The purpose of species distribution modeling is to build a map of the predicted species distribution.

Section1

Extracting species data from a wide variety of databases and the different ways that we can process some of the most important and environmental parameters to protect their data.

Species’ data

Environmental parameters

*where to obtain r*

*##Climate data*

*http://www.diva-gis.org/climate*

[*http://www.worldclim.org/*](http://www.worldclim.org/)

*## Elevation data*

*http://www.cgiar-csi.org/data/srtm-90m-digital-elevation-database-v4-1#download*

[*http://srtm.csi.cgiar.org/SELECTION/inputCoord.asp*](http://srtm.csi.cgiar.org/SELECTION/inputCoord.asp)

*## Land use data*

*http://due.esrin.esa.int/page\_globcover.php*

*## Population raster*

*http://sedac.ciesin.columbia.edu/data/sets/browse?facets=theme:population&facets=data-type:raster*

1. Accessing and cleaning GBIF Data

New excel ---Data—import form csv

1. Accessing GBIF Data via R
2. Access climate&other via R

##### get raster map

library(raster)

my0 = getData('GADM', country='MYS', level=0) #country outline

my1 = getData('GADM', country='MYS', level=1) #states included

par(mfrow=c(1,2))

plot(my0, main="Adm. Boundaries Malaysia Level 0")

plot(my1, main="Adm. Boundaries Malaysia Level 1")

## world climate

###Bioclimatic variables bio1, bio2 different mean

climate = getData('worldclim', var='bio', res=2.5) #resolution 2.5

plot(climate$bio1, main="Annual Mean Temperature")

plot(climate$bio5, main="Maximum Temperature")

1. Working with elevation data

setwd([*http://srtm.csi.cgiar.org/SELECTION/inputCoord.asp*](http://srtm.csi.cgiar.org/SELECTION/inputCoord.asp))

library(raster)

library(rgdal)

el1=raster("srtm\_59\_12.tif")

plot(el1)

el2=raster("srtm\_60\_11.tif")

plot(el2)

par(mfrow=c(1,2))

plot(el1)

plot(el2)

# combine them together in one raster if they are adjecent

mosee = mosaic(el1, el2, fun=mean)

plot(mosee)

writeRaster(mosee,"join\_59\_60.tif")

1. Deriving Topographic Products from Elevation Data

###### derive topographic products-- slope, aspect & hillshade

### slope

mosee #lat long

## advisable to convert lat-long rasters to planar units such as UTM

## in meters

#utm projection for *north borneo*

ref = "+proj=utm +zone=50 +ellps=GRS80 +datum=NAD83 +units=m +no\_defs"

library(raster)

projected\_raster = projectRaster(mosee, crs = ref)

projected\_raster

slp=terrain(projected\_raster, opt='slope', unit='radians', neighbors=8, filename='slp2.tif')

plot(slp)

aspect=terrain(projected\_raster, opt='aspect')

## hillshade- obtained from both slope & aspect

hills = hillShade(slp, aspect, angle=40, direction=270)

plot(hills)

### Pre-Processing Raster and Spatial Data for SDMs

1. Coordinate reference system

All of your raster layers in the same coordinate reference system lat long

# Use utm +zone convert

# utm to lat-long

Ref=””

This is the definition of lat long coordinate reference systems

######### CRS of the raster data

setwd("F:\\SDM\_in R\\Data\\elev")

library(raster)

library(rgdal)

j=raster("join\_59\_60.tif")

plot(j)

j

#utm projection for north borneo

ref = "+proj=utm +zone=50 +ellps=GRS80 +datum=NAD83 +units=m +no\_defs"

projectedU = projectRaster(j, crs = ref)

projectedU

s=raster("slp2.tif")

plot(s)

s

## utm to lat-long

ref= "+proj=longlat +datum=WGS84 +no\_defs +ellps=WGS84 +towgs84=0,0,0"

projectedL = projectRaster(s, crs = ref)

projected

1. Clip raster data to given extent

Different raster have different spatial extent, restrict our analysis of our distribution modeling to a given area.

