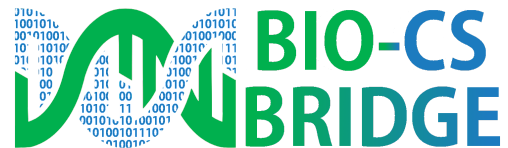


Developing an Integrative Biology and R Curriculum to Bridge Gaps in Data Literacy and Reduce Inequity in Early Professional Outlook

An Interactive Qualifying Project
Submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Bachelor of Science

By
Chloe Byrne



Date:

19 December 2022

Project Presented To:

Professor Elizabeth Ryder, Advisor

This report represents the work of one or more WPI undergraduate students submitted to the faculty as evidence of completion of a degree requirement. WPI routinely publishes these reports on the web without editorial or peer review.

Abstract

As the world becomes increasingly dependent on technology, it is becoming important for early professionals to demonstrate data and computer literacy. However, barriers exist in ensuring equitable access to these skillsets. This project seeks to combat such gaps in accessibility through the addition of a new unit to the Bio-CS Bridge curriculum. The unit will teach high school students the basics of statistics and data visualization using R and expose them to relevant intersections between computer science and biology.

Acknowledgements

I want to thank the following individuals for their support and advice throughout the completion of this IQP. Thank you to my advisor Professor Elizabeth Ryder, for her consistent support and insight. She was always available for help and was patient with me as I navigated the trials and errors of curriculum development. Thank you to Professor Carolina Ruiz for introducing me to the current workings of Bio-CS Bridge at the beginning of the project through weekly team meetings. I would also like to thank Kerri Murphy, Joshua Schroeder, Maureen Chase, and Jennifer Hardy, the educators who were kind enough to meet with me and provide their insight into my proposal for the curriculum. I appreciate everyone's time and advice throughout this process and wish the best to all who proceed with the curriculum!

Table of Contents

Abstract	1
Acknowledgements	2
List of Figures	5
List of Tables	5
1 Introduction	6
2 Background and Literature Review	9
2.1 Equity and Representation in STEM Education	9
2.2 Identity and Perception as Predictors of Academic Performance	11
2.3 Complex Hypothesis Testing and Contextual Barriers in STEM	12
2.3.1 Computer Science As a Bridge to Accessibility	13
2.3.2 The Bio-CS Bridge Initiative	14
2.3.3 Beecology	15
2.3.4 Defining Computational Thinking	16
2.4 Barriers to Interdisciplinary STEM Education and Making Computational Thinking Accessible	17
3 Methodology	19
3.1 Choosing R and Initially, Replit	19
3.2 Adjustment to R Studio	22
3.3 Creating Approachable Datasets	23
3.4 Supporting Both Teacher and Student	24
3.5 Ensuring Diversity in Curriculum	25
3.6 Backwards Design	26
4 Results	27
4.1 Implementing Backwards Design For Overall Curriculum	28
4.1.1 Identifying Desired Outcomes	29
4.1.2 Designing Effective Assessments and Activities	30
4.2 Developing Activities to Support Independent Problem Solving	32
4.3 Ensuring Diversity in Curriculum and Combatting Imposter Syndrome	34
5 Conclusions and Recommendations	36
5.1 Conclusions	36

5.2 Recommendations	36
References	40
Appendix A: Example of a Dataset	43
Appendix B: Graphing with Beecology Lesson	44
Appendix C: Vectors and Data Frames Bonus Activities	54
Appendix D: Highlighted Profiles of Influential Role Models	56

List of Figures

Figure 1: An Example of A Stopping Point	33
Figure 2: Setting a Working Directory Checkpoint	33
Figure 3: Consistent Variable Calling Checkpoint	34
Figure 4: Marie Maynard Daly Profile	35

List of Tables

Table 1: Weintrop’s Computational Thinking	15-16
Table 2: The Phases of Backwards Design	25
Table 3: Utilizing Backwards Design for Development of Overall Curriculum	28-29

1. Introduction

In a world driven by data and digital technology, universal literacy in such topics has become a necessity in the pursuit of equity in the STEM fields. However, various factors including socioeconomic status and demographic representation determine who has access to such technology as well as the resources to become comfortable utilizing it. With computer science (CS) often only being offered as an elective course and at few educational institutions, students lack the opportunity to realize relevant overlaps between STEM and CS. Often lost is the opportunity for students of diverse backgrounds to place themselves in such fields, creating early inequity in opportunity for students without access to a wide range of curricular offerings. One way of combating such inequity early and providing students everywhere access to CS exposure is the interdisciplinary implementation of CS in required science courses. Education researchers emphasize the use of computer science as a means of supporting literacy in data as well as comfort in data interpretation, analysis and visualization (Kjelvik & Schultheis, 2019). With the addition of data familiarity to a traditional STEM education, students may have the opportunity to more comfortably visualize complex STEM topics as well as become more confident contributing to such fields. This project seeks to develop a curriculum that supports universal data literacy as well as combats imposter syndrome in the biological sciences by providing support to teachers and students alike in tackling data related topics within the context of biology.

Data literacy can be defined as “the ability to understand and evaluate the information that can be obtained from data” (Kjelvik & Schultheis, 2019). The ability to understand and interpret datasets as well as make predictions based on existing data serves as a stepping stone to achievement in higher STEM education. This reach is not limited to STEM though, and is not contingent on the implementation of in-depth computer science and data analysis. For students in an interdisciplinary Science, Technology and Society (STS) program, basic digital literacy and course motivation were measured predictors for later achievement in STS (Pala & Başıbüyük, 2021). However, programs such as this are rare, and the United States’ teaching frameworks often neglect to incorporate interdisciplinary lessons to support the development of such skills (Kafai & Proctor, 2021). Furthermore, teachers in pilot studies where similar skills are

implemented in fundamental science courses often report feeling unsupported and uncomfortable teaching curriculum for which they lack training (Yadav, et al., 2016). This lack of support for educators and students alike raises concerns about ensuring that education adjusts alongside the shift to a digital world. Thus, it is important that proposed additions to existing curriculum both integrate what teachers are experienced with as well as provide adequate tools to support their learning of emerging skill sets.

The Bio-CS Bridge project was developed to assist high school teachers in merging computational thinking with standard science curricula. The project, in conjunction with Beecology, a citizen science effort to collect ecological data on pollinator species, utilizes computer science and computational thinking to complement understanding of biology concepts. To date, Bio-CS Bridge has curricula involving Python, Netlogo, Starlogo and HTML/CSS/JavaScript and engages students in creation and use of simulations and other software, hypothesis testing, and data generation, visualization, and analysis.

The goal of this IQP project was to expand upon the existing Bio-CS Bridge curricula by designing a unit meant to guide students in understanding, analyzing and visualizing biological data using the programming language, R. Inspired by research on the effects of student self perception on their performance in the STEM fields, the unit was designed to allow students to overcome the barrier of intimidation when approaching science and programming. This included the goal of combatting a common phenomenon known as “imposter syndrome”, which can be characterized by “chronic self doubt” and “intellectual fraudulence” in individuals who are otherwise qualified in an academic or professional setting or field (Corkingdale, 2008). Furthermore, the curriculum was designed to support students in understanding and formulating complex hypotheses in an accessible way that serves as an alternative to generating data through wet-lab experiments. Research has shown that reasoning skills, particularly those used in STEM, are shown to develop during adolescence (Lawson, et al., 1999), however traditional high school teaching frameworks neglect to expand on these skills. Opportunities to develop such skillsets become increasingly relevant as technology becomes more influential, especially those with interdisciplinary applications like data literacy.

With these ideas in mind, early lessons were designed to support the development of a solid background in the analysis and creation of graphs, as well as understanding how to choose the appropriate graph for different variable interactions. Later, R is introduced as a means of

visualizing data and students are taught syntax as well as how to read and upload a dataset to be analyzed. Eventually, students are guided in making predictions based on Beecology data and work through hypothesis testing using R to validate and visualize data. Together, these skills can serve as a gateway to data literacy and provide exposure to statistics and computer science for students at unique starting points.

2. Background and Literature Review

As the world becomes increasingly dependent on technology, skills such as computer literacy, programming, and an ability to interpret data become urgently relevant in the job market. In recent years, the addition of new technologies to the workforce has caused a widening of the income gap, with salaries increasing only for those in higher level positions that often require advanced degrees (Poliquin, 2021). This gap is predicted to grow as the influence of data and technology increases, making it important that education adjusts with it (Poliquin, 2021). While not every student is destined to become a software engineer, it is important that each individual is provided the opportunity to develop baseline data literacy and exposure to programming. Exposure to and comfort with computer science and statistics at baseline levels may grant students the confidence to participate in and adapt with their changing world. However, many youth in general fail to recognize the relevance of these skill sets when exposed to them in an isolated, academic setting. Such students might lack the confidence or engagement necessary to fuel interest in learning technologically relevant skills due to a variety of factors including but not limited to imposter syndrome, lack of representation in STEM, and disconnect from the real world applications of the field. When students are solely exposed to the “*what*” independently from the “*why*” behind a curriculum, they lack the tools necessary to fuel engagement and drive. Thus, it is important that efforts are made to change the focus of education in computer science and provide equal access to socially, environmentally and scientifically relevant curriculum for students to relate to and feel willingness to participate in.

2.1 Equity and Representation in STEM Education

Studies have shown that students who are encouraged to “do science” are more comfortable, motivated and successful when tackling Science, Technology, Engineering and Mathematics (STEM) problems than those who are told to “be a scientist” (Archer, et.al, 2010). Evidence suggests this disparity is related to equity and identity, especially as a result of the lack of diversity and representation in STEM stereotypes widely ingrained in and accepted by the public. A recent study on the effects of inclusive classroom interventions on combating stereotypical perceptions of what it means to be a “scientist” summarizes this finding, stating that “By including only aged, white men in our textbooks and lectures, we continue to reinforce the

dominant, but incorrect, narrative that has existed for centuries—that white men have been the predominant contributors to new scientific knowledge” (Sheffield, et al., 2021). The absence of representation explored by the study has grave implications, including impacting the trajectory of inclusion in scientific leadership as well as limiting the publishing of contributions by minority groups (Huntington, 2000; Noordenbos, 2002; Sheffield, et al., 2021). Relevant to this project is the impact of these same exclusions in academic settings. Studies have shown that student identity and perception in the STEM fields as well as intellectual identity have significant impacts on their motivation and performance in STEM classrooms. For example, a study by Hong & Lin-Sielger (2011) revealed that teaching students about the personal struggles of famous scientists increased their interest and success in learning physics. The finding suggests that students who are able to relate to and connect with leaders in a topic can better envision themselves succeeding in the same field, making it easier to learn with confidence.

