The functionality of RCompadre and Rage

Pol Capdevila and John Jackson

2021-03-26

## Introduction

The aim of this vignette is to give the user a walkthrough of integrating the functionality of the Rcompadre and Rage R packages.

Rcompadre is an R package that allows us to access to 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 literature.

On the other hand, Rage provides tools for manipulating and analysing MPMs. More precisely, Rage includes a wide variety of functions allowing us to calculate 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, Rage includes several functions to manipulate and transform MPMs and life tables.

## Research question

**Let’s imagine that we are interested in exploring the variation of longevity in mammals across different regions of the planet**. Collecting comparative data on mammal longevity at a global scale requires significant research effort and time, but thankfully over the years researchers have collected the necessary demographic data to calculate longevity. Where this demographic data has been published in the form of an MPM, we can use information available in COMADRE, which will allow us to make valuable comparative inferences.

## Data acquisition

We can download the COMADRE dataset from the website at [www.compadre-db.org](http://www.compadre-db.org), and use Rcompadre to manipulate the data. It is important to note that Rcompadre coerces the data into a CompadreDB S4 class object. While the specific details of this data type are beyond the scope of this vignette, overall it means that we will be able to manipulate our database using functionality already built into R. Furthermore, because we are going to manipulate the whole database we are going to use the tidyverse package to facilitate the process.

First we load the libraries:

library(Rcompadre)  
library(tidyverse)  
#> ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──  
#> ✓ ggplot2 3.3.3 ✓ purrr 0.3.4  
#> ✓ tibble 3.0.6 ✓ dplyr 1.0.4  
#> ✓ tidyr 1.1.2 ✓ stringr 1.4.0  
#> ✓ readr 1.4.0 ✓ forcats 0.5.0  
#> ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
#> x dplyr::filter() masks stats::filter()  
#> x dplyr::lag() masks stats::lag()

If you have downloaded the data directly from the website, you can load it into R using the command load. Always remember to set up the working directory where your database files are.

load("COMADRE\_v.4.21.1.0.RData")

Then you should ensure that the database is of the correct class using the command as\_cdb:

comadre <- as\_cdb(comadre)

Alternatively, the function cdb\_fetch will automatically download the latest version of COMADRE from the website:

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

You will notice that when you download the data you receive a message stating the version of the database and the date of release, which in our case is This is COMADRE version 4.21.1.0 (release date Jan\_25\_2021).

You can also check the version of the database using the function VersionData().

VersionData(comadre)  
#> $Database  
#> [1] "COMADRE"  
#>   
#> $Version  
#> [1] "4.21.1.0"  
#>   
#> $Type  
#> [1] "Release"  
#>   
#> $DateCreated  
#> [1] "Jan\_25\_2021"  
#>   
#> $TimeCreated  
#> [1] "14:59"  
#>   
#> $Agreement  
#> [1] "https://www.compadre-db.org/UserAgreement"  
#>   
#> $GeneratorScriptVersion  
#> [1] "1.3"

## Data management

The database includes a range of metadata associated with each matrix that includes taxonomic information, geolocation, details of the publication from which the matrix was obtained and so on. A full description of these variables can be found in the User Guide on the website [www.compadre-db.org](http://www.compadre-db.org). However, to avoid the trouble of searching the User Guide, we can obtain a list of the metadata simply by using the names command, in the same way that you would for a data frame.

