*#9*

*The author’s state, “we have been working on adding functionality to the existing R-package ssdtools to incorporate this capability”. In what capacity are the authors working to add this functionality?*

KM I think we should retain the sentence but alter the text a bit.. “We believe statistical mixture modelling has an important role to play in ecotoxicology and  
accordingly we hope to add this functionality to the existing R-package ssdtools in the future”. We discuss SMM and it’s advantages, it would be weird if we didn’t mention that we hope to add this functionality into the ssdtools package.

# 12

*this manuscript mainly focuses on summarizing the previous status and the recently introduced tools, the implications of the new toolbox are lack, which reduces the contribution of this manuscript on SSD evolvement*

AT I think a major point we need to highlight is the ability to collaborate using R. For example, the ability for someone to figure out multimodality and then make a pull request. Also, I noticed that there is another Shiny App out there – Shinyssd. starting page 23, should this app be reviewed as well? Shinyssd provides additional functionality in terms of data filtering not included in shinyssdtools. It seems that Shinyssd may be developed specifically for larger databases but it could also be useful for smaller databases. See: <https://www.researchgate.net/publication/333458070_shinyssd_v10_Species_Sensitivity_Distributions_for_Ecotoxicological_Risk_Assessment> for download information

#19

*To say they focused on two software packages, SSD Tool Box and ssdtools, is an overstatement. The discussion of both packages is very limited. I was expecting more.*

DF As I commented elsewhere, we can point out the we (Rebecca) will be preparing a follow-up MS looking at software, performance and case studies.

#21

*The list of software is interesting, but they do almost nothing with it. There is no useful discussion of what these packages do, how to use them, their effectiveness or limitations. Either develop this, put it in supplementary material, or omit it.*

*While the authors present a table listing nine software packages for SSD fitting, they do not present a detailed review of them. Rather, the focus on just two: the ssdtools R package and the recently released SSD Toolbox (Center for Computational Toxicology and Exposure 2020). It becomes clear that these two packages are highlighted because they offer model averaging, which is some thing they endorse.*

KM Thanks Joe! Some things to add for SSD Master.. Censoring: has an option to truncate- so excluding “tolerant” species in the right tail and only fitting the models to the remaining species. HCx: 1,2,…,98,99. CIs: classic parametric approach. Programming language: Excel,visual basic. Under URL, perhaps its worth noting that it is only available upon request. Analytical method; suggest to spell out terms in a footnote.

AT Perhaps we need to be more explicit about which software tools allow the flexibility to deal with small datasets… see my suggestion in #20.

GB I raised this issue: I note that in the paper we variously refer to the ssdtools Shiny app and the shinyssdtools app. This is a bit confusing to the reader. Should we not adopt a standard format for this?

Joe and David both agreed that the latter is appropriate so need to modify the text accordingly

# 25

*The authors express strong opposition to the use of compiled code, such as in the recent SSD Toolbox released by USEPA. So long as the methodology and algorithms are clearly documented, their opposition to compiled code seems misguided. The use of such code limits the possibility of manipulating the code to influence the results for a given study and that should increase, not decrease the confidence risk assessors have in the results. The concern expressed about possible costs to run such programs is legitimate, but as they note, a free tool is available to run the complied code used in SSD Tool Box.*

CS Yes, but even if compiled code is documented, how do you know that it was implemented properly? The beauty of open source is that you can actually see what was done and check your self. If there was some sort of “certification” of compiled code, then the objections to compiled code are less.

Secondly, compiled code is “static” and tools change. In some cases, it is not even possible to find the original source code (e.g. banking systems written in COBOL where the source is long gone or does not match the compiled code). With open source, this is not an issue.

DF Perhaps the reviewer struggled with our “jargon” statement about not being able to “look under the hood”!!

This why we are primarily opposed to locked up (compiled) code. Cogent arguments in favour of the open source model are made elsewhere in the paper as is a discussion about the pros and cons of different deployment modes. We didacknowledge that, it may be attractive from a regulatory point of view to lock the software down and we didacknowledge that SSD Toolbox will run without having to buy Matlab. The issue we highlighted was that the availability of the free enabling tool which is required to run SSD Toolbox is totally at the discretion of a third party.

#26

*It is not clear how the ability to edit open source code enhances harmonization of approaches. It seems reasonable to think new approaches or modifications could ultimately improve the science by demonstrating improvements, but it also seems clear that, at least in the short term, this will increase the variety of approaches. It would open up the possibility of some manipulation, in good faith or bad, by an applicant for product approval. This would be overcome to some degree by a requirement to provide the code to the regulatory authority reviewing the submission. It would also put greater demands on the regulators and not all regulatory authorities are blessed with the required expertise. To be sure, on page 24, they do acknowledge that “there is likely to be demand for both modifiable and “locked” (i.e. compiled) code.”*

CS “Locked” does not have to mean compiled. For example, a methodology is implemented and tested and generally approved in R (package version 2.3.4). You just specify that all analyses must use version 2.3.4. While open source and modifiable, the package is “locked” and not directly modifiable.

If you really are worried about security, you have the same issues with compiled code. How do you know that the output you are provided with actually came from the compiled code vs. a “fake” code.

The only “cure” is to publish data and reproducible code so other people can “check” your work. In particular, don’t reimplement existing methods (i.e. don’t write your own package to implement an algorithm).

RF It is helpful to have the first part of this comment for discussion, which was:  
P 9. They state that the use of an open source licencing and development platform overcomes all the deficiencies they identify in compiled code and the variety of software and approaches currently being used because open source code fosters a collaborative environment that enhances harmonization of approaches and facilitates the timely dissemination of the most up-to-date methodologies.

