```
## ******************************************************************************************** ##
# Poor geographical match between the distributions of host diversity and parasite discovery effort #

        ## Fátima Jorge and Robert Poulin
## ******************************************************************************************** ##


# Load packages
library(devtools)
library(letsR)
library(maptools)
library(sp)
library(rgdal)
library(colorRamps)


## Host
# Import the shapefile data
host <-  readShapePoly("XXX.shp", delete_null_obj = TRUE)
# option above does not work for AVES and Marine fish data: "Bailing out at geometry object ..."
# host <- readOGR("SHAPEFILENAME.shp") # use this for those cases


#crop polygon if outside the boundaries >> REPTILES, CHONDRICHTHYES, MARINEFISH_PART_1 and
MARINEFISH_PART_3 shapefiles:
#hostcrop <- crop(host, extent (-180,180,-90,90)) # crop shapefile
#plot(hostcrop)
#writeOGR(hostcrop, ".","SHAPEFILENEWNAME", driver="ESRI Shapefile") #save shapefile


# Transform host data polygons into a global PAM
# to keep track of the analysis relative running time set count = TRUE
pam_host <-  lets.presab (host, resol = 1, count = TRUE)  # global grid of 1° resolution
# pam_host <-  lets.presab (host, resol = 2, count = TRUE) # global grid of 2° resolution

# Call the object
pam_host

# Summary of the PAM
summary(pam_host)

# Plotting the geographic pattern of species richness
plot(pam_host, world = TRUE, axes = TRUE,  box = FALSE, col_rich = matlab.like2,  main = "HOSTGROUP
Species Richness- resol = 1")

# Calculate for each cell with presence data the relative number of species per cell in relation to
the total number of species in the grid
pam_host$Richness_Raster # see raster data information



  ## PARASITE ##
# Import the point data
parasite <- read.table("PARASITEDATASET.csv", sep=",", header=TRUE)
names(parasite)


#Plot
plot(parasite$Longitude, parasite$Latitude, cex=.5, col='red',  main = "HOSTGROUP Parasites
descriptions 1970-2017")

#Convert the data frame to a SpatialPointsDataFrame using the sp package
#library(sp)
#library(rgdal)
parasite.xy <- SpatialPoints(parasite[c("Longitude", "Latitude")], proj4string = CRS("+proj=longlat
+datum=WGS84"))
parasite.xy <- SpatialPointsDataFrame(parasite.xy, parasite)

Pspecies <- parasite$Species.name # create a vector with species name

# Transform parasite points into a global PAM
pam_par1<-lets.presab.points(parasite[c("Longitude", "Latitude")], species=Pspecies, resol = 1,
count=TRUE) # global grid of 1° resolution
#pam_par1<-lets.presab.points(parasite[c("Longitude", "Latitude")], species=Pspecies, resol = 2,
```

```r
count=TRUE)# global grid of 2° resolution

# confirm the data:
summary(pam_par1) # check for number of species

# Mapping species richness pattern
# Plotting the geographic pattern of species richness
plot(pam_par1, world = TRUE, axes = TRUE,  box = FALSE, col_rich = matlab.like2,  main = "HOSTGROUP
parasites species richness- resol = 1")

# Calculate for each cell with presence data the relative number of species per cell in relation to
the total number of species in the grid
pam_par1$Richness_Raster # see raster data information


                ## *** STATISTICAL ANALYSIS *** ##
## Wihout assuming causality: Is there a correlation - looking for similarity (or dissimilarity) in
patterns of spatial distribution between parasite and host richness:
