Name of Data Science Course

Learning Hub Team

2023-02-07

Table of contents

Preface

Course dates: January 30, 2023 - February 3, 2023

This is a test book built using Quarto.

About this Course

This 5-day in-person workshop will provide researchers with an overview of reproducible and ethical research practices, steps and methods for more easily documenting and preserving their data at the Arctic Data Center, and an introduction to programming in R. Special attention will be paid to qualitative data management, including practices working with sensitive data. Example datasets will draw from natural and social sciences, and methods for conducting reproducible research will be discussed in the context of both qualitative and quantitative data. Responsible and reproducible data management practices will be discussed as they apply to all aspects of the data life cycle. This includes ethical data collection and data sharing, data sovereignty, and the CARE principles. The CARE principles are guidelines that help ensure open data practices (like the FAIR principles) appropriately engage with Indigenous Peoples' rights and interests.

Schedule

Code of Conduct

Please note that by participating in this activity you agree to abide by the NCEAS Code of Conduct.

Acknowledgements

These written materials reflect the continuous development of learning materials at the Arctic Data Center and NCEAS to support individuals to understand, adopt, and apply ethical open science practices. In bringing these materials together we recognize that many individuals have contributed to their development. The primary authors are listed alphabetically in the

citation below, with additional contributors recognized for their role in developing previous iterations of these or similar materials.

Citation: Amber E. Budden, S. Jeanette Clark, Natasha Haycock-Chavez, Noor Johnson, Matthew B. Jones. 2022. Fundamentals in Data Management for Qualitative and Quantitative Arctic Research.

Additional contributors: Stephanie Hampton, Jim Regetz, Bryce Mecum, Julien Brun, Julie Lowndes, Erin McLean, Andrew Barrett, David LeBauer, Jessica Guo.

License

Part I

Welcome and Overview

Objectives

- The mission and structure of the Arctic Data Center
- How the Arctic Data Center supports the research community
- About data policies from the NSF Arctic program

Arctic Data Center Overview

The Arctic Data Center is the primary data and software repository for the Arctic section of National Science Foundation's Office of Polar Programs (NSF OPP).

We're best known in the research community as a data archive – researchers upload their data to preserve it for the future and make it available for re-use. This isn't the end of that data's life, though. These data can then be downloaded for different analyses or synthesis projects. In addition to being a data discovery portal, we also offer top-notch tools, support services, and training opportunities. We also provide data rescue services.

1 R/RStudio and Git/GitHub Setup

1.1 Learning Objectives

In this lesson, you will learn:

- How to check to make sure your RStudio environment is set up properly for analysis
- How to set up git

1.2 Logging into the RStudio server

To help prevent us from spending most of this lesson remotely troubleshooting the myriad of issues that can arise when setting up the R, RStudio, and git environments, we have chosen to have everyone work on a remote server with all of the software you need installed. We will be using a special kind of RStudio just for servers called, aptly, RStudio Server. If you have never worked on a remove server before, you can think of it like working on a different computer via the internet. Note that the server has no knowledge of the files on your local filesystem, but it is easy to transfer files from the server to your local computer, and vice-versa, using the RStudio server interface.

Here are the instructions for logging in and getting set up:

2 Introduction to R

2.1 Learning Objectives

- get oriented to the RStudio interface
- work with R in the console
- be introduced to built-in R functions
- learn to use the help pages

2.2 What is R?

There is a vibrant community out there that is collectively developing increasingly easy to use and powerful open source programming tools. The changing landscape of programming is making learning how to code easier than it ever has been. Incorporating programming into analysis workflows not only makes science more efficient, but also more computationally reproducible. In this course, we will use the programming language R, and the accompanying integrated development environment (IDE) RStudio. R is a great language to learn for data-oriented programming because it is widely adopted, user-friendly, and (most importantly) open source!

So what is the difference between R and RStudio? Here is an analogy to start us off. **If you were a chef, R is a knife.** You have food to prepare, and the knife is one of the tools that you'll use to accomplish your task.

And **if R were a knife, RStudio is the kitchen**. RStudio provides a place to do your work! Other tools, communication, community, it makes your life as a chef easier. RStudio makes your life as a researcher easier by bringing together other tools you need to do your work efficiently - like a file browser, data viewer, help pages, terminal, community, support, the list goes on. So it's not just the infrastructure (the user interface or IDE), although it is a great way to learn and interact with your variables, files, and interact directly with git. It's also data science philosophy, R packages, community, and more. So although you can prepare food without a kitchen and we could learn R without RStudio, that's not what we're going to do. We are going to take advantage of the great RStudio support, and learn R and RStudio together.

Something else to start us off is to mention that you are learning a new language here. It's an ongoing process, it takes time, you'll make mistakes, it can be frustrating, but it will be overwhelmingly awesome in the long run. We all speak at least one language; it's a similar process, really. And no matter how fluent you are, you'll always be learning, you'll be trying things in new contexts, learning words that mean the same as others, etc, just like everybody else. And just like any form of communication, there will be miscommunications that can be frustrating, but hands down we are all better off because of it.

