KEEPING STUDENTS ENGAGED WITH SHINY INTERACTIVE TOOLS

Justin R. Wishart
Department of Statistics, Macquarie University, NSW 2019, Australia
justin.wishart@mq.edu.au

Shiny, the new web application framework for R has exploded in popularity in recent years. This framework facilitates the creation of web-based applets that are interactive and thus can be used as a modern supplement to teaching techniques. The applets only require a modern JavaScript enabled web browser that allows students to use them on any device, including mobiles and tablets. Students then have the freedom to interact with them and gain an in-depth understanding of the underlying statistical concepts at their own pace. The paper will describe some applets integrated into an entry level first year University course on Statistics that has a large cohort. Possible strategies to keep students engaged with the Shiny apps are discussed and methods to facilitate student self-study.

BACKGROUND

A great way to demonstrate data analysis and data visualisation concepts to students is by extending use of the digital learning environment. Applets or small applications that achieve this goal can be created efficiently by any person familiar with the statistical language R (R Core Team, 2017). This is facilitated by an R package called shiny (Chang et al., 2017). In particular, the Integrated Development Environment (IDE) of RStudio (RStudio Team, 2017) greatly decreases app development time as the shiny package was authored and maintained by the RStudio developers themselves. Accordingly there are improvements and features that are integrated in RStudio that make applet creation much more efficient than using base R.

The applets created by Shiny are actually a combination of JavaScript and HTML front end to display the desired situation to the end user and an R server that handles the statistical computations. This combines the computational power of the statistical programming language R with the interactivity of the modern web. Since the applets are web based (but can be hosted locally if desired) it allows users to view and interact with them on any device that has web browsing capability from traditional desktops and laptop machines to smartphones and tablets. The apps being device independent allows students more flexibility to learn, at their own pace or convenience.

The interactive scope of the apps is wide with any HTML or JavaScript widget being possible to integrate into an app. So users can modify the data or analysis method and dynamically view the results or visualise the change in output. As alluded to earlier, these applications are hosted on the internet so students can use and interact with them at their leisure and own pace. Thus, improving the balance between digital and physical teaching environment in undergraduate teaching and increase flexibility in timing and delivery modes.

The apps under consideration in this paper were deployed using the shinyapps.io platform. This is a service provided by the RStudio team that provides a web address for the application and server computational time to conduct the necessary computation in R that the app requires. Their free shinyapps.io option has a modest computational allotment while their paid options allow greater computational time and other benefits that will be touched on later in this paper.

In the remainder of the paper, the focus is on the use of Shiny apps for an entry level core statistics unit offered at Macquarie University, STAT170. This unit covers basic statistical topics and competency with population sampling, hypothesis testing and statistical estimation of population parameters for i) Single population means ii) Comparison of two population means iii) Simple linear regression of two variables and iv) Chi-squared goodness of fit and independence tests . The students have no assumed knowledge of statistics and only assumes fundamental mathematical/arithmetic skills from their school studies. Many apps were created to covers these topics, the focus is only on three in particular in this paper. These apps were implemented in the course to be an optional supplement to the core material delivered in lectures and tutorials. That is, it wasn't required that the students use the apps but they were demonstrated during the lecture and students could also use them during their learning process in their own time. A moderate amount

In M. A. Sorto, A. White, & L. Guyot (Eds.), *Looking back, looking forward. Proceedings of the Tenth International Conference on Teaching Statistics (ICOTS10, July, 2018), Kyoto, Japan.* Voorburg, The Netherlands: International Statistical Institute. iase-web.org

(around 20%) of motivated students used the apps throughout the semester as part of their regular study. Almost half the cohort used the apps during peak times when the assessment task was due.

There were two different types of apps that were considered to implement into an entry level first year core statistics course, STAT170. The first type being an app that demonstrates core statistical concepts such as sampling and variation. The second type being an app that facilitates self-study with randomly generated datasets and/or scenarios where students are required to select the best response from a set of multiple choice options. Feedback and hints are optionally provided to the user if requested.