Despite many attempts to improve representation and visibility in the media, performance gaps among groups persist (Sonnenschein & Galindo, 2015). Many of the gaps seen in academic performance are in mathematic based disciplines, which often see lower performance among those from low socioeconomic backgrounds (Sonnenschein & Galindo, 2015). Similar trends have been seen between gender differences as well, with girls often showing lower performance than boys in mathematics testing, however this gap is most significant among students at higher achievement levels (Ellison & Swanson, 2010). The continued observance of such trends makes it even more critical that a curriculum is developed to make hands-on STEM exploration approachable and accessible at all levels. With demographic differences like gender and race/ethnicity showing significant effects on STEM achievement, it is important to increase access to unique means of learning as well as promote a positive and accessible attitude towards tackling new and intimidating topics. One recent study attempted to understand whether or not student gender impacts how they will benefit from the implementation of a program developed to achieve these goals (McLure, et al., 2021). The study looked at both unidisciplinary and multidisciplinary STEM curriculum and involved a total of 413 middle and early high school students in coeducational government and nongovernment schools. Here, the multidisciplinary curriculum was characterized as either science, technology, engineering *and* math while the unidisciplinary curriculum was one of science, technology, engineering, *or* math. The multidisciplinary course included math as an integrated and necessary approach for solving

problems in science, technology, and engineering. Researchers measured the effects of gender on perception of classroom climate and attitudes following the addition of the program, and found that females were most positive about such climates and their own attitudes towards STEM when taking multidisciplinary courses at a government school. The findings suggest that access to a multidisciplinary STEM curriculum in middle and high school could promote more positive self perceptions in science based environments and encourage more female students to pursue related fields at higher levels. This may be because combining science, technology and engineering with math in an overlapping context allows girls to overcome barriers to seeing themselves in the field, as it places the subject in the approachable and familiar context of the world around them.

The implications of this research amplify the need for a shift in state education frameworks and emphasize the benefits of making early education flexible among subjects. While it is important that students are exposed to a wide range of topics independently, it becomes necessary later in education that students learn how to apply and combine knowledge from different disciplines. Furthermore, students will continue to lack the ability to perceive themselves in STEM if they are not exposed to how it impacts their own lives and environments (Williams, et al., 2004). Thus, it was important that the curriculum developed for Bio-CS Bridge allowed students to identify the interactions between the science they learned in the classroom and the world around them through the lens of relevant social and environmental applications.

2.2 Identity and Perception as Predictors of Academic Performance

Research reveals identity and perception to be significantly influential on academic performance of individuals from minority groups, especially in the STEM fields. One study looked at the effects of female identity and an “unwelcoming” campus climate on undergraduate academic burnout and attitude in STEM. The study describes a phenomenon termed “women-scientist identity interference”, which is the internal conflict women face when confronting the idea that their sex is incompatible with male dominated fields, such as science (Jensen & Deemer, 2019). The results indicated that identity interference was connected to increased cynicism, lower academic performance and greater probability of burnout and emotional exhaustion in women studying a scientific or technological discipline (Jensen & Deemer, 2019). Similarly, studies have shown confidence in academic performance to be

disproportionate among genders. One study at Colorado State University found that at admission to STEM majors, women's perception of their qualifications was much weaker than that of their male counterparts, despite having similar performance (MaPhee et al., 2013). Similar burnout cynicisms were observed in other minority groups who choose to pursue STEM, and those with minority status in both ethnicity and socioeconomic status (SES) had even lower confidence than those with just one minority qualifier. However, following the conclusion of a mentoring program designed for these students, their self perceptions became more positive. These results emphasize the importance of positive support and the fostering of confident attitudes in students entering STEM disciplines, especially those who are considered minority in their discipline.

It is evident that underrepresented groups often suffer from inaccurate feelings of inadequacy in scientific and technological disciplines, as is the fact that these feelings of inadequacy have the potential to impact student performance negatively. However, these differences are often ignored and dismissed by STEM participants of non-minority status. A study at University of Wisconsin showed that a group of primarily white male department chairs rated the climate of their departments for women and minorities higher than the women and minorities themselves (Pribbenow et al. unpublished). *Scientific Teaching* by Jo Handelsman, Sarah Miller and Christine Pfund highlights the dangers of this widespread dismissal of inequity in both the sciences and society as a whole. The book serves as a guide for educators in both curricula development and instruction in STEM. It warns that by denying the existence of discrimination and inequities in STEM education, institutions and educators are unable to fully erect and implement the systems that might support students suffering from such differences. Thus, in developing the new Bio-CS Bridge data literacy and R programming unit, it was important to keep in mind students of all backgrounds and abilities.

2.3 Complex Hypothesis Testing and Contextual Barriers in STEM

The scientific method and complex hypothesis testing are foundational topics that students frequently struggle with. These topics share that they require an understanding of problem solving and the ability to apply biology to problems with a variety of contextual applications. Students who struggle with these topics may find themselves unable to relate to material that does not have any real world evidence, or if the real world evidence of the problem

at hand is not made clear to them. The clarity of a problem's real world context may also vary depending on the student's life experience and background exposure, a problem known as culture bias. An example of culture bias is often seen in standardized testing, where minority students fall to a disadvantage, since the context of the problems presented is grounded in "general knowledge" shared disproportionately by the majority (Kim & Zabelina, 2015). The clarity of real world problems may also be limited by the complexity of the topic at hand, and how concrete the variables involved are, as this may impact how easily a student can visualize the literal agents in the problem. A study by Lawson, et al. (1999) explores this, and demonstrates the existence of two different means of hypothesis rationalization. The first means of rationalization involves hypotheses with concrete variables, ie, those that can easily be visualized as objects, and the second involves hypotheses pertaining to abstract variables such as those in the subjects of chemistry and physics. The study showed that students struggled more with understanding and formulating hypotheses involving variables that could not be easily visualized due to factors limiting students from perceiving them, such as size. In summary, student success was determined by the "abstractness" of the given hypotheses; however, the effects of student prior experience on their perception of "abstractness" was not discussed. While the study briefly addresses the addition of technology, such as the microscope allowing for ease of such limitations, it does not explore the impact computer science could have as a visualization tool to aid in STEM classes.

2.3.1 Computer Science As a Bridge to Accessibility

One potential way to confront the barriers that make it difficult for students to contextualize complex technical and scientific concepts may be the overlap of both STEM and CS in the classroom. Students often struggle with realizing connections between the world around them and both STEM and CS. The key to confronting the barriers to both fields lies in combining them, opening the door to maximum learning (Braun & Huwer, 2022). While STEM provides real world problems for students to consider, CS provides the tools to address such problems, and has been shown to assist students in thinking computationally when integrated into STEM classes (Yang et al., 2021). In the absence of STEM, CS may appear dry and lacking in meaning to students who struggle to see its potential, and vice versa. Combining the

disciplines may offer a bridge between student and subject, as it may allow students to visualize concepts which once appeared distant, unimportant and disengaging. Similarly, data literacy and visualization can help apply numbers to those complex subjects, allowing students to visualize relationships through graphs. While programming itself often appears intimidating to students, introducing it at an early age in conjunction with its real world applications can build gradual comfort in students. This can be achieved not solely through code, but also through the introduction of software and tools made by code to solve a real world problem in science. An example of this is Foldit, a game designed to teach anyone the biochemistry of protein folding. Foldit calls on its users to discover new folding patterns through game-type challenges, and has been effective in solving relevant problems in protein biochemistry (Kleffner et al., 2017). Another example of this can be seen in Nova Labs, now marketed as Eterna, an interactive crowdsourcing initiative by PBS that allows anyone to discover new RNA molecules. The site allows users to model unique RNA folding patterns in the form of a game, and sends the data back to researchers to be used in categorization of potential molecules (Kleffner et al., 2017). As the tool is user friendly and designed to be easily learned by anyone, it has allowed students as young as elementary school to identify RNA molecules with serious applications in medicine. Students who have contributed to the discovery of such molecules have even been named in papers about their findings. By providing an easily accessible opportunity for people of all ages and backgrounds to contribute to research and be recognized for doing so, Nova has utilized computer science in a way that aids in bridging the gap between science and the person.

FoldIt and PBS' Nova Labs are not the only attempts to use technology to make science more accessible to the public. Other non-game based attempts, particularly in ecology, seek to crowdsource more directly, asking users to collect data about attributes of their environments to help solve real world problems in ecology. One such effort is the Beecology and Bio-CS Bridge collaboration, which utilizes data collected by citizen science web app users to teach students data analysis and visualization as well as hypothesis testing.

2.3.2 The Bio-CS Bridge Initiative

By shifting the range of education from covering rigid frameworks to allowing for flexibility in interdisciplinary instruction, schools can support student adaptation to a changing

world. Bio-CS Bridge, a National Science Foundation funded project started in 2015, attempts to do just this through the development, testing, and implementation of interdisciplinary computer science and biology curriculum at the high school level. The curriculum is inspired by the Beecology project, a citizen science effort to approach the issue of pollinator decline. Using Beecology as its contextual connection to the real world, Bio-CS Bridge integrates computational and biological approaches to solve real-world problems, often by means of simulation, data analysis and visualization. While rooted in biology, the project exposes students to a wide variety of CS material that allows students to apply their learning using real world methods of problem solving. Currently, there exists Bio-CS Bridge material to support learning in Python, HTML/CSS/JavaScript, Netlogo and Starlogo, as they pertain to biology and Beecology. These lessons have collectively worked to promote the development of computational thinking in students from participating institutions. Existing curriculum was developed by and in consultation with a team of biology and computer science educators from Massachusetts and consists of thematically related biology, computer science, and data visualization units which can be used in conjunction with one another depending on student and classroom needs.

2.3.3 Beecology

The Beecology project was designed in an effort to better understand observed declines in pollinator species over recent years through citizen observation and data collection (Beecology Project). The project calls on “citizen scientists” to crowdsource and log information on native pollinator species through the use of a web app, which is accessible on most common devices. In conjunction with Bio-CS Bridge, the Beecology team also works to communicate trends in pollinator-plant interaction data through visualization tools available to the public and to schools as educational material. Since Bio-CS Bridge seeks to expose students to real world applications of computer science, it uses these visualization tools to educate about realistic and urgent applications in ecology. This collaboration may help inspire and push for the shift to more interdisciplinary and engaging educational pedagogy in STEM, as well as promote data literacy among students and educators alike.

2.3.4 Defining Computational Thinking

Since state standards in education rarely include guidance on interdisciplinary computer science teaching, discourse exists surrounding what it means to think computationally. For the creators of Bio-CS Bridge, this conversation was integral to developing concrete and effective goals for their curriculum. Bio-CS Bridge promotes the idea that computational thinking can be achieved by “engag[ing] students and teachers in scientific practices using biological data that they collect themselves, and computational tools that they design and implement, to address a complex real-world problem.” (Bio-CS Bridge). Previous curriculum designed for the project is influenced by the Mathematics and Science Practices Taxonomy of Weintrop et al. (2016), which digests computational thinking into four distinct foci (Table 1). The first focus is on data, and involves data collection, creation, manipulation, analysis and visualization. Second is understanding how to design, construct, assess, and use modeling and simulations to make predictions and test hypotheses in a virtual environment. Next is general problem solving and developing independence in the exploration of different problem solving approaches, especially while troubleshooting. Finally, computational thinking involves what is known as “systems thinking”. Systems thinking constitutes being capable of understanding relationships, leveled thinking, and communicating information about relationships in a complex system (Weintrop, et al., 2016). Shown in Table 1 below is an outline of Weintrop’s Computational Thinking.

