Tutorial to analyze the distribution of species along environmental and geographic gradients in Amazonia

Developed by Cristian Dambros, Gabriela Zuquim, and Gabriel Moulatlet

June 2020

Contents

1 Tutorial to analyze the distribution of species along environmental and geographic gradients in Amazonia

This script exemplifies analyses using a single taxonomic group. The tutorial uses data and R functions provided in `.csv` and `.R` formats. R scripts with all the code presented here is also provided (`RCode.R` with comments and `RCode_lowdoc.R` withouth comments). For combined analyses without requiring repetition of this code multiple times when analyzing multiple datasets contact the authors

2 Import packages and functions

2.1 Load R packages

For the analyses of this manuscript, we use four R packages that are loaded using the code below. Other packages were used to extract environmental data from raster files and to generate maps of the study area. However, these steps are not shown in this tutorial.

```
library(vegan) # install.packages("vegan")
library(ecodist) # install.packages("ecodist")
library(MuMIn) # install.packages("MuMIn")
library(ape) # install.packages("ape")
```

2.2 Load R functions created specifically for this paper

In addition to these R packages, functions were created to facilitate some of the analyses (eg. multiple regression on distance matrices) for several taxa simultaneously and for plotting the results from these analyses. These functions are not available in R packages, but are provided along with this tutorial and can be loaded using the `source` function

```
# Script needs to be in the same folder as the `.RData` file
source("RFunctions.R")

# Used if dissimilarities corrected for undersampling are used
source("https://raw.githubusercontent.com/csdambros/R-functions/master/chaodist.R")
```

3 Import data

The data used is provided in two tables with biotic and abiotic information.

3.1 Species occurrence data in long format

The `occLong` data is a table with the biotic data (species occurrences) in the long format. The long format is ideal for storing data. In this format, each row represents a record and all the raw information collected in the sampling process can be maintained without losing information. The information provided in these data has not been modified or summarized in any way that could hide details or any information about the individual records obtained for the specimens (as would be the case if a presence x absence matrix was provided).

```r
# Import the occLong table
occLong<-read.csv("occLongBioGeoAmazonia.csv")

# show first rows and columns of data
occLong[1:5,1:5]
```

```
##      ID            plotID  site module subplot
## 1 75883 BR319_M01_TN_0500 BR319    M01     230
## 2 75884 BR319_M01_TN_0500 BR319    M01      80
## 3 76032 BR319_M01_TN_0500 BR319    M01     180
## 4 76079 BR319_M01_TN_0500 BR319    M01      80
## 5 76146 BR319_M01_TN_0500 BR319    M01      80
```

3.2 Environmental data

The `env` data is a table with the abiotic data (predictor variables). In this table, each row represents a sampling site. The env data has information for all sites in Amazonia included in the study, not only for the taxa used for demonstration in this tutorial. The PlotID column represents the site identification and has a matching column in the occurrence table.

```r
# Import the env table
env<-read.csv("envBioGeoAmazonia.csv")


# show first rows and columns of data
env[1:5,1:5]
```

```
##              plotID site  grid module SectionLength
## 1 BR319_M01_TN_0500 <NA> BR319    M01          <NA>
## 2 BR319_M01_TN_1500 <NA> BR319    M01          <NA>
## 3 BR319_M01_TN_2500 <NA> BR319    M01          <NA>
## 4 BR319_M01_TN_3500 <NA> BR319    M01          <NA>
## 5 BR319_M01_TN_4500 <NA> BR319    M01          <NA>
```

```r
# Assign names for the rows of this table (important for analyses later)
rownames(env)<-env$plotID
```

3.3 Data inputation

Unfortunately, not all sampling sites have environmental data. Some sites, such as the Jaú National Park are in locations of difficult access and only some data are available for this site. To use these data in subsequent analyses, we performed data imputation. Imputation fills the missing data and was performing by randomly selecting data from other plots where data are available. This procedure is conservative from the statistical point of view because it adds noise to the data reducing the significance and explanatory power of predictor variables (i.e. does not increase type I error rates. Any statistically significant result would still be significant if the inputted data were removed from analyses) [1].

[1]Note that imputation involves assigning random values to some variables. This might make the results presented in models with these variables to be slightly different every time the model runs. This process can also make the model coefficients to be slightly different here and in the published manuscript. However, the differences were always very small (usually in the third decimal place) and we have not observed differences in the significance of the association of these or other variables and species composition.

```
### Input missing clay data from other samples (random sample from other plots)
### !Adds noise to the data, potentially reducing the power of statistical tests

# Soil clay
env$clay<-ifelse(is.na(env$clay),
                 sample(env$clay[!is.na(env$clay)],replace = TRUE),
                 env$clay)

# Soil bases
env$SumofBases_cmol.log<-ifelse(is.na(env$SumofBases_cmol.log),
                                    env$KrigeSoil_fernR,
                                    env$SumofBases_cmol.log)
```

4 Data preparation

4.1 Create a short table (abundance x species matrix)

Although the `occLong` table has lots of information in detail, this table needs to be modified into a short table to run the analyses. The Jaccard similarity index and other metrics are measured by comparing sites and species. To make these comparisons programs and functions usually use a short table as input. The short table has sampling plots as rows and species as columns and is filled with abundance or presence/absence data for each species in each sampling plot. This short table can be easily created from the `occLong` table using the `tapply` function in R.

```
attach(occLong)

occAB<-tapply(n,list(plotID,species),sum)

## Replace NAs with zeroes
occAB<-ifelse(is.na(occAB),0,occAB)

detach(occLong)
```

The abundance table created above can be easily converted into a presence-absence table by replacing values above 0 by 1, and leaving values equal 0 as 0.

```
## Create Presence/Absence matrix
occPA<-ifelse(occAB>0,1,0)
```

4.2 Check correlation between environmental variables

Results from multiple regression models can be misleading if correlated variables are included because the model calculates partial p-values. i.e. the association of a variable after the removal of the effect of all other variables from the model. When variables are correlated to each other, their association with the response variable cannot be disentangled. Therefore, no effect can be detected after the removal of one of the variables (it is almost as removing the variable itself!). Therefore, it is interesting to remove correlated variables before running these models.

```
cor(env[,c("sa.latlong.treecover",
           "SumofBases_cmol.log",
           "clay",
           "CHELSA_bio_5",
           "CHELSA_bio_6",
           "CHELSA_bio_17")],
       use="pairwise.complete")
```

```
##                       sa.latlong.treecover SumofBases_cmol.log         clay
## sa.latlong.treecover           1.000000000        -0.020563865  0.003484682
## SumofBases_cmol.log           -0.020563865         1.000000000 -0.015470016
## clay                           0.003484682        -0.015470016  1.000000000
## CHELSA_bio_5                   0.132238187         0.003874775 -0.100759823
## CHELSA_bio_6                   0.084789332        -0.378614321  0.032294978
## CHELSA_bio_17                 -0.036417416        -0.274979880  0.124062721
##                       CHELSA_bio_5 CHELSA_bio_6 CHELSA_bio_17
## sa.latlong.treecover   0.132238187  0.084789332   -0.03641742
## SumofBases_cmol.log    0.003874775 -0.378614321   -0.27497988
## clay                  -0.100759823  0.032294978    0.12406272
## CHELSA_bio_5           1.000000000  0.005640386   -0.42615727
## CHELSA_bio_6           0.005640386  1.000000000    0.66584282
## CHELSA_bio_17         -0.426157272  0.665842816    1.00000000
```

In these data, only the climatic CHELSA_bio_5 (tempMax) and CHELSA_bio_17 (precDryQ) will be used because they represent temperature and precipitation and are not correlated to other variables. Other predictor variables are not strongly correlated to each other and are ok to be included as independent predictors. It is important to note that the choice of what correlated variables will be included must be based on the biological meaning of the variable, not only the correlation to other variables.