ANIMATED DEMONSTRATION AND SELF STUDY APPLICATIONS

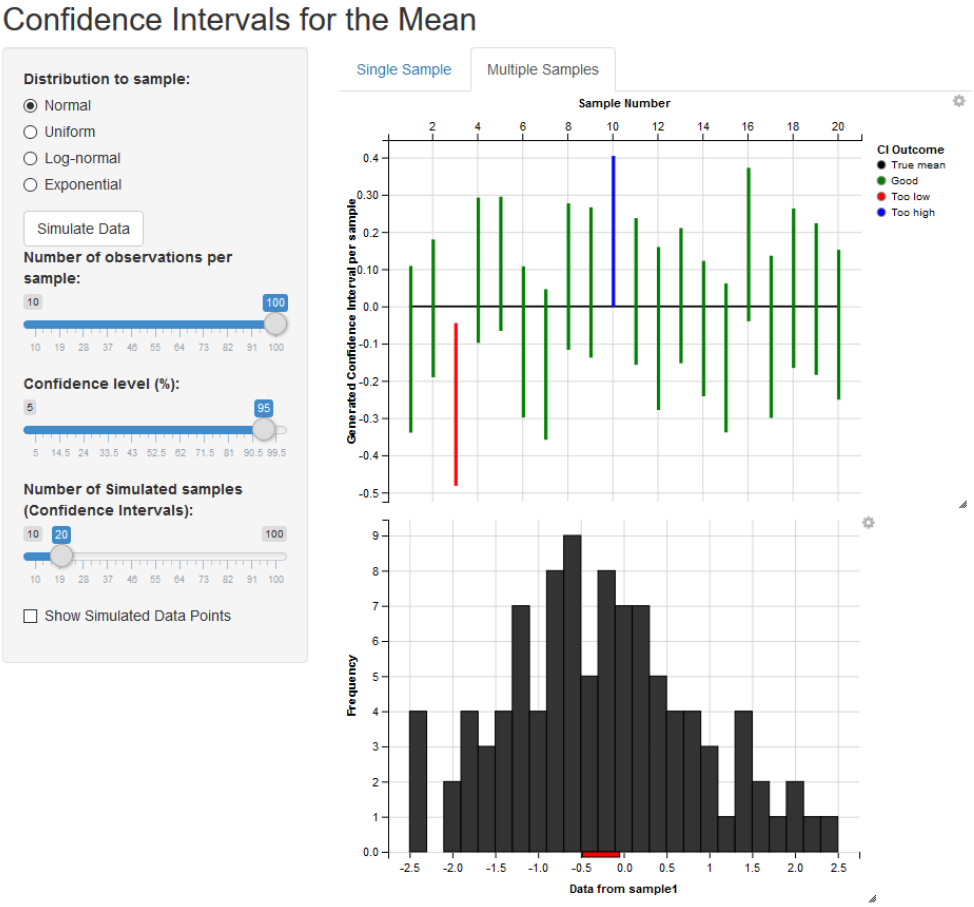

Figure 1: Confidence interval simulation application interface

Animated demonstration Apps

One way to keep students engaged is to visualise potentially difficult concepts. For the uninitiated, concepts of sampling and confidence intervals can be difficult to grasp. As such, applets have been used to help visualise these concepts and been implemented in other universities. See for example, the ShinyEd suite of packages by Çetinkaya-Rundel and Cohen (2014) at Duke University and Doi et al. (2016) with a suite of apps at California Polytechnic State University.

A classical way to display data using R is to use base graphics, lattice graphics (Sarkar (2008)) or ggplot2 graphics (Wickham (2009)). One of the disadvantages of using these standard packages is that the graphics were designed to be static and for print purposes. When used in a dynamic applet environment, they are slow as the plot is being redrawn and rendered. A graphical package in R that alleviates this shortcoming is ggvis by Chang and Wickham (2016). The ggvis package is based on Vega, a visualisation grammar that facilitates drawing web graphics making it perfectly suited to the web displays of and reactive nature of shiny applications. Consequently, the plots render quickly and have the option of an animated transition.

An example of the output is shown in Figure 1 (albeit only a static example in this document). The user is given control over the data to be simulated with control widgets shown on the left panel. The ggvis plots are shown on the right hand side and animate dynamically after the user has confirmed their input settings. Every time the simulate button is clicked, a fresh sample is generated and the plots update and animate to their new position seamlessly and quickly.

There is also ability to inspect data based on the cursor position. In the app shown in Figure 1, the data shown in the histogram defaults to the first sample of generated data. However, the user can inspect the other datasets by hovering the cursor over the sample's confidence interval in the top plot. In Figure 1 the third sample was inspected in this manner by hovering the cursor over the third confidence interval causing the histogram to animate to the third sample.

