rgbif: a package for working with species occurrence data in R

Scott Chamberlain*,[a]

[a] *University of California, Berkeley, CA, USA*

Abstract

1. xxx
2. xxx
3. xxx
4. xxxx

Introduction

Perhaps the most fundamental element in many fields of ecology is the inividual. How many individuals of each species in a given location forms the basis for many sub-fields of ecology and evolution. Some research questions necessitate collecting new data, while others can easily take advantage of existing data. In fact, some ecology fields are built largely on existing data, e.g., macro-ecology (Brown 1995; Beck *et al.* 2012).

Data on individuals, including which species, and where they're found, can be used for a large number of research questions. In addition, the pool of questions we can answer becomes much larger with more and better data. In addition to wide utility, this data is important for conservation. Biodiversity loss is one of the greatest challenges of our time (Pimm *et al.* 2014). Some have called this the sixth great mass extinction (Ceballos *et al.* 2015). Given this challenge there is a great need for data on specimen records, whether collected from live sightings in the field or specimens in museums.

There are many online services that collect and maintain specimen records. However, Global Biodiversity Information Facility (hereafter, GBIF, http://www.gbif.org/) is the largest collection of biodiversity records globally, currently with 580 million records, 1.6 million taxa, 15,000 datasets from 770 publishers (figures collected on 2015-10-04). Many large biodiversity warehouses such as iNaturalist, VertNet, and USGS's BISON all feed into GBIF.

Herein, we describe a library (rgbif (Chamberlain *et al.*)) for working with GBIF data in the R programming environment (R Core Team 2014). R is an extremely widely used language in academia,

*Corresponding author
Email address: `scott(at)ropensci.org` (Scott Chamberlain)

and in non-profit and private sectors. Importantly, R makes it easy to do all of the steps of the research process, including data management, data manipulation and cleaning, statistics, and vizualization. Thus, an R client for getting GBIF data is a powerful tool for reproducible research.

The rgbif package

The `rgbif` package is completely written in R, uses an MIT license to maximize use everywhere. `rgbif` is developed publicly on GitHub at https://github.com/ropensci/rgbif, where development versions of the package can be installed, and bugs and feature requests reported. Stable versions of `rgbif` can be installed from CRAN, the distribution network for R packages. `rgbif` is part of the rOpenSci project, a developer network making R software to facilitate reproducible research.

Package interface

`rgbif` is designed following the GBIF Application Programming Interface, or API. The GBIF API has four major components: registry, species names, occurrence data, and maps. We ignore maps in `rgbif` as it is concerned with generating maps for web applications. `rgbif` has a suite of functions dealing with each of registry, species names, and occurrence data - we'll go through each in turn describing design and example usage.

Registry

The GBIF reqistry API services are spread across four sets of functions:

- Datasets
- Installations
- Networks
- Nodes
- Organizations

Datasets

Search for datasets

```
res <- dataset_search(query = "oregon")
res$data$datasetTitle[1:10]
```

```
 [1] "SDNHM Birds Collection"
 [2] "CM Birds Collection"
 [3] "condoncollection"
 [4] "Taxonomy in Flux Checklist"
 [5] "Wool carder bees of the genus Anthidium in the Western Hemisphere"
 [6] "Bryophyte Collection - University of Washington Herbarium (WTU)"
 [7] "University of British Columbia Herbarium (UBC) - Bryophytes Collection"
 [8] "UWFC Ichthyology Collection"
 [9] "Lichen Collection - University of Washington Herbarium (WTU)"
[10] "UWBM Mammalogy Collection"
```

Get dataset metrics

```
res <- dataset_metrics(uuid='66dd0960-2d7d-46ee-a491-87b9adcfe7b1')
df <- data.frame(rank = names(res$countByRank),
                 count = unname(unlist(res$countByRank)))
knitr::kable(df)
```

rank	count
SPECIES	52452
GENUS	12930
VARIETY	4806
SUBSPECIES	4440
SERIES	1079
TRIBE	844
FAMILY	509
SUBTRIBE	327
SUBFAMILY	303
SUBGENUS	241

rank	count
FORM	239
SECTION	82
SUBVARIETY	4
KINGDOM	1

Networks, nodes, and installations

Here, we search for the first give GBIF networks, returning just the key and title fields.

```
networks(limit=10)$data$title
```

```
 [1] "GBIF Backbone Sources"
 [2] "Canadensys"
 [3] "Southwest Collections of Arthropods Network (SCAN)"
 [4] "VertNet"
 [5] "Dryad"
 [6] "GBIF Network"
 [7] "The Knowledge Network for Biocomplexity (KNB) "
 [8] "Online Zoological Collections of Australian Museums (OZCAM)"
 [9] "Catalogue of Life"
[10] "Ocean Biogeographic Information System (OBIS)"
```

Species

Occurrences

GBIF provides two ways to get occurrence data: through the `/occurrence/search` route (see `occ_search`), or via the `/occurrence/download` route (many functions, see below). `occ_search()` is the main funtion for the search route, and is more appropriate for smaller data, while `occ_download*()` functions are more appropriate for larger data requests.

