The functionality of RCompadre and Rage

Pol Capdevila and John Jackson

2021-04-07

## Introduction

The aim of this vignette is to give users a walk through of the functionality of the Rcompadre and Rage R packages.

The Rcompadre package allows users to access and manipulate data from the COMPADRE and COMADRE Plant and Animal Matrix Databases (Salguero-Gomez et al. 2015, 2016). These two databases contain matrix population models (MPMs) and associated metadata obtained from the published and “grey” literature.

The Rage package provides tools for conducting aging- and life history-focussed analysis of MPMs. Rage includes a wide variety of functions the calculation of life history traits (e.g. longevity, generation time), vital rates (e.g. growth, reproduction) and to run perturbation analyses (e.g. vital rate elasticities). Furthermore, the package includes functions to manipulate and transform MPMs and life tables.

## Research question (start of manuscript section)

In the following example we illustrate the use of Rcompadre to carry out typical data download and preparation tasks for an analysis relevant to comparative population dynamics research and implemented using Rage.

Let’s imagine that we are interested in exploring variation in mammal longevity across different ecoregions. The demographic information contained in MPMs can be used to estimate longevity (see SI in Jones et al, 2014) and we can therefore leverage the large taxonomic and geographic coverage of COMADRE to make interesting comparative inferences.

# [RCOMPADRE SECTION BEGINS]

The first task is to install the latest version of Rcompadre. RCompadre is developed on the github platform and can therefore be installed using remotes::install\_github and loaded using library as follows:

install.packages("remotes")  
remotes::install\_github("jonesor/Rcompadre")

After ensuring that the packages are installed we load them:

library(Rcompadre)  
library(tidyverse)

After that, users can download the latest version of the COMPADRE or COMADRE databases from the website (or load from a local file) using the function cdb\_fetch. Note that users can also load local files with the same function x <- cdb\_fetch("PATH/TO/FILE/COMADRE\_v.4.21.1.0.RData"). After loading the data, a message is displayed stating the database version number, date of release, and links to the user agreement and citation request.

comadre <- cdb\_fetch("comadre")  
#> This is COMADRE version 4.21.1.0 (release date Jan\_25\_2021)  
#> See user agreement at https://compadre-db.org/Help/UserAgreement  
#> See how to cite at https://compadre-db.org/Help/HowToCite

### Data management

The COMADRE database includes metadata associated with each matrix including taxonomic information, geolocation, and details of the source publication. A full description of these variables can be found in the User Guide via the website [www.compadre-db.org](http://www.compadre-db.org). We can get an idea of the richness of this metadata simply by using the names command.

names(comadre)

The MPMs are contained in a list column called mat where each element contains a list of the four matrices, **A** and the submatrices **U**, **F** and **C** (see above). The list also includes information on matrix stage definitions. All other columns of the COMADRE database object are ordinary vectors.

We can explore the taxonomic groups available in the dataset.

table(comadre$Class)  
#>   
#> Actinopterygii Adenophorea Amphibia   
#> 196 4 24   
#> Anthozoa Arachnida Ascidiacea   
#> 102 5 18   
#> Aves Bivalvia Branchiopoda   
#> 409 30 7   
#> Cephalaspidomorphi Clitellata Demospongiae   
#> 9 3 13   
#> Diplopoda Echinoidea Elasmobranchii   
#> 1 11 4   
#> Gastropoda Gymnolaemata Insecta   
#> 20 10 73   
#> Malacostraca Mammalia Maxillopoda   
#> 30 2181 11   
#> Polychaeta Reptilia Revtraviricetes   
#> 19 168 2   
#> Scyphozoa Secernentea Spirochaetes   
#> 2 4 9   
#> Thaliacea   
#> 1

We see that COMADRE contains information for many taxonomic groups. However, in this case we are only interested in mammals.Rcompadre allows us to subset the database easily using the function filter from dplyr as if our CompadreDB object was an ordinary data frame.

mammals <- comadre %>%  
 filter(Class == "Mammalia")  
  
table(mammals$Order)  
#>   
#> Artiodactyla Carnivora Cetacea   
#> 137 223 58   
#> Chiroptera Cingulata Dasyuromorphia   
#> 1 3 3   
#> Didelphimorphia Diprotodontia Lagomorpha   
#> 1 28 10   
#> Perissodactyla Primates Proboscidea   
#> 1 1557 3   
#> Rodentia Sirenia   
#> 153 3

### Data checking

Although COMPADRE and COMADRE data undergo rigorous checks before being made available, that doesn’t necessarily mean that all data will meet the exact requirements of any particular analysis. Rcompadre includes several functions for checking the data.

