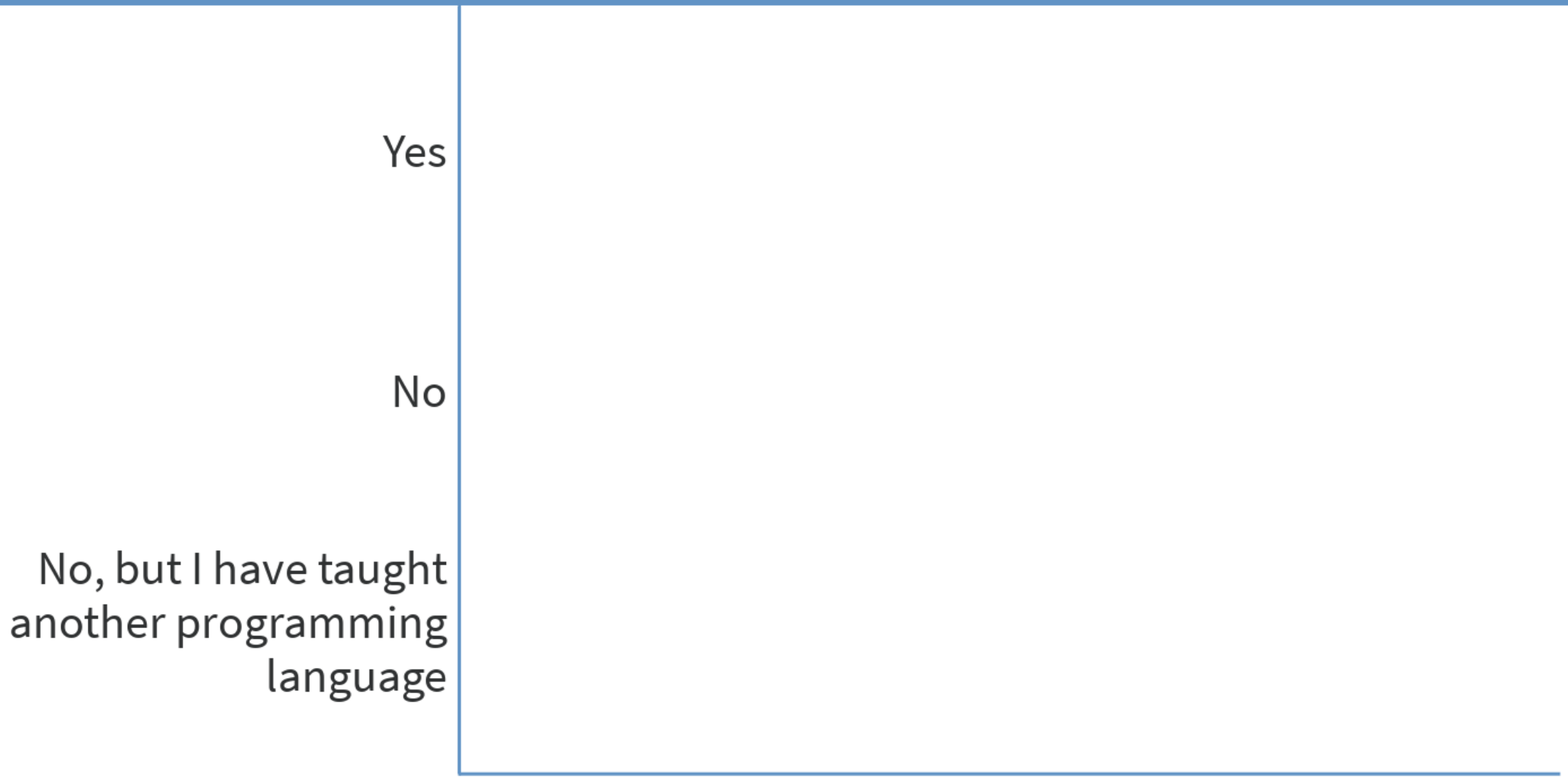
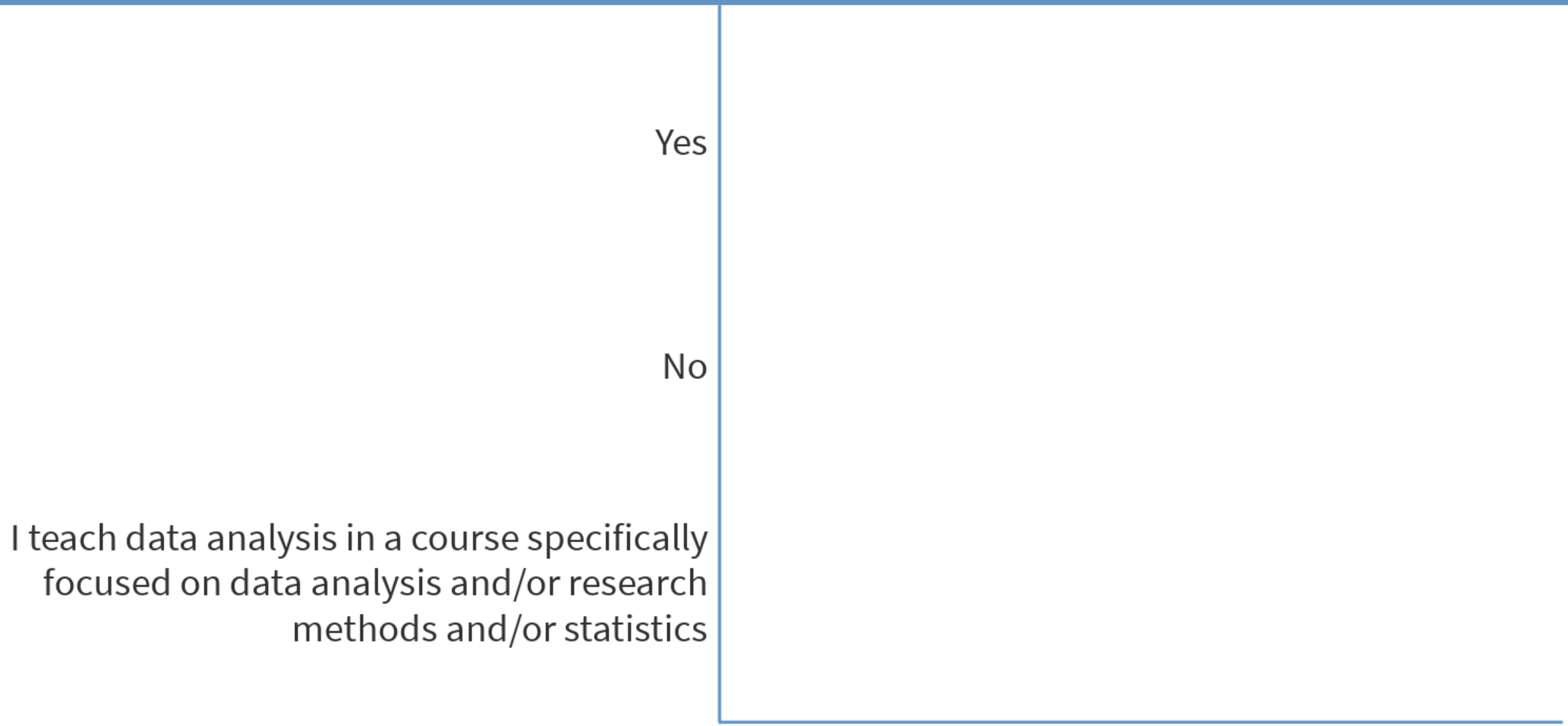
Teaching R and data analysis interactively with {swirl}

Paige Parry, George Fox University

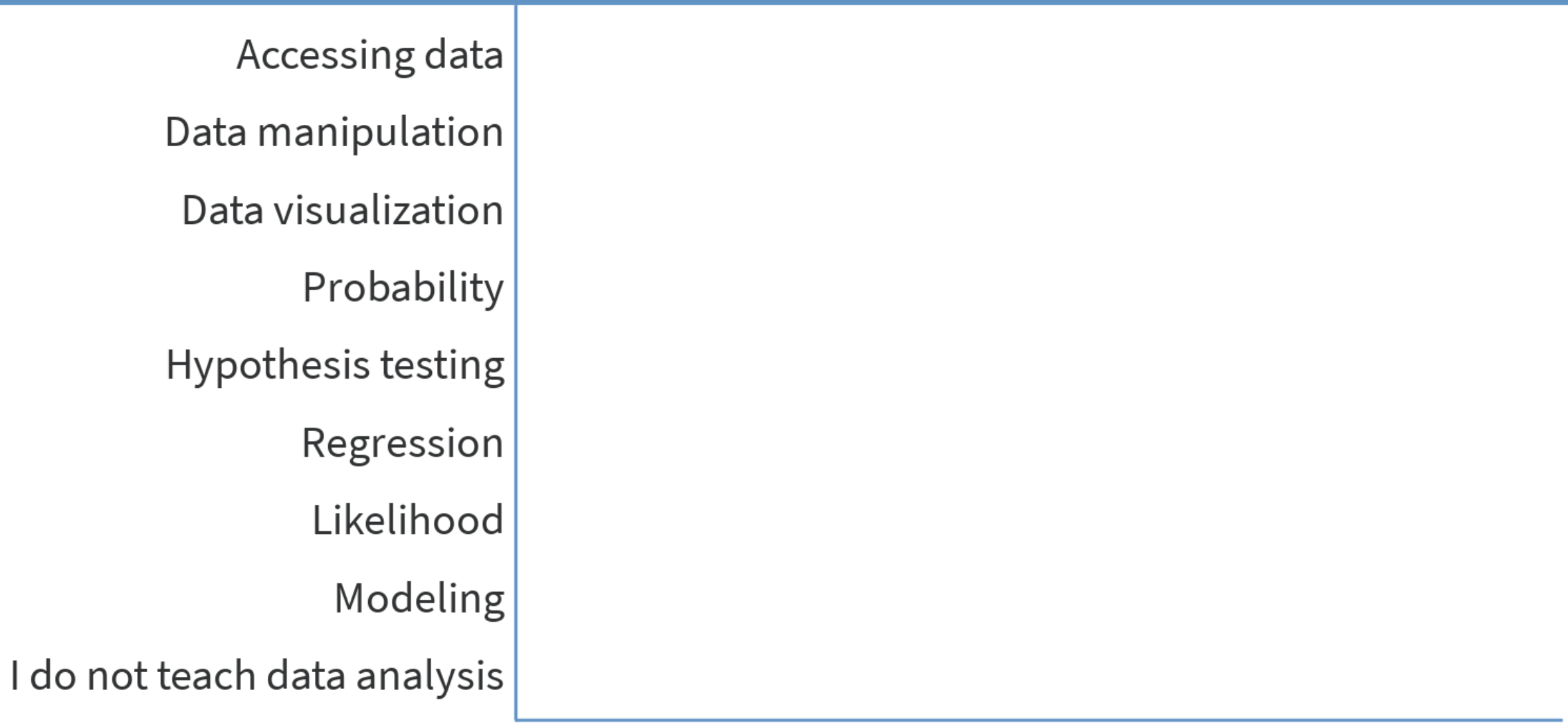
Have you taught R in an undergraduate context?



