Data gathering and (pre-)processing

Ian Ondo

2023-05-29

Contents

Package notes:
We need to install the following set of R packages to successfully run the codes in this vignette:

- **data.table** (Dowle and Srinivasan 2021)
- **rWCVP** (Brown et al. 2023)

Introduction

In this vignette, we will show how to compile and format occurrence data from various data sources available online, or as zipped (compressed) files using **UsefulPlants**. We will focus on two online databases:

- *Global Biodiversity Information Facility* (**GBIF**, *https://www.gbif.org*)
- *Botanical information and Ecology Network* (**BIEN version 4.1**, *https://bien.nceas.ucsb.edu/bien*)

Both have R packages facilitating the download of occurrence records directly from R. For example, the function *get_gbifid_* from the R package **taxize** (Scott Chamberlain and Eduard Szocs 2013) and function *occ_download* from the R package **rgbif** (Chamberlain and Boettiger 2017) can be used for matching species names and downloading occurrence records from GBIF respectively, while the function *BIEN_occurrence_species* from the R package **BIEN** (Maitner 2020)helps to extract occurrence records from the BIEN database.
To our knowledge, other databases such as **RAINBIO** (http://rainbio.cesab.org/), **speciesLink** (http://www.splink.org.br/), **BioTIME** (https://biotime.st-andrews.ac.uk/) or **Genesys** (https://www.genesys-pgr.org)

do not provide R packages to access their data, but snapshots of the entire database can be accessed, sometimes on request, and often shared as a (compressed) file. In this case the **data.table** (Dowle and Srinivasan 2021) is a very useful R package in terms of efficiency and speed to process these type of datasets.

I. Download a list of useful plants species names

Download the useful plants database used in Pironon and Ondo et al. 2023 here, and unzip it into a folder, e.g. *"usefulplant_data"*. Then, read in the data and filter as needed:

```
# load the data into memory
useful_plant_db <- data.table::fread(input="usefulplant_data/useful_plants_data_list.csv",
                                     h=TRUE,
                                     showProgress=FALSE)

useful_plant_db

# get the list of all useful plants names in the database
useful_plant_names <- useful_plant_db %>%
  dplyr::pull(binomial_acc_name) # extract binomial accepted names

# optionally filter e.g. edible plants
edible_plant_names <- useful_plant_db %>%
  dplyr::filter(HumanFood==1) %>%         # select edible plants
  dplyr::pull(binomial_acc_name)
```

II. Download occurrence data from GBIF

UsefulPlants has a function called `queryGBIF` that uses both *get_gbifid_* and *occ_download* to obtain occurrence records from GBIF from a list of species names. The function `get_gbiftaxonkey` is used internally to get taxon *keys*, and the essential of the code is described in this blog *downloading a long species list on GBIF*, feel free to have a look at it.
Basically, we first need to create a GBIF account because we are going to use GBIF credentials as explained in the blog to login to GBIF and download occurrence records for a fairly large number of species.

II.1 *create a GBIF account*

Go to the GBIF website homepage https://www.gbif.org →click on the `login` button on the top-right corner of the page →click on `register` →fill up the fields to create your account. Then, save your *GBIF username*, *email* associated with username and *password* for logging in to GBIF somewhere safe, for example in your .Renviron file:

```
install.packages("usethis") # if not yet installed
library(usethis)

# open your .Renviron file
edit_r_environ()

# write down your GBIF credentials as environmental variables
GBIF_USER = "your_username"
GBIF_EMAIL = "your_email_address"
GBIF_PWD = "your_password"

# save your changes, close the file and restart your R session
```

Check that your GBIF credentials are now accessible by typing in the console e.g. `Sys.getenv("GBIF_USER")`.

II.2 *use queryGBIF to download occurrence records*

```
# load the useful plants package
library(UsefulPlants)

# make sure species names are correctly formatted
edible_plant_names <- gsub("_"," ",edible_plant_names)

# specify a folder where to store the zip file of your downloads
# it is recommended to use a specific folder for better traceability
dwnld_dir = tempdir()

# run the query using the vector of plant species names
gbifDATA = queryGBIF(species_name=edible_plant_names,
                gbif_download_dir=dwnld_dir,
                # time in seconds before dropping out the connection
                time_out = 300)
```

By default, **queryGBIF** will return after `time_out` seconds. If GBIF has not finished processing your query or if something went wrong, `queryGBIF` will returns `NULL` and an error message, in this case:

- **Login** to your account on GBIF.
- Go to your downloads and **check** out the status of your query.
- Once available, you can manually **download** your dataset as zip file in the directory of your choice.