names(comadre)  
#> [1] "mat" "MatrixID" "SpeciesAuthor"   
#> [4] "SpeciesAccepted" "CommonName" "Kingdom"   
#> [7] "Phylum" "Class" "Order"   
#> [10] "Family" "Genus" "Species"   
#> [13] "Infraspecies" "InfraspeciesType" "OrganismType"   
#> [16] "DicotMonoc" "AngioGymno" "Authors"   
#> [19] "Journal" "SourceType" "OtherType"   
#> [22] "YearPublication" "DOI\_ISBN" "AdditionalSource"   
#> [25] "StudyDuration" "StudyStart" "StudyEnd"   
#> [28] "ProjectionInterval" "MatrixCriteriaSize" "MatrixCriteriaOntogeny"  
#> [31] "MatrixCriteriaAge" "MatrixPopulation" "NumberPopulations"   
#> [34] "Lat" "Lon" "Altitude"   
#> [37] "Country" "Continent" "Ecoregion"   
#> [40] "StudiedSex" "MatrixComposite" "MatrixSeasonal"   
#> [43] "MatrixTreatment" "MatrixCaptivity" "MatrixStartYear"   
#> [46] "MatrixStartSeason" "MatrixStartMonth" "MatrixEndYear"   
#> [49] "MatrixEndSeason" "MatrixEndMonth" "CensusType"   
#> [52] "MatrixSplit" "MatrixFec" "Observations"   
#> [55] "MatrixDimension" "SurvivalIssue" "\_Database"   
#> [58] "\_PopulationStatus" "\_PublicationStatus"

You will notice that each element of the mat column contains a list of the four matrices (**A**, **U**, **F**, **C**) and information on matrix stages while the other columns are ordinary vectors. The **A** matrix represents the full MPM, which has been split into three additional constituent matrices based on the nature of the demographic processes involved: The **U** matrix, which summarises growth and survival; the **F** matrix, which summarises sexual reproduction; and the **C** matrix, which summarises asexual (clonal) reproduction. These additional matrices sum to equal the **A** matrix (**A** = **U** + **F** + **C**).

Given that we are only interested in mammals, we can explore the taxonomic groups available in the dataset.

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

We can see here that C0MADRE contains information for many different taxonomic groups. However, in this case we are only interested in mammals. The functionality in Rcompadre and its data class allows us to subset the database easily. We can simply use the function filter from dplyr as if our cdb object was a data frame.

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

If we want to check if a species is in the database we can use the cdb\_check\_species function. For example, we can ask if the species *Ursus arctos* is present, with the result given as TRUE/FALSE.

cdb\_check\_species(mammals, "Ursus arctos")  
#> species in\_db  
#> 1 Ursus arctos TRUE

Alternatively, using the return\_db argument cdb\_check\_species can return a subset of the database restricted to the matched species name(s).

arctos <- cdb\_check\_species(mammals, "Ursus arctos", return\_db = TRUE)  
arctos  
#> A COM(P)ADRE database ('CompadreDB') object with 1 SPECIES and 17 MATRICES.  
#>   
#> # A tibble: 17 x 59  
#> mat MatrixID SpeciesAuthor SpeciesAccepted CommonName Kingdom Phylum Class  
#> <list> <int> <chr> <chr> <chr> <chr> <chr> <chr>  
#> 1 <Comp… 240677 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 2 <Comp… 240678 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 3 <Comp… 240679 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 4 <Comp… 249895 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 5 <Comp… 249896 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 6 <Comp… 249897 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 7 <Comp… 249898 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 8 <Comp… 249899 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 9 <Comp… 249900 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 10 <Comp… 249901 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 11 <Comp… 249902 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 12 <Comp… 249903 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 13 <Comp… 249904 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 14 <Comp… 249905 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 15 <Comp… 249906 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 16 <Comp… 249907 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> 17 <Comp… 249908 Ursus\_arctos… Ursus arctos Brown bear Animal… Chord… Mamm…  
#> # … with 51 more variables: Order <chr>, Family <chr>, Genus <chr>,  
#> # Species <chr>, Infraspecies <chr>, InfraspeciesType <chr>,  
#> # OrganismType <chr>, DicotMonoc <chr>, AngioGymno <chr>, Authors <chr>,  
#> # Journal <chr>, SourceType <chr>, OtherType <chr>, YearPublication <chr>,  
#> # DOI\_ISBN <chr>, AdditionalSource <chr>, StudyDuration <chr>,  
#> # StudyStart <chr>, StudyEnd <chr>, ProjectionInterval <chr>,  
#> # MatrixCriteriaSize <chr>, MatrixCriteriaOntogeny <chr>,  
#> # MatrixCriteriaAge <chr>, MatrixPopulation <chr>, NumberPopulations <dbl>,  
#> # Lat <dbl>, Lon <dbl>, Altitude <dbl>, Country <chr>, Continent <chr>,  
#> # Ecoregion <chr>, StudiedSex <chr>, MatrixComposite <chr>,  
#> # MatrixSeasonal <chr>, MatrixTreatment <chr>, MatrixCaptivity <chr>,  
#> # MatrixStartYear <chr>, MatrixStartSeason <chr>, MatrixStartMonth <chr>,  
#> # MatrixEndYear <chr>, MatrixEndSeason <chr>, MatrixEndMonth <chr>,  
#> # CensusType <chr>, MatrixSplit <chr>, MatrixFec <chr>, Observations <chr>,  
#> # MatrixDimension <int>, SurvivalIssue <dbl>, \_Database <chr>,  
#> # \_PopulationStatus <chr>, \_PublicationStatus <chr>