Do you currently integrate data analysis into yourundergraduate biology course(s)?



If you currently teach data analysis, what topics do you address?



What are some of the challenges that you have identified or perceive to be associated with teaching programming/R in an undergraduate biology course?

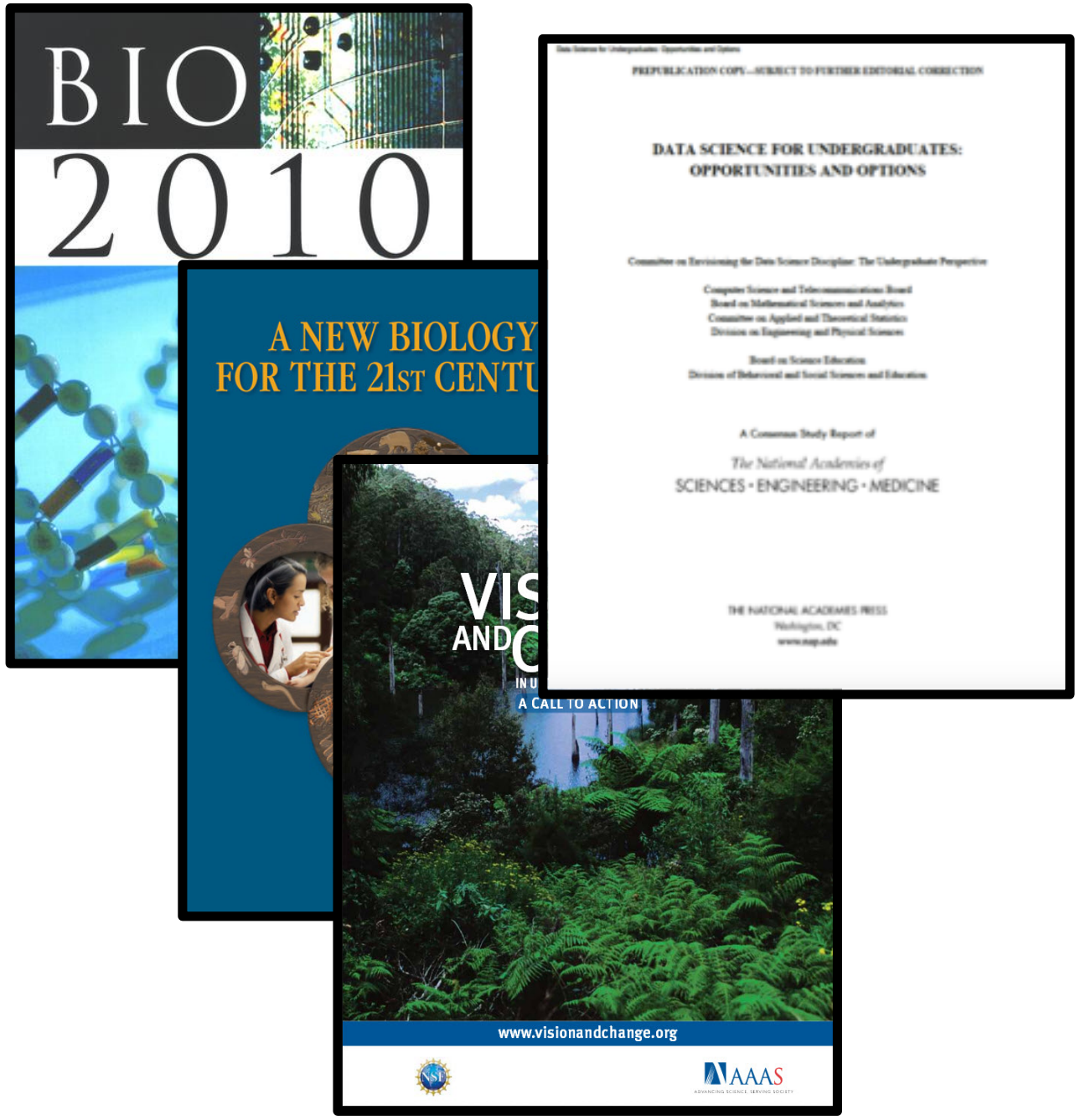
Top

In this session we will:

- Discuss the value and challenges of teaching R and data analysis to undergraduate biology students
- Learn the basic structure of swirl
- Practice swirl using existing, user-contributed lessons
- Develop custom swirl lessons using the swirlify package


WHENEVER I LEARN A NEW SKILL I CONCOCT ELABORATE FANTASY SCENARIOS WHERE IT LETS ME SAVE THE DAY.
OH NO! THE KILLER MUST HAVE FOLLOWED HER ON VACATION!
BUT TO FIND THEM WE'D HAVE TO SEARCH THROUGH 200 MB OF EMAILS LOOKING FOR SOMETHING FORMATTED LIKE AN ADDRESS!
IT'S HOPELESS!
EVERYBODY STAND BACK.
I KNOW REGULAR EXPRESSIONS.
PERL!
TAP TAP

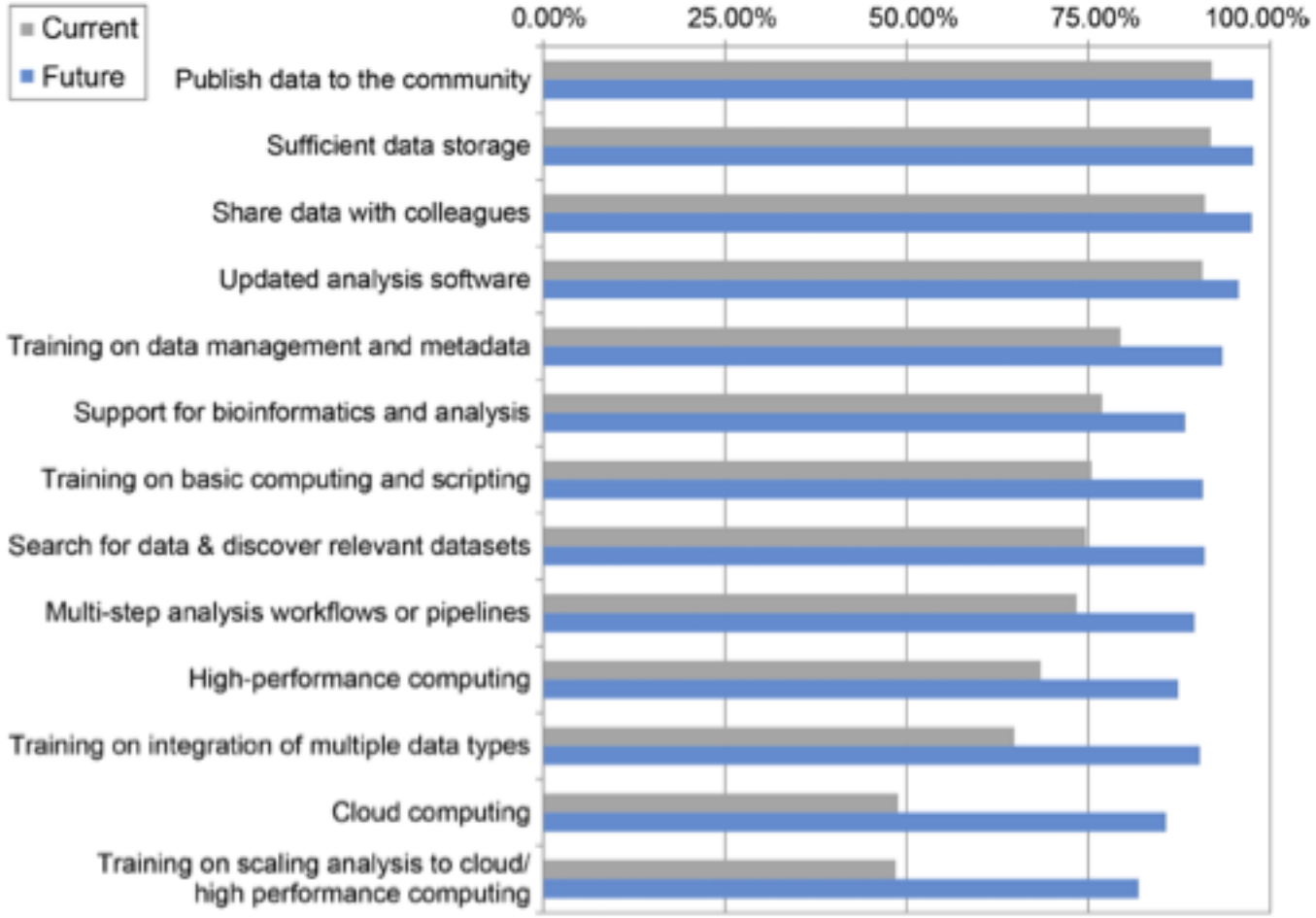

BIO
2010
A NEW BIOLOGY
FOR THE 21ST CENTURY
VISION
AND CHANGE
IN UNDERGRADUATE BIOLOGY EDUCATION
A CALL TO ACTION
www.visionandchange.org
NSF
AAAS



"Work across nearly all domains is becoming more data driven, affecting both the jobs that are available and the skills that are required. As more data and ways of analyzing them become available, more aspects of the economy, society, and daily life will become dependent on data. In future decades, all undergraduates will profit from a fundamental awareness of and competence in data science."

Training in data analysis and programming is among the most pressing unmet needs in biology

Training in data analysis and programming is among the most pressing unmet needs in biology



Barone et al. 2017. Unmet needs for analyzing biological big data: A survey of 704 NSF principal investigators. PLoS Comput Biol 13(10): e1005755.

Training in data analysis and programming is among the most pressing unmet needs in biology

“Coding is ‘as important to modern scientific research as telescopes and test tubes’, but it is critical to dispel the misconception that these skills are intuitive, obvious, or in any way inherent.”

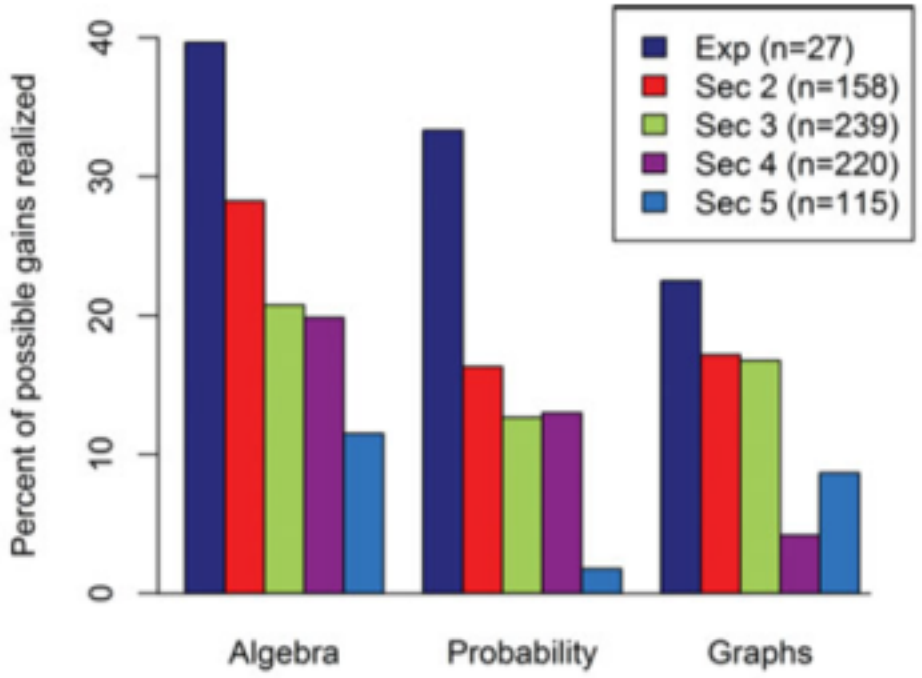
Wilson, G. et al. 2014. Best practices for scientific computing. PLoS Biology 12: e1001745.
Mills, B. 2015. Introducing Mozilla science study groups. Mozilla.

