Many students and workshop participants ask me for a (semi-)automated way to 1) download species occurrence data from GBIF into R, and 2) clean such data from common errors. The following script does that, while calling the user’s attention to the need for **properly citing the data sources** (not just GBIF, which is a repository for many sources), and for **carefully mapping and inspecting** them for additional problems which are not picked up by current tools.

```
#######################################
## DOWNLOAD AND CLEAN DATA FROM GBIF ##
#######################################

library(rgbif)
library(scrubr)
library(maps)


# IF YOU HAVE ONLY ONE SPECIES ----

myspecies <- c("Galemys pyrenaicus")

# download GBIF occurrence data for this species; this takes time if
there are many data points!
gbif_data <- occ_data(scientificName = myspecies, hasCoordinate = TRUE,
limit = 20000)

# take a look at the downloaded data:
gbif_data
# if "Records found" is larger than "Records returned", you need to
increase the 'limit' argument above -- see help(occ_data) for options
and limitations


# if your species is widespread but you want to work on a particular
region, you can download records within a specified window of
coordinates:
gbif_data <- occ_data(scientificName = myspecies, hasCoordinate = TRUE,
limit = 20000, decimalLongitude = "-10, 10", decimalLatitude = "35,
55")  # note that coordinate ranges must be specified this way:
"smaller, larger" (e.g. "-5, -2")

gbif_data

# get the DOIs for citing these data properly:
gbif_citation(gbif_data)
# note: if you need or prefer only one DOI for the entire dataset,
download the dataset directly from www.gbif.org and then import the .csv
to R. It is very important to properly cite the data sources! GBIF is
not a source, just a repository for many people who put in very hard
work to collect these data and make them available

# check how the data are organized:
```

```
names(gbif_data)
names(gbif_data$meta)
names(gbif_data$data)

# get the columns that matter for mapping and cleaning the occurrence
data:
myspecies_coords <- gbif_data$data[ , c("decimalLongitude",
"decimalLatitude", "individualCount", "occurrenceStatus", "
coordinateUncertaintyInMeters", "institutionCode", "references")]
head(myspecies_coords)

# map the occurrence data:
map("world", xlim = range(myspecies_coords$decimalLongitude), ylim =
range(myspecies_coords$decimalLatitude))  # if the map doesn't appear
right at first, run this command again
points(myspecies_coords[ , c("decimalLongitude", "decimalLatitude")],
pch = ".")

# you may notice (especially if you zoom in, e.g. by specifying a
smaller range of coordinates under 'xlim' and 'ylim' above) that many
points are too regularly spaced to be exact locations of species
sightings; rather, such points are likely to be centroids of
(relatively large) grid cells on which particular surveys was based, so
remember to adjust the spatial resolution of your analysis accordingly!

# also, these data are likely to contain species absences and location
errors, so jump to "CLEAN THE DATASET" section below - this is VERY
IMPORTANT!!!


# IF YOU HAVE MORE THAN ONE SPECIES ----

myspecies <- c("Galemys pyrenaicus", "Chioglossa lusitanica")

# download GBIF occurrence data for these species; this may take a long
time if there are many data points!