For example, for a minority of studies, it is not possible decompose **A** matrices into the **U** **F** and **C** submatrices. This is potentially important because the calculation of some life history traits requires one or more of these submatrices. For instance, in our example analysis focussing on longevity we require a **U** submatrix with no missing values.

To check, and filter out un-usable matrices, we use the function cdb\_flag and, in this case, specify checks = c("check\_NA\_U","check\_zero\_U") (see ?cdb\_flag for all available data checks). This specifies that we want to check whether the studies contain **U** matrices (check\_NA\_U) and whether these **U** matrices are all zeros (check\_zero\_U).

mammals <- cdb\_flag(mammals, checks = c("check\_NA\_U", "check\_zero\_U"))

The cdb\_flag function will return a version of the data with additional logical columns for each of the checks, which we can summarise using table.

table(mammals$check\_NA\_U)  
#>   
#> FALSE TRUE   
#> 2168 13  
table(mammals$check\_zero\_U)  
#>   
#> FALSE TRUE   
#> 2172 9

Now we can filter our data to include only data with suitable **U** matrices.

mammals <- mammals %>%  
 filter(check\_NA\_U == FALSE, check\_zero\_U == FALSE)

Finally, most COM(P)ADRE data includes geolocation information. In addition to Country and Continent, precise latitude and longitude (Lat and Lon) make it easy to visualise the spatial extent of our data set using tools in the ggplot2 and maps packages (Figure X).

*Figure 2: The spatial extent of data in the subset of mammal data used in our example analysis. Note that 269 of matrices in our set (~12%) do not have associated spatial information.*

globalDist <- ggplot(mammals, aes(x = Lon, y = Lat)) +  
 borders(database = "world", fill = "grey80", col = NA) +  
 geom\_point(alpha = 0.9) +  
 labs(x = NULL, y = NULL) +  
 theme\_minimal()

Now that we are sure that our data set contains the right information, we are ready to calculate longevity using Rage.

# [RCOMPADRE SECTION ENDS]

# [RAGE SECTION BEGINS]

In the following section we illustrate the use of some of these functions, focussing on carrying out the longevity analysis introduced above. Our first task will be to install the package from Github.

install.packages("remotes")  
remotes::install\_github("jonesor/Rage")

Then load it.

library(Rage)

Our goal is to calculate the longevity of mammals from across the world using the dataset we generated above. We can do this using the longevity function, which simulates the ages at death of a synthetic cohort of individuals following the demographic pattern described by the MPM. The function documentation (?longevity) tells us that it requires the **U** matrix, which we can obtain using the matU function from Rcompadre.

The function also requires us to define which stage we consider to be the beginning of life. This is fairly clear for most mammals, but may be contentious in some groups, such as plants with a seed bank. Although it may be desirable to do this manually in some cases, the Rcompadre function mpm\_first\_active automates the task by returning an integer index for the first active stage class, as noted by the original study author.

The longevity function also the argument lx\_crit, which defines the critical threshold of survivorship that defines longevity. The default is 0.01, meaning that longevity is defined as the age at which only 1% of the synthetic cohort remain alive. Finally, the function requires us to set a maximum age to consider (xmax, which defaults to 1000).

Thus, our first steps in this analysis are to (i) extract the **U** matrices with matU, then (ii) extract the first stage with mpm\_first\_active. We add the outputs of these functions to our data using mutate.

mammals <- mammals %>%  
 mutate(  
 matU = matU(.),  
 start\_life = mpm\_first\_active(.)  
 )

Now we can apply the longevity function over our dataset using mapply.

