How to compute functional rarity indices using `funrar`

Matthias Grenié

21 mai 2018

This document aims to guide you through the use of `funrar` to compute functional rarity indices. The aim is to reproduce the analyses from (Grenié et al. 2017) by using the Mammals dataset published on Dryad (Lawing et al. 2016) from (Lawing et al. 2017). We are going to describe the functional rarity of North American mammals from atlas data. Before running the analysis be sure to install the different packages and download the needed datasets by running the following chunk:

```r
# Install package `devtools` to install development version of `funrar`
if (!require("devtools")) {
    install.packages("devtools")
}

# Install development version of funrar
devtools::install_github("Rekyt/funrar", build_vignettes = TRUE)

# Install development version of `rdryad` if missing to download the example
# dataset
devtools::install_github("ropensci/rdryad")

# Install `ggplot2` if missing
if (!require("ggplot2")) {
    install.packages("ggplot2")
}

# Create a tutorial directory in the current directory
if (!dir.exists("funrar_tutorial")) {
    dir.create("funrar_tutorial")
}

# Move to tutorial directory
setwd("funrar_tutorial")

if (!dir.exists("data")) {
    dir.create("data")
}

# Download the example dataset from Lawning et al. 2016
# <doi:10.5061/dryad.9t0n8>
tempfile("data/lawning_data.zip")
rdryad::dryad_fetch(
    rdryad::dryad_files(rdryad::handle2doi("10255/dryad.116171"))[1],
    destfile = "data/lawning_data.zip", mode = "wb")
unzip("data/lawning_data.zip", exdir = "data")
unlink("data/lawning_data.zip")
```

Load Data

Now that all the necessary packages are installed and the example dataset downloaded, we can begin by loading and describing the files in the archive. Lawing et al. (2016) mentioned in its an atlas dataset of North American mammals. The trait matrix is in the `Traits.csv` file. It contains the trait information for all the mammal from the Jones et al. (2009) database.

```
trait_matrix = read.csv("data/DryadArchive/Traits.csv")

head(trait_matrix, 2)
```

```
##             TaxonName X1_1_ActivityCycle X5_1_AdultBodyMass_g
## 1 Caluromys derbianus                  1             5.789104
## 2 Caluromys philander                  1             5.507240
##   X8_1_AdultForearmLen_mm X13_1_AdultHeadBodyLen_mm X2_1_AgeatEyeOpening_d
## 1                      NA                  5.442418                     NA
## 2                      NA                  5.412851               4.317488
##   X3_1_AgeatFirstBirth_d X18_1_BasalMetRate_mLO2hr X5_2_BasalMetRateMass_g
## 1                     NA                   5.37115                5.840642
## 2               5.812876                        NA                      NA
##   X6_1_DietBreadth X7_1_DispersalAge_d X9_1_GestationLen_d
## 1                3                  NA                  NA
## 2                4                  NA            3.177637
##   X12_1_HabitatBreadth X22_1_HomeRange_km2 X22_2_HomeRange_Indiv_km2
## 1                    1                  NA                        NA
## 2                    2                  NA                        NA
##   X14_1_InterbirthInterval_d X15_1_LitterSize X16_1_LittersPerYear
## 1                         NA         1.144223                   NA
## 2                   12.90349         1.430311            0.6931472
##   X17_1_MaxLongevity_m X5_3_NeonateBodyMass_g X13_2_NeonateHeadBodyLen_mm
## 1             4.094345                     NA                          NA
## 2             4.330733              -1.609438                    2.302585
##   X10_1_PopulationGrpSize X23_1_SexualMaturityAge_d X10_2_SocialGrpSize
## 1                      NA                  2.351229                   0
## 2                      NA                  2.398134                   0
##   X24_1_TeatNumber X12_2_Terrestriality X6_2_TrophicLevel
## 1               NA                    2                 2
## 2          1.94591                    2                 2
##   X25_1_WeaningAge_d X5_4_WeaningBodyMass_g X13_3_WeaningHeadBodyLen_mm
## 1                 NA                     NA                          NA
## 2           4.927624               3.135494                          NA
##   XHYP_JTE
## 1       NA
## 2       NA
```

This table contains the traits of species from the Jones et al. (2009) database, with only quantitative traits. In order to compute functional rarity we need to choose a subset of traits to keep a high number of species while using enough traits to estimate functional rarity. In order to get similar results to (Grenié et al. 2017), we select the following traits: adult body mass, litter size, diest breadth, habitat breadth and terrestriality.

```
# Filter observation that are not NA in the selected columns
trait_subset = subset(trait_matrix, !is.na(trait_matrix$X5_1_AdultBodyMass_g) &
                          !is.na(trait_matrix$X15_1_LitterSize) &
                          !is.na(trait_matrix$X6_1_DietBreadth) &
                          !is.na(trait_matrix$X6_2_TrophicLevel) &
                          !is.na(trait_matrix$X12_1_HabitatBreadth) &
                          !is.na(trait_matrix$X12_2_Terrestriality))

# Conserve only the necessary columns
trait_subset = trait_subset[, colSums(is.na(trait_subset)) == 0]

head(trait_subset)
```

```
##               TaxonName X5_1_AdultBodyMass_g X6_1_DietBreadth
## 1   Calurumys derbianus             5.789104                3
## 2   Calurumys philander             5.507240                4
## 3   Chironectes minimus             6.881750                4
## 4 Didelphis marsupialis             7.034168                3
## 5  Didelphis virginiana             7.800605                4
## 6      Marmosa mexicana             3.893044                3
##   X12_1_HabitatBreadth X15_1_LitterSize X12_2_Terrestriality
## 1                    1         1.144223                    2
## 2                    2         1.430311                    2
## 3                    3         1.175573                    1
## 4                    2         1.958685                    2
## 5                    2         2.154085                    2
## 6                    2         2.451005                    2
##   X6_2_TrophicLevel
## 1                 2
## 2                 2
## 3                 2
## 4                 2
## 5                 2
## 6                 2
```