## Data checking

Even if the data in COMPADRE and COMADRE is checked before being made publicly available (see User Guide in [www.compadre-db.org](http://www.compadre-db.org)), that doesn’t necessarily mean that the data meet our requirements. Rcompadre incorporates a wide array of functions allowing us to check the data.

As we mentioned above, each element of the mat column within our database contains a list of the four matrices (**A**, **U**, **F**, **C**) and information on matrix stages. However, not all studies include the necessary information to decompose each **A** matrix into the different sub-matrices. This is important because different sub-matrices are required to calculate different life history traits. For instance, in our case here, because we are focused on longevity we need to be sure that the sub-matrix **U** is available.

To do this we will use the function cdb\_flag, and we will specify checks = c("check\_NA\_U","check\_zero\_U"). 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). For the full list of these checks, please refer to the cdb\_flag documentation.

mammals\_flag <- cdb\_flag(mammals,checks = c("check\_NA\_U","check\_zero\_U"))  
names(mammals\_flag)  
#> [1] "mat" "MatrixID" "SpeciesAuthor"   
#> [4] "SpeciesAccepted" "CommonName" "Kingdom"   
#> [7] "Phylum" "Class" "Order"   
#> [10] "Family" "Genus" "Species"   
#> [13] "Infraspecies" "InfraspeciesType" "OrganismType"   
#> [16] "DicotMonoc" "AngioGymno" "Authors"   
#> [19] "Journal" "SourceType" "OtherType"   
#> [22] "YearPublication" "DOI\_ISBN" "AdditionalSource"   
#> [25] "StudyDuration" "StudyStart" "StudyEnd"   
#> [28] "ProjectionInterval" "MatrixCriteriaSize" "MatrixCriteriaOntogeny"  
#> [31] "MatrixCriteriaAge" "MatrixPopulation" "NumberPopulations"   
#> [34] "Lat" "Lon" "Altitude"   
#> [37] "Country" "Continent" "Ecoregion"   
#> [40] "StudiedSex" "MatrixComposite" "MatrixSeasonal"   
#> [43] "MatrixTreatment" "MatrixCaptivity" "MatrixStartYear"   
#> [46] "MatrixStartSeason" "MatrixStartMonth" "MatrixEndYear"   
#> [49] "MatrixEndSeason" "MatrixEndMonth" "CensusType"   
#> [52] "MatrixSplit" "MatrixFec" "Observations"   
#> [55] "MatrixDimension" "SurvivalIssue" "\_Database"   
#> [58] "\_PopulationStatus" "\_PublicationStatus" "check\_NA\_U"   
#> [61] "check\_zero\_U"