While language is a familiar concept, programming languages are in a different context from spoken languages, but you will get to know this context with time. For example: you have a concept that there is a first meal of the day, and there is a name for that: in English it's "breakfast." So if you're learning Spanish, you could expect there is a word for this concept of a first meal. (And you'd be right: 'desayuno'). **We will get you to expect that programming languages also have words (called functions in R) for concepts as well**. You'll soon expect that there is a way to order values numerically. Or alphabetically. Or search for patterns in text. Or calculate the median. Or reorganize columns to rows. Or subset exactly what you want. We will get you increase your expectations and learn to ask and find what you're looking for.

Part II

Next Part

Objectives

- The mission and structure of the Arctic Data Center
- How the Arctic Data Center supports the research community
- About data policies from the NSF Arctic program

Arctic Data Center Overview

The Arctic Data Center is the primary data and software repository for the Arctic section of National Science Foundation's Office of Polar Programs (NSF OPP).

We're best known in the research community as a data archive – researchers upload their data to preserve it for the future and make it available for re-use. This isn't the end of that data's life, though. These data can then be downloaded for different analyses or synthesis projects. In addition to being a data discovery portal, we also offer top-notch tools, support services, and training opportunities. We also provide data rescue services.

3 Intro to RMarkdown

3.1 Learning Objectives

- explore an example of RMarkdown as literate analysis
- learn markdown syntax
- write and run R code in RMarkdown
- build and knit an example document

3.2 What's so great about literate analysis and markdown?

The concept of literate analysis dates to a 1984 article by Donald Knuth. In this article, Knuth proposes a reversal of the programming paradigm.

If our aim is to make scientific research more transparent, the appeal of this paradigm reversal is immediately apparent. All too often, computational methods are written in such a way as to be borderline incomprehensible - even to the person who originally wrote the code! The reason for this is obvious, computers interpret information very differently than people do. By switching to a literate analysis model, you help enable human understanding of what the computer is doing. As Knuth describes, in the literate analysis model, the author is an "essayist" who chooses variable names carefully, explains what they mean, and introduces concepts in the analysis in a way that facilitates understanding.

RMarkdown is an excellent way to generate literate analysis, and a reproducible workflow. RMarkdown is a combination of two things - R, the programming language, and markdown, a set of text formatting directives. In R, the language assumes that you are writing R code, unless you specify that you are writing prose (using a comment, designated by `#`). The paradigm shift of literate analysis comes in the switch to RMarkdown, where instead of assuming you are writing code, Rmarkdown assumes that you are writing prose unless you specify that you are writing code. This, along with the formatting provided by markdown, encourages the "essayist" to write understandable prose to accompany the code that explains to the human-beings reading the document what the author told the computer to do. This is in contrast to writing just R code, where the author telling to the computer what to do with maybe a smattering of terse comments explaining the code to a reader.

Before we dive in deeper, let's look at an example of what literate analysis with RMarkdown can look like using a real example. **Here** is an example of a real analysis workflow written using RMarkdown.

There are a few things to notice about this document, which assembles a set of similar data sources on salmon brood tables with different formatting into a single data source.

- It introduces the data sources using in-line images, links, interactive tables, and interactive maps.
- An example of data formatting from one source using R is shown.
- The document executes a set of formatting scripts in a directory to generate a single merged file.
- Some simple quality checks are performed (and their output shown) on the merged data.
- Simple analysis and plots are shown.

In addition to achieving literate analysis, this document also represents a **reproducible analysis.** Because the entire merging and quality control of the data is done using the R code in the RMarkdown, if a new data source and formatting script are added, the document can be run all at once with a single click to re-generate the quality control, plots, and analysis of the updated data.

RMarkdown is an amazing tool to use for collaborative research, so we will spend some time learning it well now, and use it through the rest of the course.

4 Writing Data Management Plans

- Why create data management plans
- The major components of data management plans
- Tools that can help create a data management plan
- Features and functionality of the DMPTool

4.1 When to plan: The Data Life Cycle

Shown below is one version of the Data Life Cycle that was developed by DataONE. The data life cycle provides a high level overview of the stages involved in successful management and preservation of data for use and reuse. Multiple versions of a data life cycle exist with differences attributable to variation in practices across domains or communities. It is not neccesary for researchers to move through the data life cycle in a cylical fashion and some research activities might use only part of the life cycle. For instance, a project involving meta-analysis might focus on the Discover, Integrate, and Analyze steps, while a project focused on primary data collection and analysis might bypass the Discover and Integrate steps. However, 'Plan' is at the top of the data life cycle as it is advisable to initiate your data management planning at the beginning of your research process, before any data has been collected.

References