Bioinformatic analysis & codes for:

Landscape genomics of a widely distributed snake (Dolichophis caspius, Gmelin, 1789) across eastern Europe and western Asia

Sarita Mahtani-Williams[1,2,3], William Fulton[1,2], Amelie Desvars-Larrive[1,4,5], Sara Lado[1], Jean Pierre Elbers[1], Bálint Halpern[6], Dávid Herczeg[7], Gergely Babocsay[6,8], Boris Lauš[9], Zoltán Tamás Nagy[10], Daniel Jablonski[11], Oleg Kukushkin[12,13], Pablo Orozco-terWengel[2], Judit Vörös[14,15,*], Pamela Anna Burger[1,*]

[1] Research Institute of Wildlife Ecology, Vetmeduni Vienna, Savoyenstrasse 1, A-1160 Vienna, Austria
[2] Cardiff School of Biosciences, Cardiff University, The Sir Martin Evans Building, Museum Ave, Cardiff CF103AX, UK
[3] Fundación Charles Darwin, Avenida Charles Darwin s/n, Casilla 200144, Puerto Ayora, Isla Santa Cruz, Galápagos, Ecuador
[4] Institute of Food Safety, Food Technology and Veterinary Public Health, Vetmeduni Vienna, Veterinaerplatz 1, A-1210 Vienna, Austria
[5] Complexity Science Hub Vienna, Josefstädter Straße 39, A-1080 Vienna, Austria
[6] MME Birdlife Hungary, Költő utca 21., H-1121 Budapest, Hungary
[7] Lendület Evolutionary Ecology Research Group, Plant Protection Institute, Centre for Agricultural Research, Herman Ottó út 15., H-1022 Budapest, Hungary
[8] Mátra Museum of the Hungarian Natural History Museum, Kossuth Lajos utca 40., H-3200 Gyöngyös, Hungary
[9] Association HYLA, Lipocac I.,no.7, C-10000 Zagreb, Croatia
[10] Independent researcher, Hielscherstraße 25, Berlin, G-13158, Germany
[11] Department of Zoology, Comenius University in Bratislava, Ilkovičova 6, Mlynská dolina, S-84215 Bratislava, Slovakia
[12] Department of Biodiversity Studies and Ecological Monitoring, T. I. Vyazemsky Karadag Scientific Station – Nature Reserve – Branch of Institute of Biology of the Southern Seas of the Russian Academy of Sciences, Nauki Street 24, R-298188, Theodosia, Crimea
[13] Department of Herpetology, Zoological Institute of the Russian Academy of Sciences, Universitetskaya embankment 1, R-199034, Saint Petersburg, Russia
[14] Department of Zoology, Hungarian Natural History Museum, Baross u. 13., H-1088, Budapest, Hungary
[15] Molecular Taxonomy Laboratory, Hungarian Natural History Museum, Ludovika tér 2-6., H-1083, Budapest, Hungary
* Judit Vörös and Pamela Anna Burger should be considered joint senior authors and joint corresponding authors. E-mail addresses of corresponding authors: Judit Vörös: voros.judit@nhmus.hu; Pamela Anna Burger: pamela.burger@vetmeduni.ac.at

#Summary Statistics, Population Structure, Differentiation, Admixture Plots, and NeighbourNet Network

#PLINK Version 1.9
http://zzz.bwh.harvard.edu/plink/index.shtml

Purcell S, Neale B, Todd-Brown K, Thomas L, Ferreira MAR, Bender D, Maller J, Sklar P, de Bakker PIW, Daly MJ & Sham PC (2007) PLINK: a toolset for whole-genome association and population-based linkage analysis. American Journal of Human Genetics, 81.
http://pngu.mgh.harvard.edu/purcell/plink/

##Filtering of .ped and .map files##
./plink --file CW53_24KrandomSNP --geno 0.25 --maf 0.01 --mind 0.25 --recode --out CW53_17518SNP_final --noweb

##convert to .raw format##
./plink --file CW53_17518SNP_final_popstruct2 --recodeA --out CW53_17518SNP_final_popstruct2 --noweb
#CW53_17518SNP_final.raw file duplicated and edited so population structure in Family ID column was included and shorter names for some analyses

##Conversion of .map and .ped to .gen file in PDGspider
#launch the GUI with
java -Xmx10g -Xms512m -jar PGDSpider2.jar #cd to executable folder first

 # R codes for summary statisticsand principal component analysis using adegenet package

Software Version: *adegenet* R package v. 2.1.3
https://cran.r-project.org/web/packages/adegenet/index.html

Jombart Thibout (2008) adegenet: a R package for the multivariate analysis of genetic markers *Bioinformatics*, Volume 24, Issue 11, 1 June 2008, Pages 1403–1405,
https://doi.org/10.1093/bioinformatics/btn129

#Load packages
library("adegenet")
library(ape)
library(pegas)
library(hierfstat)

#Read plink .raw format into genlight object in Rstudio

```
whipsnake_17518SNP_genlight <-
read.PLINK("/Users/Pamela/Dropbox/Caspian_whipsnake/Sambada_results/CW53_17518SNP/PLINK_and_PDGspider/CW53_17518SNP_final_popstruct.raw")

#read .gen file into genepop format
CW53_17518SNP_genepop <-
read.genepop("/Users/Pamela/Dropbox/Caspian_whipsnake/Sambada_results/CW53_17518SNP/PLINK_and_PDGspider/CW53_17518SNP_final_popstruct.gen", ncode=3L)

#Summary stats from genepop object
summary_CW53_17518_genepop <- summary(CW53_17518SNP_genepop)
str(summary_CW53_17518_genepop)
Hobs.mean <- mean(summary_CW53_17518_genepop$Hobs, na.rm = TRUE)
Hexp.mean <- mean(summary_CW53_17518_genepop$Hexp, na.rm = TRUE)
Hobs.mean
sd(summary_CW53_17518_genepop$Hobs)
Hexp.mean
sd(summary_CW53_17518_genepop$Hexp)

t.test(summary_CW53_17518_genepop$Hexp,summary_CW53_17518_genepop$Hobs,pair=T,var.equal=TRUE,alter="greater")
bartlett.test(list(summary_CW53_17518_genepop$Hexp,summary_CW53_17518_genepop$Hobs))

##Principal Component Analyses of genlight object##
pca_whipsnake_17518SNP_genlight <- glPca(whipsnake_17518SNP_genlight, nf=3)
sumeig <- sum(pca_whipsnake_17518SNP_genlight$eig)
eig_pca_whipsnake_17518SNP_genlight <-
data.frame(pca_whipsnake_17518SNP_genlight$eig)
PC1and2 <- eig_pca_whipsnake_17518SNP_genlight[1:2,]
PC1and2percentagevar <- ((PC1and2/sumeig)*100)
PC1and2percentagevar
PC1and2totalpercentagevar <- sum((PC1and2/sumeig)*100)
PC1and2totalpercentagevar


#Colorplot of PCA
myCol <- colorplot(pca_whipsnake_17518SNP_genlight$scores,
pca_whipsnake_17518SNP_genlight$scores, transp=TRUE, cex=4)
abline(h=0,v=0, col="grey")
title("Colourplot of Caspian Whipsnake individuals, axes=3")
add.scatter.eig(pca_whipsnake_17518SNP_genlight$eig[1:40],3,2,1, posi="bottomleft",
inset=.12, ratio=.275)
```

#Admixture analysis
Software Version: ADMIXTURE 1.3
https://bioinformaticshome.com/tools/descriptions/ADMIXTURE.html

Alexander DH, Lange K (2011) "Enhancements to the ADMIXTURE algorithm for individual ancestry estimation." *BMC Bioinformatics*
https://doi.org/10.1186/1471-2105-12-246

##Convert .ped and .map files into .bed .bim and .fam. For ease put input files into plink folder
./plink --file CW53_17518SNP_final_popstruct2 --make-bed --out CW53_17518SNP_final_popstruct2 --noweb
#run admixture executable for k=3-10 - .bed .bim and .fam files must be in the same folder as the admixture executable. Note cv values.
./admixture --cv CW53_17518SNP_final_popstruct2.bed 3
./admixture --cv CW53_17518SNP_final_popstruct2.bed 4
./admixture --cv CW53_17518SNP_final_popstruct2.bed 5
./admixture --cv CW53_17518SNP_final_popstruct2.bed 6
./admixture --cv CW53_17518SNP_final_popstruct2.bed 7
./admixture --cv CW53_17518SNP_final_popstruct2.bed 8
./admixture --cv CW53_17518SNP_final_popstruct2.bed 9
./admixture --cv CW53_17518SNP_final_popstruct2.bed 10

##View and plot results in R

```
setwd("your working directory")
tblK3 <- read.table("CW53_17518SNP_final_popstruct2.3.Q")
tblK4 <- read.table("CW53_17518SNP_final_popstruct2.4.Q")
tblK5 <- read.table("CW53_17518SNP_final_popstruct2.5.Q")
tblK6 <- read.table("CW53_17518SNP_final_popstruct2.6.Q")
tblK7 <- read.table("CW53_17518SNP_final_popstruct2.7.Q")
tblK8 <- read.table("CW53_17518SNP_final_popstruct2.8.Q")
tblK9 <- read.table("CW53_17518SNP_final_popstruct2.9.Q")
tblK10 <- read.table("CW53_17518SNP_final_popstruct2.10.Q")

names <- whipsnake_17518SNP_genlight$ind.names

Admixchart4 <-barplot(t(as.matrix(tblK4)), col=c("palevioletred","goldenrod",
"darkolivegreen","red"), border=NA,
 names.arg = names, cex.name=0.6, las=3)
title("Admixture analysis when K=4")
Admixchart5 <-barplot(t(as.matrix(tblK5)), col=c("red",
"palevioletred","darkolivegreen","goldenrod","lightblue"), border=NA,
 names.arg = names, cex.name=0.6, las=3)
title("Admixture analysis when K=5")
Admixchart6 <-barplot(t(as.matrix(tblK6)),
col=c("thistle","darkolivegreen","palevioletred","goldenrod","red","darkolivegreen"),
border=NA, las=3,names.arg = names, cex.name=0.6)
```

```
title("Admixture analysis when K=6")
Admixchart7 <-barplot(t(as.matrix(tblK7)),
col=c("thistle","darkolivegreen","palevioletred","red","goldenrod","lightblue","slateblue3"),
border=NA,
 names.arg = names, cex.name=0.7, las=3)
title("Admixture analysis when K=7")
Admixchart8 <-barplot(t(as.matrix(tblK8)), col=c("palevioletred","darkolivegreen",
"lightblue","coral","goldenrod","slateblue3","red", "thistle"), border=NA,
 names.arg = names, cex.name=0.7, las=3)
title("Admixture analysis when K=8")
Admixchart9 <- barplot(t(as.matrix(tblK9)),
col=c("lightgoldenrod","thistle","slateblue3","goldenrod","yellow","darkolivegreen","coral","pal
evioletred","red"), border=NA,
 names.arg = names, cex.name=0.7, las=3)
title("Admixture analysis when K=9")

par(mfrow=c(3,1))
barplot(t(as.matrix(tblK5)), col=c("red",
"palevioletred","darkolivegreen","goldenrod","lightblue"), border=NA,
 names.arg = names, cex.name=0.7, las=3)
title("Admixture analysis when K=5")
barplot(t(as.matrix(tblK7)),
col=c("thistle","darkolivegreen","palevioletred","red","goldenrod","lightblue","slateblue3"),
border=NA,
 names.arg = names, cex.name=0.7, las=3)
title("Admixture analysis when K=7")
barplot(t(as.matrix(tblK8)), col=c("palevioletred","darkolivegreen",
"lightblue","coral","goldenrod","slateblue3","red", "thistle"), border=NA,
 names.arg = names, cex.name=0.7, las=3)
title("Admixture analysis when K=5")
```

#Neighbour-network tree using SplitsTree4

Software Version: SplitsTree4 V4.10
www.splitstree.org

D.H. Huson and D. Bryant. Application of Phylogenetic Networks in Evolutionary Studies, Molecular Biology and Evolution 23(2):254-267, 2006.

#1) Create a 1-IBS distance matrix in PLINK
cd Plink folder
./plink --file filename --cluster --distance-matrix --out outfilename --noweb

#2) Use the PED file to convert it into a NEXUS file in PGDspider
#launch the GUI with java -Xmx10g -Xms512m -jar PGDSpider2.jar

#3) Copy-paste the distance matrix and the correct headers into the nexus file; the headers are found in the example folder of splitstree for an input file using a distance matrix
#BEGIN DISTANCES;
#DIMENSIONS ntax=96;
#FORMAT labels=left diagonal triangle=both;
#MATRIX
#and delete the 'SET' section. as it only gives errors but is not necessary.

paste IBS distance matrix into .nex file, replacing SNPs

###EXAMPLE INPUT FILE:
#nexus file

```
  BEGIN Taxa;
DIMENSIONS ntax=53;
TAXLABELS
[1]  'HU_PV_Gy697'
[2]  'HU_PV_Gy698'
[3]  'HU_PV_Gy838'
[4]  'HU_PV_Gy925' #...etc
;
END; [Taxa]

BEGIN Distances;
DIMENSIONS ntax=53;
FORMAT labels=left diagonal triangle=both;
MATRIX #insert distance matrix from PLINK
[1]  'HU_PV_Gy697' 0 0.0984144 0.101542 ...
[2]  'HU_PV_Gy698' 0.0984144 0 0.0852741 ...
[3]  'HU_PV_Gy838' 0.101542 0.0852741 ...
[4]  'HU_PV_Gy925' 0.0922813 0.0902578 ...
```

;
END; [Distances]

BEGIN st_Assumptions;
disttransform=NeighborNet;
splitstransform=EqualAngle;
SplitsPostProcess filter=dimension value=4;
autolayoutnodelabels;
END; [st_Assumptions]

#Estimated Effective Migration Surfaces (EEMS analysis)

Software Version: EEMS (https://github.com/dipetkov/eems)

Petkova D, Novembre J and Stephens M (2016) Visualizing spatial population structure with estimated effective migration surfaces. *Nat Genet* 48, 94–100. https://doi.org/10.1038/ng.3464

```
# R codes for plotting EEMS results:
run <- system.file("./run_10M/", package = "rEEMSplots")
eems_results <- file.path(run, " outputs")
name_figures <- file.path(path.expand("~"), " outputs")
eems.plots(mcmcpath = "./run_10M/",
           plotpath = "./test_outputs-default/testoutdeafaults",
           longlat = TRUE,
           add.grid = TRUE,
           col.grid = "gray90",
           lwd.grid = 2,
           add.outline = TRUE,
           col.outline = "black",
           lwd.outline = 2,
           add.demes = TRUE,
           col.demes = "black",
           pch.demes = 5,
          min.cex.demes = 0.5,
           max.cex.demes = 1.5,
           plot.height = 8,
           plot.width = 7,
           res = 600,
           out.pdf = FALSE)
```