Notice that we now have two new columns in our dataset named check\_NA\_U and check\_zero\_U.

head(mammals\_flag$check\_NA\_U)  
#> [1] FALSE FALSE TRUE TRUE FALSE FALSE  
head(mammals\_flag$check\_zero\_U)  
#> [1] FALSE FALSE FALSE FALSE FALSE FALSE

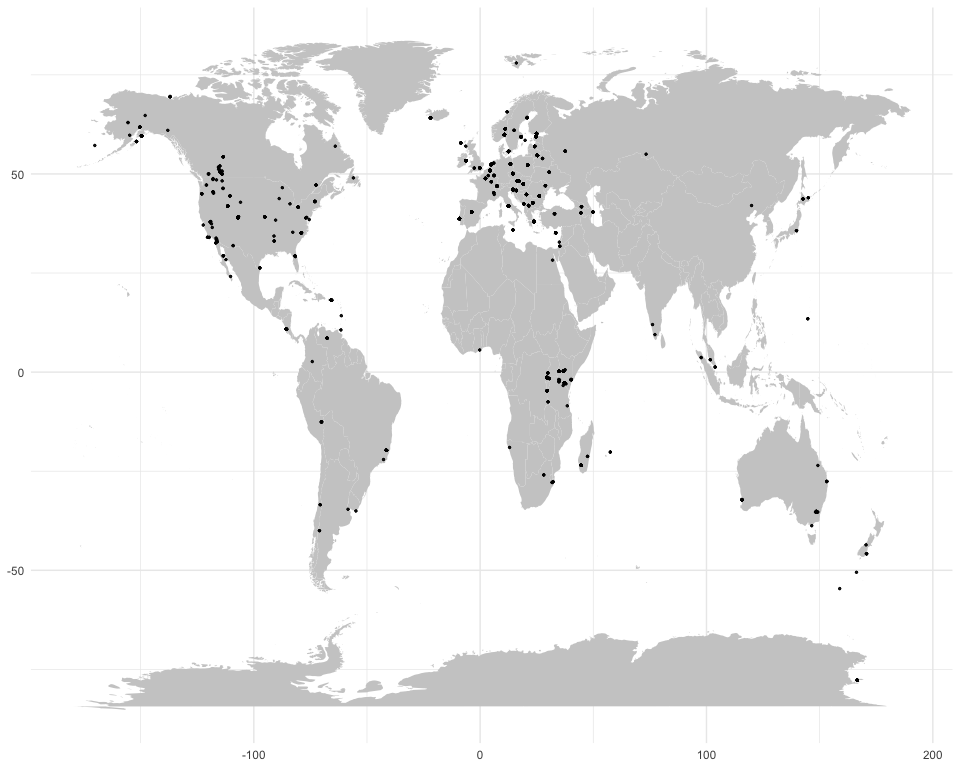
These new columns are TRUE/FALSE vectors indicating whether the **U** matrix is empty (TRUE) or not (FALSE) for $check\_NA\_U, and whether the **U** matrix has all zeros (TRUE) or not (FALSE) for $check\_zero\_U. Now we can easily subset those studies containing **U** matrices with information using filter.

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

Note that there are a wide variety of data checks that one can do with Rcompadre, but going through all of them is beyond the scope of this vignette. However, we encourage users to explore a wide range of these checks for their own research questions.

Finally, COMPADRE and COMADRE often contain information about the geolocation of each study, and this spatial information is an important component of comparative studies. This information is stored in the $Lat and $Lon columns. Using the handy additions to ggplot2 from the maps package, we can visualise this spatial information very easily. Here, each spatial point is from one of the entries in our mammals\_with\_u data. Note that not all entries (269 or ~10%) have spatial information.

ggplot(mammals\_with\_u, aes(x = Lon, y = Lat)) +  
 borders(database = "world", fill = "grey80", col = NA) +  
 geom\_point(size = 0.5, alpha = 0.9) +  
 labs(x = NULL, y = NULL) +  
 theme\_minimal()  
#> Warning: Removed 269 rows containing missing values (geom\_point).



