Supplementary information for “Climate change drives spatial mismatch and threatens the biotic interactions of the Brazil nut tree”

Sales, Rodrigues & Masiero

9/20/2020

## Data collection and cleaning

Occurrence data for the Brazil nut tree *Bertholletia excelsa* and its community of pollinators and seed dispersers were obtained from virtual databases, complemented with data from the CNCFlora (not shown here) and checked for errors, duplicates and inconsistencies.

library(spocc)
library(rvertnet)
library(plyr)

#-------- OCCURRENCES
sp\_list <- c("Bertholletia excelsa") # Brazil nut tree

# Get data
bi1 <- occ(query = sp\_list, from = c("gbif", "inat", "ecoengine"),
 has\_coords = T)

# Fixnames
bi1.fix <- fixnames(bi1, how = "query")

bi2 <- occ2df(bi1.fix) #convert to dataframe

# Remove NA
bi3 <- bi2[-which(is.na(bi2)), ]

# Convert la/long to numeric
bi3$longitude <- as.numeric(bi3$longitude)
bi3$latitude <- as.numeric(bi3$latitude)

# Remove impossible, incomplete and unlikely coordinates
dp <- bi3 %>%
 coord\_impossible() %>%
 coord\_incomplete() %>%
 coord\_unlikely()

i=1

# Thinning data
# The objective is to identify the maximum number of points that are at least a minimum distance apart.
# The `spThin` function can be used to thin data. This function can use several optimisation routines to acheive this.
tmp <- NULL
thin1 <- NULL

for (i in 1:length(sp\_list)){

 thin1 <- thin(
 dp[dp$name==sp\_list[i], ],
 verbose=F,
 long.col = "longitude",
 lat.col = "latitude",
 spec.col = "name",
 thin.par = 1.6, # points have at least a minimum distance of 1.6 km from each other
 reps = 3,
 locs.thinned.list.return = TRUE,
 write.files = F)

 thin1 <- as.data.frame(thin1)
 thin1$sp <- rep(sp\_list[i], nrow(thin1))
 tmp <- rbind(tmp, thin1)

}

## Ecological niche modelling

Here, we will model the ecological niche of the Brazil nut tree. To do so, we apply an ensemble procedure on models fitted to climate data and the occurrence records collected previously.

library(sdm)
library(raster)
library(usdm)
library(rgdal)
library(rgeos)

# --------- CLIMATE PRESENT
# IPCC6
setwd("D:/Leddiv/Climate\_IPCC6/wc2.1\_10m\_bio")
bio <- stack(list.files(pattern = ".tif"))
plot(bio$bio\_4)

# ------- CLIMATE FUTURE
setwd("D:/Leddiv/Climate\_IPCC6")
dir()

#SSP 245 ------------------------
biof <- list.files(pattern = ".tif")
t <- grep("\_ssp245\_", biof)
biof <- biof[t]

biof1.1 <- stack(biof[[1]]) # BCC-CSM2-MR 2030
biof1.2 <- stack(biof[[2]]) # BCC-CSM2-MR 2050
biof1.3 <- stack(biof[[3]]) # BCC-CSM2-MR 2070
biof1.4 <- stack(biof[[4]]) # BCC-CSM2-MR 2090

biof2.1 <- stack(biof[[5]]) # CanESM5 2030
biof2.2 <- stack(biof[[6]]) # CanESM5 2050
biof2.3 <- stack(biof[[7]]) # CanESM5 2070
biof2.4 <- stack(biof[[8]]) # CanESM5 2090

biof3.1 <- stack(biof[[9]]) # MIROC6 2030
biof3.2 <- stack(biof[[10]]) # MIROC6 2050
biof3.3 <- stack(biof[[11]]) # MIROC6 2070
biof3.4 <- stack(biof[[12]]) # MIROC6 2090