Table 1: Weintrop’s Computational Thinking

Data Practices	Modeling and Simulation Practices	Computational Problem Solving Practices	Systems Thinking Practices
Collecting Data	Using Computational Models to Understand a Concept	Preparing Problems for Computational Solutions	Investigating a Complex System as a Whole
Creating Data	Using Computational Models to Find and	Programming	Understanding the Relationship within a

	Test Solutions		System
Manipulating Data	Assessing Computational Models	Assessing Different Approaches/Solutions to a Problem	Thinking in Levels
Analyzing Data	Designing Computational Models	Developing Modular Computational Solutions	Communicating Information about a System
Visualizing Data	Constructing Computational Models	Creating Computational Abstractions	Defining Systems and Managing Complexity
		Troubleshooting and Debugging	

Adapted from Weintrop, et al., 2016.

While people often associate computational thinking with the sciences and technological disciplines, its reach is not restricted to such fields. Since a major component of computational thinking involves fine critical thinking skills, many argue that computational thinking can also be used in disciplines outside of STEM to help create a more equitable world and prevent marginalization (Kafai & Proctor, 2021). By learning to think about the world in both an empathetic and computational, critical manner, students may learn at an early age to solve its problems objectively in a way that leaves behind bias and advances towards equity. Thus, it is important that state frameworks for curriculum adjust to accommodate interdisciplinary learning that might promote engagement in solving real problems.

2.4 Barriers to Interdisciplinary STEM Education and Making Opportunities for Computational Thinking Accessible

The implementation of computational thinking in interdisciplinary classrooms might assist in bridging gaps in both self perception and technological accessibility in STEM. Furthermore, research has shown that the addition of introductory computer science and

programming in K-12 and early undergraduate education may help enhance critical thinking skills and problem solving abilities, especially when the programming taught is object oriented (Dalton & Goodrum, 1991; Norris, et al., 1992). However, previous attempts at implementing such programs have revealed shortcomings in both student and teacher preparedness (Shernoff et al., 2017; Herro & Quigley, 2016). STEM teaching frameworks by state are often strict and leave little room for the implementation of new means of learning, especially in AP courses (Kafai & Proctor, 2021). Furthermore, teachers often lack the time and resources to become acclimated to programming before teaching it. As a result, very few United States' school systems offer computer science to its students, with fewer offering it in conjunction with applicable STEM courses. Between 2018 and 2021, the percentage of US schools offering computer science jumped from 35% to 51% (*State of Computer Science Education Accelerating Action through Advocacy Advocacy Coalition*, 2021). The same report reveals that in 2022, \$65 million combined was reserved for computer science education in the U.S, more than years past. However, access to this funding isn't always implemented in ways that are effective or equitable.

Unequal access to opportunities in computer science is not simply the result of inadequate teacher support. The problem stems from a more general rigidness and lack of resources in U.S education, particularly the lack of necessary technology to accommodate all students. In 2020, the rise of the COVID-19 pandemic brought awareness of this problem, as education underwent a sudden shift to virtual settings. During this time, the U.S. Census Bureau conducted a Weekly Household Pulse Survey, which provided insight into how the pandemic impacted education for students of differing household incomes. The data revealed that children from low income households from poor states were most likely to be at technological disadvantage in the shift to online learning, and that Black students were most likely to lack access to learning devices and internet (U.S. Census Bureau). In households making less than \$25,000 a year, 12.2% of those surveyed reported rarely or never having access to a device for student learning and 9.8% reported lacking reliable internet. In Detroit, 1 in 5 households that identified as Black lacked access to such resources for student education, and in Los Angeles, while only 0.1% of White respondents reported lack of access, the percentage rose to 13.2% for Black respondents. In a similar study by EDWeek Research Center, only 59% of the teachers surveyed stated that they had enough devices to provide each student access to virtual options.

This inequity seen in access to personal learning technology contributes to the demographic gaps observed in STEM fields such as computer science. From an early age, access to computers is limited, making the transition to a technology based curriculum a difficult one. This can result in the development of psychological barriers in those who choose to pursue computer science at the college level, where students receive less one-on-one support, yet are held to equal standards that typically assume previous technological experience. One of the largest barriers, and not uncoincidentally, a phenomenon brought to light with the rise of computer science, is “imposter syndrome”. Despite proven qualifications, feelings of inadequacy persist in individuals with imposter syndrome, and can evolve to further barriers to success, as it is often seen in conjunction with anxiety disorders and depression (Bravata, et al., 2020). A possible explanation for imposter syndrome is the lack of representation or support in the fields of STEM, as it is particularly evident in women and minorities (Ahmed, et al., 2020). This hypothesis is consistent with aforementioned research, which suggests lack of representation and separation of identity to be limiting factors in student STEM success, as students often lack the ability to picture themselves in the position of a “scientist” (Archer, et al., 2010). Such a separation between student and subject could be further impacted by the inequitable access to technology in early education classrooms. Because unequal access to technology and computer literacy has such grave implications on minority perception in STEM, it is critical that schools provide an environment where students are exposed to the skills necessary to develop confidence in such fields. It is possible that the addition of such a curriculum to state frameworks might supplement the push to funding equal access to necessary technologies for all students.

3. Methodology

In order to ensure both student and teacher needs were met in the development of the curriculum, scientific teaching was researched extensively and tested theories in curriculum design were implemented. The book, *Scientific Teaching*, by Jo Handelsman, Sarah Miller and Christine Pfund was used to support backwards design, constructing a teachable unit, and ensuring recognition of diversity in teaching. Furthermore, local teachers were consulted during planning and provided insight into 1) which programming platform would be both convenient and support student privacy, 2) how to best design a curriculum appropriate for varying achievement levels and 3) how to make content appropriate and inclusive of different skill sets.

The overarching goal for the curriculum was to provide an engaging and impactful exposure to the broad applications of data science through the lens of Beecology. Targeted impacts were as follows:

- 1) Broaden students' understanding of hypothesis testing through the investigation of biological questions, accessible through data analysis using R
- 2) Break down the mental barrier that prevents many students from seeing themselves as scientists capable of integrating CS and data analysis, as well as remove intimidation from approaching code
- 3) Provide understanding of the expression of variables through plotting, graphing, and descriptive statistics
- 4) Provide the resources for students to become data literate and to begin to build programming and data analysis skills

3.1 Choosing R and Initially, Replit

Significant time and consideration were used in the process of deciding which programming language and integrated development environment (IDE) to use throughout the curriculum. While the team of educators and college students at Bio-CS Bridge previously developed curricula in data analysis, they were lacking a unit that tied together programming, statistics and data visualization in a way that was contextually relevant to a wide range of students. Furthermore, direct consultation with the same educators who contributed to the development of previous Bio-CS Bridge curriculum revealed further needs for the new unit. First, teachers expressed frustration about the various barriers that prevent the effective implementation of new interdisciplinary curricula in traditional K-12 academic settings. For one, teacher pay rarely supports the time it takes to learn emerging skills expected for educators to teach, like programming. Even with pay aside, teachers simply do not have enough time in the day to take on additional work, like online courses in programming, as grading often also occurs at home. Since teachers are educated in varying disciplines, it is not reasonable to expect them to have the background to, by default, be effective at executing the instruction of foreign skills like code. Thus, it was important that the language chosen was easy to learn from scratch both prior

to teaching and in conjunction with students, and that existing resources for independent exploration were financially accessible.

Next, it was important to consider choosing a language that could be explained to high school students and was easily adaptable for a wide range of backgrounds. Ultimately, it was important that students at unique starting points benefited from the language and platform chosen equitably. Thus, it was decided that the language must be open sourced and have accessible IDEs available for web-based use, as students are not guaranteed rights to download depending on the school district and existing privacy agreements.

Following consideration of the above points, R was chosen to be used as the language learned in the new Bio-CS Bridge unit. R is a programming language designed for computing in statistics and data visualization (The R Foundation). With careful scaffolding, R can be introduced smoothly and can be adjusted to fit varying levels of complexity. Furthermore, R is available through online platforms and is open sourced, making it a cost effective and accessible option for students and school systems.

R is commonly used in bioinformatics and most fields that require data analysis, however most students do not gain experience with it until college or graduate school. By the time a student is introduced, they may have already subconsciously identified themselves as someone incapable of working with code. This mental roadblock is what often induces imposter syndrome, preventing students from making the most of their careers and reaching their highest professional potential (Corkingdale, 2008). Furthermore, this lack of exposure is what leads to the belief that only certain people can be programmers or “scientists”. By implementing R as an alternative and accessible means of hypothesis testing, the curriculum may allow students to envision themselves in STEM roles that once appeared unachievable.

Similar considerations were taken when choosing the IDE, Replit. Replit is a free, web based IDE that allows users to program and collaborate in 50+ programming languages. Before choosing Replit, various other platforms were considered such as R Studio Cloud and Jupyter, however consultation with active teachers involved in Bio-CS Bridge revealed these options to be less than favorable. First, both platforms would require the signing of FERPA agreements to ensure protection of student privacy. The school district in question was already in FERPA agreement with Replit and teachers expressed both familiarity with and positive opinions of it. Second, Jupyter would have required use of Github to save and share code, which teachers found

to have a time consuming learning curve. In addition, Replit was already used with Bio-CS Bridge's Python unit. To ensure both students and educators comfort with the platform, a Replit User Guide was designed and a master document of 'cheat sheets' were added to provide extra resources for students and teachers.

3.2 Adjustment to R Studio

Late into the project, a shift from the use of Replit to R Studio was made. Unfortunately, many challenges arose in implementing use of Replit, however these issues arose later into the curriculum, making the shift a last minute effort to adjust. Regardless of the set back, due to the benefits of R Studio and support provided in the lessons, teachers can feel confident and supported with the adjustment.

Perhaps the most detrimental problem in implementing the use of Replit was the inability of the IDE to quickly and efficiently both install and make use of required libraries and packages, such as *ggplot2*. In R, packages are pre-programmed functions and code stored in a library for ease of use. Most of R's functionality lies in its packages, and many packages contain sample data, making them accessible and user friendly. Unfortunately, late into the lessons, it was discovered that Replit lacks the processing power necessary to install packages manually. After some troubleshooting, it was discovered that Replit does in fact harbor a feature designed for installing packages. However, it became evident after multiple attempts that the fancy UI to replace a single line of installation code only served as a distraction from Replit's lack of functionality. The packages still were unable to load efficiently, if at all. On one attempt, *ggplot2* did load, however Replit failed to recognize its existence when called on. Following the errors with package installation, the decision was made to base the majority of the curriculum around base R plotting, since this would not require packages and could allow for the use of Replit. However, Replit failed in its ability to provide friendly graphics, as the base plots generated were returned in a PDF that was both difficult to locate and read after each run.

