl contributors. (2021). *Tables.jl: An interface for tables in Julia*. https://github.com/JuliaData/Tables.jl

Rackauckas, C., Ma, Y., Noack, A., Dixit, V., Mogensen, P. K., Byrne, S., Maddhashiya, S., Santiago Calderón, J. B., Nyberg, J., Gobburu, J. V. S., & Ivaturi, V. (2020). Accelerated predictive healthcare analytics with Pumas, a high performance pharmaceutical modeling and simulation platform. *bioRxiv*. https://doi.org/10.1101/2020.11.28.402297

Storopoli, J., Huijzer, R., & Alonso, L. (2021). *Julia Data Science*. ISBN: 9798489859165

Verzani, J. (2025). *Using Julia for introductory statistics*. https://jverzani.github.io/UsingJ/

Waskom, M. L. (2021). seaborn: Statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. https://doi.org/10.21105/joss.03021

Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis* (2nd ed.). Springer-Verlag New York. ISBN: 978-3-319-24277-4

Wilkinson, L. (2005). *The grammar of graphics* (2nd ed.). Springer-Verlag New York. https://doi.org/10.1007/0-387-28695-0