Transferability of Computational Knowledge and Abilities to Field Related Applications: A Phenomenological Study of Ecology and Environmental Science Graduate Students

Allison Theobold

October 4, 2017

Abstract

Introduction

Computing has become foundational in the fields of environmental sciences, however, the gap between environmental science education and computation has become more evident. With the growth in computational power, the complexity of environmental science 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 begun to require life science students to enroll in an introductory programming course [20], while others are providing computing bootcamps for quantitative methods [**?**].

This research focuses on graduate students enrolled at a mid-size university in the Rocky Mountains, where Methods of Data Analysis I and II are graduate statistics courses required for the completion of a Master's degree in Ecology or the Environmental Sciences. These terminal statistics courses intend to prepare graduate students for the statistical and computational problems they may face as researchers and practitioners. These courses potentially provide students with their final statistical and computational skills before they begin performing independent research. By examining how Ecology and Environmental Science graduate students apply their computational skills, the present study seeks to capture a more in-depth understanding of the successes and shortfalls they experience when implementing computational thinking and skills, given their background.

Computational Thiking and Abilities of Life Science Students

Research in computational thinking for science majors is in its infancy, with only a handful of institutions performing research that specifically addresses how to teach and how students learn computational thinking. Primarily, the research being performed is implemented at the undergraduate level, with no research efforts made in the realm of graduate science students. Due to this gap in the literature, the reviewed literature were selected from research on computational thinking and abilities of undergraduate science students

Efforts in Understanding Computational Thinking

The most substantial efforts being made in understanding computational thinking are by researchers at the Harvard Graduate School of Education and researchers at EDC's Center for Children and Technology. Their efforts focus on "supporting and assessing the development of computational thinking through programming" [22] for both researchers and K-12 teachers. They have "relied primarily on three approaches: (1) artifact-based interviews, (2) design scenarios, and (3) learner documentation." [3] These scenarios were developed in collaboration with researchers at the Education Development Center and are used in the context of Scratch, a computer programming environment for use by elementary through high school students, created by the Lifelong Kindergarten Group at MIT Media Lab. The researchers at Harvard developed three different Scratch projects of increasing complexity, and in a series of interviews “students were presented with the design scenarios, which were framed as projects that were created by another young Scratcher. The students were then asked to select one of the projects from each set, and (1) explain what the selected project does, (2) describe how it could be extended, (3) fix a bug, and (4) remix the project by adding a feature." [22] Through these interviews, substantial contributions have been made in how to assess the ways students engage in computational thinking, with the ability to apply the structure of these interviews to other types of students and other types of computational thinking scenarios.

Computational Courses for Undergraduate Science Majors

Multi-disciplinary efforts have been made at Purdue, Carnegie-Mellon, Harvey Mudd, Princeton, and Winona State ([4], [6], [24], [25], [27], [28]) to create introductory computing courses with a focus on non-computer science majors, in particular science students, in fields ranging from physics to chemistry to biology. These courses are produced in collaboration with science faculty, and are intended to “lay the groundwork for computational thinking, present examples in a language familiar to the students, teach in a problem-driven way, use a programming language that right away allows students to focus on computational principles, and make effective use of visualization." [12] The structure of the computation course was informed by the physics, chemistry, and bioinformatics departments, where they provided computer science researchers with expectations on what they wished the students would learn. The structure of these undergraduate level statistics courses directly relates to the computational abilities of graduate level science researchers, as they may have graduated from an undergraduate program with no computational training. The concepts emphasized in the courses developed at Purdue, and elsewhere, can be used to inform the graduate level statistics methods courses what skills other scientific disciplines, such as physics, biology, and chemistry, believe to be the most important for students to grasp.

Given the current research focuses on assessing computational thinking and creating computational courses for undergraduate science majors, we see substantial gaps in understanding how these same issues relate to graduate science students. The statistical methods courses, taken by graduate Ecology and Environmental Science students, potentially serve as their final computational training prior to performing independent research. These such graduate level statistics courses, taken by graduate Ecology and Environmental Science students across the country, provide a natural extension to the current research on computational skills and thinking. It is the intention of this study to close this gap in research and understanding between the computational thinking and skills of undergraduate science students and that of graduate science students.

