**Environmental Science Graduate Student Strategies for Acquisition of Statistical Computing Skills**

Allison Theobold

Montana State University

allisontheobold@montana.edu

Stacey Hancock

Montana State University

stacey.hancock@montana.edu

ABSTRACT

Current scientific research requires a substantial amount of computer programming and statistical computing, but graduate students typically lack these integral skills. Consequently, many scientific graduate degree programs expect graduate students to learn these computational concepts in a statistics course. This study examines experiences of environmental science graduate students in acquiring computational knowledge and skills in the context of a methods of data analysis sequence, and the factors that foster or inhibit learning. In-depth interviews revealed three themes: peer support, a singular “consultant,” and independent research experience. These themes provide rich descriptions of graduate student experiences that better inform how both instruction and learning could be improved, with emphasis on applying these computational skills immediately in their own research.

Keywords: Statistics education research; Statistical computing; Computational thinking; CS for scientists; Computer science education

1. Introduction

With the increased focus on data-intensive research, statistical computing has become foundational in many science fields; however, the gap between education in the sciences and computation has become more evident, particularly in the environmental and life sciences. With the growth in computational power, the complexity of scientific models has followed, multiplying the computational, mathematical, and statistical expectations of researchers' abilities. Because of the importance of computational knowledge and abilities, some universities have started to require undergraduate science students to enroll in an introductory programming course (Rubinstein & Chor, 2014), while others are providing graduate students with computing bootcamps for quantitative methods (Stefan, Gutlerner, Born, & Springer, 2015) or requiring graduate level statistics coursework for degree completion. However, little is known on where and how graduate students are acquiring the computational knowledge necessary to implement statistical analyses in their field.

The intention of this study is to describe and understand what factors impact how environmental science graduate students gain computational knowledge and the ability to reason through applications of statistical computing in their disciplines. Here we address these primary research questions:

1. Where do graduate environmental science students gain the computational knowledge necessary to implement statistical analyses for applications in their disciplines?
2. How do student backgrounds differ in the impacts of these factors on inhibiting or fostering students' computational understandings and abilities?

The subjects of this study were graduate students enrolled at a mid-size university in the Rocky Mountains, where the two-semester graduate statistics sequence Methods of Data Analysis I and II is required or highly recommended for the completion of a master’s degree in many environmental science fields such as Ecology, Land Resources and Environmental Sciences, Animal and Range Sciences, and Plant Sciences. However, this terminal statistics sequence often serves as the sole computational course, and thus indirectly prepares graduate students for the statistical and computational problems they may face as researchers and practitioners. Though the term “environmental science” refers to a specific discipline in the literature, in this paper we will refer to the large assortment of fields the Methods of Data Analysis course sequence services as “environmental science.” In examining the experiences environmental science graduate students face when acquiring computational skills, the present study seeks to capture a more in-depth understanding of the successes and shortfalls these students face when procuring computational knowledge and skills.

We begin by describing the areas of research literature that specifically address the computational training of undergraduate and graduate students in the life sciences. We then describe the qualitative study implemented to explore where graduate environmental science students acquire the skills necessary to complete statistical computing applications in their field. The results presented reveal both the prevailing resources these students employ when faced with computational problems beyond their understanding, and articulate the path students follow when seeking computational assistance.

1. Computing and the environmental Sciences

Research in computational abilities of environmental science majors is in its infancy, with only a handful of institutions performing research that specifically addresses the computational training necessary to prepare students for careers post undergraduate or graduate school. In this section, we discuss briefly three areas of the research literature that informed this study. First, we review the literature on the foundational role computation has in the sciences. We then discuss the research efforts on curriculum design for introductory computing courses for non-computer science students. Finally, we describe one institution’s development and implementation of computational training for graduate students in the biological sciences.

