Modelling

```
#Restore workspace - includes all the objects/libraries from the last run. This way we don't have to ru
# library(session)
# restore.session('Modelling.Rda')
```

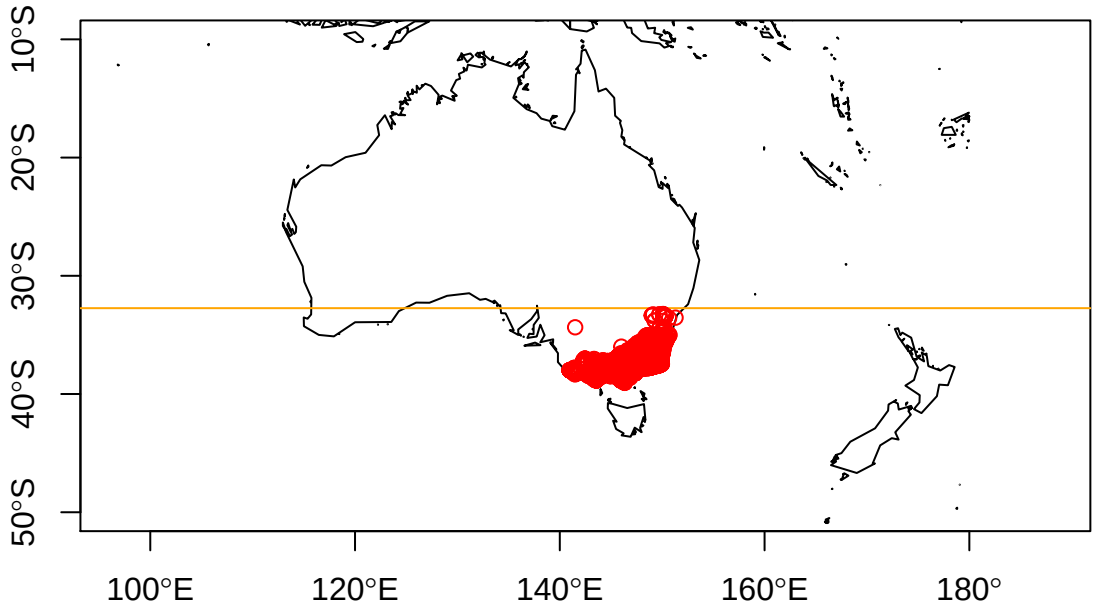
```
#Loading packages
#install.packages("pacman")
source("Functions.R")
library(pacman)
packages = c(
  "installr" ,
  "session",
  "pacman",
  "BiocManager",
  "raster",
  "rgdal",
  "dismo",
  "rJava",
  "sp",
  "maptools",
  "ROCR",
  "randomForest",
  "adabag",
  "e1071",
  "tree",
  "neuralnet",
  "shiny",
  "leaflet",
  "smds",
  "remotes",
  "rgeos",
  "rpart",
  "lubridate",
  "dplyr",
  "spatialEco",
  "SSDM",
  "testthat",
  "knitr",
  "rmarkdown",
  "DT",
  "rsconnect",
  "shinyLP",
  "shinydashboard",
  "shinybusy"
  )

p_load(packages, character.only = TRUE)
#updateR()
p_loaded(packages, character.only = TRUE)
```

```
##       installr        session         pacman    BiocManager         raster
##           TRUE           TRUE           TRUE           TRUE           TRUE
```

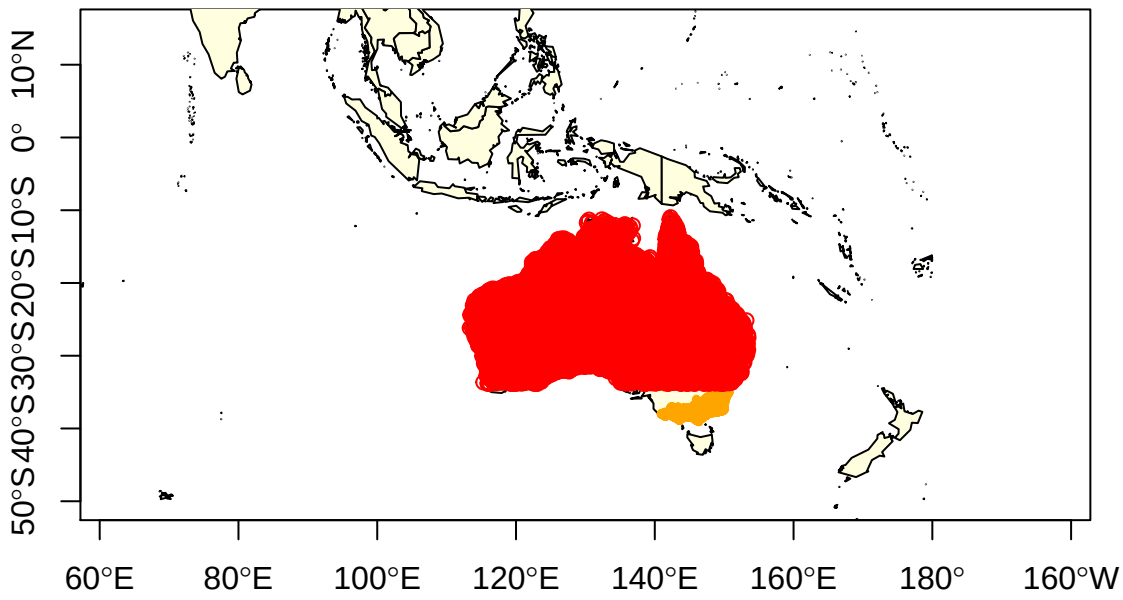
```
