Data mobilisation from GBIF to the EBV Data Portal for the Birds Directive Annex I

Notebook 04 - Calculation of Metrics for the Birds Directive Annex I Using Occurrence Cubes (Part II)

true

2024-08-23

Introduction

In this notebook, we calculate simple metrics for the EU Annex I Birds Directive based on species occurrence in GBIF. For this, a bird occurrence cube (see Notebook 01 in this repository) was previously created using the occurrence cube software developed by GBIF under the Biodiversity Building Blocks for Policy (B3) Project. Details of the data query in GBIF are available at doi: 10.15468/dl.uh84tp. In this notebook we calculate metrics 4, 5 and 6 listed below:

- Month with the highest total number of occurrences across all years
- Month with the second-highest total number of occurrences across all years
- Month with the third-highest total number of occurrences across all years

Note: This series of notebooks is part of the results of Task 3.3 of the Biodiversity Building Blocks for Policy project funded by the European Union's Horizon Europe Research and Innovation Programme (ID No 101059592). Additional notebooks exploring the results and calculating simple metrics are also available in the same repository.

Load Library and Input Data

We start by loading all the libraries needed in this notebook.

```
rm(list=ls())
gc()
```

```
##           used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells 2483764 132.7    7100552  379.3   7100552  379.3
## Vcells 9939686  75.9  656190447 5006.4 868779994 6628.3
```

```
# Load requiered libraries
library(b3gbi) # for csv occurrence cubes
library(purrr) # for data summary and grouping
library(here)
library(dplyr)
```

```r
library(lubridate) # for dates
library(terra) # for raster
library(ncdf4)
library(ggplot2)
library(sf)
library(stringr)
library(viridis)
```

Now we load all input data obtained in the previous notebooks. These are:

- The occurrence cube of the Birds Directive Annex I obtained previously through GBIF API.
- Taxonomic information of all analysed species.
- Inputs related to the EEA grid at 10 km for rasterisation.

```r
occcube <- "0062979-240626123714530"

# Load occurrence cube using b3gbi
cin <- process_cube(here(paste0("output/datacubes/csv/birds/", occcube,".csv")))

# Import the taxonomy of all species listed in the Birds Directive Annex 1 based on gbif backbone taxon
gbif_tax <- read.csv(here("input/data/birds/taxonomy/list193birds_directive_annexi_allaccepted_usagekey_

# Load reference EEA grid in raster format
gridin <- rast(here("input/grid/eeagrid_10K.tif"))
res <- res(gridin)[1] #resolution of reference raster

# Load precomputed centroids for EEA 10 km grid
coorin <- read.csv(here("input/grid/centroids/eeagrid_centroids_10K.csv"))

# Replace by corresponding column name of input dataset
colnames(coorin)[colnames(coorin) == "eeacellcode"] <- "cellCode"

# Import and convert the EU borders to a data frame
borders_eu <- st_read(here("input/grid/shp/NUTS2021_3035.shp"))
```

```
## Reading layer 'NUTS2021_3035' from data source 'H:\B3\B-Cubed_data_mobilization\input\grid\shp\NUTS2
## Simple feature collection with 44 features and 27 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 943758.8 ymin: 941658.1 xmax: 7316569 ymax: 6405005
## Projected CRS: ETRS89-extended / LAEA Europe
```

We now identify which species of the Birds Directive Annex I have data in our species occurrence cubes.

```r
# Find number of species
spskey <- unique(cin[["data"]][["taxonKey"]]) # "taxonKey" is the name of the species key in b3bgi
```

For metrics 1 to 3, see the accompanying notebook in the same repository.

