

A brief tutorial on running Maxent in R

Xiao Feng, Cassandra Walker, Fikirte Gebresenbet

July 4, 2017

Contents

1. Setup the working environment	1
1.1 Load packages	1
1.2 Set up the Maxent path	1
2. Prepare data input	2
2.1 Load environmental layers	2
2.2 Occurrence data	2
2.2.1 Download occurrence data	2
2.2.2 Clean occurrence data	2
2.3 Set up study area	4
2.4 Split occurrence data into training & testing	6
2.5 Format data for Maxent	6
3 Maxent models	7
3.1 Simple implementation	7
3.2 Predict function	9
3.3 Model evaluation	10
4 Maxent parameters	11
4.1 Select features	11
4.2 Change beta-multiplier	12
4.3 Specify projection layers	12
4.4 Clamping function	15
4.5 Cross validation	17

1. Setup the working environment

1.1 Load packages

```
library("dismo")
library("raster")
library("knitr")
require("rgeos")
```

```
## Warning: package 'rgeos' was built under R version 3.4.2
```

```
dyn.load('/Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk/Contents/Home/jre/lib/server/libjvm.dylib')
require("rJava")
#knitr::opts_knit$set(root.dir = 'd:/projects/2017_7_workshop_enm_R')
knitr::opts_knit$set(root.dir = '/Users/iel82user/Google Drive/1_osu_lab/projects/2017_7_workshop_enm_R')
```

1.2 Set up the Maxent path

```
# download maxent.jar 3.3.3k, and place the file in the desired folder
utils::download.file(url="https://raw.githubusercontent.com/mrmaxent/Maxent/master/ArchivedReleases/3.3
```

```

mode="wb") ## wb for binary file, otherwise maxent.jar can not execute
# also note that both R and Java need to be the same bit (either 32 or 64) to be compatible to run
#options( java.parameters = c("-Xss2560k", "-Xmx2g") ) ## increase memory size of the JVM, this code ma

```

2. Prepare data input

2.1 Load environmental layers

```

# load GIS layers; here we used bioclim variables as an example (downloaded from worldclim.org)
clim <- list.files("data/bioclim/",pattern=".bil$",full.names = T)
clim <- raster::stack(clim) ## stacking the bioclim variables to process them at one go

```

2.2 Occurrence data

2.2.1 Download occurrence data

```

# download occurrence data from GBIF; for example, we used nine-banded armadillo that distributes in Am
if(file.exists("data/occ_raw")){
  #cat(1)
  load("data/occ_raw")
}else{
  #cat(2)
  occ_raw <- gbif("Dasypus novemcinctus")
  save(occ_raw,file = "data/occ_raw")
  write.csv("data/occ_raw.csv")
}
#head(occ_raw)

```

2.2.2 Clean occurrence data

```

# remove bad coordinates, where either the lat or long coordinate is missing
occ_clean <- subset(occ_raw,!is.na(lat))&(!is.na(lon))
cat(nrow(occ_raw)-nrow(occ_clean), "records are removed")

```

2426 records are removed

```

# remove duplicated data based on latitude and longitude
dups <- duplicated(occ_clean[c("lat","lon")])
occ_unique <- occ_clean[!dups,]
cat(nrow(occ_clean)-nrow(occ_unique), "records are removed")

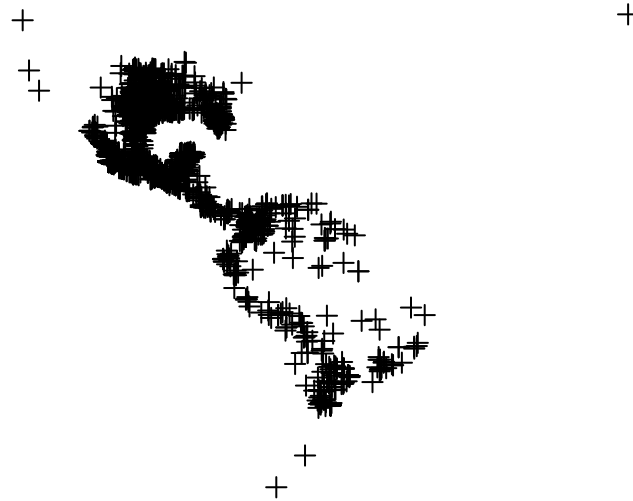
```

1506 records are removed

```

# make occ spatial
coordinates(occ_unique) <- ~ lon + lat
plot(occ_unique) ## we may notice an erroneous point