#LANDSCAPE GENOMIC ANALYSIS

#Environmental variable selection

Extract WorldClim data (raster files) for each sampling point using GPS coordinates using R.

```
install.packages("rgdal")
library(rgdal)
install.packages("sp")
library(sp)
install.packages("raster")
library(raster)

# Import worldclim data as raster files .tif
# Import sample file (including data coordinates): samples

# Transform the table in spatial object (sp) and give latitude and longitude
samples.sp<-samples
coordinates(samples.sp)<- ~ longitude+ latitude

# Function for extracting the worldclim data (imported as a raster file .tif) for each sample points:
function.extract <- function (worlclim.data, points.data)
{
 convert.raster <- raster(worlclim.data)
 extract.data <- extract(convert.raster, points.data)
 extract.data
}


# Compile final table ("final.dataset") including sample ID, sample coordinates, and selected variables.


######  Selection of explanatory variables taking into account collinearity
# (see PhD Thesis of Sylvie STUCKI, page 54)
```

Stucki, S. (2014) Développement d'outils de géo-calcul haute performance pour l'identification de régions du génome potentiellement soumises à la sélection naturelle: analyse spatiale de la diversité de panels de polymorphismes nucléotidiques à haute densité (800k) chez Bos taurus et B. indicus en Ouganda, EPFL PhD Thesis no 6014, doi:10.5075/epfl-thesis-6014

```
# The phenomenon of multicollinearity is estimated by calculating the variance inflation factor (variance inflation factor, VIF).

install.packages("usdm")
library(usdm)
```

```
install.packages("fmsb")
library (fmsb)

# The following function uses three arguments. The first is a matrix or data frame of the
# explanatory variables, the second is the threshold value to use for retaining variables, and the
# third is a logical argument indicating if text output is returned as the stepwise selection
# progresses.
# The output indicates the VIF values for each variable after each stepwise comparison.
# The function "vif_func" calculates the VIF values for all explanatory variables, removes the
variable with the highest value, and repeats until all VIF values are below the threshold. The
# final output is a list of variable names with VIF values that fall below the threshold.
# The remaining variables will be included in the final.

vif_func<-function(in_frame,thresh,trace=T,...){
 library(fmsb)
 if(any(!'data.frame' %in% class(in_frame))) in_frame<-data.frame(in_frame)

 #get initial vif value for all comparisons of variables
 vif_init<-NULL
 var_names <- names(in_frame)
 for(val in var_names){
        regressors <- var_names[-which(var_names == val)]
        form <- paste(regressors, collapse = '+')
        form_in <- formula(paste(val, '~', form))
        vif_init<-rbind(vif_init, c(val, VIF(lm(form_in, data = in_frame, ...))))
 }
 vif_max<-max(as.numeric(vif_init[,2]), na.rm = TRUE)

 if(vif_max < thresh){
        if(trace==T){ #print output of each iteration
        prmatrix(vif_init,collab=c('var','vif'),rowlab=rep(',nrow(vif_init)),quote=F)
        cat('\n')
        cat(paste('All variables have VIF < ', thresh,', max VIF ',round(vif_max,2), sep=''),'\n\n')
        }
        return(var_names)
 }
 else{
        in_dat<-in_frame
        #backwards selection of explanatory variables, stops when all VIF values are below
'thresh'
        while(vif_max >= thresh){

        vif_vals<-NULL
        var_names <- names(in_dat)

        for(val in var_names){
        regressors <- var_names[-which(var_names == val)]
```

```
        form <- paste(regressors, collapse = '+')
        form_in <- formula(paste(val, '~', form))
        vif_add<-VIF(lm(form_in, data = in_dat, ...))
        vif_vals<-rbind(vif_vals,c(val,vif_add))
        }
        max_row<-which(vif_vals[,2] == max(as.numeric(vif_vals[,2]), na.rm = TRUE))[1]

        vif_max<-as.numeric(vif_vals[max_row,2])

        if(vif_max<thresh) break

        if(trace==T){ #print output of each iteration
        prmatrix(vif_vals,collab=c('var','vif'),rowlab=rep('',nrow(vif_vals)),quote=F)
        cat('\n')
        cat('removed: ',vif_vals[max_row,1],vif_max,'\n\n')
        flush.console()
        }

        in_dat<-in_dat[,!names(in_dat) %in% vif_vals[max_row,1]]

        }

        return(names(in_dat))

  }

}
```

We used a VIF threshold = 5
vif_func(in_frame=final.dataset[,-c(1:3)],thresh=5,trace=T) # remove columns sample ID, longitude and latitude

#Samβada analysis

Software version: Samβada v.0.5.3
https://www.epfl.ch/labs/lasig/page-101934-en-html/sambada/

Stucki, S., Orozco-terWengel, P., Forester, B. R., Duruz, S., Colli, L., Masembe, C., … Joost, S. (2017). High performance computation of landscape genomic models including local indicators of spatial association. Molecular Ecology Resources, 17(5), 1072–1089. doi:10.1111/1755-0998.12629

#following Samβada manual:
#Preparing the input files

#==================
#Use PLINK to convert .map and .ped files into format sambada can use -> recode-plink
#marker information is split into genotypes: CC CG GG <- one marker. 0 or 1 represents the genotype of the individual.
  # RecodePLINK nbSamples nbSNPs inputFile outputFile
./recode-plink 53 17518 CW53_17518SNP_final CW53_17518_moleculardata

##obtain population structure information through PCA eigenvalues##
#SEE R codes for *Summary statistics, population structure, differentiation, and admixture*

sumeig <- sum(pca_whipsnake_17518SNP_genlight$eig)
> eig_pca_whipsnake_17518SNP_genlight <-
data.frame(pca_whipsnake_17518SNP_genlight$eig)
> PC12and3 <- eig_pca_whipsnake_17518SNP_genlight[1:3,]
>
> PC12and3percentagevar <- ((PC12and3/sumeig)*100)
> PC12and3percentagevar
[1] 22.331141 11.878193 6.843803
> PC12and3totalpercentagevar <- sum((PC12and3/sumeig)*100)
> PC12and3totalpercentagevar
[1] 41.05314

#INPUT FILE NOTES#
population information: can be either PCA or coefficients of membership. Multiple PCAs can be used at once.
Use genotype data converted using plink (shown above)

  #ENVIRONMENTAL INPUT FILE: population info, one Environmental variable, genotype information

PC1 PC2 PC3 bio01 3_59_CC 3_59_CT 3_59_TT 5_38_CC 5_38_CT 5_38_TT
-49.99219023 -10.46583927 32.42010141 15.29583335 1 0 0 0 0 1
-47.80979234 -9.97716999 31.11890188 16.63750001 1 0 0 0 0 1
  4.435244673 0.643552238 0.404400874 10.03333335 1 0 0 1 0 0
4.936364882 0.685585106 1.044132748 7.516666732 1 0 0 1 0 0
3.744552527 0.749777141 0.421534269 7.516666732 1 0 0 1 0 0
13.12902201 -3.542524285 -0.975583404 10.77083337 1 0 0 1 0 0

  #NULL MODEL INPUT FILE: population info, genetic marker information

PC1 PC2 PC3 3_59_CC 3_59_CT 3_59_TT 5_38_CC 5_38_CT 5_38_TT
-49.99219023 -10.46583927 32.42010141 1 0 0 0 0 1
-47.80979234 -9.97716999 31.11890188 1 0 0 0 0 1
4.435244673 0.643552238 0.404400874 1 0 0 1 0 0
4.936364882 0.685585106 1.044132748 1 0 0 1 0 0
3.744552527 0.749777141 0.421534269 1 0 0 1 0 0

#PARAMETER FILE NOTES#
#NUMMARK + NUMVARENV = total no. of columns in input file
#(ID name column removed for ease of use: ID column counts as an environmental variable) HOWEVER make sure the environmental variables and pc value rows are in the CORRECT ORDER

#PARAMETER FILE: Eg.for file testing environmental variables

HEADERS YES
WORDDELIM " " # column separator
NUMVARENV 4 # no.population info columns + environmental variable column # The population variables count as environmental variables!!!
NUMMARK 52554 # no. of genotypes = markers*3 = 17518*3 = 52554 #also nummark = number of columns - (popinfo and envar columns)
NUMINDIV 53 # no. of rows excluding header row
DIMMAX 4 # same value as NUMVARENV - calculates multivariate analysis for four environmental variables.
SAVETYPE END ALL 0.0008736049 #change this pValue to the (Benjamini and Hochberg, 1995) aka FDR corrected Pvalue (or use the Bonferroni corrected Pvalue) based on the no. SNPs - see script below
LOG TERMINAL

#PARAMETER FILE: Eg. NULL model parameter file

HEADERS YES
WORDDELIM " "
NUMVARENV 3 # just population (PC1,2,3) info columns
NUMMARK 52554
NUMINDIV 53
DIMMAX 3
SAVETYPE END ALL 0.0008736049

RUNNING SAMβADA
#=====================

./sambada PARAMETER_FILE INPUT_FILE
./sambada parameter_17518SNP_PCA123_NULL.txt CW53_17518_NULL.txt
should only take around 3-5 minutes.

OUTFILES
Use the log likelihood scores from sambada out-4 files for the population + environmental data runs in the following analysis, as that is the multivariate analysis outfile for all variables (3*population info plus one ENVAR = 4)
Use sambada out-3 file for the NULL model, as that is the outfile with multivariate analysis on all the population data only.

Example outfile.
Marker Env_1 Env_2 Env_3 Env_4 Loglikelihood Gscore WaldScore NumError Efron McF...
85411_58_GG PC1 PC2 PC3 bio01 -2750.706594130204106774 -5491.569274265341590535 ...
41675_9_GG PC1 PC2 PC3 bio01 -140.1639459431191031685 -263.8741599162882298935 ...
39684_61_AA PC1 PC2 PC3 bio01 -325.8054909283937292941 -641.7670678617208357419 ...
34436_20_AA PC1 PC2 PC3 bio01 -149.3299847129400597517 -275.8407256671650781277 ...
55454_48_TT PC1 PC2 PC3 bio01 -593.8507827223361593694 -1159.497870648320527964 ...

Compute GScore for model calculated by Sambada for each variable using the Log-likelihoods
Cf. Sambadoc-v0.5.3, page 18
We compute the G score for each model using the log-likelihoods stored in the results files for dimensions P and P + 1 (G = 2 * (lp+1-lp))

import files
null.model <- read.table("whipsnake_53_19728SNP_sambadaIn_PC123-Out-3.txt", sep = "", fill=TRUE, header = TRUE, stringsAsFactors=FALSE) # the null model includes the 3 first components from the PCA
pop.wind.model <- read.table("whipsnake_53_19728SNP_sambadaIn_PC123_wind-Out-4.txt", sep = "", fill=TRUE, header = TRUE, stringsAsFactors=FALSE)
pop.bio17.model <- read.table("whipsnake_53_19728SNP_sambadaIn_PC123_bio17-Out-4.txt ", sep = "", fill=TRUE, header = TRUE, stringsAsFactors=FALSE)
pop.bio12.model <- read.table("whipsnake_53_19728SNP_sambadaIn_PC123_bio12-Out-4.txt ", sep = "", fill=TRUE, header = TRUE, stringsAsFactors=FALSE)
pop.bio8.model <- read.table("whipsnake_53_19728SNP_sambadaIn_PC123_bio8-Out-4.txt ", sep = "", fill=TRUE, header = TRUE, stringsAsFactors=FALSE)
pop.bio7.model <- read.table("whipsnake_53_19728SNP_sambadaIn_PC123_bio7-Out-4.txt ", sep = "", fill=TRUE, header = TRUE, stringsAsFactors=FALSE)
pop.bio3.model <- read.table("whipsnake_53_19728SNP_sambadaIn_PC123_bio3-Out-4.txt ", sep = "", fill=TRUE, header = TRUE, stringsAsFactors=FALSE)
pop.bio1.model <- read.table("whipsnake_53_19728SNP_sambadaIn_PC123_bio1-Out-4.txt ", sep = "", fill=TRUE, header = TRUE, stringsAsFactors=FALSE)