## Data analyses

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

library(Rage)

If we remember, our goal is to calculate the longevity of mammals from across the world. This is made easy using Rage, which has a function to calculate longevity from MPMs called longevity. If we look at the function ?longevity we can see that it requires the matrix **U**, which we already have thanks to Rcompadre. All we need to do is extract the **U** matrix alone from the $mat column, which we can do using the matU function from Rcompadre.

The function also requires information about the first stage i.e. the point that the user considers the beginning of life. In this comparative example, we need a way of calculating this first stage as considered by the original authors for each of the matrices. While we could spend time doing this manually, Rcompadre has a function that can do it for us called mpm\_first\_active. This function returns an integer index giving the first active stage class noted by the author in the study.

Finally, the longevity function also allows us to set a maximum age using xmax (defaults to 1000) and the proportion of the initial cohort remaining before all are considered dead, lx\_crit, a critical proportion that defaults to 0.01. We will leave these at their defaults for the purposes of this example.

We will combine this workflow here, extracting all the **U** matrices with matU() and then extracting the first stage with mpm\_first\_active, applied to all the entries of our mammal data with mutate.

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

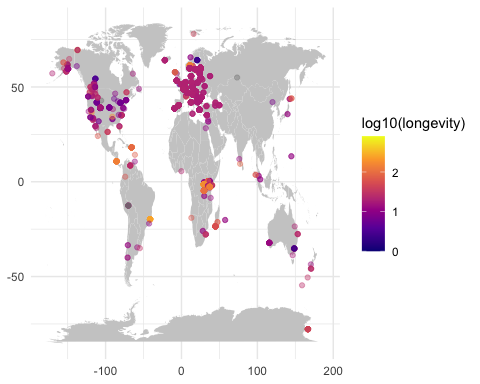
Now we can apply the longevity function over our dataset using mapply.

mammals\_long <- mammals\_with\_u %>%   
 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.

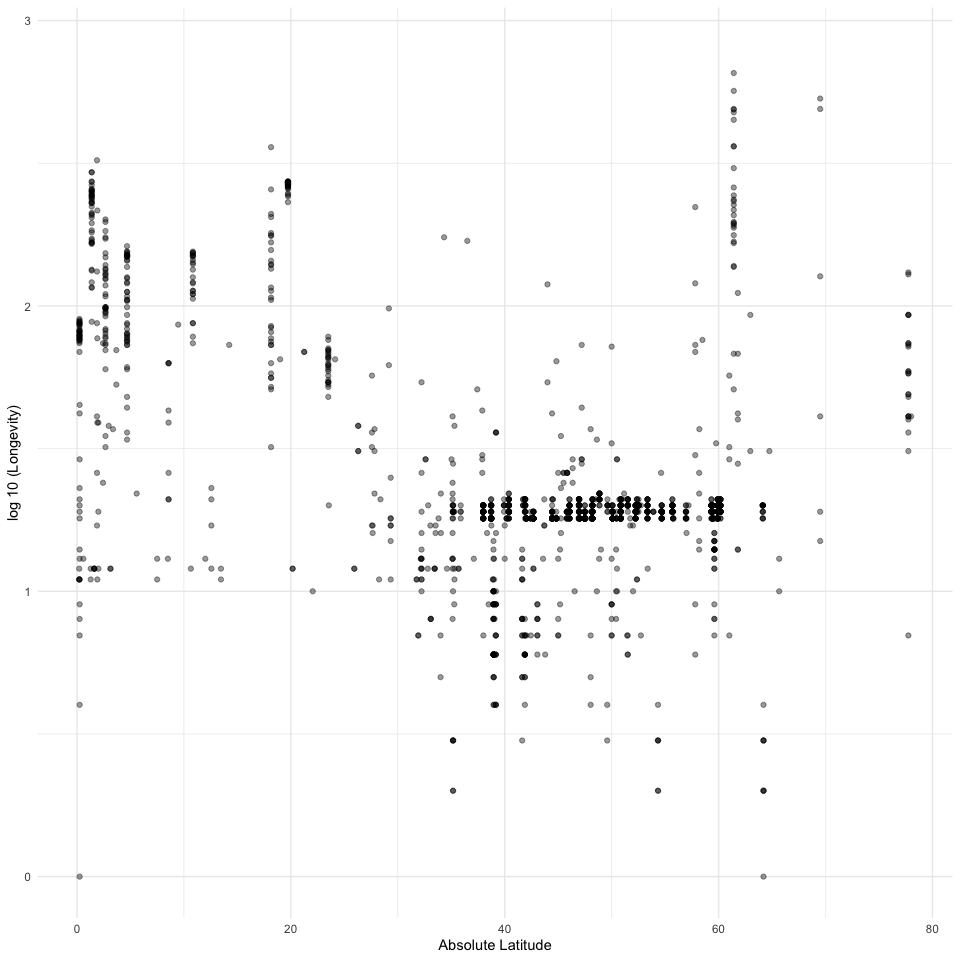
First, we will look at how longevity varies spatially. Lets repeat our map from before but now incorporating longevity on the log scale as the colour of each point.