4.3 Select only plots in environmental data where the group occurs

This step is not necessary if the environmental data already has only the plots where the focal group was obtained

```
env<-env[env$plotID%in%rownames(occPA),]
```

4.4 Standardize predictor variables

Before running the analyses, we standardized all predictor and response variables. By standardizing these variables, the coefficients for the different response or predictor variables are proportional to the variance explained by the variable and are perfectly comparable - i.e. the coefficients are not affected by the magnitude or the variance in the original values.

```
# Standardize predictor variables
envStd<-decostand(
  env[c("Long",
        "Lat",
        "clay",
        "sa.latlong.treecover",
        "SumofBases_cmol.log")]
  ,"standardize")
```

```
# Add biogeographic region to the data
# (this is a categorical predictor, not possible to standardize)
envStd$class_Ribas<-env$class_Ribas
```

5 Data analysis

As described in the main text, we analyzed the data using two approaches: Distance-based and Raw-data-based (see Tuomisto et al. 2008 for details)

5.1 Distance-based approach (Multiple Regression Distance matrices)

5.1.1 Create response variables - dissimilarity matrix

In the distance-based approach, the model is run using triangular distance/dissimilarity matrices as predictors and response variables. Here we use the pairwise Jaccard dissimilarity as the response variable. The Jaccard dissimilarity measures the percentage of shared species between each pair of plots. In the distance-based approach, the models are used to answer "why some pairs of sites share fewer species than other pairs / why some pairs of sites are less similar to each other".

```
# Create matrix with the pairwise Jaccard dissimilarities
ecoDist<-vegdist(occPA,method="jaccard",na.rm = FALSE)

# Standardize dissimilarity
# (make coefficients compable among taxa in multiple-taxa comparisons)
ecoDistStd<-decostandDist(ecoDist,na.rm=TRUE)
```

In this example and the main manuscript, the classical Jaccard dissimilarity index was used. However, two alternatives that are sometimes used and the code to generate these dissimilarities are presented below: Extended dissimilarities and the bias-corrected Jaccard index proposed by Chao et al. (2005).

In case you want to use extended dissimilarities:

```
ecoDistExt<-stepacross(ecoDist)
```

In case you want to use the Chao method for Jaccard. The Chao method corrects for non detected species but might require some amount of high- quality data (not too many rare species)

This function can be obtained in https://raw.githubusercontent.com/csdambros/R-functions/master/chaodist.R

Download to your local folder and then use

source("chaodist.R")

or

source("PathToYourFolder/chaodist.R")

ex. source("C://users/.../chaodist.R")

```
ecoDistChao<-chaodist(occAB)
```

5.1.2 Create predictor variables - distance matrices

Now that we have created all response variables that will be used, we only need to organize the predictor variables. In our case, we need to create geographic and environmental distance matrices

Distances for predictor variables are calculated as the difference in the values of the variable between each pair of plots (e.g. two sites with temperatures of 20 and 21 degrees will be represented in the pairwise distance matrix by the value of 21-20 = 1).

```
### Geographic distance in degrees
# (converted to meters using the sp package in the main analyses in the paper)
geoDist<-dist(env[c("Long","Lat")])

### Environmental distance
treeCoverDist<-dist(env["sa.latlong.treecover"])
basesLogDist<-dist(env["SumofBases_cmol.log"])
clayDist<-dist(env["clay"])

tempMaxDist<-dist(env["CHELSA_bio_5"])
precDryQDist<-dist(env["CHELSA_bio_17"])


# Difference based on biogeographic region (0 if same, 1 otherwise)
regionRibasMat<-as.matrix(dist(as.integer(env$class_Ribas)))>0
rownames(regionRibasMat)<-labels(clayDist)
regionRibasDist<-as.dist(regionRibasMat)


# Create standardized matrices
geoDistStd<-decostandDist(geoDist)

treeCoverDistStd<-decostandDist(treeCoverDist)
basesLogDistStd<-decostandDist(basesLogDist)
clayDistStd<-decostandDist(clayDist)

tempMaxDistStd<-decostandDist(tempMaxDist)
precDryQDistStd<-decostandDist(precDryQDist)

regionRibasDistStd<-decostandDist(regionRibasDist)
```

5.1.2.1 The decay in species similarity with geographic and environmental distances

One of the most conspicuous patterns in ecology is the decay in species similarity with geographic distance - i.e. the tendency to areas that are closer to resemble each other more than areas that are separated by long distances (Nekola and White 1999). This can be caused by the presence of barriers to dispersal (and the geographic distance per se can impose a limit to dispersal) or differences in the environment, which also tends to be more similar in areas that are close to each other.

It is common to compare the decay in species similarity with the increase in geographic distance and environmental distance between pairs of plots. The following code is used to generate two graphs showing these associations.

```
# Plot distance-decay
par(mar=c(5,5,3,2),mfrow=c(1,2))
```