Then we need to load one `PresAbsMatrixXXX.csv` file, which contains the presence-absence matrix with the different species. To make computation quicker and avoid clogging the RAM, we load the coarser resolution presence-absence matrix `PresAbsMatrix250.csv`.

```
pres_matrix = read.csv("data/DryadArchive/PresAbsMatrix250.csv")

pres_matrix[1:5, 1:5]
```

```
##   GlobalID Longitude Latitude Didelphis_virginiana Aplodontia_rufa
## 1   103152  -26.8586 83.26264                    0               0
## 2   103677  -93.7032 81.01685                    0               0
## 3   103682  -79.3204 81.01685                    0               0
## 4   103692  -50.5548 81.01685                    0               0
## 5   103697  -36.1720 81.01685                    0               0
```

We need then to select only species for which we have traits:

```
# Select only the subset species for which we have selected the traits
pres_subset = pres_matrix[, c(c("GlobalID", "Longitude", "Latitude"),
                          gsub(" ", "_", trait_subset$TaxonName))]
```

Compute distance matrix

Now that we have loaded the data and selected the traits, we can compute the functional distance/dissimilarity matrix that will be used in the computation of functional rarity indices.

Before computing the distance matrix we have to make sure that the trait matrix is well formatted, with row names corresponding to species names and with only traits needed for distance matrices.

```
library("funrar")

trait_format = trait_subset
rownames(trait_format) = gsub(" ", "_", trait_format$TaxonName)
trait_format = trait_format[, -1]

trait_distance = compute_dist_matrix(trait_format, metric = "gower")
```

```
## Warning in compute_dist_matrix(trait_format, metric = "gower"): Only
## numeric traits provided, consider using euclidean distance.
```

```
## Warning in cluster::daisy(traits_table, metric = metric): binary
## variable(s) 5 treated as interval scaled
```

The `compute_dist_matrix()` function is a wrapper around the `daisy()` function in package `cluster`. By default it computes Gower's distance (1971), but the distance can be changed using the `metric` argument. For example to compute euclidean trait distance, it is possible to use `compute_dist_matrix(metric = "euclidean")`. Otherwise any distance/dissimilarity method can used as long as the distance matrix is of `matrix` type with column and row names corresponding to species/individual/genotype names.

When working with mixed traits (continuous, discrete, ordinal, nominal and others), another distance metric that can be considered is the extended Gower distance (Pavoine et al. 2009), computable using the `dist.ktab()` function of the `ade4` package.

WARNING: Make sure that column types correspond to trait values before computing distances (ordinal traits as `ordered` factor for example). Also, if you are using a distance metric that is not scaled between 0 and 1, the indices won't be scaled between 0 and 1, so if you use the euclidean distance, be sure to scale between 0 and 1 before using it in functional rarity metrics.

Compute rarity indices

Notes on format

In `funrar` all the functions to compute rarity indices are provided in two format depending on the presence-absence matrix. The presence-absence data can be provided as a matrix using the "regular" version of functions. The most common format is a presence-absence matrix, such a matrix can be difficult to load using R. It has recently been suggested that the "tidy" format should be used to structure data (Wickham 2014). In tidy format, the site-species matrix has three columns: one for species, one for site and one for abundances, each row is an observation of a given species in a given site with a given abundance.

All indices functions exist in a regular form that uses the matrix format. All of them exist in a tidy or stack version using a suffix `_stack()` (and `_tidy()`).

Distinctiveness (Local Trait Rarity)

The `funrar` package computes the indices following (Violle, Thuiller, Mouquet, Munoz, et al. 2017a). Thus functional distinctiveness, i.e., the local trait rarity is defined as follow:

$$D_i = \frac{\sum_{j=1,j\neq i}^{N} d_{ij}}{N-1}$$

with i the focal species, N the total number of species in the given community and d_{ij} the dissimarity/distance coefficient between species i and j. When using abundances the functional distinctiveness is defied as:

$$D_{i,ab} = \frac{\sum_{j=1,j\neq i}^{N} d_{ij} \times A_j}{\sum_{j=1,j\neq i}^{N} A_j}$$

with A_j the relative abundance of species j the focal community.

In the package the matrix version of the function `distinctiveness()` computes distinctiveness:

```
# The site-species matrix should be formatted beforehand
pres_mat = pres_subset
rownames(pres_mat) = pres_mat$GlobalID  # Name sites
pres_mat = pres_mat[, -c(1:3)]          # Take out sites and coordinates
pres_mat = as.matrix(pres_mat)

mamm_di = distinctiveness(pres_mat, trait_distance)
```

```
## Warning in distinctiveness(pres_mat, trait_distance): Some communities had a single species in them
## Computed value assigned to 'NaN'
```

```
str(mamm_di)
```

```
##  num [1:384, 1:265] NA NA NA NA NA NA NA NA NA NA ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:384] "103152" "103677" "103682" "103692" ...
##   ..$ : chr [1:265] "Caluromys_derbianus" "Caluromys_philander" "Chironectes_minimus" "Didelphis_mar
```

Returned object is a `matrix`, with `NA` when species is absent of the community. The warning message tells use that some communities had a single species them, thus they had `NaN`, to make them differ from absent species with `NA` values. Either you can use the matrix format or transform it to a data frame using the `matrix_to_stack()` function where you indicate the name of the column that will contain the values, one for the row names and one for the column names:

```
df_di = matrix_to_stack(mamm_di,
                        value_col  = "di",      # Name of the column with values
                        row_to_col = "site",    # Name of column with row names
                        col_to_col = "species") # Name of column with col names

head(df_di)
```

```
##               species   site        di
## 1 Caluromys_derbianus 172743 0.2949394
## 2 Caluromys_derbianus 172748 0.3010718
## 3 Caluromys_derbianus 172753 0.2995905
## 4 Caluromys_derbianus 180471 0.3099495
## 5 Caluromys_derbianus 188340 0.2823999
## 6 Chironectes_minimus 172743 0.3598202
```

WARNING: When using an abundance matrix, it is important to change matrix to relative abundances using the `make_relative()` function, because D_i formula uses relative abundances.