Methodology

For this study, a pragmatic phenomenological approach is appropriate as the intention is to understand and describe common experiences in computational thinking and abilities for Ecology and Environmental Science graduate students, in applying their statistical computing skills 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, lends itself naturally to a pragmatic framework. This framework allows for a focus on the process of finding a working solution, allowing for varied solutions rather than a single solution [5].

We define the transferability of computation abilities as the ability to apply acquired computing skills from the Methods of Data Analysis classroom to applications related to ecological fields. Unlike typical definitions of computational abilities, which focus on a student’s understanding and fluency of computer program(s), we chose to include computational thinking in our definition. In relation to ecological applications, this can include, fluency in performing computing tasks (programming abilities), 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.

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 the Ecology and Environmental Science departments were considered. These students are taking the course as a coursework requirement for their respective Masters programs.

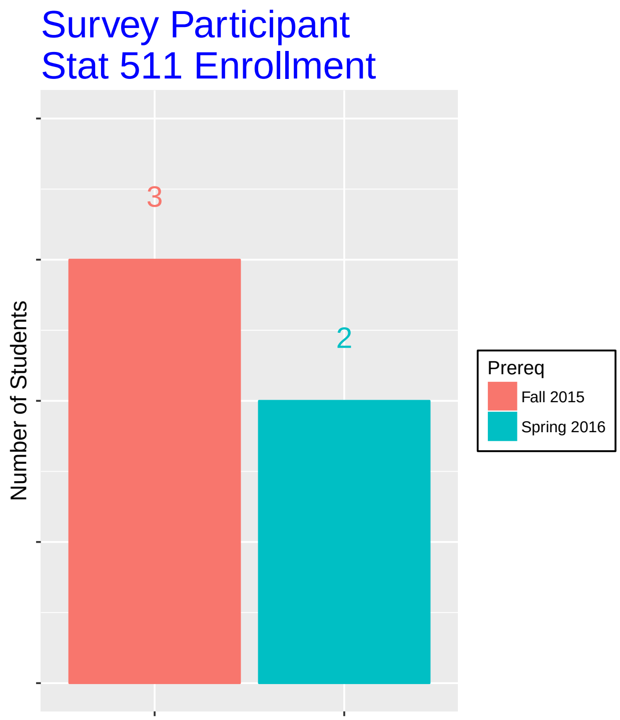
Students were requested to complete a survey detailing their previous courses, statistics, computer science, and field related, describe their computer programming abilities, and outline their research experience. There were a total of 17 graduate Ecology and Environmental Science students enrolled in this course, and a criterion sampling procedure was used. The students requested to participate in the study were selected so that,

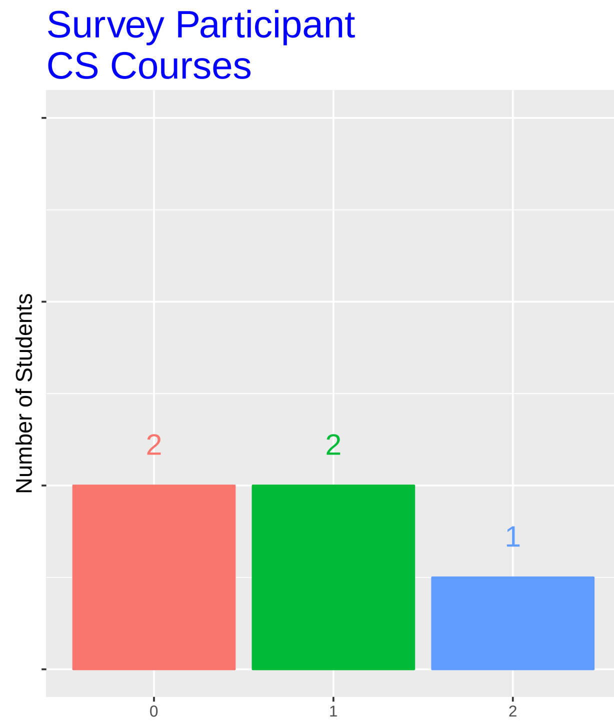
- they had taken Statistics 511 in the last 2 years,
- they had a variety of programming backgrounds, and
- they had a variety of levels of independent research experience.

