EDS 221: Scientific Programming Essentials

Allison Horst

2021-03-18

Contents

1 **Scientific programming essentials for environmental data science** **5**
  1.1 Course introduction 6

2 **Setup** **7**

3 **Data types and structures** **9**

4 **Methods** **11**

5 **Iteration** **13**

6 **Conditionals** **15**

7 **Logicals** **17**

8 **Functions** **19**
  8.1 Function components 19
  8.2 Writing simple functions 20
  8.3 Nested functions 22
  8.4 Functions with iteration and conditionals 22
  8.5 Useful function features 22
  8.6 Testing functions 22
  8.7 Iterating functions 22
  8.8 Resources on building, testing, & documenting functions 22

9 **Tidy data** **23**

10 **Data wrangling & viz in the tidyverse** **25**

11 **Troubleshooting** **27**

Chapter 1

Scientific programming essentials for environmental data science

UC SANTA BARBARA
Bren School of Environmental Science & Management

Material disclaimer and use

This book was created by Allison Horst for EDS 221 (Scientific Programming Essentials) in the Master of Environmental Data Science program in the Bren School of Environmental Science and Management, UC Santa Barbara. It accompanies lecture, computational lab and discussion materials that may or may not be linked to throughout the book. This book is intended as a supplemental resource for some parts of the course. In other words, it is not intended as a standalone textbook.

Acknowledgement

I create my courses by standing on shoulders of many giants in R, data science, and education communities. The wealth and quality of open educational resources (OERs) in data science has made teaching in the field fun, innovative, and inspiring. I've tried to thoroughly credit resources that I have pulled from

and adapted for this book, and I welcome additions if I have missed any that should be included. Thanks, friends.

1.1 Course introduction

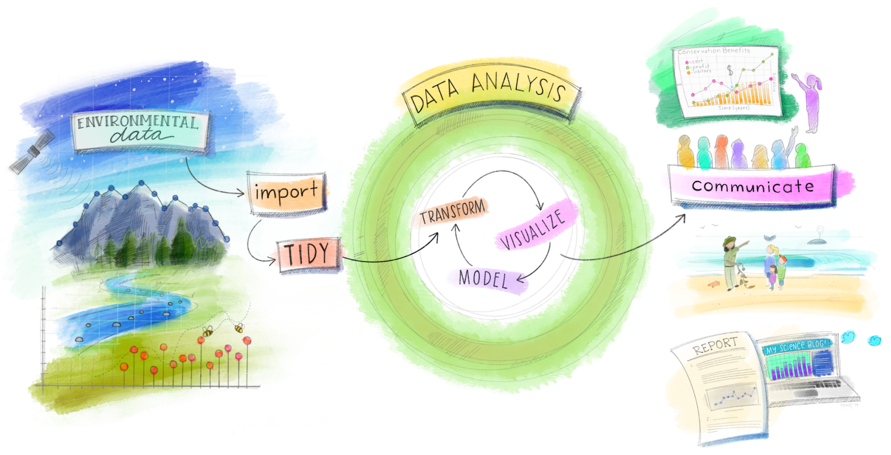

Figure 1.1: Slide from Dr. Julia Lowndes' 2019 keynote talk at useR conference (illustration by Allison Horst).

As nicely summarized in the title of a 2018 NCEAS post: *"The next generation of environmental scientists are data scientists"*.

Over the next year in MEDS you'll build skills to responsibly apply advanced methods in environmental modeling, spatial data analysis, and machine learning to investigate and solve environmental problems.

To get there, however, you'll need a strong foundation in programming basics like: understanding types and structures of data, basic data wrangling and visualization, algorithm development with functions, loops, and conditionals, and how to troubleshoot. While working in the weeds of programming, we'll also learn and reinforce transferable habits for reproducible workflows, robust file paths, version control, data organization, and more.