Self-Study Applications

Another key goal in the STAT170 course is to keep students engaged since the failure rate can get quite high with students that lack mathematical/statistical skills. A strategy to target this is to allow students the freedom to self-test themselves on concepts and review core skill sets covered in lectures and practiced in tutorial hours. To keep the self-testing fresh and interesting, datasets are re-simulated for each question. Hints and feedback can be provided along the way to help facilitate their use. In some apps, students can specify which type of data or scenario is simulated, in others it is not user specifiable and chosen at random.

Figure **1**Figure 2 and Figure 3 show two applications given to the students for their self-study purposes. Figure 2 shows an app that gives students practice in using the traditional statistical distribution tables to compute or bound probabilities. Namely, compute exact or precise probabilities in the standard normal scenario but produce bounds in the t-distribution or Chi-square distribution case. The incorrect distractor answers were generated by common mistakes and corresponding values nearby in the normal table. For example if the correct answer for a calculation was a tail probability with value p in the normal table, distractors would be values in the normal table near *p* selected at random and also *1-p* and its equivalent neighbours in the table.

Figure 3 shows an app that randomly generates two datasets with two numeric variables. The data is shown on a scatterplot and also available in csv format for students to download. Students are then asked to compute the correlation for each dataset and select the closest response from the options.

In the self-study apps, students are offered multiple choice responses with the standard radio button interface, a type of web based widget that allows one and only one response. When the app is initialised, no response is chosen. After a student selects a response, the student is given either immediate feedback or delayed feedback depending on the app. If the app is a single question such as the statistical table lookup, then the feedback is immediate in the form of a green tick or red cross respectively for a correct or incorrect response. If the app has more than one question such as the correlation app above, the student can check their answers with a button to see the feedback. See for example the 'Check answers!' button on the top right of Figure 3. When the student resets the app and a dataset is resimulated, the multiple choice responses are reset to have no filled in/chosen response. To ensure proper resetting of values on the R server side, this step requires some extra JavaScript code that is not part of the shiny package but the relevant code can be passed on to the interested reader if requested.

The apps can also offer incremental hints and feedback with the use of check boxes that are visible or invisible depending on the progress of the student within the app. These checkboxes can be toggled to display and hide the relevant feedback as specified For example in the statistical table lookup app (see Figure 2), a student is initially only shown the random variable (either $Z \sim N(0,1)$ or $X \sim N(\mu, \sigma)$) and the relevant randomly generated probability statement. Below the question statement there is a 'Hints and Feedback' section which initially is blank with only a single checkbox allowing the student to see the plot of the equivalent area under the curve for the probability question. After and only after the student has made their selection does the second checkbox appear that allows the student to see the full algebraic working to derive the final answer. Similar to the situation described in the previous paragraph, when the student is finished with the question, they can reset the app in its entirety where all the hints and feedback are hidden and the checkboxes and radio buttons reset to their empty/unselected positions.