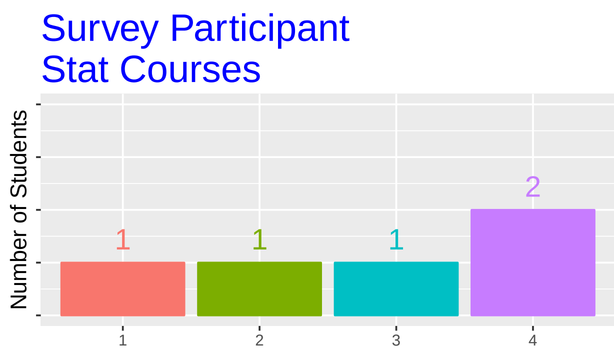
These students were requested to participate in the preliminary survey, itemizing their course work, their program of study, and their independent research experience. A total of 16 students completed the survey, eight of which were graduate students in the areas of Environmental Science and Ecology. All of the potential participants were requested to participate in the study, of which five agreed. All of the students who agreed to participate identified as female. The table and plots below describe the demographics of the study participants.

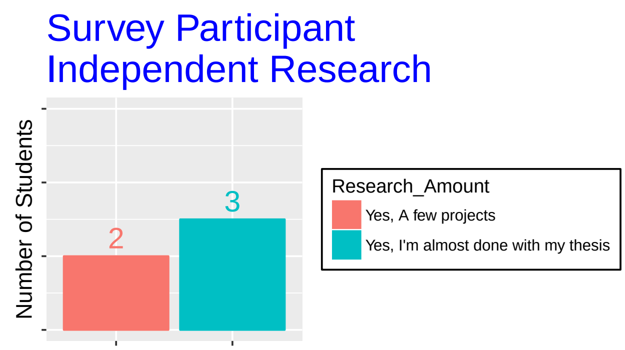
Name (pseudonym)	**Degree**	**Program**
Beth	Master's	Animal Range Science
Catherine	Master's	Environmental Science
Kelly	Master's	Ecology
Robin	Doctorate	Environmental Science
Stephanie	Master's	Environmental Science

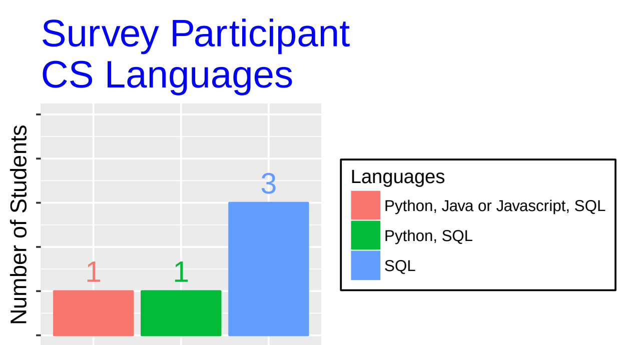
Table 1: Sample Description


Survey Participant
Stat 511 Enrollment
Number of Students
3
2
Prereq
Fall 2015
Spring 2016


Survey Participant
CS Courses
Number of Students
2
2
1
0
1
2


Survey Participant
Stat Courses
Number of Students
1
1
1
2
1
2
3
4


Survey Participant
Independent Research
Number of Students
2
3
Research_Amount
Yes, A few projects
Yes, I'm almost done with my thesis


Survey Participant
CS Languages
Number of Students
1
1
3
Languages
Python, Java or Javascript, SQL
Python, SQL
SQL

Data Collection

Following the preliminary survey, students who agreed to be interviewed were given a questionnaire detailing their coursework, detailing their acquired computational knowledge, and discussing their experiences in acquiring this knowledge. These surveys were modified from surveys administered by researchers investigating computational thinking at Harvard using Scratch, which provided a rich rubric of assessing students' evolving fluency in computational practices [3].