#SSP 370 ------------------------
biof <- list.files(pattern = ".tif")
t <- grep("\_ssp370\_", biof)
biof <- biof[t]

biof4.1 <- stack(biof[[1]]) # BCC-CSM2-MR 2030
biof4.2 <- stack(biof[[2]]) # BCC-CSM2-MR 2050
biof4.3 <- stack(biof[[3]]) # BCC-CSM2-MR 2070
biof4.4 <- stack(biof[[4]]) # BCC-CSM2-MR 2090

biof5.1 <- stack(biof[[5]]) # CanESM5 2030
biof5.2 <- stack(biof[[6]]) # CanESM5 2050
biof5.3 <- stack(biof[[7]]) # CanESM5 2070
biof5.4 <- stack(biof[[8]]) # CanESM5 2090

biof6.1 <- stack(biof[[9]]) # MIROC6 2030
biof6.2 <- stack(biof[[10]]) # MIROC6 2050
biof6.3 <- stack(biof[[11]]) # MIROC6 2070
biof6.4 <- stack(biof[[12]]) # MIROC6 2090

# Check for name consistency

nome\_vars <- c("bio\_1", "bio\_2", "bio\_3", "bio\_4", "bio\_5", "bio\_6",
 "bio\_7", "bio\_8", "bio\_9", "bio\_10", "bio\_11", "bio\_12",
 "bio\_13", "bio\_14", "bio\_15", "bio\_16", "bio\_17",
 "bio\_18", "bio\_19" )

names(biof1.1) <- nome\_vars
names(biof1.2) <- nome\_vars
names(biof1.3) <- nome\_vars
names(biof1.4) <- nome\_vars

names(biof2.1) <- nome\_vars
names(biof2.2) <- nome\_vars
names(biof2.3) <- nome\_vars
names(biof2.4) <- nome\_vars

names(biof3.1) <- nome\_vars
names(biof3.2) <- nome\_vars
names(biof3.3) <- nome\_vars
names(biof3.4) <- nome\_vars

names(biof4.1) <- nome\_vars
names(biof4.2) <- nome\_vars
names(biof4.3) <- nome\_vars
names(biof4.4) <- nome\_vars

names(biof5.1) <- nome\_vars
names(biof5.2) <- nome\_vars
names(biof5.3) <- nome\_vars
names(biof5.4) <- nome\_vars

names(biof6.1) <- nome\_vars
names(biof6.2) <- nome\_vars
names(biof6.3) <- nome\_vars
names(biof6.4) <- nome\_vars

#------- SOIL
library(ncdf4)
setwd("D:/Leddiv/Soil/HWSD\_1247/data")
s1 <- list.files(pattern = ".nc4")
s1 <- s1[c(3, 7, 19, 20, 21, 22, 23, 24, 25, 27, 28)] # Topsoil features - Amazon soils are poor, so topsoil mostly influence seedling establishment
soil <- stack(s1)
names(soil)

soil <- resample(soil, bio[[1]], method = "ngb")

bio <- stack(soil, bio)

biof1.1 <- stack(biof1.1, soil)
biof1.2 <- stack(biof1.2, soil)
biof1.3 <- stack(biof1.3, soil)
biof1.4 <- stack(biof1.4, soil)

biof2.1 <- stack(biof2.1, soil)
biof2.2 <- stack(biof2.2, soil)
biof2.3 <- stack(biof2.3, soil)
biof2.4 <- stack(biof2.4, soil)

biof3.1 <- stack(biof3.1, soil)
biof3.2 <- stack(biof3.2, soil)
biof3.3 <- stack(biof3.3, soil)
biof3.4 <- stack(biof3.4, soil)

biof4.1 <- stack(biof4.1, soil)
biof4.2 <- stack(biof4.2, soil)
biof4.3 <- stack(biof4.3, soil)
biof4.4 <- stack(biof4.4, soil)

biof5.1 <- stack(biof5.1, soil)
biof5.2 <- stack(biof5.2, soil)
biof5.3 <- stack(biof5.3, soil)
biof5.4 <- stack(biof5.4, soil)

biof6.1 <- stack(biof6.1, soil)
biof6.2 <- stack(biof6.2, soil)
biof6.3 <- stack(biof6.3, soil)
biof6.4 <- stack(biof6.4, soil)

