IMPROVING STATISTICAL COMMUNICATION IN STATISTICAL COMPUTING COURSES

Hunter Glanz and Shannon Pileggi
Statistics Department, California Polytechnic State University, 1 Grand Avenue, San Luis Obispo, 93407
hglanz@calpoly.edu

Statistical computing courses often revolve around a statistically-oriented programming language and involve curriculum motivated by problems in the field of statistics. Programming-focused learning objectives can facilitate an excessively large focus on merely completing a particular programming task or recovering a certain table of results, when the course should be just as much about statistics. In the midst of wondrously more computing ability by statistics students in recent years, there must remain the ability to communicate about the methods implemented and results visualized. With the advent of RStudio, Jupyter Notebooks, and other tools, incorporating statistical communication into your statistical computing course has never been easier.

BACKGROUND

People have been doing statistics for much longer than the field is usually thought to have been around. Communication, in the field of statistics, boasts an importance unmatched in any other field precisely because every other field uses statistics in some way. As the science dealing with the collection, analysis, interpretation, presentation, and organization of data, statistics answers the calls of all commercial and academic endeavors to grow and advance. In our increasingly data-driven world, the demand for statistics knowledge has exploded. In efforts to better educate future statisticians, numerous researchers and educators pour over best practices and develop key recommendations to guide course construction. From K-12 to introductory college courses to whole degrees in statistics, documents such as the Guidelines for Assessment and Instruction in Statistics Education (GAISE, 2016) help shape curriculum development. As the field of statistics changes, the GAISE guidelines also evolve while maintaining some key tenets such as (1) teach statistical thinking, (2) foster active learning, and (3) use technology to explore concepts and analyze data. While the bulk of these efforts focus on statistics education in general, the recent growth of data science and markedly increased computing power warrants comments about the specific need for more computational ability from our students.

Since the original GAISE 2005 Report, the use of technology in introductory statistics courses has become all but standard with innovators like Chance et al. (2007) helping make packages from Microsoft Excel spreadsheets to Tinkerplots to Minitab and R more accessible to educators at all levels. Today's ubiquity of personal computing power transformed this presence of technology in statistics classrooms into actual programming ability on the part of many of these students. In 2010, Nolan and Temple Lang called for more computing in statistics curricula with emphasis on three key components: 1) broaden statistical computing, 2) deepen computational reasoning and literacy, and 3) compute with data in the practice of statistics. As the field of data science continues to grow, and statistics along with it, these ideas get echoed and adapted. For example, Horton et al. (2014) suggested the incorporation of more data science into introductory statistics courses and Hardin et al. (2015) emphasized "preparing students to 'think with data'."

Programs in statistics across the globe possess a unprecedented synchronicity with respect to computing, but the most important skill of all cannot be forgotten: communication. In fact, according to the National Association of Colleges and Employers' *Job Outlook 2016* report employers rated [verbal] communication skills the most important when assessing skills/qualities (NACE, 2016). Underemphasized, perhaps because it has been around for so long, statistical communication can and should be incorporated into even computing courses.

In 2014, Baumer et al. lauded R Markdown as a "reproducible analysis tool" for introductory statistics courses. The next stages of the evolution of these types of tools exist now in the form of R Notebooks and Jupyter Notebooks. The remainder of this article aims to demonstrate how these tools and a venue like YouTube can help you seamlessly integrate statistical communication-themed learning objectives into a statistical computing course.

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

THE PROBLEM

To stay competitive and current with the field of statistics, students must be equipped with more computational skills than ever before. To meet this demand, programs in statistics and data science offer more computationally oriented courses with increased emphasis on the role of computer science in the completion of today's problems in statistics. At California Polytechinic State University, San Luis Obispo, the goals of these courses are at least three-fold:

1. Proficiency with a particular software package like R, SAS, Python, etc.
2. Execution of traditional and more modern statistical methods.
3. Accurate interpretation and effective communication of results.