Following the survey, students were asked to complete an ecological application of statistical computing. These problems assessed students' abilities to both statistically and computationally reason through applications, provide descriptions of how they computationally think through the problems, as well as outline the 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, using data from the Blackfoot River, collected by Montana Fish, Wildlife, and Parks in years 1989 to 2006.

It should be noted that a potential limitation of using an interview environment for students to computationally reason through a set of problems is the possibility that students felt nervous, pressured, or fearful of not knowing a correct process to achieve the desired result. Additionally, as a doctoral student in statistics, it is possible that the students felt a power difference, specifically regarding computational knowledge. To alleviate these feelings, the interviewer worked on the same problem beside the participants, acting as a collaborative peer. The interview protocol included the interviewer providing hints if/when a student got stuck or was unsure of how to proceed. This environment is believed to have helped aid suppressing fear and allowed students to more openly work through problems with the researcher.

Data Analysis

Interviews and survey responses for each participant were transcribed, with participants names removed and pseudonyms given. The interviews were analyzed, so as to describe the thought process of each student working through the application, describing the concepts they used to complete each task, why they believed these to be the correct concepts to use, and how they acquired their knowledge of these concepts. Themes specific to each individual's computational thinking processes were then described, including, but not limited to, synthesis of experiences, coursework, and learned skills. With these individual themes, we were able to compare across individuals commonalities that emerge.

Participants were provided with an itemized detail of how they completed the problem and the transcription of their interview. This allowed participants to check for accuracy of their statements and the interviewers description of their thought processes in completing the problems. The authenticity of the study, it's ability to capture the true experiences of students' abilities to think through computational problems, is enhanced with the lack researcher engagement with students prior to their participation in the study. This ensured that no student felt more comfortable in the interview environment, working through the application, than any other student.

Results

The three main themes which 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 has impacted participants' abilities to transfer computational skills to applications in their field. The second theme provides participants' descriptions of how an single graduate student or faculty influenced their computational abilities. The third theme recounts how participants' independent research experiences have affected their computational abilities. In addition, their research experiences helped to emphasize the importance of computational literacy to their field. In the following section, themes are delineated with quotations from participants to ensure authenticity of descriptions of their experiences.

Peer Support

All participants spoke of the support they have 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 turn 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, a Animal Range Science Master's student, shared that when she reaches a point in coding when she doesn't know how to do something she turns 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 is faced with computational problems beyond her knowledge she has never been forced to "go beyond talking to her lab-mates" for assistance.

However, Kelly has also had 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.

Singular Consultant

When describing whom they seek out for computational help, every participant described an all-knowing past or current graduate student whom they seek out for computational assistance. These figures serve as a single consultant, with whom these students have had the "best," most productive, experiences in finding solutions to computational problems that have arisen. For Beth, this single consultant comes 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 answer it.

For Kelly, another graduate student on the same project as her serves as this consultant. She described computational problems she has encountered in her thesis, when she turned to this particular graduate student for help, adding that other graduate students in their department also use 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, serves as the computational consultant for the many members of the Environmental Science department. With her experiences teaching herself `R`, she is able to "explain code in a way that makes sense," says Robin a fellow Environmental Science doctoral student who often seeks out help from Stephanie. With an adviser from a computational background and a project which performs computer modeling, she "has to learn code." Additionally, her laboratory often works in collaboration with faculty from the computer science department, where she and her lab-mates are 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,' " says Robin. Stephanie stated that graduate students seek her assistance "daily" or "at minimum two to three times a week." In contrast, when Stephanie experiences difficulty in performing computational tasks, she finds 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.