Uniqueness (Regional Trait Rarity)

Similarly uniqueness U_i is defined following (Violle, Thuiller, Mouquet, Munoz, et al. 2017a) definition:

$$U_i = \min(d_{ij}), \forall j \in [1; N], j \neq i$$

with d_{ij} the functional distance between species i and j with $j \neq i$, N being the number of species at the global level.

Uniqueness is thus the nearest neighbor distance in the functional distance from the focal species considering all species present in the global functional space. You can use `funrar` to compute uniqueness using `uniqueness()` and `uniqueness_stack()` function:

```
mamm_ui = uniqueness(pres_mat, trait_distance)

head(mamm_ui)
```

```
##                  species         Ui
## 1   Caluromys_derbianus 0.06074864
## 2   Caluromys_philander 0.01051369
## 3   Chironectes_minimus 0.07132308
## 4 Didelphis_marsupialis 0.04744362
## 5  Didelphis_virginiana 0.04744362
## 6      Marmosa_mexicana 0.01630376
```

The output object is a data.frame containing species names and uniqueness values in `Ui` column.

Scarcity (Local Geographical Rarity)

Scarcity measures the local rarity of a species, in terms of relative abundance, it is defined as:

$$S_i = \exp(-N \times \ln 2 \times A_i)$$

where N is the number of species of the local community and A_i the relative abundance of the focal species i. S_i increases towards one when the focal species has a small relative abundance, and decreases towards zero when the species is dominant in a species rich community. When all species have the same relative abundance – i.e., $A_i = 1/N$ –, $S_i = 0.5$.

Our example dataset does not contain abundances but you can compute scarcity using the same functions as before `scarcity()` and `scarcity_stack()`.

NOTE: When computing scarcity, it is important to use **relative abundances** as input. Thus before using `scarcity()` on a matrix you should convert it to a relative abundances matrix using `make_relative()`.

Restrictedness (Regional Geographical Rarity)

Restrictedness aims to measure the regional range of a species, for the moment it is defined as follow:

$$R_i = 1 - \frac{K_i}{K_{tot}}$$

with R_i the restrictedness of species i, K_i the number of sites where species i occur and K_{tot} the total number of sites.

You can compute restrictedness providing only the site-species matrix to `restrictedness()` and `restrictedness_stack()` functions:

```
df_ri = restrictedness(pres_mat)

head(df_ri)
```

```
##                  species        Ri
## 1   Caluromys_derbianus 0.9869792
## 2   Caluromys_philander 1.0000000
## 3   Chironectes_minimus 0.9869792
## 4 Didelphis_marsupialis 0.9687500
## 5  Didelphis_virginiana 0.7630208
## 6      Marmosa_mexicana 0.9739583
```

The output is a data.frame with a single value of restrictedness per species in the `Ri` column. See alternative options in the notes section below.

Notes on computing rarity indices

How to compute global Di or local Ui?

Sometimes it can be interesting to compute distinctiveness at the global level. Because distinctiveness is meant to be mesured at the local-level normally one solution is to "trick" the `funrar` package by creating a fake community with all the species present:

```
# Take the first line of presence matrix, note the argument `drop = FALSE` to
# keep matrix structure
global_com = data.frame(species = colnames(pres_mat),
                        site    = "global")

global_di = distinctiveness_stack(global_com, "species", "site",
                                  dist_matrix = trait_distance)
```

```
## No relative abundance provided, computing Di without it
```

```
head(global_di)
```

```
##                 species   site        Di
## 1   Caluromys_derbianus global 0.3076991
## 2   Caluromys_philander global 0.2949208
## 3   Chironectes_minimus global 0.3107858
## 4 Didelphis_marsupialis global 0.3172887
## 5  Didelphis_virginiana global 0.3418078
## 6      Marmosa_mexicana global 0.3496853
```

Similarly you may want to compute uniqueness at the site-level. In this case the code is a little trickier, as each site should be considered a pool. Also you have to choose if you consider the functional space of each local community or a global one. Here is an example while keeping the same global functional space:

```
# Subset sites where there are more than one species
subset_mat = pres_mat[rowSums(pres_mat) > 1,]

com_ui = apply(subset_mat, 1,
               function(x, dist_m = trait_distance) {
                   # Subselect only present species
                   present_sp = x[x > 0 & !is.na(x)]
                   present_sp = as.matrix(present_sp)
                   present_sp = t(present_sp)

                   # Compute local uniqueness
                   suppressMessages(uniqueness(present_sp, dist_m))})
str(head(com_ui, 2))
```

```
## List of 2
##  $ 103152:'data.frame':  2 obs. of  2 variables:
```

```
##   ..$ species: Factor w/ 2 levels "Mustela_erminea",..: 1 2
##   ..$ Ui     : num [1:2] 0.414 0.414
##  $ 103692:'data.frame':  2 obs. of  2 variables:
##   ..$ species: Factor w/ 2 levels "Canis_lupus",..: 1 2
##   ..$ Ui     : num [1:2] 0.085 0.085
```

Better perforamnces: using sparse matrices

Alternative measure of Ri

By default and as it was suggested in (Violle, Thuiller, Mouquet, Munoz, et al. 2017b) restrictedness, R_i is computed as follow:

$$R_i = 1 - \frac{K_i}{K_{tot}}$$

with K_i the number of sites where species i occur and K_{tot} the total number of sites. As the species occupies less and less sites, R_i tends to one. The problem here is that R_i equals one only for a species that occupies no site in the dataset, which makes no sense. Thus we can standardize R_i on the value of restrictedness for species occupying a single site R_{one}:

$$R_i^* = \frac{R_i}{R_{one}} = \frac{1 - \frac{K_i}{K_{tot}}}{1 - \frac{1}{K_{tot}}} = \frac{K_{tot} - K_i}{K_{tot} - 1}$$

with R_i^* the new version of R_i. In this case, when a species occupies a single site, R_i equals zero, while it equals one when occupying all the sites.

It is possible to use this way of computing restrictedness using the argument `relative = TRUE` in `restrictedness()` and `restrictedness_stack()` functions.