The latter one or two goals may not even make the list of course objectives for a standard statistical computing or data science course, or receive only cursory attention if they do. It is convenient to focus on implementation, algorithmic thinking, and general programming ability in such courses. This results from both the primary objective of the course being computational and from the feeling that any increased attention to statistical methodology or communication necessarily means sacrificing time on these ever-valuable computer science foundations. It can become easy to lose sight of the statistics in such courses, or pass off rote statistical analyses as sufficient. In particular, students often begin by effectively redoing problems from their introductory statistics courses, but with the use of their new programming abilities. However, these tasks alone fail to challenge students' abilities to solve new statistics problems and undermine the importance of the new software tools they are learning about. While students certainly get practice with statistical software, they can easily miss the big picture of the workflow in statistics problems. Getting caught in the traditional cycle of

a. Importing a new data set
b. Translating the research questions into the statistical methods required
c. Applying said statistical methods
d. Displaying results and/or graphs
e. Reciting interpretation of results

can be as mechanical as slogging through their 50th hypothesis test problem in their introductory course. Instead of the cycle above, students need to learn that the statistics workflow is fluid and not cookie-cutter. Figure 1 aptly describes this iterative process which can take a different, albeit reproducible, form for each new problem (Grolemund, 2017). Although this figure was created by RStudio to demonstrate the use of R in statistics projects, it is a universal workflow that can be generally applied to different programming languages. The blocks in blue correspond to the traditional cycle of analysis described above, and the blocks in green describe other items performed in statistical software to achieve those analysis objectives.

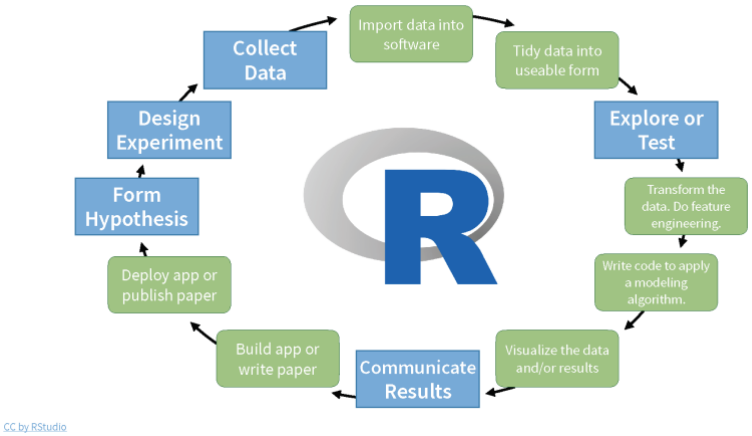


Figure 1. Statistics project workflow.

A statistical computing course should not boil down to a computer science course in a statistical programming language, with hints of applications in statistics provided through data-based examples and assignments. A statistical computing course can and should be *an equitable pursuit of all three of the ideal course objectives listed above*. Instead of merely presenting and demonstrating the capabilities of a new software package, with each lesson students should also understand where the material fits in the realm of statistical analysis and the ways in which to transform output into meaningful insights. With this template in mind, educators should aim to involve all three course objectives throughout lectures, in-class activities, homework assignments, projects, and exams. For example, the split-apply-combine paradigm (Wickham, 2011) is a topic common to many computing courses and does not need to be statistically oriented. However, in a *statistical computing* course it can be taught in the context of obtaining descriptive statistics or even fitting statistical models. This then requires further discussion about interpreting and presenting the results since students should not just be copying raw output into a report or memo.

New software tools and the leveraging of existing technologies in innovative ways make this proposed synthesis possible in an ultimately unprecedented and streamlined way.

VIDEO PROJECT

The vastness of statistical software packages like R, SAS, and Python all dwarf a single college course. The breadth of demands that careers in statistics can carry mean students must, at the very least, be made aware of this vastness. All the better, freedom to explore these packages and apply the implementations they see fit can significantly improve students' curiosity, passion, and understanding of the software language as a whole. To this end we use a *video tutorial project* to introduce students to the ideas of (1) they will inevitably have to learn a new feature of a statistical software package that they were not explicitly instructed on how to use in class, and (2) it is likely at some point for them to have to explain the new feature to a colleague.