If everything worked, the output of `queryGBIF` is a `data.table` object with your occurrence records with the following columns:

- *"species"*
- *"fullname"*,
- *"decimalLongitude"*
- *"decimalLatitude"*
- *"countryCode"*
- *"coordinateUncertaintyInMeters"*
- *"year"*
- *"individualCount"*
- *"gbifID"*
- *"basisOfRecord"*
- *"institutionCode"*
- *"establishmentMeans"*
- *"is_cultivated_observation"*
- *"sourceID"* (always GBIF)

Find documentation on most of the fields *https://www.gbif.org/data-quality-requirements-occurrences*.

If your automatic download failed and you ended up downloading a zip file manually, you can still format your dataset as follows:

```
# get the path to your zip file
path_to_my_zip.file <- "path/to/yourzip.file"

# read in the data from your zip file
gbifDATA <- data.table::fread(cmd=paste("unzip -p",path_to_my_zip.file),
                                      header=TRUE,
                                      showProgress=FALSE,
                                      na.strings=c("",NA),
                                      fill=FALSE,
                                      quote = "",
                                      select=c("species",
                                               "taxonRank",
                                               "infraspecificEpithet",
                                               "decimalLongitude",
                                               "decimalLatitude",
                                               "countryCode",
                                               "coordinateUncertaintyInMeters",
                                               "year",
                                               "gbifID",
                                               "basisOfRecord",
                                               "institutionCode",
                                               "establishmentMeans",
                                               "individualCount"))

gbifDATA<-na.omit(gbifDATA, cols= c("decimalLatitude", "decimalLongitude"))

# format the data:
gbifDATA[, `:=`(taxonRank = ifelse(taxonRank %in% c("SPECIES","GENUS"),
                                   NA,
                                   ifelse(taxonRank=="FORM", "f.",
                                          ifelse(taxonRank=="SUBSPECIES", "subsp.",
                                                 ifelse(taxonRank=="VARIETY","var.",
                                                        taxonRank)))),
                    countryCode = countrycode::countrycode(countryCode,
                                                           origin =  'iso2c',
                                                           destination = 'iso3c',
                                                           nomatch = NA),
                    establishmentMeans = ifelse(establishmentMeans=="INTRODUCED",
                                                "Introduced",
                                                ifelse(establishmentMeans== "NATIVE",
                                                       "Native",
                                                       establishmentMeans)))]

# create 'fullname', 'is_cultivated_observation' and 'sourceID' columns
gbifDATA[, `:=`(fullname = ifelse(is.na(infraspecificEpithet),
                                  paste(species),
                                  paste(species, taxonRank, infraspecificEpithet)),
                is_cultivated_observation = NA,
                sourceID = 'GBIF')]
```

```
# delete taxonRank and infraspecificEpithet columns
gbifDATA[, `:=`(taxonRank = NULL, infraspecificEpithet = NULL)]

# remove fossil records
gbifDATA <- gbifDATA[basisOfRecord!="FOSSIL_SPECIMEN"]

# set column order
colNames = c("species",
             "fullname",
             "decimalLongitude",
             "decimalLatitude",
             "countryCode",
             "coordinateUncertaintyInMeters",
             "year",
             "individualCount",
             "gbifID",
             "basisOfRecord",
             "institutionCode",
             "establishmentMeans",
             "is_cultivated_observation",
             "sourceID")

data.table::setcolorder(gbifDATA, colNames)

# set the key to the species column to enable fast binary search
data.table::setkey(gbifDATA, 'species')
```

III. Download occurrence data from BIEN

Similarly, **UsefulPlants** has a function called `queryBIEN` to download occurrence records from the BIEN database. Internally, `queryBIEN` uses `BIEN_occurrence_species` with the following default arguments:

```
cultivated = TRUE,
only.new.world = FALSE,
all.taxonomy = TRUE,
native.status = TRUE,
observation.type = TRUE,
political.boundaries = TRUE,
natives.only = FALSE,
collection.info = TRUE
```

Check out the meaning of these arguments from the package help here *manual*

```
# run the query using the vector of plant species names previously defined in section II.2
bienDATA = queryBIEN(species_name=edible_plant_names)
```

If everything works fine, `bienDATA` should be a `data.table` object formatted as `gbifDATA`, i.e. with the same column names.

