Getting started and function documentation

The script file ‘**functions.R**’ currently contains 11 functions (not sourced yet).

7 read data from the system and 4 from the cloud. The functions that communicate with the cloud are documented in another .docx file.

**7 functions** are described in this file.

**readcleanrawdata(rawpath)**

This function needs to be **run only once on every new eBird data release**. It reads the .txt file, adds group.id (assigning unique sampling id to group.id for lists that are not shared) and no.sp (the number of species reported in a list) columns, day, week, fort, month, year columns, and selects only important columns (stored in vector ‘imp’ in the function and subsequently in other functions).

‘rawpath’ currently defaults to "ebd\_IN\_relAug-2018.txt"; this is the current eBird data release for India and must be stored in the working directory (and changed accordingly) along with default argument to use the function.

The function writes a ‘data.RData’ file to the working directory and a ‘indiaspecieslist.csv’ file (data frame of all species with associated species-specific variables). These files must remain for use in other preliminary functions and are stored (and will be updated) in the shared team Google Drive.

libraries required – lubridate, tidyverse

**createmaps(g1,g2,g3,g4,g5,path1,name1…path3,name3)**

g1-g5 are **five different grid resolutions** that default to 20,40,60,80,320 km; path1-path3 are the names of the folders (in the working directory) where **country, state and district** spatialfiles name1-name3 are stored (currently in the Google Drive, copy to working directory); check function for default paths and names

This function needs to be run **only when you require different gridding options, or when country, state or district shapefiles become more current** (the files are from a 2011 release). It reads shapefiles and creates 8 spatial polygon data frame objects that can then be used for plotting and to talk to the data.

The function writes a ‘maps.RData’ file to the working directory. This file must remain for use in other preliminary functions and is stored (and will be updated) in the shared team Google Drive.

libraries required – tidyverse, rgdal, sp

**createmask(grid,path1,name1)**

This function creates a **plotting mask and a border for grids**.

‘grid’ currently defaults to 27, feels arbitrary but is the highest resolution at which masking appears complete; path1 and name1 are the same as in the previous function

The function writes a ‘mask.RData’ file to the working directory. This file must remain for use for grid plotting and is stored (and will be updated if required) in the shared team Google Drive. It **does not need to be run again unless either the resolution needs to be changed or the default India shapefile changes**.

libraries required – tidyverse, ggfortify, sp, rgdal, raster

**addmapvars(datapath,mappath)**

‘datapath’ is the path to the .RData file from readcleanrawdata() and mappath is the path to the .RData file from createmaps()

This function **needs to be run only when either of these .RData files are modified**. It adds spatial map variables from spatial polygon data frames to the data itself (adds 7 new columns that correspond with 7 of 8 SPDFs in maps.RData and adds a column called LOCALITY.HOTSPOT that attracts all personal locations into nearest hotspots). This is essential both for subsequent spatial analyses and for plotting.

The function writes a ‘dataforspatialanalyses.RData’ file to the working directory. This file must be loaded every time and is stored (and will be updated if required) in the shared team Google Drive.

libraries required – tidyverse, data.table, sp, rgeos

**plotfreqmap(data,resolution,species,mappath,maskpath)**

This function is entirely for plotting frequencies at different resolutions on an India map.

‘data’ can be any subset of data of the data format in ‘dataforspatialanalyses.RData’; resolution can take any of “g1”, “g2”, “g3”, “g4”, “g5”, “district”, “state” where g1-g5 correspond to 20, 40, 60, 80, 320 or as this definition evolves (please modify documentation when required); ‘species’ is the common name of any bird species found in India, ‘mappath’ and ‘maskpath’ are defaulted to load previously created .RData files

The function returns a ggplot object and tidyverse must be loaded to view it.

libraries required – tidyverse, ggfortify, viridis

**expandbyspecies(data, species)**

This function is for expanding from a presence only data format to presence absence for selected species (‘speclist’) in each list. It returns an expanded data frame.

libraries required - tidyverse

**freqcompare(data,species,tempres,spaceres)**

This function is for comparing trivial frequencies of reporting for a single given ‘species’ common name with modelled frequencies at the country scale. It calls expandbyspecies() and allows data to be input at various ‘tempres’ and ‘spaceres’; these can take the values “fortnight”, “month”, “none” and “g2”, “g4”, “none” respectively. “g2” and “g4” correspond to 40 and 80 km grids.

Various combinations of these will filter data by combinations of time-space where a species is known to be present. This will be particularly useful for migratory species and others of the sort where reporting frequencies are meaningful only when there is a possibility of a species being present.

The 2 models are currently very simple and only include no.sp as a covariate with different dependence structures for random effects; the 1st has state/hotspot and the 2nd has g5/g3/g1. The function returns a data frame with 6 rows and 2 columns. The first 4 rows are increasingly spatially hierarchical summaries from the data and the last two are predicted frequencies from the models.

This function can take time to run (depending on species) and **must be continuously modified for better understanding of abundance and reporting frequencies**.