Response to second review

October 10, 2016

Reviewer 1

(1) The paper seems to me inappropriate for the JCGS. Because it focuses purely on software, I think that (once the other issues are addressed) it would be more appropriate for the Journal of Statistical Software or the R Journal.

---

(2) Although the software looks promising, unless I've missed something, it currently has very limited functionality – indeed, not enough to sustain most introductory statistics courses. This isn't a permanent limitation, however, since the authors have created interesting infrastructure for extending the software. In its current state, the software doesn't quite look ready for prime time.

---

(3) The authors ignore other (and at present, much more capable) GUIs for R – the R Commander and Deducer. Since these are the most direct competitors to intRo – more so than, e.g., Excel and JMP – it seems obvious that the authors should address similarities, differences, advantages, and disadvantages. One obvious difference is that these other GUIs run on individual computers rather than being web-based, and it would also make sense to address more generally the advantages and disadvantages of the two approaches. (For example, in my tests of intRo, I noticed slow response, even with very small data sets. Of course, it's hard to know to what to attribute that – it could as easily have been a network issue as slowness at the server end.)

Reviewer 2

(1) While the authors have made a pretty good fist of addressing everything I asked for and it is pretty well good to go from my point of view. But I think they can still strengthen their case and their appeal somewhat. I think a title like the following would suggest better the new focus of the paper: "Customisable introductory statistics software via intRo"

We agree, thanks for the suggestion. We have changed the title to "Designing Modular Software: A Case Study in Introductory Statistics", which we feel highlights the customizable aspects of intRo.

(2) Also there are 2 big ideas in here about software that go beyond the particular application: extensibility (the ability to "easily" add functionality) and customisation/simplification (the ability to easily hide functionality) and a way of implementing these. I think these 2 elements and their advantages should be highlighted from the get go, starting with the abstract. The advantages of extensibility are obvious but the desirability of being able to simplify the view of software a particular audience gets needs highlighting. (Also the idea that they enable this by the way in which the instance of the software that will be served up to the user is assembled on the fly at run time.) Detailed comments follow. (Perhaps unfortunately, many draw on what I have said above.):

We have updated the abstract to reflect the new focus of the paper, which is writing extensible software through the use of modular programming within the framework of shiny. Additionally, we've highlighted the ability to remove functionality and present a

simplified view to the user in the abstract. The paper itself now addressed the "big ideas" of extensibility and customization much more thoroughly.

(3) Page 2 bottom: could note that having a large group of students install software on their own machines can be a nightmare with differing hardware configurations, versioning etc.

We added a sentence in section 2, page 5 top to address this.

(4) Page 3, last para pf Section 1 needs breaking up.

The content of this paragraph is no longer in the paper as a single paragraph, but rather redispersed into multiple sections.

(5) Page 4, Note that Fig 1 is a schematic, not a view of the software.

We have added the verbiage "schematic" in the text and figure caption to note this.

(6) Page 7 last line: no need for "introductory" here. These ideas apply generally.

Good point. We removed the word "introductory" from the sentence.

(7) Page 8 ff (also page 10-11): I think this story would be better spun in terms of someone writing an incorporating a new module (a few tiny wording changes).

We changed the wording to indicate the example walks through the process of creating a module, as well as including it.

Associate Editor

(1) The authors have responded to some of the reviewers' comments, but there is still more work to do. The motivation for intRo is as a simpler interface to R (for teaching statistics AND generating interest in R), but this should just be background information for this article. There are many other R GUIs that are MUCH more developed for teaching statistics (that also generate R code as well). The features of intRo that are for the student/user are less interesting and do not add anything beyond existing software. The main focus of the article should be on the novel aspects of this work, which is the modularity/extensibility - the features of intRo that are for the teacher/developer.

We have refocused the article onto the novel aspects of modularity and extensibility by reqriting the introduction and reframing the article as a case study in modularity. Accordingly, we have moved details about intRo as an interface to R to section 2.

(2) The comparison with existing software (Rcmdr, Deducer, iNZight, Shiny) should be more in-depth, but focused on the modularity/extensibility issue.

We have added comparisons to more software, including Radiant, to the introduction. Within these comparisons, we focused on extensibility, modularity, reproducibility, licensing, and the functionality of each piece of software. We have also included a table that summarises these comparisons.