##          rgdal          dismo          rJava             sp       maptools
##           TRUE           TRUE           TRUE           TRUE           TRUE
##           ROCR   randomForest         adabag         e1071           tree
##           TRUE           TRUE           TRUE           TRUE           TRUE
##      neuralnet          shiny       leaflet           smds        remotes
##           TRUE           TRUE           TRUE           TRUE           TRUE
##          rgeos          rpart     lubridate          dplyr     spatialEco
##           TRUE           TRUE           TRUE           TRUE           TRUE
##           SSDM       testthat          knitr      rmarkdown             DT
##           TRUE           TRUE           TRUE           TRUE           TRUE
##      rsconnect        shinyLP shinydashboard      shinybusy
##           TRUE           TRUE           TRUE           TRUE
```

```
#Downloading raster environmental data from Worldclim.
#http://worldclim.org/version2
predictors <- getData("worldclim",var="bio",res = 2.5)
names(predictors) = c(
  "Annual Mean Temperature",
  "Mean Diurnal Range" ,
  "Isothermality",
  "Temperature Seasonality",
  "Max Temperature of Warmest Month",
  "Min Temperature of Coldest Month",
  "Temperature Annual Range",
  "Mean Temperature of Wettest Quarter" ,
  "Mean Temperature of Driest Quarter",
  "Mean Temperature of Warmest Quarter",
  "Mean Temperature of Coldest Quarter",
  "Annual Precipitation",
  "Precipitation of Wettest Month",
  "Precipitation of Driest Month",
  "Precipitation Seasonality",
  "Precipitation of Wettest Quarter",
  "Precipitation of Driest Quarter",
  "Precipitation of Warmest Quarter",
  "Precipitation of Coldest Quarter"
  )

extent_australia = extent(100,160,-50,15)
```

```
Agile.c = read.csv("AgileC.csv") #Reading the data in; 2 columns containing the longitude and latitude
Agile.c = Agile.c[,c(2:3)]

Agile.bg_extent = extent(100,160,max(Agile.c$lat) + 0.5, -10) #Coordinates around Australia where our o
data("wrld_simpl")
plot(wrld_simpl, xlim = c(125, 160), ylim= c(-50,-10), axes = TRUE)
abline(h = (max(Agile.c$lat)) + 0.5, col = 'orange')
points(Agile.c, col = 'red')
```

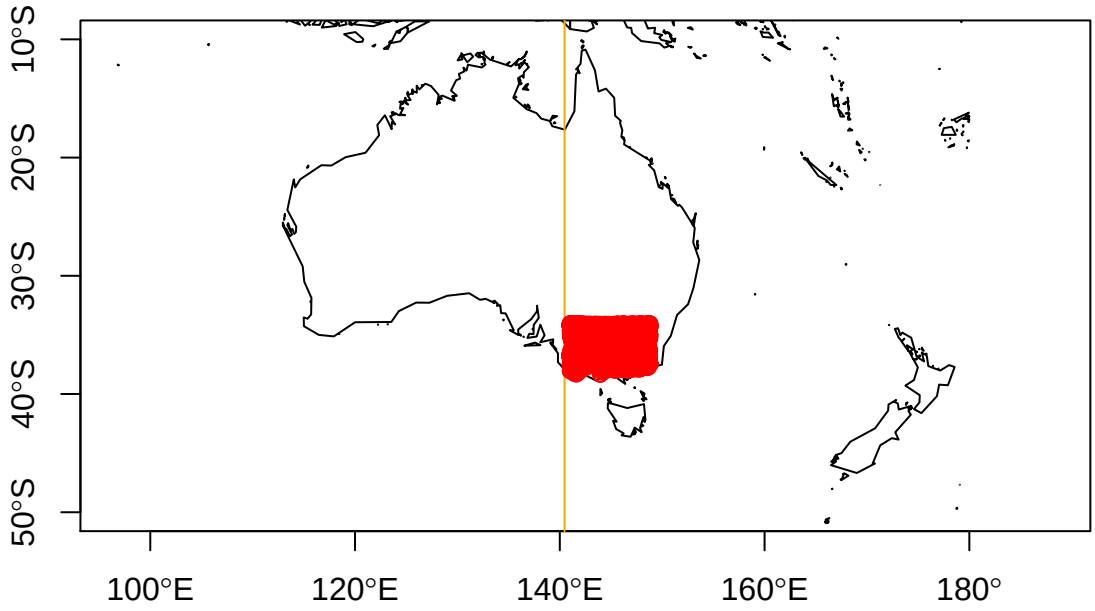



```
#Returns dataframe containing lon/lat values
Agile.bg = generateRandomPoints(Agile.c, Agile.bg_extent)



