Introduction to lifeR

Jeffrey C. Oliver

1 June, 2022

Summary

lifeR is an R package for identifying locations to visit in order to increase your bird species list count. The package relies on the eBird API to query for recent observations and compare them to a user's species list. The lists can be life lists, year lists, etc. The primary output is a report of sites to visit in a region where species *not* on the user's list have been reported recently.

Installation

Currently, you will need to install the package from GitHub; you can do this with help from the remotes package:

```
install.packages("remotes")
remotes::install_github(repo = "jcoliver/lifeR")
```

If you want to have the introductory vignette included in the installation, then pass `build_vignettes = TRUE` in the call to `install_github()`:

```
remotes::install_github(repo = "jcoliver/lifeR", build_vignettes = TRUE)
```

Using lifeR

Requirements

You will need a minimum of three pieces of information to use lifeR:

1. An eBird API key
2. A list of species you have seen (life list, year list, etc.)
3. A pair of latitude and longitude coordinates to center the search for potential locations.

1. eBird API key

Because lifeR relies on the public eBird API, you will need to have your own unique API key to use this package. Visit https://ebird.org/api/keygen to get an eBird API key (note you will be prompted to log into your eBird account). The key will be a random string of letters and numbers. I suggest you copy the key and save it to a plain text file on your computer; you can use a program like Notepad or TextEdit to create plain text files. Save this file somewhere you will remember (like "My Documents") and name it something that is recognizable (e.g. "ebird-api-key.txt"). We will use this file later to access the eBird API.

2. Your species list

This should be a list of the species you have already seen, but it can vary based on the list you would like to add species to. If you want to find sites where birds you have never seen have been observed recently, you would use your life list. If you are working on a Big Year, you would use your year list. Trying to break 300 species for your Arizona year list? Then yes, use your Arizona species list for the current year. These can all be downloaded from your eBird account by going to https://ebird.org/lifelist/, using the dropdown menus near the top of the page to select the region and time range of your list, then downloading the list through the "Download (csv)" link near the upper-right corner of the page. Save the file somewhere where you will remember where it is.

3. Latitude and longitude

Finally, lifeR is designed to identify sites in a region for you to visit. Therefore, you will need to provide information on *where* that region is. The package uses a latitude and longitude coordinates to serve as the "center" of the region of interest and finds sites within a certain radius of that center. The maximum radius is 50 kilometers - if you want to search a larger area, you will just need to define two centers (information on doing this is provided below). If you do not know the latitude and longitude of your "center" I recommend Google Maps, which allows you to right- or control-click on a location on a map, and the latitude and longitude coordinates will be displayed. Note lifeR requires decimal degree formatted data (e.g. -110.92, *not* 110° 55' 12").

Create the report

Now that you have those three pieces of information, you can build a report of sites to visit.

A minimalist example

This example creates a report from a single center, near McCall, Idaho, USA. It uses two functions from lifeR:

- `SplitNames` a helper function to deal with the format of species names returned from eBird, and
- `SitesReport` a function that handles all the querying of eBird, comparing lists, and building the report

```r
library(lifeR)
# To use the sample list included in this package
list_file <- system.file("extdata", "example-list.csv", package = "lifeR")

# If you have your list file downloaded, replace the line above with one that
# indicates the location of your list file, e.g.
# list_file <- "~/Desktop/ebird_world_year_2021_list.csv"

# Read the list of species into memory
user_list <- read.csv(file = list_file)

# Extract the common names of species from your list
my_species <- SplitNames(x = user_list$Species)$Common

# Read in eBird API key from a text file; replace the argument to file with
# the actual location of your eBird key file
key <- scan(file = "ebird-api-key.txt", what = "character")

# A single center requires vector of coordinates
# Change these, unless you really want to go birding near McCall, Idaho
locs <- c(45, -116)
SitesReport(centers = locs,
            ebird_key = key,
            species_seen = my_species)
```

The resulting report will be called Goals-Report.html and you can open it in your favorite web browser.

