

Group 3. fish traits

Aurele Toussaint (aurele.toussaint@cnrs.fr)

Autumn 2024

Contents

```
## step 1. load the data.
traits = read.csv("traitsFish.csv", h = T, sep = ",")

## step 2. data exploration
# Here we use common R function to explore the files
# Getting some statistics
ncol(traits) # number of columns
nrow(traits) # ... rows
dim(traits)  # dimensions
names(traits) # names of columns (colnames works as well)
row.names(traits) # if there are row names
str(traits)
head(traits) # just upper rows
summary(traits) # short summary of each variable

## Step 2: set a problematic

## Step 3. Design analyses
library(funspace)

# Run PCA
traitScale = scale(log10(traits[, !colnames(traits) %in% c("row.number", "Site1", "species", "group")]+1))
pca.trait = princomp(traitScale, cor = TRUE)

# Building the functional trait space (using the first two PCs)
trait_space_global = funspace(x = pca.trait, PCs = c(1, 2), n_divisions = 300)
plot(x = trait_space_global, type = "global", quant.plot = TRUE, arrows = TRUE, arrows.length = 0.1)

if (!require("geometry")) install.packages("geometry", repos = "http://cran.us.r-project.org")
library(geometry)
library(FD)
library(funspace)

# Step 1: Define a function to calculate FRic using the convex hull
calculate_fric <- function(traits, comm) {
  traits = traits[names(which(apply(comm, 2, sum) > 0)),]
  comm = comm[,names(which(apply(comm, 2, sum) > 0))]
  # Check if there are enough points to form a convex hull
```

```

if (nrow(traits) > ncol(traits)) {
  fric <- dbFD(traits,comm,calc.FRic = TRUE, calc.CWM = TRUE, calc.FDiv = F)$FRic
} else {
  fric <- 0 # If not enough points, FRic is 0
}
return(fric)
}

# Step 2: Create a null model function to randomize species' traits and recalculate FRic
null_model_fric <- function(traits, comm, num_simulations = 999) {
  null_fric_values <- matrix(0,nc=num_simulations,nr = nrow(comm),
    dimnames = list(rownames(comm),paste0("Rep.",1:num_simulations)))

  fd.rand = list()
  for (rand in 1:num_simulations){
    sample.comm = comm
    sample.comm[] = 0
    for(j in 1:nrow(comm)){
      sample.comm[j, names(sample(comm[j,],sum(comm[j,])))] = 1
    }
    null_fric_values[,rand] = calculate_fric(traits,sample.comm)
  }

  return(null_fric_values)
}

# Step 3: Compare observed FRic with the null model
compare_fric_with_null <- function(traits,comm, num_simulations = 999) {
  # Calculate species richness
  richness = apply(comm,1,sum)

  # Calculate observed FRic
  observed_fric <- calculate_fric(traits,comm)

  # Generate null FRic values using the null model
  null_fric_values <- null_model_fric(traits,comm, num_simulations)

  # Calculate ses and p-value: Proportion of null FRic values greater than or equal to observed FRic
  ses <- (observed_fric - apply(null_fric_values,1,mean,na.rm=T)) / apply(null_fric_values,1,sd,na.rm=T)
  p_value <- apply(rbind(observed_fric,t(null_fric_values)),2,function(x){
    length(x[-1][x[1] >= x[-1]])/length(x)
  })

  # Return the results
  return(cbind.data.frame(richness = richness,
    observed_fric = round(observed_fric,3),
    mean_exp = round(apply(null_fric_values,1,mean),3),
    ses = round(ses,3), p_value = round(p_value,3)))
}

# Step 4: Trait and community
traits <- pca.trait$scores[,c(1,2)]

```

```

# Compare observed FRic with null model
result <- compare_fric_with_null(traits, comm, num_simulations = 99)

# Print the results
trait_space_comm = funspace(x = pca.trait, PCs = c(1, 2),
                           group.vec = comm[1,,drop = F],
                           n_divisions = 300)

plot(x = trait_space_comm, type = "groups",
     quant.plot = TRUE, globalContour = T,
     pnt = T,
     pnt.cex = 0.1,
     pnt.col = rgb(0.2, 0.8, 0.1, alpha = 0.2), axis.title.line = 1)

```