G.scores.comparison.models
 <- function (null.model, pop.envir.model) {
  # extract rows with same marker name (the function add data from table y to the table x)
  extract.common.markers <- inner_join(null.model, pop.envir.model , by="Marker")
  #Compute the GScore for each model using the log-likelihoods stored in the results files for dimensions P and P + 1 (G = 2 * (lP+1-lP))
  G.scores <- 2*(extract.common.markers$Loglikelihood.y - extract.common.markers$Loglikelihood.x)
  result <- cbind.data.frame (extract.common.markers$Marker, G.scores)
  result
}

```
G.scores.null.versus.pop.wind <- G.scores.comparison.models (null.model= null.model,
pop.envir.model = pop.wind.model )
G.scores.null.versus.pop.bio17 <- G.scores.comparison.models (null.model= null.model,
pop.envir.model = pop.bio17.model )
G.scores.null.versus.pop.bio12 <- G.scores.comparison.models (null.model= null.model,
pop.envir.model = pop.bio12.model )
G.scores.null.versus.pop.bio8<- G.scores.comparison.models (null.model= null.model,
pop.envir.model = pop.bio8.model )
G.scores.null.versus.pop.bio7 <- G.scores.comparison.models (null.model= null.model,
pop.envir.model = pop.bio7.model)
G.scores.null.versus.pop.bio3<- G.scores.comparison.models (null.model= null.model,
pop.envir.model = pop.bio3.model )
G.scores.null.versus.pop.bio1 <- G.scores.comparison.models (null.model= null.model,
pop.envir.model = pop.bio1.model)

##### Calculating adjusted p-values from GScores (for model calculated by Sambada for each
variable)
# The pvalues have to be corrected for multiple testing
# We use the Benjamini & Hochberg (1995) ("BH" or its alias "fdr") as recommended page 18 of
the manual.
# we keep only significant pvalues (i.e. < 0.01)

corrected.signif.pvalue.from.Gscores <- function(Gscores)
{
  non.corrected.pvalues <- pchisq(Gscores [,2], df=1, lower.tail=FALSE)
  corrected.pvalues <- p.adjust(non.corrected.pvalues, method = "BH")
  corrected.pvalues.table <- cbind.data.frame (Gscores[,1], corrected.pvalues )
  extract.signif <- corrected.pvalues.table [corrected.pvalues.table [,2] < 0.01, ]
  extract.signif
}

corrected.signif.pvalues.pop.wind <-
corrected.signif.pvalue.from.Gscores(Gscores=G.scores.null.versus.pop.wind)
corrected.signif.pvalues.pop.bio17 <-
corrected.signif.pvalue.from.Gscores(Gscores=G.scores.null.versus.pop.bio17)
corrected.signif.pvalues.pop.bio12 <-
corrected.signif.pvalue.from.Gscores(Gscores=G.scores.null.versus.pop.bio12)
corrected.signif.pvalues.pop.bio8 <-
corrected.signif.pvalue.from.Gscores(Gscores=G.scores.null.versus.pop.bio8)
corrected.signif.pvalues.pop.bio7 <-
corrected.signif.pvalue.from.Gscores(Gscores=G.scores.null.versus.pop.bio7)
corrected.signif.pvalues.pop.bio3 <-
corrected.signif.pvalue.from.Gscores(Gscores=G.scores.null.versus.pop.bio3)
corrected.signif.pvalues.pop.bio1 <-
corrected.signif.pvalue.from.Gscores(Gscores=G.scores.null.versus.pop.bio1)
```

```
# number of significant models:
nb.signif.model.pop.wind <- nrow (corrected.signif.pvalues.pop.wind)
nb.signif.model.pop.wind

nb.signif.model.pop.bio17 <- nrow (corrected.signif.pvalues.pop.bio17)
nb.signif.model.pop.bio17)

nb.signif.model.pop.bio12 <- nrow (corrected.signif.pvalues.pop.bio12)
nb.signif.model.pop.bio12

nb.signif.model.pop.bio8 <- nrow (corrected.signif.pvalues.pop.bio8)
nb.signif.model.pop.bio8

nb.signif.model.pop.bio7 <- nrow (corrected.signif.pvalues.pop.bio7)
nb.signif.model.pop.bio7

nb.signif.model.pop.bio3 <- nrow (corrected.signif.pvalues.pop.bio3)
nb.signif.model.pop.bio3

nb.signif.model.pop.bio1 <- nrow (corrected.signif.pvalues.pop.bio1)
nb.signif.model.pop.bio1
```

 # Supplemental methods for running IGA Stacks 1.45 pipeline

Software Version Stacks v 1.45
https://catchenlab.life.illinois.edu/stacks/

J. Catchen, P. Hohenlohe, S. Bassham, A. Amores, and W. Cresko. *Stacks: an analysis tool set for population genomics*. Molecular Ecology. 2013.
J. Catchen, A. Amores, P. Hohenlohe, W. Cresko, and J. Postlethwait. *Stacks: building and genotyping loci de novo from short-read sequences*. G3: Genes, Genomes, Genetics, 1:171-182, 2011.

```
# Final result is a list of genes for each climatic variable that is associated with SNPs determined
by Sambada
#
#
#
## Note that reads in this directory (Sample.fq.gz) are concatenated (not interleaved) in the form
### Forward-sequence1
### Forward-sequence2
### Forward-sequence3
### Forward-sequence4
### Forward-sequence5
### Reverse-sequence1
```

```
### Reverse-sequence2
### Reverse-sequence3
### Reverse-sequence4
### Reverse-sequence5

## Step 1 Run ustacks
cd /genetics/Burger/Whipsnake/denovo-concatentated-paired-reads
x=1
while read i; do
  /opt/stacks-1.45/ustacks \
  -i $x \
  -t gzfastq \
  -f $i.fq.gz \
  -p 75 \
  -H -d -r \
  --alpha 0.05 \
  --max_locus_stacks 2 \
  -m 3 -M 2 --model_type bounded --bound_high 0.1 \
  -o ./ > ${i}.ustacks.log 2>&1
  ((x++))
done < samples

## Step 2 Run cstacks
cd /genetics/Burger/Whipsnake/denovo-concatentated-paired-reads
/opt/stacks-1.45/cstacks \
-n 2 \
-P ./ \
-M popmap \
-p 75 > cstacks.log 2>&1

## Step 3 Run sstacks
cd /genetics/Burger/Whipsnake/denovo-concatentated-paired-reads
/opt/stacks-1.45/sstacks \
-P ./ \
-M popmap \
-p 75 > sstacks.log 2>&1

## Step 4 Run rxstacks
cd /genetics/Burger/Whipsnake/denovo-concatentated-paired-reads
/opt/stacks-1.45/rxstacks \
-b 1 \
-P ./ \
-o ./corr \
-t 75 \
--lnl_lim -50 \
--alpha 0.05 \
--model_type bounded --bound_high 0.1 \
```

```
--lnl_filter \
--conf_filter --conf_lim 0.75 > rxstacks.log 2>&1

## Step 5 Re-run cstacks on rxstacks corrected data
cd /genetics/Burger/Whipsnake/denovo-concatentated-paired-reads/corr
/opt/stacks-1.45/cstacks \
-n 2 \
-P ./ \
-M ../popmap \
-p 75 > cstacks-corr.log 2>&1

## Step 6 Re-run sstacks on cstacks catalog
cd /genetics/Burger/Whipsnake/denovo-concatentated-paired-reads/corr
/opt/stacks-1.45/sstacks \
-P ./ \
-M ../popmap \
-p 75 > sstacks-corr.log 2>&1

## Step 10 Run populations with random snp per rad locus
cd /genetics/Burger/Whipsnake/denovo-concatentated-paired-reads/corr
/opt/stacks-1.45/populations \
-t 75 \
-P ./ \
-p 1 \
-r 0.75 \
-m 3 \
-M ../popmap \
--fasta \
--vcf --write_random_snp > populations-corr-write_random_snp.log 2>&1

## Step 11 Rename output files and convert from VCF to PLINK
cp batch_1.vcf whipsnake-ddrad-raw-variants-for-sambada-write_random_snp.vcf
cp batch_1.fa whipsnake-ddrad-RAD-loci-for-sambada-write_random_snp.fa

vcftools --vcf whipsnake-ddrad-raw-variants-for-sambada-write_random_snp.vcf --plink --out
whipsnake-ddrad-raw-variants-for-sambada-write_random_snp
cat whipsnake-ddrad-RAD-loci-for-sambada-write_random_snp.fa |paste - - |grep "Allele_0"|tr '\t'
'\n' |perl -pe "s/>CLocus_(\d+)_.+/>whipsnake-RAD-locus-\1/" |paste - - |awk '!seen[$1]++' |tr '\t'
'\n' > tmp2 && mv tmp2 whipsnake-ddrad-RAD-loci-for-sambada-write_random_snp.fa

## Step 12 Make Garter Snake Genome BLASTdb
### get genome
wget
ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/077/635/GCF_001077635.1_Thamnophis_sirtali
s-6.0/GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.fna.gz
### unzip it
gunzip GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.fna.gz
```

```
### make blastdb
/usr/bin/makeblastdb -dbtype nucl -in GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.fna
&

## Step 13 Determine what genes contain the RAD loci
### get gene annotations
wget
ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/077/635/GCF_001077635.1_Thamnophis_sirtali
s-6.0/GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.gff.gz
### unzip them
gunzip GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.gff.gz


#####files-with-snps#####
bio01
bio03
bio17
bio08
bio12
wind
#####files-with-snps#####

## Step 14 get Genes with SNPs associated with environmental conditions
### loop through each file of SNPs associated with environmental conditions
perl -pe "s/whipsnake-RAD-locus-//g" whipsnake-ddrad-RAD-loci-for-sambada-
write_random_snp.fa > whipsnake-ddrad-RAD-loci-for-sambada-write_random_snp.fa2

while read i;do
  perl -pe "s/(\d+)_\d+_\w+/>\1\t/" ${i} > ${i}.locus
  cat whipsnake-ddrad-RAD-loci-for-sambada-write_random_snp.fa2 | paste - - |grep -f ${i}.locus
| tr '\t' '\n' > ${i}.fa
### perform blast with minimum evalue 1e-30
  /usr/bin/blastn -db GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.fna -query ${i}.fa -
evalue 1e-30 -num_threads 70 -outfmt 6 -out ${i}.fa-against-
GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.fna.blast
### get only best BLAST hit per RAD locus
  awk '!seen[$1]++' ${i}.fa-against-GCF_001077635.1_Thamnophis_sirtalis-
6.0_genomic.fna.blast > ${i}.fa-against-GCF_001077635.1_Thamnophis_sirtalis-
6.0_genomic.fna.blast.filtered
### convert BLAST coordinates to BED coordinates
  cut -f 2,9,10 ${i}.fa-against-GCF_001077635.1_Thamnophis_sirtalis-
6.0_genomic.fna.blast.filtered |awk '{if ($2 > $3) print $1,$3,$2; else print $1,$2,$3;}' OFS='\t'
FS='\t' |bedtools sort > ${i}.fa-against-GCF_001077635.1_Thamnophis_sirtalis-
6.0_genomic.fna.blast.filtered.bed
### determine what genes have the RAD loci
  bedtools intersect -a GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.gff -b ${i}.fa-
against-GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.fna.blast.filtered.bed | awk
```

```
'$3=="gene"' |cut -f 9 |cut -d ";" -f 3|perl -pe "s/Name=//g" |sort -u > ${i}.fa-against-
GCF_001077635.1_Thamnophis_sirtalis-6.0_genomic.fna.blast.filtered.bed.genes
  cp ${i}.fa-against-GCF_001077635.1_Thamnophis_sirtalis-
6.0_genomic.fna.blast.filtered.bed.genes ${i}.genes
done < files-with-snps

tail -n +1 bio01.genes bio03.genes bio17.genes bio08.genes bio12.genes wind.genes
==> bio01.genes <==
CSF3R
IGF2BP2
LOC106545699
MPC1
SYT1

==> bio03.genes <==
ACVR1B
ADGRL1
ADSSL1
ANP32A
BAIAP2L2
CACNA1I
CACNA1S
CNKSR1
DFNA5
DOCK7
FGF9
GNG4
KCNH5
LOC106538198
LOC106541221
LOC106545011
LOC106546904
LOC106547407
LOC106548980
LOC106549984
LOC106554113
LOC106554861
LOC106554868
MALL
NFASC
PAM
PLXND1
PRPF31
RHEB
RTN1
RYR2
SGSH
```

SLC35F2
STT3B
TCF7L2
ZFAND4

==> bio17.genes <==
DAGLA

==> bio08.genes <==
KIAA0408

==> bio12.genes <==

==> wind.genes <==
LOC106538264
LOC106547407
LOC106549984
LOC106555322

 # Calculate sequence depth/ coverage for all SNPs and all individuals

Software version:VCFtools v.0.1.16
https://vcftools.github.io/index.html

The Variant Call Format and VCFtools, Petr Danecek, Adam Auton, Goncalo Abecasis, Cornelis A. Albers, Eric Banks, Mark A. DePristo, Robert Handsaker, Gerton Lunter, Gabor Marth, Stephen T. Sherry, Gilean McVean, Richard Durbin and 1000 Genomes Project Analysis Group, **Bioinformatics**, 2011

```
# Get individual names
cut -d " " -f 1 CW53_17518SNP_final.ped > CW53_17518SNP_final.individuals

# Get SNP identifiers
cut -f 2 CW53_17518SNP_final.map > CW53_17518SNP_final.snps

# Get raw SNPs
wget https://www.dropbox.com/s/inrklf5mxha6gfa/whipsnake-ddrad-raw-variants-for-sambada-write_random_snp.vcf.gz

# Filter raw SNPs to keep only those in CW53_17518SNP_final.map
zgrep -f <(perl -pe "s/(.+)/\t\1\t/g" CW53_17518SNP_final.snps) \
whipsnake-ddrad-raw-variants-for-sambada-write_random_snp.vcf.gz | \
cat <(zgrep "^#" whipsnake-ddrad-raw-variants-for-sambada-write_random_snp.vcf.gz) - | \
gzip > whipsnake-ddrad-raw-variants-for-sambada-
write_random_snp_filterd_by_CW53_17518SNP_final.snps.vcf.gz

# Rename GR_KL_J57 to TU_KL_J57
zcat whipsnake-ddrad-raw-variants-for-sambada-
write_random_snp_filterd_by_CW53_17518SNP_final.snps.vcf.gz | \
perl -pe "s/GR_KL_J57/TU_KL_J57/g" |gzip > \
whipsnake-ddrad-raw-variants-for-sambada-
write_random_snp_filterd_by_CW53_17518SNP_final.snps_GR_KL_J57_renamed_to_TU_KL
_J57.vcf.gz

# Use VCFtools to keep only individuals in CW53_17518SNP_final.ped
vcftools --gzvcf whipsnake-ddrad-raw-variants-for-sambada-
write_random_snp_filterd_by_CW53_17518SNP_final.snps_GR_KL_J57_renamed_to_TU_KL
_J57.vcf.gz \
--keep CW53_17518SNP_final.individuals --recode --out CW53_17518SNP_final

# VCFtools running log
#VCFtools - 0.1.16
#(C) Adam Auton and Anthony Marcketta 2009

#Parameters as interpreted:
```

```
#         --gzvcf whipsnake-ddrad-raw-variants-for-sambada-
write_random_snp_filterd_by_CW53_17518SNP_final.snps_GR_KL_J57_renamed_to_TU_KL
_J57.vcf.gz
#         --keep CW53_17518SNP_final.individuals
#         --out CW53_17518SNP_final
#         --recode

#Using zlib version: 1.2.11
#Keeping individuals in 'keep' list
#After filtering, kept 53 out of 96 Individuals
#Outputting VCF file...
#After filtering, kept 17518 out of a possible 17518 Sites
#Run Time = 1.00 seconds

# Get DP (depth/coverage values for each individual at each SNP)
# with individual names
grep -v "^##" CW53_17518SNP_final.recode.vcf | \
cut -f 10- |perl -pe "s/.\/.:(\d+):\d+,\d+/\1/g"| \
perl -pe "s/.\/.:(\d+):.,./\1/g" \
> CW53_17518SNP_final.recode.vcf.DP.with.individual.names

# Get DP (depth/coverage values for each individual at each SNP)
# without individual names
grep -v "^#" CW53_17518SNP_final.recode.vcf | \
cut -f 10- |perl -pe "s/.\/.:(\d+):\d+,\d+/\1/g"| \
perl -pe "s/.\/.:(\d+):.,./\1/g" \
> CW53_17518SNP_final.recode.vcf.DP.without.individual.names

# Do summary analysis in R
# In R 3.6.3
test <- read.table("CW53_17518SNP_final.recode.vcf.DP.with.individual.names",header=T)
test.df <- t(as.data.frame(do.call(cbind, lapply(test, summary))))

# DP values (coverage for each individual at all SNPs)
```

