"LOOKS OKAY TO ME": A STUDY OF BEST PRACTICE IN DATA ANALYSIS CODE REVIEW

Amal Abdel-Ghani[1], Kelly Bodwin[2], Amelia McNamara[1], Allison Theobold[2], and Ian Flores Siaca[3]
[1]University of St Thomas
[2]California Polytechnic University, San Luis Obispo
[3]Voltron Data
kbodwin@calpoly.edu

Education in statistical computing requires that we train students not only in programming skills and principles, but also in good data science habits. In this study, we investigate the question of how certain habits, routines, or intuitions contribute to quality analysis, in the context of identifying errors in an existing work. Volunteers from two populations—professional data scientists and mid-degree college students—were supplied with pre-populated R Markdown notebooks and asked to comb through the reports' code and discussion in search of errors. We then conducted a qualitative analysis of subject behavior during the study, based on video recordings of these sessions. Ultimately, we identified many common themes in how the subjects interacted with the code, text, and integrated development environment during their error-checking process.

INTRODUCTION

Conducting and verifying data analyses are two key skills for statisticians and data scientists at every level in their career (Harraway & Barker, 2005; Stodden, 2014). Much emphasis has been put on conducting analysis, but the movement toward reproducible research and the collaborative nature of the statistical workplace underscores the importance of verifying analysis. Because curricular experts tout the benefits of providing novices an authentic experience with data (GAISE College Report ASA Revision Committee, 2016; Roseth et al., 2008), novices should be exposed to both the production and verification of data analysis.

When it comes to performing analysis, expert statisticians often follow the same steps. A common workflow includes taking time to understand the problem, conducting exploratory data analysis (EDA), specifying a model and assessing its validity, and communicating the findings (Greenhouse & Seltman, 2018; Wickham & Grolemund, 2017). Novices may not initially follow these steps, but this expert method of data analysis can be instilled in novices through curricular experience. Many curricula incorporate students engaging authentically with the data analysis cycle (Baumer, 2015; Cetinkaya-Rundel, 2021).

Much like with performing data analysis, the repeated experience statistics experts have with troubleshooting and verifying analyses allows them to formulate successful strategies over time. However, the development of curricular materials on troubleshooting strategies has lagged behind. This is due in part to a lack of research about the behaviors expert statisticians employ when troubleshooting, as well as a lack of research on how best to ingrain these effective problem-solving frameworks.

We can look to the computer science education field as a source of related work. There, research suggests novices tend to focus on getting a direct solution to a problem rather than pursuing deeper understanding. In contrast, experts achieve a systematic view of the issue by understanding how the program functions, testing more thoroughly, and having a greater awareness of what the correct outputs should be (Li et al., 2019). Computer scientists foster expert-level troubleshooting strategies in novices through systems where learners are able to generate a hypothesis about what they think is wrong, test their hypothesis, and evaluate the results (Jonassen & Hung, 2006). We suspect that differences between expert and novice problem-solving approaches in statistics and data science may mirror those in computer science.

Peng et al. (2021) have performed an initial study on how students deal with real-world troubleshooting opportunities in data analysis. In their work, graduate students were given hypothetical data analysis case studies with explicitly stated unexpected outcomes. Students worked backwards to diagnose possible sources of the discrepancy and recommend next steps. Researchers were interested in whether students could identify possible errors in the analysis and suggest specific follow-up procedures to eliminate the error. This was a pilot study, so although the authors found that all students were able to suggest a follow-up, they did not have any overall conclusions from the study.

In S. A. Peters, L. Zapata-Cardona, F. Bonafini, & A. Fan (Eds.), *Bridging the Gap: Empowering & Educating Today's Learners in Statistics. Proceedings of the 11th International Conference on Teaching Statistics (ICOTS11 2022), Rosario, Argentina.* International Association for Statistical Education. iase-web.org

Our work seeks to build the literature of troubleshooting strategies used by experts and novices when trying to identify errors in data analyses. By analyzing the actions experts take in troubleshooting scenarios and contrasting them with actions that novices take, we can begin to dissect how these expert-level strategies and skills can be transferred to novice statisticians. Ideally, after these strategies have been identified, they could be integrated into curriculum, and novices could learn to use them in much the same way students learn from the ways expert strategies for data analysis are now taught. Bridging the gap between experts and novices begins with identifying their differences when faced with the same situation.

METHODOLOGY

Similar to Peng et al. (2021), our work provided analysts with completed analysis, which they then needed to investigate for errors. However, in our study some participants were given a document with no errors, whereas others were provided a document with many errors. Documents were randomized to participants. All materials were provided as R Markdown documents on a standardized RStudio Cloud workspace.