IV. Formatting occurrence data from *RAINBIO*, *speciesLink*, *BioTIME* and *Genesys*

For consistency between the different databases, we need to format each occurrence dataset the same way as we did for *GBIF* and *BIEN*.
Let's assume that each dataset comes as a zipped folder, if not the case, replace `paste("unzip -p","path/to/file.zip")` by the path to your unzipped file in the relevant part of the code.

```
# define column names
colNames = c("species",
             "fullname",
             "decimalLongitude",
             "decimalLatitude",
             "countryCode",
             "coordinateUncertaintyInMeters",
             "year",
             "individualCount",
             "gbifID",
             "basisOfRecord",
             "institutionCode",
             "establishmentMeans",
             "is_cultivated_observation",
             "sourceID")
```

RAINBIO

```
# read in the data
rainbioDATA <- data.table::fread(paste("unzip -p","path/to/file.zip"),
                                 header=TRUE,
                                 showProgress=FALSE,
                                 select=c("tax_sp_level",
                                  "species",
                                  "decimalLatitude",
                                  "decimalLongitude",
                                  "iso3lonlat",
                                  "basisOfRecord",
                                  "institutionCode",
                                  "catalogNumber",
                                  "coly"))

#------------
#= formatting
#------------
# change the column names
data.table::setnames(rainbioDATA,
                     c("species",
                       "fullname",
                       "decimalLatitude",
                       "decimalLongitude",
                       "countryCode",
                       "basisOfRecord",
                       "institutionCode",
                       "gbifID",
```

```
                        "year"))

# remove NA coordinates
rainbioDATA <- na.omit(rainbioDATA, cols= c("decimalLatitude", "decimalLongitude"))

# create additional columns: 'coordinateUncertaintyInMeters' , 'is_cultivated_observation',
# 'establishmentMeans', 'individualCount' and 'sourceID'

rainbioDATA[ ,`:=`(coordinateUncertaintyInMeters = NA,
                   is_cultivated_observation = "No",
                   establishmentMeans = NA,
                   individualCount = NA,
                   sourceID = 'RAINBIO')]

# set column order
data.table::setcolorder(rainbioDATA, colNames)

# set the key to the species column to enable fast binary search
data.table::setkey(rainbioDATA, 'species')
```

speciesLink

```
# read in the data
spLinkDATA <- data.table::fread(paste("unzip -p","path/to/file.zip"),
                                    header=TRUE,
                                    showProgress=FALSE,
                                    encoding = 'UTF-8',
                                    na.strings=c("",NA),
                                    select= c("query",
                                              "scientificname",
                                              "longitude",
                                              "latitude",
                                              "country",
                                              "coordinateprecision",
                                              "yearcollected",
                                              "individualcount",
                                              "catalognumber",
                                              "basisofrecord",
                                              "institutioncode"))

#------------
#= formatting
#------------
# change the column names
data.table::setnames(spLinkDATA,
                     c("species",
                       "fullname",
                       "decimalLatitude",
                       "decimalLongitude",
                       "countryCode",
                       "coordinateUncertaintyInMeters",
                       "year",
```

```
                        "individualCount",
                        "gbifID",
                        "basisOfRecord",
                        "institutionCode"))

# remove NA coordinates
spLinkDATA <- na.omit(spLinkDATA, cols= c("decimalLatitude", "decimalLongitude"))

spLinkDATA[ ,`:=`(species = paste0(substr(species,1,1),
                                   tolower(substr(species,2,nchar(species)))),
                  countryCode = ifelse(countryCode=='Brasil','Brazil',countryCode),
                  basisOfRecord = ifelse(basisOfRecord=="S","SPECIMEN",
                                         ifelse(basisOfRecord== "O",
                                                "OBSERVATION",
                                                basisOfRecord)),
                  is_cultivated_observation = NA,
                  establishmentMeans = NA,
                  sourceID = 'spLink')]

spLinkDATA[ ,countryCode:= countrycode::countrycode(countryCode,
                                                    origin =  'country.name',
                                                    destination = 'iso3c',
                                                    nomatch = NA)]
# set column order
data.table::setcolorder(spLinkDATA, colNames)

# set the key to the species column to enable fast binary search
data.table::setkey(spLinkDATA, 'species')
```