# ignoring spatial autocorrelations
# PERASON'S CORRELATION COEFICIENT [assumes normality]:

# statistics do not work on raster files ("'x' must be a numeric vector"), need to extract values:
P<-getValues(pam_par1$Richness_Raster)
H<-getValues(pam_host$Richness_Raster)

#Plot data:
plot(P~H)

# Correlations
#assuming normality:
HPcorP <- cor.test(P, H, method = "pearson")
HPcorP

#if data is not normal => non parametric correlation: Spearman:
HPcorS <- cor.test(P, H, method = "spearman") # data is not normal so is more appropriate
HPcorS

# attention joint absence (double zeros) contributes to similarity > need to remove double zeros:
#option:
HP = data.frame (H,P, e=H+P)
HPno0<-subset(HP, e!=0)

#assuming normality:
HPcorP00 <- cor.test(HPno0$P, HPno0$H, method = "pearson")
HPcorP00

#if data is not normal => non parametric correlation: Spearman:
HPcorS00 <- cor.test(HPno0$P, HPno0$H, method = "spearman") # data is not normal so is more appropriate
HPcorS00

# To control for spatial non-independence, adjust sample size, rather than explicitly modelling
spatial non-independence
library(SpatialPack)

#need to extract coordinates from raster file:
#host
Hspts <- rasterToPoints(pam_host$Richness_Raster, spatial = TRUE)
Hllprj <-  "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs +towgs84=0,0,0"
Hllpts <- spTransform(Hspts, CRS(Hllprj))

Hspts@data <- data.frame(Hspts@data, long=coordinates(Hspts)[,1],
                         lat=coordinates(Hspts)[,2])
head(Hspts@data)

#parasite
Pspts <- rasterToPoints(pam_par1$Richness_Raster, spatial = TRUE)
Pllprj <-  "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs +towgs84=0,0,0"
Pllpts <- spTransform(Pspts, CRS(Pllprj))
Pspts@data <- data.frame(Pspts@data, long=coordinates(Pspts)[,1],
                         lat=coordinates(Pspts)[,2])
head(Pspts@data)

# modified T test:
coords <- Pspts@data[,2:3]
```

```
sp.Corr_13 <- summary(modified.ttest(H,P,coords,nclass = 13)) # number of classes: the reliability of
this correction is directly related to the estimated degree of spatial autocorrelation, and in turn
the estimated degree of spatial autocorrelation varies according to the number of distance classes.
Fewer distance classes result in larger distance class intervals and a lower value of spatial
autocorrelation, while more distance classes imply smaller distance class intervals with higher values
of spatial autocorrelation (Fortin, 1999) [from Fortin & Payette 2002]

plot(x=sp.Corr_13$coef[,1],y = sp.Corr_13$coef[,4],type = "l",ylab = "Moran's I -
Parasite",xlab="Upper bound of distance")
#compute for x variable:
plot(x=sp.Corr_13$coef[,1],y = sp.Corr_13$coef[,3],type = "l",ylab = "Moran's I - Host",xlab="Upper
bound of distance")
sp.Corr_13

sp.Corr_5 <- summary(modified.ttest(H,P,coords,nclass = 5))
sp.Corr_5
plot(x=sp.Corr_5$coef[,1],y = sp.Corr_5$coef[,4],type = "l",ylab = "Moran's I - Parasite",xlab="Upper
bound of distance")

sp.Corr_20 <- summary(modified.ttest(H,P,coords,nclass = 20))
sp.Corr_20
plot(x=sp.Corr_20$coef[,1],y = sp.Corr_20$coef[,4],type = "l",ylab = "Moran's I -
Parasite",xlab="Upper bound of distance")
#compute for x variable:
plot(x=sp.Corr_20$coef[,1],y = sp.Corr_20$coef[,3],type = "l",ylab = "Moran's I - Host",xlab="Upper
bound of distance")

#any difference in sgnificance?


#now with the dataset without double zeros:
HPb = data.frame (H,P, e=H+P, Pspts@data[,2], Pspts@data[,3])
HPbno0c<-subset(HPb, e!=0)
names(HPbno0c)

# modified T test:
coordsb = data.frame(HPbno0c$Pspts.data...2., HPbno0c$Pspts.data...3.)