gbif_data <- occ_data(scientificName = myspecies, hasCoordinate = TRUE,
limit = 500)  # decrease the 'limit' if you just want to see how many
records there are without waiting all the time that it will take to
download the whole dataset

# take a look at the downloaded data:
gbif_data
# if, for any species, "Records found" is larger than "Records
returned", you need to increase the 'limit' argument above -- see
help(occ_data) for options and limitations

# get the DOI for citing these data properly:
gbif_citation(gbif_data)  # unfortunately it is more complicated to
obtain with R a proper citation for a dataset with multiple species. To
get a DOI for these data, download the dataset directly from www.gbif.org
and then import the .csv to R. It is very important to properly cite
```

```
the data sources! GBIF is not a source, just a repository for many
people who put in very hard work to collect these data and make them
available

# if your species are widespread but you want to work on a particular
region, you can download records within a specified window of
coordinates:
gbif_data <- occ_data(scientificName = myspecies, hasCoordinate = TRUE,
limit = 20000, decimalLongitude = "-10, 10", decimalLatitude = "35,
55")  # note that coordinate ranges must be specified this way:
"smaller, larger" (e.g. "-5, -2")

gbif_data

# check how the data are organized:
names(gbif_data)
names(gbif_data[[myspecies[1]]])
names(gbif_data[[myspecies[1]]]$meta)
names(gbif_data[[myspecies[1]]]$data)


# create and fill a list with only the 'data' section for each species:
myspecies_coords_list <- vector("list", length(myspecies))
names(myspecies_coords_list) <- myspecies
for (s in myspecies) {
  coords <- gbif_data[[s]]$data[ , c("decimalLongitude",
"decimalLatitude", "individualCount", "occurrenceStatus", "
coordinateUncertaintyInMeters", "institutionCode", "references")]
  myspecies_coords_list[[s]] <- data.frame(species = s, coords)
}
lapply(myspecies_coords_list, head)

# collapse the list into a data frame:
myspecies_coords <- as.data.frame(do.call(rbind,
myspecies_coords_list), row.names = 1:sum(sapply(myspecies_coords_list,
nrow)))
head(myspecies_coords)
tail(myspecies_coords)

# map the occurrence data:
map("world", xlim = range(myspecies_coords$decimalLongitude), ylim =
range(myspecies_coords$decimalLatitude))  # if the map doesn't appear
right at first, run this command again
points(myspecies_coords[ , c("decimalLongitude", "decimalLatitude")],
col = myspecies_coords$species, pch = ".")

# you may notice (especially if you zoom in, e.g. by specifying a
smaller range of coordinates under 'xlim' and 'ylim' above) that many
points are too regularly spaced to be exact locations of species
sightings; rather, such points are likely to be centroids of
(relatively large) grid cells on which particular surveys were based,
so remember to adjust the spatial resolution of your analysis
```

```
accordingly!


# CLEAN THE DATASET! ----

# mind that data often contain errors, so careful inspection and
cleaning are necessary!
# here we'll first remove records of absence or zero-abundance (if
any):
names(myspecies_coords)
sort(unique(myspecies_coords$individualCount))  # notice if some points
correspond to zero abundance
sort(unique(myspecies_coords$occurrenceStatus))  # check for different
indications of "absent", which could be in different languages! and
remember that R is case-sensitive
absence_rows <- which(myspecies_coords$individualCount == 0 |
myspecies_coords$occurrenceStatus %in% c("absent", "Absent", "ABSENT",
"ausente", "Ausente", "AUSENTE"))
length(absence_rows)
if (length(absence_rows) > 0) {
  myspecies_coords <- myspecies_coords[-absence_rows, ]
}

# let's do some further data cleaning with functions of the 'scrubr'
package (but note this cleaning is not exhaustive!)
nrow(myspecies_coords)
myspecies_coords <- coord_incomplete(coord_imprecise(coord_impossible(
coord_unlikely(myspecies_coords))))
nrow(myspecies_coords)

# map the cleaned occurrence data:
map("world", xlim = range(myspecies_coords$decimalLongitude), ylim =
range(myspecies_coords$decimalLatitude))  # if the map doesn't appear
right at first, run this command again
points(myspecies_coords[ , c("decimalLongitude", "decimalLatitude")],
col = myspecies_coords$species, pch = ".")
# possible erroneous points e.g. on the Equator (lat and lon = 0)
should have disappeared now

# also eliminate presences with reported coordinate uncertainty
(location error, spatial resolution) larger than 5 km (5000 m):
myspecies_coords <- coord_uncertain(myspecies_coords,
coorduncertainityLimit = 5000)
nrow(myspecies_coords)
# but note that this will only get rid of records where coordinate
uncertainty is adequately reported, which may not always be the case!
Careful mapping and visual inspection is necessary

# map the cleaned occurrence records with a different colour on top of
the raw ones:
points(myspecies_coords[ , c("decimalLongitude", "decimalLatitude")],
pch = 20, cex = 0.5, col = "turquoise")…
```