Point-by-point response letter to comments on "mistr: A Computational Framework for Mixture and Composite Distributions"

In the following, we repeat all comments by the reviewer in italic font and provide comments and explanations as well as elaborate which changes were incorporated in the manuscript to address the issues raised.

We hope that the revised version of our manuscript is suitable for publication in *The R Journal*.

Issues in the Paper

1. Motivation through more accurate fitting of financial data:
   Having lauded the paper in general, the motivation for mixture distributions through modeling financial log returns is a bit too bold for me.

   It is true that often real world data shows evidence that assumptions of the Black-Scholes world (i.e., independent log-normal returns) are violated. The argument is more subtle though: regarding statistical evidence that for instance data shows higher kurtosis than a Gaussian distribution is less striking than it seems, given that empirical kurtosis is (a) highly variable (its variance relies on 8th moments) and (b) highly prone to outliers, so often this evidence will rely on just some very few data points. Allowing for the fact that only the majority of the data is well fit by a normal distribution and that there may be outliers which are non-normal, but also from a possibly unpredictable and fluctuating, hence hard-to model distribution, things get much less clear-cut. In particular if one bases a decision on a Gaussian model not on goodness-of-fit criteria alone but rather on some underlying stable aggregation mechanism which suggests some central limit theorem phenomenon, a modeling with normal distributions may be less naive than it seems on first glance.

   This is not to say that mixtures are not a welcome tool for modeling and that they really can help improving on normal modeling. In particular, if one manages to capture a group of outliers which can be modeled homogeneously by one distribution, a mixture is an excellent way to do so.

   Risk management with its focus on the tails of a distribution is more delicate and a much better motivation for mixtures than is global goodness of fit: The CLT argument for the normal distribution does not cover tail behaviour, and hence particalr mixtures like the PNP distributions provided for in **mistr** *indeed are very useful.*
   RESPONSE: We thank the reviewer for pointing this out. We have rewritten the motivation part and put more emphasis on the risk management application.

2. Related work:
   The authors seem to have missed some related prior work. First of all, in other domains, composite distributions are often called spliced distributions, which should be mentioned, so that (a) the paper can also be found by those using the term "splicing" and (b) the paper could also be related to work from these domains.

With this in mind the authors should relate their approach to the one in R packages **evmix**, **OpVar** *and* **RelRs**. *According to the CRAN taskview for distributions, for general mixtures one could also consider R packages* **gamlss.mx**. *Of particular relevance seems to be the following paper: Reynkens, T., Verbelen, R., Beirlant, J. and Antonio, K. (2017). "Modelling Censored Losses Using Splicing: a Global Fit Strategy With Mixed Erlang and Extreme Value Distributions". Insurance: Mathematics and Economics, 77, 65-77.*
RESPONSE: Thank you for all these references. We have included them to the manuscript.

3. Infrastructure for mixtures and composites in R package distr:
The authors acknowledge prior work in R package **distr**, *but say that to the best of their knowledge "it does not support any tools to work with composite distributions". This is true in fact. Perhaps the authors could be more precise, stating that while* **distr** *does cover general finite mixtures of distributions through the generating function* `UnivarMixingDistribution`, *so far there is no corresponding generating function for composite distributions-you would need to write one yourself, using generating function* `UnivarMixingDistribution` *and method* `Truncate` *to condition distributions to splicing intervals.*
RESPONSE: Thank you for pointing this out, we have added the information to the paragraph.

4. S3, S4, and R6 and class design:
I would like to know a little more on why the authors decided to use S3 and not S4 or R6 classes. Which drawbacks do they see in S4 and R6? In particular the authors should be more explicit about base classes, respectively their strategy to hinder an unbounded growth of the number of new distribution classes.
RESPONSE: The main reason for the choice of S3 was the idea of a light-weight framework. The S4 class on the other side, felt unnecessary for the functionalities offered by **mistr**.

   For the case of R6 and reference classes in general, R6 is still considered as a possible future additional implementation, to allow "python-like" chaining when working with the distribution object.

5. Transformation framework:
The authors provide an impressive framework of monotone transformations and to this end, similarly to the distribution arithmetics available in **distr** *(which could be mentioned in the paper as related approach), overload mathematical operators* $*, +, \ldots$. *Still, the statement page 5, that "[t]he transformation framework currently allows for all standard monotone transformations (like addition, subtraction, multiplication, division, logarithm, exponential and monotonic power trans- formations)" might be misleading as to binary operators like* $+, -, *, /, \vee$, *as one one of the operands must be a numeric to work* $-X + Y$ *is not implemented if both* X *and* Y *are distributions.*
RESPONSE: Thank you for this important remark. It is now clearly stated, that in the case of binary operators the distribution can be combined only with numeric values.