sp.Corrb_13 <- summary(modified.ttest(HPbno0c$P, HPbno0c$H,coordsb,nclass = 13))
sp.Corrb_13
plot(x=sp.Corrb_13$coef[,1],y = sp.Corrb_13$coef[,4],type = "l",ylab = "Host Moran's I",xlab="Upper
bound of distance")
plot(x=sp.Corrb_13$coef[,1],y = sp.Corrb_13$coef[,3],type = "l",ylab = "Parasite Moran's
I",xlab="Upper bound of distance")


sp.Corrb_5 <- summary(modified.ttest(HPbno0c$P, HPbno0c$H,coordsb,nclass = 5))
sp.Corrb_5
plot(x=sp.Corrb_5$coef[,1],y = sp.Corrb_5$coef[,4],type = "l",ylab = "Moran's I - Host",xlab="Upper
bound of distance")

sp.Corrb_20 <- summary(modified.ttest(HPbno0c$P, HPbno0c$H,coordsb,nclass = 20))
sp.Corrb_20
plot(x=sp.Corrb_20$coef[,1],y = sp.Corrb_20$coef[,4],type = "l",ylab = "Host Moran's I",xlab="Upper
bound of distance")
plot(x=sp.Corrb_20$coef[,1],y = sp.Corrb_20$coef[,3],type = "l",ylab = "Parasite Moran's
I",xlab="Upper bound of distance")


## cor.spatial - Tjostheim’s Coefficient: is a variant of the correlation coefficient (cor) to be used
in a spatial statistics context.
# with double zeros
HP_cor.spatial <- cor.spatial(H, P, coords)
HP_cor.spatial
summary(HP_cor.spatial)

# without double zeros
HP_cor.spatialb <- cor.spatial(HPbno0c$P, HPbno0c$H,coordsb)
HP_cor.spatialb
summary(HP_cor.spatialb)


## codispersion
```

```
# useful method for exploring species(parasites)-environment(host) relationships in a spatially
explicit context
# Computes the codispersion coefficient between two spatial variables for a given number of classes
for the lag distance.
# with double zeros
z2<-codisp(H, P, coords, nclass = 13)
z2
plot(z2)

# without double zeros
z2b<-codisp(HPbno0c$P, HPbno0c$H, coordsb, nclass = 13)
z2b
plot(z2b)


## RELATIVE EFFORT ##
# Divide species richness raster by total number of species
# Parasite
pam_par1 # CHECK NUMBER OF SPECIES AND USE IT BELLOW!

Rel_pam_par1<-pam_par1$Richness_Raster / 111 # add total number of parasite species
Rel_pam_par1 # confim the atributes of the raster layer, should be equal to the previous pam_par1
$Richness_Raster
plot(Rel_pam_par1)


# Host
pam_host # CHECK NUMBER OF SPECIES AND USE IT BELLOW!

Rel_pam_host<-pam_host$Richness_Raster / 1111 # add total number of host species
Rel_pam_host # confim the atributes of the raster layer, should be equal to the previous pam_par1
$Richness_Raster
plot(Rel_pam_host)

# using overlay function:
HPrelEff<-overlay(Rel_pam_par1, Rel_pam_host, fun=function(r1, r2) {return(r1-r2)})
plot(HPrelEff)

#create a breackpoint for values so it is possible to colour 0 as white
#breakpoints <- c(-0.5,-0.04, -0.03, -0.2, -0.1, 0.0) # when no P > H
breakpoints <- c(-0.03,-0.02, -0.01, -0.025, 0.0, 0.01, 0.02) # when P > H
my.colors= colorRampPalette(c("darkblue","blue","lightblue","white","white","orangered", "red"))

plot(HPrelEff, col=my.colors(255))
map(interior=T,add=T)


########################
## HOST SHARING - HOST ##
########################

# Load packages
library(devtools)
library(letsR)
library(maptools)
library(sp)
library(rgdal)
library(colorRamps)

data <- read.table("Data2.csv", sep=",", header=TRUE)
names(data)
attach(data)

#to subset dataset in the different date range:
amp <- data[ which(HostTaxon == "amphibia"),]
rep <- data[ which(HostTaxon == "Reptilia"),]
bird <- data[ which(HostTaxon == "Aves"),]
mam <- data[ which(HostTaxon == "Mammalia"),]
fresF <- data[ which(HostTaxon == "Fresh_Pisces"),]
MarF <- data[ which(HostTaxon == "Mar_Pisces"),]

detach(data)

## AMPHIBIAN
```

```
# HOST:
#Convert the data frame to a SpatialPointsDataFrame using the sp package
Hamp.xy <- SpatialPoints(amp[c("Longitude", "Latitude")], proj4string = CRS("+proj=longlat
+datum=WGS84"))
Hamp.xy <- SpatialPointsDataFrame(Hamp.xy, amp)

Hspecies.a <- amp$HostSpecies # create a vector with species name