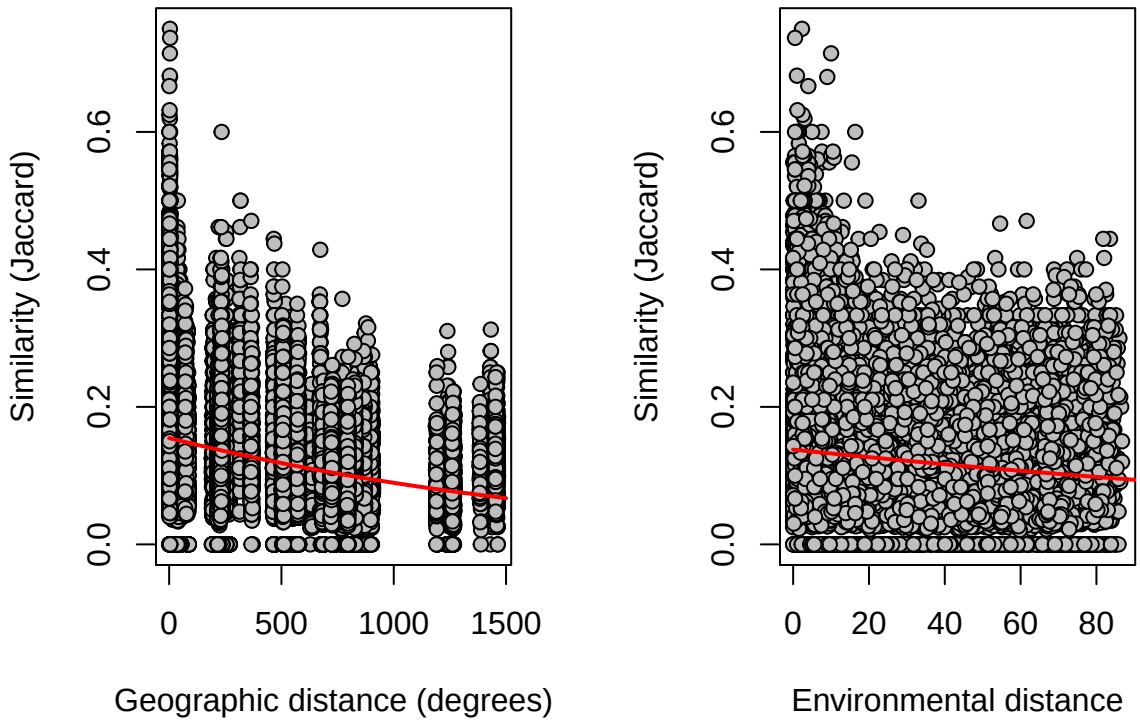
```r
plot(geoDist*111,1-ecoDist,
     xlab="Geographic distance (degrees)",ylab="Similarity (Jaccard)",
     pch=21,bg="grey")

# Fit an exponential decay line
#Model (do not use p-values from this model!!)
glm1<-glm(1-ecoDist~I(geoDist*111),family = binomial(link="log"))
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```r
# Add line to graph
plot(function(x){plogis(cbind(1,x)%*%coef(glm1))},xlim=c(0,1500),add=TRUE,lwd=2,col=2)


# It is possible to include argument start=c(-0.08,-0.009) in the glm if there is no convergence

#### The same for environmental distance

plot(clayDist,1-ecoDist,
     xlab="Environmental distance",ylab="Similarity (Jaccard)",
     pch=21,bg="grey")

# Fit an exponential decay line
#Model (do not use p-values from this model!!)
glm1<-glm(1-ecoDist~clayDist,family = binomial(link="log"))
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```r
# Add line to graph
plot(function(x){plogis(cbind(1,x)%*%coef(glm1))},xlim=c(0,100),add=TRUE,lwd=2,col=2)
```



5.1.3 Multiple regression on distance matrices (MRM)

We can run regression models associating predictor distance matrices to the response dissimilarity matrix. However, dissimilarity values in the matrix are not independent of each other as required in classical regression and ANOVA models. This happens because each plot is used multiple times for comparisons to all other plots. To not inflate Type I error rates, p-values can be calculated by randomly permuting plots (not dissimilarity values). The `MRM` function from the `ecodist` package does just this.

To facilitate the analyses, we created the `MRM4` function. This function is very similar to the `MRM` function from the `ecodist` package. However, the function automatically compares the predictor and response matrices so that they have the same size (i.e. it is possible to provide matrices of different dimensions) and standardizes the predictor and response matrices to make the coefficients comparable. This also makes these coefficients equivalent to those obtained in a Mantel test with the advantage that more than one predictor variable can be included in the same model (as in a partial Mantel test).

```
# Check variables individually
# Using the created MRM4 function
MRM4(ecoDistStd,geo=log(geoDist+0.01))

# The same uging the MRM function
MRM(ecoDistStd~log(geoDist+0.01)) # attention, not standardized here
MRM(ecoDistStd~precDryQDistStd)
MRM(ecoDistStd~tempMaxDistStd)
MRM(ecoDistStd~clayDistStd)
MRM(ecoDistStd~basesLogDistStd)
```

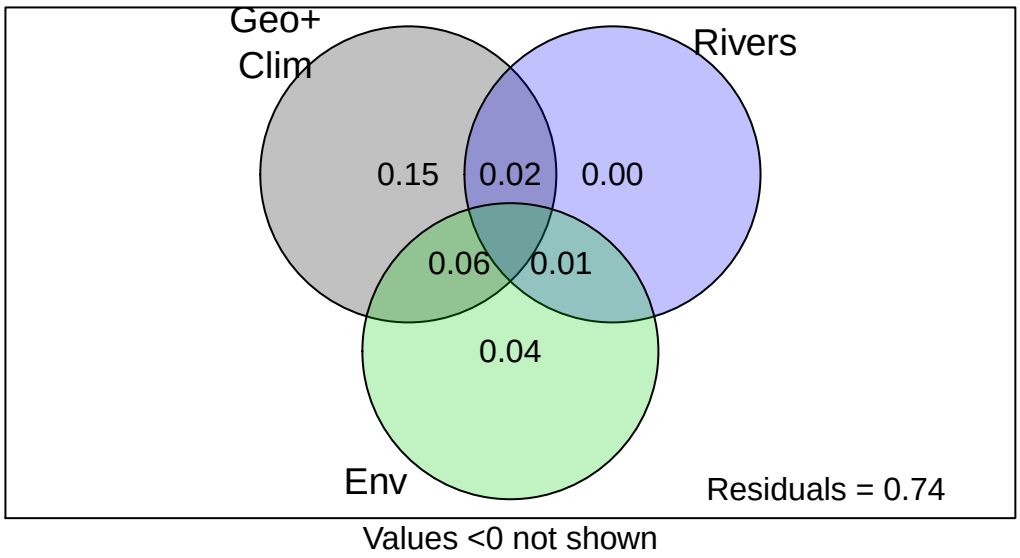
```
MRM(ecoDistStd~treeCoverDistStd)
MRM(ecoDistStd~regionRibasDistStd)
```

```
# Combining variables in a single model
MRM4(ecoDistStd,
     geo=log(geoDist+0.01),
     clay=clayDistStd,
     bases=basesLogDistStd,
     tree=treeCoverDistStd)
```

```
## $coef
##               ecoDist  pval
## Int      1.545229e-15 0.002
## p1.geo   4.194228e-01 0.001
## p2.clay  1.346689e-01 0.001
## p3.bases 1.492529e-01 0.001
## p4.tree  1.030849e-01 0.001
##
## $r.squared
##        R2      pval
## 0.2567162 0.0010000
##
## $F.test
##        F   F.pval
## 1683.561    0.001
```

It is important to note that climatic distances and biogeographic differences were not included in the multiple regression model because they are almost perfectly correlated to geographic distance. The contribution of these variables is shown in the variance partitioning below. For almost all taxonomic groups (with exception of bats), geographic distance could explain more variation in species dissimilarity than climatic differences. For almost all taxonomic groups (with exception of birds), geographic distance explained more variation in species composition than differences in biogeographic region (see below). The use of these variables instead of geographic distance in the MRM model would produce a significant effect (although weaker) because they are associated with changes in species composition that could be explained by isolation by distance.

```
# Variance partitioning
#(climate and geographic distance shown combined in a single circle)

r2part<-varpart4(ecoDist,
                 list(log(geoDist+0.01),precDryQDist,tempMaxDist),
                 list(regionRibasDist),
                 list(clayDist,treeCoverDist,basesLogDist))

#plot.varpart3(r2part,col=adjustcolor(c(1,4,3),0.3),xlim=c(4.5,5.5),ylim=c(4.5,5.5),border = TRUE)
plot(r2part,bg=adjustcolor(c(1,4,3),0.4),Xnames=c("Geo+\nClim","Rivers","Env"))
```



5.2 Raw-based approach (PCoA)

Differently from the distance-based models above, raw-data based approaches have each sampling site as a sampling unit in the analyses (not pairs of sites). For each site, a value is attributed to represent the composition of species of this site. Usually, this value is obtained from ordination techniques, such as a Principal Component Analysis (PCA) or Principal Coordinates Analysis (PCoA). In PCA and PCoA analyses, sites with similar values along the ordination axes have similar attributes (species), therefore, changes in species composition along environmental gradients can be evaluated by regressing these PCA or PCoA ordination axes against these gradients. In PCoA analyses, a pairwise dissimilarity matrix (e.g. Jaccard dissimilarity) can be used to ordinate sites by their species composition.