```

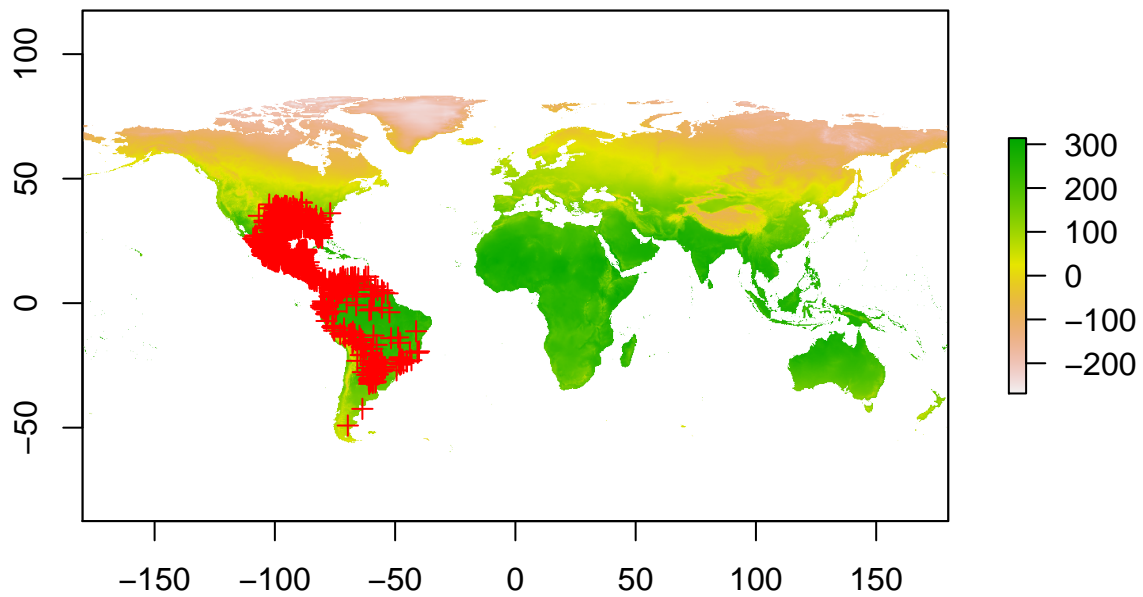


```
# remove some errors
occ_unique <- occ_unique[which(occ_unique$lon > -110 &
                              occ_unique$lon < -40),]

# make occ sparse (keep one occ per cell)
cells <- cellFromXY(clim[[1]], occ_unique)
dups <- duplicated(cells)
occ_final <- occ_unique[!dups,]
cat(nrow(occ_unique) - nrow(occ_final), "records are removed")
```

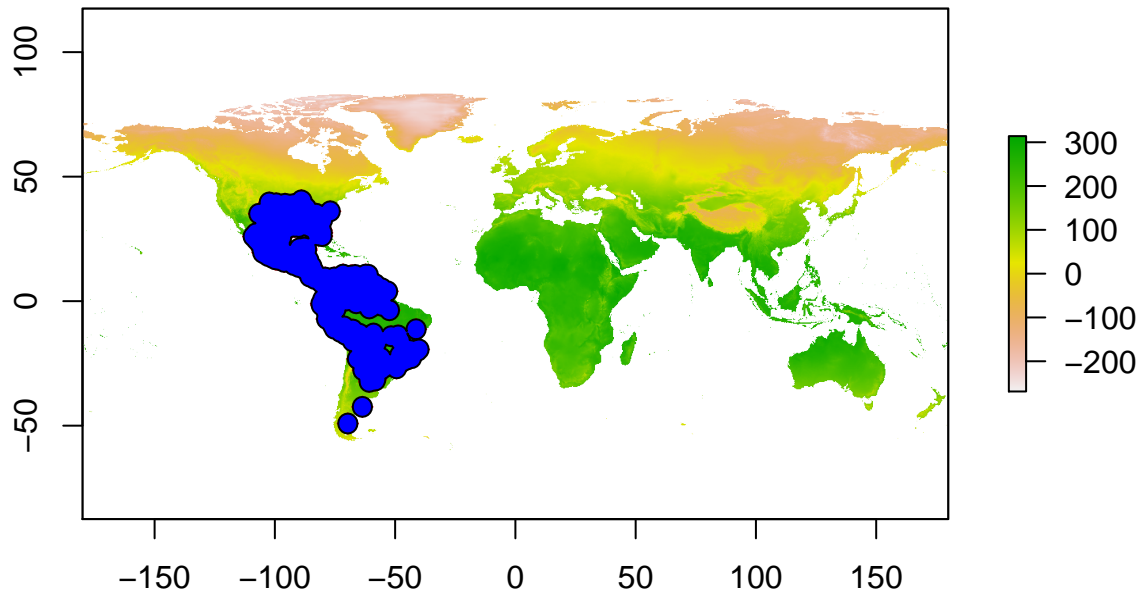
```
## 1124 records are removed
```

```
plot(clim[[1]]) ## to draw the first layer (or replace [[1]] with any nth number of the layers with in
plot(occ_final, add=T, col="red") ## the 'add=T' tells R to put the incoming data on the existing layer (
```