# Transform parasite points into a global PAM
pam_Hamp<-lets.presab.points(amp[c("Longitude", "Latitude")], species=Hspecies.a, resol = 2,
count=TRUE) # global grid of 2° resolution
# to keep track of the analysis relative running time set count = TRUE

# confirm the data:
summary(pam_Hamp) # check for number of species

# Plotting the geographic pattern of species richness
plot(pam_Hamp, world = TRUE, axes = TRUE,  box = FALSE, col_rich = matlab.like2,  main = "Amphibian
hosts from species description 1970-2017 resol = 2") #save plot 6x8 in

# Calculate for each cell with presence data the relative number of species per cell in relation to
the total number of species in the grid
pam_Hamp$Richness_Raster # see raster data information

# PARASITE:
#Convert the data frame to a SpatialPointsDataFrame using the sp package
amp.xy <- SpatialPoints(amp[c("Longitude", "Latitude")], proj4string = CRS("+proj=longlat
+datum=WGS84"))
amp.xy <- SpatialPointsDataFrame(amp.xy, amp)

Pspecies.a <- amp$Species # create a vector with species name

# Transform parasite points into a global PAM
pam_amp<-lets.presab.points(amp[c("Longitude", "Latitude")], species=Pspecies.a, resol = 2, count=TRUE)
# global grid of 2° resolution
# to keep track of the analysis relative running time set count = TRUE

# confirm the data:
summary(pam_amp) # check for number of species

# Plotting the geographic pattern of species richness
plot(pam_amp, world = TRUE, axes = TRUE,  box = FALSE, col_rich = matlab.like2,  main = "Amphibian
parasites species description 1970-2017  resol = 2") #save plot 6x8 in

# Calculate for each cell with presence data the relative number of species per cell in relation to
the total number of species in the grid
pam_amp$Richness_Raster # see raster data information

#[... all other data subsets]



## *** STATISTICAL ANALYSIS *** ##
# data visualization:
#library("ggpubr")

# statistics do not work on raster files ("'x' must be a numeric vector"), need to extract values:
PA<-getValues(pam_amp$Richness_Raster)
HA<-getValues(pam_Hamp$Richness_Raster)

#[... all other data subsets]

#need to extract coordinates from raster file (doesn't matter which one):
Pspts <- rasterToPoints(pam_amp$Richness_Raster, spatial = TRUE)
Pllprj <-  "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs +towgs84=0,0,0"
Pllpts <- spTransform(Pspts, CRS(Pllprj))
Pspts@data <- data.frame(Pspts@data, long=coordinates(Pspts)[,1],
                         lat=coordinates(Pspts)[,2])
head(Pspts@data)

# attention joint absence (double zeros) contributes to similarity > need to remove double zeros:
#option:
HP_a = data.frame (HA,PA, e=HA+PA, Pspts@data[,2], Pspts@data[,3])
HPno0_a<-subset(HP_a, e!=0)
```

```
#[... all other data subsets]

# check normality:
# # Shapiro-Wilk normality test
shapiro.test(HPno0_a$HA) # if significant = data distribution is significantly different from normal
distribution
shapiro.test(HPno0_a$PA)
#[... all other data subsets]


# check data distribution:
# ------------------------

# do we have overdispersion? i.e. variance greater than mean?
ma=mean(HPno0_a$PA)
ma
siga=sd(HPno0_a$PA)
siga

#[... all other data subsets]


# 3.0 Spatial generalised linear mixed models (GLMM):
# -------------------------------------------------------
# following Dorman_etal2007_SAC-Script
# Spatial GLMM are generalised linear models (GLMs) in which the linear predictor may contain random
effects and within-group errors may be spatially autocorrelated
# However, the modelled correlation structure is not directly used when making predictions.
# this spatial models usual managed to decrease spatial autocorrelation in the residuals, but may not
be able to completely eliminate it.