All at once

The `funrar()` and `funrar_stack()` functions let you compute all indices at once. It outputs a list with all the indices indexed (`Ui` = uniqueness, `Di` = distinctiveness, `Ri` = restrictedness).

```r
all_indices = funrar(pres_mat, trait_distance)
```

```
## Warning in distinctiveness(pres_matrix, dist_matrix): Some communities had a single species in them
## Computed value assigned to 'NaN'
```

```r
str(all_indices, max.level = 1)
```

```
## List of 3
##  $ Ui:'data.frame':  265 obs. of  2 variables:
##  $ Di: num [1:384, 1:265] NA NA NA NA NA NA NA NA NA NA ...
##   ..- attr(*, "dimnames")=List of 2
##  $ Ri:'data.frame':  265 obs. of  2 variables:
```

Note on parallelism

The `funrar` package does not include functions to be run in parallel. However, the computation of indices can be parallelized by splitting site-species matrix in sub-matrices and computing the indices on those parts, using the same trait distance matrix.

Visualization

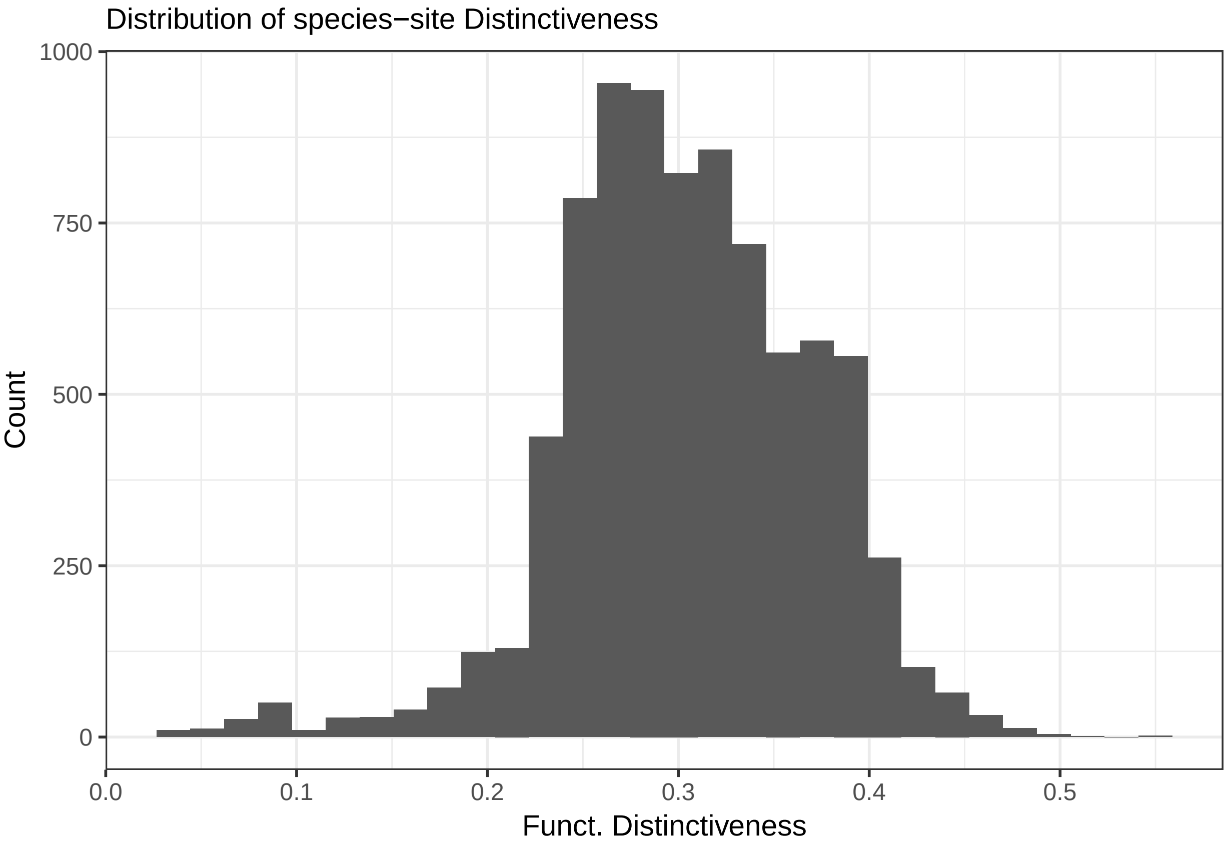
Indices distribution

```
library("ggplot2")

dist_di = ggplot(df_di, aes(di)) +
    geom_histogram() +
    labs(x = "Funct. Distinctiveness", y = "Count",
         subtitle = "Distribution of species-site Distinctiveness") +
    theme_bw()

dist_di
```