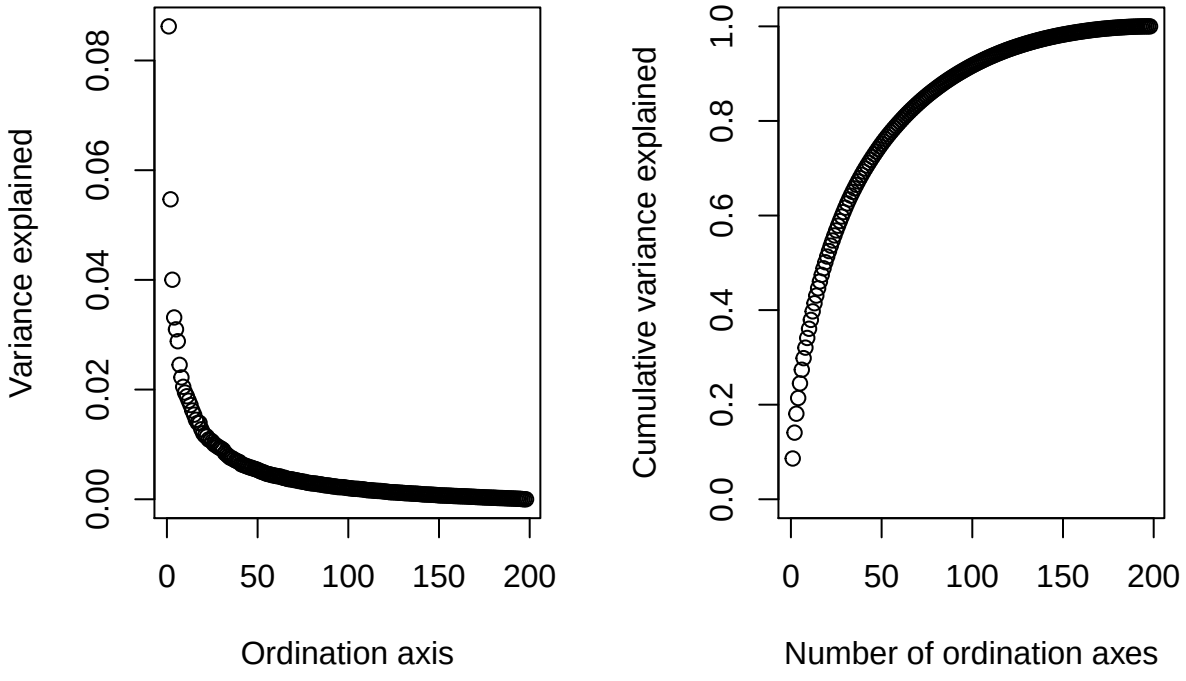
5.2.1 Create response variables - PCoA axes

```
# Run PCoA using the jaccard dissimilarity matrix calculated above
pcoa<-scores(cmdscale(ecoDist,eig=TRUE))

# The same as above but preserving the eigenvalues and adding a constant
# Eigenvalues are necessary to calculate the variance captured by the axes
# Adding a constant assures that the sum of variance of all axes is 100%
pcoaEig<-cmdscale(ecoDist,eig = TRUE,add = TRUE)


# Standardize response variables (make coefficients compable among taxa in multiple-taxa comparisons)
pcoaStd<-decostand(pcoa,"standardize")
```

5.2.1.1 Calculate percentage of variation captured by PCoA axes

The PCoA analysis generated several ordination axes, which could be used individually as response variables in regression models (or used in combination in a distance-based RDA analysis - capscale). However, the first ordination axis (largest associated eigenvalue) always have more variation captured, followed by the second axis, and so on. Therefore, the change in species composition from one site to the other is largely represented in the first ordination axes and the first one or two are often used in regression models.

```
par(mfrow=c(1,2))

# Percentage of variation explained by PCoA axes
plot(pcoaEig$eig/sum(pcoaEig$eig),xlab="Ordination axis",ylab="Variance explained")

# Cumulative
plot(cumsum(pcoaEig$eig/sum(pcoaEig$eig)),ylim=c(0,1),
     xlab="Number of ordination axes",
     ylab="Cumulative variance explained")
```



5.2.2 Modify contrast in biogeographic region (predictor variable)

In R, levels in categorical variables (factors) are by default ordered alphabetically. In linear models (eg. ANOVA using the lm function), the first level is used as a contrast. The remaining coefficients represent differences relative to the contrast. For the analyses conducted here, it is interesting to set the **Inambari**

region as the contrast because this is the region neighboring most of the other biogeographic regions. To set **Inambari** as a contrast, one can recreate the categorical variable representing biogeographic regions so that the levels are not in alphabetical order.

```
# Put Inambari as reference for contrast

# Include in origintal environmental data
env$class_Ribas<-factor(env$class_Ribas,
                        levels=c("Inambari",
                                 "Guiana",
                                 "Napo",
                                 "Negro",
                                 "Rondonia",
                                 "Tapajos",
                                 "Tapajos_South"))

# Include in standardized environmental data
envStd$class_Ribas<-factor(env$class_Ribas,
                           levels=c("Inambari",
                                 "Guiana",
                                 "Napo",
                                 "Negro",
                                 "Rondonia",
                                 "Tapajos",
                                 "Tapajos_South"))
```