2.3 Set up study area

```
# this creates a buffer around the occurrence data
occ_buff <- buffer(occ_final,4) ## 4 decimal degree
plot(clim[[1]]) ## this plots the first element ([[1]]) in the raster stack and adds the occurrence data
plot(occ_final,add=T,col="red") ## this adds the occurrence data
plot(occ_buff,add=T,col="blue") ## this adds the buffer polygon
```



```

# if the area we will mask from is very large, use crop first; it will crop from a rectangle and then mask
studyArea <- crop(clim,extent(occ_buff)) ## gives a coarser rectangle of the study area (a rectangle enclosing the study area)

studyArea <- mask(studyArea,occ_buff) ## the 'study area' created by cropping and masking the raster stack

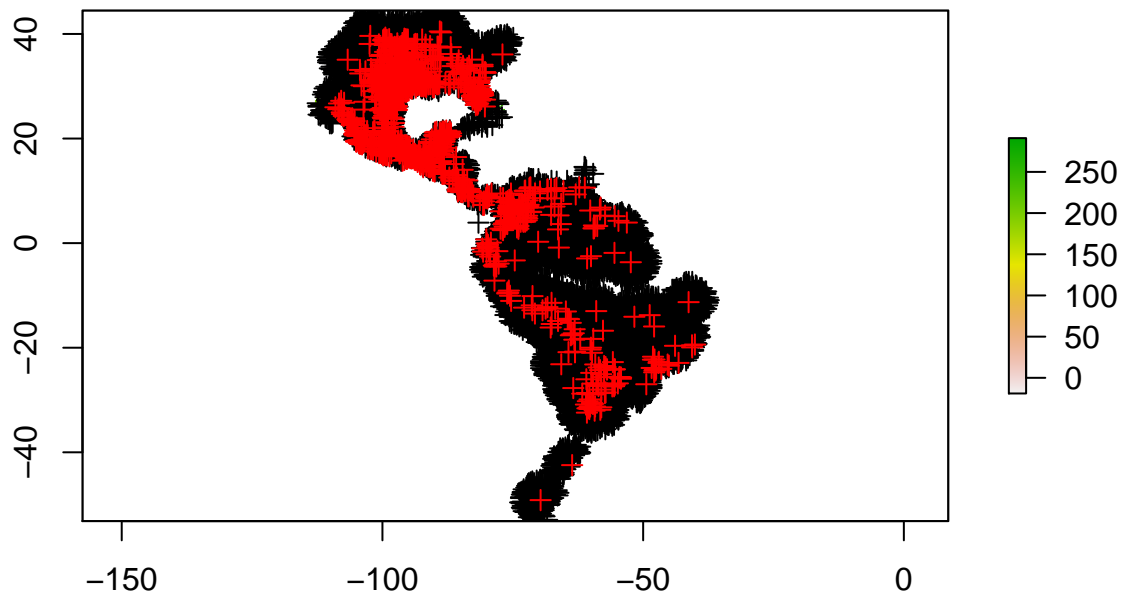
# save the buffer areas into raster files
writeRaster(studyArea,
            filename=paste0("data/studyarea/",names(studyArea),".asc"), ## a series of names for output files
            format="ascii", ## the output format
            bylayer=TRUE, ## this will save a series of layers
            overwrite=T)

# select background points from this buffered area
set.seed(1) ## when the number provided to set.seed() function, the same random sample will be selected

bg <- sampleRandom(x=studyArea,
                  size=10000,
                  na.rm=T, ## na.rm is telling R to remove the 'Not Applicable' points,
                  sp=T) ## sp is telling R to give us a spatially points

plot(studyArea[[1]])
plot(bg,add=T) ## add the background points to the plotted raster
plot(occ_final,add=T,col="red") ## add the occurrence data to the plotted raster

```



2.4 Split occurrence data into training & testing

```
# randomly select 50% for training
set.seed(1) ## get the same random sample for training and testing
selected <- sample(1:nrow(occ_final),nrow(occ_final)*0.5)
occ_train <- occ_final[selected,] ## this is the selection
occ_test <- occ_final[-selected,] ## this is the opposite of the selection
```

2.5 Format data for Maxent

```
# extracting env conditions for training occ from the raster stack; a data frame is returned
p <- extract(clim,occ_train) ## env conditions for training occ this makes a dataframe since environment
p_test <- extract(clim,occ_test) ## env conditions for testing occ
a <- extract(clim,bg) ## env conditions for background
pa <- c(rep(1,nrow(p)), rep(0,nrow(a))) ## repeat the number 1 as many numbers as the number of rows in
## (rep(1,nrow(p)) creating the number of rows as the p data set to have the number one as the indicator
## rep(0,nrow(a)) creating the number of rows as the a data set to have the number zero as the indicator
## The c combines these ones and zeros into a new vector that can be added to the Maxent table
pder <- as.data.frame(rbind(p,a)) ## this makes a data frame with the environmental attributes of the p
```

3 Maxent models

3.1 Simple implementation

```
# mod <- maxent(x=clim, p=occ_train)
mod <- maxent(x=pder, ## env conditions
             p=pa,    ## 1:presence or 0:absence
             path=paste0(getwd(), "/output/maxent_outputs"), ## folder to store maxent output; if we do
             args=c("responsecurves") ## a lot of parameters can be specified here
             )
## the maxent functions runs a model in the default settings..to change these parameters, you have to t

# view a maxent model in a html browser
mod

## class      : MaxEnt
## variables: bio1 bio10 bio11 bio12 bio13 bio14 bio15 bio16 bio17 bio18 bio19 bio2 bio3 bio4 bio5 bio6

# view detailed results
mod@results

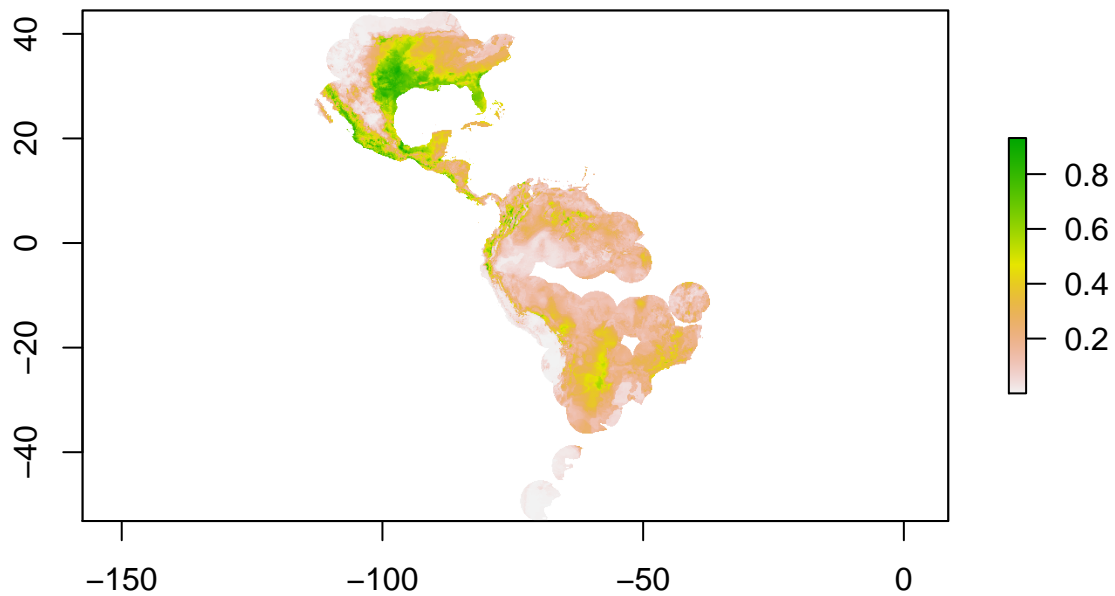
##                                     [,1]
## X.Training.samples                 655.0000
## Regularized.training.gain          0.7265
## Unregularized.training.gain        0.9604
## Iterations                         500.0000
## Training.AUC                       0.8596
## X.Background.points                10575.0000
## bio1.contribution                  17.1627
## bio10.contribution                 20.4753
## bio11.contribution                 8.7616
## bio12.contribution                 8.4875
## bio13.contribution                 1.8276
## bio14.contribution                 0.7496
## bio15.contribution                 9.2740
## bio16.contribution                 0.6694
## bio17.contribution                 0.6045
## bio18.contribution                 0.9334
## bio19.contribution                 0.9610
## bio2.contribution                  1.0134
## bio3.contribution                 15.1186
## bio4.contribution                  1.3084
## bio5.contribution                 8.2928
## bio6.contribution                  3.3366
## bio7.contribution                  0.3255
## bio8.contribution                  0.1911
## bio9.contribution                  0.5069
## bio1.permutation.importance        16.4025
## bio10.permutation.importance        10.6769
## bio11.permutation.importance         3.4186
## bio12.permutation.importance         7.1080
## bio13.permutation.importance         3.0867
## bio14.permutation.importance         3.9966
## bio15.permutation.importance        20.4166
## bio16.permutation.importance         0.3416
```