Individual	Mininum	1st_Quartile	Median	Mean	3rd_Quartile	Maximum
AL_BO_721	0	28	41	39.860258	53	130
AL_PE_1856	0	51	68	65.832173	82	175
BG_SO_1415	0	5	10	10.463637	15	45
BG_SO_764	0	21	29	28.712581	37	97
TU_KL_J57	0	50	68	64.256251	81	196
GR_LO_763	0	60	88	83.578491	110	265
GR_SA_D19	0	15	22	21.568101	28	76
GR_SA_D8	0	5	8	8.671310	12	31
HR_BR_B01	0	17	24	23.979906	31	78
HR_BR_B02	0	20	28	27.667085	35	98
HR_Oi_O02	0	15	27	30.276801	43	124

HR_PP_L11	0	9	15	16.298835	23	69	
HU_DF_DU2	0	13	19	20.025688	27	73	
HU_DF_DUJ22		0	42	58	57.825950	74	172
HU_DT_DF1	0	60	83	81.562907	105	243	
HU_FH_Z024	0	35	46	45.097328	57	149	
HU_HU_Gy697		0	7	12	13.609830	19	70
HU_PA_Z027	0	8	12	12.711668	17	48	
HU_PT_PV2	0	36	52	52.416143	68	194	
HU_PV_Gy698		0	37	50	48.822411	62	145
HU_PV_Gy838		0	7	13	13.459470	19	62
HU_PV_Gy925		0	50	74	70.666115	93	242
HU_PV_Gy955		0	12	17	18.335883	24	74
HU_PV_Gy957		0	10	16	16.918427	23	61
HU_PV_Z003	0	16	22	21.831088	28	69	
HU_SH_Gy693		0	69	91	88.201050	111	240
HU_VB_Sz1	0	26	37	38.382007	50	135	
HU_VB_Sz12	0	36	49	49.369106	63	156	
HU_VB_Sz13	0	21	31	31.807398	42	118	
HU_VB_Sz16	0	23	33	33.880637	44	126	
HU_VB_Sz17	0	54	73	70.395251	89	242	
HU_VB_Sz2	0	24	35	36.516954	48	135	
HU_VB_Sz6	0	27	39	39.501313	51	133	
HU_VB_Sz7	0	26	37	37.343761	49	122	
HU_VB_Sz8	0	24	36	36.950851	49	129	
MK_BK_1577		0	38	53	51.825380	67	133
MK_PJ_1632	0	35	47	45.949309	58	125	
MK_Pi_1514	0	15	21	20.956159	27	69	
RS_BU_Y3	0	6	10	9.853579	14	44	
RS_CU_1708	0	52	76	73.368250	97	208	
RS_ZL_Y5	0	38	55	56.483731	75	226	
RS_ZL_Y6	0	63	84	80.624158	101	240	
UA_BDT_1184		0	71	104	97.963409	129	312
UA_BO_2389	0	6	9	10.009133	14	44	
UA_KU_1185	0	13	21	21.569757	30	87	
UA_KU_1186	0	41	59	57.026772	74	174	
UA_KU_2383	0	46	68	67.596073	90	195	
UA_MM_2382		0	80	115	109.534193	145	362
UA_PE_2384	0	39	53	51.941660	65	153	
UA_PT_2385	0	38	58	56.260817	76	182	
UA_SK_1183	0	40	53	51.311051	64	145	
UA_VU_2391	0	5	8	8.418141	12	31	
UA_YA_2386	0	54	78	74.233931	98	268	

```
# In R 3.6.3
test <- read.table("CW53_17518SNP_final.recode.vcf.DP.without.individual.names",header=F)
summary(unlist(test))
```

DP values (coverage values for all SNPs and all individuals)

Mininum	1st_Quartile	Median	Mean	3rd_Quartile	Maximum
0.00	18.00	36.00	43.43	63.00	362.00

Dolichophis caspius Project

Exploratory Data Analysis

16/09/2020

Data set of environmental variables

1. We used R to extract the data from raster files (=worldclim data).

For each samples, environmental data have been extracted from freely available raster files of environemental variables.

We used current climate data (1970-2000) from the WorldClim Version 2 dataset (Fick & Hijmans, 2017). The environmental variables encompassed average monthly mean and maximum temperature (°C), precipitation (mm), wind speed (m.s-1), water vapor pressure (kPa), solar radiation (kJ.m-2.day-1), and the 19 WorldClim bioclimatic variables at 30 arc-second resolution (Supplementary Table 3). The values of the WorldClim variables at the sample locations were extracted from the raster datasets (extract function in the R package raster).

We have also extracted land cover data (see in script) but these data have not been included in the final data set because some European countries were not covered (See CORINE landcover data set, available at: https://land.copernicus.eu/pan-european/corine-land-cover).

The final data set count 90 samples and 98 variables.

Codes to extract environmental data from the raster files and construct the data set are provided in the .Rmd document but not shown on the output.

2. From this initial data set, 53 samples and 43 variables were selected (see CW_53ID_47var.csv).

3. From these 43 variables, 8 have been removed before correlation analysis: January, February, November and December for the variables "Wind speed" and "Solar radiation".

```
# Import the data set including 53 individuals and 47 variables
# change the path to import the dataset
CW53ID47var.tot <- read.csv("CW_53ID_47var.csv",header=T,check.names=F)

#from these variables, 8 have been removed before correlation analysis
#January, February, November and December for Wind Speed and Solar radiation

# I have the library ratser loaded and there is a conflict with the function "select" between
the package raster and dplyr. To use the select function from dplyr, I have added this comman
d:
conflict_prefer("select", "dplyr")

CW53ID47var.tot %>% select (-c(srad01.extract, srad02.extract, srad11.extract, srad12.extract
, wind01.extract, wind02.extract, wind11.extract, wind12.extract)) -> CW53ID39var
# 39 environmental variables remain.

# we add the countries (for later analyses)
CW53ID39var$country <- substr(CW53ID47var.tot[,1], 1, 2)

# Get the headers of the data set
```

```
headers <- colnames(CW53ID39var)
print(headers)
```

```
##  [1] "ddRAD.code"     "longitude"      "latitude"       "srad03.extract"
##  [5] "srad04.extract" "srad05.extract" "srad06.extract" "srad07.extract"
##  [9] "srad08.extract" "srad09.extract" "srad10.extract" "wind03.extract"
## [13] "wind04.extract" "wind05.extract" "wind06.extract" "wind07.extract"
## [17] "wind08.extract" "wind09.extract" "wind10.extract" "bio01.extract"
## [21] "bio02.extract"  "bio03.extract"  "bio04.extract"  "bio05.extract"
## [25] "bio06.extract"  "bio07.extract"  "bio08.extract"  "bio09.extract"
## [29] "bio10.extract"  "bio11.extract"  "bio12.extract"  "bio13.extract"
## [33] "bio14.extract"  "bio15.extract"  "bio16.extract"  "bio17.extract"
## [37] "bio18.extract"  "bio19.extract"  "grass"          "imp"
## [41] "clc"            "fty"            "country"
```

```
#colnames(headers) <- "List of the variables in the final table"

#kable(headers, booktabs = T)
```

Correlation analysis of environmental variables

Because multicollinearity of variables can result in erroneous modelling (Prunier et al., 2015), we removed highly correlated environmental predictors. The variance inflation factor (VIF) was calculated for each environmental variable to detect the presence of collinearity (Belsley et al., 1980) with a cut-off value of VIF < 5 following Stucki et al., (2017). The VIF was calculated using the R package fmsb.

Notes from the PhD of Sylvie STUCKI (page 54):
The selection of explanatory variables for a multivariate model must take into account their mutual correlations. If the predictors are too interdependent, the estimated values for the regression parameters become unstable. In this case, these parameters can vary radically by including or excluding some points of the model, whereas they should remain stable during a small modification of the points used in the regression. This phenomenon of multicollinearity can be estimated by calculating the variance inflation factor (variance inflation factor, VIF). It depends on the coefficient of determination R2 of the linear regression of a predictor according to the others.

VIF = 1 / (1-R2)

The VIF is calculated for each explanatory variable and the limit value is usually set at about 5 (Dobson and Barnett, 2008). In the case of bivariate models, the VIF is also calculated with a linear regression between predictors.
A VIF smaller than 5 corresponds to a correlation between predictors smaller than 0.9.

(S. Stucki set the limit at 0.8, a VIF of about 2.8, to limit multicollinearity and reduce the calculation time.

- The smallest possible value of VIF is one (absence of multicollinearity.
- As a rule of thumb, a VIF value that exceeds 5 or 10 indicates a problematic amount of collinearity (James et al. 2014).
  (http://www.sthda.com/english/articles/39-regression-model-diagnostics/160-multicollinearity-essentials-and-vif-in-r/)

```
##The following function uses three arguments. The first is a matrix or data frame of the ex
planatory variables,
#the second is the threshold value to use for retaining variables, and the third is a logical
argument indicating if text output
#is returned as the stepwise selection progresses. The output indicates the VIF values for ea
```

```
ch variable after each stepwise
#comparison. The function "vif_func" calculates the VIF values for all explanatory variables,
removes the variable with the highest value,
#and repeats until all VIF values are below the threshold. The final output is a list of var
iable names with VIF values that fall
#below the threshold. With the remaining variables we can create a linear model using explana
tory variables with less collinearity.

vif_func<-function(in_frame,thresh=10,trace=T,...){

  if(any(!'data.frame' %in% class(in_frame))) in_frame<-data.frame(in_frame)

  #get initial vif value for all comparisons of variables
  vif_init<-NULL
  var_names <- names(in_frame)
  for(val in var_names){
    regressors <- var_names[-which(var_names == val)]
    form <- paste(regressors, collapse = '+')
    form_in <- formula(paste(val, '~', form))
    vif_init<-rbind(vif_init, c(val, VIF(lm(form_in, data = in_frame, ...))))
  }
  vif_max<-max(as.numeric(vif_init[,2]), na.rm = TRUE)

  if(vif_max < thresh){
    if(trace==T){ #print output of each iteration
      prmatrix(vif_init,collab=c('var','vif'),rowlab=rep('',nrow(vif_init)),quote=F)
      cat('\n')
      cat(paste('All variables have VIF < ', thresh,', max VIF ',round(vif_max,2), sep=''),'\
n\n')
    }
    return(var_names)
  }
  else{

    in_dat<-in_frame

    #backwards selection of explanatory variables, stops when all VIF values are below 'thres
h'
    while(vif_max >= thresh){

      vif_vals<-NULL
      var_names <- names(in_dat)

      for(val in var_names){
        regressors <- var_names[-which(var_names == val)]
        form <- paste(regressors, collapse = '+')
        form_in <- formula(paste(val, '~', form))
        vif_add<-VIF(lm(form_in, data = in_dat, ...))
        vif_vals<-rbind(vif_vals,c(val,vif_add))
      }
      max_row<-which(vif_vals[,2] == max(as.numeric(vif_vals[,2]), na.rm = TRUE))[1]

      vif_max<-as.numeric(vif_vals[max_row,2])
```

```
      if(vif_max<thresh) break

      if(trace==T){ #print output of each iteration
        prmatrix(vif_vals,collab=c('var','vif'),rowlab=rep('',nrow(vif_vals)),quote=F)
        cat('\n')
        cat('removed: ',vif_vals[max_row,1],vif_max,'\n\n')
        flush.console()
      }

      in_dat<-in_dat[,!names(in_dat) %in% vif_vals[max_row,1]]

    }

    return(names(in_dat))

  }

}

# we use the function for quantitative variables only (and we remove id and geocoordinate col
umns)
results.vif <- vif_func(in_frame=CW53ID39var[,-c(1:3,39:43)],thresh=5,trace=T)
```

The following variables are selected:

- wind04.extract
- bio01.extract
- bio03.extract
- bio07.extract
- bio08.extract
- bio12.extract
- bio17.extract

Principal Component Analysis

Question: If you do a PCA only on the environmental variable for each sample, will they cluster out the same way as the genetics?