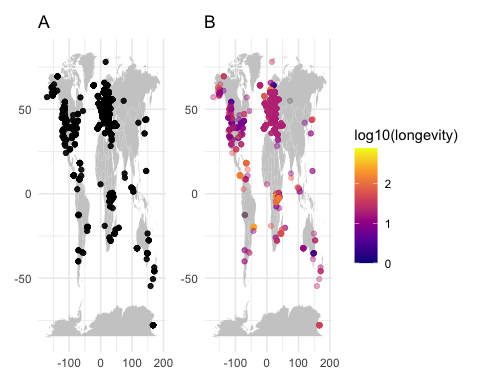
mammals <- mammals %>%  
 mutate(longevity = mapply(longevity, matU, start\_life))

Now that we have the longevity data for all of our mammals we can actually explore its variation across the globe and across different orders of mammals.

First, we will look at how longevity varies spatially. Lets repeat our map from before but now incorporating longevity on the log scale as the colour of each point.

globalDist\_longevity <- ggplot(mammals, aes(Lon, Lat)) +  
 borders(database = "world", fill = "grey80", col = NA) +  
 geom\_point(aes(colour = log10(longevity)), alpha = 0.4) +  
 scale\_colour\_viridis\_c(option = "C") +  
 labs(x = NULL, y = NULL) +  
 theme\_minimal()

#> Warning: Removed 269 rows containing missing values  
#> (geom\_point).  
  
#> Warning: Removed 269 rows containing missing values  
#> (geom\_point).

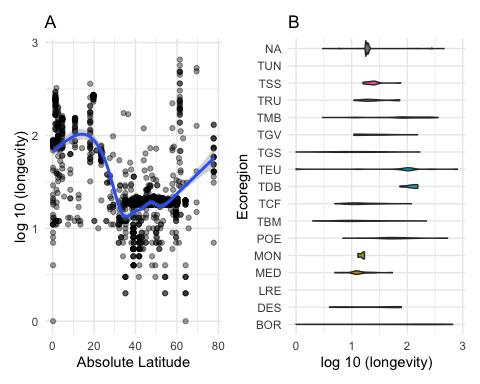


#> Warning: Removed 269 rows containing missing values  
#> (geom\_point).  
  
#> Warning: Removed 269 rows containing missing values  
#> (geom\_point).

Although we see that longevity varies spatially, it isn’t clear whether there are any spatial patterns. Lets look at whether longevity varies with either absolute latitude or across Ecoregion (we could naturally investigate associations with other variables in the metadata).

longevity\_lat <- ggplot(mammals, aes(x = abs(Lat), y = log10(longevity))) +  
 geom\_point(alpha = 0.4) +  
 labs(x = "Absolute Latitude", y = "log 10 (longevity)") +  
 theme\_minimal() +  
 geom\_smooth()  
  
longevity\_ecoregion <- ggplot(mammals, aes(  
 x = Ecoregion, y = log10(longevity),  
 fill = Ecoregion  
)) +  
 geom\_violin(show.legend = FALSE) +  
 labs(x = "Ecoregion", y = "log 10 (longevity)") +  
 coord\_flip() +  
 theme\_minimal()

#> `geom\_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'  
#> Warning: Removed 278 rows containing non-finite values  
#> (stat\_smooth).  
#> Warning: Removed 278 rows containing missing values  
#> (geom\_point).  
#> Warning: Removed 42 rows containing non-finite values  
#> (stat\_ydensity).



#> `geom\_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'  
#> Warning: Removed 278 rows containing non-finite values  
#> (stat\_smooth).  
#> Warning: Removed 278 rows containing missing values  
#> (geom\_point).  
#> Warning: Removed 42 rows containing non-finite values  
#> (stat\_ydensity).

There appears to be a decline in longevity at intermediate latitudes (Figure XA), and there is some variation among ecoregions (Figure XB). These patterns could well be driven by data biases, but would make an interesting topic for future exploration.

With the broad range of tools at our fingertips in Rcompadre and Rage, we can begin to address such detailed comparative questions with increased ease.