Metrics 4, 5 and 6: Months with the highest, second-highest, and third-highest total number of GBIF occurrences across all years We identify the months with the highest (Metric 4), second highest (Metric 5), and third highest (Metric 6) number of records occurrences at GBIF.

```r
maxrec_month123 <- function(cin, gridin, coorin){
  # Find number of species
  spskey <- unique(cin[["data"]][["taxonKey"]])

  # Create empty raster for metric 4
  r4 <- rast(ext(gridin), resolution=res(gridin), nlyrs=length(spskey), crs=crs(gridin))
  values(r4) <- NA

  # Create empty raster for metric 5
  r5 <- rast(ext(gridin), resolution=res(gridin), nlyrs=length(spskey), crs=crs(gridin))
  values(r5) <- NA

  # Create empty raster for metric 6
  r6 <- rast(ext(gridin), resolution=res(gridin), nlyrs=length(spskey), crs=crs(gridin))
  values(r6) <- NA

  # i <- 57

  for (i in 1:length(spskey)){ # 1:length(spskey)){
  # print(i)

  # Subset one species
  spsi <- cin[["data"]][cin[["data"]]$taxonKey == spskey[i], ]

  # Convert date from character to numeric
  todates <- as.data.frame(str_split(spsi$yearMonth, "-", simplify = TRUE))
  colnames(todates) <- c("year", "month")

  # Add year and month columns separate to the initial data
  spsi$year <- todates$year
  spsi$month <- todates$month

  # Aggregate occurrences by cell and month
  month_occ <- spsi %>%
      group_by(cellCode, month) %>%
      summarize(total_occurrences = sum(obs), cellCode = first(cellCode)) %>%
      ungroup()

    # Calculate metrics 4, 5, and 6 (1st, 2nd, and 3rd highest months)
    metric4 <- month_occ %>%
      group_by(cellCode) %>%
      slice_max(total_occurrences, n = 1, with_ties = FALSE) %>%
      ungroup()

    metric5 <- month_occ %>%
      group_by(cellCode) %>%
      slice_max(total_occurrences, n = 2, with_ties = FALSE) %>%
      slice_min(total_occurrences, n = 1, with_ties = FALSE) %>%
      ungroup()

    metric6 <- month_occ %>%
      group_by(cellCode) %>%
      slice_max(total_occurrences, n = 3, with_ties = FALSE) %>%
```

```
      slice_min(total_occurrences, n = 1, with_ties = FALSE) %>%
      ungroup()

  metricx_coor <- function(metricx, coorin, res){

    # Merge pixel coordinates with the corresponing EEA grid ID pre-rasterisation
    metriccoorx <- merge(metricx[,c("month", "cellCode")], coorin, by="cellCode")
    # print(length(unique(metriccoorx$x)))

    if(length(unique(metriccoorx$x)) < 10){
      #add second empty point
      x <- metriccoorx$x[1] + res
      y <- metriccoorx$y[1] + res
      metriccoorx <- rbind(metriccoorx, c(NA, NA ,NA, x, y))
    }
    return(metriccoorx)
  }

  # Run function for each metric
  metric4_coor <- metricx_coor(metric4, coorin, res)
  metric5_coor <- metricx_coor(metric5, coorin, res)
  metric6_coor <- metricx_coor(metric6, coorin, res)

  # Rasterize data
  r4[[i]]  <- rast(metric4_coor[,c("x", "y", "month")], type="xyz", crs=crs(gridin), extent=ext(gridin))
  r5[[i]]  <- rast(metric5_coor[,c("x", "y", "month")], type="xyz", crs=crs(gridin), extent=ext(gridin))
  r6[[i]]  <- rast(metric6_coor[,c("x", "y", "month")], type="xyz", crs=crs(gridin), extent=ext(gridin))

 }

  # Assign name of species key to the corresponding raster 'layer'
  names(r4) <- spskey
  names(r5) <- spskey
  names(r6) <- spskey

  results_list <- list(month_with_highest_records = r4, month_with_2ndhighest_records = r5, month_with_3
  return(results_list)

}
```

In the following code chunk, we compute metrics 4, 5 and 6.

```
# Calculate metrics 4, 5 and 6
months_high_rec <- maxrec_month123(cin, gridin, coorin)
```

```
# Calculate metrics 4, 5 and 6
month_1sthigh <- months_high_rec[[1]]
month_2ndhigh <- months_high_rec[[2]]
month_3rdhigh <- months_high_rec[[3]]
```

```
# Convert raster layers to data frames
spi <- 57
df1 <- as.data.frame(month_1sthigh[[spi]], xy=TRUE)
```