# ------ SDM
resu <- matrix(nrow = length(sp\_list), ncol = 15)
colnames(resu) <- c("sp\_name","records","AUC","COR","Deviance","TSS","iniDist" ,
 "finalDist1","finalDist2","finalDist3","finalDist4","finalDist5",
 "finalDist6","finalDist7", "finalDist8")

vars <- NULL

t=1

#for (i in 1:length(sp\_list)) {

# Create a column of species
t <- as.character(sp\_list[i])
sp <- dp[dp$sp==t, c('Longitude','Latitude')]
sp$species <- 1

# Transform to spatial data
coordinates(sp) <- ~ Longitude + Latitude

# Create "response variable"
sp$species <- 1

# Remove occurrences outside PanAmazonia
shape <- readOGR("D:/Leddiv/shapefiles/PanAmaz/pan\_amazonia\_draft.shp")
buf <- raster::buffer(shape, 0.5) # Include a buffer around the shape to allow for border uncertainty
sp@proj4string <- buf@proj4string #Transform to same projection

sp <- sp[!is.na(over(sp, buf)), ] #Select only occurrences within PanAmazonia
plot(sp)

write.csv(data.frame(sp), "D:/Leddiv/lilian/tmp2/Ocorr\_castanheira.csv")

# Remove collinear variables
# Climate
spx <- extract(bio, sp, na.omit=T) # extract from file
spx <- data.frame(spx) #convert to dataframe

v <- vifcor(spx, th = 0.6) # check collinearity (variance inflation and correlation)

write.csv(data.frame(v@results), "D:/Leddiv/lilian/tmp2/VIF\_castanheira.csv")

bio\_i <- exclude(bio, v) # exclude collinear predictors
names(bio\_i)

# BBox
bb <- bbox(sp)
extent(bb)
bb.buf <- extent(bb[1]-35, bb[3]+15, bb[2]-30, bb[4]+20)
bio\_i <- crop(bio\_i, bb.buf) # species-specific backgroud

#Remove Andes
andes <- readOGR("D:/Leddiv/lilian/Bats/Antonelli-et-al\_PNAS/Zenodo\_scripts\_and\_data\_Antonelli-et-al\_PNAS/input/shapes/neotropics\_areas.shp")
andes <- andes[andes$ECO\_NAME=="Andes", ]
plot(andes)
buf.an <- raster::buffer(andes, -0.8)
plot(buf.an, add=T)

bio\_ii <- mask(bio\_i, buf.an, inverse=T)

plot(bio\_ii$bio\_4)

# Define directory to save tmp files
setwd("D:/Leddiv/lilian/tmp2/Mapas")

# generate sdmData
d <- sdmData(species~., train=sp, predictors= bio\_ii, bg=list(n=10000))
d

# generate sdm model
m <- sdm(species ~ . , d, methods=c("brt", "maxlike"),
 replication='sub',test.percent=30, n=30,
 parallelSettings = list(ncore=12, method='parallel'))

m

#save.image("D:/Leddiv/lilian/tmp2/Workspace\_frugs.RData")

# Get variables importance
z <- getVarImp(m)
svg (paste0("D:/Leddiv/lilian/tmp2/Resu/", # Your directory
 gsub(" ", "\_", t), "\_varimp.svg"))
plot(z)
dev.off()

# Get the relationship with each predictor
m1 <- rcurve(m)
svg (paste0("D:/Leddiv/lilian/tmp2/Resu/", # Your directory
 gsub(" ", "\_", t), "\_predcurve.svg"))
plot(m1)
dev.off()

# Ensembling Present
en <- ensemble(m, bio\_i,
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))

en <- sdmvspecies::rescale(en)

plot(en, main = t)
writeRaster(en, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_pres.tif"), format = "GTiff", overwrite=TRUE)

# Evaluation
e <- getEvaluation(m)

# Save which variables were used and evaluation results for all species
resu[i, "sp\_name"] <- gsub(" ", "\_", t)
resu[i, "records"] <- nrow(as.data.frame(d))-10000
resu[i, "AUC"] <- round(mean(e$AUC), 2)
resu[i, "COR"] <- round(mean(e$COR), 2)
resu[i, "Deviance"] <- round(mean(e$Deviance),2)
resu[i, "TSS"] <- round(mean(e$TSS),2)

vars <- c(t, names(bio\_i), vars)