ggplot(mammals\_long, 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).

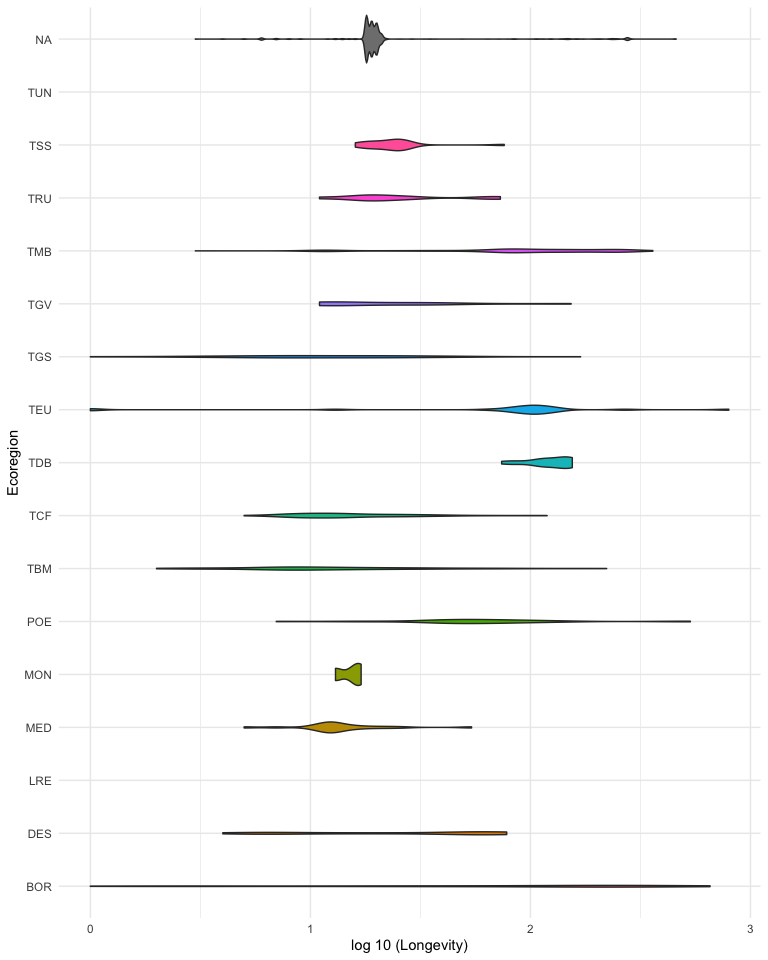


Although we can see the variability of longevity spatially, it isn’t clear if there are any spatial patterns here. Lets look at whether longevity varies with either absolute latitude or across Ecoregion codes (also stored in Rcompadre).

