Lab 10A - Provenance

Sydne Record

November 26, 2019

**Learning goals:**

To learn how to collect and interpret data provenance in R. To this end we will:

1. become familiar with RDataTracker,
2. and generate provenance of an R script.

5 pts of this assignment will be allotted for submitting a word file knitted by RMarkdown.

**Getting started**  
The RDataTracker tools have not yet been formally compiled on the CRAN server. They are available on GitHub here: <https://github.com/End-to-end-provenance>. For this lab, you will need to install the following packages from the End-to-end-provenance GitHub page using the devtools command. First, make sure that you have the devtools package installed.

install.packages('devtools')  
library(devtools)

You will need to install the following packages from GitHub: rdtLite, provParseR, provSummarizeR, and provExplainR. Here is some example code to get you started on installing packages from GitHub using devtools:

devtools::install\_github("End-to-end-provenance/rtdLite")

You will need to run a similar line of code for each of the packages you need to download from GitHub. Also, don’t forget to use the library() function to load the packages into your R session after installing the packages, so that you are able to access their functionality. Finally, if you are having download issues on your own machine, all of these packages are available on the Bryn Mawr R Studio server (<https://rstudio.brynmawr.edu/>).

\*\* An example brought to you by BIOL330 - Ecological Modeling A couple weeks ago in the Ecological Modeling class, students were gathering data on species occurences to fit species distribution models. There are a variety of sources for locality data (i.e., species occurrences) - some are databases that often overlap in what is available data, whereas others are unique. For example, locality information associated with specimens in natural history museums are largely available in VertNet [(http://vertnet.org/)](http://vertnet.org/) and the Global Biodiverity Information Facility (GBIF, <http://www.gbif.org/>). Here is the prompt that I gave the students for using the spocc package in R to download occurence data from Vertnet:

For this assignment lab we will focus on locality data for Least Bell’s Vireo (*Molothrus ater obscurus*) in San Diego, California from VertNet. Use the *occ()* function from the spocc package and name the object containing the occurrence data *bell*. Hint: check out the help on vertnetops within the *occ* function documentation.

Although students were all given the same prompt, they came up with very different data sets with vastly different numbers of rows. In the Moodle folder for this lab, I give you code examples for two different approaches students took in files names ‘bell.R’ and ‘bell2.R’. Download both of these files and put them into your working directory.

# provExplainR - How do two scripts differ?

One the files that I supplied to you (bell.R) has two errors. Before fixing the errors, run some provenance collection on the bell2.R script. First, create a new folder in your working directory to write the provenane outputs to using the code below.

dir.create(provbella)

Next you want to collect the provenance with the code below:

prov.run("bell.R") # Collect provenance  
prov.quit() # Stop collecting provenance

You will note that a new file has appeared in the folder called ‘provbellb’ that you created in your working directory.

## Question 1: Correct the two errors in the ‘bell.R’ script and fix it. Also, add a line of code that writes the bell2sd object to a file in your working directory called ‘bell.csv’. Save your updated script with the same name (i.e., bell.R). (10 pts)

## Question 2: Now create a new folder in your working directory called ‘provbellb’. Run the provenance collection again on the updated version of the bell.R script that you just saved. This should output a new provenance file into the provbellb folder you just made. (10 pts)

Now we can use the provExplainR package to see how the two scripts differ.

## Question 3: Use the prov.diff.script function to see how the bell.R script changed over the course of this lab. (10 pts)

The prov.diff.script function may be helpful in your collaborative group project if you are trying to figure out how a script differs as you pass it between partners on the project, or if you are simply trying to remember how you have altered a script yourself. This is starting to get at a term called version control in which you try to keep track of how a script changes through time. We will talk in class about GitHub, which is another platform for tracking versions of scripts.

# provSummarize

Another really useful tool provided by RDataTracker is the prov.summarize.run() function of the provSummarizeR package.

## Question 4: Write some code to execute the prov.summarize.run() function on the ‘bell2.R’ script and write the provenance outputs to a text file in your working directory. Include this text file with your lab submission (you will be able to upload multiple files on Moodle).

## Question 5: How do the objects bell and bell2sd differ in their dimensions (provide any code used to generate your answers)? Isn’t it crazy that the same prompt with the same function could give such different results? (5 pts)

Hopefully, this lab gives you an appreciation for how tricky it can be to have reproducible code even with the best of intentions.