Reviewer 1 Comments

An apparent advantage of the implementation of flexplot is that it is built on ggplot2, which allows it to leverage ggplot2 features (themes, perhaps scales also?) w/o having to redo these in flexplot. However, as illustrated in the code on p. 22, doing something as simple as changing the size of points turns out to be daunting, and would be a limitation to using flexplot. I'm not sure how to handle this in the paper. But a suggestion would be to provide for more a few more options in the code or global flexplot option()s.

> Response: Yes, themes are easily changed, as is the scaling of the axes. And, yes, some options are more difficult to change than others. I have to strike a balance between complexity and ease of use. Flexplot already has 21 options and I don't want to add to that. I have, however, added a new function called modify_points that does exactly what the analysis in the paper does, but with a much easier interface. See pages 21-22.

This paper does not distinguish between **analysis** plots, for the analyst that should be very easy to produce, try out different things, etc. and **presentation** plots that often need to be simplified, polished, annotated, labeled, etc. for clear communication. There is much greater need for control of graphic details in the later. Somehow this should be mentioned.

> Response: This was a good point. I have added a footnote (Footnote #1) that makes that distinction and states that flexplot can be used for both purposes (though plots for presentation will likely require some tweaks).

Maybe just me, but "visuals" is the more general term used by graphic designers, infographics folks, whereas what you are really talking about is "graphs". "Non-visuals" (p. 29) are numerical displays.

> Response: I reviewed the document and replace "visuals" with "graphs" throughout.

NB: I had enormous difficulty trying to print this ms. From Adobe Acrobat. The symptom is that it gets stuck (at Fig. 12) in the 'flattening stage' and has to be aborted. This has to do with the complexity of the figures, perhaps a transparency issue. For re-submission, please try to tame the PDF that you submit.

> Response: I'm not entirely sure how to do that. I made an attempt by compressing the image using photoshop. Hopefully that solves the problem.

p. 3-5: Intro – The link between use of graphics and the replication crisis seems week. More generally, a fair bit of the introduction on the advantages of graphics could be compressed. Better to focus on what is new here – advantages of flexplot over other software.

> Response: In the current revision, I have de-emphasized the replication crisis. Instead, I mention it in the introductory paragraph as a bridge to why graphics are important. I have also condensed the section on the advantages of graphics. See page 3.

p.7: code examples (and elsewhere)—hard to read because comments get pushed to another line. Use a smaller font, abbreviate, etc. to make more readable. You can probably tweak the Rmarkdown to get better results.

> Response: Excellent suggestion. I modified the code to produce smaller text for code examples.

p.9, L6-8: R notation y ~ X1 + X2 is not an equation. It is a model **formula**, that makes certain implicit assumptions, e.g., the intercept is included and there is an error term.

> Response: This was a good point. The text has been changed. See page 8.

p.10, L2-7: In R, plot() is a generic function with S3 methods for different classes of objects. That is a great strength, and makes plot() far more flexible than you imply.

> Response. This is also a good point. I now clarify that my comments were about the base plot function, and I also mention how plot is an S3 method that can be used for various classes. See Footnote 2.

p.11, L3: The cowplot package is now superseded by patchwork.

> Response. I wasn't aware of patchwork. This is an excellent package that makes it much easier than cowplot. All cowplot uses have been replaced with patchwork.

p.16 (& elsewhere): Avoid cute comments in code examples

> Response. I feel like this suggestion is a matter of taste. Removing the comments doesn't affect the usability of flexplot or the message of the paper. Some people enjoy comments that inject personality. Some do not. However, if the editor feels the comments in the R code distract from the paper, I will remove them.

p.16: Association plots – arguably much better would be a mosaic plot.

> Response: I decided to go with association plots for two primary reasons. First, mosaic plots require people to make judgments about *area* which we are not good at (Cleveland & McGill, 1984). Second, creating mosaic plots would require an additional package dependency (technically *import*), which I am wary to do. (I want to keep flexplot as lightweight and robust as possible).

Fig. 18: One way to avoid some of the problems in the top panel would be to make the plot much larger.

> Response: This is true and I have added a parenthetical statement saying such (page 17).

p.19: added.plot() – Perhaps mention & compare with car::avPlot, avPlots. A weakness of your scheme for is to use the last variable in the model formula as the one to show the plot for. (a) this seems counter-intuitive; arguably it should be the first. (b) Worse is that if you want more than one for a given model, you have to keep rearranging the model formula. Better: provide an optional argument, e.g., x= a number referring to the position of the variable in the model formula.

> Response: I added a note about avPlots in car (saying how avPlots use base plot, while added_plot uses flexplot). See Footnote 3. As for the second suggestion, that's a great idea. I modified the function to allow users to specify the location or name of the variable they wish to have on the x-axis. See pages 19-20.

p.21: Footnote 1: This is fine, but you should perhaps make the actual code for figs available as a supplementary file on the web.

> Response: This was a good suggestion. I added a note about how the source file can be found on Github (Footnote 4).