# In the absence of a perfect model, however, doing something is better than doing nothing (Keitt et
al. 2002).

# Shperical distance decay function to generate the spatial error
library(MASS)
library(ncf)
library(rsq)
library(r2glmm)
library(nlme)

#
# without 00

# AMPHIBIAN
#--------------
group.a <- factor(rep("a",nrow(HPno0_a)))
dat.a <- cbind(HPno0_a, group.a)
# attach(dat.a2) #For some reason, the data have to be attached AND specified in the formula!
# GLMM fits with spherical correlation structure structure

#Poisson
model.p.a <- glmmPQL(PA ~ HA, random=~1|group.a,
                      data=dat.a,
                      correlation=corSpher(form=~(Pspts.data...2.)+(Pspts.data...3.)),
                      family = poisson) # if overdisoersion is detected in data fit: family =
quasipoisson())
summary(model.p.a)

# check heteroscedasticity of the best model:
plot(model.p.a)
# If there is absolutely no heteroscedastity, you should see a completely random, equal distribution
of points throughout the range of X axis and a flat red line.

# and plot variogram of the residuals
# We want the variogram to be flat when we include a spatial random effect with the correct structure.
plot(Variogram(model.p.a,resType="normalized"))

# plot Moran's I
#qp
correlog1.a.glmm.p <- correlog(dat.a$Pspts.data...2., dat.a$Pspts.data...3., residuals(model.p.a),
                                 na.rm=T, increment=1, resamp=0)
# now plot only the first 100 distance classes:
par(mar=c(5,5,0.1, 0.1))
```

```
plot(correlog1.a.glmm.p$correlation[1:100], type="b", pch=16, cex=1.5, lwd=1.5,
     xlab="distance", ylab="Moran's I", cex.lab=2, cex.axis=1.5); abline(h=0)

# [... GLMM for other vertebrate groups]



##########################
# HOST-SHARING: PARASITES
##########################

#to subset dataset in the different date range:
aca <- data[ which(ParTaxon == "Acanthocephala"),]
ces <- data[ which(ParTaxon == "Cestode"),]
nem <- data[ which(ParTaxon == "Nematoda"),]
dig <- data[ which(ParTaxon == "Digenea"),]

# [... script as described above for host sharing - host]

######################
# CUMULATIVE ANALYSIS
######################

## By host group:
# with vectors without absence data (zeros)
#remove zeros
P_a<-subset(PA, PA!=0)
P_r<-subset(PR, PR!=0)
P_b<-subset(PB, PB!=0)
P_m<-subset(PM, PM!=0)
P_ff<-subset(PFF, PFF!=0)
P_mf<-subset(PMF, PMF!=0)

# Cumulative analysis:
permute <- 999

# Marine fish:
lst_mf = matrix(NA, permute + 1, length(P_mf))
lst_mf[1,] <- cumsum(P_mf)

# loop
for (i in 1:permute) {

  lst_mf[i+1,] <- cumsum(sample(P_mf))
}
CI_mf <- apply(lst_mf, 2, quantile, probs = c(0.025, 0.975))

# plot
x_mf <- 1:ncol(lst_mf)
plot(x_mf, xlim=c(0,525), ylim=c(0,1750), main="Cumulative vertebrate parasite species discovery from
1970-2017", xlab="number of sampled grid cells", ylab= "Number of species", lst_mf[1,], type='l',
col="dark blue")
#lines(x_mf, CI_mf[1,], lty=2, col="dark blue")
#lines(x_mf, CI_mf[2,], lty=2, col="dark blue")

#or as polygon
polygon(x_mf, CI_mf[1,], lty=2, col="dark blue")
polygon(x_mf, CI_mf[2,], lty=2, col="dark blue")

# [repeat loop for all other vectors, i.e. P_m, P_ff, P_b, P_r, P_a, adding the polygon or lines to
the marine fish plot]


## by parasite group:
#remove zeros
P_ac<-subset(PAc, PAc!=0)
P_c<-subset(PC, PC!=0)
P_n<-subset(PN, PN!=0)
P_d<-subset(PD, PD!=0)

# [... script as described above for each vertebrate host]

## end
```