Dear Editor,

Thank you for considering our paper for the RJournal and for allowing us the opportunity to make revisions. Our point-by-point responses to both reviewers are provided from the next page. We refer to the reviewer who sent the attachment entitled “2-review-2” as “Reviewer 1” and the other reviewer as “Reviewer 2”. We first address the points that both reviewers made together and then their individual points separately. Original comments from reviewers are in the black bullet points and our replies are by the white arrows. Groups of comments/replies are separated by line breaks.

Kind Regards,

Raphael Sonabend

**Reviewer 1 and Reviewer 2**

* **Reviewer 1** 2. The section on alternative work should mention the `distributional` package by Mitchell O'Hara-Wild, which is arguably the most modern S3 approach to random variables in R at the moment.

**Reviewer 2** Related work: The authors seem to have missed some related prior work. In particular, one more package that also offers object-oriented framework for distributions is distributional.

* + We would like to thank both reviewers for alerting us to this package. When we submitted this paper the `distributional` package had only recently been released to CRAN and as such we were not aware of it. Our ‘related work’ section has been updated to add this package.
* **Reviewer 1** For example, S3 is described as a downgrade of S4.

**Reviewer 2** The paper states multiple times that distr6 “is the ‘official’ upgrade” to distr. While this may be true for the class choice, it is not fully true for the functionalities.

* + We understand that the terms in our paper “upgrade” and “downgrade” were unintentionally ‘loaded’ or misleading. Our terms were simply meant to acknowledge the chronology of implementation of paradigms in R, i.e.: S3, S4, R6. This was not a statement about if one paradigm is better than another (we certainly don’t believe this is the case) or about functionality.
  + These terms have been removed completely from the paper.