## bio17.permutation.importance	0.5438
## bio18.permutation.importance	0.6744
## bio19.permutation.importance	1.3227
## bio2.permutation.importance	1.5541
## bio3.permutation.importance	2.9937
## bio4.permutation.importance	20.8722
## bio5.permutation.importance	1.2303
## bio6.permutation.importance	3.2102
## bio7.permutation.importance	1.0008
## bio8.permutation.importance	0.0737
## bio9.permutation.importance	1.0766
## Entropy	8.5492
## Prevalence..average.of.logistic.output.over.background.sites.	0.2410
## Fixed.cumulative.value.1.cumulative.threshold	1.0000
## Fixed.cumulative.value.1.logistic.threshold	0.0625
## Fixed.cumulative.value.1.area	0.8119
## Fixed.cumulative.value.1.training.omission	0.0046
## Fixed.cumulative.value.5.cumulative.threshold	5.0000
## Fixed.cumulative.value.5.logistic.threshold	0.1355
## Fixed.cumulative.value.5.area	0.6447
## Fixed.cumulative.value.5.training.omission	0.0229
## Fixed.cumulative.value.10.cumulative.threshold	10.0000
## Fixed.cumulative.value.10.logistic.threshold	0.1833
## Fixed.cumulative.value.10.area	0.5157
## Fixed.cumulative.value.10.training.omission	0.0473
## Minimum.training.presence.cumulative.threshold	0.0200
## Minimum.training.presence.logistic.threshold	0.0054
## Minimum.training.presence.area	0.9608
## Minimum.training.presence.training.omission	0.0000
## X10.percentile.training.presence.cumulative.threshold	17.8502
## X10.percentile.training.presence.logistic.threshold	0.2531
## X10.percentile.training.presence.area	0.3762
## X10.percentile.training.presence.training.omission	0.0992
## Equal.training.sensitivity.and.specificity.cumulative.threshold	32.3187
## Equal.training.sensitivity.and.specificity.logistic.threshold	0.3702
## Equal.training.sensitivity.and.specificity.area	0.2168
## Equal.training.sensitivity.and.specificity.training.omission	0.2168
## Maximum.training.sensitivity.plus.specificity.cumulative.threshold	25.9220
## Maximum.training.sensitivity.plus.specificity.logistic.threshold	0.3177
## Maximum.training.sensitivity.plus.specificity.area	0.2765
## Maximum.training.sensitivity.plus.specificity.training.omission	0.1389
## Balance.training.omission..predicted.area.and.threshold.value.cumulative.threshold	2.0374
## Balance.training.omission..predicted.area.and.threshold.value.logistic.threshold	0.0965
## Balance.training.omission..predicted.area.and.threshold.value.area	0.7536
## Balance.training.omission..predicted.area.and.threshold.value.training.omission	0.0076
## Equate.entropy.of.thresholded.and.original.distributions.cumulative.threshold	11.3240
## Equate.entropy.of.thresholded.and.original.distributions.logistic.threshold	0.1949
## Equate.entropy.of.thresholded.and.original.distributions.area	0.4881
## Equate.entropy.of.thresholded.and.original.distributions.training.omission	0.0534

3.2 Predict function

```
# maxent.R doesnt give us a prediction of training data/layers (unless you specify the projection layer)  
# a maxent model (in R) can be projected on raster layers or a dataframes
```

```
# example 1, project to out study area [raster]  
ped1 <- predict(mod,studyArea)  
plot(ped1)
```



