Licensing: Attribution-NonCommercial-NoDerivatives 4.0 International (<https://creativecommons.org/licenses/by-nc-nd/4.0/>)



Matt Watts, Kristal Spreadborough, R Support Group presentation, 12 July 2019, The University of New England

# Crossref metadata analysis

SLIDE 1

Hello and thank you for coming along today. In today’s presentation, Matt and I will be discussing how to conduct metadata analysis using R. For those of you who are new to this kind of analysis, we’ll first give you a bit of context about metadata analysis in general. Then we’ll talk about the specific project that gave rise to this presentation. Next, we’ll open R and walk through some code together. Finally, we’ll do some activities and open up for general discussion.

## Overview and context

### What is metadata analysis?

SLIDE 2

When we talk about metadata, we’re referring to the stuff that describes a thing. List of authors, title, and abstract are all examples of metadata for a publication.

Metadata is incredibly valuable.

SLIDE 2 - click

When you visit the library website and search for a book using Primo, the Library search interface, you are searching metadata.

SLIDE 2 - click

When you use Google Scholar and search for an article, you are using metadata.

SLIDE 2- click

When you define a set of search terms to complete your literature review, you are using metadata. If you’ve ever interacted with a database or a search engine like the ones I’ve just mentioned, then you’ve actually already engaged in a kind of metadata analysis.

Metadata underpins a great deal of our research. Search interfaces allow us to easily interact with metadata on the common databases. But we can interact with metadata in much more sophisticated ways.

SLIDE 3

You see, metadata does not exist exclusively in online databases. In most cases, we’ll talk about a few in a moment, metadata can be extracted from a database and downloaded to your computer. Once it is downloaded to your computer, you can use a programing language such as R to analyse it in any way you wish. For example, you could find every paper that used a specific phrase in their abstract. You could find the exact number of times that a key word occurs. You could find which research themes are more common than others, and identify what might be emerging research themes.

In summary, the chances are you are already engaging in metadata analysis in some way, today we want to show you how you can take your analysis skills to the next level using R.

### Getting the metadata?

SLIDE 4

Before you can analyse metadata using R, you first need to *get* the metadata. To do this, you need to identify your databases.

There are a number of databases at our disposal for extracting metedata. A list of databases that the library holds subscriptions for is available here: <http://une.au.libguides.com/az.php?>

SLIDE 4 - Click

Once you’ve selected your database, you need to extract the metadata. You can extract the metadata in two ways. Some databases will have an existing bulk download option. This means that the database provider has already compiled a bunch of metadata in files and all you have to do is download it. Depending on the kind of database you’re using, we’ll talk more about this in a moment, you may be able to access all the metadata in a database, or only a small portion of the metadata.

The second way you can extract metadata is through web scraping. This involves writing a script that will trawl through a database and pull down all the content you’re interested in. You can do this in R, Matt does it all the time. We might do a session on this method of data extraction later in the year.

SLIDE 5

Some databases are open access, which means that you can freely download and use the metadata. Many of the well known databases however, such as Scopus (<https://www.scopus.com/search/form.uri?display=basic>) and JStor (<https://www.jstor.org/>), are proprietary. This means they are owned by a company and we have to pay to access the database. You can still get the metadata you want from these sources, but there are several considerations:

1. Bulk downloads may be limited. For example, JStor will do a bulk download on your behalf, but only up to 20 000 records.
2. The APIs for commercial databases are not openly available.
3. While it may be possible, scraping of commercial databases may not be ethical. For example, Plos states that they do not encourage scraping of their database, though it is not explicitly prohibited.
4. We may get in trouble for scraping too much. This is a little ambiguous for some of the commercial API’s. This is an emerging area that some vendors may not have legally defined yet. We don’t want UNE to get in trouble with commercial vendors.