**Reviewer 1 (“2-review-2”)**

* `distr6` is a sophisticated but puzzling package. The code appears to be high quality and reliable, and I imagine will find some small to moderate amount of use in the statistical community. However the design choices in the package seem to me to be motivated primarily by aesthetic considerations regarding object orientation. Further, there is very little new functionality in `dist6` that is unavailable elsewhere in the R ecosystem, and what new functionality `distr6` provides is poorly motivated as a solution to applied problems.
* It is important to realize that the primary use cases for random variable packages are to enable repeated sampling from distributions and likelihood evaluation. The choice of OOP paradigm for these goals is almost entirely irrelevant and reduces in the end to syntactic differences at the user level. Further, both sampling and likelihood evaluation do not require any sort of stateful computation, so there is no pragmatic need for encapsulation.
  + We would like to thank the reviewer for their review and would like to take this opportunity to address their points, starting with their initial feedback of the package in general and some confusion around the purpose of the package.
  + It is clear from this review that we have fallen short in clearly articulating and presenting the primary motivation and purpose of the package. Our package is not a “random variable package” but a “probability distributions package”. Therefore, the primary use-cases are for modelling and prediction of distributions and not sampling or likelihood evaluation. We have now explicitly stated this in the introduction of the paper.
  + We would therefore like to highlight that all design choices were motivated and designed with the end-user in mind in order to optimise the experience of the package as a package for class object-oriented programming and in particular object-oriented representations of probability distributions.
  + We would also like to point out that there is significant, new functionality when considering this is the first distributions interface in R6, which is gaining popularity in R. As well as in contributions to further class object-oriented programming in R.
* 1. The article is overall relatively clear with several sections that would benefit from revision. Most notably, the introduction begins by differentiating between probability measures and their associated laws. It is unclear why this differentiation is necessary programmatically and I suspect the vast majority of readers unfamiliar with measure theory will stop here. In the introduction some additional care to differentiate between classes and class instances would also go a long way.
  + Our mathematical discussion is motivated for the exact purpose of differentiating between: i) classes and class instances (see first paragraph of ‘Distributions as software objects and mathematical objects’); and ii) random variables and probability distributions (this is made clearer with a new subsection heading). This is necessary programmatically as it motivates the difference between programming random variables and distributions (see the point above).
  + With respect to putting off readers, we believe this discussion is important and is thus retained. However, we understand the reviewer’s point and would like to point out that the paper can still be understood should a reader choose to skip this section.
* Stylistically, the article uses the passive voice extensively, as well as meta-language such as "thus", "therefore", and so forth. I personally would prefer to see the article written in the active voice and without meta-language.
  + We acknowledge the reviewer’s preference here, but we chose a language style that we are used to writing and reading in journals. We would be happy to adapt this style if the Editor states a more active style is preferred for this journal.
* 3. A substantial portion of the paper is devoted to comparing systems for object orientation in R. This discussion came across as intermittently condescending and naïve.
  + Firstly, we would like to apologise to the reviewer. It was certainly never our intention to be condescending and we also do not want to appear naïve, especially as we have experience in multiple paradigms and are interested and have a lot of respect for the implementation of R and its paradigms throughout the history of the language.
  + We would also like to highlight that the page of the paper concerned with comparing object orientation in R is focused on comparing functional and class object-oriented programming and not in ranking or directly comparing paradigms against one another. We have attempted to make our language less subjective in order to make this clearer.
* There is no acknowledgment of the fact that S3 is the dominant paradigm in R for reasons beyond historical lock-in; in fact S3 has numerous advantages that make it well suited for working within R's functional programming style.
  + We would like to point out that we do in fact acknowledge the power of S3 for its abilities in dispatch and readability. The purpose of comparison to S3 is to highlight it is a functional and not object-oriented paradigm, we have no criticism of S3 itself.
  + We have decreased the overall section but added slightly more discussion around the benefits of S3.
* Numerous mistakes about the technical details of S3 (i.e. the claim that all S3 objects are based on lists, that S3 doesn't support sub-classing, , etc) render the dismissal of S3 in this section particularly grating.
  + We are very sorry to read this and especially that you found this section “grating” and we certainly don’t dismiss S3.
  + With respect to the comment about ‘lists’, we acknowledge that we have used terminology that conflicts with usual R terminology. By typed lists we refer to object classes as named structures (<http://adv-r.had.co.nz/S3.html>), our terminology refers to a structure being similar (but not identical) to a list with added meta-information (such as described here <http://adv-r.had.co.nz/Data-structures.html>).
  + In regards to “sub-classing”, we write that inheritance in S3 is possible but is not “well formalised” as per Chambers (2014).
* The class hierarchy of `distr6` also feels fairly arcane to me; especially the necessity of wrappers. This could either be an artifact of the presentation in the paper, the design choices themselves, or my own limited understanding.
* I should add that I really dislike the use of decorators; an OOP design concept that is present nowhere else in R and is totally foreign to users and only serves to hide functionality from users.
* Using OOP "designs patterns" in my opinion is very often a bad idea, as they are very hard to do right. The design patterns may be done well from the developers POV in `distr6`, but from a user perspective there's too much going on.
  + distr6 was designed alongside the authors of distr and after discussions with the authors of Distributions.jl. We also had discussions with end-users, in particular those making distributional predictions, which was our initial primary use-case. Therefore, all decisions regarding class hierarchy, decorators, design patterns, wrappers, etc. were made after extensive discussion and design.
  + Whilst we have not received any other negative feedback about the actual design, we acknowledge the reviewer’s opinion and as such have scheduled internal discussions about the package design.
  + Regarding the point about “too much going on”: We believe the abstraction principles in the design provide a fairly simple (relative to the R6 paradigm) interface for the majority of users and we agree that the extra features are unnecessary in simple use-cases. However, in modelling of distributions, or manipulation of several higher-order distributions, we believe the design patterns improve the package performance significantly.
* 4. Regarding the design principles, D2 may make sense internally but it certainly doesn't make a difference at the user level. Users don't care how the package arrives at an answer, only that it is correct. Perhaps this choice is motivated by compute time? Anyway I don't really follow.
  + This point again relates back to the motivation/purpose of the package. In predictive modelling or in the more complex setting of auto-ML, there can be a significant difference between numerical and analytical results as well as run-time in creating these results. Therefore, for our primary users this is indeed an important design principle, hence it being high up on our list.
  + We have added to the motivation of this principle to make this clearer.
* D4 I found particularly problematic. The right-to-left internal priority system will almost certainly lead to user level errors. Some additional messaging system to let the user know which parameters are being used to specify the model would go a long way.
* In particular, I think it should be an error (or at least a warning) to pass more parameters to a distribution constructor than are necessary to identify a distribution.
  + Thank you for this comment and having discussed internally we have implemented a more robust system as suggested (<https://github.com/alan-turing-institute/distr6/pull/241>). This is now reflected in the paper.
* With respect to D6, I think adding an optional S3 interface to the R6 system is too much. Having 2-3 ways to do everything clutters the user interface -- I strongly suggest an interface where there is a single canonical way to do things.
  + Thank you for this suggestion and again it is one we have discussed internally in detail. Our decision was to enable the interface but not to actively ‘advertise’ it to users. We have found that in this way, the majority of users are happy with the R6 interface however several users have given very positive feedback about the option.
  + We have opened a new discussion on GitHub to get more recent feedback from users (<https://github.com/alan-turing-institute/distr6/discussions/239>).
* 5. `distr6` repeatedly claims to be "scalable", but the only benchmark it presents is a contrived example to measure the overheard of various dispatch operations.
  + We would like to address these two points separately.
  + By scalability we refer to the ability of the software to grow without minimal overhead. In this sense, the use of wrappers, decorators, and a clear class hierarchy enables the package to be highly scalable and extensible. We have added in clear definitions for both these terms when they are first used.
  + Regarding our “contrived example”, the benchmark referred to is independent of the package implementation and is specific to the object-oriented comparison in the paper. Interestingly, this would be the first published benchmark of object construction time in R that we are aware of, despite it being a trivial benchmark.
* 6. I would also appreciate an example of how the Kernels in the package simplify some previously difficult computation.
  + Thank you for this comment but it is unclear to us what “computation” is referred to here.
* I also didn't follow the difference between the product vs paired sampling modes.
  + We have extended the discussion and added examples to make the difference clearer both mathematically and in the package.
* 7. Is there a need to programmatically query distributional properties? Why can't this just live in the documentation? `set6` does seem like a useful contribution here but again I find the design choice odd.
  + We include this as firstly:
  + It is `usual’ in class object-oriented programming and moreover R6 actively encourages this by allowing the traits of a class to be queried without construction.
  + It is actually essential programmatically for several methods, for example getting the distribution support.