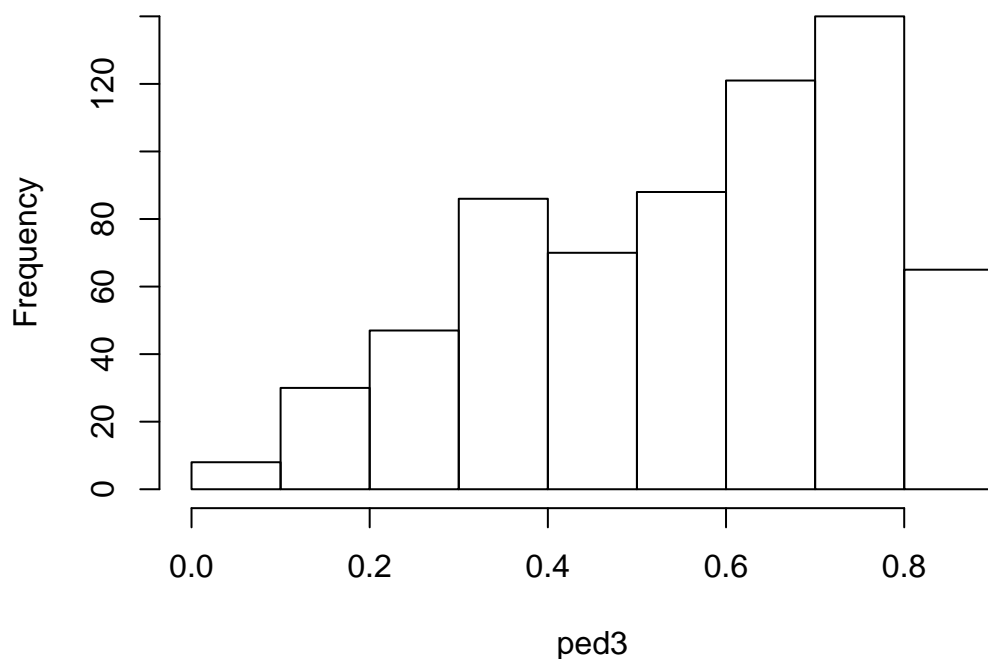
```
# example 2, project to the world  
#ped2 <- predict(mod,clim)  
#plot(ped2)
```

```
# example 3, project training occurrences [dataframes]  
ped3 <- predict(mod,p)  
head(ped3)
```

```
## [1] 0.7553921 0.3420225 0.5019929 0.5993227 0.7655950 0.7684774
```

```
hist(ped3)
```

Histogram of ped3



3.3 Model evaluation

```
# using "training data" to evaluate  
# "evaluate" is an evaluation function from dismo package; p= presence and a=background  
mod_eval_train <- dismo::evaluate(p=p,a=a,model=mod) #p & a are dataframes (the p and a are the training data)  
print(mod_eval_train)
```

```
## class      : ModelEvaluation  
## n presences : 655  
## n absences  : 10000  
## AUC         : 0.8807075  
## cor         : 0.4044683  
## max TPR+TNR at : 0.3175671
```

```
# This is the test AUC  
mod_eval_test <- dismo::evaluate(p=p_test,a=a,model=mod)  
print(mod_eval_test) # training AUC may be higher than testing AUC
```

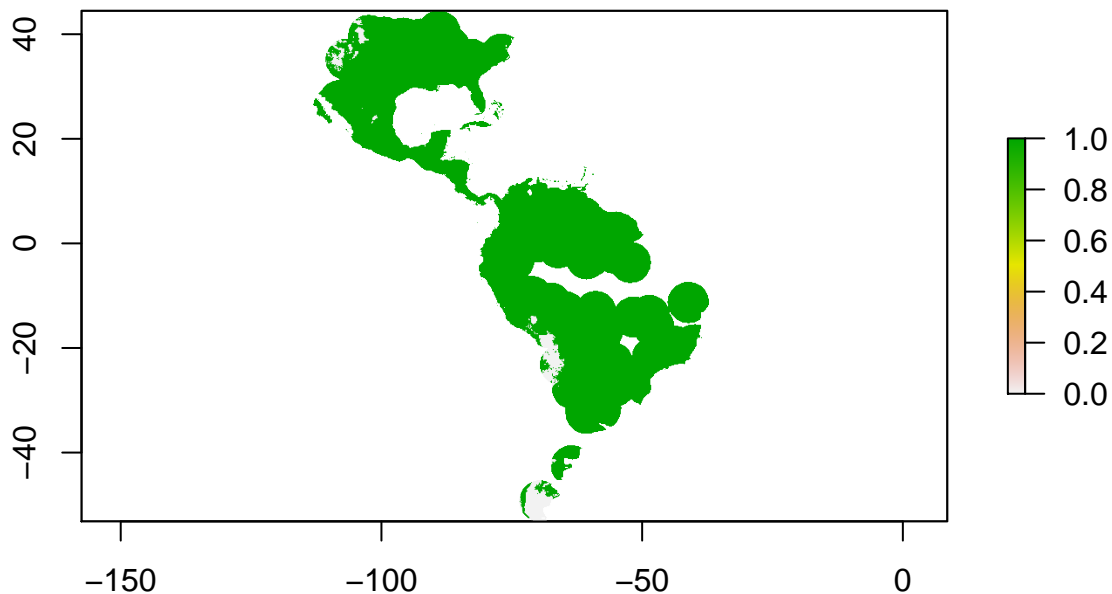
```
## class      : ModelEvaluation  
## n presences : 657  
## n absences  : 10000  
## AUC         : 0.8401474  
## cor         : 0.3532565  
## max TPR+TNR at : 0.3763733
```

```

# calculate thresholds of models
# threshold function is in dismo and based on the evaluation function
thd1 <- threshold(mod_eval_train,"no_omission") # 0% omission rate [minimum training presence]
thd2 <- threshold(mod_eval_train,"spec_sens") # hiest TSS

# Only plot the predictions that are higher than the threshold....giving a binary output
plot(ped1>=thd1) ## plotting points that are above the previously calculated tresholed value

```



4 Maxent parameters

4.1 Select features

```

# load the function that prepares parameters for maxent
source("code/Appendix2_prepPara.R")

mod1_autofeature <- maxent(x=pder[c("bio1","bio4","bio11")], ## env conditions, here we selected only 3
                           p=pa, ## 1:presence or 0:absence
                           path=paste0(getwd(),"/output/maxent_outputs1_auto"), ## path of maxent output, this is the
                           args=prepPara(userfeatures=NULL) ) ## default is autofeature

# or select Linear& Quadratic features
mod1_lq <- maxent(x=pder[c("bio1","bio4","bio11")], ## env conditions, here we selected only 3 predictors
                  p=pa, ## 1:presence or 0:absence
                  path=paste0(getwd(),"/output/maxent_outputs1_lq"), ## path of maxent output, this is the