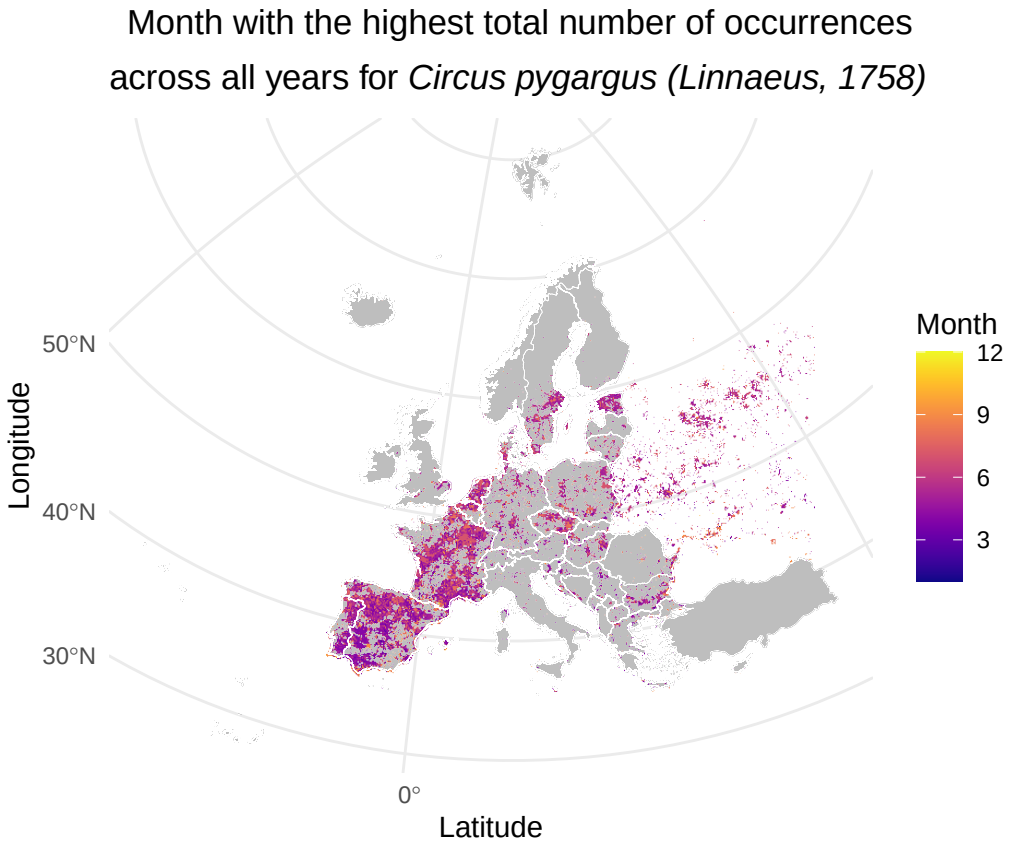
```r
spid <- as.numeric(colnames(df1)[3])
colnames(df1)[colnames(df1) == colnames(df1)[3]] <- "Value"
df1[["Value"]] <- df1[["Value"]]

df2 <- as.data.frame(month_2ndhigh[[spi]], xy=TRUE)
spid <- as.numeric(colnames(df2)[3])
colnames(df2)[colnames(df2) == colnames(df2)[3]] <- "Value"
df2[["Value"]] <- df2[["Value"]]

df3 <- as.data.frame(month_3rdhigh[[spi]], xy=TRUE)
spid <- as.numeric(colnames(df3)[3])
colnames(df3)[colnames(df3) == colnames(df3)[3]] <- "Value"
df3[["Value"]] <- df3[["Value"]]


# Find the specie scientific name
spi_name <- gbif_tax %>%
  filter(acceptedUsageKey %in% spid)
```