######### Clip raster data to a given extent

setwd("F:\\SDM\_in R\\Data\\worldclim")

library(raster)

library(rgdal)

tmean=raster("tmean1\_29.tif")

tmean

tmin=raster("tmin1\_29.tif")

tmin

precip=raster("prec1\_29.tif")

precip

##altitude raster

alt=raster("alt\_29.tif")

alt

plot(alt)

#read in a shapefile

setwd("F:\\SDM\_in R\\Data\\MYS\_adm\\states\\P\_malaysia")

library(rgdal)

pm=readOGR(".", "P\_malaysia")

plot(pm, add=T) ## add shape file to this roster add=T

## clip this entire big roster to the extent of Peninsular Malaysia.

altc = crop(precip, pm)

plot(altc)

### I'm going to flip the rest of our target thrusters to use further on.

But essentially we can just crop crop up rosters to any given spatial extent using the crop function.

setwd("F:\\SDM\_in R\\Data\\2\_Raster data process")

lulc1=raster("GLOBCOVER\_L4\_200901\_200912\_V2.3.tif")

plot(lulc1)

lulcc = crop(lulc1, pm)

plot(lulcc)

writeRaster(altc,"F:\\SDM\_in R\\Data\\1\_Raster data\\preciptn.tif")

1. Resize the raster data

## aligning 校正 change resolution

######## resize/resample data- change spatial resolution

require(raster)

setwd("F:\\SDM\_in R\\Data\\1\_Raster data")

land=raster("landuse1.tif")

land

plot(land)

alt

landC=resample(land, asp, method="bilinear")

## first arguement is the raster whose spatial resolution we want to change

## second raster is the one whose resolution we want to apply on 1

plot(landC)

landC

asp

writeRaster(landC,"F:\\SDM\_in R\\Data\\1\_Raster data\\bioclim\_land\\land.tif")

1. Basic data visualization

**interactive leaflet map**

**############ More data visualizations**

**library(spocc)**

**## get data for multiple species from multiple DBs**

**spp= c("Rhinoplax vigil", "Buceros rhinoceros", "Anthracoceros malayanus")**

**dat = occ(query = spp, from ='gbif', gbifopts = list(hasCoordinate=TRUE))**

**dat**

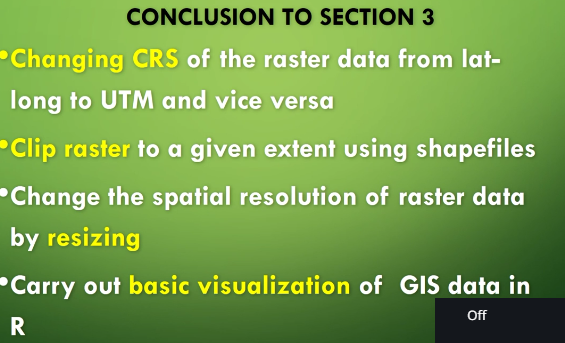
**library(mapr)**

**map\_leaflet(dat) #leaflet based interactive map**

**library(ggplot2)**

**library(ggmap)**

**map\_ggmap(dat) ##ggmap version**



Classical SDM techniques

BIOCLIM climate-envelope model

############# Bioclim- Climate envelope model

##### uses presence only data

## Climate change is creating new challenges for biodiversity conservation.

##Climate envelope models describe the climate where a species

#currently lives (its climate "envelope")

## map the geographic shift of that envelope under climate change

library(dismo)

library(rgeos)

setwd("F:\\SDM\_in R\\Data\\1\_Raster data\\bioclim")

horn=read.csv("hornbill\_my1.csv")

head(horn)

horn1= horn[,-1]#first column not needed, drop first column

ext = extent(99, 105, 1.2, 6.7) #geographic extent of Peninsular M'Sia

# longitude 1, longitude 2, latitude 1, latitude 2

library(rgdal)

all.worldclim = raster::getData("worldclim", res = 10, var = 'bio')

msia.worldclim = crop(all.worldclim, extent(99, 105, 1.2, 6.7))

## set up the bounding box of your map

h.extent = extent(min(horn1$long -1),

max(horn1$long + 1),

min(horn1$lat - 1),

max(horn1$lat + 1))

h.map = gmap(h.extent, type = 'satellite', latlon = TRUE)

plot(h.map)

points(Mercator(horn1[,c('long','lat')]), pch = 16, col = 'red')