ggplot(mammals\_long, aes(x = abs(Lat), y = log10(longevity))) +  
 geom\_point(alpha = 0.4) +  
 labs(x = "Absolute Latitude", y = "log 10 (Longevity)") +  
 theme\_minimal()  
#> Warning: Removed 278 rows containing missing values (geom\_point).

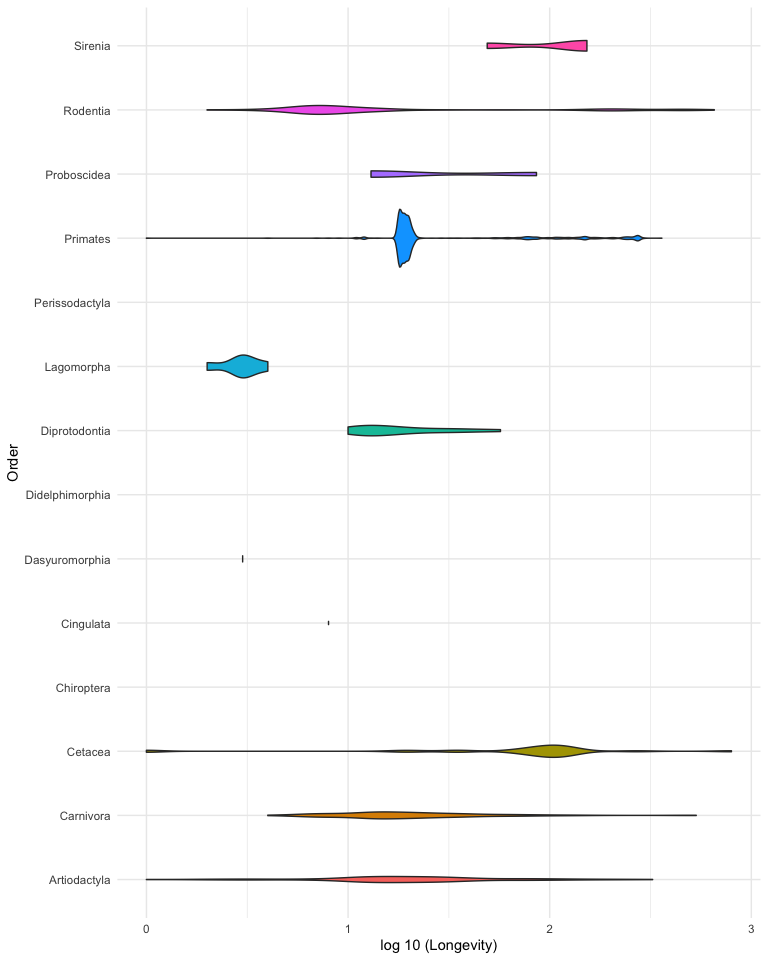


ggplot(mammals\_long, aes(x = Ecoregion, y = log10(longevity),  
 fill = Ecoregion)) +  
 geom\_violin(show.legend = F) +  
 labs(x = "Ecoregion", y = "log 10 (Longevity)") +  
 coord\_flip() +  
 theme\_minimal()  
#> Warning: Removed 42 rows containing non-finite values (stat\_ydensity).



Finally, we will look at how longevity varies across the different orders of the mammals.

ggplot(mammals\_long, aes(x = Order, y = log10(longevity),  
 fill = Order)) +  
 geom\_violin(show.legend = F) +  
 labs(x = "Order", y = "log 10 (Longevity)") +  
 coord\_flip() +  
 theme\_minimal()  
#> Warning: Removed 42 rows containing non-finite values (stat\_ydensity).



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