Upon these building blocks established in EDS 221, you'll be able to incrementally grow your advanced environmental data science toolkit to enter the workplace at the leading edge of quantitative methods in the field.

Chapter 2

Setup

Intro to programming and the tools we're using.

Chapter 3

Data types and structures

Data structures info...

Chapter 4

Methods

We describe our methods in this chapter.

Chapter 5

Iteration

Iteration

Chapter 6

Conditionals

Chapter 7

Logicals

Chapter 8

Functions

Writing functions to implement algorithms is a fundamental skill for every environmental data scientist. Functions can reduce repetition, increase efficiency and elegance, and facilitate reuse and sharing. Functions built by other developers will be ingrained into your code, but it's also critical that you can build, test, document, and share **your own** functions.

This chapter covers:

- Function structure
- Writing basic functions
- Nested functions
- Functions with iteration and conditions
- Useful function features
- Testing
- Documentation
- Applied examples

8.1 Function components

At the most basic level, a function takes an input, does something to it (a calculation, transformation, etc.), and returns an output.

For example, we can write a function that doubles the input value. In *function notation* seen in math, that would be:

$$f(x) = 2x$$

where x is the input, and $f(x)$ is the output. The function f acts on input x by doubling the input value.

How can we create a function to do the same thing in R? An R function would look like this:

```
double_it <- function(x) {
  2*x
}
```

What are these different pieces of that function?

- `double_it`: this assigns the function a name which we'll need to use it later on.
- `function(x)`: says we're making a function, and defines the function **formals** (arguments / parameters). This function expects a single input argument, `x` (you can check what the formals are using `formals(function_name)`).
- `{ 2*x }`: the **body** of the function, where we tell it what to do with the inputs. Note the braces (i.e. squiggly brackets).

Try out the function by inputting both a single value, and a vector of values. Note that vectorization is the default - meaning that the function is applied to each element in a vector.

```
double_it(x = 20)
```

```
## [1] 40
```

```
vec <- c(2, 4, 50) # Create a vector with multiple values

double_it(vec) # Function acts on each element in the vector
```

```
## [1]   4   8 100
```

Those are the main pieces. But don't worry, it gets a lot more interesting. Let's start by writing a few of our own functions.

8.2 Writing simple functions

Let's practice writing a few simple functions using established relationships in environmental science.

8.2.1 Example 1: Fish standard weight

"Standard weight" is how much we *expect* a fish to weigh, give the species and fish length, and the nonlinear relationship is given by:

$$W = aL^b$$

where L is total fish length (centimeters), W is the expected fish weight (grams), and a and b are species-dependent parameter values.

Write a function to calculate fish weight based on a, b, and fish length, then estimate the weight of several fish based on the following parameter estimates for Hawaiian fish from Peyton et al. (2015):

- *Chanos chanos* (milkfish): $a = 0.0905$, $b = 2.52$
- *Sphyraena barracuda* (great barracuda): $a = 0.0181$, $b = 3.27$
- *Caranx ignobilis* (giant trevally): $a = 0.0353$, $b = 3.05$

Function:

```r
predict_weight <- function(a, length, b) {
  a*(length^b)
}
```

Using the function:

1. Estimate the mass of a 160 cm long great barracuda.
2. Estimate the mass of a 118 cm long milkfish.

Thinking ahead: how might we make this function simpler for a user? For example, maybe a user can input the *species*, and the parameters a and b can be correctly updated for that species? We'll learn how to add this kind of functionality in upcoming sections.

8.2.2 Example 2:

8.2.3 Example 3:

8.3 Nested functions

8.4 Functions with iteration and conditionals

8.5 Useful function features

8.6 Testing functions

8.7 Iterating functions

8.8 Resources on building, testing, & documenting functions

- Ch. 6 - Functions in *Advanced R* by Hadley Wickham

Chapter 9

Tidy data

Chapter 10

Data wrangling & viz in the tidyverse

Chapter 11

Troubleshooting