# Find binarization threshold
df <- data.frame(as.data.frame(d),coordinates(d)) # presence points and predictors associated
pr <- extract(en, df[,c('Longitude','Latitude')])

ev <- evaluates(df$species, pr) # evaluate prediction (observed vs expected)
th <- ev@threshold\_based$threshold[[2]] # threshold that maximizes sensitiv + specificity

# Binary prediction
pa <- en
pa[] <- ifelse(pa[] >= th, 1, 0) # convert from continuous to binary

plot(pa, main = t)

# present PA
writeRaster(pa, format = "GTiff",
 paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_pres\_PA.tif"),
 overwrite = T)

resu[i, "iniDist"] <- length(pa[pa==1])

# Ensembling future
# SSP 245 --------------------------------------------------------------
# 2030

enf1.1 <- ensemble(m, crop(subset((biof1.1), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf2.1 <- ensemble(m, crop(subset((biof2.1), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf3.1 <- ensemble(m, crop(subset((biof3.1), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))

fut <- mean(enf1.1, enf2.1, enf3.1)
fut <- sdmvspecies::rescale(fut)
plot(fut, main = t)
writeRaster(fut, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_Mit\_2030.tif"), format = "GTiff", overwrite=TRUE)

# Binary prediction
pa <- fut
pa[] <- ifelse(pa[] >= th, 1, 0) # convert from continuous to binary
plot(pa)
writeRaster(pa, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_Mit\_2030\_PA.tif"), format = "GTiff", overwrite=TRUE)

resu[i, "finalDist1"] <- length(pa[pa==1])

# 2050
enf1.2 <- ensemble(m, crop(subset((biof1.2), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf2.2 <- ensemble(m, crop(subset((biof2.2), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf3.2 <- ensemble(m, crop(subset((biof3.2), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))

fut <- mean(enf1.2, enf2.2, enf3.2)
fut <- sdmvspecies::rescale(fut)
writeRaster(fut, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_Mit\_2050.tif"), format = "GTiff", overwrite=TRUE)
plot(fut, main = t)

# Binary prediction
pa <- fut
pa[] <- ifelse(pa[] >= th, 1, 0) # convert from continuous to binary
plot(pa, main = t)

writeRaster(pa, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_Mit\_2050\_PA.tif"), format = "GTiff", overwrite=TRUE)

resu[i, "finalDist2"] <- length(pa[pa==1])

# 2070
enf1.3 <- ensemble(m, crop(subset((biof1.3), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf2.3 <- ensemble(m, crop(subset((biof2.3), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf3.3 <- ensemble(m, crop(subset((biof3.3), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))

fut <- mean(enf1.3, enf2.3, enf3.3)
fut <- sdmvspecies::rescale(fut)
plot(fut, main = t)
writeRaster(fut, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_Mit\_2070.tif"), format = "GTiff", overwrite=TRUE)

# Binary prediction
pa <- fut
pa[] <- ifelse(pa[] >= th, 1, 0) # convert from continuous to binary
writeRaster(pa, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_Mit\_2070\_PA.tif"), format = "GTiff", overwrite=TRUE)
plot(pa)
resu[i, "finalDist3"] <- length(pa[pa==1])

# 2090
enf1.4 <- ensemble(m, crop(subset((biof1.4), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf2.4 <- ensemble(m, crop(subset((biof2.4), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf3.4 <- ensemble(m, crop(subset((biof3.4), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))

fut <- mean(enf1.4, enf2.4, enf3.4)
fut <- sdmvspecies::rescale(fut)
plot(fut, main = t)

writeRaster(fut, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_Mit\_2090.tif"), format = "GTiff", overwrite=TRUE)

# Binary prediction
pa <- fut
pa[] <- ifelse(pa[] >= th, 1, 0) # convert from continuous to binary
plot(pa)
writeRaster(pa, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_Mit\_2090\_PA.tif"), format = "GTiff", overwrite=TRUE)

resu[i, "finalDist4"] <- length(pa[pa==1])