Statistical computing courses in R or Python accommodate this type of project with incredible ease. As both languages boast large communities of active contributors, students never fail to identify topics of interest outside the traditional course content. In our *video tutorial project* in the course "Statistical Computing with R" we suggest a short list of R packages, that we spend little to no time on in class, for them to select (of course they may choose a package not on the list with our approval). These include, but are not limited to: DT, plotly, leaflet, twitteR, flexdashboard, and htmlwidgets.

In groups of two to three, students select a package to explore and learn more about. Then, the students submit an outline of their "video storyboard," which amounts to a rough draft of the code they will demonstrate with their verbal script. Next, they produce a 5 minute video tutorial on the package and upload that video to YouTube. Lastly, students assess each other's videos during a peer evaluation portion of the video grading.

We assign this project 2-3 weeks into the term, just after covering the very basics of the language, and allow the students about one week to complete it. It can seem daunting at first, but students get reminded that the project consists of understanding what their chosen package does and how to use it, not how the package works or was built. With this in mind the assignment effectively synthesizes our three main course objectives:

1. Students get exposure to new, personally chosen content about the statistical software package expanding their view of it and cementing the key elements of it that exist in every package.
2. All of the approved packages deal with data management, data manipulation, data visualization, or statistical methodology in some way.
3. Students must think critically about how to explain new statistical computing functionality when both creating their tutorial video and in critiquing their peers' videos.

We also ask students to vote on the most useful video, the most entertaining video, and the best overall video. These bonuses further incentivize high quality work and entice groups to have fun with the assignment by including elements of humor, pop culture references, or other engaging tidbits.

THE R/JUPYTER NOTEBOOK FORMAT

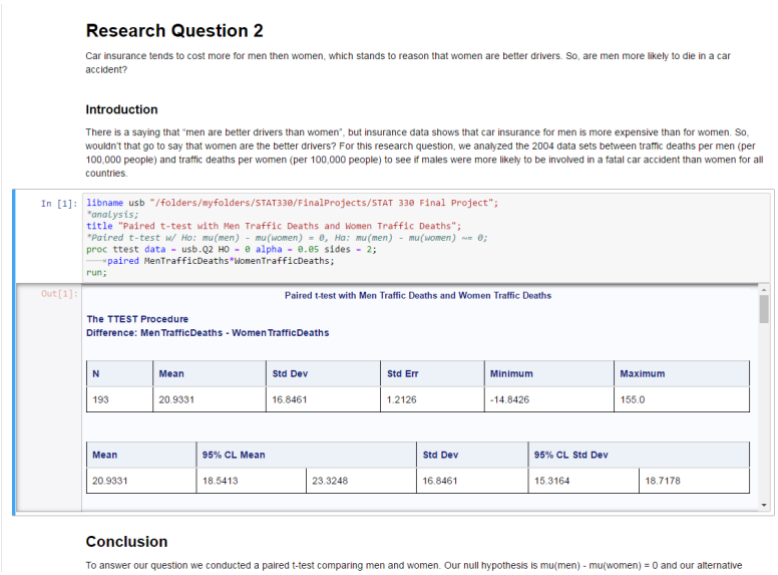
Historically, programming has been done in dedicated applications. R and SAS are programs that you can write and run code in. For most languages, many people use editors (VIM, Notepad++, emacs, etc.) or integrated development environments (IDEs) to write code in. These common practices in industry and academia inevitably involve teaching it in similar ways. However, doing so invites the disproportionate emphasis on computer science skills mentioned above. For problems in statistics and in statistical computing courses, these tools often feel clunky. Importing data, manipulating it, summarizing it, visualizing it and then analyzing it make up a large portion but not the entire project workflow. The soft skills consistently required by industry, interpretation and communication, tend to suffer when assignments end with implementing some sort of statistical analysis in software without explanation of results.