6. Confidence intervals for QQplots:
The authors provide a very nice interface to ggplot infrastructure (which, for instance, **distr** *lacks). They should however clarify that the confidence intervals they provide for their QQ plot are pointwise and (as stated) asymptotic. For alternatives, see Almeida, A., Loy, A., and Hofmann, H. (2018). ggplot2 Compatible Quantile-Quantile Plots in R. The R Journal, 10(2), 248-261.*
RESPONSE: Thank you for spotting this. We have added this information to the paragraph. In addition, thank you for the great and recent reference. We have included it in the subsection as a possible alternative.

7. Issue of smooth transitions for composite distributions:
The composite distributions typically suffer from the fact that without special precautions the

transitions will not be smooth, so that densities will exhibit jumps at the breakpoints. This will induce undesirable instability effects at estimation once observations come to lie close to breakpoints, if these are only prespecified up to a location shift. The authors should comment on this.

RESPONSE: We thank the reviewer for pointing this out. The two models in **mistr** (PNP and GNG) contain the continuity conditions exactly to exclude these effects. However, we understand that **mistr** also allows to create discontinuous composite models and to use `d()` function to fit them. To ensure that this is well-known by the users, we have added a sentence to the "Data modeling" section.

Issues in the Implementation

1. Accessors to cdf, pdf, quantile, and rng function:
   The authors provide accessors to the cdf, pdf, quantile, and rng function of their distribution objects, conclusively named by the same prefixes `r`, `d`, `p`, *and* `q` *as* **stats** *implementations of distributions. This is handy and easy to remember, but also debatable; some prominent R programmers (not this reviewer) strongly object to such overloading practice and stipulate that this should even be forbidden in R. So far, this is not the case, and this practice is still allowed and used, e.g., in package* **distr**. *Still, even today, this comes with some drawbacks:* **RStudio** *and, even more drastically,* **Jupyter IRKernel** *catch away calls to* `q` *before feeding the console input (or, worse, in case of* **Jupyter IRKernel**, *whatever R code within package namespaces) into the standard R interpreter. So exported S3 method* `q` *from package* **mistr** *will show unexpected behaviour in these settings. Hence, to make code safe in these environments, one should have aliases, at least to* `q`.

   RESPONSE: Thank you for mentioning this not very well-known problem. Luckily for **mistr**, this problem has already been spotted during the development and mistr contains aliases for all the main functions: `mistr_p`, `mistr_d`, `mistr_q` and `mistr_r`. This information and the corresponding problematics are available in the vignette of the package and in the help files of the mentioned functions.

2. Moments for Pareto and friends:
   Some risk quantifications like expected shortfall and expectiles rely on existence of moments. Now typically for Pareto, GPD, and other heavy-tailed distributions, moments need not exist. The code in **mistr** *so far does not issue any warnings in such situations, but should.*

   RESPONSE: Thank you for spotting this. We have added the necessary warnings to the `risk` methods.

3. Quantile function:
   The authors provide a quantile function which is obtained by numerical inversion of the cdf. This is fine. However, they do this inversion (a) in a non vectorized form through calls to uniroot and (b) they do it on runtime. An alternative producing a vectorized function at generation time would be to evaluate the cdf on a prescribed grid of q-values, obtain p values, use these p values as x-grid, and the q values as y-grid in a call to `approxfun` *and return the function (and potentially smooth out values). This is helpful as to speed and is the approach pursued in* **distr**.

   RESPONSE: Indeed, the numerical inversion of the cdf is used to calculate the quantiles of mixture distributions since the closed form of the quantile function is in this cases usually not available. However, after careful deliberation we have decided to use the current implementation, as it offers more control over the accuracy than the method using `approxfun` and for smaller vectors the speed is not an issue. Additionally, the evaluation on runtime goes more with the ideology of **mistr**, as only minimum amount of preparations is done when a distribution is created or transformed, since it is not clear what purpose will the distribution serve.

On the other side, we understand the need for a faster quantile function in case of mixtures. For this purpose we added a new function called `q_approxfun`, which takes a distribution object and returns a function that uses `approx` to approximate the quantiles as described above. This function will be released with the next CRAN release of **mistr**. Thank you for pointing this out.