Independent Research Experience

The third theme was the computational knowledge students acquired while participating in independent research. Involvement in independent research helped students to take their course knowledge and transfer it to applications, seeing how messy non-classroom applications can be. 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, and performing research for a Master's thesis, or ultimately a doctoral dissertation. Catherine, who still faces everyday computational struggles, attributes 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 could begin to transfer the statistical knowledge she learned in the classroom.

> What I struggled with is 511 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 producing the most substantial contributions to her computational abilities. Often she attributed her intuition for solving computational problems to experiences she had, "merging data sets" and learning to use conditional statements for her project. She emphasized the importance of her classroom 511 and 512 knowledge in understanding "what statistical method to use," but for becoming more computationally fluent she attributes 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 in data structures, subsetting data "using qualifiers and criteria," sorting data, all using SQL statements.

Discussion

The intention of this study was to describe and understand what factors impact Ecology and Environmental Science graduate students' computational knowledge and ability to reason through applications in their disciplines. Students who participated in the study both described their experiences with computational thinking and their ability to reason through applications related to their field, as well as worked through computational problems, which challenged their computational understandings and abilities.

Three themes arose from the study, characterizing the factors with the greatest impact on the participants' experiences in acquiring computational knowledge and abilities, as related to applications in their fields. First, participants voiced the importance of their experiences performing independent research as a substantial influence on their abilities to reason through and perform computational problems. Through independent research, the participants were able to play with real-world data and applications outside of what they saw in the classroom. These experiences also opened the door to the unease that comes when one is asked to perform computational tasks beyond one's knowledge. In these circumstances, the participants discussed their reliance on peers and singular consultants to aid them in accomplishing computational tasks.

The second and third themes, focus both on the participants acquisition of computational knowledge and skills, as well as how they cope when faced with computational expectations beyond their ability. The theme of peer support was initially discussed by the participants, in their interviews, as a mechanism they use when their "code doesn't run" or when they are asked (or need) to do something outside of their current computational understandings. However, this theme continued to emerge as the participants worked through the computational problems, often attributing their knowledge of computational procedures to a friend or fellow graduate student helping them "do it with their data." These peers offer 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," as well as being "brushed off" because their adviser thought they should "be able to figure out how to do it."

In a direct connection to the participants' discomfort in asking for help from an adviser, the third theme of a singular consultant emerged. These singular consultants serve as an all-knowing individual, from whom the participants have either had the "best" experiences with, where the individual spends the necessary time to explain the concepts, or the consultant has always been capable of providing the participant with an answer to their problem. These figures serve a similar role to peers, where the participants are both able to seek computational help and acquire new computational skills and understandings through their interactions. However, as opposed to the help participants received from peers, the students never voiced any negative experiences when seeking help from these consultants.

The second purpose of this study was to describe how these factors impacted students' computational abilities across different levels of understanding. 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 completed a Bachelors without any computational elements to their coursework, while those who were exposed to small amounts of computing in their undergraduate coursework, such as 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 serves as the computational consultant for the Environmental Science department. She still described the theme of seeking help from her peers, predominantly her labmates, but, for her, the singular consultant was her adviser. Potentially due to her larger computational understanding, Stephanie voiced that she feels less of a power difference than her peers, when seeking help from her adviser.

Conclusion & Implications

The findings of this study, better inform a concept map of how students acquire the skills necessary to perform computational applications in their fields. With the emergent themes describing where students are acquiring these necessary skills, the below concept map reflects these changes.