While these problems greatly reduced Replit's ability to support use of R beyond the barebone fundamentals, a larger detriment was found in the IDE's lack of community and resulting lack of resources. While Replit is well known by educators for teaching Python and some other common languages, it could be argued that this success lies in the community

surrounding these areas. With the majority of R users being in higher education or the workforce, few use online IDEs like Replit. As a result, it was extremely difficult to find troubleshooting resources online, making the R learning curve a lot steeper. While it is true that the curriculum would provide the necessary steps for students to implement the material, unanticipated errors occur all the time and it is important that students are capable of navigating such situations semi-independently. Additionally, with few resources online available for support using Replit for R, choosing to use the IDE would have left teachers alone in both learning the software themselves and supporting students. Thus, it was decided that while Replit is fantastic for classroom use in languages that it already supports, it is not the ideal solution for those lacking an established community.

R Studio was chosen as the alternative programming environment. R Studio is a development environment for R with availability as both downloadable software and as a free online cloud environment. Designed with functionality in mind, R Studio is used commonly in education, research, and industry. This is heavily attributed to its ability to load a wide range of packages, which allows for R implementation for a variety of applications. While R Studio is less directed towards youth in its design and graphics than Replit, it is more accurately representative of software used by the field and can be introduced appropriately provided students are granted adequate support. The decision to switch to R Studio as the chosen IDE for the Bio-CS Bridge curriculum called for increased support for both teacher and student, which was ensured in the design of the lessons and pace of introduction to more advanced concepts in R. Despite this setback, switching to R Studio resulted in a stronger curriculum overall and will provide students with necessary exposure to common tools in data science and STEM. Furthermore, this decision will allow students to overcome the intimidation speed bump often accompanied by learning new technical skills early on, setting them up for more confident learning later in their academic and professional careers.

3.3 Creating Approachable Datasets

While Beecology allows users to filter for variables of interest and download the data for those variables in a csv file, it made more sense to provide students smaller, cleaned datasets that would be easy to read and use. The datasets were created to ensure that students had just enough

information necessary to complete the assignments without overwhelming them with information. This would allow students to become comfortable reading a .csv file and understand how its data translates to R. An example of a dataset can be found in Appendix A.

3.4 Supporting Both Teacher and Student

One of the greatest limitations when utilizing computer science funding effectively is lack of support for teachers in implementing foreign curriculum (Sentance & Csizmadia, 2017). Teachers often lack the background necessary to teach computer science and are expected to learn such skills outside of paid work time. Because of this, the curriculum for this project was designed with the needs of both teachers and students in mind. In developing the new Bio-CS Bridge unit, it was critical that materials were designed to support educators so that they may be prepared to come to class confident, with positive attitudes towards the material to be taught. As new skills, especially in technology, can be intimidating, it is important that teachers approach instruction with both evident positivity and empathy. By providing the materials to wholly support teachers in learning new data skills, Bio-CS bridge may help such educators serve as valuable resources to their students. Thus, student worksheets were designed in an easy to follow, self-guided manner with supplemental teacher notes at specified checkpoints for each worksheet. The combined materials allow for a flexible instruction style which grants students the independence to work at unique paces without demanding excessive guidance from the teacher. This flexibility allows teachers to serve more students at unique levels while feeling balanced and confident in the material themselves. In its formatting, the curriculum allows each student to work at a pace appropriate and beneficial to them alongside their peers, under the same teacher.

Curriculum was designed with student engagement in mind. Studies have shown that even when computer science is offered in schools, students are still reluctant to take it as it is treated as an elective and not a graduation requirement (Kafai & Proctor, 2021). This is true even at the AP level. While the number of students taking computer science in high school has increased in recent years, reach remains limited. With barriers such as intimidation, insufficient teacher support and lack of accessible technology in place, access remains inequitable. Thus, the goal of this project was to design a curriculum that garners student interest while simultaneously

making them feel supported and comfortable exploring new technological skills. Doing so was achieved through the following efforts.

3.5 Ensuring Diversity in Curriculum

Scientific Teaching discusses the importance of recognizing, appreciating and adjusting for student differences, as all are components of cultivating a successful and universally beneficial classroom. As discussed earlier, Kafai and Proctor (2021) too recognize this in their conversation on computational thinking. They express their belief that computational thinking will expand to involve social, cultural and political dimensions, and that addressing the inequities in the world will require leaders and society to think computationally about such problems (Kafai & Proctor, 2021). In saying this, Kafai and Proctor reveal another aspect of why interdisciplinary learning is so relevant. While it is important to teach some core STEM skills independently, it is also important to guide students in recognizing how their courses overlap and can impact one another. Thus, in developing the new Bio-CS Bridge unit, it was integral that the content of lessons and datasets were biologically relevant and representative of real world problems, to reveal to students that data analysis is a useful tool in solving such problems. For example, one dataset allowed students to look at the relationship between bee tongue length and species, to observe patterns in the trait as well as its relevance to the functional biodiversity of the pollinators. Another prompted students to look at the relationship between Iris petal length and width, again using real world data from Beecology. The contextually diverse nature of allowing Biology and computer science to overlap may allow students of differing interests and backgrounds to relate to the curriculum and stay engaged throughout. Furthermore, as the curriculum allows for reflection on data validity and bias in general, students are prompted to make connections to examples outside of Beecology. With the potential for students to make connections to socially and politically relevant topics through this reflection, students are guided in making connections to the world around them and understanding *why* what they are learning is important.

In addition to diverse contextual examples, the curriculum was developed with inclusivity in mind. Contextual information was explained to prevent bias. This was important, as studies have shown cultural background to affect understanding of and performance on standardized

testing, which are often unintentionally biased towards non minority populations (Dobrescu, et al. 2021). Additionally, profiles of diverse biologists, biochemists and data scientists were added throughout the lessons to allow students to relate more closely to the material and feel represented.

3.6 Backwards Design

Backwards design is the practice of designing academic curriculum based on predetermined goals for learning (Wiggins & McTighe, 2011). As stated in its name, backwards design works backwards to ensure that the content of curriculum and supporting materials all supplement specific objectives. This design method allows educators to maintain direction in their lessons as well as establish clear communication with students of expectations from both parties upfront. The four steps involved in backwards design are as follows.

Table 2: The Phases of Backwards Design

1) Identify Desired Outcomes	2) Assessment	3) Activities	4) Alignment
Develop appropriate goals for learning, aligning with Weintrop’s Computational Thinking	Decide how learning will be measured	Develop material to build student skill sets	Assess if activities and assessments align with goals. If not, go back and revise the material to match goals

Adapted from Wiggins & McTighe, 2011.

Backwards design was implemented in the creation of each lesson, as it was important that goals were in place to maintain balance between Biology and CS in the curricula. A fear was that the interdisciplinary nature of the curriculum would leave room for confusion and feelings of inadequate direction among students. This was prevented by ensuring that goals were communicated from the start and that each step in the tutorials was explained, along with

mention of connections between disciplines when necessary. Checkpoints were included following integral technical steps, when it might be important for students to pause and reach out for help. The lessons were revised multiple times each to ensure these goals were effectively achieved and to prevent disproportionate skew to one area of the combined disciplines. While frequent revision resulted in less content, the depth of the existing tutorials allows for students to learn at a comfortable pace and learn to both troubleshoot and ask for help in a comfortable, low risk setting.

4. Findings and Results

Despite various setbacks, this IQP allowed for the successful development of an integrative R curriculum to supplement computational thinking alongside the existing Bio-CS Bridge material. The curriculum was developed using Backwards Design methods with Weintrop's Computational Thinking in mind. These resources supplemented the pursuit of two main goals. The first goal was to increase data literacy among high school students while exposing them to the range of possible applications in biology. The second and perhaps most fundamental goal was to ensure that the curriculum was structured in a friendly, informative and approachable manner that was conducive of combatting imposter syndrome among students of all backgrounds. The final introductory unit consists of of six thoughtfully paced lessons, titled as follows:

1. Introduction to Programming For Biology
2. Getting Started with RStudio and R
3. Vectors and Data Frames
4. Graphing with Beecology
5. Graphing with Beecology Part Two
6. Considering Bias in Citizen Science Data

Together, these lessons work to provide support for students of diverse starting points, and are constructed in a manner that allows for both independent and teacher supported learning. The lessons and associated code are included in their entirety in a zipped file associated with this document, and an example of a lesson can be found in Appendix B.

4.1 Implementing Backwards Design For Overall Curriculum

The Bio-CS Bridge R unit was developed using a method of Backwards Design, as outlined in Table 2. While the final curriculum is limited to six lessons total, these lessons allow for a carefully crafted introduction to R, statistics and data visualization. The lessons are structured to build off one another, so that students have the opportunity to become independent in their learning as well as confident in implementing new skills. Much revision was necessary to ensure the six lessons supplemented one another without leaving gaps. Some tutorials were rewritten multiple times resulting in drastic but necessary changes. While the Backwards Design process was time consuming, it was found to be necessary in the development of a truly thorough curriculum. Below is the general outline of goals and Backwards Design methods used in revision of the curriculum at both lesson level points and as a whole.

Table 3: Utilizing Backwards Design for Development of Overall Curriculum

Identify Desired Outcomes	Assessment	Activities	Alignment
<p>Students will be able to think in levels and demonstrate the ability to make hypotheses</p> <p>Students will be able to problem solve common software issues and become comfortable navigating computational tools such as R Studio</p> <p>Students will understand mean, median, and mode as well as other</p>	<p>Learning will be measured through a mix of written questions and tutorial based activities. Students will not be assessed through test based measures, but will instead log their answers so that teachers can provide constructive feedback during self paced classroom work sessions.</p>	<p>Activities will include a mixture of tutorial based lessons which will promote skill development in R and data visualization as well as thought and reflection. Six lessons total will be created to introduce students to R and have them reflect on the importance of data throughout.</p>	<p>The organization and content of the lessons will be revised continuously. Lessons will be adapted for both functional reasons as well as alignment efforts. Weekly meetings with an advisor will allow for brainstorming of solutions and discussion of each lesson's efficacy as well as the curriculum's trajectory.</p>

<p>summary statistics</p> <p>Students will develop an intuition of when to use different types of data visualizations</p> <p>Students will develop comfort with R and programming</p> <p>Students will leave the unit with an understanding of how programming can be used in Biology</p> <p>Students will leave having combatted feelings of imposter syndrome and see themselves as capable “scientists”</p>			
--	--	--	--

Adapted from Wiggins & McTighe, 2011.