* 1. Importance of Computing in the Sciences

Over the last decade, the life and environmental science fields have seen a rapid increase in the use of computation and analytical tools to model phenomena across many disciplines of inquiry. In some scientific fields, such as biology and chemistry, the recent ability to collect multitudes of data easily and quickly have made computational abilities vital to researchers and practitioners. Meanwhile, fields previously thought to be niche disciplines, such as computational and mathematical biology, are now “becoming an integral part of the practice of biology across all fields” (Stefan et al., 2015. p. 2). Across a large sector of scientific domains, applications of mathematical and statistical techniques, such as Markov Chain Monte Carlo, neural networks, and agglomerative hierarchical clustering, have become essential computational understandings for field applications (Weintrop et al., 2016). With these advances in data collection, visualization, analysis, and interpretation, as well as computational power, analytical methods, and detailed computational models, scientific fields are undergoing a renaissance. These advances have, however, created a growing need for scientists to receive an appropriate education in computational methods and techniques. The need for computation in education for mathematics and science is greater than ever (Fox & Ouellette, 2013).

* 1. Computational Courses for Undergraduate Science Majors

Multi-disciplinary efforts have been made at Purdue, Carnegie-Mellon, Harvey Mudd, Princeton, and Winona State universities to create introductory computing courses with a focus on non-computer science undergraduate majors, in particular science students, in fields ranging from physics to chemistry to biology (Cortina (2007); Sedgewich & Wayne (2008); Sedgewich & Wayne (2015); Wilson et al. (2008); Wing (2006)). These courses were produced in collaboration with science faculty, and are intended to begin each student’s journey into computing. Students are presented examples in a familiar language that allows them to focus on the foundational principles of each computational problem (Hambrusch et al., 2009).

The research on undergraduate level computing courses directly relates to the computational abilities of graduate level science researchers, as these students have most likely graduated from an undergraduate science program with little to no computational training. The concepts emphasized in the courses developed at Purdue, and elsewhere, can be used to inform the environmental sciences and Statistics departments as to what computational skills other scientific disciplines, such as physics, biology, and chemistry, believe to be the most important for students to grasp.

* 1. Computational Training for graduate science majors

Researchers in the Department of Biological and Biomedical Sciences at Harvard have developed an intensive course that introduces graduate students to the "fundamentals of programming, statistics, and image and data analysis through the use of MATLAB" (Stefan et al., 2015. p. 2). This course is framed not only with the goal of students developing programming skills, but also emphasizing students learning how to algorithmically reason through a computational problem, and developing statistical understandings. The structure of the 50-hour course dedicates the first two days to an introduction to programming using MATLAB, where students learn a variety of topics, including creating variables, performing basic variable operations, indexing, logicals, functions, conditionals, and loops. Day 3 is dedicated to developing statistical understandings, including probability distributions, hypothesis testing, p-values, bootstrapping methods, and multiple testing. These courses are given twice a year, once prior to the start of the school year as new graduate students are attending orientation, and a second time for "students who realize the need for such training later in their studies" (Gutlerner & Van Vactor, 2013).

In introducing beginning graduate students to these concepts, researchers hoped to lower the computational barrier for students taking courses, empower students to learn computational tools on their own, and allow for other courses to "build upon this foundation and integrate quantitative methods throughout the curriculum" (Stefan et al., 2015. p. 2). Survey results from the last five course offerings (Spring 2012 to Spring 2014) indicated that “students report significant gains in their self-assessed programming ability,” with students reporting that some of the concepts “around statistics [are the] most challenging” (Stefan et al., 2015. p. 8). These surveys also indicated that, following completion of the course, students believed they had acquired practical quantitative and computational skills that would prepare them for research in their field, recognized the importance of computational and quantitative methods in their field, felt confident in the methods they had learned, and would recommend that other graduate students learn these types of methods.

Graduate level terminal statistics courses, such as Methods of Data Analysis I and II, are taken by graduate environmental science students across the country, potentially acting as the sole or final computational training students receive prior to performing independent research. Examining such courses provides a natural extension of the research on computational and statistical training of graduate students in the biological sciences, as both provide the computational training used by graduate students throughout their coursework and in their independent research. Environmental science students experience similar computational burdens to students in the biological sciences, with computational expectations often being placed on them with little to no training. The intention of this study was to close this gap in the research, and to understand the resources environmental science graduate students invoke when learning and reasoning through applications of statistical computing in their field.