#SSP 370 ------------------------
# 2030

enf4.1 <- ensemble(m, crop(subset((biof4.1), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf5.1 <- ensemble(m, crop(subset((biof5.1), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf6.1 <- ensemble(m, crop(subset((biof6.1), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))

fut <- mean(enf4.1, enf5.1, enf6.1)
fut <- sdmvspecies::rescale(fut)
plot(fut, main = t)
writeRaster(fut, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_BAU\_2030.tif"), format = "GTiff", overwrite=TRUE)

# Binary prediction
pa <- fut
pa[] <- ifelse(pa[] >= th, 1, 0) # convert from continuous to binary

plot(pa, main = t)

writeRaster(pa, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_BAU\_2030\_PA.tif"), format = "GTiff", overwrite=TRUE)

resu[i, "finalDist5"] <- length(pa[pa==1])

# 2050
enf4.2 <- ensemble(m, crop(subset((biof4.2), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf5.2 <- ensemble(m, crop(subset((biof5.2), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf6.2 <- ensemble(m, crop(subset((biof6.2), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))

fut <- mean(enf4.2, enf5.2, enf6.2)
fut <- sdmvspecies::rescale(fut)
plot(fut, main = t)
writeRaster(fut, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_BAU\_2050.tif"), format = "GTiff", overwrite=TRUE)

# Binary prediction
pa <- fut
pa[] <- ifelse(pa[] >= th, 1, 0) # convert from continuous to binary
writeRaster(pa, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_BAU\_2050\_PA.tif"), format = "GTiff", overwrite=TRUE)

resu[i, "finalDist6"] <- length(pa[pa==1])
plot(pa, main = t)

# 2070
enf4.3 <- ensemble(m, crop(subset((biof4.3), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf5.3 <- ensemble(m, crop(subset((biof5.3), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf6.3 <- ensemble(m, crop(subset((biof6.3), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))

fut <- mean(enf4.3, enf5.3, enf6.3)
fut <- sdmvspecies::rescale(fut)
plot(fut, main = t)

writeRaster(fut, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_BAU\_2070.tif"), format = "GTiff", overwrite=TRUE)

# Binary prediction
pa <- fut
pa[] <- ifelse(pa[] >= th, 1, 0) # convert from continuous to binary
writeRaster(pa, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_BAU\_2070\_PA.tif"), format = "GTiff", overwrite=TRUE)

plot(pa, main = t)

resu[i, "finalDist7"] <- length(pa[pa==1])

# 2090
enf4.4 <- ensemble(m, crop(subset((biof4.4), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf5.4 <- ensemble(m, crop(subset((biof5.4), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))
enf6.4 <- ensemble(m, crop(subset((biof6.4), names(bio\_i)), bb.buf),
 setting=list(method='weighted',stat='TSS'),
 parallelSettings = list(ncore=12, method='parallel'))

fut <- mean(enf4.4, enf5.4, enf6.4)
fut <- sdmvspecies::rescale(fut)
writeRaster(fut, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_BAU\_2090.tif"), format = "GTiff", overwrite=TRUE)

# Binary prediction
pa <- fut
pa[] <- ifelse(pa[] >= th, 1, 0) # convert from continuous to binary
writeRaster(pa, paste0('D:/Leddiv/lilian/tmp2/Mapas/',
 gsub(" ", "\_", t),"\_BAU\_2090\_PA.tif"), format = "GTiff", overwrite=TRUE)

resu[i, "finalDist8"] <- length(pa[pa==1])

#}

write.csv(resu, "D:/Leddiv/lilian/tmp2/Resu/resu\_Castanheira.csv")

write.csv(vars, 'D:/Leddiv/lilian/tmp2/Resu/vars\_Castanheira.csv')

## Forest cover thresholds

Then, we simulate defaunation based on minimum habitat size. First, we will create a raster grid of 10km² with percent forest cover.