The leap from a traditional computing assignment that ends with analysis to one that includes context, interpretation, and insight often seems like the difference between a weekly homework assignment and a course-long project. The former usually just consists of one or more computing tasks, while the latter builds the computing tasks into a multi-page report with contextual exposition and inference. The reason being interpretations and insight should not just be left as comments at the end of or throughout the code script, but fleshed out in a comprehensive and professional way. Consequently, a proper report culminating the entire problem in context with concluding remarks involved performing all of the computational work, exporting the code, output, and graphics, then organizing that work throughout a body of text using something like Microsoft Word. This piece-meal workflow detracts from the learning experience, artificially inflates the time-to-completion for students, and negates today's ever increasing focus on reproducibility.

While RMarkdown and Project Jupyter have been around for some time now, only recently has their use truly exploded. The *notebook* format, manifested as R Notebooks and Jupyter Notebooks, provide an elegant solution to the fragmentation problem detailed above. Notebooks consist of a series (any number) of cells used for either writing and executing code or displaying markdown text. These cells can be as small or large as desirable, and can be arranged in any order throughout the notebook. Any number of code cells can be run at once and produce results in-line within the notebook. While R Notebooks execute R code, Jupyter Notebooks support over 40 programming languages including R, Python, SAS, Julia, and much more. The non-code cells support markdown text and LaTeX-like typesetting. In the end, notebooks can be saved in a number of formats including .html and .pdf. No longer is the ideal statistical computing assignment out of reach:

1. Students' experience with statistical software becomes more user-friendly with an all-in-one environment like a notebook.
2. The statistics problem workflow can be easily separated into distinct cells or groups of cells, and narrated with explanatory text.
3. The markdown cells within the notebook along with in-line results and graphics completely merge the computational work with report construction. The notebook eliminates the need to export results and then re-organize them in an entirely different environment. Statistical communication is now a seamless part of the statistical computing assignment!

As essentially *dynamic documents*, notebooks provide a vehicle for students to create a *computational narrative* involving both live code to be run and expository text. Both Jupyter and R Notebooks support executing code cells in any order you like and producing output in-line. Figure 2 shows a Jupyter Notebook with a SAS kernel demonstrating this feature.



Research Question 2

Car insurance tends to cost more for men then women, which stands to reason that women are better drivers. So, are men more likely to die in a car accident?

Introduction

There is a saying that "men are better drivers than women", but insurance data shows that car insurance for men is more expensive than for women. So, wouldn't that go to say that women are the better drivers? For this research question, we analyzed the 2004 data sets between traffic deaths per men (per 100,000 people) and traffic deaths per women (per 100,000 people) to see if males were more likely to be involved in a fatal car accident than women for all countries.

```
In [1]: libname usb "/folders/myfolders/STAT330/FinalProjects/STAT 330 Final Project";
        *analysis;
        title "Paired t-test with Men Traffic Deaths and Women Traffic Deaths";
        *Paired t-test w/ Ho: mu(men) - mu(women) = 0, Ha: mu(men) - mu(women) ^= 0;
        proc ttest data = usb.Q2 HO = 0 alpha = 0.05 sides = 2;
        ----paired MenTrafficDeaths*WomenTrafficDeaths;
        run;
```

Out[1]:

Paired t-test with Men Traffic Deaths and Women Traffic Deaths

The TTEST Procedure

Difference: MenTrafficDeaths - WomenTrafficDeaths

N	Mean	Std Dev	Std Err	Minimum	Maximum
193	20.9331	16.8461	1.2126	-14.8426	155.0

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
20.9331	18.5413	23.3248	16.8461	15.3164	18.7178

Conclusion

To answer our question we conducted a paired t-test comparing men and women. Our null hypothesis is mu(men) - mu(women) = 0 and our alternative

Figure 2. In-line output within the notebook interface: Jupyter Notebook with SAS kernel.

Upon completion, these dynamic documents can be converted to easy-to-read HTML or PDF files for sharing with collaborators or submission to instructors for assessment. These tools also admit document customization so that the converted files do not just look like a series of text and output chunks. Figure 3 gives one example where the HTML output file from an R Notebook contains a clickable table of contents.