4.1.1 Identifying Desired Outcomes

Early on, desired outcomes were developed for the intended trajectory of the curriculum. The overall goal was simple: that students see themselves as capable of participating in science and technology as a result of the unit. With imposter syndrome rampant and affecting minority groups disproportionately (Ahmed, et al., 2020), it was important that the curriculum developed would cater to students of all backgrounds.

At the curriculum level, it was desired that students develop data literacy as a result of completing the unit. Early background research established that it is becoming increasingly important in most fields for professionals to hold some degree of data literacy. By allowing the opportunity for Bio-CS Bridge students to manipulate and visualize data themselves, it was hoped that students would overcome any barriers in developing confidence working with data early. Specifically, this involved designing a curriculum that would allow students to learn when

it was appropriate to use different types of statistical measurements in describing data as well as when to use different methods of visualization. Students were to be given practice in hypothesis generation and testing using real world data from the Beecology project. Furthermore, they were to be actively involved in inquiry at the intersection between data science and biology with the goal of seeing *themselves* as active citizen scientists as a result.

4.1.2 Designing Effective Assessments and Activities

The final curriculum ended up containing six lessons which collectively work towards the goal of testing hypotheses about Beecology data in R. The first lesson provides background for the unit and is titled “Introduction to Programming for Biology”. This lesson has students consider why data and programming may be used in biology as well as introducing the concept of citizen science. While the first lesson is concise and contains no hard skill development, it sets students up for success by allowing for clear communication of goals early on. This way, students are not confused or intimidated when they are presented with a more technically heavy software section in lesson 2.

Lesson 2 is titled “Getting Started with R Studio and R”. This lesson is a brief tutorial which runs through downloading R Studio and setting up the workspace. Additionally, it introduces some basic programming concepts to allow students familiarity with running code in the environment.

Lesson 3 is titled “Vectors and Data Frames” and expands more into the skills required to successfully navigate R. It explores the creation, manipulation and editing of vectors, the most basic data structure in R. Additionally, the lesson introduces students to the concept of a data frame. First, students are introduced to Iris, a built-in data frame in R which contains information on the petal length and width of different species of the Iris flower. The lesson teaches students how to access the data in the Iris data frame. Since the data frame contains numerical data, it is an ideal starting point to introduce summary statistics. Leveraging this, the lesson prompts students to consider the importance of such statistics in describing data. Students then are guided in creating their own data frame using data manually sourced from the Beecology website. The lesson includes a bonus activity for those who progress quickly, which allows students to plot their data frames and test informal hypotheses about the variables within them.

The next lesson is titled “Graphing with Beecology”, and in conjunction with its follow up, “Graphing with Beecology: Part Two”, serves as the heart of the unit. This is the first lesson which introduces plotting outside of a bonus activity and is the first to ask for the formal generation of hypotheses. Additionally, the lesson is where students truly begin to see for themselves how using R may assist them in answering biological questions about real data. Prior to the lesson, students are provided a dataset which has been crafted from the larger Beecology dataset to provide only the variables necessary to complete the lesson. This decision was made to ensure that students are not overwhelmed when presented with an Excel file, as many likely will have not been exposed to one prior. The dataset titled, *beeData*, contains the variables for time, year, month, species, behavior, gender, and tongue length. The time variable includes the exact date and time the data was collected, as this format was used with mutating data and plotting with *ggplot*. While month and year could have been extracted from the time variable, they were added separately to make graphing clearer in introductory lessons. Tongue length was selected as a bee characteristic to be analyzed, as it allows students to make hypotheses and observations about the functional biodiversity observed across species. Using the *beeData* dataset, students learn skills including loading a dataset into R, extracting information from said dataset, manipulating data, retrieving and interpreting statistics, plotting data, and generating and testing hypotheses. These skills are learned within the context of real data to determine how functionally diverse an ecosystem is, considering questions involving the distribution of functional traits across species.

Next in the curriculum is lesson 5, which is titled “Graphing with Beecology: Part Two”. This follows the trajectory of “Graphing with Beecology”, and prompts students to expand on the skills that they learned in the previous lesson, but with less guidance. The lesson contains frequent checkpoints which serve to review and reinforce skills. This format allows students to become comfortable with independent problem solving as well as tackling tasks on their own. In addition to having students create new plots, the lesson teaches students to customize plots to be easier to read by moving their elements as well as changing the color scheme. Next, the lesson introduces the concept of installing packages. Here, the packages *ggplot2* and *dplyr* are used to manipulate a new dataset and plot a time series showing the occurrence of bee species over a season. Using these packages, students learn to plot the three tongue lengths over the bee season in a step by step tutorial, allowing them to test their previous hypotheses about functional

diversity across bee species. The second dataset contains only three bee species, which were intentionally chosen for their different tongue lengths. The goal in doing so was to allow students to compare the graphs and consider why the patterns shown in the two are identical. They are guided in realizing that bee species correlates to bee tongue length, and that certain tongue lengths (thus different species) are observed at different periods in the season. Students are also prompted to consider why they observe such a pattern, as well as whether or not they would observe the same pattern when using the larger data set. As a bonus, students are asked to generate a similar time series using the larger dataset.

The sixth and final lesson in the Bio-CS Bridge R unit is titled “Considering Bias in Citizen Science Data”. This lesson serves as a wrap up to the introductory R curriculum and puts the unit in context of data science as a whole. Students are prompted to reflect on the hypotheses they tested and consider what factors might have affected the validity of their results. The goal of this lesson is to have students recognize some common flaws associated with citizen science. Furthermore, the lesson was created to make students aware of the importance of being able to consider what bias might exist in any data they may analyze in the future.

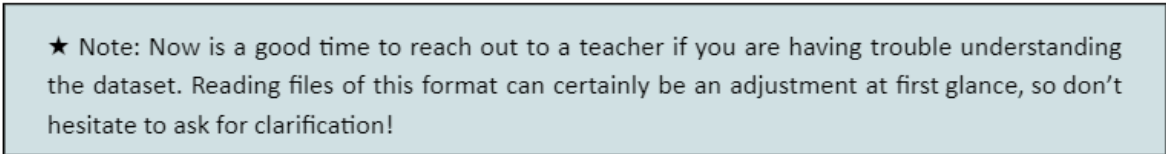
4.2 Developing Activities to Support Independent Problem Solving

When developing the structure of the unit, it was decided that assessments would be non intimidating and support skill building as opposed to formal examinations. Thus, the assessment and activity portion of Backwards Design molded together to form the six lessons. These lessons build off one another to provide a solid foundation in R as a language as well as an understanding of statistics and plotting. Consultation with teachers affiliated with Bio-CS Bridge earlier in the project confirmed that students in a classroom often start at different comfort and exposure levels in statistics and math. While one teacher requested that the unit include AP Statistics related curriculum, other educators suggested that this might be too intimidating for students who have not yet had such exposure. Thus, it was important that the curriculum materials were friendly at all levels and allowed for students to work at a pace comfortable to the individual. Bonus activities were included where applicable for students who progress more quickly, while the lessons start at a basic level to support those who have limited background. Examples of these bonus activities can be seen in Appendix C. It was assumed that all

participating students have no previous, or at most limited, R exposure, so the language was explained in depth at the very basic level with clear definitions and pauses where necessary.

In addition to providing explanation where necessary, cautious efforts were made to ensure that the lessons provided physical stopping points for student reflection. As the curriculum was developed by a student just four years out of high school, these stopping points were intuitive. However, teachers are encouraged to review the lessons prior to dissemination regardless in case they feel adjustment is necessary to meet the individual needs of each classroom. An example of a stopping point can be seen below in Figure 1.

Figure 1: An Example of A Stopping Point



★ Note: Now is a good time to reach out to a teacher if you are having trouble understanding the dataset. Reading files of this format can certainly be an adjustment at first glance, so don't hesitate to ask for clarification!

As seen in the example shown in Figure 1, notes of this nature were consistently worded in a friendly manner that reminded students of the importance of being able to reach out for help. Oftentimes, independent work time can result in students feeling lost or unable to speak up. Checkpoints such as the one shown above were regularly added to avoid this and to promote teacher involvement through independent work time.

Additionally, as seen in the example above, explanations were provided when it was necessary for students to understand when a skill might be an important one to learn confidently. Figures 2 and 3 below show additional examples of this. Figure 2 shows a checkpoint which follows steps to set a working directory in R. Since failing to set a working directory correctly often results in common errors, it was important to make students aware of the importance of completing these steps carefully early on. Such mindfulness can also be seen in Figure 3, which shows a note reminding students to be consistent in their use of variable names in their code. Since many students enter the unit not having programmed before, these reminders may help to avoid common errors.

Figure 2: Setting a Working Directory Checkpoint

★ Now is a good time to pause and ask your teacher for help if you are having trouble setting the working directory. This is a crucial step in making sure your code runs smoothly later!

Figure 3: Consistent Variable Calling Checkpoint

★ Note: Be sure to check that you use the variable name you used for species. Errors commonly occur when we are not consistent with variable names, so this is always important to be mindful of!

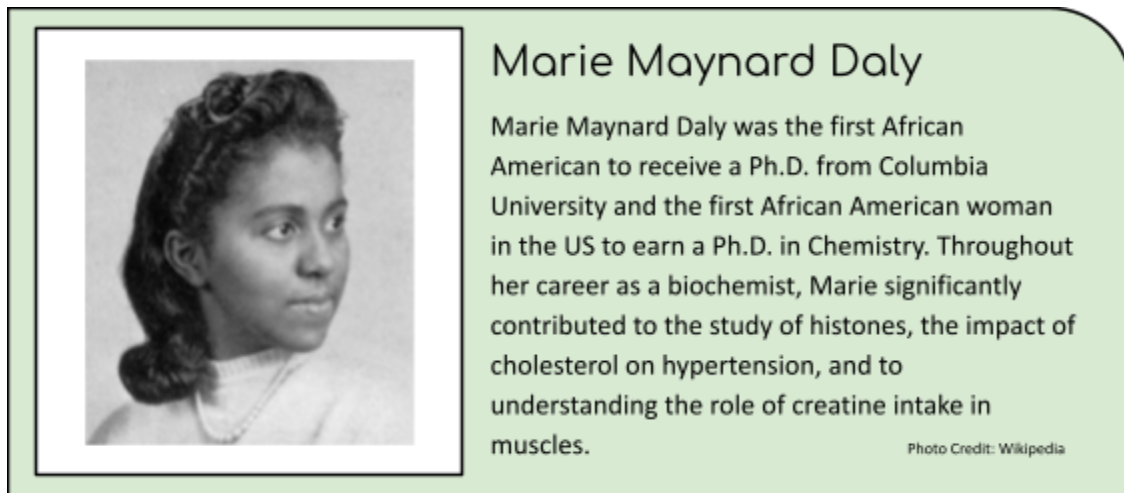
4.3 Ensuring Diversity in Curriculum and Combatting Imposter Syndrome

Since imposter syndrome disproportionately affects women and minorities, it was a goal throughout the development of the curriculum to ensure that the lessons catered to students of all backgrounds (Ahmed, et al., 2020). This process involved making sure that the examples in the lessons were all supplemented with contextually relevant explanations when necessary. This was done in an attempt to prevent culture bias, as studies have shown that bias in curriculum and assessment material can create unfair grounds for its completion by students of different backgrounds (Dobrescu, et al. 2021). Furthermore, all activities were phrased in an “Imagine you...” format, where the students themselves are called to envision themselves as data scientists whose job is to test a hypothesis using Beecology data. This practice supported the goal of allowing students not only to learn science but also to “do science”, allowing them to build self confidence and a sense of belonging in the field.