We found that the participants do not learn the concepts, related to accomplishing computational applications in a hierarchy. The importance of background remains, however. For these participants, the elements of background that proved to be of the most help were both undergraduate statistics courses and pre-graduate

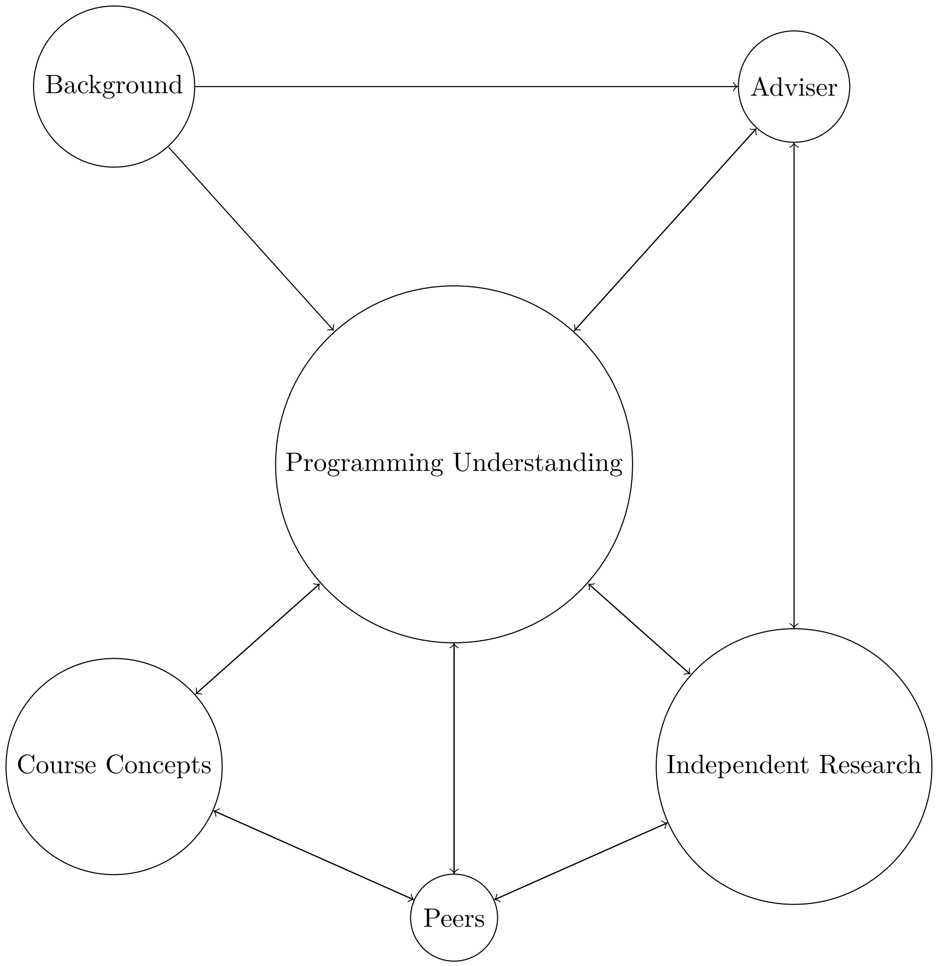


Figure 1: Computational Knowledge Acquisition Concept Map

research. The adviser plays an important role in students acquiring the computational knowledge necessary to perform applications, by both emphasizing the importance of these skills, as well as introductions (or recommendations) for students to store their data in 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. Additionally, this study found that, although they are not often used, advisers are viewed as an accessible way for students to better understand the computation necessary for their independent research projects, which overall contributes to better computational understanding and skills for these students.

At the center of the map we find programming understanding the bottleneck described by all students in their abilities to successfully accomplish computational applications in their fields. This understanding is informed, albeit weakly, by students' backgrounds and their advisers. However, the majority of the students' understanding comes from their course work, research, and collaboration with their peers. The programming understandings that these students attributed to their coursework, were primarily the low-level concepts described in the **Definition of Terms**. These concepts were found to not directly inform a students research, but instead the concepts were understood through the use of peer interaction prior to implementation in their own research. The programming understandings informed by a student's independent research, in conjunction with peer collaboration, were described by participants to be largely high-level concepts. These understandings stem from the need to find computational solutions to the applications they are currently working with. These solutions often require higher level programming concepts, but they are accomplished without students learning the lower level concepts at work in the solutions.