# raster of tree cover in the future
library(raster)

tree <- raster("C:/Users/lilia/OneDrive/Pos-doc/PNPD/Cap1/UFO/Defor/tree\_South\_fut.tif")

tree[tree==2] <- 0 # keep only cells of "trees"
plot(tree)



ref <- aggregate(tree, fact=10, fun=sum) # set resolution of a "landscape" (10km²) and sum the number of tree cover cells
plot(ref)



Now, we will apply the 30% threshold of forest cover. We consider that suitable climate regions with less than 30% forest cover are considered unable to support viable populations of the Brazil nut tree.

#### Habitat thresholds Brazilian nut tree
# In this script, I will remove from the areas predicted to be deforested from the tree potential distribution

library(raster)
library(rgeos)
library(rgdal)

# Selecting the file with tree persistence probability in the future (Hewson, 2019)
desm <- raster("D:/Leddiv/Land-use/Hewson/Global\_Prediction\_2029.tif") # raster of tree cover in the future
Neo <- readOGR("D:/Leddiv/shapefiles/Neotropics/neotropica.shp")

tree <- crop(desm, Neo) # Cropping for Neo just to make things faster
tree[tree==2] <- 0 # keep only cells of "trees"
plot(tree)

ref <- raster("D:/Leddiv/lilian/tmp2/Mapas/Bertholletia\_excelsa\_pres\_PA.tif") #Brazilian nut tree - reference raster

tree1 <- resample(tree, ref) # Set same extent and resolution, with default method "bilinear"
plot(tree1)

tree2 <- tree1

tree2[tree1 < 0.3 ] <- 0 # Remove cells with >30% remaining tree cover in the future
tree2[tree2 > 0] <- 1
plot(tree2)

# Clipping Bertholletia distribution
setwd("D:/Leddiv/lilian/tmp2/Mapas")
dir()

b <- list.files(pattern = "Bertholletia\_excelsa")
bb <- grep(b, pattern = "\_PA.tif")

b <- b[bb]

ref1 <- raster(extent(ref), res = res(ref)/2)
i=1
for (i in 1: (length(bb)-1)) { # Set present as it is - not remove

 a <- raster(b[i])
 plot(a)
 d <- sum(a, tree2)
 d[d < 2] <- 0
 d[d > 0] <- 1
 d <- resample(d, ref1, method = "ngb")
 plot(d)
 writeRaster(d, paste0("D:/Leddiv/lilian/tmp2/Mapas2/",names(a), "\_filtered.tif"),
 format = "GTiff", overwrite = T)

 }

# Present
plot(raster(b[9]))
d <- resample(raster(b[9]), ref1, method = "ngb")
writeRaster(d, paste0("D:/Leddiv/lilian/tmp2/Mapas2/",(b[9]), "\_filtered.tif"),
 format = "GTiff", overwrite = T)

## Dispersal-restricted potential distribution of animals

In addition to climate suitability and forest cover thresholds, we also simulate occupancy of potential suitable future areas under dispersal constraints posed by landscape fragmentation. The absence of trees was considered a strong barrier to dispersal

library(raster)
library(rgdal)
library(MigClim)
library(sp)

library(MigClim)
#library(reshape2)

#-------------------------------------------------------------------------------------
# Barrier - <30% forest cover

desm <- raster("D:/Leddiv/Land-use/Hewson/Global\_Prediction\_2029.tif") # raster of tree cover in the future
Neo <- readOGR("D:/Leddiv/shapefiles/Neotropics/neotropica.shp")

tree <- crop(desm, Neo) # Cropping for Neo just to make things faster
tree[tree==2] <- 0 # keep only cells of "trees"
plot(tree)

tree1 <- aggregate(tree, 20)
plot(tree1)

barrier <- tree1

barrier[barrier >= 0.3 ] <- 0 # Remove cells with >30% remaining tree cover in the future
barrier[barrier > 0 ] <- 1 # Remove cells with >30% remaining tree cover in the future
plot(barrier)