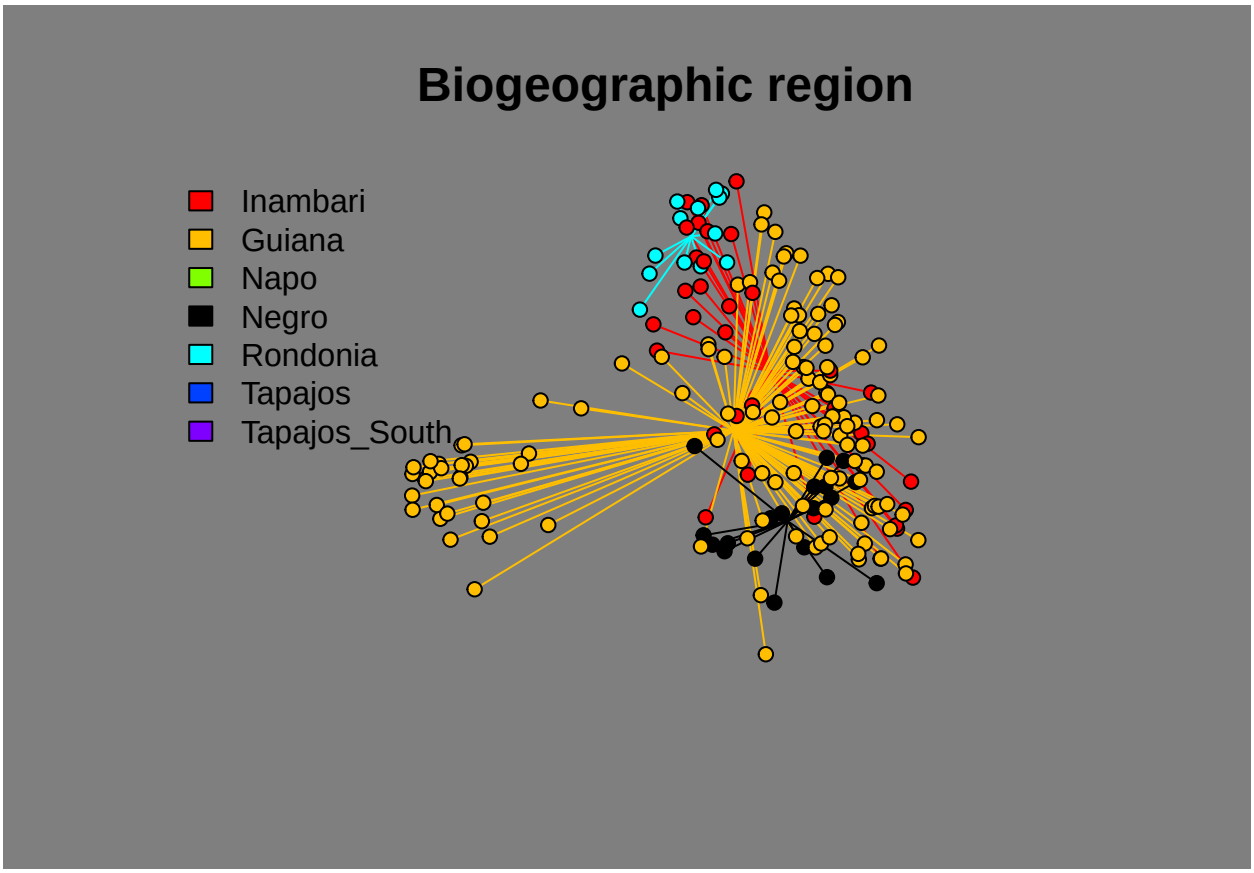
5.2.2.1 Graph showing species composition in biogeographic regions

An interesting graph to show the difference in species composition among regions using the ordination axes is a biplot. In this graph, the first and second pcoa axes are used in the x and y axes and the points are represented by sampling plots. Colors were used to represent distinct biogeographic regions.

```
colors<-rainbow(8)
colors[4]<-"black"
par(bg="grey50")
ordiplot(pcoa,type="n",axes=FALSE,ann=FALSE)
```

```
## species scores not available
```

```
ordispider(pcoa,env$class_Ribas,col=colors,pch=21,bg=1)
points(pcoa,pch=21,bg=colors[env$class_Ribas],col=1)
par(bg="white")

legend("topleft",legend = levels(env$class_Ribas),fill = colors,cex=1,bty = "n")
title("Biogeographic region",cex.main=1.5)
```



5.2.3 Multiple Regression using first PCoA axis

The first step to analyze the data using the pcoa axes is to include a model with all predictor variables of interest. Some of the variables in this complete model will be removed by comparing all the possible submodels that could be created (model selection).

```
CompleteModel<-lm(pcoaStd[,1]~
                    class_Ribas+
                    clay+SumofBases_cmol.log+
                    sa.latlong.treecover+
                    Lat+
                    Long,
                  data=envStd)


summary(CompleteModel)
```

```
##
## Call:
## lm(formula = pcoaStd[, 1] ~ class_Ribas + clay + SumofBases_cmol.log +
##     sa.latlong.treecover + Lat + Long, data = envStd)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.53534 -0.40172 -0.05459  0.30161  1.55764
```

```
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -0.11208    0.13809  -0.812 0.417999
## class_RibasGuiana     0.11766    0.18132   0.649 0.517196
## class_RibasNegro      0.64748    0.18963   3.414 0.000782 ***
## class_RibasRondonia  -0.42490    0.20558  -2.067 0.040111 *
## clay                 -0.04814    0.04871  -0.988 0.324247
## SumofBases_cmol.log  -0.27165    0.05280  -5.145 6.68e-07 ***
## sa.latlong.treecover -0.01306    0.04121  -0.317 0.751688
## Lat                  -0.75831    0.08922  -8.500 5.64e-15 ***
## Long                  0.54087    0.06189   8.739 1.26e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.542 on 189 degrees of freedom
## Multiple R-squared:  0.7181, Adjusted R-squared:  0.7062
## F-statistic: 60.19 on 8 and 189 DF,  p-value: < 2.2e-16
```

5.2.4 AIC table comparing all model subsets

The model selection used here was based on the sample corrected Akaike Information Criterion. This procedure is easy to automate using the `dredge` function from the `MuMIn` package.

```
options(na.action = "na.fail")

# Compare all submodels by AIC
AICmodels<-dredge(CompleteModel,extra = list("R^2"),fixed = c("Lat","Long"))
```

```
## Fixed terms are "Lat", "Long" and "(Intercept)"
```

```
# Show the sum of variable weights (their importance) in all models
importance(AICmodels)
```

```
##                      Lat  Long SumofBases_cmol.log class_Ribas clay
## Sum of weights:      1.00 1.00 1.00                1.00        0.35
## N containing models:   16   16    8                   8           8
##                      sa.latlong.treecover
## Sum of weights:      0.26
## N containing models:    8
```

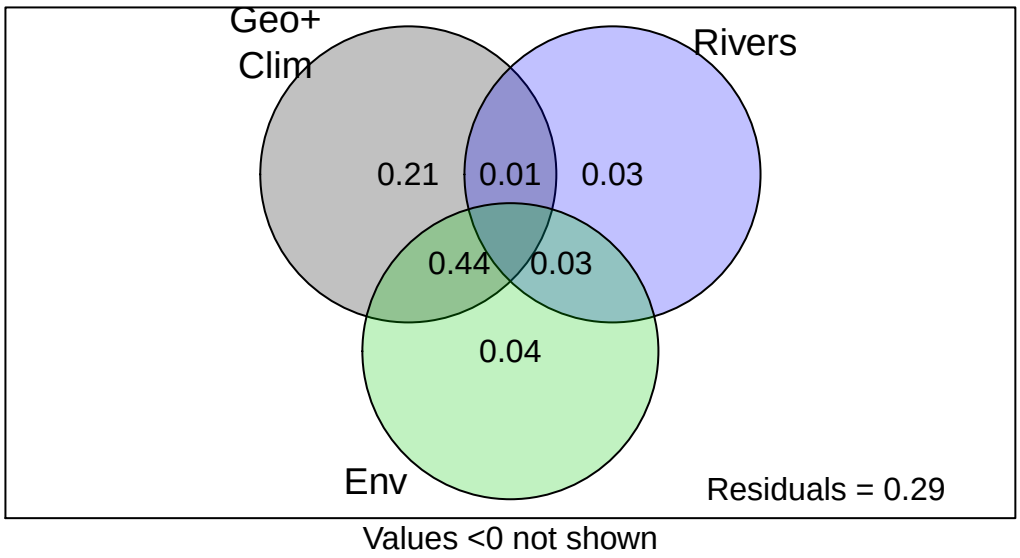
```
# Get the best model
BestModel<-get.models(AICmodels, 1)[[1]]

# show results from the best AIC model
summary(BestModel)
```

```
##
## Call:
## lm(formula = pcoaStd[, 1] ~ class_Ribas + SumofBases_cmol.log +
##     1 + Lat + Long, data = envStd)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.54947 -0.40710 -0.05911  0.29557  1.56994
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -0.06140    0.12816  -0.479  0.63242
## class_RibasGuiana    0.04470    0.16590   0.269  0.78789
## class_RibasNegro     0.60823    0.18457   3.295  0.00117 **
## class_RibasRondonia -0.40422    0.20336  -1.988  0.04828 *
## SumofBases_cmol.log -0.27979    0.05204  -5.377  2.2e-07 ***
## Lat                 -0.70677    0.07344  -9.624  < 2e-16 ***
## Long                 0.54349    0.06091   8.924  3.7e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5407 on 191 degrees of freedom
## Multiple R-squared:  0.7165, Adjusted R-squared:  0.7076
## F-statistic: 80.47 on 6 and 191 DF,  p-value: < 2.2e-16
```