Note: A PCA transforms a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called principal components (PC).

```
# Import the dataset that contains also the population data
CW53ID47var.tot2 <- read.csv("CW_53ID_47var_v2.csv",header=T,check.names=F)

# we follow the same steps as previously
#from these variables, 8 have been removed before correlation analysis
#January, February, November and December for Wind Speed and Solar radiation

# I have the library ratser loaded and there is a conflict with the function "select" between
the package raster and dplyr. To use the select function from dplyr, I have added this comman
d:
conflict_prefer("select", "dplyr")
```

```
CW53ID47var.tot2 %>% select (-c(srad01.extract, srad02.extract, srad11.extract, srad12.extract
, wind01.extract, wind02.extract, wind11.extract, wind12.extract)) -> CW53ID39var2
# 39 environmental variables remain.

# we add the countries (for later analyses)
CW53ID39var2$country <- substr(CW53ID47var.tot2[,1], 1, 2)


data.pca.env <- CW53ID39var2 %>% select (country, population , wind04.extract,bio01.extract,
bio03.extract, bio07.extract,
                                         bio08.extract, bio12.extract, bio17.extract)

  colnames(data.pca.env) <- c("country", "population", "wind04", "bio01", "bio03", "bio07", "
bio08", "bio12", "bio17")

# give id sample as row names
rownames(data.pca.env) <- CW53ID39var2$ddRAD.code

# apply PCA
# install.packages("FactoMineR")
# library(FactoMineR)
#res.pca <- PCA(data.pca.env [,-1],
#               scale.unit = TRUE,# standardize the the data
#                ncp = 5, # number of dimension to keep in the anaylsis
 #               graph = FALSE)

res.pca <- prcomp(data.pca.env[, -c(1,2)],  scale = TRUE)

var <- get_pca_var(res.pca)


# eigenvalues
kable(get_eigenvalue(res.pca), booktabs = T, "pipe",
             caption = "Eigenvalues.")
```

Eigenvalues.

	eigenvalue	variance.percent	cumulative.variance.percent
Dim.1	2.4386448	34.837782	34.83778
Dim.2	1.9769628	28.242326	63.08011
Dim.3	1.3444574	19.206534	82.28664
Dim.4	0.7497715	10.711021	92.99766
Dim.5	0.2698724	3.855320	96.85298
Dim.6	0.1343005	1.918578	98.77156
Dim.7	0.0859907	1.228438	100.00000

Scree plot

Visualize eigenvalues (scree plot): Shows the percentage of variances explained by each principal component.

```
fviz_eig(res.pca, caption = "")
```

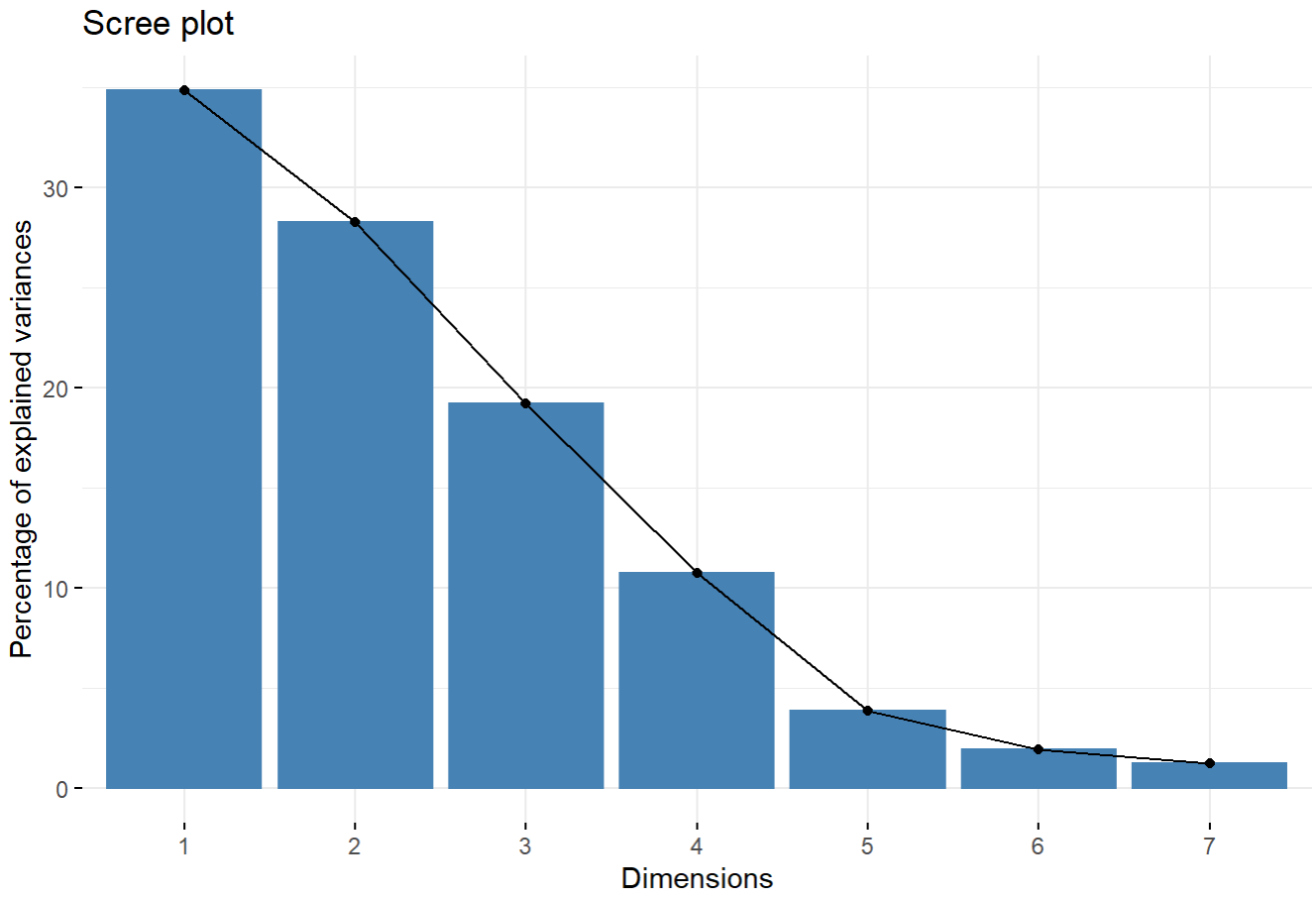


Figure 1. PCA of the environmental predictor variables. Scree plot showing the percentage of variances explained by each principal component.

Eigenvalues can be used to determine the number of principal components to retain after PCA (Kaiser 1961):
* An eigenvalue > 1 indicates that PCs account for more variance than accounted by one of the original variables in standardized data. This is commonly used as a cutoff point for which PCs are retained. This holds true only when the data are standardized.
* You can also limit the number of component to that number that accounts for a certain fraction of the total variance. For example, if you are satisfied with 70% of the total variance explained then use the number of components to achieve that.

Correlation circle and quality of representation

```
# Color by cos2 values: quality on the factor map
fviz_pca_var(res.pca, col.var = "cos2",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE, # Avoid text overlapping
             caption = ""
             )
```

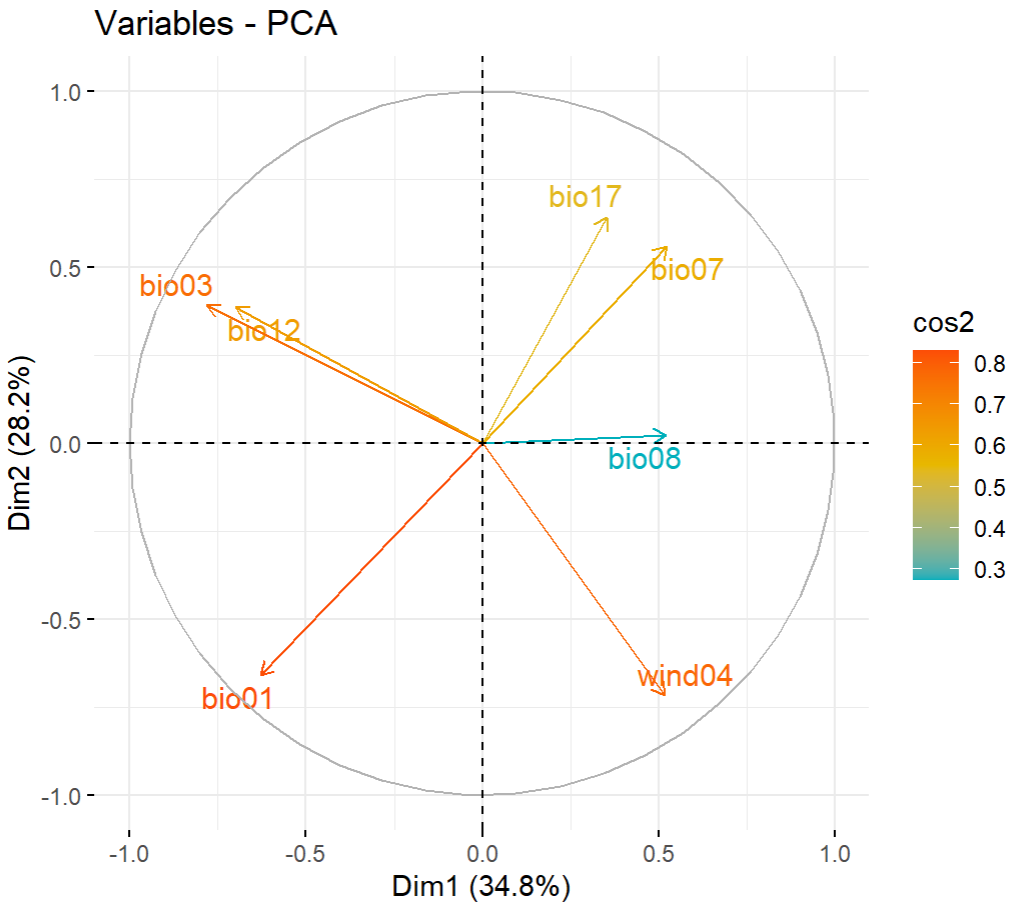


Figure 2. PCA of the environmental predictor variables. Variable correlation plot. The colors represents the quality of representation (cos2) of the variables.

The plot above is also known as variable correlation plots. It shows the relationships between all variables. It can be interpreted as follow:

- Positively correlated variables are grouped together.
- Negatively correlated variables are positioned on opposite sides of the plot origin (opposed quadrants).
- The distance between variables and the origin measures the quality of the variables on the factor map. Variables that are away from the origin are well represented on the factor map.

The quality of representation of the variables on factor map is called cos2 (square cosine, squared coordinates).

Contributions of variables to PCs

The contributions of variables in accounting for the variability in a given principal component are expressed in percentage. The quality of representation of the variables on factor map is called cos2 (square cosine, squared coordinates).

```
corrplot(var$cos2, is.corr=FALSE)
```

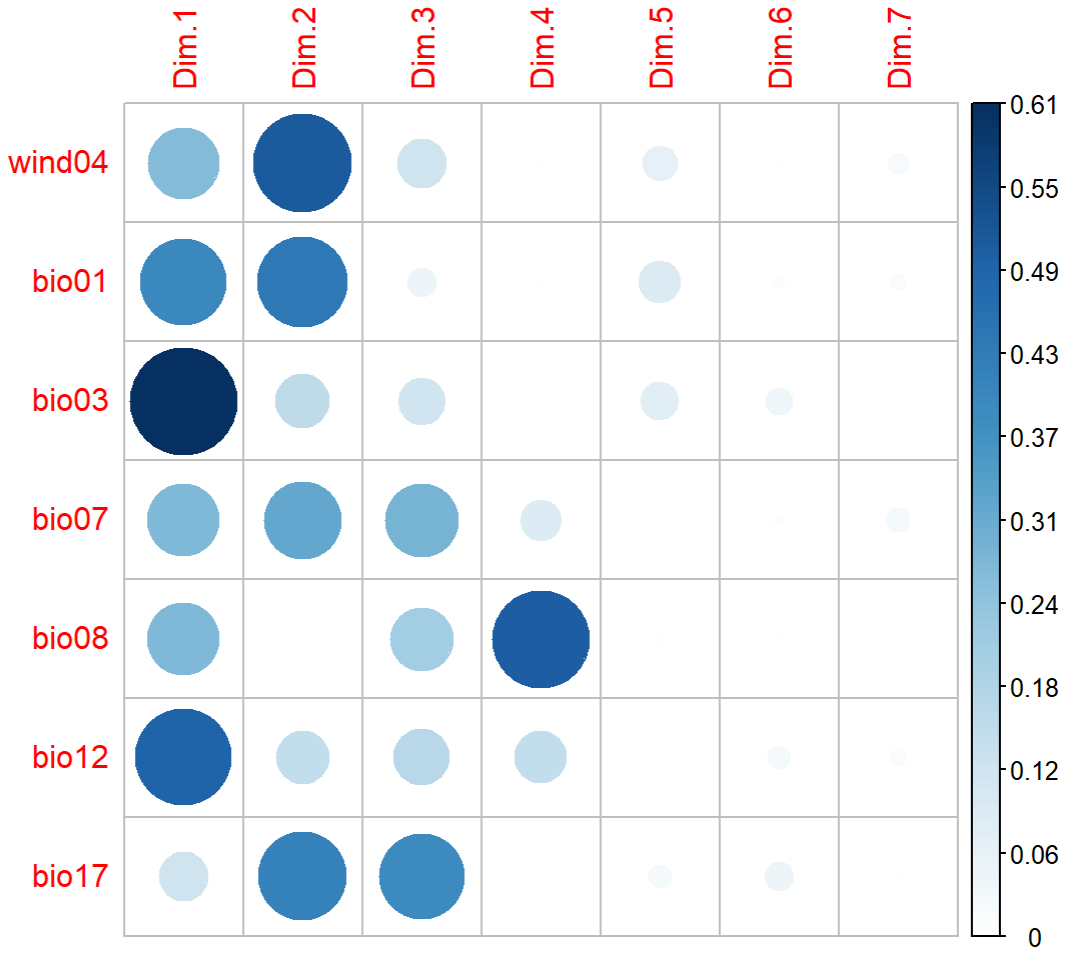


Figure 3. Quality of representation (cos2) of the variables.

Note that, A high cos2 indicates a good representation of the variable on the principal component. In this case the variable is positioned close to the circumference of the correlation circle.
A low cos2 indicates that the variable is not perfectly represented by the PCs. In this case the variable is close to the center of the circle.
Variables that are closed to the center of the plot are less important for the first components.

The correlation between a variable and a principal component (PC) is used as the coordinates of the variable on the PC.