In addition to phrasing exercises in a way that allowed students to put themselves in the position of a “scientist”, each lesson includes a profile of a biologist, data scientist, or biochemist of diverse backgrounds who have made significant contributions to their fields. An example can be seen in Figure 4 of one of these profiles. The example highlights Marie Maynard Daly, an influential biochemist and the first African American woman in the United States to earn a Ph.D. in Chemistry. Other role models included Patricia S. Cowings, Rosalind Franklin, and Baruj

Benacerraf. The additional profiles can be found in Appendix D, as well as a general outline that can be used for future Bio-CS Bridge curriculum.

Figure 4: Marie Maynard Daly Profile



Ultimately, the curriculum achieved its goal of providing a solid introduction to R programming while making clear the intersection between data science and biology. The use of frequent checkpoints and reminders grants the unit an approachable air and allows students to feel comfortable moving at their own unique paces. Additionally, the ability of the lessons to build off one another creates a stable ladder for students to climb when approaching more challenging areas of the unit. This structure promotes the development of confidence and comfort in students and encourages fearless learning. Most importantly, the curriculum is welcoming to students of all backgrounds and plays a role in combating imposter syndrome by providing representation for students who often struggle most with the phenomenon. This aligns well with the goal set early in the project to allow for students to see themselves as scientists. Hopefully the unit can be built upon with more advanced R concepts in the future. The conclusions and recommendations below provide some insight in moving forward with this endeavor.

5. Conclusions and Recommendations

5.1 Conclusions

While the Bio-CS Bridge R unit was created with many goals in mind, it is important to acknowledge that all educational material has its positives and negatives. Many road blocks arose in the creation of the lessons, the most influential being the setbacks associated with Replit early in the project. With problem solving consuming much of the IQP, curriculum development was often found in the alignment stage of Backwards Design. Despite this, it is hoped that the time spent revising allows for a more positively impactful unit overall. The finalized curriculum supports the original goals set in the initial stages of the project, thus it is hoped that the unit can achieve its purpose of allowing students to become confident in their scientific and data abilities. With the world constantly undergoing technological expansion, it is of utmost importance that students of all backgrounds have the opportunity to gain exposure to fields such as data science. Furthermore, it is important that such exposures allow for real world problem solving opportunities as well as the possibility for students to relate back to their learning. With this in mind, the current Bio-CS Bridge R curriculum should be used to provide a solid foundation in the basics of statistics, R and data visualization.

5.2 Recommendations

The following suggestions might serve as beneficial in both the implementation of the new Bio-CS Bridge R unit as well as the development of lessons beyond the introductory level. These recommendations serve as suggestions for future direction as opposed to strict guidelines.

Teachers are Encouraged to Pace The Curriculum Appropriately Based On Individual Classroom Needs

The material in the Bio-CS Bridge R unit was developed so that students can work at a pace that makes them comfortable. The curriculum often reinforces tasks learned in previous lessons (e.g.. having students install new packages and repeat other basic tasks as lessons progress) to allow students the opportunity to become independent in the work. While this can be

effective in creating a safe environment for self paced learning, it is recommended that teachers do not forget to be consistently involved in the process. For example, there are various notes in the lessons that remind students when might be an appropriate time to check in with a teacher for help. It is expected that educators remain available during classroom completion of these activities and read all code and teacher notes prior to student exposure. That way, student learning can be supplemented at all paces and teachers can be prepared for the unique questions that students might develop along the way.

Additionally, it is recommended that educators stay up to date on their own continued education. Implementing interdisciplinary curriculum effectively can be difficult when one lacks the cross disciplinary knowledge to disseminate teaching. Thus, districts participating in the Bio-CS Bridge Initiative are highly encouraged to develop an empathetic mindset towards educators, and provide adequate support in allowing for such continued education. This may include paid workshops, or time allotted in the workday to become comfortable with foreign material, such as new programming languages.

Moving Towards More Accessible IDEs

R Studio is an excellent IDE for working with data in R, as it was designed for visualizations of the sort and is often used in industry and academia. However, switching to R Studio created some barriers to accessibility that were originally avoided in the decision to use Replit. Thus, while R Studio works well for those who are able to download it, there is some concern for classrooms that do not have access to devices that can access downloadable software. Oftentimes, school districts do not have permission to allow teachers or students to download new software onto school provided devices, and some devices, such as tablets, do not have the same functionality. Thus, it is important that the developers of the Bio-CS Bridge curriculum stay up to date on current IDEs so that an open sourced, web based option can be made available if necessary.

Additionally, in the final weeks of developing the Bio-CS Bridge R unit, R Studio released an announcement that they would be renaming to “Posit” and adapting their software to also integrate Python, another language popular in data science. Currently, it is unknown if this shift will affect the functionality of R Studio or the current R Bio-CS Bridge lessons. It is

important that educators participating in the program as well as future Bio-CS Bridge curriculum developers remain aware of this shift so that they may adapt if necessary. Another consideration related to this might be to create a merged Bio-CS Bridge Python and R curriculum using the software, depending on what the new functionality of Posit offers.

Expanding the Bio-CS Bridge R Unit

While the current Bio-CS Bridge R unit is complete at the introductory level, it would be ideal for future Bio-CS Bridge curriculum developers to expand on the curriculum to allow it to dive deeper into data exploration as well as R itself. One of the initial goals set out in the initial Backwards Design process (Table 3) was that “Students will develop an intuition of when to use different types of data visualizations” (Byrne, 28). This goal was semi-achieved, as students created bar plots and time series line plots in the curriculum. However, the lessons were not developed to a point that allowed for student freedom of choice in the creation of their graphs. If setbacks with Replit had not occurred, it might have been possible to see the curriculum include more independent challenges. Thus, advice for future curriculum developers include the following.

Future additions to the Bio-CS Bridge R unit would benefit from including more examples of various graph types as well as structured hypotheses to show students when use of each graph type is appropriate. Once the material sets students up to understand each graph type individually, it would be ideal to have students answer more open-ended questions that grant them greater freedom in their visualizations. This might allow students to become more independent in their understanding of how to represent data. The unit could potentially be wrapped up with a final project that might involve having students research an interest in the Beecology dataset, generate a hypothesis, and source the necessary data from the Beecology website to work with in R. This would allow students to utilize the Weintrop computational thinking skills of “Data Collection” and “Data Manipulation” as well as allow them more independence in their work. It would also prompt students to become comfortable reading and citing relevant literature, which they can use in comparison to the Beecology dataset. Additionally, future paths for the curriculum might involve expanding deeper into *ggplot2* graphing capabilities, as the current unit only touches on it very briefly. Other avenues might

include merging the curriculum with more advanced AP Statistics, which high schoolers often complete in conjunction with biology. This would involve a more fleshed out curriculum that aligns with the goals for AP Statistics and allows students to apply their learning in real world applications. By expanding the curriculum to overlap at the AP level, Bio-CS Bridge students are provided further incentive to see the applications of their learning and perhaps even to pursue data science later on. Finally, the unit would benefit from the addition of more complete answer sheets and teachers' notes to ensure both educators and students receive the support necessary for success.

Ultimately, the new Bio-CS Bridge R unit should be implemented with the goal of supporting all students in mind. Every individual deserves an opportunity to be supported as they enter new and intimidating fields, such as computer science. It is the job of educators, curriculum developers, and school districts alike to create a safe and supportive environment at every level of this process. With this conscious effort, the unit will achieve the project goal of allowing all students to see themselves as scientists.

References

- Aharon, G. (2017) Students' attitudes towards interdisciplinary education: a course on interdisciplinary aspects of science and engineering education, *European Journal of Engineering Education*, 42:3, 260-270, DOI: 10.1080/03043797.2016.1158789
- Ahmed, A., Kaushal, A., Cruz, T., Kobuse, Y., & Wang, K. (2020). Why is there a higher rate of impostor syndrome among BIPOC?. <https://doi.org/10.5281/zenodo.4310477>
- Archer, L., DeWitt, J., Osborne, J., Dillon, J., Willis, B. and Wong, B. (2010), "Doing" science versus "being" a scientist: Examining 10/11-year-old schoolchildren's constructions of science through the lens of identity. *Sci. Ed.*, 94: 617-639. <https://doi.org/10.1002/sce.20399>
- Beecology Project. (n.d.). <https://beecology.wpi.edu/>
- Bio-CS Bridge. (n.d.). <https://biocsbridge.wpi.edu>
- Bravata, D. M., Watts, S. A., Keefer, A. L., Madhusudhan, D. K., Taylor, K. T., Clark, D. M., Nelson, R. S., Cokley, K. O., & Hagg, H. K. (2020). Prevalence, Predictors, and Treatment of Impostor Syndrome: a Systematic Review. *Journal of general internal medicine*, 35(4), 1252–1275. <https://doi.org/10.1007/s11606-019-05364-1>
- Braun, D., & Huwer, J. (2022). Computational literacy in science education—A systematic review. *Frontiers in Education*, 7. <https://doi.org/10.3389/feduc.2022.937048>
- Corkindale, G. (2008). *Overcoming Imposter Syndrome*. Harvard Business Review. <https://hbr.org/2008/05/overcoming-imposter-syndrome>
- Dobrescu, L., Holden, R., & Motta, A., Cultural Context in Standardized Tests (2021). UNSW Economics Working Paper 2021-08, <http://dx.doi.org/10.2139/ssrn.3983663>
- Galindo, C., & Sonnenschein, S. (2015). Decreasing the SES math achievement gap: Initial math proficiency and home learning environments. *Contemporary Educational Psychology*, 43, 25–38. <https://doi.org/10.1016/j.cedpsych.2015.08.003>
- Handelsman, J., Miller, S., & Pfund, C. (2006) *Scientific teaching*. Wisconsin Program for Scientific Teaching ; Roberts and Co. ; W.H. Freeman.
- Kafai, Y. B., & Proctor, C. (2022). A Revaluation of Computational Thinking in K–12 Education: Moving Toward Computational Literacies. *Educational Researcher*, 51(2), 146–151. <https://doi.org/10.3102/0013189X211057904>