1. Methodology

For this study, a pragmatic phenomenological approach was appropriate, as the intention was to understand and describe common experiences in computational thinking and abilities for environmental science graduate students when applying their computational skills and understandings to applications in their field. The focus of this study, on the actions taken by students in the process of computationally reasoning through an application, lent itself naturally to a pragmatic framework. This framework allowed for an emphasis on the process of finding a working solution, allowing for varied solutions rather than a single solution (Creswell, 2013).

We examined factors that impacted how environmental science graduate students gained computational knowledge and the ability to reason through applications in their disciplines. Unlike typical definitions of computational knowledge and abilities, which focus on a student's understanding and fluency of computer programming, in our definition, we chose to also include knowledge of data structures, the ability to reason through problems with a given set of tools, as well as knowledge of resources that could provide assistance in solving a particular problem.

* 1. Participants

Students were selected from the Methods of Data Analysis II course in spring of 2017. These students were sampled following their spring break, nearly halfway through the course. Only graduate students from environmental science fields were considered. These students were taking the course for their respective master’s or doctoral programs.

Students were requested to complete a survey detailing their previous statistics and computer science courses, describe the computer languages they were familiar with, and outline their independent research experience. A total of eight graduate environmental science students were enrolled in this course in the spring of 2017, all of which completed the survey. All eight of these students were then asked to participate in an in-depth interview, of which five agreed. All of the students who agreed to be interviewed identified as female, and all exhibited the following characteristics:

* had taken Methods of Data Analysis I in the last two years,
* had a variety of programming backgrounds, and
* had a variety of levels of independent research experience.

Further details of the five participants are summarized in Table 1.

Table 1. Academic demographics of participants: degree seeking, program of study, academic year they took Methods of Data Analysis (MDA) I (Fall, Spring), number of statistics and computer science (CS) courses they have taken (undergraduate and graduate), programming languages they are familiar with, and amount of independent research they had completed.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***Name (pseudonym)*** | ***Degree*** | ***Program*** | ***MDA I*** | ***Stat Courses*** | ***CS Courses*** | ***CS Languages*** | ***Independent Research*** |
| Beth | MS | Animal Range Science | F 2015 | 4 | 0 | SQL | Thesis |
| Catherine | MS | Environmental Science | F 2015 | 1 | 0 | SQL | Thesis |
| Kelly | MS | Ecology | S 2016 | 3 | 1 | SQL | Thesis |
| Robin | PhD | Environmental Science | F 2015 | 2 | 2 | Python, SQL | A few projects |
| Stephanie | MS | Environmental Science | F 2015 | 4 | 0 | Python, Java, SQL | Thesis |

1. Data Collection

Following the preliminary survey, students who agreed to be interviewed were asked to describe their coursework, where and how they acquired their computational knowledge, and discuss their experiences in acquiring these understandings. This interview was modified from surveys administered by researchers investigating computational thinking at Harvard using Scratch, which provided a rich rubric of assessing students' experiences in performing computational applications (“Interviewing Students,” n.d.). The full interview protocol is included as an Appendix.

Following the interview, students were asked to work through an ecological application of statistical computing. These problems assessed students' abilities to reason through applications of statistical computing , and outlined any gaps noticeable in students' ability to transfer their computational knowledge to applications. Computational problems were developed in collaboration with statistics faculty at the university. The analysis in this paper is based on the five interviews on computational knowledge aquisition; students' abilities to reason through computational applications will be used in future analyses.

* 1. Data Analysis

Interviews and survey responses for each participant were transcribed, with participants' names removed and pseudonyms given. The interviews were then open coded so as to describe the statistical computing skills they used to complete each task, and how they acquired their knowledge of these concepts. The transcripts were read numerous times in order to segment the data and construct themes specific to each individual's acquisition of statistical computing skills. With these individual themes, we were able to compare commonalities that emerged across participants.