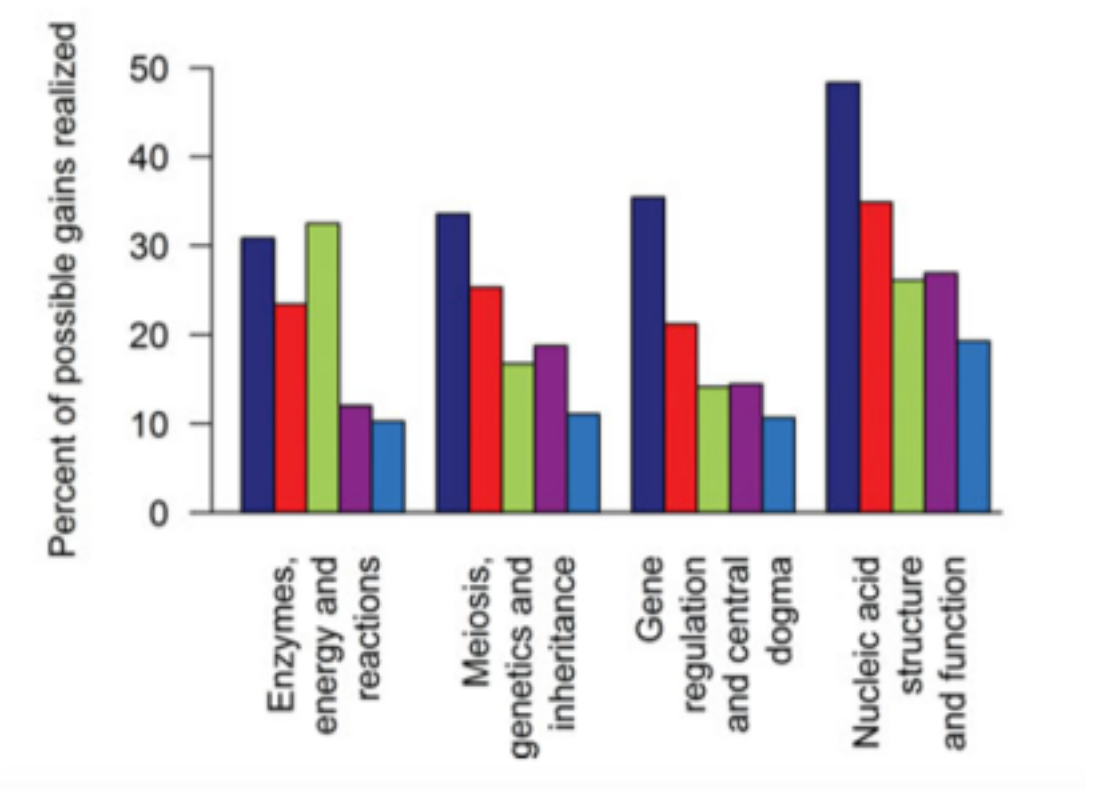
Challenges associated with teaching programming to undergraduate biology students:

- Poor student attitudes toward quantitative exercises; “math fear” and “math anxiety”
- Difficult to teach content and programming skills simultaneously (too little time)
- Lack of curricula accessible to undergraduates
- Steep learning curve due to little to no experience with programming
- Precision necessary to execute code
- Others?

Biology students may learn programming and analysis skills best when integrated with biology

Biology students may learn programming and analysis skills best when integrated with biology





Hester, S. et al. 2014. Integrating quantitative thinking into an introductory biology course improves students' mathematical reasoning in biological contexts. CBE Life Sciences Education 13: 54-64.

What makes teaching scientific computing different from teaching introductory computer science?

1. Scientists work with entities such as signals, images, systems of equations, data tables, etc. Structures such as priority queues and B-trees are of no use or interest to science students.
2. For a scientist, computation is a tool rather than the object of interest. Science students need to see the scientific utility of programming.
3. Scientists have very limited time to devote to the formal study of computation.
4. Scientists use graphics extensively, even at an introductory level.

Kaplan 2004. Teaching computation to undergraduate scientists. SIGSCE 04:3-7.

Why use R?

- Developed as a user-friendly application primarily for data analysis, statistics, and graphics.
- Used extensively in scientific research
- Higher-level programming language with extensive libraries (packages)
- Active user group and substantial online support (mailing lists, user-contributed documentation, Stackoverflow)
- Built-in graphics capabilities
- Data can be read in, graphed, modeled, etc. in only a few lines of code

Learn R, in R.

swirl teaches you R programming and data science
interactively, at your own pace, and right in the R console!

Navigate to: swirlstats.com

Step 1: Open RStudio

Step 2: Install swirl

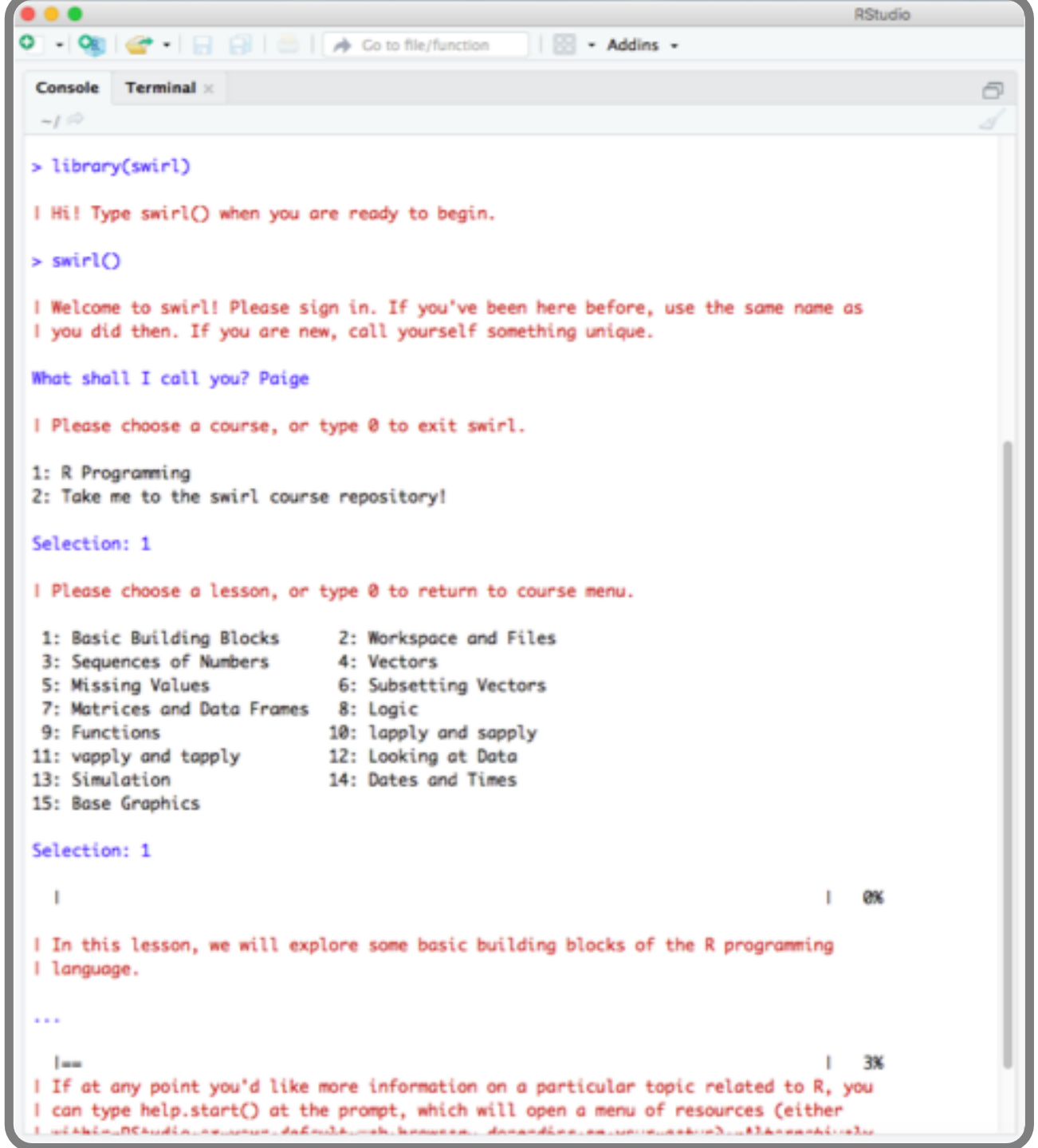
```
> install.packages("swirl")
```

Step 3: Start swirl

```
> library(swirl)
> swirl()
```

Step 4: Install an existing course

https://github.com/swirldev/swirl_courses#swirl-courses


RStudio
Go to file/function
Addins
Console
Terminal
> library(swirl)
| Hi! Type swirl() when you are ready to begin.
> swirl()
| Welcome to swirl! Please sign in. If you've been here before, use the same name as
| you did then. If you are new, call yourself something unique.
What shall I call you? Paige
| Please choose a course, or type 0 to exit swirl.
1: R Programming
2: Take me to the swirl course repository!
Selection: 1
| Please choose a lesson, or type 0 to return to course menu.
1: Basic Building Blocks
2: Workspace and Files
3: Sequences of Numbers
4: Vectors
5: Missing Values
6: Subsetting Vectors
7: Matrices and Data Frames
8: Logic
9: Functions
10: lapply and sapply
11: vapply and tapply
12: Looking at Data
13: Simulation
14: Dates and Times
15: Base Graphics
Selection: 1
| 0%
| In this lesson, we will explore some basic building blocks of the R programming
| language.
...
|== 3%
| If at any point you'd like more information on a particular topic related to R, you
| can type help.start() at the prompt, which will open a menu of resources (either

Installing courses from the swirl repository automatically:

Step 1: Navigate to the swirl course repository and choose a course

https://github.com/swirldev/swirl_courses#swirl-courses

Step 2: Open the swirl library

```
> library(swirl)
```

Step 3: Install the course from the console

```
> install_course("Course Name")
```

Step 4: Start swirl

```
> swirl()
```

Installing courses from the swirl repository manually:

Step 1: Navigate to the swirl course repository and choose a course

https://github.com/swirldev/swirl_courses#swirl-courses

Step 2: Download the swirl course master zip file

https://github.com/swirldev/swirl_courses/archive/master.zip

Step 3: Open the swirl library

```
> library(swirl)
```

Step 4: Install the course from the console, specifying the full file path to the zip file

```
> install_course_zip("/Users/pparry/Desktop/swirl_courses-
  master.zip", multi=TRUE, which_course="Data Analysis")
```