- Kim, K. H., & Zabelina, D. (2015). Cultural Bias in Assessment: Can Creativity Assessment Help? *The International Journal of Critical Pedagogy*, 6(2).
<https://libjournal.uncg.edu/ijcp/article/view/301/856>
- Kjelvik, M., & Schultheis, E. (2019). Getting Messy with Authentic Data: Exploring the Potential of Using Data from Scientific Research to Support Student Data Literacy. *CBE life sciences education*. 18. 10.1187/cbe.18-02-0023.
- Kleffner, R., Flatten, J., Leaver-Fay, A., Baker, D., Siegel, J., Khatib F., & Cooper, S. (2017) Foldit Standalone: a video game-derived protein structure manipulation interface using Rosetta. *Bioinformatics* btx283.
- MacPhee, D., Farro, S., & Cannetto S., (2013) Academic Self-Efficacy and Performance of Underrepresented STEM Majors: Gender, Ethnic, and Social Class Patterns. *ASAP*
<https://doi.org/10.1111/asap.12033>
- Pala, Ş.M., Başibüyük, A. (2021). The Predictive Effect of Digital Literacy, Self-Control and Motivation on the Academic Achievement in the Science, Technology and Society Learning Area. *Tech Know Learn*. <https://doi.org/10.1007/s10758-021-09538-x>
- Prinnebow, C., Sheridan, J., Carnes, M., Fine, E., Handlesman, J. (2006) The department chair and climate: Contradicting perceptions. Unpublished
- R Core Team (2021). R: A language and environment for statistical computing. *R Foundation for Statistical Computing*, Vienna, Austria. <https://www.R-project.org/>.
- Sentance, S., & Csizmadia, A. (2017) Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Educ Inf Technol* 22, 469–495.
<https://doi.org/10.1007/s10639-016-9482-0>
- Sheffield, S.L., Cook, M.L., Richezza, V.J. *et al.* (2021) Perceptions of scientists held by US students can be broadened through inclusive classroom interventions. *Commun Earth Environ* 2, 83. <https://doi.org/10.1038/s43247-021-00156-0>
- State of Computer Science Education Accelerating Action Through Advocacy Advocacy Coalition.* (2021). https://advocacy.code.org/2021_state_of_cs.pdf
- Weintrop, D., Beheshti, E., Horn, M., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *J Sci Educ Technol*, (25), 127-147. 10.1007/s10956-015-9581-5
- Wiggins, G., & McTighe, J. (2011). The Understanding by Design Guide to Creating

High-Quality Units. *Association for Supervision and Curriculum Development*.

<http://www.ascd.org/Publications/Books/Overview/The-Understanding-by-Design-Guide-to-Creating-High-Quality-Units.aspx>

- Williams, W., Papierno, P., Makel, M., Ceci, S. (2004). Thinking Like A Scientist About Real-World Problems: The Cornell Institute for Research on Children Science Education Program. *J Applied Developmental Psychology*.
<https://doi.org/10.1016/j.appdev.2003.11.002>
- Yadav, A., Gretter, S., Hambrusch S., & Sands, P. (2016) Expanding computer science education in schools: understanding teacher experiences and challenges, *Computer Science Education*, 26:4, 235-254, DOI: 10.1080/08993408.2016.1257418
- Yang, D., Baek, Y., Ching, Y.-H., Swanson, S., Chittoori, B., & Wang, S. (2021). Infusing Computational Thinking in an Integrated STEM Curriculum: User Reactions and Lessons Learned. *European Journal of STEM Education*, 6(1), 04.
<https://doi.org/10.20897/ejsteme/9560>

Appendix A: Example of A Dataset

	A	B	C	D	E	F	G
1	time	year	month	species	behavior	gender	tongue_length
2	2022-09-29T00:00:0	2022	9	impatiens	nectar	female	medium
3	2022-09-21T20:46:4	2022	9	impatiens	nectar	male	medium
4	2022-09-21T20:40:3	2022	9	impatiens	nectar	male	medium
5	2022-09-12T00:00:0	2022	9	vagans	nectar	male	medium
6	2022-09-12T00:00:0	2022	9	impatiens	nectar	male	medium
7	2022-09-12T00:00:0	2022	9	impatiens	nectar	male	medium
8	2022-09-11T00:00:0	2022	9	impatiens	nectar	female	medium
9	2022-09-09T00:00:0	2022	9	impatiens	nectar	female	medium
10	2022-09-08T00:00:0	2022	9	impatiens	nectar	female	medium

Appendix A shows the first ten lines of BeeData.

Appendix B: Graphing with Beecology, Lesson Example

Graphing with Beecology

Name _____ Date: _____ Period: _____

You will now be combining your knowledge in ecology with your new R programming skills to test hypotheses about Beecology data. The dataset contains information about pollinator species observed by Beecology users around the New England area.


For each activity below:

 means to complete this task


 means to write an answer here

Activity #1: Asking Questions and Making Predictions






In this section, you will be considering ecologically relevant questions that can be explored through the generation of hypotheses and visualization of data. You will later test your hypotheses using R and will see how programming can supplement our understanding of biology!

- a.  Let's start by creating a new folder on your laptop titled Beecology. Load the dataset provided by your teacher in Excel, saving it to the Beecology folder.

★ This is important, since R requires working files to be stored in the location as any data used in the code. We want to make sure our materials are in an organized location to prevent path errors later. **Path errors** are errors relating to the **path** your device takes to find a file.

- b.  Take a look at the top row of the Excel file; it contains the variable names. The columns under each are the collected data for each variable by Beecology users.

★ Note: Now is a good time to reach out to a teacher if you are having trouble understanding the dataset. Reading files of this format can certainly be an adjustment at first glance, so don't hesitate to ask for clarification!

- c.  Below, list the variable names you see in the dataset. If you have any confusion about what they might be, turn to the Beecology website for guidance.
- d.  Looking at the variables you listed, can you identify one measure of the functional diversity of bumblebees in the dataset? Write your answer below.
- e.  Using your understanding of biology, do you predict that the functional diversity of bumblebees has changed over time? If so, in what ways?
- f.  What variables in the dataset might differ over time? Can you make any predictions?
- g.  Given the variables in the dataset, make a hypothesis about the data. Give a brief explanation for your prediction and an idea of how you may test it.

When you have finished, move to the next page for Activity #2.

Activity #2: Preparing R for Data Analysis

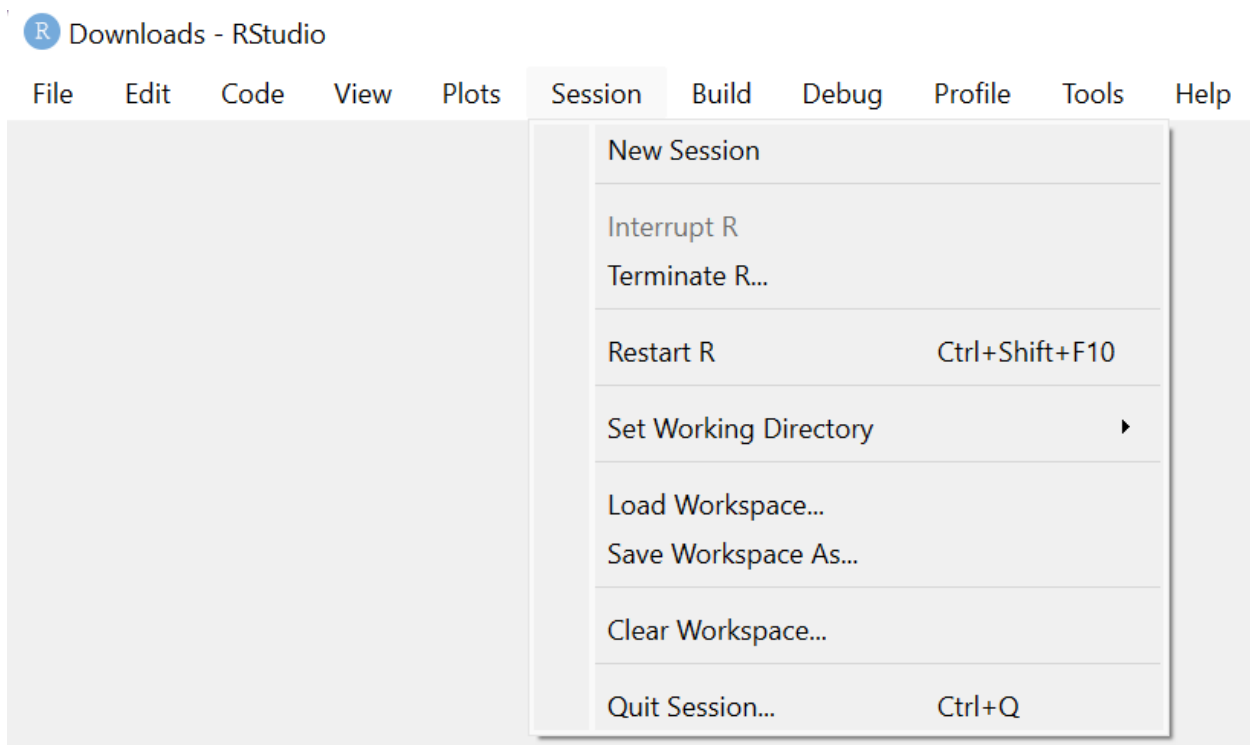
Thanks to data analysis tools such as R, we can test hypotheses right on our laptops, so long as we have the data ready and accessible. In this section, we are going to test some of the hypotheses you made above using R and graphing.


Setting A Working Directory

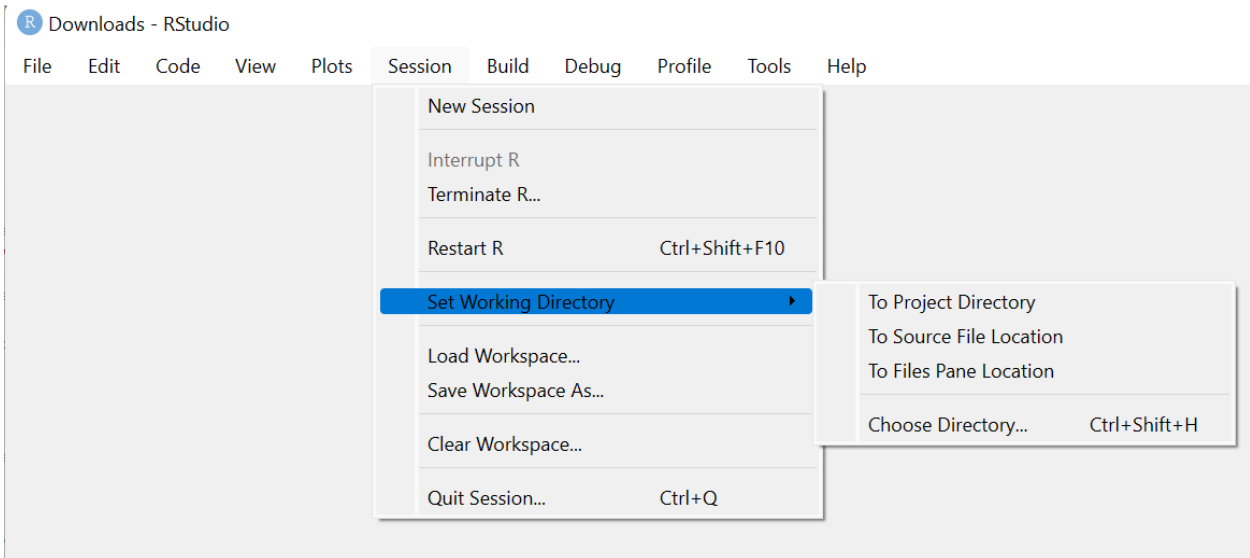
1. Open R Studio. Create a new file titled hypothesisTesting and save it to the Beecology folder.