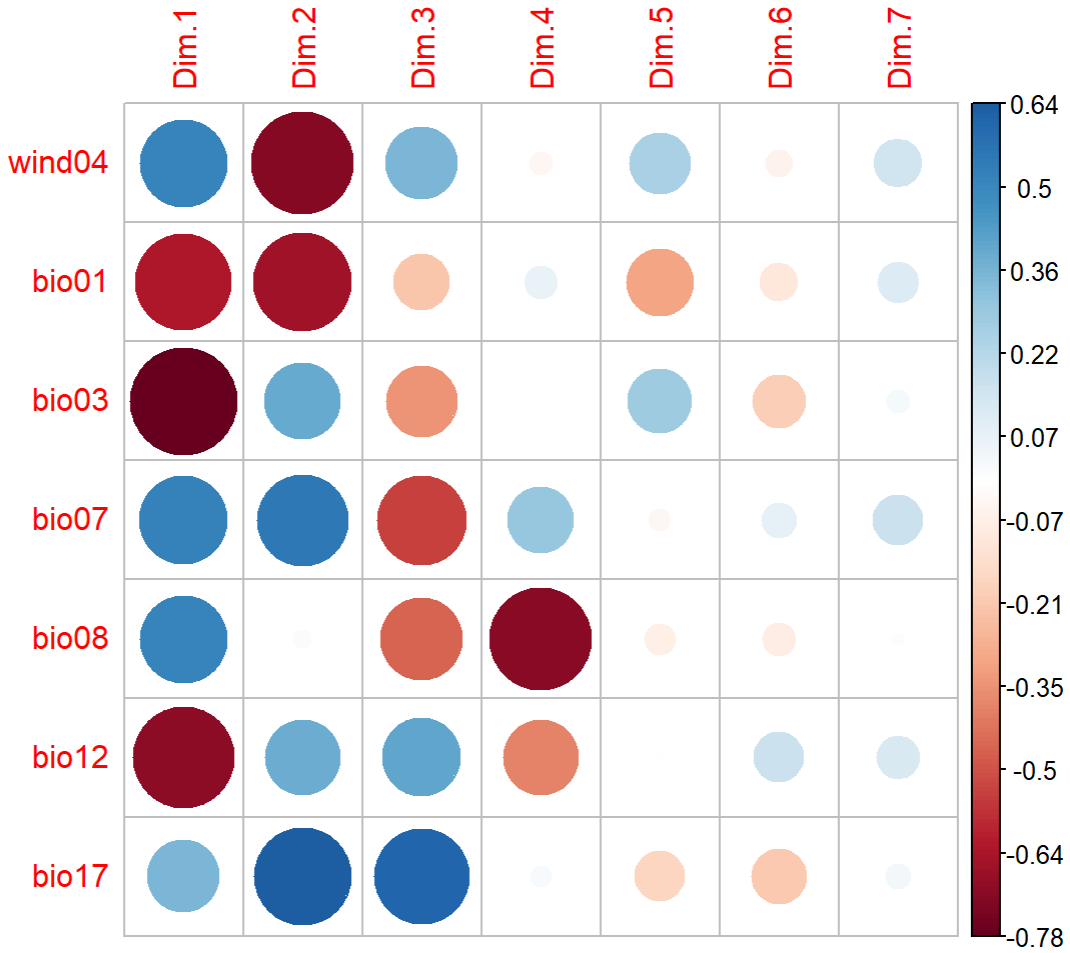
```
# Coordinates of variables
kable(var$coord, booktabs = T, "pipe",
              caption = "Eigenvalues.")
```

Eigenvalues.

	Dim.1	Dim.2	Dim.3	Dim.4	Dim.5	Dim.6	Dim.7
wind04	0.5174529	-0.7164935	0.3563262	-0.0365007	0.2532967	-0.0504892	0.1545009
bio01	-0.6292828	-0.6594686	-0.2145583	0.0756026	-0.3073388	-0.0983980	0.1149524
bio03	-0.7817712	0.3924930	-0.3440186	-0.0018026	0.2779706	-0.1945436	0.0362800
bio07	0.5238856	0.5595169	-0.5344000	0.2993963	-0.0321364	0.0818477	0.1718468

bio08	0.5222110	0.0215447	-0.4604000	-0.7108077	-0.0645508	-0.0731720	0.0097379
bio12	-0.7006732	0.3847408	0.4113856	-0.3832726	-0.0049857	0.1692975	0.1273150
bio17	0.3544304	0.6427298	0.6215427	0.0306037	-0.1695993	-0.2085836	0.0418877

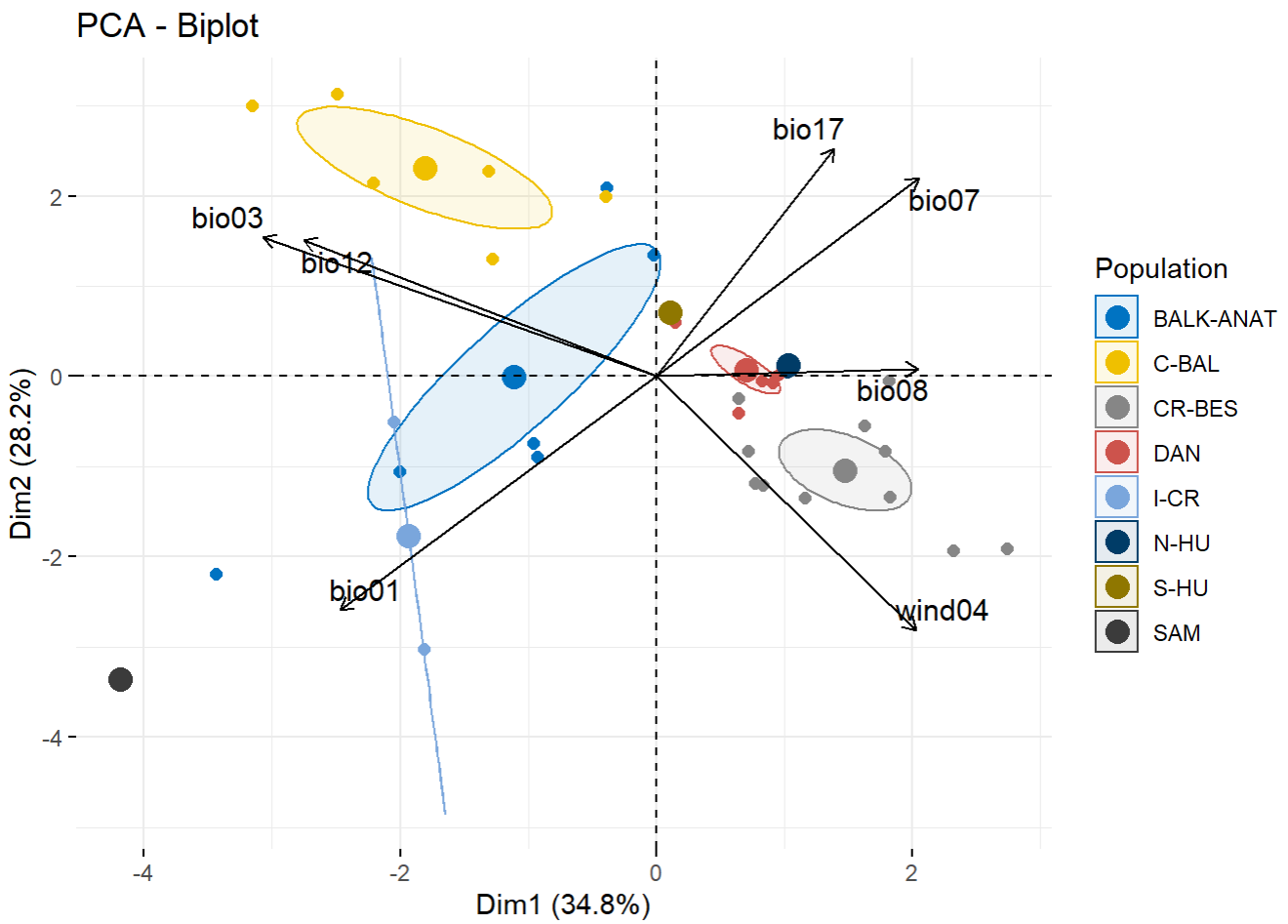
```
corrplot(var$coord, is.corr=FALSE)
```



Graph of individuals

The variable “population” will be used as grouping variable.

```
plot9 <- fviz_pca_biplot(res.pca, geom.ind = "point", pointshape = 21,
             pointsize = 2,
             fill.ind = data.pca.env$population,
             col.ind = "black",
             palette = "jco",
             addEllipses = TRUE,
             ellipse.type = "confidence", # add confidence ellipses
             label = "var",
             col.var = "black",
             repel = TRUE,      # Avoid label overplotting
             habillage = data.pca.env$population,
             legend.title = "Population")
plot(plot9, caption = "")
```




```
plot9.1 <- fviz_pca_biplot(res.pca, geom.ind = "point", pointshape = 21,
             pointsize = 2,
             fill.ind = data.pca.env$population,
             col.ind = "black",
             palette = "jco",
             label = "var",
             col.var = "black",
             repel = TRUE,      # Avoid label overplotting
             habillage = data.pca.env$population,
             legend.title = "Population",
             mean.point = FALSE) # remove mean points

plot9.2 <- fviz_pca_biplot(res.pca, geom.ind = "point", pointshape = 21,
             pointsize = 2,
             fill.ind = data.pca.env$population,
             col.ind = "black",
             palette = "jco",
             addEllipses = TRUE,
             ellipse.type = "confidence", # add confidence ellipses
             label = "var",
             col.var = "black",
             repel = TRUE,      # Avoid label overplotting
             habillage = data.pca.env$population,
             legend.title = "Population",
             mean.point = FALSE)# remove mean points
```

```
plot9.3 <- fviz_pca_biplot(res.pca, geom.ind = "point", pointshape = 21,
             pointsize = 2,
             fill.ind = data.pca.env$population,
             col.ind = "black",
             palette = "jco",
             label = "var",
             col.var = "black",
             repel = TRUE,      # Avoid label overplotting
             habillage = data.pca.env$population,
             legend.title = "Population") # KEEP MEANS


# save the plot
#save_plot("PCA.tiff", plot8,
  #        base_aspect_ratio = 1.3 # make room for figure legend
  #        )

ggsave("PCA.tiff", plot = plot9, height = 10, width = 15,units = c("cm"),dpi = 400) # with co
nfidence ellipses and means
ggsave("PCA_no_means_no_ellipses.tiff", plot = plot9.1, height = 10, width = 15,units = c("cm
"),dpi = 400) # remove the dots showing the mean coordinates and remove ellipses
ggsave("PCA_no_means.tiff", plot = plot9.2,height = 10, width = 15,units = c("cm"),dpi = 400)
# remove the dots showing the mean coordinates and keep ellipses
ggsave("PCA_means_no_ellipses.tiff", plot = plot9.3,height = 10, width = 15,units = c("cm"),d
pi = 400) # remove the dots showing the mean coordinates and keep ellipses
```

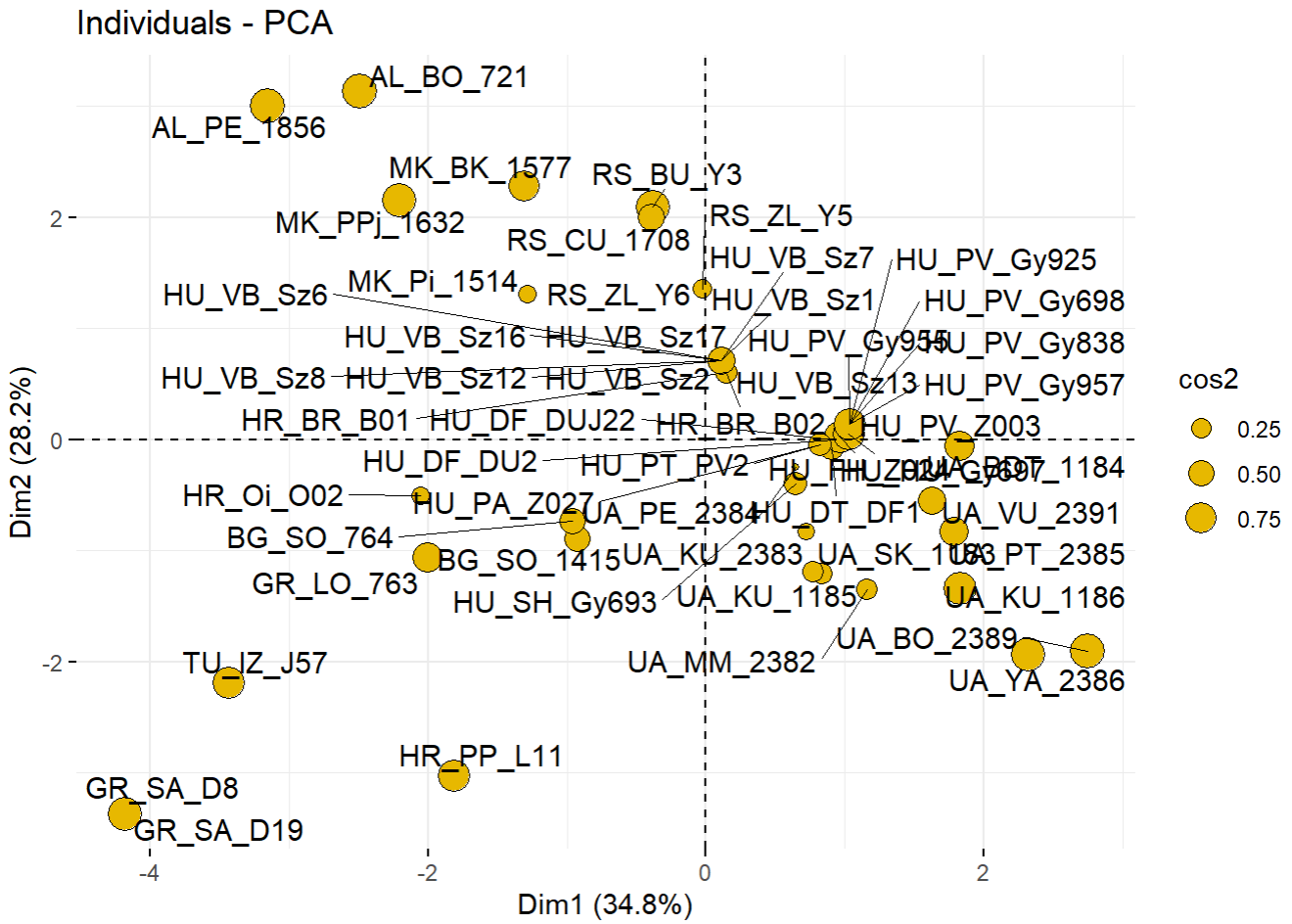
Figure 4. Principal component analysis (PCA) biplot of individuals (n = 53) and explanatory variables (n = 7). The biplot shows the PCA scores of the explanatory variables as vectors (in black) and individuals (colored dots) of each population: HERE WRITE THE MEANING FOR EACH ACRONYM OF POPULATION, of the first (x-axis) and second (y-axis) principal components (PCs). Individuals on the same side as a given variable should be interpreted as having a high contribution on it. The magnitude of the vectors (lines) shows the strength of their contribution to each PC. Vectors pointing in similar directions indicate positively correlated variables, vectors pointing in opposite directions indicate negatively correlated variables, and vectors at proximately right angles indicate low or no correlation. Colored ellipses (size determined by a 0.95-probability level) show the observations grouped by population. In each population, the mean point (barycenter) is displayed as a bigger dot.

To get a better visualisation of the data, I have also performed some clustering analyses (see: http://www.sthda.com/english/upload/hcpc_husson_josse.pdf).