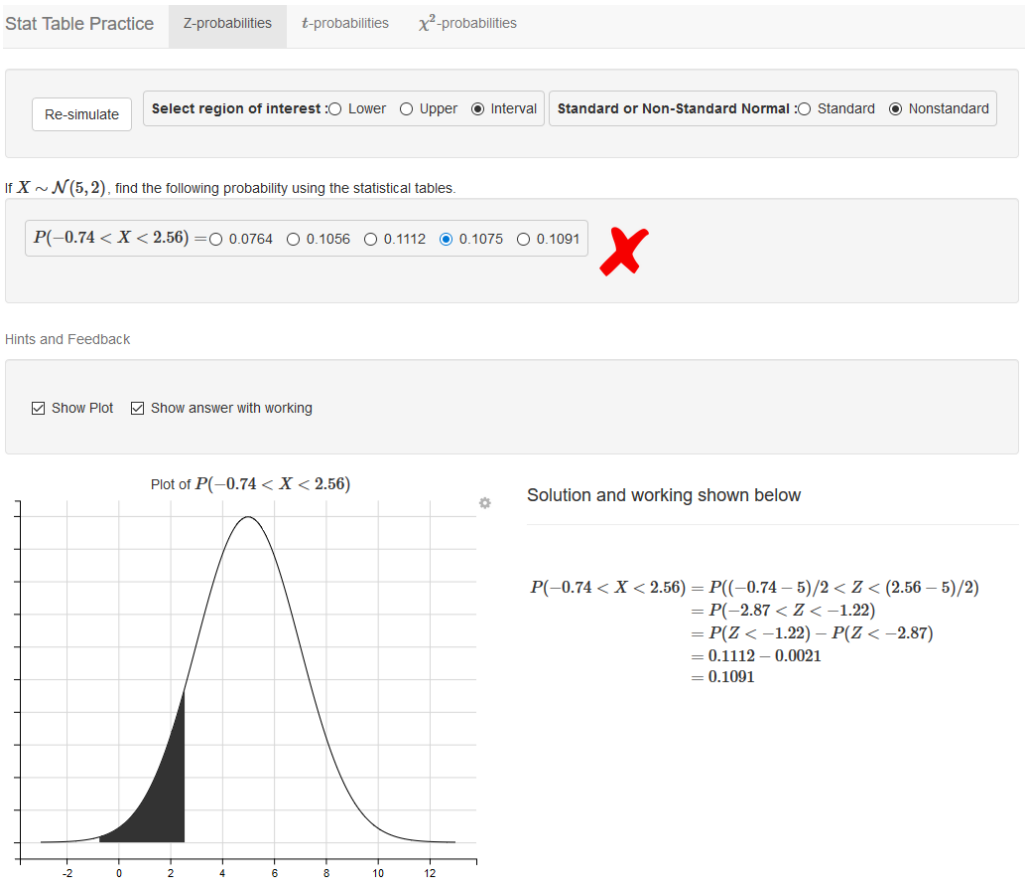

Figure 2: Statistical table lookup app showing simulation options and multiple choice interface along with incremental hints and feedback

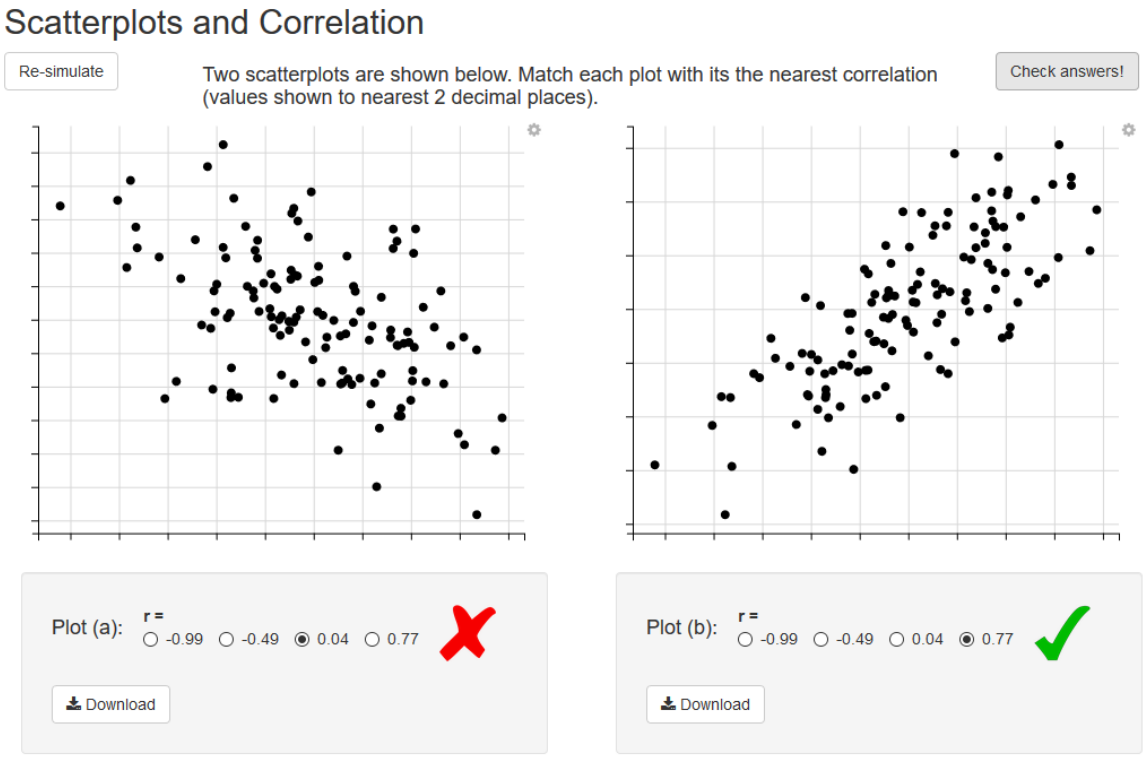

Figure 3: Computing the correlation of randomly generated datasets.