Participants were provided with an itemized detail of how they completed the problem and the transcription of their interview. The inclusion of member checking allows participants to check for accuracy of their statements. The authenticity of the study, its ability to capture the true experiences of students' abilities to think through computational problems, is enhanced with the lack of researcher engagement with students prior to their participation in the study. This ensured that no student felt more comfortable in the interview environment, articulating their experiences, than any other student.

1. results

Three main themes emerged from the interviews during the data analysis process. The names of these themes, their characteristics, and sample responses are described in this section.

The first theme describes how peer support had impacted participants' abilities to transfer statistical computing skills to applications in their field. The second theme provides participants' descriptions of how a singular graduate student or faculty “consultant” influenced the development of their computational abilities. The third theme recounts how participants' independent research experiences had affected the development of their computational abilities, and helped to emphasize the importance of computational literacy to their field. In the following sections, themes are delineated with quotations from participants to ensure authenticity of descriptions of their experiences.

* 1. Peer Support

All participants spoke of the support they had received from fellow graduate students when performing computational tasks. The students described how, when they are unsure of how to complete a computational task for their research, they turned to fellow graduate students for help. Participants described instances when the computational tasks required of them were beyond their current knowledge or occasions when they had attempted to complete a problem with all of their knowledge and sought out help from a fellow graduate student. For example, Kelly, an Animal Range Science master’s student, shared that when she reached a point in coding when she doesn't know how to do something she turned to one of her lab-mates:

I've been to a point where I didn't know how to do something with my knowledge or what I can find online, and then I'll go to one of my lab-mates.

Catherine, a master’s student in Environmental Science, spoke of the expectations of her advisers that the computational problems she was being asked to perform were “easy, since she had all the information.” However, she has had numerous experiences where she did not have the knowledge necessary to perform the task or she was missing “little caveats" that kept her from fully being able to perform the tasks. When faced with these problems, she “reached out to previous students that had taken the course.”

Robin, a doctoral student in Environmental Science, reiterated Catherine's experiences, describing how she reached out to other graduate students in other labs for help with computational problems. Stephanie, also a doctoral student in Environmental Science, described how when she was faced with computational problems beyond her knowledge she had never been forced to “go beyond talking to her lab-mates” for assistance.

Peer support does not always provide an optimal solution, however. For example, Kelly described negative experiences when seeking computational assistance from graduate students not of close proximity to her:

When I'm struggling with something and I go to other grad students, they'll say “I did this the other day. I'll send you my code.” I've found most of the time I don't understand what they've done enough to plug in what I want and make it work. There have been a few times when making tables and plots and someone sends me their code and I can just plug in my data and it works just fine. I've had less success with that.

* 1. Singular Consultant

When describing whom they seek out for computational help, every participant described an all-knowing past or current graduate student whom they have saught out for computational assistance. These figures served as a singular consultant, with whom these students had the "best," most productive, experiences in finding solutions to computational problems that had arisen. For Beth, this singular consultant came in the form of a past graduate student from Animal Range Sciences who was hired to help faculty complete projects:

We have a guy who used to be a student in our department and then he was hired on again to help finish some projects, but he got his master’s in Statistics. He is very helpful with [pointing out what's wrong with your code]. He's very good with code and if I have a quick question he can always answer it.

For Kelly, another graduate student on her same project served as this consultant. Kelly described computational problems she had encountered in her thesis, when she turned to this particular graduate student for help, adding that other graduate students in their department also used this person as a consultant for their computational problems. “The other grad student on this project is so well versed in R that he's unofficially become the person that people go to with questions.”

Through her computational struggles, Catherine found assistance from previous graduate students from the department, but she found the most assistance from a previous graduate student “who had left the department and was off professionally somewhere else, but he still took the time to help walk me through [my code]."