```

```
args=prepPara(userfeatures="LQ") ) ## default is autofeature, here LQ represents Linear&
```

4.2 Change beta-multiplier

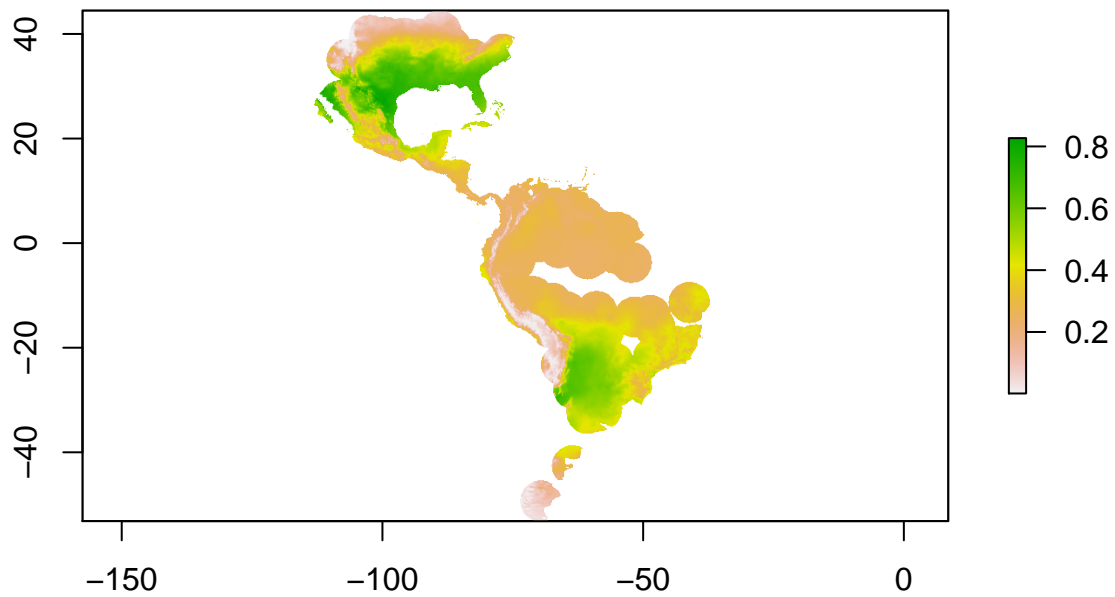
```
#change betamultiplier for all features
mod2 <- maxent(x=pder[c("bio1","bio4","bio11")],
               p=pa,
               path=paste0(getwd(),"/output/maxent_outputs2_o.5"),
               args=prepPara(userfeatures="LQ",
                             betamultiplier=0.5) )

mod2 <- maxent(x=pder[c("bio1","bio4","bio11")],
               p=pa,
               path=paste0(getwd(),"/output/maxent_outputs2_complex"),
               args=prepPara(userfeatures="LQH", ## include L, Q, H features
                             beta_lqp=1.5, ## use different betamultiplier for different features
                             beta_hinge=0.5 ) )
```

