The Journal of Open Source Software

JuliaCall: an R package for seamless integration between R and Julia

Changcheng Li[1]

1 Department of Statistics, Pennsylvania State University at University Park

DOI: 10.21105/joss.01284

Software

- Review
- Repository
- Archive

Submitted: 22 February 2019
Published: 13 March 2019

Summary

R is a widely used software and computing environment for statistics, which provides a variety of statistical techniques and packages (R Core Team, 2018). Many R packages provide interfaces to other computer languages to bring new functionalities to R or to accelerate computations, such as Rcpp (Eddelbuettel & François, 2011) and V8 (Ooms, 2019). The package JuliaCall provides an R interface to Julia, which is a computer language for high-performance scientific computing (Bezanson, Edelman, Karpinski, & Shah, 2017). JuliaCall embeds Julia in R, and provides functions to evaluate Julia commands, to call Julia functions, to transmit data objects between R and Julia, and so on. It also provides many utilities for user convenience. For example, JuliaCall gives detailed error messages for the embedded Julia. It also provides Julia package management functions such as installation and loading, and utility functions to get the documentation of Julia functions. JuliaCall can also be used in R Markdown document as the engine of Julia language, see Section 2.77 in Xie, Allaire, & Grolemund (2018).

Some R packages wrap Julia packages based on JuliaCall to provide new functionalities or performance improvements to some existing packages in R. autodiffr (Li, 2018a) provides automatic differentiation to R functions by wrapping ForwardDiff.jl and ReverseDiff.jl. convexjlr (Li, 2018b) is an R package for Disciplined Convex Programming (DCP) providing a high level wrapper for Convex.jl. diffeqr (Rackauckas, 2018) solves differential equations in R using DifferentialEquations.jl. FixedEffectjlr (Loualiche, 2018) estimates large fixed effects models in R by providing an interface to FixedEffectModels.jl.

Besides JuliaCall, some other packages also provide interfaces between R and Julia: R packages XRJulia, RJulia, and the Julia package RCall.jl. The package XRJulia connects to Julia from R (Chambers, 2017). It uses JavaScript Object Notation (JSON) format to transmit data, while JuliaCall copies objects in memory between R and Julia. It has performance disadvantages compared to JuliaCall. Table 1 depicts the times needed to transmit a 500×500 matrix full of ones from R v3.5.2 to Julia v1.0.3 using JuliaCall v0.16.4 and XRJulia Github master b6224fa at the time of writing (there is no released version of XRJulia to support Julia v1.0 yet). The times are measured by R package microbenchmark with 1000 evaluation times. The script with the benchmark code and setup instructions can be found in the paper directory in JuliaCall Github repository. In the header of the table, "lq" means lower quantile and "uq" means upper quantile. From Table 1, it can be seen that JuliaCall has a speed advantage of transmitting data between R and Julia. The package RJulia (Gong, Haverty, Keys, & Maechler, 2017) also embeds Julia in R, but its functionality is quite limited, has not been updated for more than one year, and does not support Julia v1.0 and v1.1 at the time of writing. RCall.jl is a Julia package which embeds R in Julia. It is a

dependency for `JuliaCall`, and `JuliaCall` utilizes `RCall.jl`'s type conversion between `R` and `Julia`. `JuliaCall` integrates well with `RCall.jl`, and it is the default for `JuliaCall` to load `RCall.jl` in the embedded `Julia` automatically at starting. With `JuliaCall` and `RCall.jl`, it is easy to use `R` from `Julia` and `Julia` from `R`.

Table 1: Time measurements for `XRJulia` and `JuliaCall` to transmit a 500×500 matrix from `R` to `Julia`.

Time in ms	min	lq	mean	median	uq	max
XRJulia	24.680699	30.65626	39.29180	33.98684	38.83510	454.6484
JuliaCall	9.790442	11.79500	16.39797	12.55267	13.90892	406.0156

Users can get stable releases of `JuliaCall` from CRAN, and the latest development version from `JuliaCall` Github repository. Documentation can be found in the package as well as on CRAN. Bug reports and other feedback can be submitted to GitHub issue page.

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

Chambers, J. M. (2017). *Extending R*. CRC Press. doi:10.1201/9781315381305

Eddelbuettel, D., & François, R. (2011). Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, *40*(8), 1–18. doi:10.18637/jss.v040.i08

Gong, Y., Haverty, P. M., Keys, O., & Maechler, M. (2017). *rjulia: Integrating R and Julia – Calling Julia from R*. Retrieved from https://github.com/armgong/rjulia

Li, C. (2018a). *autodiffr: Automatic Differentiation in R through 'Julia'*. Retrieved from https://github.com/Non-Contradiction/autodiffr

Li, C. (2018b). *convexjlr: Disciplined Convex Programming in R using 'Convex.jl'*. Retrieved from https://github.com/Non-Contradiction/convexjlr

Loualiche, E. (2018). *FixedEffectjlr: A package for fast Fixed Effect Models in R*. Retrieved from https://github.com/eloualiche/FixedEffectjlr

Ooms, J. (2019). *V8: Embedded JavaScript Engine for R*. Retrieved from https://CRAN.R-project.org/package=V8

R Core Team. (2018). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Rackauckas, C. (2018). *diffeqr: Solving Differential Equations (ODEs, SDEs, DDEs, DAEs)*. Retrieved from https://github.com/JuliaDiffEq/diffeqr

Xie, Y., Allaire, J. J., & Grolemund, G. (2018). *R markdown: The definitive guide*. CRC Press. doi:10.1201/9781138359444