Implications

This study better informs the faculty and staff at this institution of the computational needs of graduate students in the fields of Ecology and Environmental Sciences. In the discussion of how students are acquiring the computational understandings necessary to successfully perform field related applications, we are able to see how both instruction and learning could be improved. It is possible that students are acquiring computational skills in the classroom, however there is also the possibility that the concepts being taught are not at a low enough level for students to understand them independently of their peers. To better inform faculty in these departments, a thorough investigation of both the coursework (syllabus) and structure (lecture, laboratory, etc.) of courses completed by these participants could be performed. This will allow for a discussion of how to best teach these computational concepts, so that students leave the classroom with understandings they can implement immediately in their own research.

Apendix

Definitions of Terms

Background

The background could change a student's ability to reason through applications in a variety of ways. For example,

- it can be associated with what computer programming knowledge they have going in to the courses,
- it can be associated with what statistical and mathematical knowledge they have going in to the course,
- taking inquiry based statistics, mathematics, or courses related to their field can change their ability to reason through application problems, and
- being exposed to research based courses can shed light on how statistics and computer programming are being used in their field.

Attitude

As discussed in the literature of science education, student attitudes could affect learning and achievement in a variety of ways [23]. For example, a student could

- have a negative attitude toward taking statistics, computer science, or other computational courses,
- have low motivation in learning concepts taught in background and/or statistics courses,
- not value the material they are taught in STAT 511/512,
- remain calm or become irritated when working through new computational applications.

Low-level Programming Concepts

These are concepts potentially learned in a computer science (CS) course or a field related course. They are, however, concepts that are explicitly taught in STAT 511.

- Logical statements (e.g. if else, true/false)
- Using built in functions in R (or other software)
- Commenting their code, to know why they did what they did
- Trouble-shooting error messages

High-level Programming Concepts

These are concepts also potentially learned in a CS course or a field course. They are concepts explicitly taught in STAT 512.

- Building functions (from scratch)

- Objects and classes of objects (numeric/character/factor)
- Loops
- Conditional statements

Data Structures

These are concepts related to both basic and more advanced understandings of data formats. These concepts could have been introduced in an Introductory Statistics course, an -ometrics course, or a course related to their field. These concepts are explicitly taught in STAT 511.

- Data Format (what the data look like)
  - Retrieving the data (importing it, irregardless of format)
  - Naming variables
  - How to handle missing data
- Manipulating Data (data wrangling)
  - Creating variables (new or from existing variables, polynomials, and interactions)
  - Subsetting data (removing rows/columns)

References

[1] Bazeley, P. (2009). Analyzing qualitative data: More than 'identifying themes,' The Malaysian Journal of Qualitative Research, 2, 1-18.

[2] Colyar, J. (2009). Becoming writing, becoming, writers, Qualitative Inquiry, 15(2), 421-436.

[3] Computational Thinking | Assessing. (n.d.). Retrieved February 26, 2017, from http://scratched.gse.harvard.edu/ct/assessing.html

[4] Cortina, T. J.(2007). An introduction to computer science for non-majors using principles of computation. In I. Russell, S. M. Haller, J. D. Dougherty, and S. H. Rodger, editors, Proceedings of the 38th ACM SIGCSE Technical Symposium on Computer Science Education, 218-222.

[5] Creswell, J. W. (2013). Qualitative Inquiry & Research Design. Thousand Oaks, California: Sage Publications.

[6] Dodds, Z., Libeskind-Hadas, R., Alvarado, C., & Kuenning, G. (2008). Evaluating a breadth-first CS 1 for scientists. In SIGCSE '08: Proceedings of the 39th ACM SIGCSE Technical Symposium on Computer Science Education, 266-270.

[7] Foss, S. & Waters, W. (2007). Chapter 4. Developing the itinerary: The preproposal. Destination Dissertation: A Traveler's Guide to a Done Dissertation (pp.35-71). Lanham, MD: Rowman & Littlefield.