One participant, Stephanie, a Environmental Science doctoral student, served as this singular computational consultant for the many members of the Environmental Science department. With her experiences teaching herself R, she was able to “explain code in a way that makes sense,” says Robin, a fellow Environmental Science doctoral student who has often saught out help from Stephanie. With an adviser from a computational background and a project which performs sophisticated statistical modeling, Stephanie “has to learn code.” Additionally, her laboratory often worked in collaboration with faculty from computer science, where she and her lab-mates were taught computer science coding practices and jargon. “Stephanie has gotten good at teaching it, because everyone on our floor is like ‘I can't do this, Stephanie help me’,” said Robin. Stephanie stated that graduate students have saught her assistance “daily” or “at minimum two to three times a week.” In contrast, when Stephanie experiences difficulty in performing computational tasks, she has found solace in her lab-mates and ultimately, when necessary, with her adviser:

My entire lab works in the same room and my adviser's door is always open. So if someone is having a major issue, whoever is in the room can hear that. If [my adviser] hears me ask [lab-mate] how to do something and he knows how, he just shouts how to do it. So it's a very group oriented dynamic. I've never had to go beyond the people in my lab.

* 1. Independent Research Experience

The third theme was the computational knowledge students acquired when they participated in independent research. Involvement in independent research helped students to take their course knowledge and transfer it to statistical computing applications, seeing the messiness of non-classroom applications. These experiences came predominantly in the form of working as a research assistant prior to entering graduate school, collaborating on a project in the first year of graduate school, or performing research for a master’s thesis, or ultimately a doctoral dissertation.

Catherine, who still faced everyday computational struggles, attributed the majority of her application-specific computational knowledge to her experiences in independent research. She emphasized the importance of understanding how to work in a computing environment, such as R, which she learned from performing research, before she began to transfer the statistical knowledge she had learned in the classroom:

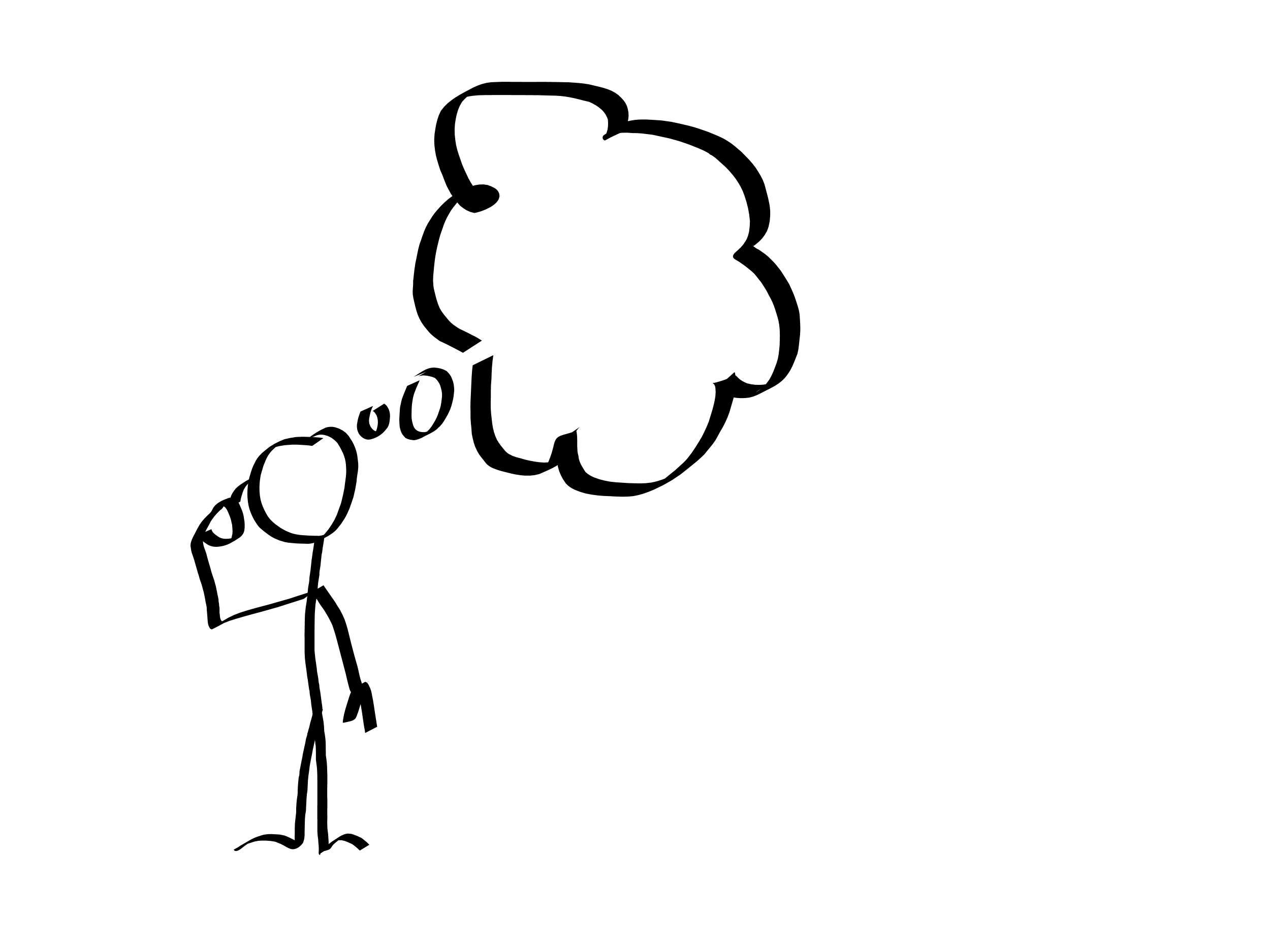
What I struggled with is [Methods of Data Analysis I] covers theory really well, but since I was new I spent most of my time trying to figure out how to apply that theory in [R]. And even now I struggle transferring from R into actual statistical theory, when I'm writing my thesis. The way I had to approach it was I had to learn the R first, then I was able to look back on what I had actually done, in order to learn the statistics.