#species geo-locations

## Use the bioclim function, which takes your climate layers and the long and lat columns (in that order)

h.bc = bioclim(msia.worldclim, horn1[,c('long','lat')])

par(mfrow = c(4,4))

response(h.bc)

par(mfrow = c(1,1))

horn.d <- bioclim(msia.worldclim, horn1[,c('long','lat')])

par(mfrow = c(4,4))

response(horn.d)

par(mfrow = c(1,1))

horn.d.pred <- predict(object = horn.d, msia.worldclim)

plot(horn.d.pred, main = 'sdm predictions using climate layers')

#### evaluate model performance

#### background data (pseudo-absences) needed for this

#### determine if the model can differentiate bw the habitat & the background

#species presence

head(horn1)

plot(msia.worldclim) #predictors

horn.d = bioclim(msia.worldclim, horn1[,c('long','lat')])

## bioclimatic model

ext = extent(99, 105, 1.2, 6.7)

backg = randomPoints(msia.worldclim, n=1000,ext=ext, extf = 1.25)

#background/pseudo-absence data

e = evaluate(horn1, backg, horn.d, msia.worldclim)

#presence, background, model, predictors

e

plot(e, 'ROC')

Maxtent sdm

we are going to learn to implement some of the most important and common machine learning algorithms in our and evaluating their performance using AUC.

 Pre-processing Steps Prior to Modelling With Presence & Absence Data

We can use pseudo absences or background data if we want to test our models and see how good or bad it is.

############ Preprocessing-- defining presence & absence

setwd("F:\\SDM\_in R\\Data\\1\_Raster data\\bioclim")

library(raster)

datafiles = Sys.glob("\*.tif") #Or whatever identifies your files

stck = stack() #empty stack for raster

for(i in 1:NROW(datafiles)){

tempraster = raster(datafiles[i])

stck = stack(stck,tempraster)

}

stck #raster predictors as a stack

plot(stck,2)

### presence data

horn=read.csv("hornbill\_my1.csv")

head(horn)

horn1= horn[,-1]#first column not needed

points(horn1, col='blue') ## these are my species geo-location data

prs1= extract(stck, horn1) ##we are going to extract the presences from under these Bluepoint.

set.seed(1)

backgr = randomPoints(stck, 500) #500 random points

### And now I'm going to select 500 random points using the function random points and I'm going to get my 500 points from my stack and then I'm going to choose these absence values because for species distribution model we need both. Presence and absence of pseudo absences.

absvals = extract(stck, backgr) #choose absence values from the background

pb = c(rep(1, nrow(prs1)), rep(0, nrow(absvals))) ## store in pb

sdmdata = data.frame(cbind(pb, rbind(prs1, absvals))) ## convert to data frame

head(sdmdata)

sdmdata=na.omit(sdmdata) ## omit na

summary(sdmdata)

tail(sdmdata)

############select an area of absence based on ecological

#######considerations

library(raster)

e = drawExtent()##dynamically select absence area

abs=crop(stck, e)

plot(abs)

backgr=randomPoints(abs,135)#80-2-ratio

absvals2 = extract(abs, backgr)#400 pseuduo-absence

pb = c(rep(1, nrow(prs1)), rep(0, nrow(absvals2)))

smdata=data.frame(cbind(pb,rbind(prs1,absvals2)))

head(sdmdata)

Prior to Implementing Machine Learning

########### Pre-proessing for machine learning

write.csv(sdmdata,"Pres\_abs.csv")

setwd("F:/SDM\_in R/Data/1\_Raster data/bioclim\_land")

pa=read.csv("Pres\_abs.csv")

#pa=na.omit(pa)

head(pa)

summary(pa)

library(caret)

set.seed(1) #pseudo-repeatability

trainIndex = createDataPartition(pa$pb, p = .75,

list = FALSE,

times = 1) #y as basis of splitting

training = pa[ trainIndex,] #75% data for model training

testing= pa[-trainIndex,] #25% for model testing

head(training)

Prior to Implementing Machine Learning5:32