Design of R Markdown Notebooks

Participants in this study were supplied with two pre-populated R Markdown notebooks: one analyzing a dataset of NBA "Player of the Week" awards from the 2018 season, and one analyzing voting data from the 2011 Spanish national election. Each notebook contained built-in errors throughout the data analysis and/or discussion of results. The R Markdown notebooks, as well as translations into Jupyter notebooks containing equivalent Python code, are available online (Bodwin et al., 2022). The notebooks contain a mixture of written text and code, and the code can be executed to produce output (models, plots, etc.).

The notebooks were extended from an exam in an undergraduate introductory R programming course using the NBA dataset. We adapted a selection of the exam questions into the complete analysis and summary found in the notebook for this study. Then, based on a list of common student errors on the exam coupled with a holistic list of common data analysis errors compiled by our team of researchers, we deliberately embedded errors into the notebook. The second notebook, which was based on the Spanish elections data, was constructed to mirror the length, complexity, style, and error types of the NBA notebook.

Importantly, none of the deliberately embedded errors triggered Error or Warning messages when run; rather, they were mistakes designed to reflect an incomplete understanding of the analysis. Specifically, we classified the embedded errors into three categories:

- *Data Understanding:* These errors represent situations where the analyst operated on incorrect assumptions about the content or structure of the supplied dataset.
- *Statistical Concepts*: These errors represent misinterpretations of statistical results or misapplications of statistical tests.
- *Programming Skills:* These errors, the most common in the notebooks, represent implementations of R code that do not accomplish the expressed goal.

Some conceptual errors were embedded in the written text in the notebook, such as misinterpretations of a *p*-value. Others, however, were present in the code. Although these were non-programming errors (i.e., they did not throw a warning or error in R), they were mistakes in the code. For example, both notebooks contained a two-sample *t*-test, with the alternate hypothesis mis-specified in the `infer::t_test()` function; a one-sided test was used where a two-sided test would have been appropriate. Although this error manifests in the code, it derives from a misunderstanding of the correct specification of hypotheses; thus, we classify it as a *Statistical Concepts* error. Similarly, consider the code snipped below from the NBA notebook:

```
nba <- nba %>%
  mutate(
    Height = parse_number(Height),
    Weight = parse_number(Weight)
      )
```

In this example, the analyst is attempting to convert the `` `Height` `` and `` `Weight` `` columns from a string data type (e.g., "`70in`") into numeric data. The usage of the `parse_number()` function is

correct and computes without error. However, the analyst has failed to account for the fact that units are mixed in the data—some heights are listed in inches ("`70in`"), whereas some are in centimeters ("`178cm`"). Converting such data into the numbers (`70` and `178`) is a misrepresentation of the information. We would thus consider this to be a *Data Understanding* error.

To be a true *Programming Skills* error, the mistake needed to be related to code behavior. For example, in one snippet, an adjustment is made to the NBA dataset, but the updated dataset was not re-assigned to a new or existing object. In some cases, an apparent coding error could be due to either a misapplication of programming *or* a lack of conceptual or data understanding. One such example is the use of `str_detect(Position, "G")` within a particular data pipeline, with the goal of identifying all "Guard" position players in the NBA dataset. Although it is true that Guards are demarked as "`G`" in the Position column, this task does not succeed, as it also misidentifies Forward-Guards ("`F-G`") as Guards. This error could be due to *Data Understanding* if the analyst was not aware of the "`F-G`" values; or it could be due to *Programming Skills* if the analyst did not understand that `str_detect()` uses regular expressions and will also consider "`F-G`" a match.

Ultimately, our goal in the design of the error-embedded notebooks was to study code review in a full data analysis context. Rather than isolating code to debug, data to describe, or statistical concepts to interpret, we presented all these skills in a blended, real-world report context. As such, our study can examine participant behavior in a way that is controlled, but also realistic to contemporary hands-on data analysis.

Participants

This study was interested in comparing the error detection strategies between two populations, novice and experienced data scientists. Experienced data scientists were recruited through the population of RStudio certified instructors (https://education.rstudio.com/trainers/), emails, and social media posts. A total of 12 instructors volunteered to participate. These participants had a variety of backgrounds but were primarily employed by academic institutions or industry. Novice data scientists were recruited from a mid-level undergraduate college course in R programming at the California Polytechnic University (Cal Poly), consisting of primarily of second- or third-year students pursuing a major or minor degree in statistics. Students were provided extra credit for participation in the study, with the option to complete an alternative assignment for extra credit. A total of four students participated in this study.