**Reviewer 2 (“2-review-3”)**

* Packages distr and also mistr provide extensive (monotone) transformation frameworks for the defined random variables that allow to define and work with much more complicated distributions. Similarly, the convolution framework of distr is more general and allows for much more complicated transformations as distr6. It should be clearly stated in the article that monotone transformations are not offered to avoid misunderstandings that may occur by users of distr and mistr (mostly if the word ‘upgrade’ is used).
* Truncation: The current implementation of the truncation wrapper is limited only to left-open intervals, which is a strong limiting factor when constructing such distributions.
* While the quantile function of a mixture distribution is usually not available in a closed form, it could be still evaluated using numerical inversion of the cdf:
* The efficiency of some functions could be improved just by simple reordering (with small adjustments).
* The pdf evaluation of a convolution in distr6 is performed using numerical integration of the convolution formula. However, there is no way for the user to adjust the tolerance and subdivisions of the integrate() function, which could be a desired choice.
  + We would like to thank the reviewer for the extensive review of both the paper and the package itself. We agree with all of the above points and in fact have already identified these in GitHub issues, projects, and pull requests:
    - <https://github.com/alan-turing-institute/distr6/issues/224>
    - <https://github.com/alan-turing-institute/distr6/issues/202>
    - <https://github.com/alan-turing-institute/distr6/issues/160>
    - <https://github.com/alan-turing-institute/distr6/issues/159>
    - <https://github.com/alan-turing-institute/distr6/issues/135>
    - <https://github.com/alan-turing-institute/distr6/issues/65>
    - <https://github.com/alan-turing-institute/distr6/projects/5>
    - <https://github.com/alan-turing-institute/distr6/tree/scale-wrapper>
  + In order to make it clearer in the paper that we acknowledge there are areas where the package currently falls short, we have added a section for “Future Updates” to highlight these areas.
* References: Some of the references seem to have outdated/non-functional links and doi. This could have occurred because of a delay in the publication phase.
  + Many thanks for pointing these out, the links and doi have been fixed.