Multiple sites, with names

The example above only uses one center on which to base searches and does not provide a name for that center (it will be given the default name of "Center 1" in the report). In the example below, we provide two starting points. Note that a list of nearby sites *for each starting point* will be included in the results. That is, the report will include the top five sites near the first set of latitude and longitude coordinates and the top five sites near the second set of latitude and longitude coordinates. The example assumes you have loaded in your species list and the location of your eBird API key file as in the first example, above.

```r
# For more than one location, centers can be a matrix or a data frame, here
# we use a matrix of two sites
loc_mat <- matrix(data = c(39.5, -118.8, 39, -119.1), nrow = 2, byrow = TRUE)

# Instead of default "Center 1" and "Center 2", we can use custom names
loc_names <- c("Fallon", "Yerington")

# Sites report now uses loc_names in the output
SitesReport(centers = loc_mat,
            ebird_key = key,
            species_seen = my_species,
            center_names = loc_names)
```

Multiple sites from a data frame, narrower range, and informative report file name

The `SitesReport` can also accommodate a data frame of latitude and longitude coordinates. You can also give your report a different name (with `report_filename`) and specify where it is saved (with `report_dir`).

```r
loc_df <- data.frame(latitude = c(39.5, 39, 40),
                     longitude = c(-118.8, -119.1, -118.6))
loc_names <- c("Fallon", "Yerington", "Humbolt Wildlife")

# We can set the area to search by passing values to the dist argument
SitesReport(centers = loc_df,
            ebird_key = key,
            dist = 25,
            species_seen = my_species,
            center_names = loc_names,
            report_filename = "Nevada-sites",
            report_dir = "~/Desktop") # Saves report to desktop
```

Helpful tips

- This should go without saying, but you are very unlikely to see *all* of the species that have been identified as being recently observed at a given site. There might be recent observations that result in a Northern Goshawk showing up in the output report, but lifeR makes no guarantees that you will actually see that godforsaken bird.
- Along similar lines, this report makes no claims on the veracity of observations on which the top sites are identified. Difficult to identify species or species only detected by call or song often find their way onto these reports. We leave it to individual users to decide if it is worth the trip to a site where someone claims to have identified a Chihuahuan Raven preening its pale neck feathers amongst an unkindness of Common Ravens.
- The longer the list you use (e.g. your life list, your year list, etc), the fewer sites/species will be identified in the report. If you find you are running `SitesReport` and few or no sites are being returned, try:
  - A shorter, more restrictive list (e.g. use your year list instead of your life list)
  - A larger radius to search (up to a maximum of 50km)
  - Additional centers to start your search from (i.e. one that is 100km from your home base)

How does it work?

The main goal, identifying sites to visit to find species you have yet to see, uses eBird's API to query for recent local observations. Specifically, the `SitesReport` function:

1. Queries eBird to find all species seen within a certain radius of the latitude and longitude coordinates you provide,
2. Compares the list of recently observed species to the list you provide (e.g. your year list), to identify the unseen species,
3. For each unseen species, queries eBird again, this time to find all the nearby *sites* at which that species was recently seen,
4. Identifies the sites with the most unseen species,
5. Creates maps, tables, and the final report.

Perhaps you are asking yourself "what is the difference between steps 1 and 3?" The two separate queries are required because the first only returns the most recent observation of each species for a given region. That is, even though the Black-bellied Whistling Duck has been seen at 14 nearby sites, only the most recent observation (and therefore locality information) is returned by eBird in step 1. It is not until step 3 where we retrieve *all* the sites at which that species was recently seen.

What doesn't the package do?

- **Connect to your eBird account**. The eBird API is designed to query publicly available data about observations. It does not provide a way to query about individual users' accounts. It is for this reason that you are responsible for downloading your species list and feeding it to `SitesReport()`.
- **Include additional measures of confidence**. It would be nice if lifeR could include some estimate of the likelihood of seeing each species based on how many people have seen it or the average number of individuals observed. It *would* be nice, but those data are not available through the eBird API. The only information that is returned by eBird is details for the most recent observation of a particular species at a particular site; that is, there is no means of determining if a species was seen by 100% of the observers on a particular day, or by 5% of the observers.