Data Collection

During recruitment, prospective participants were requested to complete a survey consenting to participate in the study and for their interview to be recorded (Cal Poly IRB: 2020-135). Following the preliminary survey, participants who agreed to be interviewed were video recorded while working through the two R Markdown notebooks. At the beginning of the interview, a member of the research team reminded the participant of the intentions of the study and requested that the participant talk out loud as they worked through each report. Participants were given 20 minutes to examine each notebook, for a total interview length of 40 minutes. As participants worked through each report, the interviewer played a minor role, mainly speaking to remind participants to be more verbal with their thinking. Namely, the interviewer did not confirm or deny if a participant had found an error; nor did they manage the time allocated to each report. As such, some participants were able to work through each report in its entirety, whereas others were only able to get halfway through each report.

The analysis in this paper is based on the strategies participants used while searching for errors in the reports provided.

Data Analysis

The analysis of the videos took place over three stages (Miles et al., 2020). In the first stage, names were removed from the videos; the videos were hosted on screenlight.tv; and three videos were randomly assigned to five members of the research team. Each member was instructed to watch the videos and create descriptive codes of actions participants performed when searching for an error (e.g., edits code, scrolls up to check earlier work, speaks through what the code is doing). After the videos were coded, the research team met to begin the second stage of analytical coding. During this meeting,

the descriptive codes discovered by each member were discussed. Categories that held across multiple interviews were retained. At the adjournment of this meeting, a codebook of eight themes and subthemes had been developed. At the end of this stage, canonical examples (video links) of each theme were added to the codebook.

In the final analysis stage, the first author independently coded each video, coding each action performed by a participant according to the eight agreed-upon themes. Before watching the videos, the author familiarized herself with the themes and subthemes using the canonical examples and met with the third author to confirm her understanding of the themes. While analyzing the videos, the first author discovered two themes of actions which were not captured by the original codebook. The first and third authors met to discuss these new themes, as well as any actions taken by participants that did not clearly conform to a specific theme.

RESULTS

The data that resulted from the video coding is quite rich, and contains timestamps of each action taken by participants, the overall theme at each point, and the subtheme in that category. One high-level result is simply the number of times each theme was observed in the data. Table 1 provides a summary of the themes, ordered by frequency of use by experts. Because the sample sizes of the two groups were unequal, we have divided the total number of times the theme was observed in a particular group by the number of participants in the group, to form a rough average.

Table 1. Average theme counts for expert and novice participants

Theme	Expert (n = 12)	Novice (n = 4)
Interaction with supplied code chunks	31	29
Verbalizing	31	22
Suggesting edits	26	15
Integrated Development Environment (IDE) fluency	25	40
Physical interaction with document	22	16
Viewing data	5	3
Pain points	4	1
Verifying data	4	0
Seeking help	3	1
Statistical tests	2	0

Note: The total number of occurrences of the theme for each group was divided by the sample size (n = 12 and 4, respectively) to compute an average amount of theme usage.

On average, the theme performed most often by experts (and second most often by novices) was "interaction with supplied code chunks." On average, experts performed actions in this theme 31 times, compared to 29 times for novices. Because interviews were approximately 40 minutes long, this means participants were doing actions in this theme almost every minute, on average. The "interaction with supplied code chunks" theme captures how participants engaged with and manipulated the provided code chunks to better understand the analyses. The actions within this theme included running whole code chunks or partial code chunks, instances when the code was tweaked and re-run, if chunks were reconstructed in more familiar code styles, and when participants passed by code chunks without running them in full. Actions in this theme were characteristic of the way in which participants ran and tweaked the code to help identify errors within the outputs.

This theme was the most common for experts, and was also frequently performed by novices, so it shows similarity of behavior between the two groups. This was not true for every theme. Looking at the ordered list in Table 1, it is possible to see the divergence between the two groups.

Notably, the theme that occurred most often for novices was "Integrated Development Environment (IDE) fluency." Experts performed actions within this theme as well, but not as frequently. On average, each novice performed 40 IDE fluency actions, whereas experts only used actions in the theme 25 times on average. This theme captured participants' use of the RStudio IDE. Actions within this theme included using keyboard shortcuts, looking at objects in the different panes within R, opening

datasets in a variety of ways, rearranging panes, restarting R between analyses, and using common buttons such as 'knit.' Greater IDE fluency is generally acquired as coders gain more experience with the platforms with which they are working. Due to this, we were surprised that novices had a greater average occurrence of this theme compared to experts despite their lesser experience in R. However, it is likely that novices were more likely to use feature of the IDE, such as clicking on a dataset in the Environment page to see it, rather than programmatic methods to accomplish a similar goal, such as running the `head()` function.