Kelly described her experiences with data management for her master’s thesis as having produced the most substantial contributions to her computational abilities. Often she attributed her intuition for solving statistical computing problems to experiences she had “merging data sets” and learning to use conditional statements for her project. She emphasized the importance of her classroom Methods of Data Analysis knowledge in understanding “what statistical method to use,” but for becoming more computationally fluent she attributed that to her research experiences: “The data management stuff comes from independent research, trial and error, getting myself through.” Similar sentiments were voiced by Beth, with the majority of her computational knowledge stemming from her independent research. With the recommendation of her adviser, she taught herself how to create an Access database to store her data. In storing her project in this manner, she was able to learn important concepts about data structures, subsetting data “using qualifiers and criteria,” sorting data, all using SQL statements.

1. Discussion

The intention of this exploratory study was to describe and understand where environmental science graduate students gain computational knowledge and the ability to reason through statistical computing applications in their disciplines, as well as how student backgrounds differ in the impact of these factors. Students who participated in the study described their experiences in acquiring computational skills and their ability to reason through computing applications related to their field.

A path diagram, depicting the resources students rely on when faced with applications of statistical computing, is shown in Figure 1.

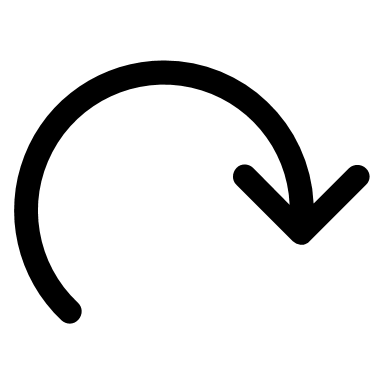
Figure 1. Resources used in reasoning through statistical computing applications.

**?**

Background, Independent Research,

Coursework







First, when these graduate students encountered a statistical computing problem they pulled upon the knowledge they had acquired through their undergraduate background, graduate coursework and independent research. For these participants, the elements of background that proved to be of the most assistance were their undergraduate statistics courses and pre-graduate research. The computational understandings that these students attributed to the Methods of Data Analysis I and II courses were primarily low-level concepts, such as logical statements, using built-in R functions, adding comments to their code, and limited trouble-shooting of error messages. Additionally, these concepts were found to only be fully understood through the use of peer interaction, as they were being implemented in their own research.