RStudio

Go to file/function | Addins

Console | Terminal

```
> install_course_zip('/Users/pparry/Desktop/swirl_courses-master.zip', multi=TRUE, which_course="
Data Analysis")

| Course installed successfully!

> swirl()

| Welcome to swirl! Please sign in. If you've been here before, use the same name as you did
| then. If you are new, call yourself something unique.

What shall I call you? Paige

| Please choose a course, or type 0 to exit swirl.

1: Data Analysis
2: R Programming
3: Take me to the swirl course repository!

Selection: 1

| Please choose a lesson, or type 0 to return to course menu.

1: Central Tendency
2: Dispersion
3: Data Visualization

Selection: 1

| Attempting to load lesson dependencies...

| Package 'plotrix' loaded correctly!

| This lesson requires the 'openintro' package. Would you like me to install it for you now?

1: Yes
2: No

Selection: 1

| Trying to install package 'openintro' now...

| Package 'openintro' loaded correctly!
```

Installing a custom swirl course:

Step 1: Save course as .swc file to any handy directory

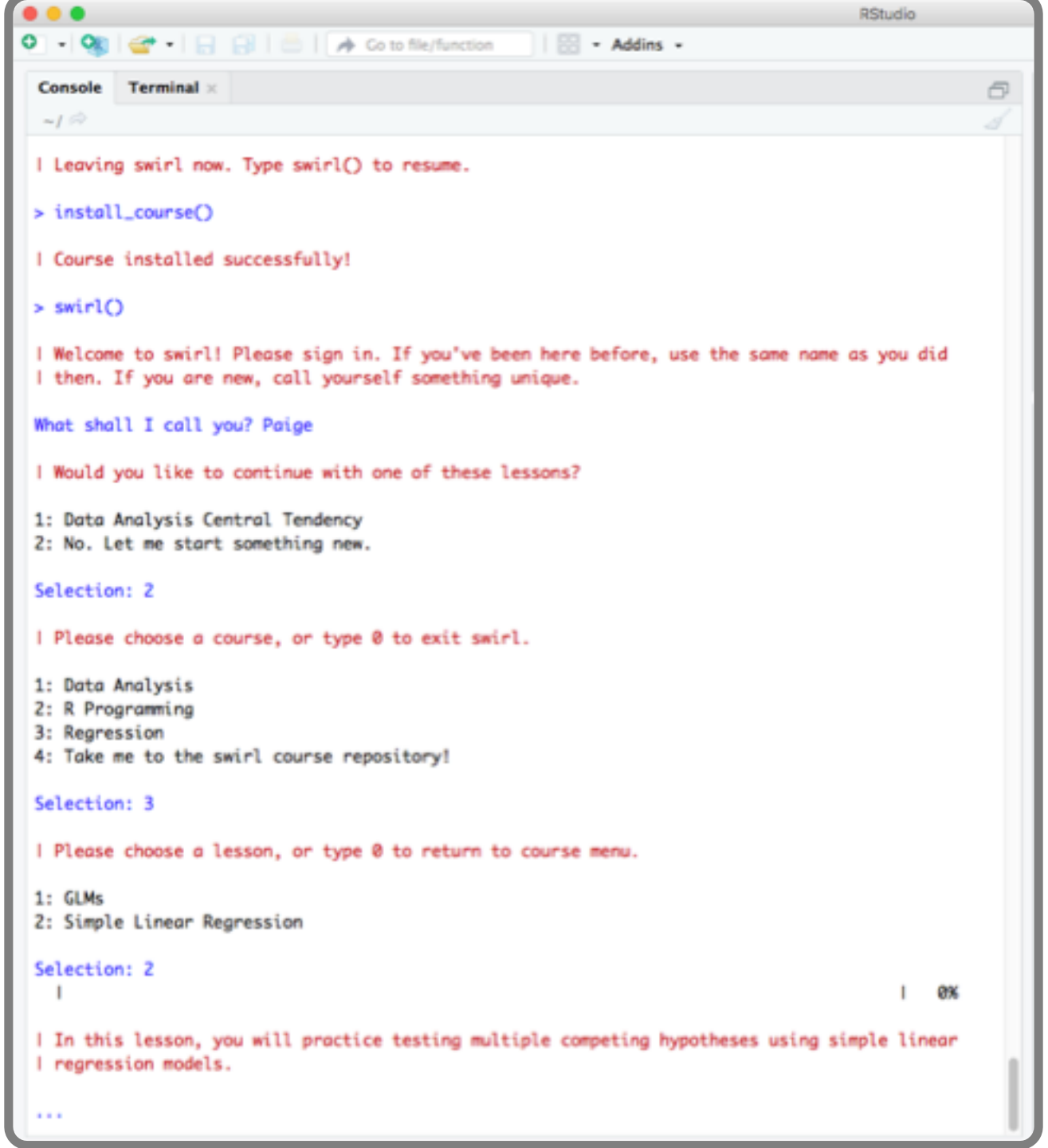
Step 2: Initiate course installation from console

```
> install_course()
```

Step 3: When prompted, navigate to directory and select course

Step 4: Start swirl and navigate to course

Work through the Simple Linear Regression lesson to see an example of integrating programming, data analysis, and biology learning


RStudio
Go to file/function
Addins
Console
Terminal
| Leaving swirl now. Type swirl() to resume.
> install_course()
| Course installed successfully!
> swirl()
| Welcome to swirl! Please sign in. If you've been here before, use the same name as you did
| then. If you are new, call yourself something unique.
What shall I call you? Paige
| Would you like to continue with one of these lessons?
1: Data Analysis Central Tendency
2: No. Let me start something new.
Selection: 2
| Please choose a course, or type 0 to exit swirl.
1: Data Analysis
2: R Programming
3: Regression
4: Take me to the swirl course repository!
Selection: 3
| Please choose a lesson, or type 0 to return to course menu.
1: GLMs
2: Simple Linear Regression
Selection: 2
| 0%
| In this lesson, you will practice testing multiple competing hypotheses using simple linear
| regression models.
...

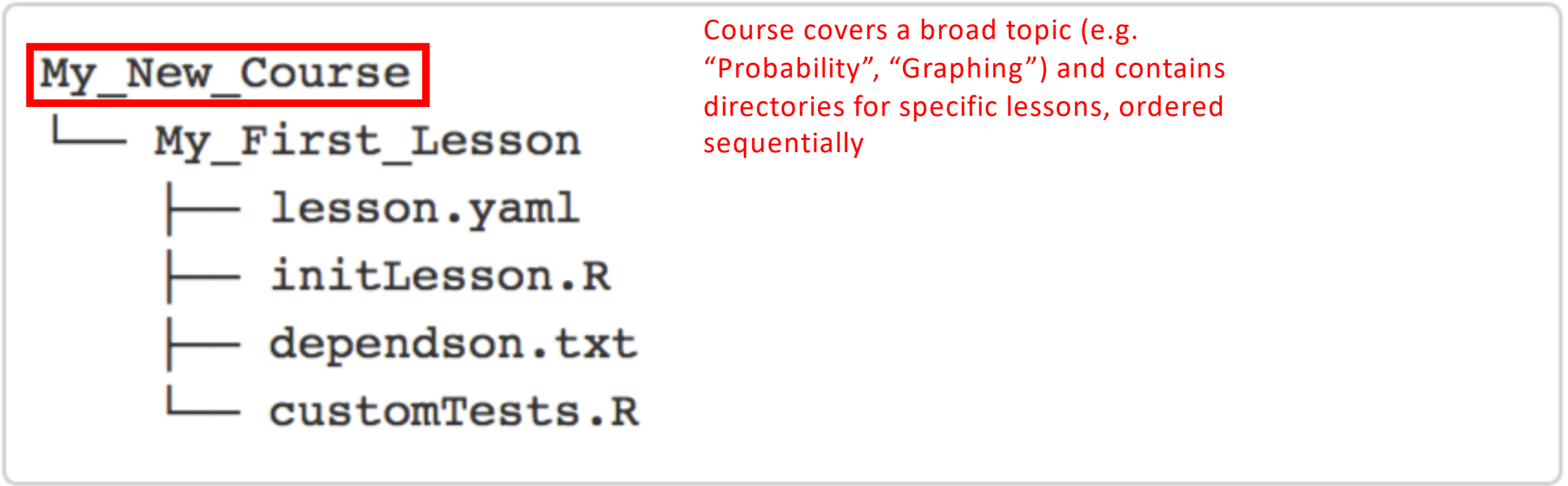
Creating your own course with swirlify:

swirlstats.com/swirlify

Swirl course structure:

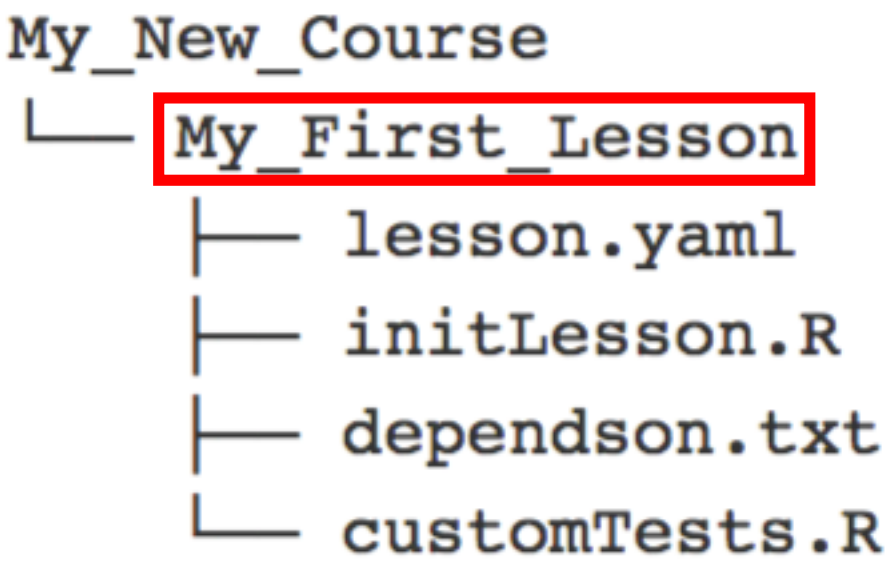
```
My_New_Course
└── My_First_Lesson
    ├── lesson.yaml
    ├── initLesson.R
    ├── dependson.txt
    └── customTests.R
```

Swirl course structure:



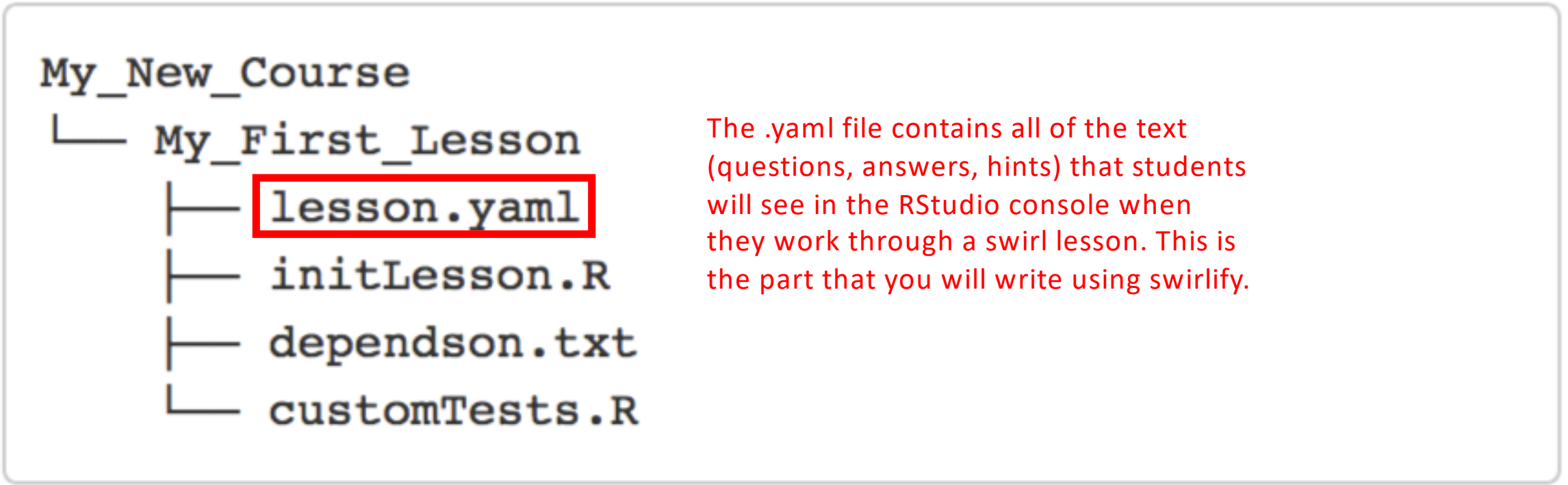
Course covers a broad topic (e.g. “Probability”, “Graphing”) and contains directories for specific lessons, ordered sequentially

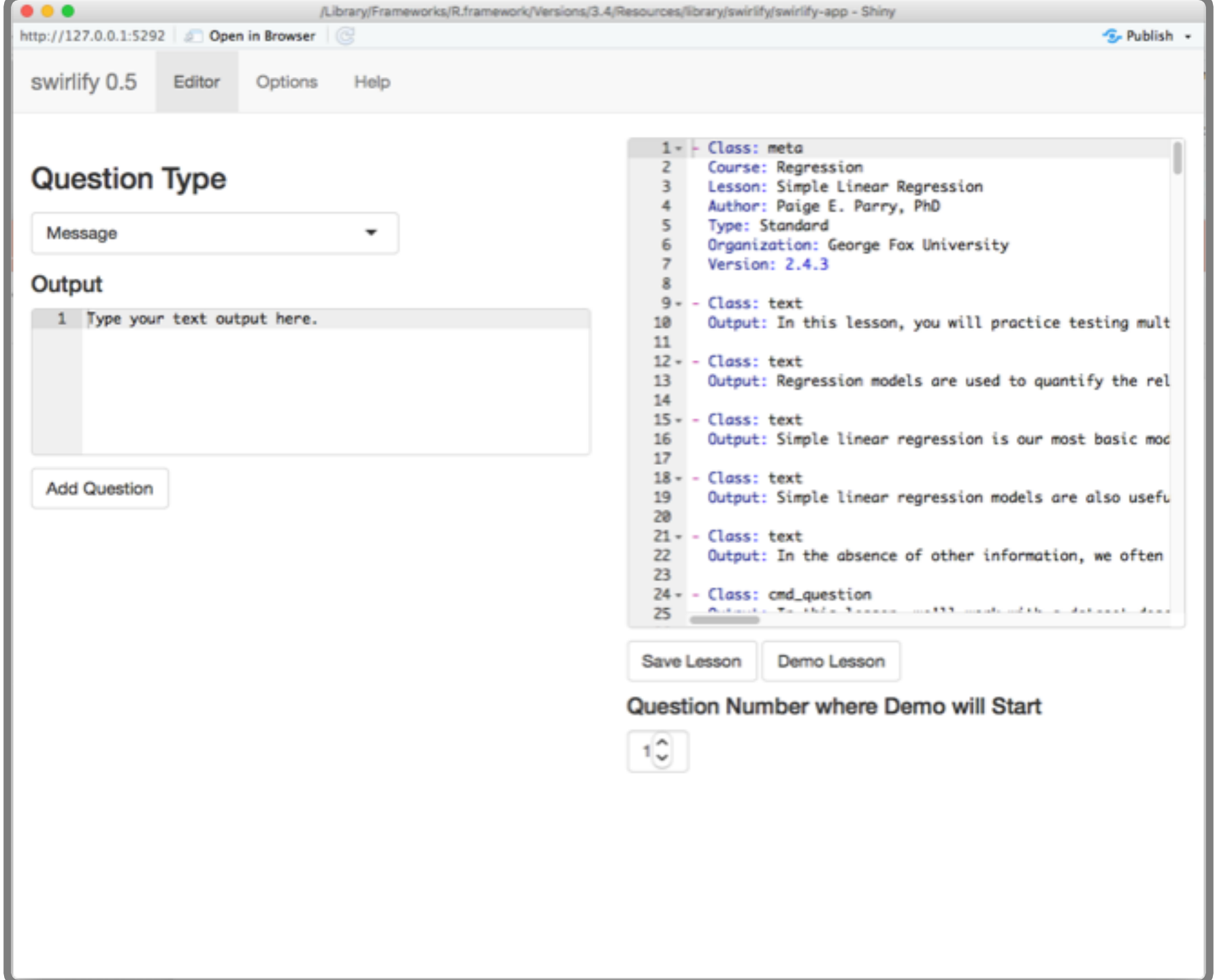
Swirl course structure:



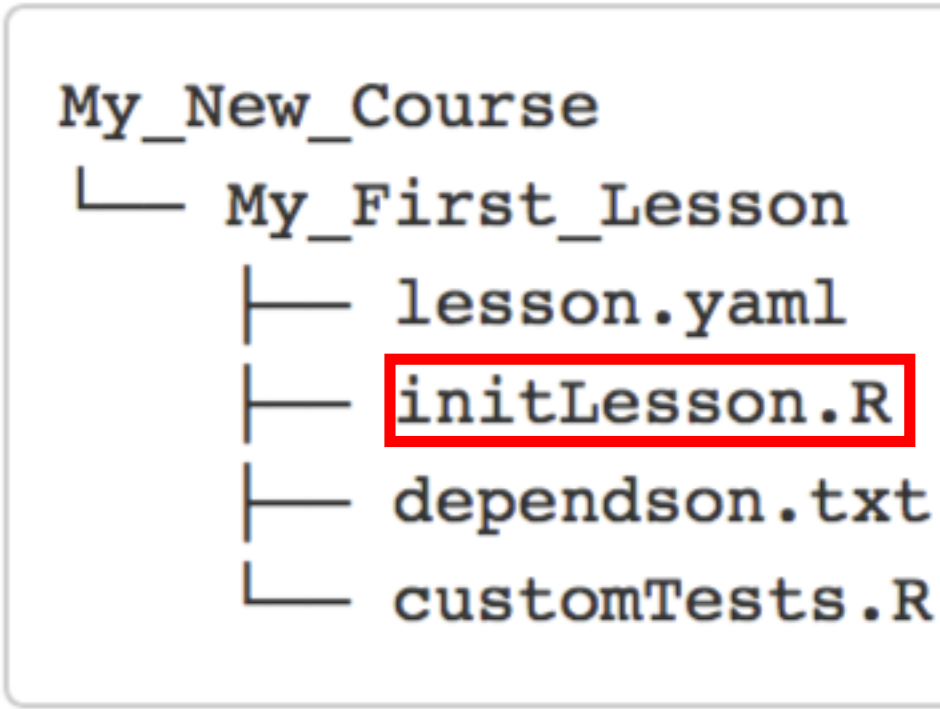
Each lesson directory inside of a course contains all of the files necessary to execute a specific lesson. Lessons cover specific topics that fall within the course theme.

Swirl course structure:

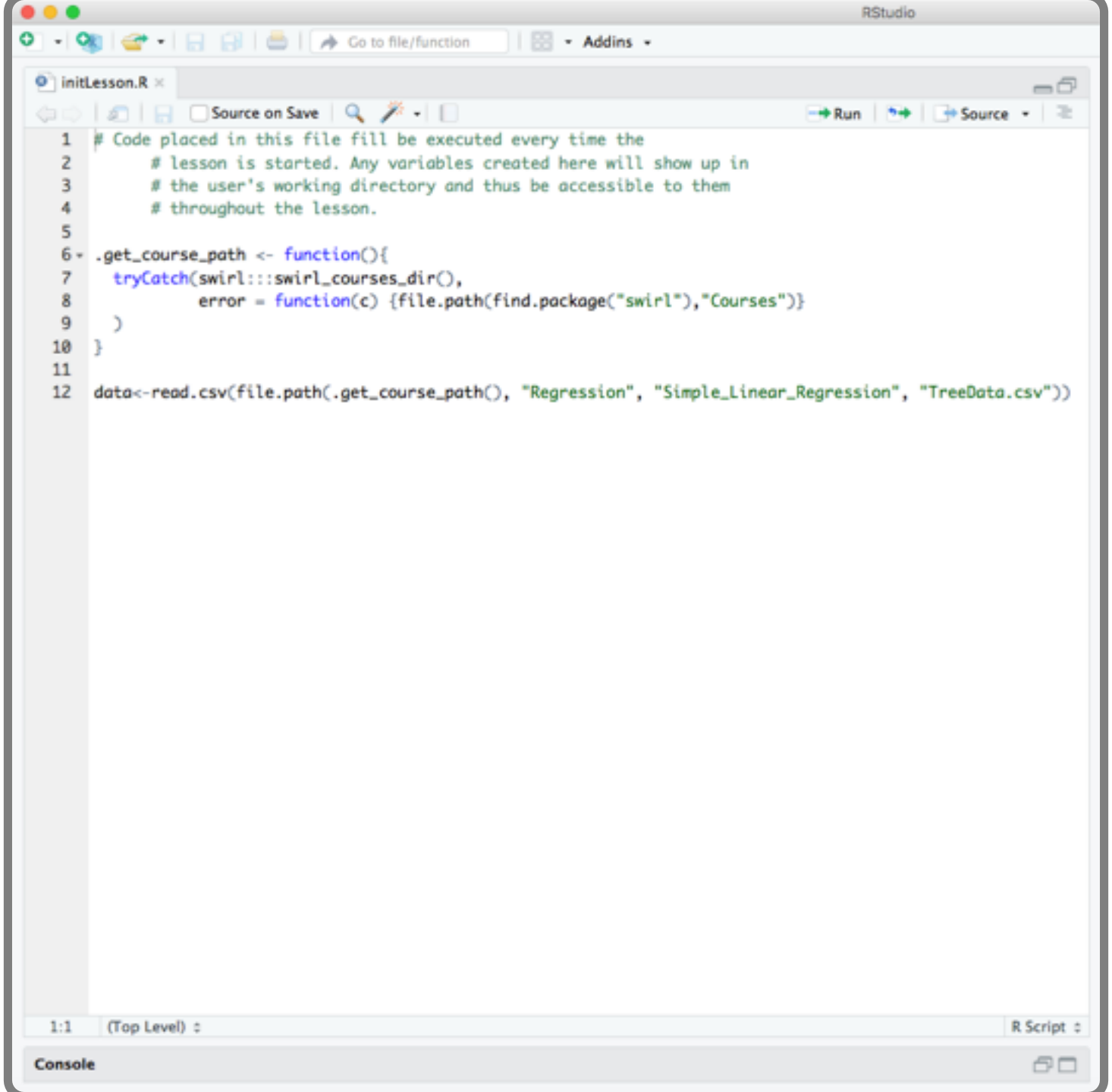



/Library/Frameworks/R.framework/Versions/3.4/Resources/library/swirlify/swirlify-app - Shiny
http://127.0.0.1:5292
Open in Browser
Publish
swirlify 0.5
Editor
Options
Help
Question Type
Message
Output
1 Type your text output here.
Add Question
1 - Class: meta
2 Course: Regression
3 Lesson: Simple Linear Regression
4 Author: Paige E. Parry, PhD
5 Type: Standard
6 Organization: George Fox University
7 Version: 2.4.3
8
9 - Class: text
10 Output: In this lesson, you will practice testing mult
11
12 - Class: text
13 Output: Regression models are used to quantify the rel
14
15 - Class: text
16 Output: Simple linear regression is our most basic moc
17
18 - Class: text
19 Output: Simple linear regression models are also usefu
20
21 - Class: text
22 Output: In the absence of other information, we often
23
24 - Class: cmd_question
25
Save Lesson
Demo Lesson
Question Number where Demo will Start
1

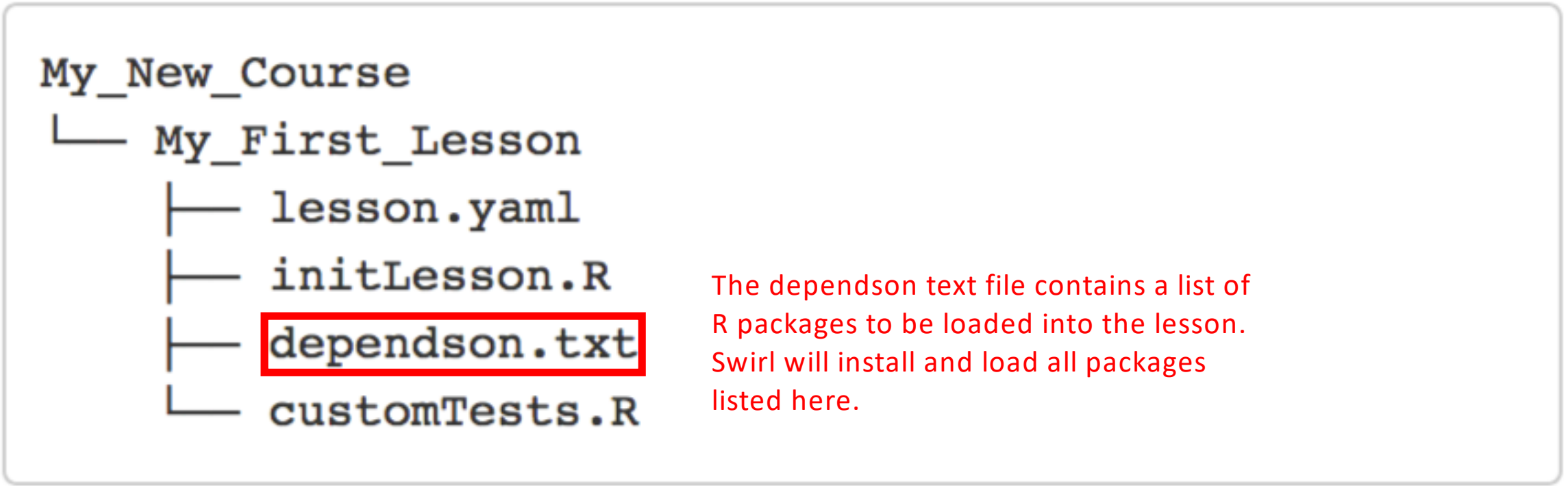
Swirl course structure:



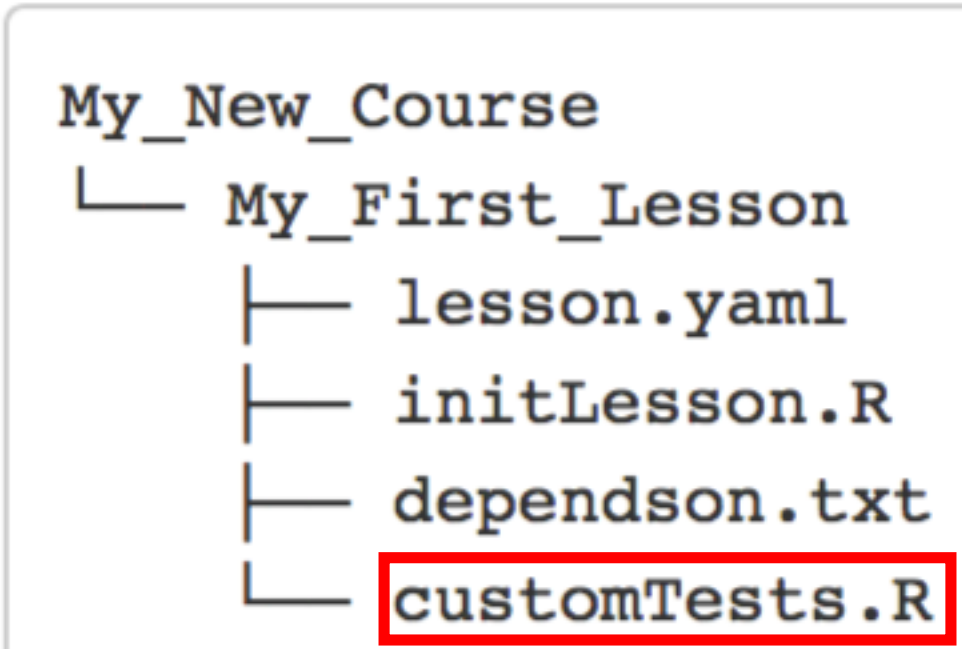
The initLesson file is an R script that runs each time the lesson is started and can be used to load environmental variables or data into the lesson.


RStudio
Go to file/function
Addins
initLesson.R
Source on Save
Run
Source
# Code placed in this file fill be executed every time the
# lesson is started. Any variables created here will show up in
# the user's working directory and thus be accessible to them
# throughout the lesson.
.get_course_path <- function(){
tryCatch(swirl:::swirl_courses_dir(),
error = function(c) {file.path(find.package("swirl"),"Courses")}
)
}
data<-read.csv(file.path(.get_course_path(), "Regression", "Simple_Linear_Regression", "TreeData.csv"))
1:1
(Top Level)
R Script
Console

Swirl course structure:



Swirl course structure:



This script can be used to write custom functions to test whether the answer to a swirl question is correct or not. Swirl already includes answer testing functionality, but some questions may require that you write custom tests.

Creating a new swirl lesson in swirlify:

Step 1: Install swirlify in RStudio

```
> install.packages("swirlify")
```

Step 2: Load swirlify

```
> library(swirlify)
```

Step 3: Set working directory to the directory in which you want to store your course

```
> setwd("your_directory_path_here")
```

Step 4: Create lesson and launch swirlify shiny app for new lesson in new course

```
> swirlify("My Lesson", "My Course")
```

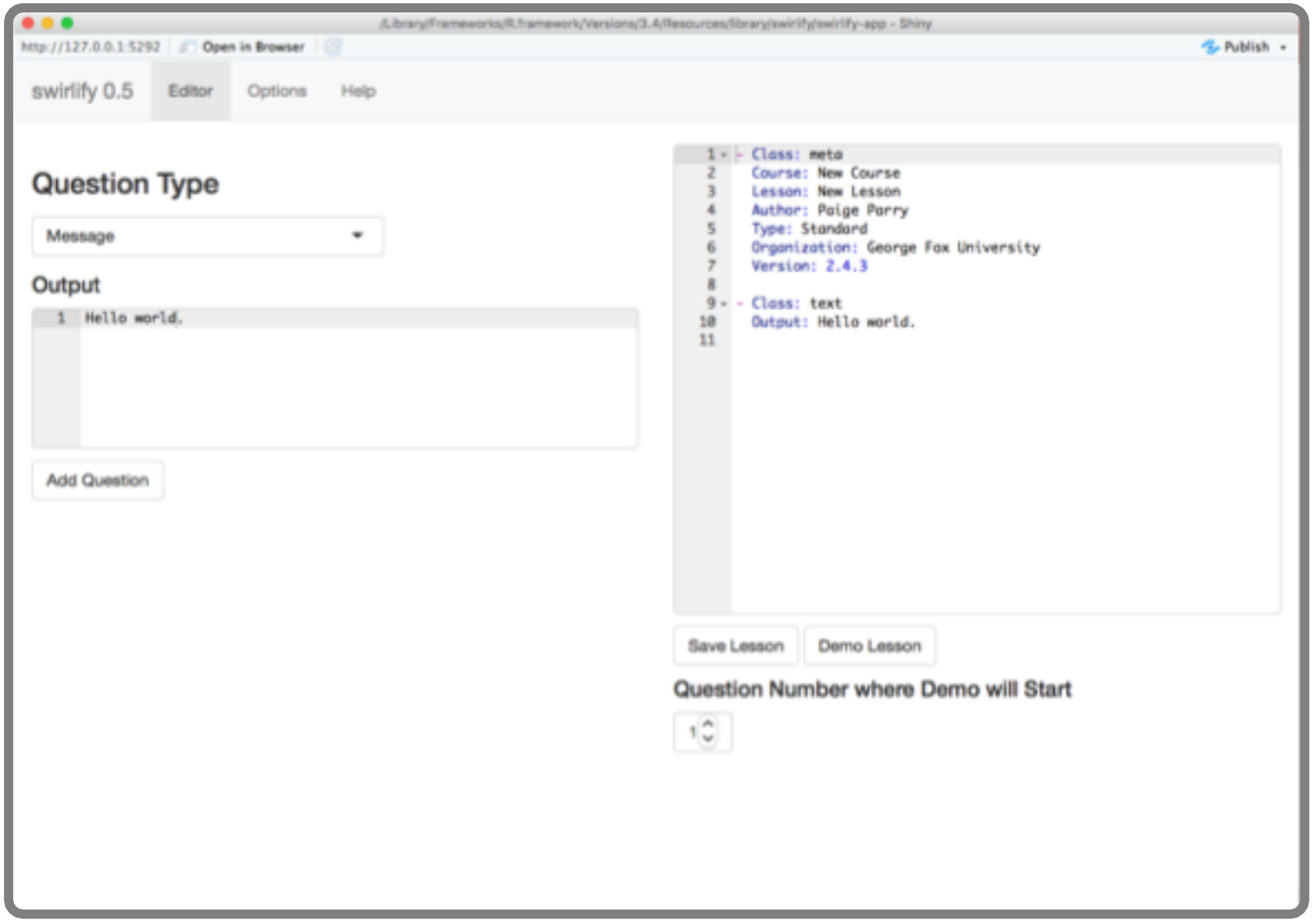
Question classes:

The meta question

```
1 - Class: meta
2   Course: New Course
3   Lesson: New Lesson
4   Author: Paige Parry
5   Type: Standard
6   Organization: George Fox University
7   Version: 2.4.3
8
```

Question classes:

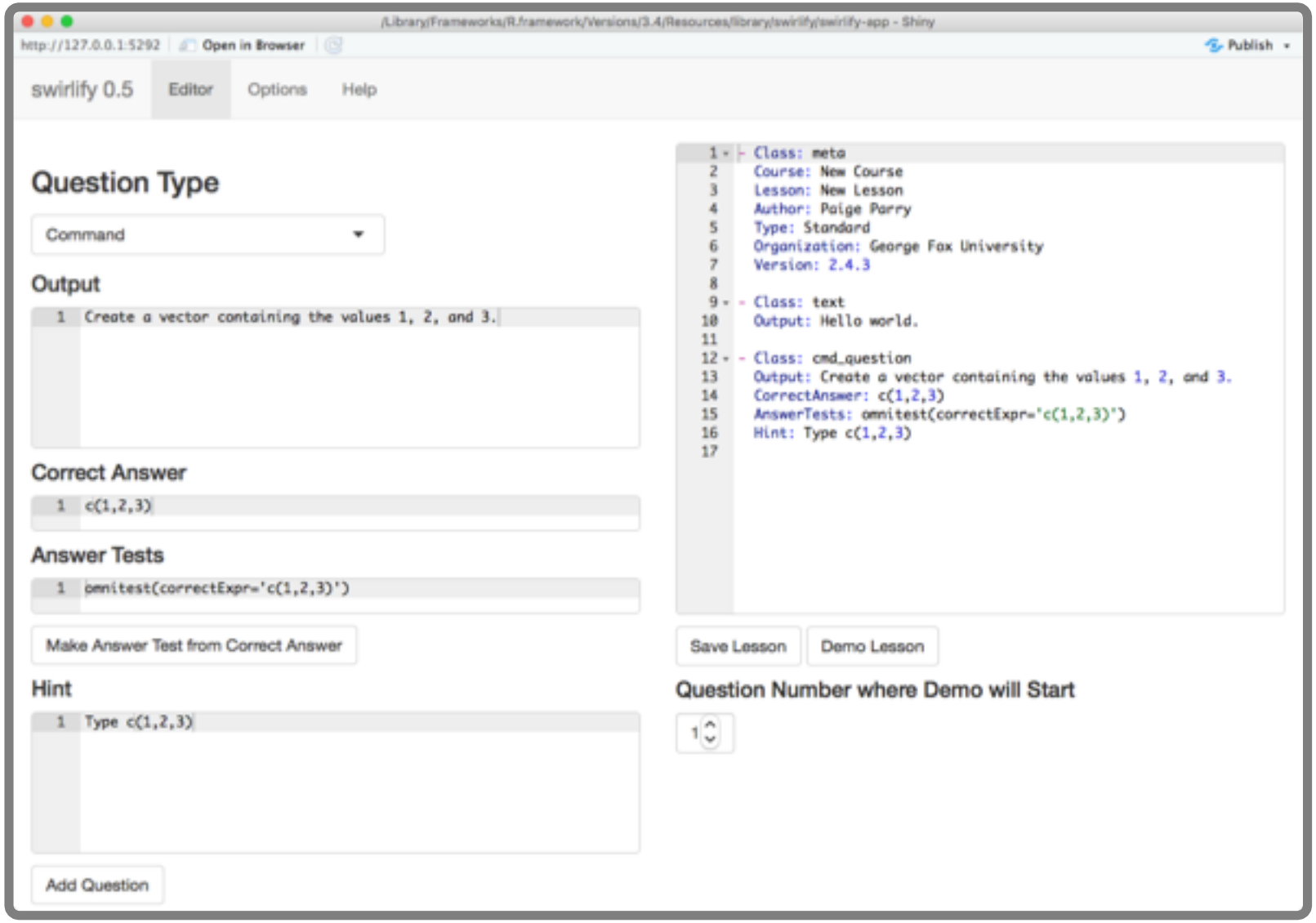
Message questions



**Tip: if you want to include apostrophes, quotations, or colons in your text, enclose the entire text string in quotations.*

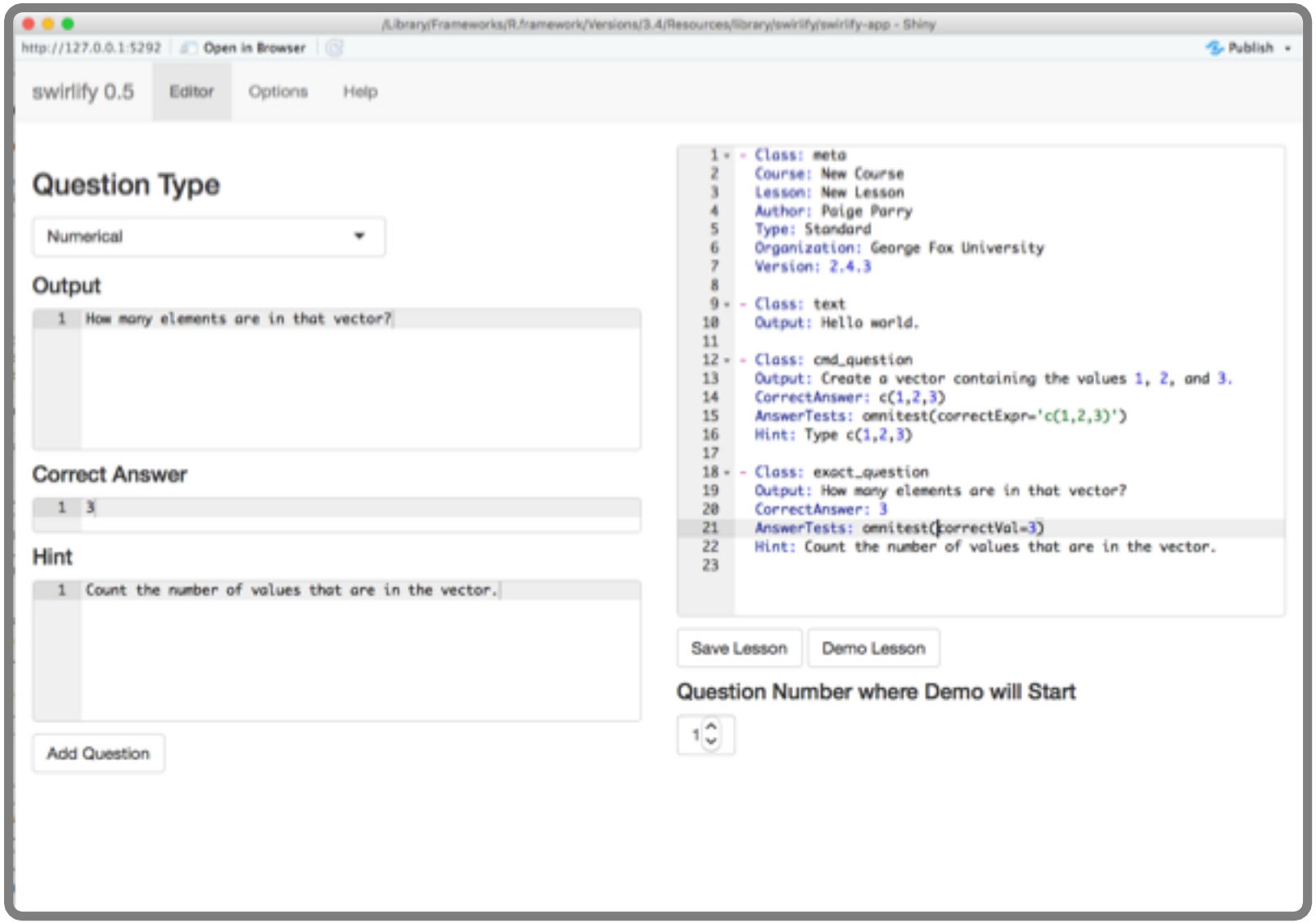
Question classes:

Command questions



Question classes:

Numerical questions



Check out the swirlify documentation for additional question types

http://swirlstats.com/swirlify/writing.html#types_of_questions

Including data in a lesson:

Step 1: Save data file (I recommend .csv) to same directory as lesson

Step 2: Open initLesson.R file associated with lesson

Step 3: Insert .get_course_path function

```
> .get_course_path <- function(){
      tryCatch(swirl:::swirl_courses_dir(),
      error=function(c) {file.path(find.package("swirl"),"Courses")}
      )
    }
```

Step 4: read in data file with .get_course_path function

```
> data <- read.csv(file.path(.get_course_path(), "My_Course",
      "My_Lesson", "data.csv"))
```

Finishing your lesson:

Step 1: Save your lesson out in the shiny app and close the app.

Step 2: Test the lesson in the Rstudio console. Running a test will check for syntax errors and print error messages to the console.

```
> test_lesson()
```

Step 3: Demo lesson to make sure that you are satisfied with what your students will experience.

```
> demo_lesson()
```

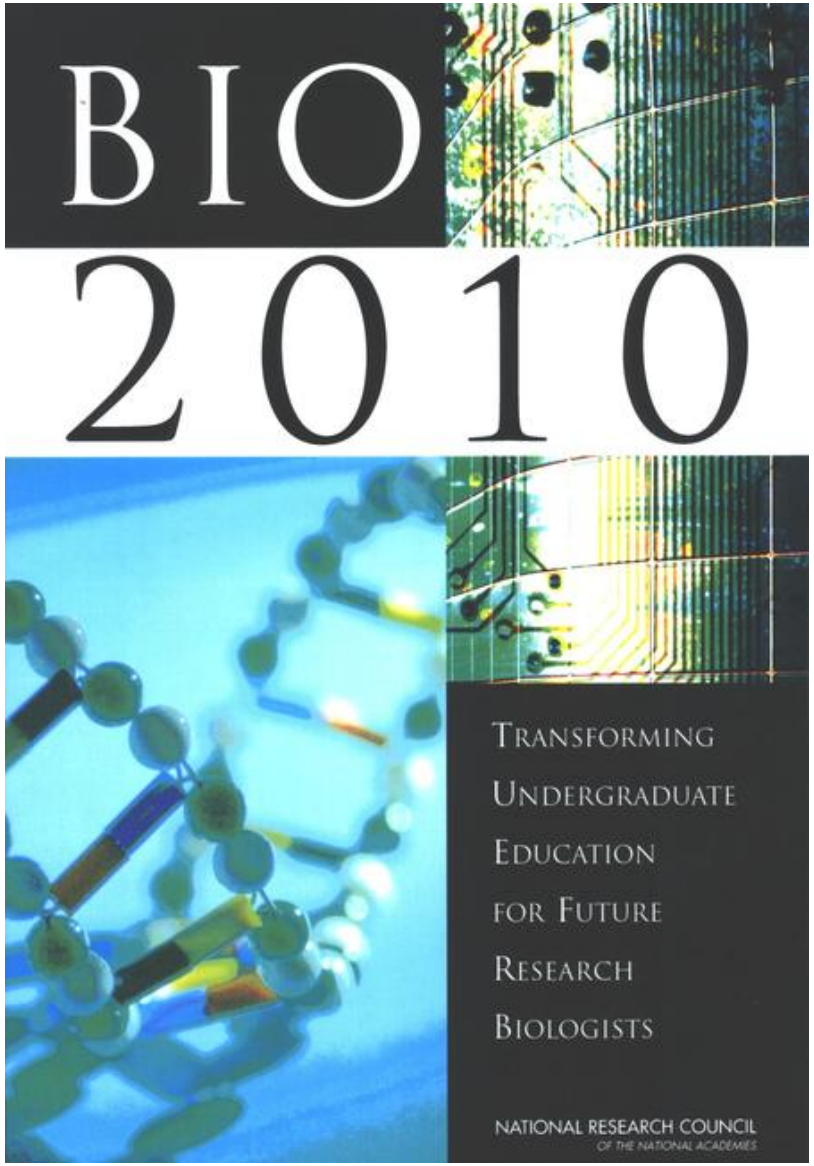
Step 4: When all lessons in a course are completed, pack course to a .swc file for sharing

```
> pack_course()
```