Next, we need to connect our **working directory**, or more simply, the folder in which we will store our working materials. Setting a working directory allows R Studio to know where to go when we request information, such as data files, from our local computer. To set your working directory to the Beecology folder, follow the following steps.

2. Click “Session”, located on the top bar and pictured below.



3.  Next, select “Set Working Directory”, as shown below. From the drop down menu, click “Choose Directory” and select the Beecology folder from your laptop.



4.  To double check you have set the correct working directory, use the command below and run the code. The working directory should contain the location of the Beecology folder if set correctly.

```
# Retrieve the location of your working directory  
getwd()
```

★ Now is a good time to pause and ask your teacher for help if you are having trouble setting the working directory. This is a crucial step in making sure your code runs smoothly later!

Loading the Data

1. If you haven't already, make sure both the Excel file provided by your teacher and the R hypothesisTesting file are stored in your Beecology folder.
2. Now, it is time to load the data into R Studio. Go to R Studio and type the following line into your hypothesisTesting file. You will need to replace the path with the file's location on your laptop.

```
beeData <- read.csv("C:/Users/UserName/Beecology/BeeData.csv")  
beeData
```

3. To load the first few lines of the dataset, use the head() command, as practiced in the previous lesson.
4. Take a look at the summary provided by head() when run. What variables are contained in the dataset? Do these match with those you identified in the Excel sheet?

Activity #3: Testing Hypotheses and Building Graphs

Now, we want to test hypotheses using R and graphing. Let's first prepare our data to be plotted.

Preparing the Data

Next, we need to convert the variables to **factors**. Factors are the data objects that R uses to store categorized data in levels, a way that can be quantified and visualized. Because the data is categorical, we need to tell R to make a count of each category.

- a. To convert the variable for bee gender to a factor, you can use the below code.

```
gender <- factor(beeData$gender)
```

Here,



gender is the name of the variable which you are assigning the factor to
factor() is the function being used to convert the variable to a factor
beeData\$gender is the object being converted to a factor

beeData is the dataset

\$ denotes that the next item is a variable in the dataset

The 'gender' following \$ is the variable in beeData being converted to a factor

You are setting gender equal to the value of beeData\$gender after it has been converted to a factor.

- b.  Now, on your own, try using this structure to convert the non numeric variables in the Beecology dataset to factors.
- c.  In your own words, explain why it is important to convert non-numeric variables into factors.

Creating Data Frame from Factors

Now, we want to create a new data frame from beeData, but this time, with all the variables converted to the correct format. To do so, we can use the data.frame() command. You explored this command in the previous lesson, but we will give a review to refresh.

The data.frame() command works as follows:

```
df <- data.frame(column1, column2, column3)
```

Here, columns 1 through 3 are the columns of variables contained in the dataframe. In our example, these would be the beeData variables converted to factors as well as the numeric variables from beeData. The 'df' at the start of the code is simply the name of the data frame. We can call ours whatever we wish so long as it is appropriate and follows naming conventions.

1. ✓ Now, use the above format to make a data frame containing all the new factor variables as well as the numeric variables from `beeData`. Print the data frame by calling on the name, as you would a variable or vector name.
2. ✓ Suppose we want to see the summary statistics for the new data frame. Use the `summary()` function to retrieve this.
3. ✎ What information does the summary function give you about the data frame?
4. ✎ Which of these data are meaningful? Is there a reason to have the same summary statistics for categorical data, such as `species` and `tongue_length`, as for time measurements like `month` and `year`? Why or why not?
5. ✎ Are there more male or female bees in the dataset?
6. ✎ Which tongue length is predominant in the dataset? Why might this be?

Plotting A Variable

Before we begin to test hypotheses of two variables, let's start by plotting the count of single variables.

This can be very useful if we want to see the total counts of categorical data as compared to one another.


To start, let's plot the total counts of each bee species in the dataset.

1. ✎ Without looking at the summary statistics, do you expect to see similar numbers of every species in the dataset? Why or why not?

To plot the counts for each species in a bar graph, we can use the plot function. The previous lesson gave a brief introduction to plot() in the bonus section, but if you did not make it there don't fret! We will provide more explanation here.




The plot() function takes either an x and y value in the form of vectors, a data frame containing two columns, or a vector variable that can be plotted by count. In this case, we will be plotting a singular vector variable, as seen below:

```
plot(variable)
```

2.  To plot the total counts of each species in the Beecology dataset, run the below code.


```
plot(species)
```

★ Note: Be sure to check that you use the variable name you used for species. Errors commonly occur when we are not consistent with variable names, so this is always important to be mindful of!

3.  Which bee species is seen most often by Beecology users?
4.  Do you have any hypotheses about why this might be?
5.  Now, try on your own making plots for the single variables behavior and tongue length.

Next, let's look at the distribution of tongue length over species. This will allow you to look at two variables compared.


First, we will make a hypothesis.

6.  Consider the following and make a hypothesis. Do you anticipate seeing a mix of lengths among species or will bees of the same species have the same tongue length classification? Write your hypothesis below and explain your reasoning.

Now, we will make a **table** of the variables to make them easy to plot. A **table** in R gives the frequency of categorical variables in the form of columns and rows. We can use this to get the count of overlap between two variables.

A table can be created and stored using the `table()` function as shown below:

```
tableName <- table(column1, column2)
```

7. Create a table called *speciesTongue* of the variables *species* and *tongue length*.
8. Display the table by calling its name.
9.  Thus far, does your hypothesis seem to be supported by the data in the table?

Next, we will go step by step in building a **bar plot**. A **bar plot** is a plot that shows the frequency of categorical data using rectangles, the height of which represents count.

To create a bar plot in R, we can use the `barplot()` function. Let's go step by step to build a bar plot!

10. Start by running the below code in R Studio.


```
barplot(speciesTongue, beside = TRUE)
```

Here, `barplot()` takes the table *speciesTongue* and creates a barplot showing the count of species type for each tongue length. The '`beside = TRUE`' portion tells R that the bars should be grouped beside one another as opposed to stacked.


★ Feel free to play around and remove the '`beside = TRUE`' portion to see how the graph looks without it! Just be sure to change the code back before moving on in the tutorial.

11. Now, let's work on adding a title. To add a title, you can adjust your code to include a '`main =`' line, as shown below. Add the '`main =`' line to your `barplot` code and replace '`TITLE`' with an appropriate title for the graph.

```
barplot(speciesTongue, main="TITLE", beside = TRUE)
```



12.  Now, we need to add a label for the x axis. We can do this by adding 'xlab = "LABEL"' as shown below. Replace the 'LABEL' in the line with an appropriate label for the x axis and run the code.

```
barplot(speciesTongue, main="TITLE", xlab="LABEL",  
        beside = TRUE)
```

13.  Finally, we are missing a legend. Add a legend by using the 'legend = rownames()' line below.

```
barplot(speciesTongue, main="TITLE", xlab="LABEL", legend =  
        rownames(speciesTongue), beside = TRUE)
```

 **Congratulations! You just made your first bar plot in R!** 

14.  Does the plot support your hypothesis? Why or why not?
15.  What observations can you make about the data shown in the table and the bar plot?
-

Appendix C: Vectors and Data Frames Bonus Activities


Bonus Activity: Plotting

If you have finished the lesson, now's the chance to get a sneak peek at graphing in R!


Suppose we want to graph the observances of *B. impatiens* by Beecology users across 2021 using the data frame just created. We can use the plot function to do so.

The plot function takes two variables, the first being the X values and the second being the Y values. Its format can be seen below:

```
plot(x, y)
```

 Which variable will be on the x axis? On the y? Does it matter for this type of data?

We can also use the plot() function by inserting the data frame in the parentheses. This is because our data frame only contains two variables.

 Using the plot() formula above, plot the count of bee observances for each month throughout the year of 2021.

Did you plot either of the following?


```
plot(months, beeCount)
plot(monthlyCounts)
```

 Is there a difference between the two when plotted?

Bonus Activity #2: Plotting with Iris

Now back to Iris.

Suppose we wish to see the relationship between petal width and petal length of the flowers in the Iris dataset. First, let's consider how our variables will be used.

 Which variable will be on the x axis? On the y? Does it matter for this type of data?

Next, we need to understand how variables can be extracted from data frames. In R, variable names from data frames and datasets are denoted as the following:

```
dataset$variableName
```

An example from Iris can be seen with the Species variable below:

```
iris$Species
```

To make the species variable easier to call on later, we can assign it the variable “species”:

```
species <- iris$Species
```

✓ Now, using the steps above, assign petal length and width to variables PL and PW.

✓ Using the variables you just created, use the plot() formula above to plot the relationship between PL and PW.

Congrats, you just made your first graphs in R!

Appendix D: Highlighted Profiles of Influential Role Models



Patricia S. Cowings

Patricia S. Cowings is a psychophysiological who is most famous for helping NASA combat space sickness. Her methods work by analyzing biological feedback data and have been successfully implemented in NASA's space missions. Though she never went to space, Patricia was the first woman trained to work as a "scientist astronaut" by NASA.

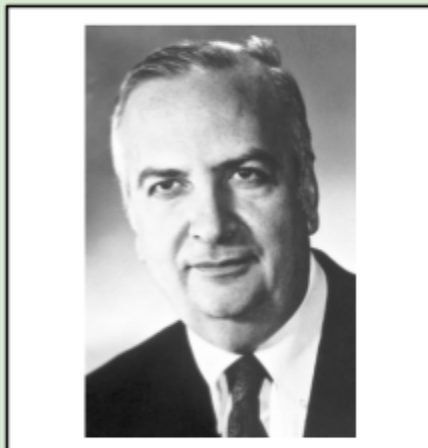
Photo Credit: NASA.gov



Rosalind Franklin

Rosalind Franklin was a British x-ray crystallographer whose most famous work includes the discovery of DNA's density as well as her finding that its structure is helical. Rosalind laid the foundation for later discoveries about DNA and also contributed to advances in understanding the structure of viruses.

Photo Credit: Vittorio Luzzati



Baruj Benacerraf

Baruj Benacerraf is a Venezuelan-American Nobel Prize winner, who heavily contributed to the field of immunology. Baruj won the Nobel Prize while working at Harvard Medical School, where he and his colleagues showed that the immune system's responses are regulated by genetically determined cellular surface structures.

Photo Credit: The Nobel Foundation Archive