Large is of course a subjective term. When you hit a "large dataset" will depend primarily on the size of the your data request. GBIF imposes for any given search a limit of 200,000 records in the search

service, after which point you can't download any more records for that search. However, you can download more records for different searches.

We think the search service is still quite useful for many people even given the 200,000 limit. For those that need more data, we have created a similar interface in the `download_*()` functions, that should be easy to use. Users should take note that using the download service has a few extra steps to get data into R, but is straight-forward.

Download API

The download API syntax is similar to the occurrence search API in that the same parameters are used, but the way in which the query is defined is different. For example, in the download API you can do greater than searches, whereas you can not do that in the occurrence search API. Thus, we can't make the query interace exactly the same for both search and download functions.

Using the download service can be as few as three steps.

Request data download given a query. Here, we xxxx

```
"xxx"
```

```
[1] "xxx"
```

You can check on when the download is ready using the functions `occ_download_list()` and `occ_download_meta()`. When it's ready use `occ_download_get()` to download the dataset to your computer.

```
(res <- occ_download_get("0000066-140928181241064", overwrite = TRUE))
```

```
<<gbif downloaded get>>
  Path: ./0000066-140928181241064.zip
  File size: 0.14 MB
```

What's printed out above is a very brief summary of what was downloaded, the path to the file, and its size (in human readable form).

Next, read the data in to R using the function `occ_download_import()`.

```
library("dplyr")
dat <- occ_download_import(res)
dat %>%
  select(gbifID, decimalLatitude, decimalLongitude)
```

```
       gbifID decimalLatitude decimalLongitude
1  657590544              NA               NA
2  657679551              NA               NA
3  657791316        37.70805        -118.4162
4  658180562              NA               NA
5  441881672              NA               NA
6  911596181              NA               NA
7   56454601              NA               NA
8  657848913              NA               NA
9  658187373              NA               NA
10 658279212        38.95917        -106.9892
..       ...             ...              ...
```

Search API

The search API is very similar to the download API, but is meant for smaller data acquisition jobs.

Conclusions and future directions

Acknowledgements

This project was supported in part by the Alfred P Sloan Foundation (Grant 2013-6-22).

Data Accessibility

All software, scripts and data used in this paper can be found in the permanent data archive Zenodo under the digital object identifier (DOI). This DOI corresponds to a snapshot of the GitHub repository at github.com/sckott/msrgbif.

References

Beck, J., Ballesteros-Mejia, L., Buchmann, C.M., Dengler, J., Fritz, S.A., Gruber, B., Hof, C., Jansen, F., Knapp, S., Kreft, H., Schneider, A.-K., Winter, M. & Dormann, C.F. (2012). Whats on the horizon for macroecology? *Ecography*, **35**, 673–683. Retrieved from http://dx.doi.org/10.1111/j.1600-0587.2012.07364.x

Brown, J.H. (1995). *Macroecology*. University of Chicago Press.

Ceballos, G., Ehrlich, P.R., Barnosky, A.D., Garcia, A., Pringle, R.M. & Palmer, T.M. (2015). Accelerated modern human-induced species losses: Entering the sixth mass extinction. *Science Advances*, **1**, e1400253–e1400253. Retrieved from http://dx.doi.org/10.1126/sciadv.1400253

Chamberlain, S., Ram, K., Barve, V. & Mcglinn, D. *Rgbif: Interface to the global 'biodiversity' information facility 'aPI'*. Retrieved from https://github.com/ropensci/rgbif

Pimm, S.L., Jenkins, C.N., Abell, R., Brooks, T.M., Gittleman, J.L., Joppa, L.N., Raven, P.H., Roberts, C.M. & Sexton, J.O. (2014). The biodiversity of species and their rates of extinction, distribution, and protection. *Science*, **344**, 1246752–1246752. Retrieved from http://dx.doi.org/10.1126/science.1246752

R Core Team. (2014). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. Retrieved from http://www.R-project.org/