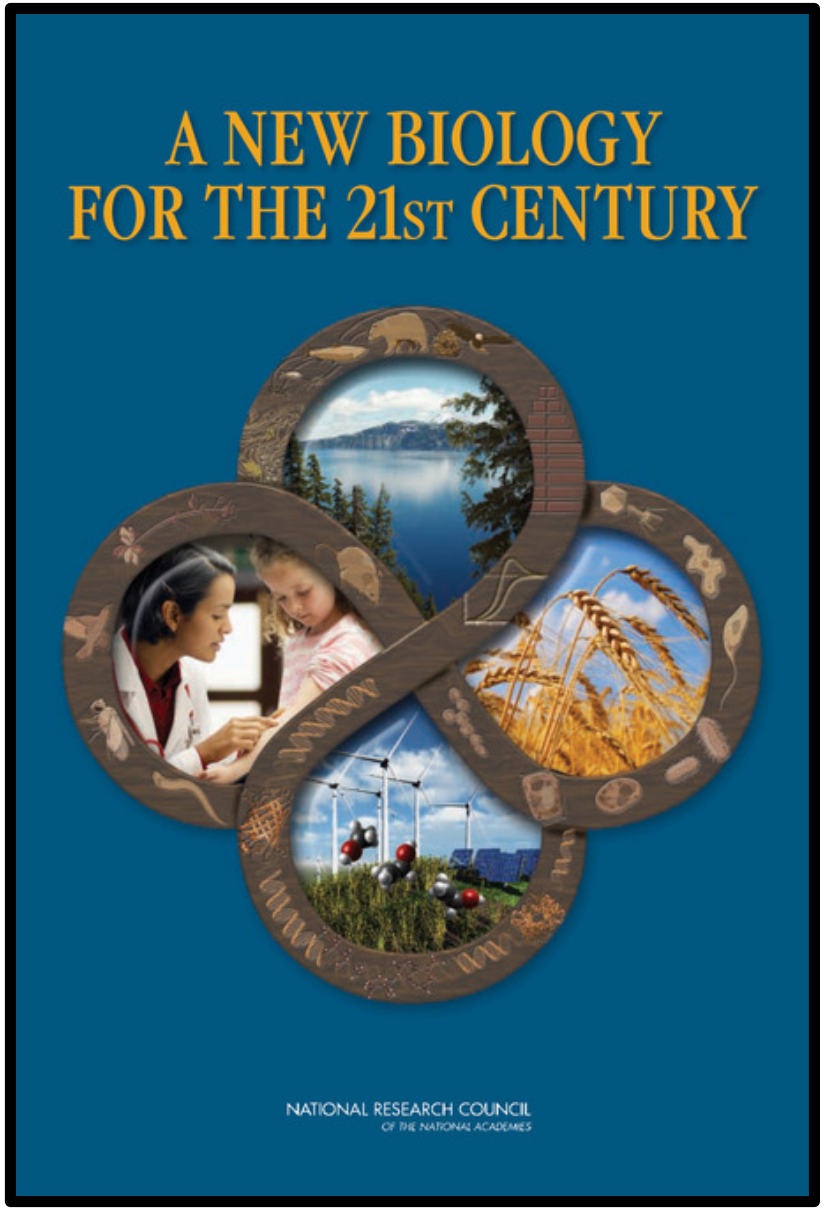
**For all of these functions to work properly, you must point swirlify to the lesson and course you want to test and pack by setting the working directory appropriately.*

Time to create your own lesson!

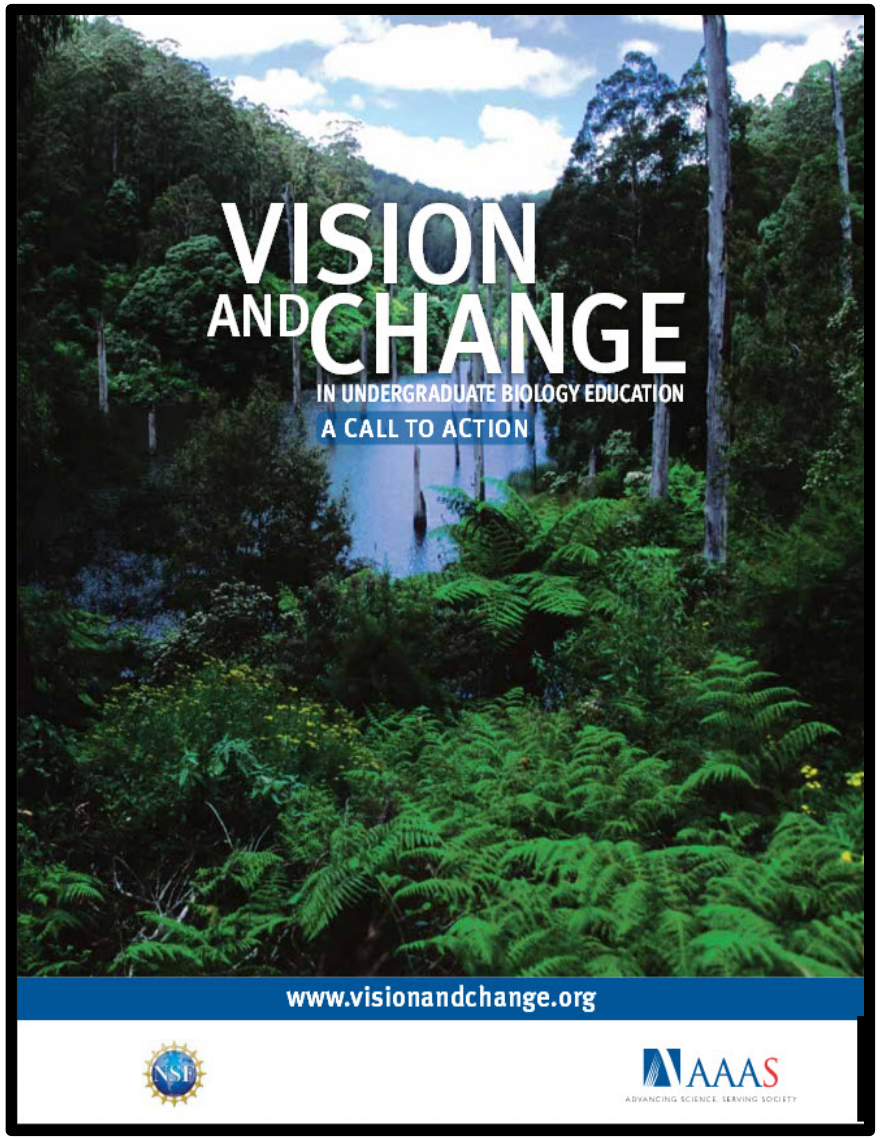
- Dataset 1: Survival and fitness of Atlantic salmon smolts
- Dataset 2: Comparing urban and forest soil characteristics
- Dataset 3: Oral contraceptive use and prostate cancer
- Dataset 4: Spread of RNA viruses specialized on cancer-derived vs non-cancerous cells



"Quantitative analysis, modeling, and prediction play increasingly significant day-to-day roles in today's biomedical research…life science majors [should] become sufficiently familiar with the elements of programming to carry out simulations of physiological, ecological, and evolutionary processes. They should be adept at using computers to acquire and process data, carry out statistical characterization of the data and perform statistical tests, and graphically display data in a variety of representations…it is essential that biology undergraduates become quantitatively literate."



"Quantitative analysis, physics, and chemistry are necessary to understand complex issues, along with biology…each institution of higher education [should] reexamine its current curricula and ensure that biology students gain a strong foundation in mathematics, physical and chemical sciences, and engineering as biology research becomes increasingly interdisciplinary."



"The application of quantitative approaches (statistics, quantitative analysis of dynamic systems, and mathematical modeling) is an increasingly important basic skill utilized in describing biological systems. Developing the ability to apply basic quantitative skills to biological problems should be required of all undergraduates, as they will be called on throughout their lives to interpret and act on quantitative data from a variety of sources…Today, modeling is a standard tool for biologists, so basic skills in implementing computational algorithms for models are increasingly being incorporated into the undergraduate curriculum."

Data Science for Undergraduates: Opportunities and Options

PREPUBLICATION COPY—SUBJECT TO FURTHER EDITORIAL CORRECTION

DATA SCIENCE FOR UNDERGRADUATES: OPPORTUNITIES AND OPTIONS

Committee on Envisioning the Data Science Discipline: The Undergraduate Perspective

Computer Science and Telecommunications Board
Board on Mathematical Sciences and Analytics
Committee on Applied and Theoretical Statistics
Division on Engineering and Physical Sciences

Board on Science Education
Division of Behavioral and Social Sciences and Education

A Consensus Study Report of

The National Academies of
SCIENCES • ENGINEERING • MEDICINE

THE NATIONAL ACADEMIES PRESS
Washington, DC
www.nap.edu

"Data science is emerging as a field that is revolutionizing science and industries alike. Work across nearly all domains is becoming more data driven, affecting both the jobs that are available and the skills that are required. As more data and ways of analyzing them become available, more aspects of the economy, society, and daily life will become dependent on data. In future decades, all undergraduates will profit from a fundamental awareness of and competence in data science."

What are the characteristics of an accessible, useful language for teaching scientific computation and analysis?

1. The language must be simple to learn so that most of the instruction can be focused on data analysis and visualization.
2. The language must make clear the general programming concepts required to perform analyses (e.g. input/output should be straightforward and quick).
3. The language must offer basic operators relevant to scientists (e.g. integration of programming language and graphics tools).
4. The language must be general enough that topics of importance in computer science can be illustrated (e.g. functions, variables, arguments, values, recursions).

Kaplan 2004. Teaching computation to undergraduate scientists. SIGSCE 04:3-7.