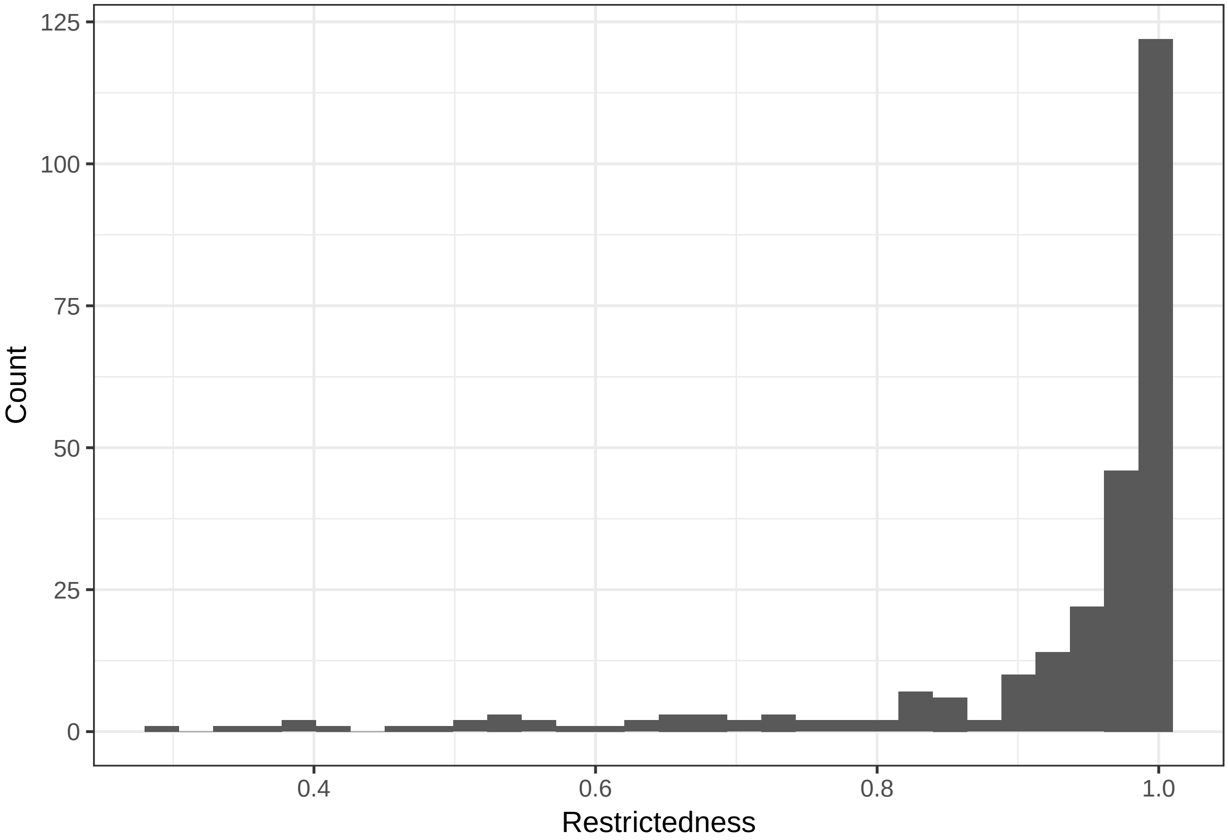
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
dist_ri = ggplot(df_ri, aes(Ri)) +
    geom_histogram() +
    labs(x = "Restrictedness", y = "Count") +
    theme_bw()

dist_ri
```

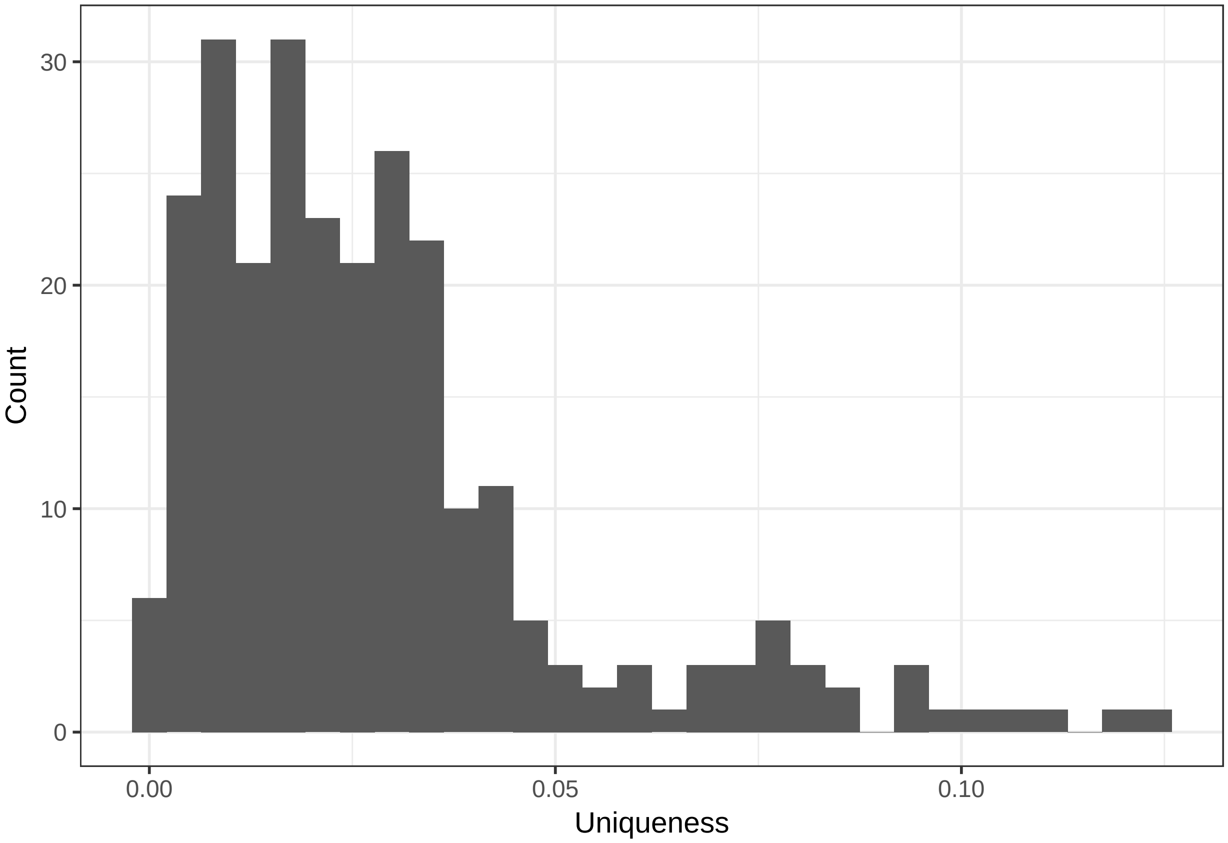
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
dist_ui = ggplot(mamm_ui, aes(x = Ui)) +
    geom_histogram() +
    labs(x = "Uniqueness", y = "Count") +
    theme_bw()

dist_ui
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Indices correlations

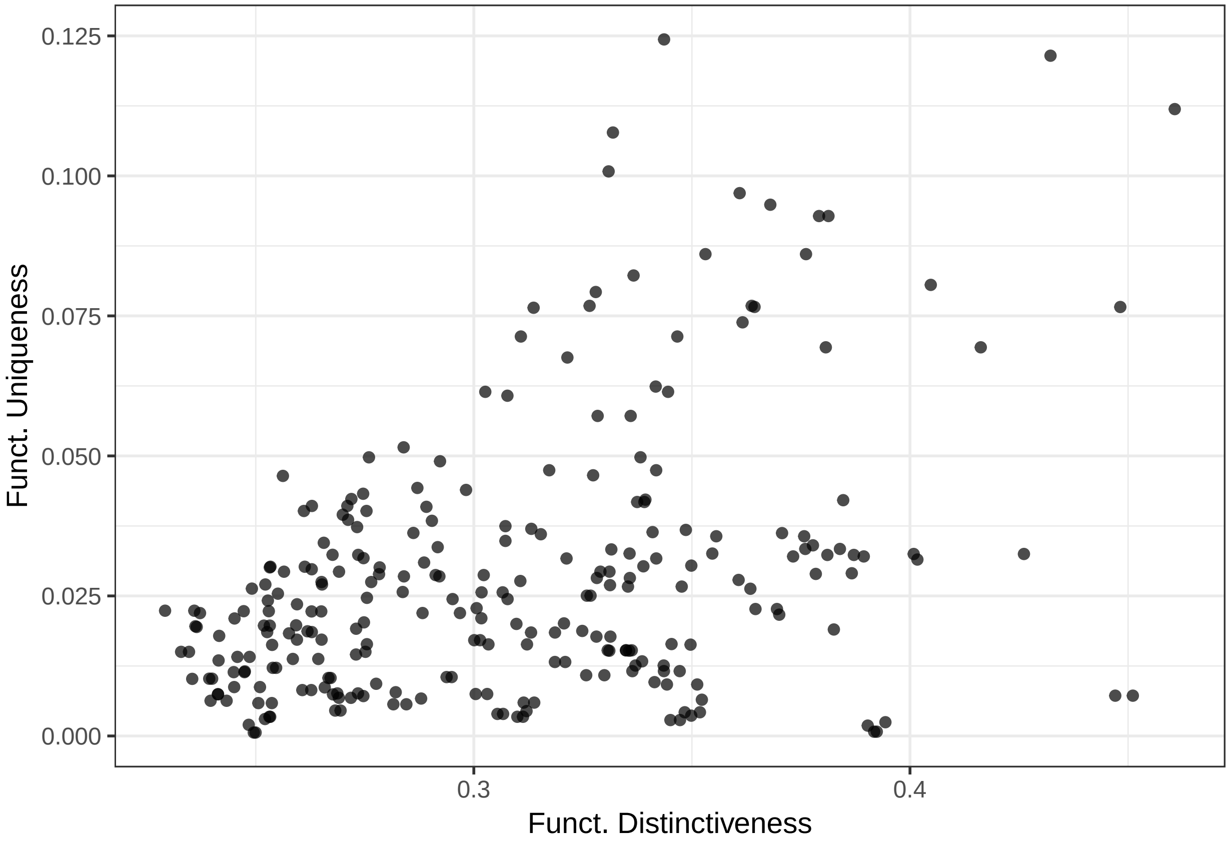
```
mamm_indices = merge(df_ri, global_di[, c(1, 3)], by = "species")
mamm_indices = merge(mamm_indices, mamm_ui, by = "species")