### Context to today’s presentation

SLIDE 6

We are currently completing a project at Research UNE which involves us gathering and analysing large quantities of metadata. One of the sources from which we pull metadata is Crossref (<https://www.crossref.org/>). There are a number of ways to pull data down from such sources. One way is through the API. Crossref has a REST API (<https://github.com/CrossRef/rest-api-doc>) which we could use to scrape metadata. However, this API only allows 100 000 records to be downloaded per day. Since we wanted to pull down all one hundred million records stored in Crossref, scraping the API was not a viable option. Intead, we used the bulk download option to bring down all the records in Corssref at once.

This was possible because Crossref is open access, which means that all the metadata available through Crossref is freely available to anyone.

We obtained our bulk download from this GitHub repository: <https://github.com/greenelab/crossref> . In the “Other resources” section, you can see the sources for bulk download. We used Brian Newbold’s September 2018 release of the bulk crossref-works data: <https://archive.org/download/crossref_doi_dump_201809>

crossref-works.2018-09-05.json.xz 33.2GB XZ archive.

## Method for data extraction

SLIDE 7

We extracted the xz archive in Linux with the “unxz” command like this:

|  |
| --- |
| unxz crossref-works.2018-09-05.json.xz |

Note that we backed up the xz file first, because “unxz” deletes the original file after uncompressing it.

The file is a JSON ascii file of over 300GB where each line is a publication record. We partitioned the JSON into chunks with 1 million lines in each chunk to make it more tractable:

|  |
| --- |
| split -l 1000000 crossref-works.2018-09-05.json.xz split |

This created 100 files with 1 million lines in each. The first file was splitaa.

We use R to read the JSON records from ascii and save them as compressed Rdata:

|  |
| --- |
| inData <- readLines("splitaa") save(**inData**,file="splitaa.Rdata") |

SLIDE 8

Here’s a script that compresses all the chunks so we can save a lot of time and disk space analysing and storing the data:

|  |
| --- |
| alphabet <- c("a","b","c","d","e","f",  "g","h","i","j","k","l",  "m","n","o","p","q","r",  "s","t","u","v","w","x",  "y","z") length(alphabet) # 26  inFiles <- c() for (i in 1:26) {  for (j in 1:26)  {  inFiles <- c(inFiles,paste0("split",alphabet[i],alphabet[j]))  } }  for (i in 1:length(inFiles)) {  # i <- 2  cat(i," ")  inFile <- inFiles[i]  cat(inFile," ")  if (file.exists(inFile))  {  inData <- readLines(inFile)  cat("read ")  save(inData,file=paste0(inFiles[i],".Rdata"))  cat("saved ")  }  cat("\n") } |

SLIDE 9

We provide you with the compressed Rdata for a single chunk so you can follow along with subsequent steps. You can download the sample file here:

<http://marxan.io/downloads/crossref_workshop/splitaa.Rdata> (220 megabytes)

The source code used with the sample file is here:

<http://marxan.io/downloads/crossref_workshop/crossref_workshop_11Jul2019.R>

In the sample code, you’ll need to change the directory we read data from and write to. You’ll also need to install the packages “rjson”, “sqldf”, and “data.table”.

PAUSE HERE TO GET EVERYONE SET UP WITH DIRECTORIES AND INSTALLING PACKAGES.

## Extract affiliation name from JSON data

SLIDE 10

We want to know which publication records are affiliated with UNE, so the first step is extracting affiliation name from the JSON for 100 million records. We can run this for 1 million records using the splitaa.Rdata provided:

PAUSE HERE TO ALLOW EVERYONE TIME TO RUN THIS CODE

## Query affiliation name and extract UNE affiliated JSON records

SLIDE 11

We run a series of queries against the affiliation names we extracted to identify which records are affiliated with UNE:

PAUSE HERE TO ALLOW EVERYONE TIME TO RUN THIS CODE

## Extract “title” for UNE affiliated JSON records

SLIDE 12

We extract a table of records for UNE affiliated records with some metadata elements common across records, including “title”.

PAUSE HERE TO ALLOW EVERYONE TIME TO RUN THIS CODE

# Activities

SLIDE 13

## Activity 1

Extract metadata for this DOI: 10.1177/0040517514545256

## Activity 2

Extract metadata for this subject: Ecology