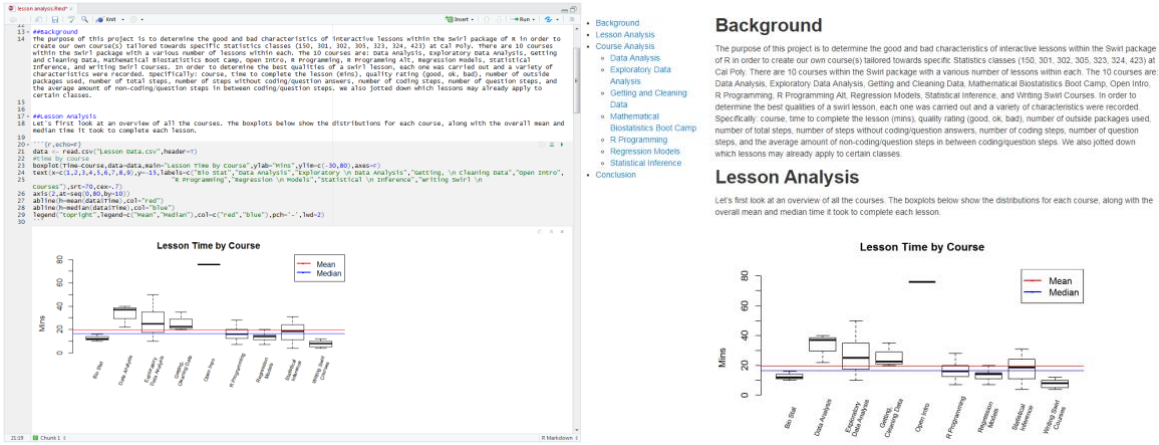


Figure 3. R Notebook with in-line output (left) and corresponding HTML output file with clickable table of contents (right).

CONCLUSION

The statistics project workflow has historically been fragmented and tedious, inadvertently resulting in statistical computing coursework focusing almost exclusively on programming ability with a particular statistical software package. The R and Jupyter Notebook environments allow for all parts of the workflow to occur organically in one place. We propose the use of these tools along with other modern technologies, like YouTube, to incorporate more statistical communication into courses in statistical computing. The learning objectives of these courses only get enhanced as

students tackle whole statistics projects, leveraging their new computational ability and existing methodological knowledge to draw meaningful insights from real world data.

REFERENCES

Baumer, B., Cetinkaya-Rundel, M., & Bray, A. (2014). R Markdown: Integrating A Reproducible Analysis Tool into Introductory Statistics. *Technology Innovations in Statistics Education*, *8*(1). Retrieved from https://escholarship.org/uc/item/90b2f5xh

Chance, B., Ben-Zvi, D., Garfield, J., & Medina, E. (2007). The Role of Technology in Improving Student Learning of Statistics. *Technology Innovations in Statistics Education*.

GAISE College Report ASA Revision Committee. (2016). *Guidelines for Assessment and Instruction in Statistics Education (GAISE) College Report*. American Statistical Association. Retrieved from http://www.amstat.org/asa/files/pdfs/GAISE/GaiseCollege_Full.pdf

Grolemund, G. (2017). Figure of Data Science Cycle. Licensed under CC BY 4.0 (Ed.), *Introduction to Master the Tidyverse* . Joint Statistical Meetings.

Hardin, J., Hoerl, R., Horton, N. J., Nolan, D., Baumer, B., Hall-Holt, O., … Ward, M. D. (2015). Data Science in Statistics Curricula: Preparing Students to "Think with Data." *American Statistician*. https://doi.org/10.1080/00031305.2015.1077729

Horton, N. J., Baumer, B. S., & Wickham, H. (2014). Teaching precursors to data science in introductory and second courses in statistics. *ArXiv:1401.3269 [Cs, Stat]*.

NACE Staff. (2016). Employers: Verbal Communication Most Important Candidate Skill. Retrieved May 10, 2018, from http://www.naceweb.org/career-readiness/competencies/employers-verbal-communication-most-important-candidate-skill/

Nolan, D., & Temple Lang, D. (2010). Computing in the statistics curricula. *American Statistician*. https://doi.org/10.1198/tast.2010.09132

Wickham, H. (2011). The Split-Apply-Combine Strategy for Data Analysis. *Journal of Statistical Software*. https://doi.org/10.18637/jss.v040.i01