#---------------------------------------------------------------------------------
# Disperseres
# Dasyprocta = 2.33 km/yr

sp.disp <- c("Dasyprocta\_leporina",
 "Dasyprocta\_fuliginosa",
 "Dasyprocta\_ruatanica",
 "Dasyprocta\_azarae",
 "Dasyprocta\_punctata",
 "Dasyprocta\_variegata")

tabela.resultados <- NULL #Tabela para guardar os resultados

i=1

for (i in 1: length(sp.disp)) { #

 #Kernel
 print(sp.disp[i])
 k <- round((2.33\*20)/18) # disp.dispersão em 20 anos, em nº de células (0.16º = 18km²)
 ke <- seq(from=1, to=0.05, length.out= (k+1))

 setwd("D:/Leddiv/lilian/tmp2/Mapas3")

 dir()
 # Initial distribution - iniDist
 # Final distribution - hsMap – binary file

 aa <- paste0(sp.disp[i],"\_pres\_hsmap\_clipIUCN.tif")

 tmp <- raster(aa)
 tmp[tmp>1] <- 1
 plot(tmp)
 iniDist <- tmp
 iniDist[is.na(iniDist)] <- 0
 plot(iniDist)
 iniDist <- as.data.frame(iniDist, xy=T)

 #head(iniDist)
 #nrow(iniDist)

 # Barrier
 l1 <- resample(barrier, tmp, method = "ngb")
 plot(l1)

 barr <- as.data.frame(l1)
 barr[is.na(barr)] <- 0
 #head(barr)
 #nrow(barr)

 # --- hsMaps
 maps <- list.files(pattern = paste0(sp.disp[i]))
 a <- maps[grep("hsmap", maps)]

 #### Mit
 b <- grep("Mit", a)
 hsMap <- stack(maps[b])
 hsMap[is.na(hsMap)] <- 0
 hsMap <- as.data.frame(hsMap)

 hsMap <- round(hsMap)
 #nrow(hsMap)
 #head(hsMap)

 ### Dispersal

 MigClim.migrate(iniDist = iniDist,
 hsMap = hsMap,
 rcThreshold = 1,
 envChgSteps = ncol(hsMap), dispSteps = 4,
 dispKernel= ke, overWrite = T,

 barrier = T, barrierType = “strong”, replicateNb=3,
 testMode=FALSE,
 lddFreq=0.05, lddMinDist=length(ke)+1, lddMaxDist=nrow(hsMap),

 fullOutput=T, keepTempFiles=FALSE)

 setwd("D:/Leddiv/lilian/tmp2/Mapas3/MigClimTest")
 #dir()

 #Saving results in a single spreadsheet

 resultados <- read.table("MigClimTest\_summary.txt", h=T, stringsAsFactors = FALSE, row.names = NULL)
 resultados <- data.frame(as.list(ceiling(colMeans(resultados[,2:ncol(resultados)]))))
 resultados$simulName <- paste0(sp.disp[i], "\_Mit")

 tabela.resultados <- rbind (tabela.resultados, resultados)

 distr.sp <- stack("MigClimTest1\_raster.asc",
 "MigClimTest2\_raster.asc",
 "MigClimTest3\_raster.asc")

 distr.sp <- mean(distr.sp)

 m <- c(2, 29999, 2, 29999, 30000, 3)
 rclmat <- matrix(m, ncol=3, byrow=TRUE)
 rc <- reclassify(distr.sp, rclmat)

 s <- calc(rc, fun=function(x){ x[x < 0] <- 4
 return(x)} )

 plot(s)

 setwd("D:/Leddiv/lilian/tmp2/Mapas\_finais")
 writeRaster(s, paste0(sp.disp[i],"\_Mit",".tif"), format="GTiff", overwrite=T)

}

write.csv(tabela.resultados, "D:/Leddiv/lilian/tmp2/Resu/Tabela.resultados\_dispersores.csv")

Note that the final map has 4 categories 0 : always unsuitable 1 : climate refugia - always suitable 2 : potential colonization (became suitable and is accessible) 3 : dispersal limitation (became suitable but is unaccessible) 4 : non-analog climate (became unsuitable)

## OGR data source with driver: ESRI Shapefile
## Source: "C:\Users\lilia\OneDrive\Doutorado\Shapefiles\PanAmaz\pan\_amazonia\_draft.shp", layer: "pan\_amazonia\_draft"
## with 1 features
## It has 1 fields

