DOI: 10.21105/joss.02053

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Submitted: 27 January 2020
Published: 05 February 2020

Latexify.jl, translating mathematical Julia objects to renderable equations and tables.

Niklas Korsbo[1, 2]

1 The Sainsbury Laboratory, Cambridge University **2** Department of Applied Mathematics and Theoretical Physics, Cambridge University

Summary

Human-understandable representation and visualisation of data and objects is important for understanding and communicating the results of scientific software.

Latexify.jl is a tool for converting Julia (Bezanson, Edelman, Karpinski, & Shah, 2017) objects to humanly accessible and renderable equations and tables. It allows for the conversion of multiple types to LaTeX or Markdown-formatted strings and, in many development environments, for immediate rendering of the result. Among the supported inputs is a class of Expressions that describe mathematical equations. These can be translated into LaTeX formatted mathematics and can be outputted as LaTeX environments such as align, equation or in-line equations. The conversion can recurse through many container types, even if they contain mixed types, and produce resulting tables, arrays or aligned equations for LaTeX or Markdown. The output is configurable and a recipe system makes it easy to extend Latexify.jl to work with custom types without having to modify Latexify.jl itself. This becomes especially powerful in combination with Julia's metaprogramming facilities and easily generated domain-specific languages (DSLs). Latexify.jl can, for example, output the system of differential equations (and much more) that is automatically generated by a chemical reaction arrow DSL provided by DiffEqBiological.jl (Rackauckas & Nie, 2017).

The package aims to support the scientific work-flow through facilitating inspection, automation and representation. Simple inspection of the equations that a computational model is ultimately simulating may increase comprehensibility and improve troubleshooting. It, furthermore, allows for de-mystification of computational models generated using DSLs. The programmatic formatting of equations and tables enables their automatic inclusion into generated documents, documentation, reports or posts (possibly in combination with tools such as Weave.jl (Pastell, 2017)). Such programmatic translation can also help to ensure an accurate correspondence between what software does and what reports or articles claim that they do. It is also just rather convenient.

Acknowledgements

I acknowledge the contributions from the Julia programming community and especially those who through issues and pull requests have improved Latexify.jl. I would also like to acknowledge my PhD supervisor, Henrik Jönsson, who supports me in my work while allowing me to allocate my efforts as I see fit.

The development of this package was, in part, supported by the Gatsby Charitable Foundation, grant GAT3395-PR4.

References

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

Pastell. (2017). Weave.jl: Scientific reports using julia. *Journal of Open Source Software*, *2*(11), 204. doi:doi:10.21105/joss.00204

Rackauckas, C., & Nie, Q. (2017). DifferentialEquations.jl – a performant and feature-rich ecosystem for solving differential equations in julia. *The Journal of Open Research Software*, *5*(1). doi:10.5334/jors.151