STUDENT AND INSTRUCTOR FEEDBACK

Students appreciated the use if the dynamic apps and commented that it solidified their understanding of the material. One such student commented

"… helped me immensely in preparing for the STAT170 midsemester exam (which went very well)."

Due to the various backgrounds of students in this cohort, students were offered both standard probability notation as shown in Figure 1 and simplified notation and sentences in their questions. Having the option of both was well received. It supported the weaker students that are apprehensive about mathematical notation. It also was appreciated by students that intended to do further study in an area that involves further statistical courses in their program. One student responded with feedback

"I especially appreciate the proper probability notation—I know it is being shared between classes anyway, but it's easier for me to understand when it's worked mathematically, and I'll be taking higher level stat classes."

A student and instructor independently suggested to add a competitive element where historical performance of the user is displayed within the app. Namely, the apps keep a log of the user's historical performance in the application and displays it on the screen. This is a possible extension that can be implemented since the apps themselves are hosted on the premium offering of the shinyapps.io platform.

One of the premium/paid features of the shinyapps.io service is user authentication. This requires users to authenticate with a username and password before being allowed access to the apps. In our case, we pursued this option to ensure the apps were being used by our students and not being used by the greater public. The main concern and impetus for doing this is that even the paid premium options have a limited computational time on the server and we did not want the apps to be unavailable due to exceeding this allowance.

A convenient byproduct of this authentication is the application can keep track of the user activity such as historical performance in the apps and logs. These logs can then be dynamically updated and relevant information displayed on the screen as the user progresses and be retained when the user leaves the app and then returns at an unspecified time in the future. Therefore a potential improvement of the app is to implement this feature and add the competitive aspect showing perhaps the longest streak of consecutive correct answers, the total number of correct answers and total number of attempts.

In the current offering of apps there is no engagement analytics of the app apart from usage numbers. That is, all that is currently recorded is the students opening and running the app for a period of time. There is possibility to record specific actions by the student via the use of google analytics recording where students click. This can be used to determine if students make common mistakes. However it is not implemented at this stage due to time constraints.

CONCLUSION

The applets here were designed to be an additional aid to facilitate learning and a guided practice tool with hints and feedback. The applets discussed here achieved this by using the web based graphical extensions to encourage users to become involved in the app and not be discouraged by unnecessary rendering times. The multiple choice applets were well received by students and instructors alike who commended the ability to attempt questions, check hints and feedback and then reset on a fresh sample.

In future offerings it is planned to incorporate the feedback, allowing more simulation options and to incorporate a competitive element to allow users to continue to use the applications and improve their score. It is also planned to incorporate more engagement metrics and analytics to record student performance on the apps and reconcile that with their performance in the unit via the use of google analytics. Combining this with the student authentication should provide a rich picture of the impact the apps have through their learning process and performance in assessment tasks.

REFERENCES

Chang, W., Cheng, J., Allaire, JJ., Xie, Y., & McPherson J. (2017). shiny: Web Application Framework for R [R computer software package]. Retrieved from https://CRAN.R-project.org/package=shiny

Chang, W., & Wickham, H. (2016). ggvis: Interactive Grammar of Graphics [R computer software package]. Retrieved from https://CRAN.R-project.org/package=ggvis

Çetinkaya-Rundel, M., & Cohen, B. (2017, July 15). *ShinyEd*. Retrieved from http://www2.stat.duke.edu/~mc301/shinyed/

Doi, J., Potter, G., Wong, J., Alcaraz, I., and Chi, P. (2016). Web Application Teaching Tools for Statistics Using R and Shiny. *Technology Innovations in Statistics Education, 9*(1). Retrieved from https://escholarship.org/uc/item/00d4q8cp

R Core Team (2017). R: A language and environment for statistical computing [Computer Software]. Retrieved from https://www.R-project.org/

RStudio Team (2017). RStudio: Integrated Development for R [Computer Software]. Retrieved from http://www.rstudio.com/

Sarkar, D. (2008). *Lattice: Multivariate Data Visualization with R.* New York, NY: Springer-Verlag.

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. New York, NY: Springer-Verlag.