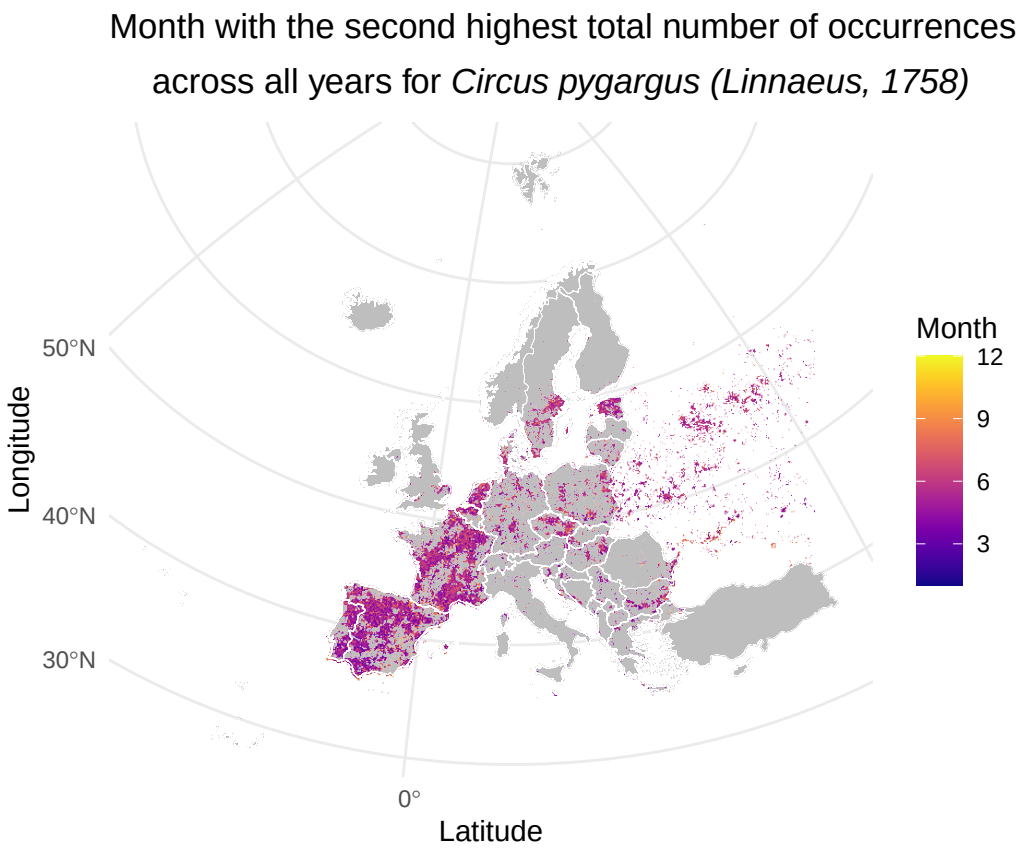
```r
ggplot() +
  geom_sf(data = borders_eu, fill = "gray") +
  geom_raster(data = df1, aes(x = x, y = y, fill = Value)) +
  geom_sf(data = borders_eu, fill = NA, color = "white") +
  scale_fill_viridis(name="Month", option = "C", direction = 1) + #
  theme_minimal() +
  labs(
    title = bquote(atop("Month with the highest total number of occurrences",
                        "across all years for " * italic(.(spi_name$scientificName)))),
    x = "Latitude",
    y = "Longitude",
    fill = "Month")
```




```
# save figure if needed
ggsave(here("output/figures/nb_birds/ias_metric4_id82.png"))
```