bioTIME

```
# read in the data
biotimeDATA  <- data.table::fread(paste("unzip -p","path/to/file.zip"),
                                  header=TRUE,
                                  showProgress=FALSE,
                                  select=c("GENUS_SPECIES",
                                           "LATITUDE",
                                           "LONGITUDE",
                                           "YEAR",
                                           "sum.allrawdata.ABUNDANCE",
                                           "SAMPLE_DESC"))

#---------------
#= formatting
#---------------

# change the column names
data.table::setnames(biotimeDATA, c("species",
                                    "decimalLatitude",
                                    "decimalLongitude",
                                    "year",
                                    "individualCount",
```

```
                                    "gbifID"))

# remove NA coordinates
biotimeDATA  <- na.omit(biotimeDATA, cols= c("decimalLatitude", "decimalLongitude"))

# create additional columns: 'fullname', 'coordinateUncertaintyInMeters','institutionCode',
#'countryCode', 'basisOfRecord', 'establishmentMeans', 'is_cultivated_observation'
# and 'sourceID'
biotimeDATA[ ,`:=`(fullname = species,
                   coordinateUncertaintyInMeters = NA,
                   institutionCode = NA,
                   countryCode = NA,
                   basisOfRecord = NA,
                   establishmentMeans = NA,
                   is_cultivated_observation = "No",
                   sourceID = 'BIOTIME')]

# set column order
data.table::setcolorder(biotimeDATA, colNames)

# set the key to the species column to enable fast binary search
data.table::setkey(biotimeDATA, 'species')
```

Genesys

```
# read in the data
genesysDATA <- data.table::fread(paste("unzip -p","path/to/file.zip"),
                                    header=TRUE,
                                    showProgress=FALSE,
                                    na.strings=c("",NA),
                                    select= c("GENUS",
                                   "SPECIES",
                                   "SUBTAXA",
                                   "DECLONGITUDE",
                                   "DECLATITUDE",
                                   "ORIGCTY",
                                   "COLLSRC",
                                   "INSTCODE",
                                   "UUID",
                                   "COLLDATE",
                                   "COORDUNCERT",
                                   "SAMPSTAT"))

#---------------
#= formatting
#---------------

# change the column names
data.table::setnames(genesysDATA,
                     c("genus",
                       "species",
                       "subtaxa",
```

```
                        "decimalLongitude",
                        "decimalLatitude",
                        "countryCode",
                        "basisOfRecord",
                        "institutionCode",
                        "gbifID",
                        "year",
                        "coordinateUncertaintyInMeters",
                        "is_cultivated_observation"))

# format the data:
# concatenate genus+species,
# create full species name,
# extract the year of the collection,
# transform information about cultivated observation and basis of records in categorical variable
genesysDATA[, `:=`(species = paste(genus, species),
                   fullname = ifelse(is.na(subtaxa),
                                     paste(genus, species),
                                     paste(genus, species, subtaxa)),
                   year = as.numeric(stringr::str_extract(year,"[[:digit:]]{4}")),
                   is_cultivated_observation = ifelse(is_cultivated_observation==999,
                                                      NA,
                                                      ifelse(is_cultivated_observation > 300,
                                                             "Yes",
                                                             "No")),
                   basisOfRecord = cut(basisOfRecord,
                                       breaks=c(10, 16, 29, 59, 63),
                                       labels=c(
                                         "Wild habitat",
                                         NA,
                                         "Cultivated habitat",
                                         "Wild habitat")
                                       ))]

# create additional columns: 'establishmentMeans', 'individualCount' and
# 'sourceID' and delete 'genus' and 'subtaxa'
genesysDATA[ ,`:=`(genus = NULL,
                   subtaxa = NULL,
                   establishmentMeans = NA,
                   individualCount = NA,
                   sourceID = 'GENESYS')]

# set column order
data.table::setcolorder(genesysDATA, colNames)

# set the key to the species column to enable fast binary search
data.table::setkey(genesysDATA, 'species')
```

References

Brown, Matilda J. M., Barnaby E. Walker, Nicholas Black, Rafaël Govaerts, Ian Ondo, Robert Turner, and Eimear Nic Lughadha. 2023. "rWCVP: A Companion r Package to the World Checklist of Vascular Plants." *New Phytologist*.

Chamberlain, Scott, and Carl Boettiger. 2017. "R Python, and Ruby Clients for GBIF Species Occurrence Data." *PeerJ PrePrints*. https://doi.org/10.7287/peerj.preprints.3304v1.

Dowle, Matt, and Arun Srinivasan. 2021. *Data.table: Extension of 'Data.frame'*. https://CRAN.R-project.org/package=data.table.

Maitner, Brian. 2020. *BIEN: Tools for Accessing the Botanical Information and Ecology Network Database*. https://CRAN.R-project.org/package=BIEN.

Scott Chamberlain, and Eduard Szocs. 2013. "Taxize - Taxonomic Search and Retrieval in r." *F1000Research*. https://f1000research.com/articles/2-191/v2.