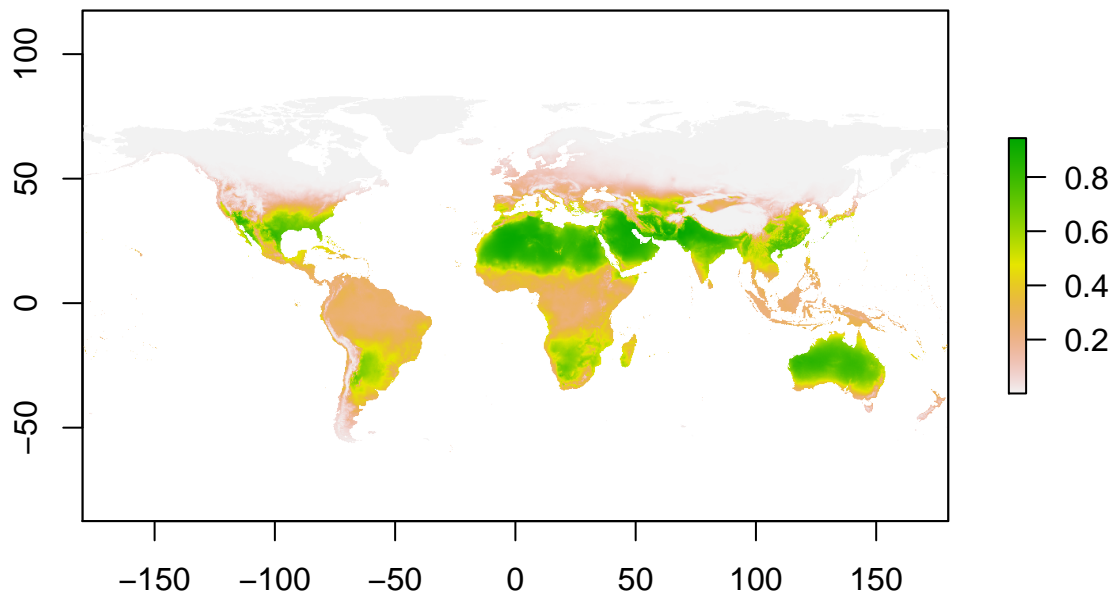
4.3 Specify projection layers

```
# note: 1)the projection layers must exist in the hard disk (as relative to computer RAM); (2) the name
mod3 <- maxent(x=pder[c("bio1","bio11")],
               p=pa,
               path=paste0(getwd(),"/output/maxent_outputs3_prj1"),
               args=prepPara(userfeatures="LQ",
                             betamultiplier=1,
                             projectionlayers="/Users/iel82user/G

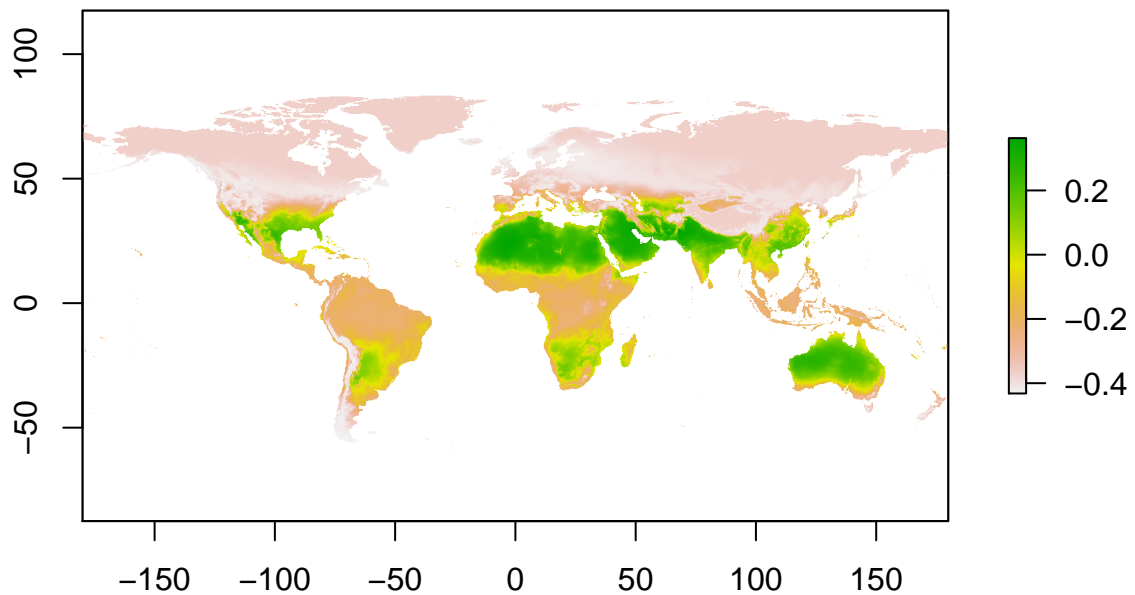
# load the projected map
ped <- raster(paste0(getwd(),"/output/maxent_outputs3_prj1/species_studyarea.asc"))
plot(ped)
```



```
# we can also project on a broader map, but please caustion about the inaccuracy associated with model
mod3 <- maxent(x=pder[c("bio1","bio11")],
               p=pa,
               path=paste0(getwd(), "/output/maxent_outputs3_prj2"),
               args=prepPara(userfeatures="LQ",
                             betamultiplier=1,
                             projectionlayers="/Users/iel82user/G
# plot the map
ped <- raster(paste0(getwd(), "/output/maxent_outputs3_prj2/species_bioclim.asc"))
plot(ped)
```



```
# simply check the difference if we used a different betamultiplier
mod3_beta1 <- maxent(x=pder[c("bio1","bio11")],
  p=pa,
  path=paste0(getwd(),"/output/maxent_outputs3_prj3"),
  args=prepPara(userfeatures="LQ",
    betamultiplier=100, ## for an extreme example, set beta as 100
    projectionlayers="/Users/iel82user/Google Drive/1_osu_lab/projects/2017_7_w
ped3 <- raster(paste0(getwd(),"/output/maxent_outputs3_prj3/species_bioclim.asc"))
plot(ped-ped3) ## quickly check the difference between the two predictions
```