Participants voiced the importance of their experiences performing independent research as having a substantial influence on their abilities to reason through and perform the computational tasks required for various statistical analyses. Through independent research, the participants were able to play with real-world data and applications outside of what they had encountered in the classroom. The programming understandings informed by a student's independent research, in conjunction with peer collaboration, were described largely as high-level concepts, such as user-defined functions, conditional statements, and loop implementation. Students described their independent research as having opened the door to experiencing the unease that comes when one is asked to perform computational tasks beyond one's knowledge. In these circumstances, students stated that they would ask for help from the people with whom they feel the most comfortable.

In a direct connection to the participants' discomfort in asking for help from an adviser, the theme of a singular consultant emerged. These singular consultants serve as an all-knowing individual, from whom the participants had either had the “best” experiences with, where the individual spends the necessary time to explain the concepts, or the consultant had always been capable of providing the participant with an answer to their problem. These figures served as the first line of defense when compuational problems arose, where participants were both able to seek computational help and acquire new computational skills and understandings through their interactions. If due to time or physical constraints, this consultant was unavailable to the graduate student, they then turned to their peers.

Peer support was initially discussed by the participants in their interviews as a mechanism they used when their “code doesn't run” or when they were asked (or needed) to do something beyond their current computational understandings. However, this theme continued to emerge as the participants worked through computational problems, often attributing their knowledge of a computational procedure to a friend or fellow graduate student helping them “do it with their data.” These peers offered an avenue for students to seek help, often voiced to be more comfortable than asking an adviser, where participants described both the fear of asking and “feeling dumb,” or being “brushed off” because their adviser thought they should “be able to figure out how to do it.” However, as opposed to the help participants receive from their singular consultant, the students also voiced negative experiences they had encountered when seeking help from their peers, such as a peer sending them their code that they do not understand.

Lastly, the adviser played an important role in students acquiring the computational knowledge necessary to perform applications. Despite students’ reluctance to seek out computational assistance from their adviser, these individuals did often emphasize the importance of statistical computing skills, as well as introduced (or recommend) students to store their data using an Access database. The ability of many participants to understand both data structures and sorting or filtering data was largely attributed to their experiences working with these types of databases. Although this study found that advisers were often considered as the last line of defense, they were viewed as an accessible way for students to better understand the statistical computing necessary for their independent research projects, which overall contributed to better computational understanding and skills for these students.

The second purpose of this study was to describe how student backgrounds differ in the impacts of these themes on students' statistical computing abilities. The theme of research experiences, with its overall positive tone by participants, produced different experiences for students with fewer computational skills and understandings than students with more. The frustrations of simple tasks, such as subsetting data or removing NA's, were felt by the participants who had completed a bachelor's without any computational elements to their coursework, while those who were exposed to small amounts of computing in their undergraduate coursework, such as a general computer science courses, a GIS course, or experience with Access databases, were able to begin computational tasks in their research walking and not crawling.

The largest difference in the impacts of a factor between computational skill groups, came in the theme of a singular consultant. One participant, Stephanie, who entered graduate school after completing a year's work as a research assistant, working in R, instead served as the singular computational consultant for many graduate students in the Environmental Science department. She still described the theme of seeking help from her peers, predominantly her lab-mates, but, for her, the singular consultant was her adviser. Potentially due to her more substantial computational background, Stephanie voiced that she felt less of a power difference than her peers did in seeking help from her adviser.

While the methodology used to determine graduate environmental science students’ experiences with statistical computing in the present study provided useful themes, it is not without its limitations. Eliciting descriptions of computational knowledge acquisition yielded varied experiences with each of the main themes, but richer data could be gathered in a future longitudinal study, following graduate students throughout their program of study. While the current study provided a window into graduate environmental science students’ computation experiences, it is important for future studies to investigate what types of computational knowledge students are in acquiring throughout their coursework, and the computational burden students experience in varying aspects of their coursework and research.