In addition to showing the individual coefficients for each variable, a variance partitioning is helpful to visualize the explained variance for each group of variables. In the following graph, all environmental variables were shown combined, but the results would be similar when using only soil bases as the predictor variable (the variable in the best AIC-ranked model)

```
# Variance partitioning
r2part<-varpart(pcoaStd[,1],
                ~Lat+Long,
                ~class_Ribas,
                ~clay+
                  SumofBases_cmol.log+
                  sa.latlong.treecover,
                data = envStd)

plot(r2part,bg=adjustcolor(c(1,4,3),0.4),Xnames=c("Geo+\nClim","Rivers","Env"))
```




```
# Show parts with corresponding size
#plot.varpart3(r2part,xlim = c(4,6.2),col=adjustcolor(c(1,4,3),0.4))
```

An important step in any model is to check for spatial autocorrelation in model residuals. If autocorrelation exists in model residuals, then the sample independence assumption of the regression model is violated. This can inflate Type I error rates and leads to false claims that variables are significantly associated at a given threshold (e.g. 0.05).

```
#### Test for spatial autocorrelation in residuals
Moran.I(BestModel$residuals,as.matrix(geoDist))
```

```
## $observed
## [1] -0.02294574
##
## $expected
## [1] -0.005076142
##
## $sd
## [1] 0.004133137
##
## $p.value
## [1] 1.535764e-05
```

5.2.5 Using Moran EigenVector Maps as spatial predictor

For the taxa shown here, we can observe that model residuals have spatial autocorrelation ($p = 1.4 \times 10^{-5}$ or $p < 0.05$). This means that the analysis is violating the independence of sampling units requirement of the regression models. To correct for this problem, it is possible to include other more complex spatial variables as predictor variables in the model, so that they capture the entire spatial component of the data. Because the regression model calculates partial coefficients and p-values, the estimates for the other variables in the model will represent the results after the removal of the effect of these spatial variables (and any other variable in the model), so it will be corrected for spatial autocorrelation.

As an alternative to including Latitude and Longitude as linear predictor variables, we used Moran Eigenvector Maps as predictors. MEMs are also linear predictors, but they represent more complex forms of spatial autocorrelation. MEMs can represent the entire spatial arrangement of the data from fine to broad spatial scales. Here we define MEMs in the simplest form using the geographic distance matrix. There are many different ways to create MEMs (see Dray et al. 2012; Legendre and Gauthier 2012; Baumann 2019 for more details) that might be interesting to add biological realism to the spatial structure (e.g. directional dispersal). However, testing these more complex forms of autocorrelation has not changed substantially the results and it was not the focus of this study.

5.2.5.1 Create MEMs

The simplest way to create spatial vectors is to calculate the eigenvectors of the geographic distance matrix. This procedure does not require any additional R package or the creation of complex network matrices. This procedure produces n spatial vectors that can be used as independent predictor variables in regression models. To reduce the number of vectors, we picked only those with spatial autocorrelation.

```
# Run Eigen analysis to generate eigenvectors
E<-eigen(as.matrix(geoDist))

# Calculate spatial autocorrelation in vectors
MoranPval<-{}
for(m in 1:ncol(E$vectors)){
  MoranPval[m]<-Moran.I(E$vectors[,m],as.matrix(geoDist))$p.value
}
```

5.2.5.2 Run all raw-based data analyses using MEMs

Because not all MEMs have significant spatial autocorrelation, we can select only those that are significant to reduce the number of spatial covariates in the model (see Dray et al. 2012).

```
# Run model with significant vectors as covariates
CompleteModel<-lm(pcoaStd[,1]~
                    class_Ribas+
                    clay+
                    SumofBases_cmol.log+
                    sa.latlong.treecover+
                    E$vectors[,MoranPval<0.05],
                  data=envStd)

summary(CompleteModel)
```

```
##
## Call:
## lm(formula = pcoaStd[, 1] ~ class_Ribas + clay + SumofBases_cmol.log +
```

```
##     sa.latlong.treecover + E$vectors[, MoranPval < 0.05], data = envStd)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.34345 -0.20450 -0.02857  0.17473  1.18871
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -1.062e+01  8.261e+00  -1.286   0.2000
## class_RibasGuiana              1.194e+00  2.098e-01   5.694 4.83e-08 ***
## class_RibasNegro               1.263e+01  6.525e+00   1.936   0.0544 .
## class_RibasRondonia           -5.363e-01  2.197e-01  -2.441   0.0156 *
## clay                          -7.495e-02  3.160e-02  -2.372   0.0187 *
## SumofBases_cmol.log           -3.203e-02  3.736e-02  -0.857   0.3923
## sa.latlong.treecover          -8.593e-03  3.240e-02  -0.265   0.7912
## E$vectors[, MoranPval < 0.05]1 -1.195e+02  1.120e+02  -1.067   0.2873
## E$vectors[, MoranPval < 0.05]2 -1.017e+01  5.437e+00  -1.870   0.0631 .
## E$vectors[, MoranPval < 0.05]3 -1.662e+01  6.548e+00  -2.538   0.0120 *
## E$vectors[, MoranPval < 0.05]4 -3.301e+01  1.665e+01  -1.982   0.0490 *
## E$vectors[, MoranPval < 0.05]5  2.460e+01  1.982e+01   1.241   0.2162
## E$vectors[, MoranPval < 0.05]6 -3.142e+01  2.808e+01  -1.119   0.2646
## E$vectors[, MoranPval < 0.05]7 -2.378e+00  1.129e+01  -0.211   0.8334
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3467 on 184 degrees of freedom
## Multiple R-squared:  0.8878, Adjusted R-squared:  0.8798
## F-statistic: 111.9 on 13 and 184 DF,  p-value: < 2.2e-16
```

```
# Compare all submodels by AIC
AICmodels<-dredge(CompleteModel,extra = list("R^2"))
```

```
## Fixed term is "(Intercept)"
```

```
importance(AICmodels)
```