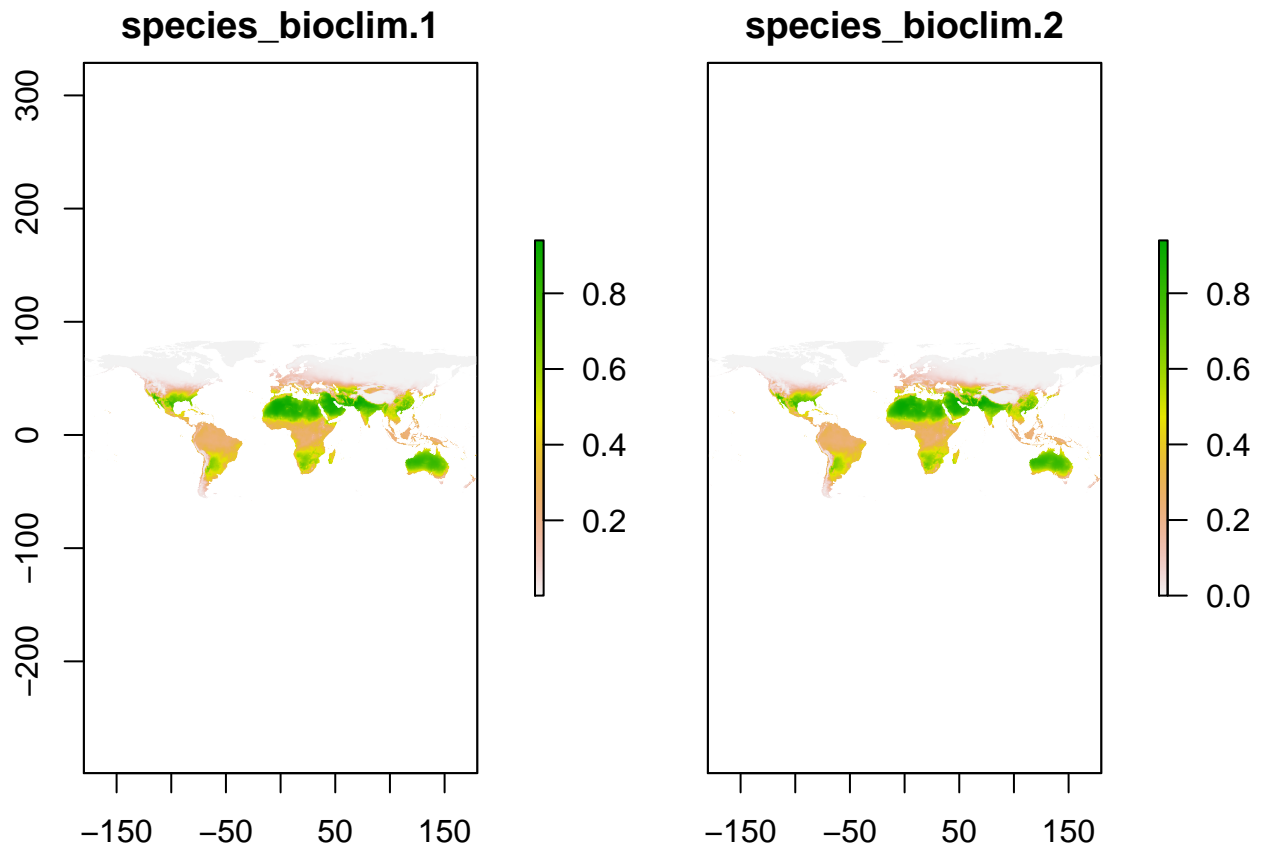


4.4 Clamping function

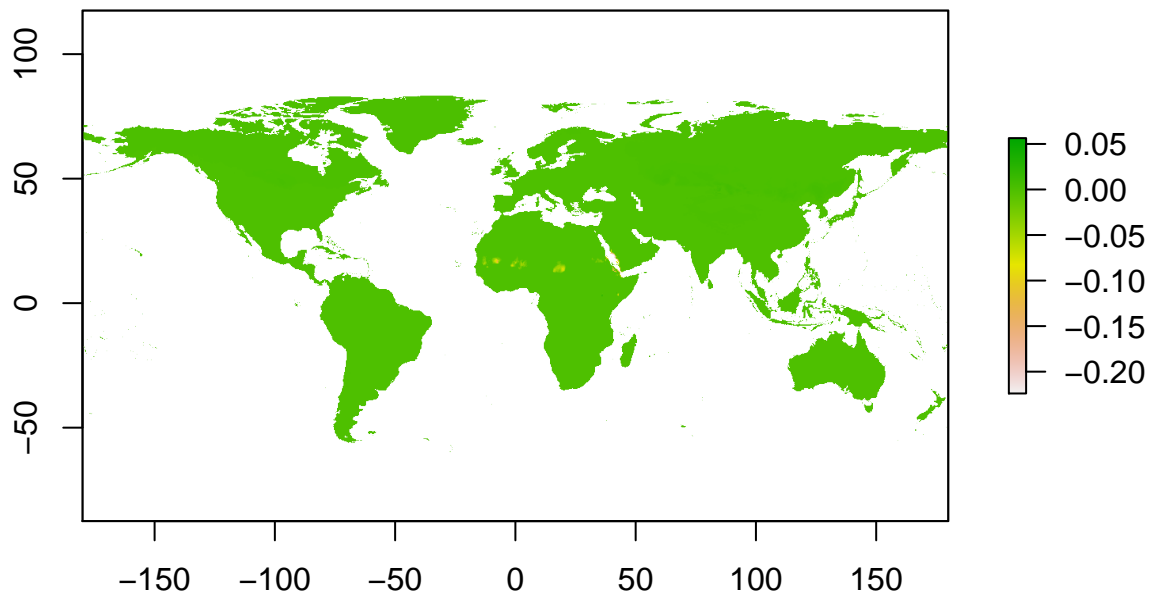
```
# enable or disable clamping function; note clamping function is involved when projecting
mod4_clamp <- maxent(x=pder[c("bio1","bio11")],
                    p=pa,
                    path=paste0(getwd(),"/output/maxent_outputs4_clamp"),
                    args=prepPara(userfeatures="LQ",
                                   betamultiplier=1,
                                   doclamp = TRUE,
                                   projectionlayers="/Users/iel82user/Google Drive/1_osu_lab/projects/2

mod4_noclamp <- maxent(x=pder[c("bio1","bio11")],
                      p=pa,
                      path=paste0(getwd(),"/output/maxent_outputs4_noclamp"),
                      args=prepPara(userfeatures="LQ",
                                     betamultiplier=1,
                                     doclamp = FALSE,
                                     projectionlayers="/Users/iel82user/Google Drive/1_osu_lab/project

ped_clamp <- raster(paste0(getwd(),"/output/maxent_outputs4_clamp/species_bioclim.asc") )
ped_noclamp <- raster(paste0(getwd(),"/output/maxent_outputs4_noclamp/species_bioclim.asc") )
plot(stack(ped_clamp,ped_noclamp))
```



```
plot(ped_clamp - ped_noclamp) ## we may notice small difference, especially clamp shows higher prediction
```

4.5 Cross validation

```
mod4_cross <- maxent(x=pder[c("bio1","bio11")], p=pa,
  path=paste0(getwd(),"/output/maxent_outputs4_cross"),
  args=prepPara(userfeatures="LQ",
    betamultiplier=1,
    doclamp = TRUE,
    projectionlayers="/Users/iel82user/Google Drive/1_osu_lab/proj/
  replicates=5, ## 5 replicates
  replicatetype="crossvalidate") ) ##possible values are: cross
```