Finally, it should be noted the present study focused on describing environmental science graduate students’ experiences in acquiring statistical computing knowledge, but not in what computational knowledge they possessed. Therefore, we have learned primarily about the resources students relied on when they experienced computational expectations beyond their ability. Again, in performing a longitudinal study, researchers would gain the ability to isolate the specific statistical computing knowledges students acquire and where they are learned.

1. conclusion

Statistical computing has become a foundational aspect of research in the environmental sciences. This small-scale study brings forward the experiences of graduate envrionmental science students in acquiring the computational understandings necessary to successfully perform field related statistical applications. These themes bring to light the factors students relied upon when gaining these computational understandings, and suggest the need for the integration of computational training into these programs. The present study begins the discussion of the computational knowledge necessary for research in the environmental sciences, as it provides an overview of the resources graduate students relied on in their aquistion of these understandings. To better inform faculty in these departments, a thorough investigation of both the coursework and structure of courses completed by these participants could be performed. This would allow for a discussion of how to best integrate these computational concepts into current coursework requirements, so that students leave the classroom with understandings they can implement immediately in their own research.

ACKNOWLEDGEMENTS

We would like to specially thank the participants from this study, without whom this research would not have been possible. We would also like to thank Jennifer Green and Megan Wikstrom for their insightful comments on this paper.

REFERENCES

Bloom, B. S. (1956). *Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain*. New York: David McKay Co Inc.

Cortina, T. (2007).An introduction to computer science for non-majors using principles of computation. *Proceedings of the 38th SIGCSE technical symposium on computer science education, 39*, 218–222.

Creswell, J. (2013). *Qualitative inquiry & research design* (3rd ed.). Thousand Oaks, California: Sage Publications.

Interviewing students about scratch programming experiences (n.d.). Retrieved 2018, May 26, from http://scratched.gse.harvard.edu/ct/files/Student\_Interview\_Protocol.pdf.

Fox, J. A., & Ouellette, B. F. F. (2013). Education in computational biology today and tomorrow. *PLOS Computational Biology, 9*(12), 1-2.

Gutlerner, J. L., & Van Vactor, D. (2013). Catalyzing curriculum evolution in graduate science education. *Cell, 153*(4), 731-736.

Hambrusch, S., Hoffmann, C., Korb, J., Haugan, M., & Hosking, A. (2009). A multidisciplinary approach towards computational thinking for science majors. *Proceedings of the 40th ACM technical symposium on computer science education*, 183-187.

Rubinstein, A., & Chor, B. (2014). Computational thinking in life science education. *PLOS Computational Biology, 10*(11), 1-5.

Sedgewich, R., & Wayne, K. (2008). *Introduction to programming in java*. Boston: Addison Wesley.

Sedgewich, R., & Wayne, K. (2015). *Introduction to programming in python*. Boston: Addison Wesley.

Stefan, M. I., Gutlerner, J. L., Born, R. T., & Springer, M. (2015). The quantitative methods boot camp: Teaching quantitative thinking and computing skills to graduate students in the life sciences. *PLOS Computational Biology, 11*(4), 1-12.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127-147.

Wilson, G., Alvarado, C., Campbell, J., Landau, R., & Sedgewich, R. (2008). CS-1 for scientists. *Proceedings of the 39th SIGCSE technical symposium on computer science education*, 36-37.

Wing, J. (2006). Computational thinking. *Communications of ACM, 49*(3), 33-35.

Allison theobold

1923 W beall street

bozeman, mt 59718

APPendix

Interview Protocol

Describe a time when your code didn't run as you wanted.

Describe how you investigated the cause of the problem and fixed the problem.

Describe other ways you could have fixed the problem.

Describe a time when you could not find a way to fix your code.

Where did you turn to for help and why?

Describe how you found advice or support by using someone else's code on a project or homework. Why did you seek out advice or support?

Describe a time you used the code from another homework or project as part of your homework or project. How often do you use previous code on a current project or homework?

Describe a time you modified existing code (either someone else's or your own) to improve or enhance it.

Where have you learned the statistical computing skills necessary for your course work and research?