head(mamm_indices)
```

```
##                        species        Ri        Di           Ui
## 1           Alouatta_palliata 0.9895833 0.3903147 0.0018494850
## 2              Alouatta_pigra 0.9895833 0.3917859 0.0007629465
## 3 Ammospermophilus_interpres 0.9869792 0.2784100 0.0300946580
## 4  Ammospermophilus_leucurus 0.9557292 0.2820863 0.0078079717
## 5   Ammospermophilus_nelsoni 1.0000000 0.2879160 0.0066825931
## 6      Antilocapra_americana 0.9192708 0.3206707 0.0200992421
```

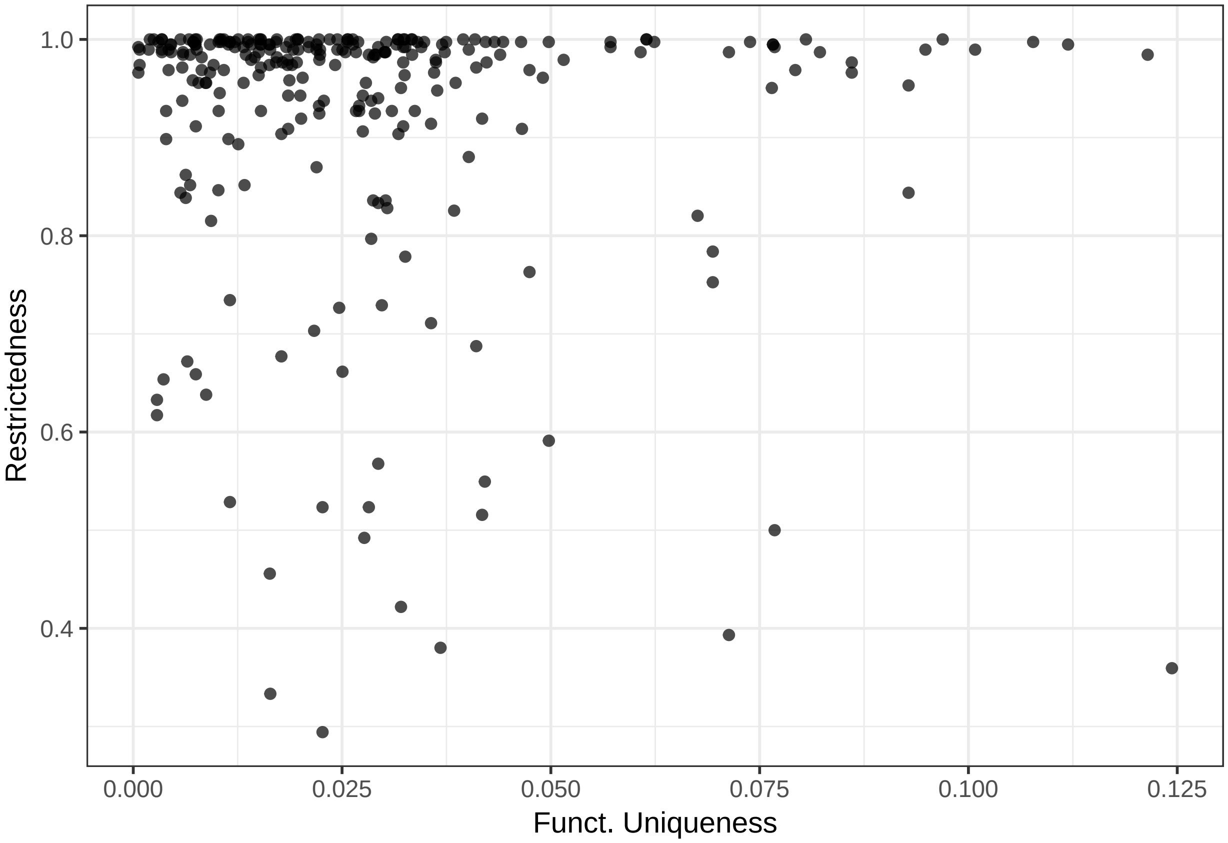
```
di_ui = ggplot(mamm_indices, aes(Di, Ui)) +
    geom_point(alpha = 0.7) +
    labs(x = "Funct. Distinctiveness",
         y = "Funct. Uniqueness") +
    theme_bw()