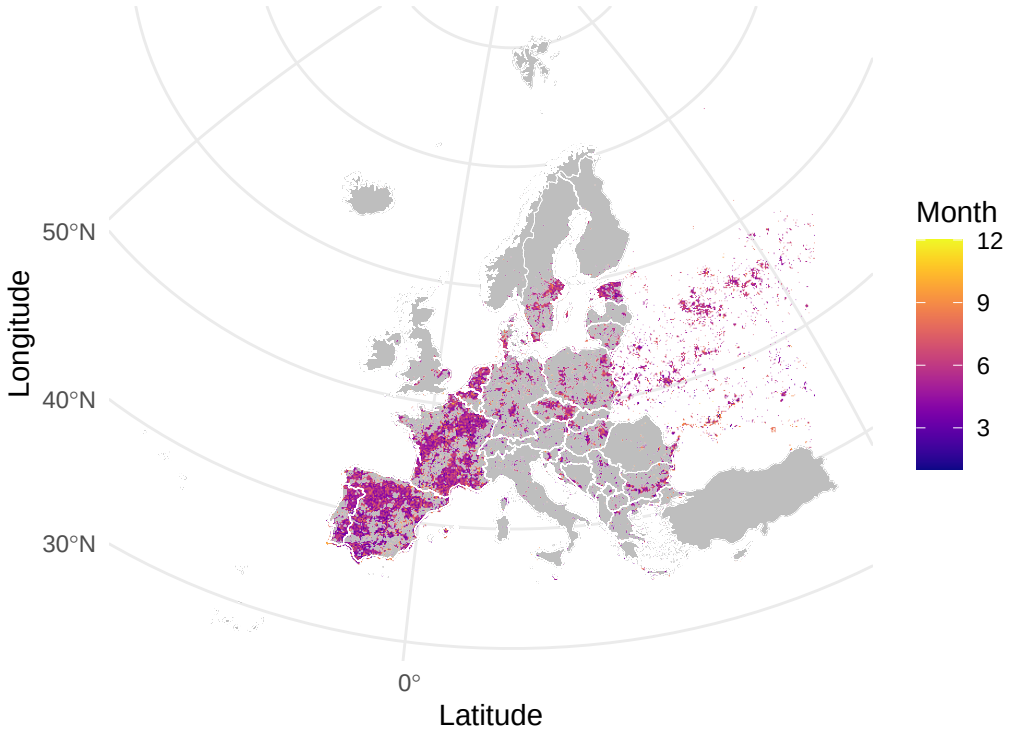
```
ggplot() +
  geom_sf(data = borders_eu, fill = "gray") +
  geom_raster(data = df2, aes(x = x, y = y, fill = Value)) +
  geom_sf(data = borders_eu, fill = NA, color = "white") +
  scale_fill_viridis(name="Month", option = "C", direction = 1) + #
  theme_minimal() +
  labs(
    title = bquote(atop("Month with the second highest total number of occurrences",
                        "across all years for " * italic(.(spi_name$scientificName)))),

    x = "Latitude",
    y = "Longitude",
    fill = "Month")
```




```
# save figure if needed
ggsave(here("output/figures/nb_birds/ias_metric5_id82.png"))
```

```
ggplot() +
  geom_sf(data = borders_eu, fill = "gray") +
  geom_raster(data = df3, aes(x = x, y = y, fill = Value)) +
  geom_sf(data = borders_eu, fill = NA, color = "white") +
  scale_fill_viridis(name="Month", option = "C", direction = 1) + #
  theme_minimal() +
  labs(
    title = bquote(atop("Month with the third highest total number of occurrences",
                        "across all years for " * italic(.(spi_name$scientificName)))),
    x = "Latitude",
    y = "Longitude",
    fill = "Month")
```




```
# save figure if needed
ggsave(here("output/figures/nb_birds/ias_metric6_id82.png"))
```

Saving data sets as individual tiffs

```
# Save files
writeRaster(month_1sthigh, here("output/datacubes/tif_metrics/birds/04_birds_month_1sthigh.tif"), datat
writeRaster(month_2ndhigh, here("output/datacubes/tif_metrics/birds/05_birds_month_2ndhigh.tif"), datat
writeRaster(month_3rdhigh, here("output/datacubes/tif_metrics/birds/06_birds_month_3rdhigh.tif"), datat
```