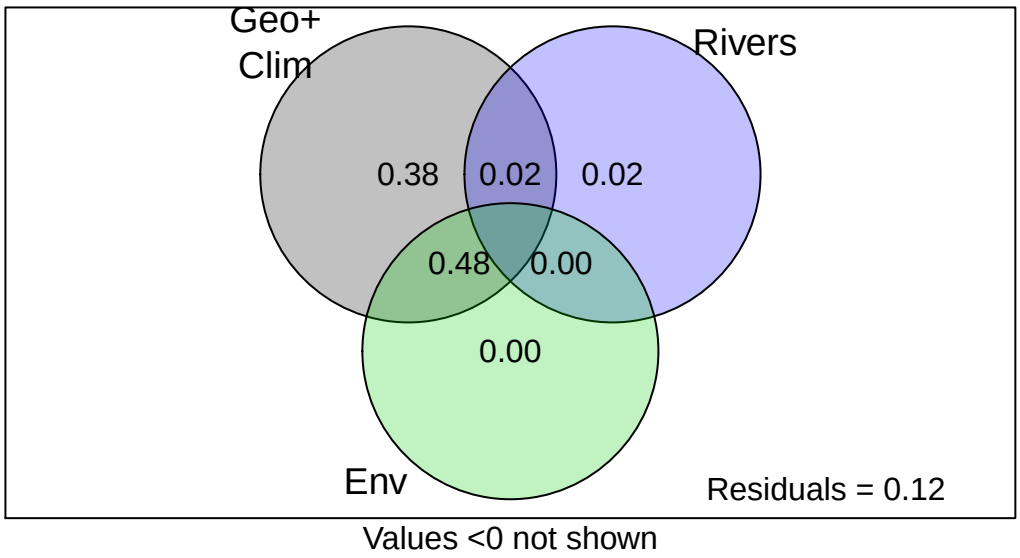
```
##                      E$vectors[, MoranPval < 0.05] class_Ribas clay
## Sum of weights:      1.00                          1.00        0.89
## N containing models:   16                            16          16
##                      SumofBases_cmol.log sa.latlong.treecover
## Sum of weights:      0.33                0.25
## N containing models:   16                  16
```

```
# Get the best model
BestModel<-get.models(AICmodels, 1)[[1]]
summary(BestModel)
```

```
##
## Call:
## lm(formula = pcoaStd[, 1] ~ class_Ribas + clay + E$vectors[,
##     MoranPval < 0.05] + 1, data = envStd)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.35102 -0.21774 -0.02728  0.17627  1.17461
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -10.92544    8.22633  -1.328  0.18577
## class_RibasGuiana                1.23734    0.20023   6.180 3.97e-09 ***
## class_RibasNegro                13.10347    5.83069   2.247  0.02579 *
## class_RibasRondonia              -0.54890    0.21023  -2.611  0.00977 **
## clay                             -0.07950    0.03106  -2.559  0.01129 *
## E$vectors[, MoranPval < 0.05]1 -122.69164  111.46722  -1.101  0.27245
## E$vectors[, MoranPval < 0.05]2  -10.62805    4.78212  -2.222  0.02746 *
## E$vectors[, MoranPval < 0.05]3  -17.12027    5.91098  -2.896  0.00423 **
## E$vectors[, MoranPval < 0.05]4  -34.19885   14.81600  -2.308  0.02209 *
## E$vectors[, MoranPval < 0.05]5   25.65092   17.76944   1.444  0.15055
## E$vectors[, MoranPval < 0.05]6  -32.13673   27.81960  -1.155  0.24950
## E$vectors[, MoranPval < 0.05]7   -2.49146   11.19274  -0.223  0.82409
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3456 on 186 degrees of freedom
## Multiple R-squared:  0.8873, Adjusted R-squared:  0.8806
## F-statistic: 133.1 on 11 and 186 DF,  p-value: < 2.2e-16
```

```
# varpart
# Variance partitioning
r2part<-varpart(pcoaStd[,1],
                ~E$vectors[,MoranPval<0.05],
                ~class_Ribas,
                ~clay+
                  SumofBases_cmol.log+
                  sa.latlong.treecover
                ,data = envStd)


plot(r2part,bg=adjustcolor(c(1,4,3),0.4),Xnames=c("Geo+\nClim","Rivers","Env"))
```




```
# Show parts with corresponding size
# plot.varpart3(r2part,xlim = c(4,6.2),
#               col=adjustcolor(c(1,4,3),0.4),
#               values = FALSE,border=c(1,1,1))


#### Test for spatial autocorrelation in residuals
Moran.I(BestModel$residuals,as.matrix(geoDist))
```

```
## $observed
## [1] -7.490392e-05
##
## $expected
## [1] -0.005076142
##
## $sd
## [1] 0.004124357
##
## $p.value
## [1] 0.2252788
```

In case you want to use a more typical MEM analysis with all the associated complexities, the `adespatial` package has several functions specifically designed for this purpose. The code and results are shown below but are not used further in this tutorial.

```r
# Load the adespatial R package
library(adespatial)

# Create and extract
E2<-as.matrix(dbmem(geoDist,MEM.autocor = "all",thresh = NULL))

MoranPval2<-{}
for(m in 1:ncol(E2)){
  MoranPval2[m]<-Moran.I(E2[,m],as.matrix(geoDist))$p.value
}
```

```r
# Run model with significant vectors as covariates
CompleteModel<-lm(pcoaStd[,1]~
                    class_Ribas+
                    clay+
                    SumofBases_cmol.log+
                    sa.latlong.treecover+
                    E2[,MoranPval2<0.05],
                  data=envStd)

summary(CompleteModel)
```

```
##
## Call:
## lm(formula = pcoaStd[, 1] ~ class_Ribas + clay + SumofBases_cmol.log +
##     sa.latlong.treecover + E2[, MoranPval2 < 0.05], data = envStd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3497 -0.2131 -0.0312  0.1961  1.1912
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -0.869173   0.190791  -4.556 9.46e-06 ***
## class_RibasGuiana              0.819751   0.189686   4.322 2.53e-05 ***
## class_RibasNegro               3.548795   1.524477   2.328  0.02100 *
## class_RibasRondonia           -0.209099   0.132728  -1.575  0.11687
## clay                          -0.083042   0.031837  -2.608  0.00984 **
## SumofBases_cmol.log           -0.030017   0.037170  -0.808  0.42039
## sa.latlong.treecover           0.001498   0.034661   0.043  0.96558
## E2[, MoranPval2 < 0.05]MEM1   -0.501458   0.122059  -4.108 5.98e-05 ***
## E2[, MoranPval2 < 0.05]MEM2    0.644619   0.073828   8.731 1.47e-15 ***
## E2[, MoranPval2 < 0.05]MEM3    0.015567   0.047492   0.328  0.74345
## E2[, MoranPval2 < 0.05]MEM5   -0.845876   0.415840  -2.034  0.04337 *
## E2[, MoranPval2 < 0.05]MEM6    0.157677   0.111830   1.410  0.16023
## E2[, MoranPval2 < 0.05]MEM197  0.500357   0.086092   5.812 2.66e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3456 on 185 degrees of freedom
## Multiple R-squared:  0.8879, Adjusted R-squared:  0.8806
## F-statistic: 122.1 on 12 and 185 DF,  p-value: < 2.2e-16
```

```r
# Compare all submodels by AIC
AICmodels<-dredge(CompleteModel,extra = list("R^2"))
```

```
## Fixed term is "(Intercept)"
```

```r
importance(AICmodels)
```