CONCLUSION

This qualitative analysis of expert and novice troubleshooting strategies in data analysis is ongoing. We have presented a novel set of data analytic error categories: *Data Understanding, Statistical Concepts,* and *Programming Skills.* Errors in these categories have been carefully embedded into sets of structured notebooks, to provide a framework for assessing error-finding strategies. The notebooks we have developed are Creative Commons licensed, and could be used by other researchers or instructors seeking to teach or assess data analytic troubleshooting skills (Bodwin et al., 2022). In fact, the Python translations of the notebooks have already been used for a replication of this work (Robinson et al., 2022). Our three-stage coding practice, starting with separate coding by multiple authors and funneling down to a final set of themes, gives credence to the set of codes we eventually identified. These codes were divided into 10 themes, which are summarized in Table 1. Although we cannot provide a full analysis of expert and novice troubleshooting behavior in this manuscript, we identified the most common theme of behaviors exhibited by experts ("interaction with supplied code chunks") and contrast it with the most common theme exhibited by novices ("IDE fluency"). Future work will dig further into the data to better identify trends within and between groups. We encourage other researchers to consider extensions to our work, using the provided notebook materials.

ACKNOWLEDGEMENTS

This research was generously supported by a seed grant from RStudio, and continued support from the University of St. Thomas. The authors thank Greg Wilson and Phillip Burkhart for their contributions to error ideas in notebooks and to the video analysis and theme identification.

REFERENCES

Baumer, B. (2015). A data science course for undergraduates: Thinking with data. *The American Statistician*, *69*(4), 334–342. https://doi.org/10.1080/00031305.2015.1081105

Bodwin, K. N., Flores Siaca, I., & Robinson, D. (2022). *Materials for "Looks okay to me": A study of best practice in data analysis code review.* Zenodo. https://doi.org/10.5281/zenodo.6419727

Cetinkaya-Rundel, M. (2021). *Data science in a box*. https://datasciencebox.org/

GAISE College Report ASA Revision Committee. (2016). Guidelines for assessment and instruction in statistics education college report 2016. American Statistical Association. http://www.amstat.org/education/gaise

Greenhouse, J. B., & Seltman, H. J. (2018). On teaching statistical practice: From novice to expert. *The American Statistician*, *72*(2), 147–154. https://doi.org/10.1080/00031305.2016.1270230

Harraway, J., & Barker, R. (2005). Statistics in the workplace: A survey of use by recent graduates with higher degrees. *Statistics Education Research Journal*, *4*(2), 43–58. https://doi.org/10.52041/serj.v4i2.514

Jonassen, D. H., & Hung, W. (2006). Learning to troubleshoot: A new theory-based design architecture. *Educational Psychology Review*, *18*(1), 77–114. https://doi.org/10.1007/s10648-006-9001-8

Li, C., Chan, E., Denny, P., Luxton-Reilly, A., & Tempero, E. (2019). Towards a framework for teaching debugging. In *Proceedings of the Twenty-First Australasian Computing Education Conference* (pp.79–86). Association for Computing Machinery. https://doi.org/10.1145/3286960.3286970

Miles, M. B., Huberman, A. M., & Saldaña, J. (2020). *Qualitative data analysis*. Sage.

Peng, R. D., Chen, A., Bridgeford, E., Leek, J. T., & Hicks, S. C. (2021). Diagnosing data analytic problems in the classroom. *Journal of Statistics and Data Science Education*, *29*(3), 267–276. https://doi.org/10.1080/26939169.2021.1971586

Robinson, D., Ernst, N. A., Vargas, E. L., & Storey, M.-A. D. (2022). *Error identification strategies for python Jupyter notebooks (arXiv:2203.16653)*. arXiv. https://doi.org/10.48550/arXiv.2203.16653

Roseth, C. J., Garfield, J. B., & Ben-Zvi, D. (2008). Collaboration in learning and teaching statistics. *Journal of Statistics Education*, *16*(1). https://doi.org/10.1080/10691898.2008.11889557

Stodden, V. (2014). The reproducible research movement in statistics. *Statistical Journal of the IAOS, 30*(2), 91–93. https://doi.org/10.3233/SJI-140818

Wickham, H., & Grolemund, G. (2017). *R for data science: Import, tidy, transform, visualize, and model data*. O'Reilly.