di_ui
```



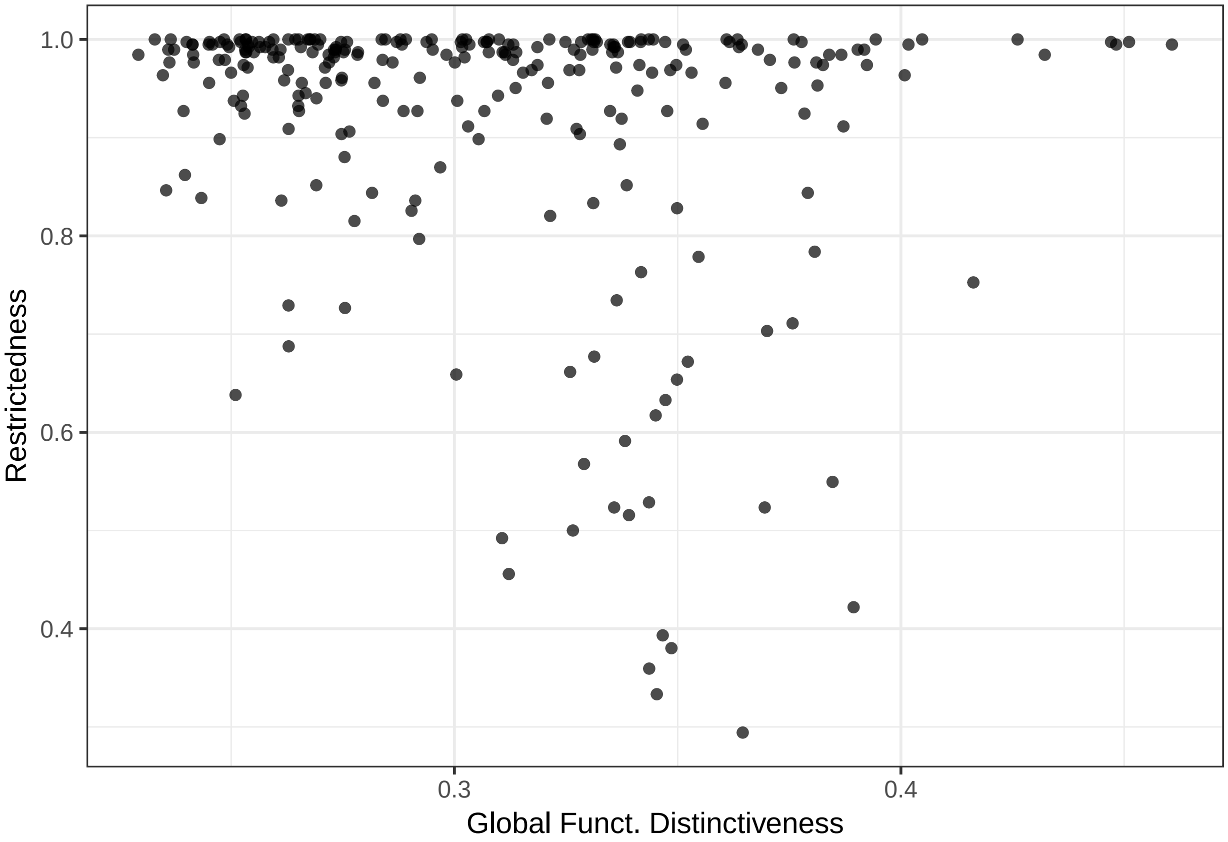
```r
ui_ri = ggplot(mamm_indices, aes(Ui, Ri)) +
    geom_point(alpha = 0.7) +
    labs(x = "Funct. Uniqueness",
         y = "Restrictedness") +
    theme_bw()

ui_ri
```



```
di_ri = ggplot(mamm_indices, aes(Di, Ri)) +
    geom_point(alpha = 0.7) +
    labs(x = "Global Funct. Distinctiveness",
         y = "Restrictedness") +
    theme_bw()