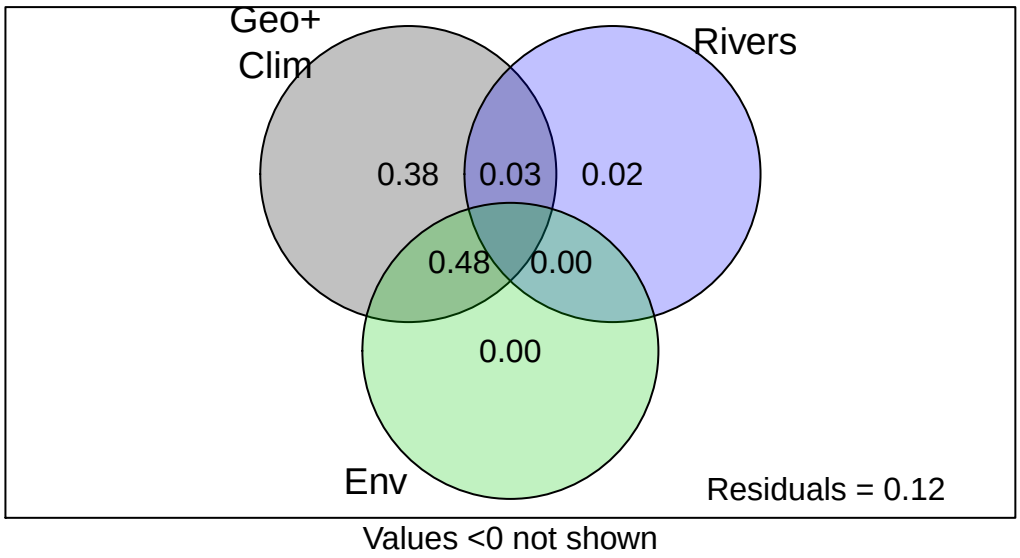
```
##                      E2[, MoranPval2 < 0.05] class_Ribas clay
## Sum of weights:      1.00                    1.00        0.94
## N containing models:   16                      16          16
##                      SumofBases_cmol.log sa.latlong.treecover
## Sum of weights:      0.32                0.24
## N containing models:   16                  16
```

```r
# Get the best model
BestModel<-get.models(AICmodels, 1)[[1]]
summary(BestModel)
```

```
##
## Call:
## lm(formula = pcoaStd[, 1] ~ class_Ribas + clay + E2[, MoranPval2 <
##     0.05] + 1, data = envStd)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.35408 -0.21066 -0.02361  0.18431  1.17907
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -0.89307    0.18779  -4.756 3.94e-06 ***
## class_RibasGuiana              0.86556    0.15650   5.531 1.06e-07 ***
## class_RibasNegro               3.48015    1.39515   2.494  0.01348 *
## class_RibasRondonia           -0.20645    0.13153  -1.570  0.11820
## clay                          -0.08711    0.03118  -2.794  0.00574 **
## E2[, MoranPval2 < 0.05]MEM1   -0.52467    0.10737  -4.887 2.20e-06 ***
## E2[, MoranPval2 < 0.05]MEM2    0.66629    0.06298  10.580  < 2e-16 ***
## E2[, MoranPval2 < 0.05]MEM3    0.01970    0.04645   0.424  0.67196
## E2[, MoranPval2 < 0.05]MEM5   -0.81945    0.37355  -2.194  0.02949 *
## E2[, MoranPval2 < 0.05]MEM6    0.15235    0.09088   1.676  0.09535 .
## E2[, MoranPval2 < 0.05]MEM197  0.50178    0.08081   6.209 3.37e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3443 on 187 degrees of freedom
## Multiple R-squared:  0.8875, Adjusted R-squared:  0.8815
## F-statistic: 147.5 on 10 and 187 DF,  p-value: < 2.2e-16
```

```r
# varpart
# Variance partitioning
r2part<-varpart(pcoaStd[,1],
                ~E2[,MoranPval2<0.05],
```

```
                ~class_Ribas,
                ~clay+
                  SumofBases_cmol.log+
                  sa.latlong.treecover
                ,data = envStd)


plot(r2part,bg=adjustcolor(c(1,4,3),0.4),Xnames=c("Geo+\nClim","Rivers","Env"))
```




```
# Show parts with corresponding size
# plot.varpart3(r2part,xlim = c(4,6.2),
#               col=adjustcolor(c(1,4,3),0.4),
#               values = FALSE,border=c(1,1,1))


#### Test for spatial autocorrelation in residuals
Moran.I(BestModel$residuals,as.matrix(geoDist))
```

```
## $observed
## [1] -6.128972e-05
##
## $expected
## [1] -0.005076142
##
## $sd
```

```
## [1] 0.004124393
##
## $p.value
## [1] 0.2240227
```

5.2.6 Tukey HSD test comparing regions

Finally, we compared all pairs of biogeographic regions separated by rivers to test if they differ in species composition. This comparison between the levels of a categorical variable can be performed using an ANOVA test and then an *a posterori* Tukey test correcting for multiple comparisons.

```
# ANOVA test
anovaResu<-aov(pcoaStd[,1]~class_Ribas,data=envStd)

# Results from the ANOVA test
summary(anovaResu)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## class_Ribas   3   5.95  1.9831   2.014  0.113
## Residuals   194 191.05  0.9848
```

```
# Tukey Honest Significance Difference test
TukeyResu<-TukeyHSD(anovaResu,ordered=FALSE)$class_Ribas

# Results from the Tukey test
TukeyResu
```

```
##                         diff        lwr       upr     p adj
## Guiana-Inambari   -0.2678618 -0.7571880 0.2214644 0.4892921
## Negro-Inambari     0.1298444 -0.6029803 0.8626692 0.9677655
## Rondonia-Inambari -0.5803438 -1.4156179 0.2549302 0.2762279
## Negro-Guiana       0.3977062 -0.2336105 1.0290229 0.3628300
## Rondonia-Guiana   -0.3124821 -1.0602865 0.4353223 0.7004085
## Rondonia-Negro    -0.7101883 -1.6358250 0.2154485 0.1959575
```

```
# Create categories to show the same (and in the same order) for all groups
# The do not change (only make graphs more comparable)
combs<-combn(levels(envStd$class_Ribas),2)
names<-paste(combs[2,],combs[1,],sep="-")
TukeyResu<-TukeyResu[match(names,rownames(TukeyResu)),]

# Plot results from Tukey test
par(mar=c(3,9,2,1))
plot(NA,xlim=c(-2.5,3),ylim=c(1,nrow(TukeyResu)),axes=FALSE,ann=FALSE)

points(TukeyResu[,1],1:nrow(TukeyResu))
arrows(TukeyResu[,2],
       1:nrow(TukeyResu),
       TukeyResu[,3],
       1:nrow(TukeyResu),
       angle = 90,code = 3,length = 0.04)
```

```
abline(v=0,lty=2)
axis(1)
axis(2,at = 1:nrow(TukeyResu),labels = names,las=2,col.axis=adjustcolor(1,0.3))
axis(2,at = 1:nrow(TukeyResu),labels = rownames(TukeyResu),las=2)
box()
```