Graph of individuals

Just to show you that point overlap a lot (the point size corresponds to the cos2 of the corresponding individuals)

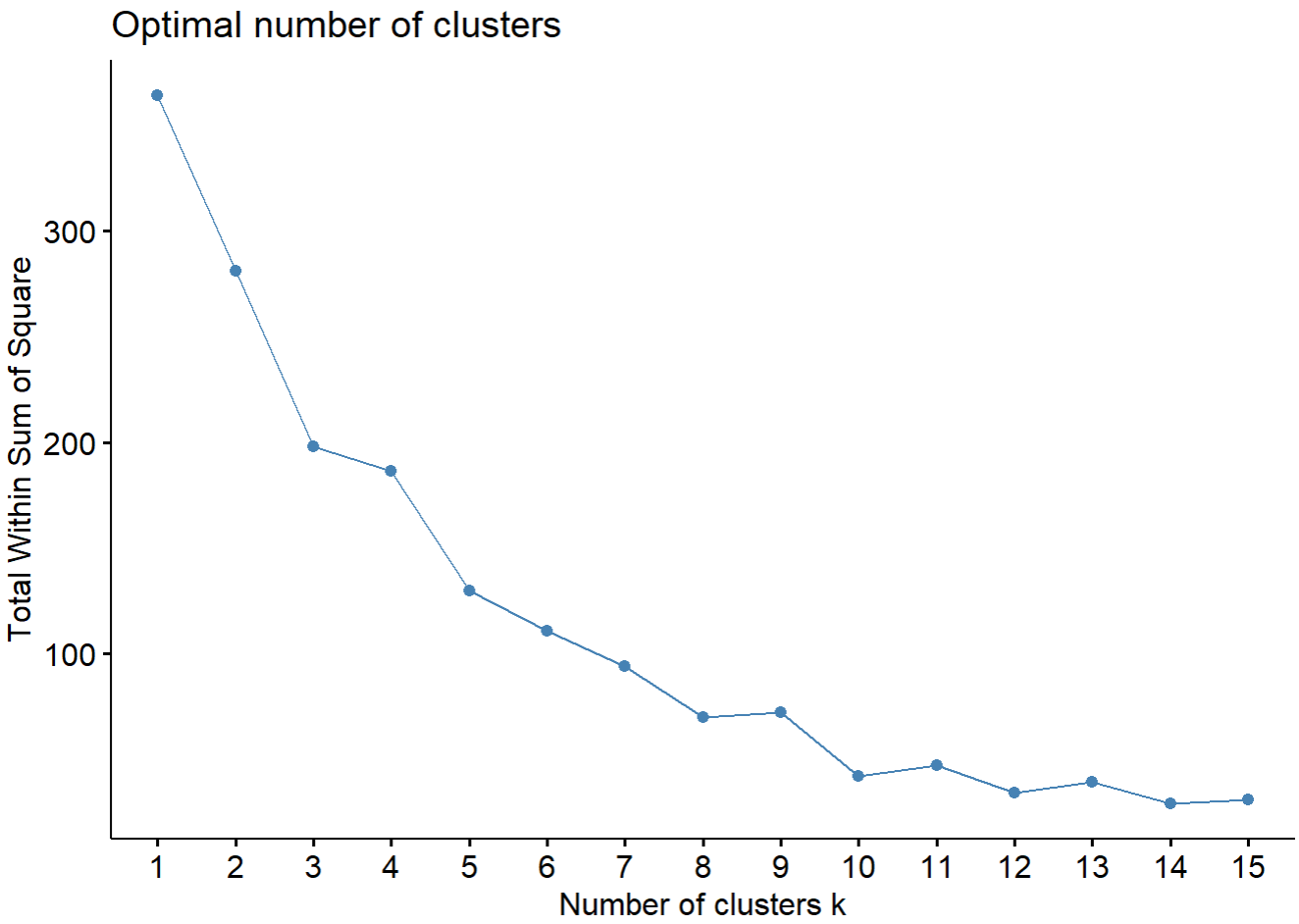
```
fviz_pca_ind(res.pca, pointsize = "cos2",
             pointshape = 21, fill = "#E7B800",
             repel = TRUE # Avoid text overlapping (slow if many points)
             )
```



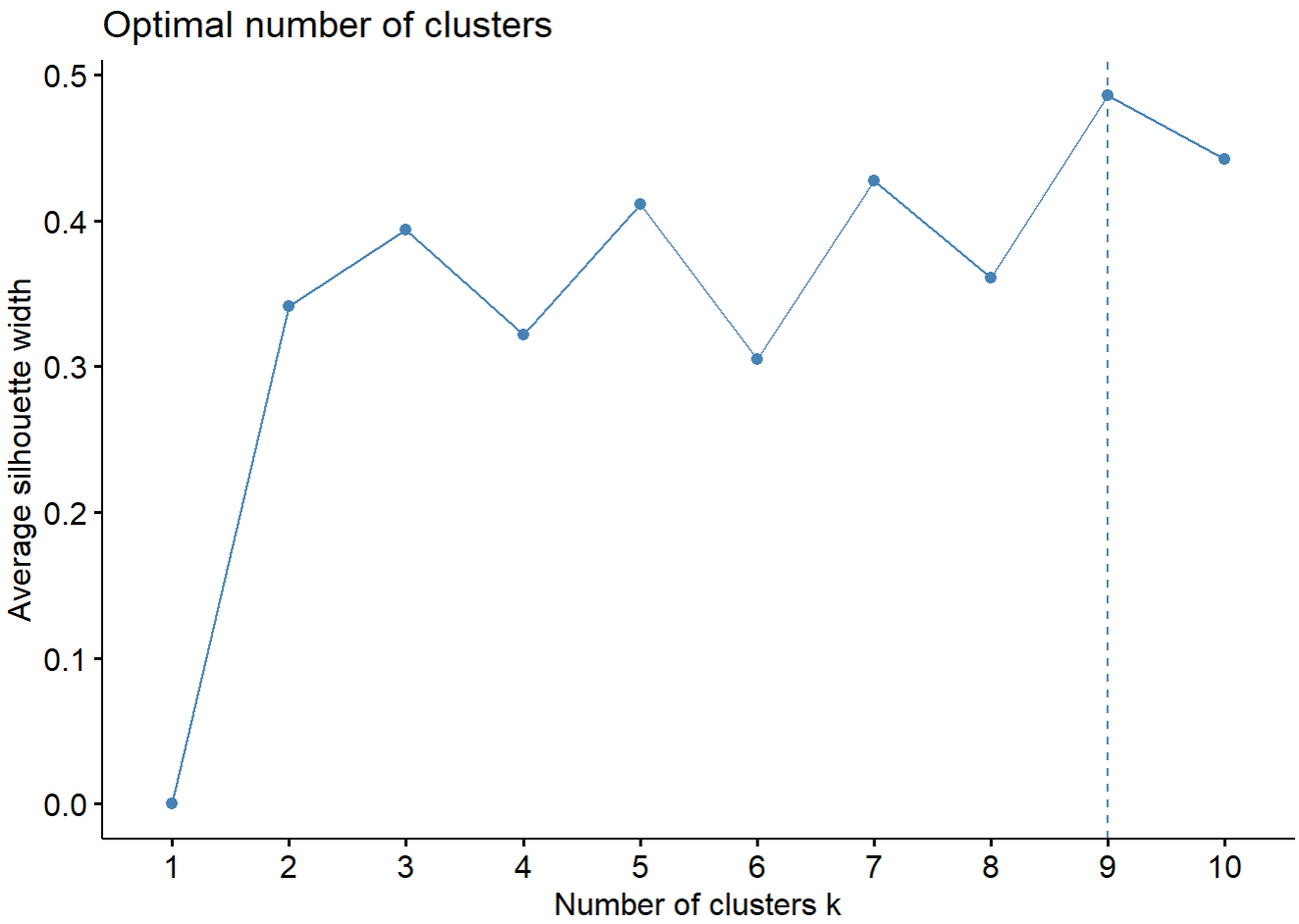
K-means clustering

(see: https://doi.org/10.3390/j2020016) K-means clustering (MacQueen 1967) is one of the most commonly used unsupervised machine learning algorithm for partitioning a given data set into a set of k groups (i.e. k clusters), where k represents the number of groups pre-specified by the analyst.

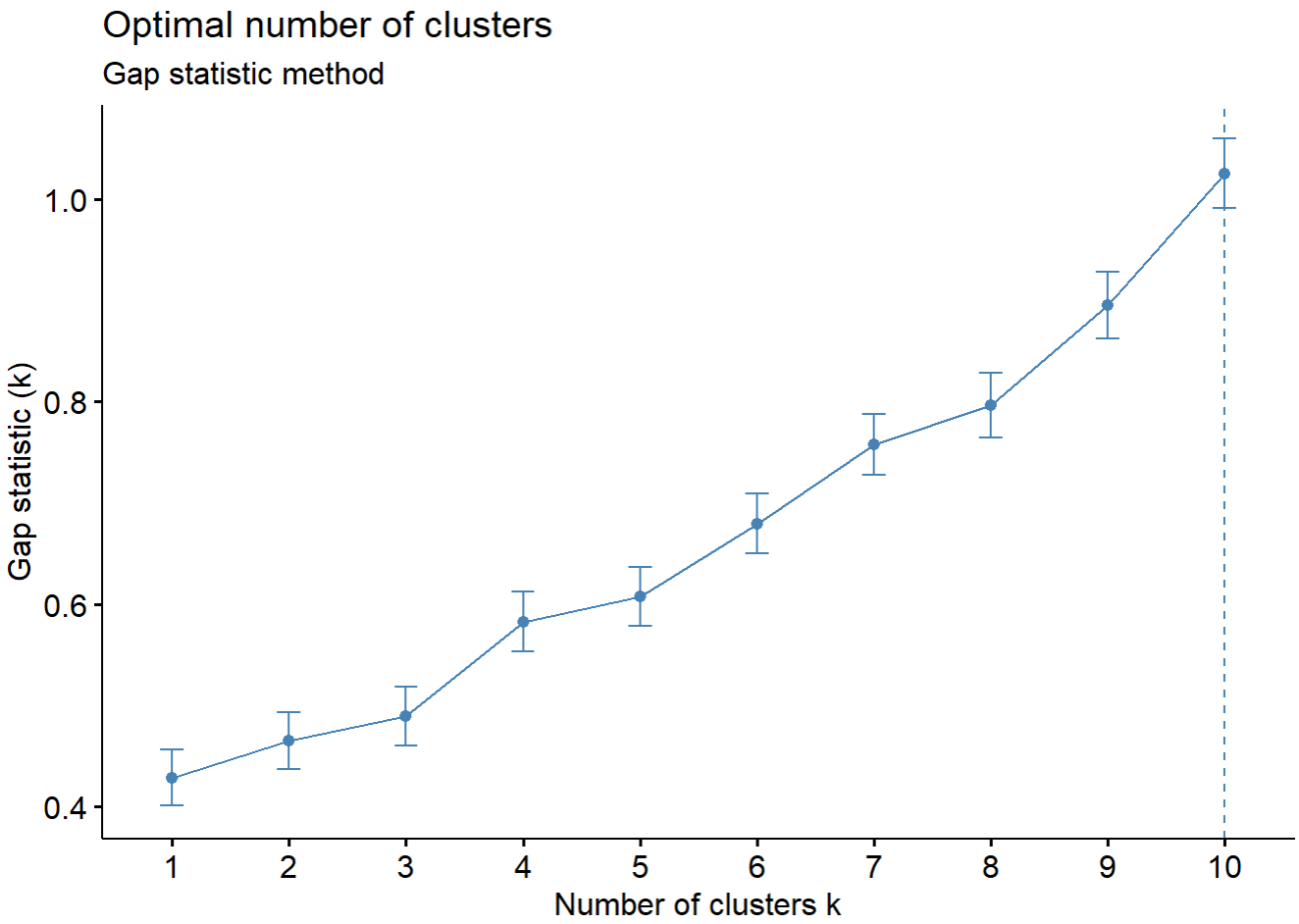
```
# scale the data
  mydata <- scale(data.pca.env[,-c(1,2)])
  # Determine number of clusters
# we use the elbow method
  n_clust1<-fviz_nbclust(mydata, kmeans, method = "wss",k.max = 15) # elbow methd (CITE)
  n_clust1
```




```
# Silhouette method
  n_clust2<- fviz_nbclust(mydata, kmeans, method = "silhouette")
  n_clust2
```




```
set.seed(123)
n_clust3 <- fviz_nbclust(mydata, kmeans, nstart = 25,  method = "gap_stat", nboot = 50)+
labs(subtitle = "Gap statistic method")
n_clust3
```



From these three methods, the number of clusters should be 9 or 10, which is more than the number of countries.

We set the number of clusters to 8 where the curve on the elbow method is starting to have a diminishing return.