di_ri
```



Functional Space

The aim here is to use dimension reduction technique (such as Principal Component Analysis or Principal Coordinates analysis) and reduce the functional space to a few key axes. Then to map on the projected points the values of our indices to see how they are associated to certain traits.

Maps

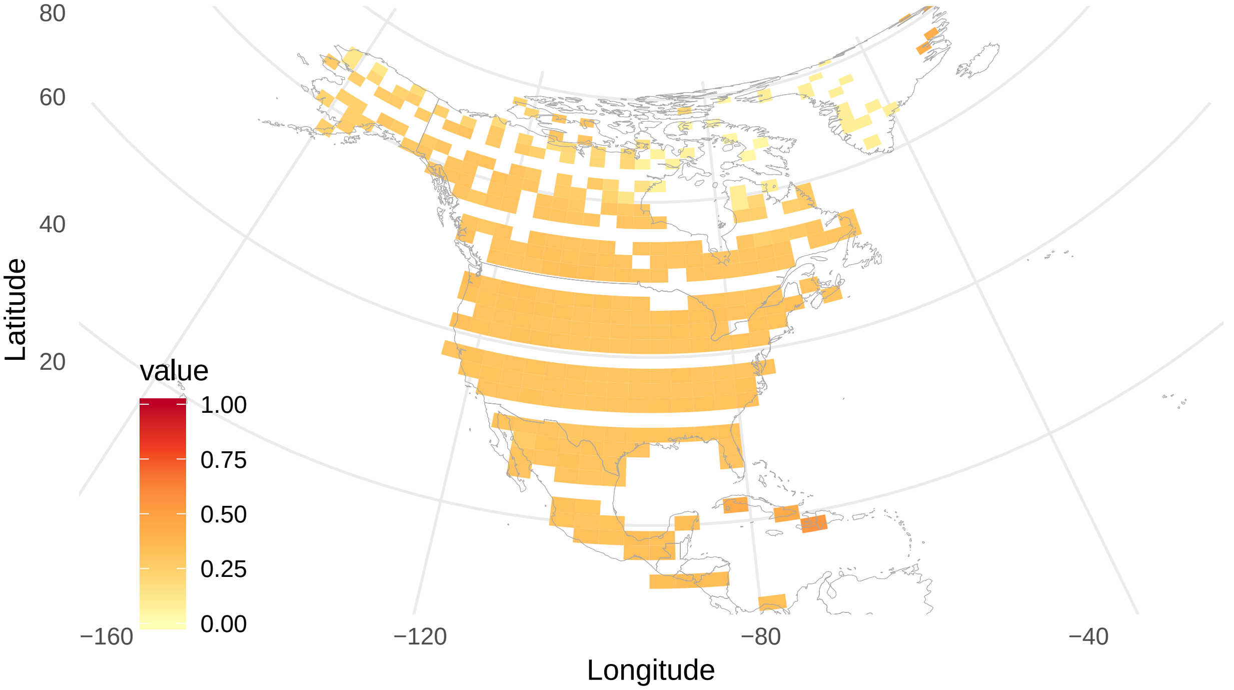
We can for example draw a map of site average functional distinctiveness:

```
# Compute average distinctiveness per site
site_di = with(df_di, aggregate(di, list(site), mean))
names(site_di) = c("site", "Di_avg")

# Add site coordinates
site_di = merge(site_di, pres_matrix[, c(1:3)], by.x = "site", by.y = "GlobalID")

# Draw map
map_di = ggplot(site_di, aes(x = Longitude, y = Latitude, z = Di_avg)) +
    stat_summary_2d(bins = 50, color = NA) +
    borders(xlim = c(-170, -23), ylim = c(9.5,83), size = 0.01,
            colour = "gray65") +
    coord_map("albers", at0 = 40, lat1 = 20,
              xlim = range(site_di$Longitude),
              ylim = range(site_di$Latitude)) +
    scale_fill_distiller(palette = "YlOrRd", direction = 1,
                         limits = c(0, 1)) +
    theme_minimal() +
    theme(legend.position = c(0.1, 0.2)) +
    borders(xlim = c(-170, -23), ylim = c(9.5,83), size = 0.1,
           colour = "gray65")

map_di
```



References

Gower, J. C. 1971. “A General Coefficient of Similarity and Some of Its Properties.” *Biometrics* 27 (4): 857–71. https://doi.org/10.2307/2528823.

Grenié, Matthias, Pierre Denelle, Caroline M. Tucker, FranÃ§ois Munoz, and Cyrille Violle. 2017. “Funrar: An R Package to Characterize Functional Rarity.” *Diversity and Distributions* 23 (12): 1365–71. https://doi.org/10.1111/ddi.12629.

Jones, Kate E., Jon Bielby, Marcel Cardillo, Susanne A. Fritz, Justin O’Dell, C. David L. Orme, Kamran Safi, et al. 2009. “PanTHERIA: A Species-Level Database of Life History, Ecology, and Geography of Extant and Recently Extinct Mammals.” *Ecology* 90 (9): 2648–8. https://doi.org/10.1890/08-1494.1.

Lawing, A. Michelle, Jussi T. Eronen, Jessica L. Blois, Catherine H. Graham, and P. David Polly. 2016. “Data from: Community Functional Trait Composition at the Continental Scale: The Effects of Non-Ecological Processes,” April. https://doi.org/10.5061/dryad.9t0n8.

———. 2017. “Community Functional Trait Composition at the Continental Scale: The Effects of Non-Ecological Processes.” *Ecography* 40 (5): 651–63. http://onlinelibrary.wiley.com/doi/10.1111/ecog.01986/full.

Pavoine, Sandrine, Jeanne Vallet, Anne-Béatrice Dufour, Sophie Gachet, and Hervé Daniel. 2009. “On the Challenge of Treating Various Types of Variables: Application for Improving the Measurement of Functional Diversity.” *Oikos* 118 (3): 391–402. https://doi.org/10.1111/j.1600-0706.2008.16668.x.

Violle, Cyrille, Wilfried Thuiller, Nicolas Mouquet, FranÃ§ois Munoz, Nathan J. B. Kraft, Marc W. Cadotte, Stuart W. Livingstone, and David Mouillot. 2017a. “Functional Rarity: The Ecology of Outliers.” *Trends in Ecology & Evolution* 32 (5): 356–67. https://doi.org/10.1016/j.tree.2017.02.002.

Violle, Cyrille, Wilfried Thuiller, Nicolas Mouquet, François Munoz, Nathan J. B. Kraft, Marc W. Cadotte, Stuart W. Livingstone, and David Mouillot. 2017b. “Functional Rarity: The Ecology of Outliers.” *Trends in Ecology & Evolution* 32 (5): 356–67. https://doi.org/10.1016/j.tree.2017.02.002.

Wickham, Hadley. 2014. “Tidy Data.” *Journal of Statistical Software* 59 (1): 1–23. https://doi.org/10.18637/jss.v059.i10.