Making code open source and hosting this on a collaborate platform like github means that everyone has equal access to the same base code and can use it as they like. This fact would inherently increase the likelihood that multiple jursisdictions will in fact use the code – potentially building onto it their own needs. These modifications and updates can be absorbed into the original code with things like pull requests, ultimately improving the software for everyone. Different jurisdictions will always have different perspectives and needs, thus a single universal method will be unlikely to ever be adopted. But at least some alignment of methods will happen naturally in a situation where everyone has access to the same tools, but those tools remain flexible enough to accommodate individual needs. This is only possible in an open source environment.

We have a be a little careful here though, because there can be open-source compiled code. It might be we need Jo to carefully word some of these parts of the text. The key is that it is open source, and collaboratively hosted (or whatever the right word is). From there it can be compiled (for example there is compile Stan code in R) or not. Perhaps better words are propriety? or some other way of referring to restricted license commercial software. David’s other points about the user-base and choosing a platform that is widely used, as well as being free and widely available is key to this argument as well.

#33

*Discussion of ssdtools software. He (sic) authors note that while it is possible to fit a single model, the emphasis in this package is on model averaging. The distributions available for this purpose are log-normal, log-logistic, and gamma by default and optionally Burr III, LogGumbel, and Gompertz. I have found the first four of these useful over many studies, the last two much less so. It is good that these distributions are available, as the original BurrliOZ software that offered Burr III and some others is no longer supported and the SSD Master which offers Gompertz among others, is deeply flawed.*

CS Agree and thanks the reviewers for their comment.

DF Is this just for noting? Burrlioz 2.0 still fits Burr provided N large enough.

#34

*Discussion of SSD Toolbox software. Log-transformed data can be fit by six distributions: normal, logistic, triangular, Gumbel, Weibull, and Burr. It is also interesting that four different fitting methods are available. These are maximum likelihood, moment matching, cdf linearization, and Bayesian methods. The authors question the inclusion of the triangular distribution as having tail characteristics not often seen in practice. I agree with that criticism. It would be valuable to expand this section to indicate what these methods do. At present, they offer little beyond dismissing linearization and favoring maximum likelihood. It would be really interesting, for example, to have a discussion of what Bayesian methods offer.*

CS Some wordsmithing needed. Maximum likelihood, method of moments, or non-linear fits using the cdf are ways to estimating the parameters of the underlying distribution. This is a wealth of statistical literature on the pros and cons of each (computations simplicity, unbiased estimates of parameters etc) and I don’t think we want to go there.

Cdf linearization is method of finding se for the quantiles. This is different from fitting the distributions.

Bayesian methods are overlaid on maximum likelihood methods. Key advantages is the ability to incorporate prior information (e.g. you may have prior information based on other chemicals), or for more complex situation (e.g. individual endpoints have uncertainty, are censored) for which MLE are numerically complex to define (require integration) but MCMC does a numerical integration.

DF Given we’re already over the word count and that we’ve been asked to expand on qute a few topics, I suggest we just point to some references on Bayesian methods. Fox (2010) is already in our references but we could add one or two more.

#33&34

*The discussion of these two SSD packages is very brief and gives only the barest indications of functionality. Indeed, the summaries provided here are essentially all they offer. It would be helpful to discuss what options exist, if any, for omitting one or more available models from use in the averaging process. It is entirely possible that one or more models will deviate so far from the data as to render them not just useless but actually detrimental. I have observed this with SSD Master on many occasions and with other packages on occasion. Reliance on the stated AIC criteria alone for down weighing these models may not be sufficient. It would have enhanced the paper to have evaluated this issue for each package.*

RF I am a little confused why they think there are two packages? DO they think ssdtools and the shiny ap are two separate packages? If so we can just clarify the wording in the MS, and we only need to provide more information for the ssdtools R package. The shiny is really just an interface.

DF Agreed. Plus I think there are other points of clarification.

1. both ssdtools shiny and SSD Toolbox are flexible wrt model inclusion/exclusion;
2. the whole point of AICc was to down weight ‘detrimental’ models!;
3. with due respect, I wouldn’t be using SSD Master as a yardstick – it was a useful tool but limited by Excel functionality (or lack thereof).

DF It might be worth adding to the Table 1 an indication of the distribution(s) each uses. Obviously for ssdtools this would be “various” – but where there are only a couple they probably should be listed (e.g. BurrliOz). I haven’t used SSD Master but from the text in our paper it looks like you can fit a range of distributions, but it does not use model averaging. I’m not sure what evidence the reviewer bases his statement that “*Reliance on the stated AIC criteria alone for down weighing these models may not be sufficient.*” So far with all the examples I have tried, AICc seems to be doing a pretty good job. As David says in point 2, the whole point is that it down weights these detrimental models. We could reference Schwarz and Tillmanns (2019) as evidence of the stability of the AICc based model averaging method. Or a pers com from one of the Canadians that have has used the method in their derivations – have any of you guys found the AICc to be insufficient in terms of identifying bad fitting models? My simulation study (too complex to include in this paper) suggests it may start to fail to select the “correct” distribution with small sample sizes, but with real data we do not know the correct distribution in any case – thus a model averaged version is probably safer than arbitrary selection of a single wrong distribution in any case.

As an aside – From what I can gather on their instructions the contents of tables are not included in the word limit, which means expanding Table 1 with more detail would provide a means of expanding our ‘review’ of the available methods without necessarily blowing out the word limit.