The total variance in your data set that is explained by the clustering with k=8 is **81.8 %**.

sqrt(2/53)

Hierarchical clustering using the Ward method

We use number of clusters = 8.

```
# Use hcut() which compute hclust and cut the tree
  hc.cut <- hcut(mydata, k=8,  hc_method = "ward.D2")
  # Visualize dendrogram
  fviz_dend(hc.cut, show_labels = TRUE, rect = TRUE, cex = 0.6)
```

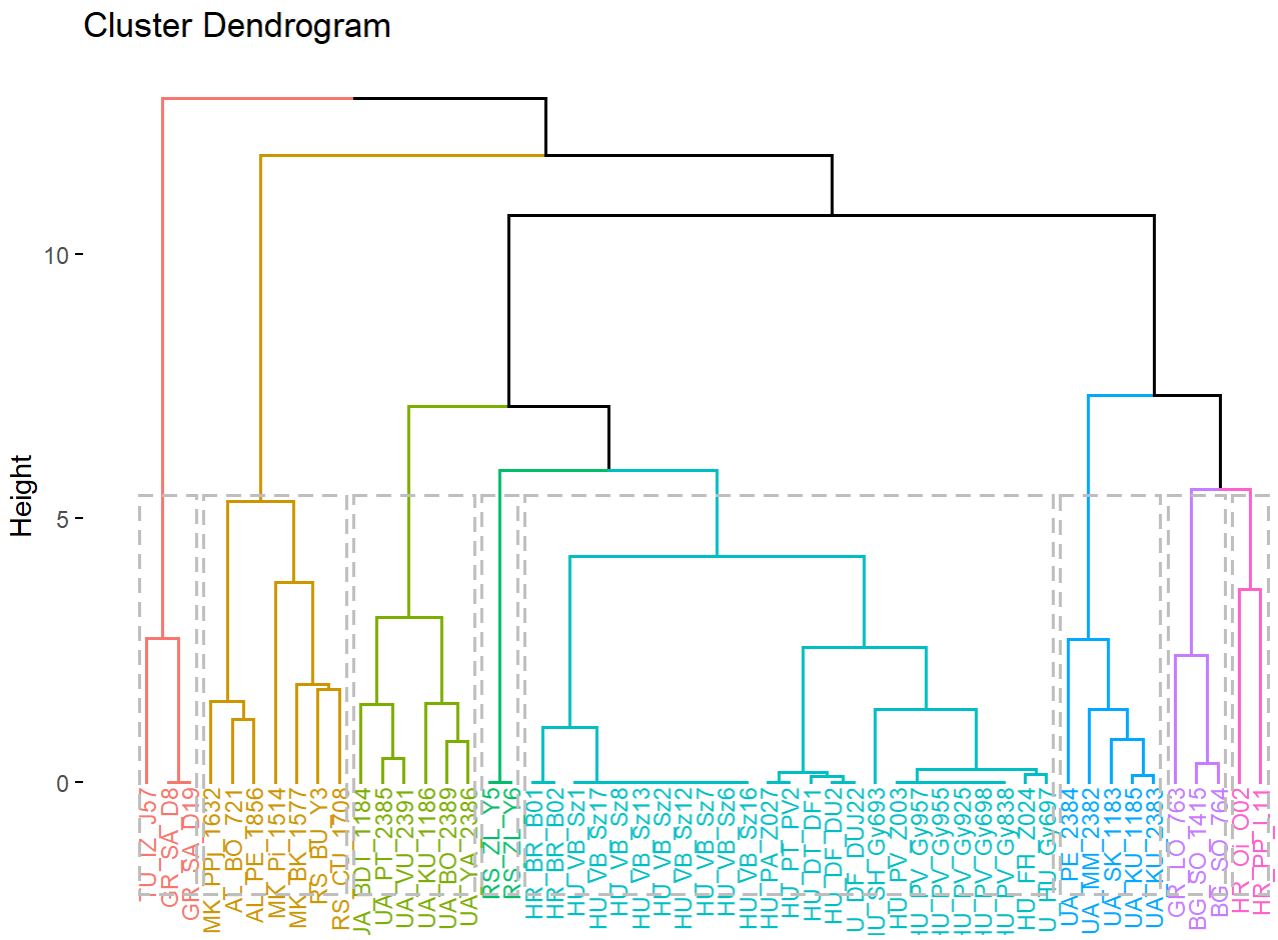


Figure 5. Hierarchical clustering using the Ward method: dendrogramm (k = 8).

```
# Visualize cluster
fviz_cluster(hc.cut, frame.type = "convex", labelsize = 6, repel = TRUE)
```

Figure 6. Hierarchical clustering using the Ward method: Factor map (k = 8).

Hierarchical Clustering on Principal Components (HCPC)

```
res.pca2 <- PCA(data.pca.env[,-c(1,2)],
                ncp = 3, # we use 3 PCs
                scale.unit = TRUE, # scale the data
                graph = FALSE)
# The PCA function keeps the first three dimensions (ncp=3) and thus the hierarchical cluster
ing only used these two dimensions.
res.hcpc <- HCPC(res.pca2, graph = FALSE)

# visualize clusters
plot.hcpc <- fviz_dend(res.hcpc,
          cex = 0.7,                     # Label size
          palette = "jco",               # Color palette see ?ggpubr::ggpar
          rect = TRUE, rect_fill = TRUE, # Add rectangle around groups
          rect_border = "jco",           # Rectangle color
          labels_track_height = 0.8, # Augment the room for labels
          main = "Dendrogram - Visualize clusters"
          )

plot(plot.hcpc, caption = "")
```

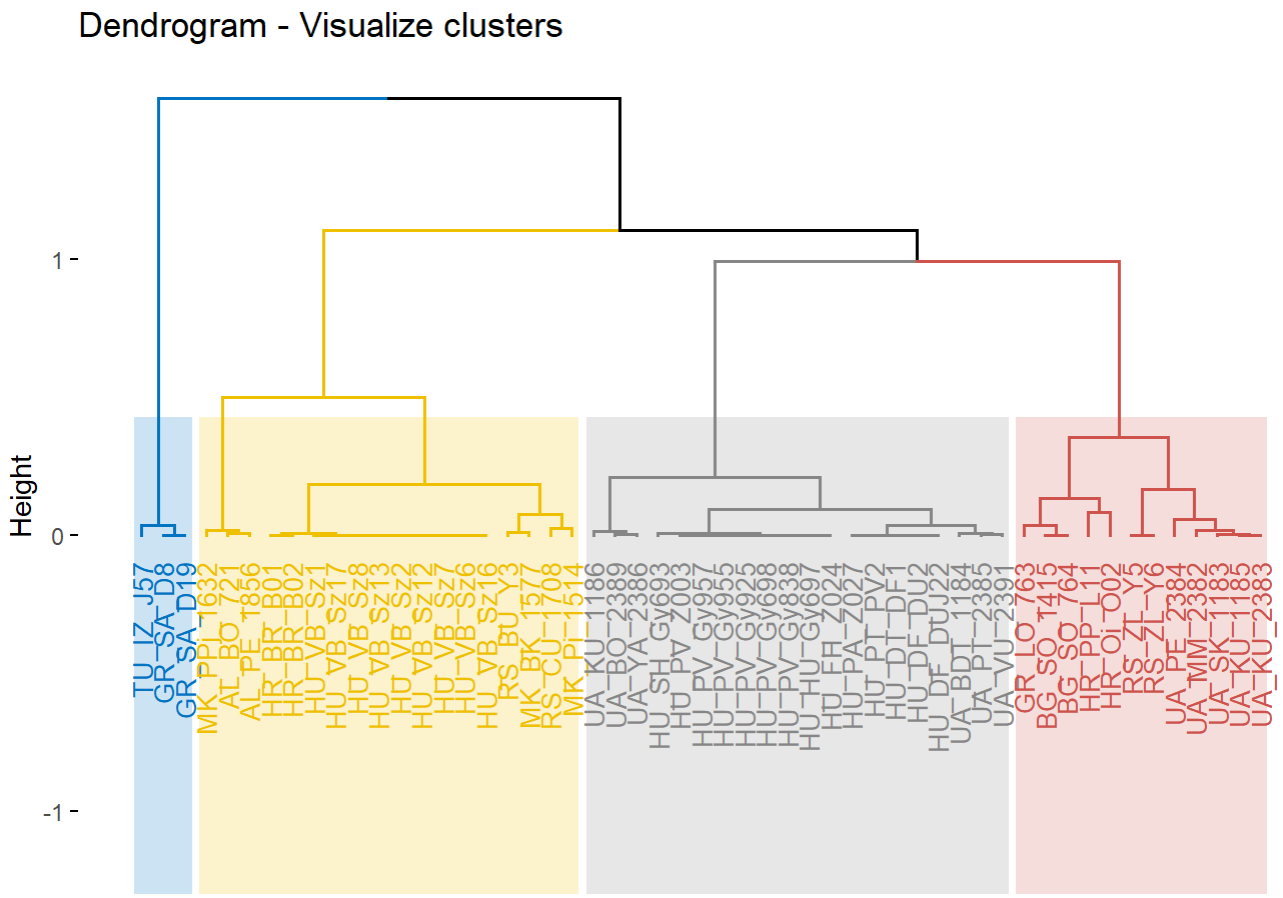


Figure 7. Dendrogram of the Hierarchical Clustering on Principal Components (HCPC).The HCPC detected four main clusters of individuals with a similar profile in terms of environmental variables.

The shape of the dendrogram suggests partitioning the individuals into four clusters

Visualize individuals on the principal component map and to color individuals according to the cluster they belong to.

```
# visualize individuals
plot.factor.map <- fviz_cluster(res.hcpc,
             repel = TRUE,            # Avoid label overlapping
             show.clust.cent = TRUE, # Show cluster centers
             palette = "jco",         # Color palette see ?ggpubr::ggpar
             ggtheme = theme_minimal(),
             main = "Factor map - Visualize individuals",
             labelsize = 6
             )
plot(plot.factor.map, caption ="")
```

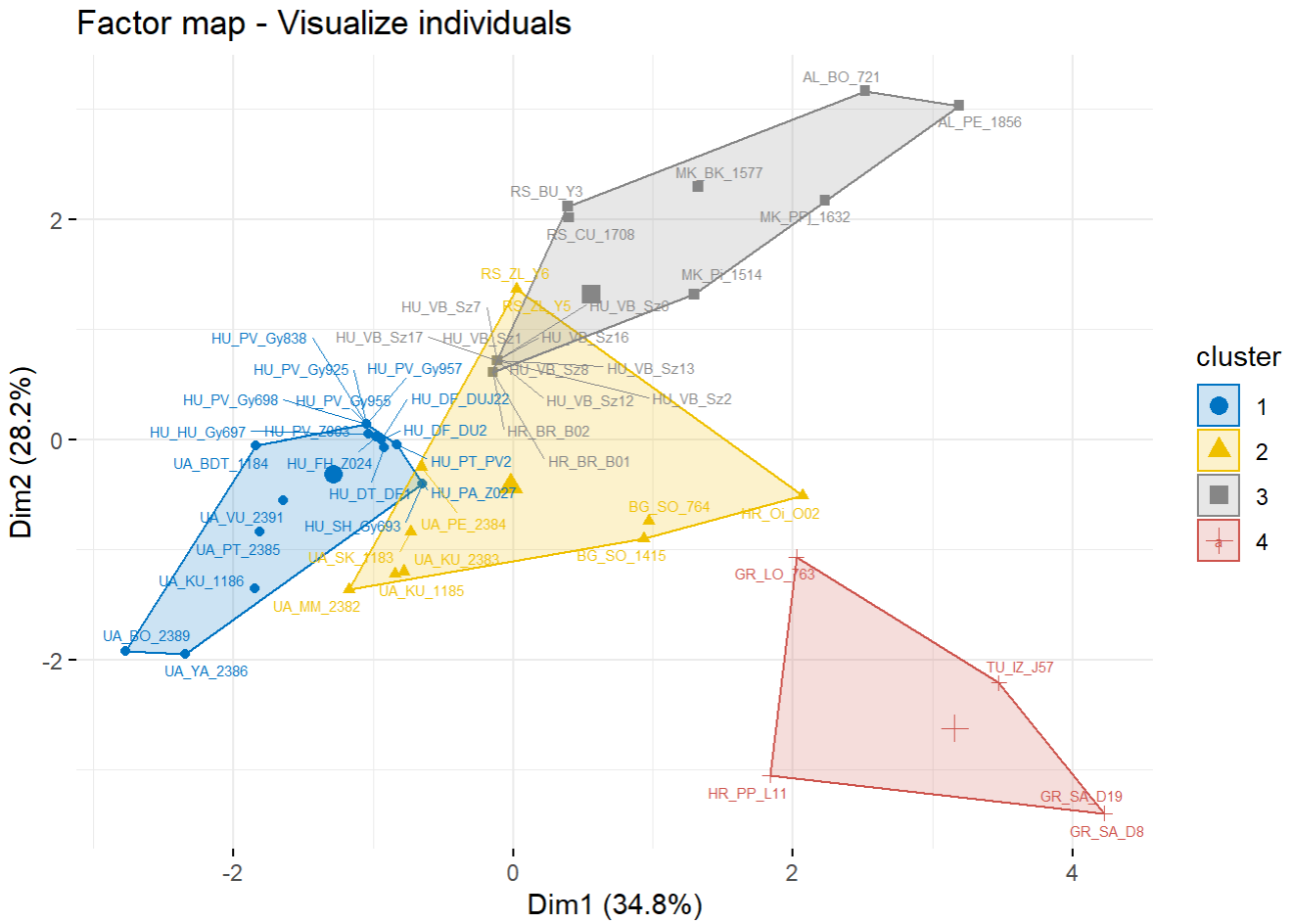


Figure 8. Hierarchical clustering on factor map (HCPC). Individual profiles are visualized in a space defined by seven environmental variables. Individuals closely positioned on the factor map and belonging to the same cluster share a similar profile in terms of environmental variables.

Environmental variables that describe the most each cluster

```
res.hcpc$desc.var$quanti
```

```
## $`1`
##           v.test Mean in category Overall mean sd in category  Overall sd
## bio08   4.347950          19.49250    15.070755       1.4760916   5.7091168
## bio07   2.508143          30.62000    29.564151       0.9942838   2.3632480
## wind04  2.195453           3.41000     3.233962       0.2681417   0.4501335
## bio03  -3.720301          27.59947    29.433655       1.1805445   2.7677382
## bio12  -3.976279         514.85000   591.792453      62.1781915 108.6297778
##              p.value
## bio08   1.374162e-05
## bio07   1.213677e-02
## wind04 2.813110e-02
## bio03   1.989856e-04
## bio12   7.000203e-05
##
## $`2`
##           v.test Mean in category Overall mean sd in category Overall sd
## bio17   3.026329           121.60    98.396226      15.6345771 26.6630545
## wind04  2.827822             3.60     3.233962       0.2863564  0.4501335
```

```
## bio07  -3.007954            27.52    29.564151      1.9405153  2.3632480
## bio08  -4.118070             8.31    15.070755      5.0113826  5.7091168
##             p.value
## bio17  2.475427e-03
## wind04 4.686593e-03
## bio07  2.630132e-03
## bio08  3.820589e-05
##
## $`3`
##           v.test Mean in category Overall mean sd in category  Overall sd
## bio03   3.814430        31.475161    29.433655       1.924639   2.7677382
## bio12   2.768341       649.944444   591.792453      94.616461 108.6297778
## bio07   2.546293        30.727778    29.564151       1.370241   2.3632480
## bio17   2.045847       108.944444    98.396226       9.436762  26.6630545
## wind04 -4.857917         2.811111     3.233962       0.379896   0.4501335
##             p.value
## bio03  1.364978e-04
## bio12  5.634245e-03
## bio07  1.088738e-02
## bio17  4.077145e-02
## wind04 1.186269e-06
##
## $`4`
##          v.test Mean in category Overall mean sd in category Overall sd
## bio01  5.762214         16.64000     11.40008       1.286810   2.116418
## bio03  2.388444         32.27402     29.43365       2.825650   2.767738
## bio08 -2.003273         10.15667     15.07075       1.405884   5.709117
## bio07 -4.258509         25.24000     29.56415       2.136914   2.363248
## bio17 -5.481377         35.60000     98.39623      30.780513  26.663055
##            p.value
## bio01 8.301771e-09
## bio03 1.691987e-02
## bio08 4.514803e-02
## bio07 2.057948e-05
## bio17 4.220272e-08
```