[8] Foss, S. & Waters, W. (2007). Chapter 6. Getting there: The dissertation proposal. Destination Dissertation: A Traveler's Guide to a Done Dissertation (pp. 113-163). Lanham, MD: Rowman & Littlefield.

[9] Geyer, T. (1983). An Introduction to Writing Computer Programs in Ecology: Its Educational Value. Journal of Biological Education, 17(3), 237-242. Retrieved from https://eric.ed.gov/?id=EJ290422

[10] Green, J., Hastings, A., Arzberger, P., Ayala, F., Cottingham, K., Cuddington, K., Davis, F., Dunne, J., Fortin, M., Gerber, L., & Neubert, M. (2005). Complexity in Ecology and Conservation: Mathematical, Statistical, and Computational Challenges. BioScience, American Institute of Biological Sciences, 55(6), 501-510. Retrieved from https://academic.oup.com/bioscience/article/55/6/501/363547/Complexity-in-Ecology-and-Conservation

[11] Gross, L. J. (1994, February). Quantitative Training for Life-Science Students [Editorial]. Viewpoint. Schatz, M. (2012). Computational Thinking in the Era of Big Data Biology. Genome Biology, 13 (11), 177. Retrieved from http://dx.doi.org/10.1186/gb-2012-13-11-177

[12] Hambrusch, S., Hoffmann, C., Korb, J., Haugan, M., & Hosking, A. (2009). A Multi-disciplinary Approach Towards Computational Thinking for Science Majors. Retrieved from http://dl.acm.org/citation.cfm?id=1508931

[13] Hildreth, L. (2016). Statistics 411/511, Methods of Data Analysis I [Syllabus]. Montana State University, Department of Mathematics and Statistics.

[14] Maxwell, J. M. (2013). Qualitative Research Design: An Interactive Approach. Thousand Oaks, California: Sage Publications.

[15] Motivation. (n.d.). In Wikipedia. Retrieved February 22, 2017, from https://en.wikipedia.org/wiki/Motivation

[16] Padro, F. F. (2010). Policy Interests Driving Promotion of STEM Programs at Higher-Education Institutions. ASQ Education Brief.

[17] Randolph, J. J. (2009). A guide to writing the dissertation literature review, Practical Assessment, Research and Evaluation, 14(13), 1-13.

[18] Ravitch, S. M., & Riggan, M. (2012). Reason & rigor: How conceptual frameworks guide research. Thousand Oaks, California: Sage Publications.

[19] Robison-Cox, J. (2014). Statistics 505, Linear Models [Course Notes]. Montana State University, Department of Mathematics and Statistics.

[20] Rubinstein A, Chor B (2014) Computational Thinking in Life Science Education. PLOS Computational Biology 10(11): e1003897. doi: 10.1371/journal.pcbi.1003897

[21] Savin-Baden, M., & Major, C. H. (2013). Chapter 5: Personal stance positionality and reflexivity. Qualitative Research: The Essential Guide to Theory and Practice. London: Routledge.

[22] Scratch - Imagine, Program, Share. (n.d.). Retrieved February 26, 2017, from https://scratch.mit.edu/

[23] Schibeci, R. & Riley II, J. (1986). Influence of students' background and perceptions on science attitudes and achievement. Journal of Research in Science Teaching, 23(3), 177-187.

[24] Sedgewick, R. & Wayne, K. (2008). Introduction to Programming in Java: An Interdisciplinary Approach. Addison Wesley.

[25] Sedgewick, R. & Wayne, K.. Introduction to Computer Science. Addison Wesley, in preparation.

[26] United States Department of Education, A Test of Leadership: Charting the Future of U.S. Higher Education, Washington, D.C.

[27] Wilson, G., Alvarado, C., Campbell, J., Landau, R., & Sedgewick, R. (2008). CS-1 for scientists. SIGCSE Bull., 40(1):36-37.

[28] Wing, J. M. (2006). Computational thinking. Commun. ACM, 49(3):33-35.