The Journal of Open Source Software

set6: R6 Mathematical Sets Interface

Raphael E.B. Sonabend[1] **and Franz J. Kiraly**[1, 2]

1 Department of Statistical Science, University College London, Gower Street, London WC1E 6BT, United Kingdom **2** Shell

DOI: 10.21105/joss.0XXXX

Software

- Review
- Repository
- Archive

Editor: Editor Name

Submitted: 01 January 1900
Published: 01 January 3030

Summary

set6 makes use of the R6 object-oriented paradigm to introduce classes for important mathematical containers, including sets, tuples, and intervals (finite and infinite). Until now, the `R` (R Core Team, 2017) programming language has traditionally supported mathematical sets in one of two ways: 1. via the five set operation functions: `union`, `intersect`, `setdiff`, `setequal`, `is.element`; and 2. via the `sets` (Meyer & Hornik, n.d.) package. `set6` uses `R6` (Chang, 2018) and has a clear class interface with minimal dependencies, which makes it the perfect dependency for any package that requires containers for R6 objects. Making use of design patterns (Gamma, Helm, Johnson, & Vlissides, 1996), such as wrappers and compositors, `set6` allows for symbolic representation of sets to ensure maximum efficiency, and to provide neat and clear print methods.

`set6` is currently being used in `distr6` (Sonabend & Kiraly, 2019), which is an object-oriented probability distributions interface, that makes use of `set6` for distribution and parameter support. Additional uses of `set6` include representing infinite sets, and constructing assertions.

The speed and efficiency of `R6` allows `set6` to be a scalable and efficient interface. A focus on symbolic representation and neat printing methods means that `set6` can accurately and clearly represent complicated sets. `set6` has the ambitious long-term goal of being the only dependency package needed for object-oriented interfaces in `R` that require clear symbolic representations of mathematical sets.

Related software includes the `sets` (Meyer & Hornik, n.d.) family of packages.

Key Design Principles

1. **Maximum user-control over set operations** - Users can select how operations act on sets, including a choice of associativity, lazy evaluation, and unicode printing.
2. **Minimal dependencies** - `set6` has the goal of being a key dependency to any object-oriented requiring representation of mathematical sets, for example for representing function inputs and supports. Therefore `set6` is itself dependent on only three packages.
3. **Transparency in sets** - `set6` priorities symbolic representation and lazy evaluation to allow for the package to be scalable and to fit into any other package. However it is ensured that this does not detract from transparency, i.e. that an object that prints well is still clear in what it actually contains. `set6` allows sets to be queried in many different ways, including calling the elements in the set (if finite), finding the bounds of the set (if numeric), and listing properties and traits.

Key Use-Cases

1. **Constructing and querying mathematical sets** - Many mathematical Set-like objects can be constructed, including sets, tuple, intervals, and fuzzy variants. Sets and tuples can contain objects of any `R` type (atomic or otherwise).
2. **Containedness checks** - Public methods allow all objects inheriting from `Set` to check if elements are contained within them. This provides a powerful mechanism for use with parameter or distribution supports for other packages.
3. **Representation of infinite sets** - Symbolic representation and lazy evaluation allows infinite (or very large) sets and intervals to be constructed. This also allows operations such as powerset to be used without crashing the system.
4. **Comparison of, possibly infinite, sets** - Two `Set` objects can be compared to check if they are equal or (proper) sub/supersets. Infix operators allow quick and neat comparison.
5. **Creation of multi-dimensional sets from simpler classes** - Common set operations, such as unions and complements are implemented, as well as products and exponents. Using these, sets of any complexity can be constructed.

Software Availability

`set6` is available on GitHub and CRAN. It can either be installed from GitHub using the `devtools` (Wickham, Hester, & Chang, 2019) library or directly from CRAN with `install.packages`. The package uses the MIT open-source licence. Contributions, issues, feature requests, and general feedback can all be found and provided on the project GitHub. Full tutorials and further details are available on the project website.

References

Chang, W. (2018). R6: Classes with Reference Semantics. Retrieved from https://cran.r-project.org/package=R6

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1996). Design Patterns: Elements of Reusable Software. doi:10.1093/carcin/bgs084

Meyer, D., & Hornik, K. (n.d.). doi:10.18637/jss.v031.i02

R Core Team. (2017). R: A Language and Environment for Statistical Computing. Vienna: R Foundation for Statistical Computing.

Sonabend, R., & Kiraly, F. (2019, July). distr6: The Complete R6 Probability Distributions Interface. CRAN. Retrieved from https://cran.r-project.org/package=distr6

Wickham, H., Hester, J., & Chang, W. (2019). devtools: Tools to Make Developing R Packages Easier. CRAN.