Agile.extent = getExtent(Agile.c) #extent of observation points

#Visualize these points
#Orange = Prescene, red = pesuedo
plot_(Agile.c, Agile.bg, extent = extent_australia)
```

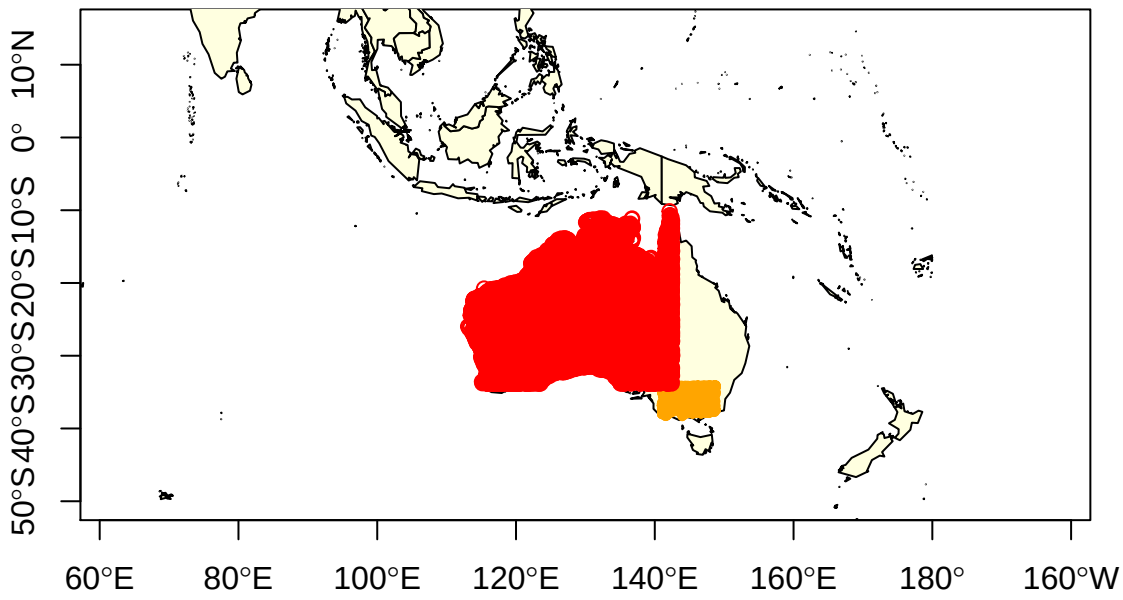



```
# Agile.sdm = extractPredictorsValues(Agile.c, Agile.bg ,predictors)
```

```
BTreeCreeper.c = read.csv("BrownTreeCreeperC.csv") #Reading the data in; 2 columns containing the longi
BTreeCreeper.c = BTreeCreeper.c[,c(2:3)]

BTreeCreeper.bg_extent = extent(100,min(BTreeCreeper.c$lon) - 0.5,-32.73765, -10) #Coordinates around A
data("wrld_simpl")
plot(wrld_simpl, xlim = c(125, 160), ylim= c(-50,-10), axes = TRUE)
abline(v = (min(BTreeCreeper.c$lon)) - 0.5, col = 'orange')
points(BTreeCreeper.c, col = 'red')
```

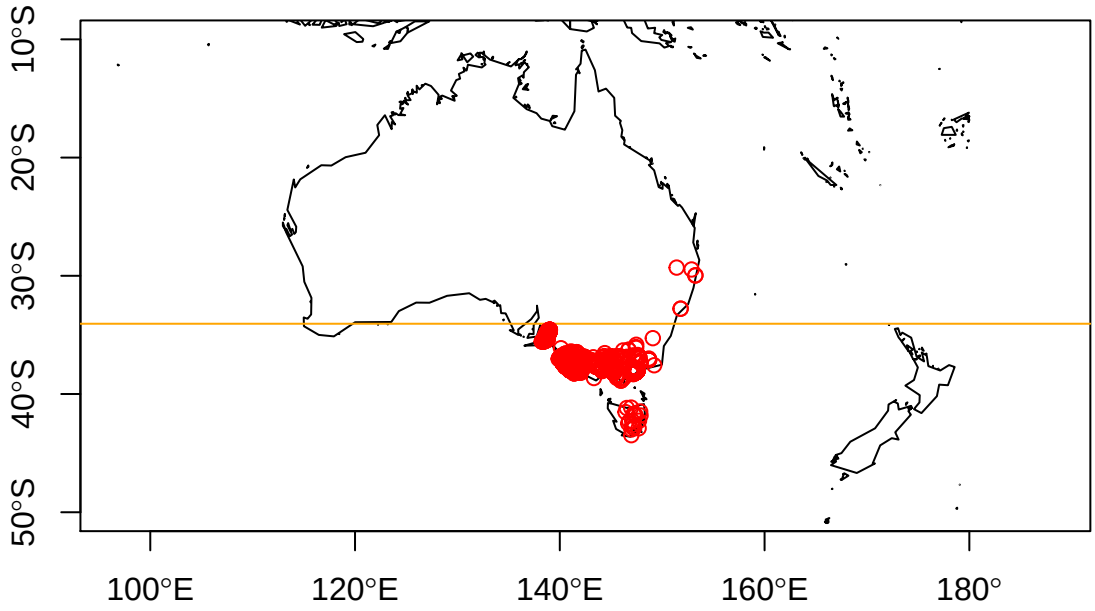



```
#Returns dataframe containing lon/lat values
BTreeCreeper.bg = generateRandomPoints(BTreeCreeper.c, BTreeCreeper.bg_extent)

BTreeCreeper.extent = getExtent(BTreeCreeper.c) #extent of observation points

#Visualize these points
#Orange = Prescene, red = pesuedo
plot_(BTreeCreeper.c, BTreeCreeper.bg, extent = extent_australia)
```

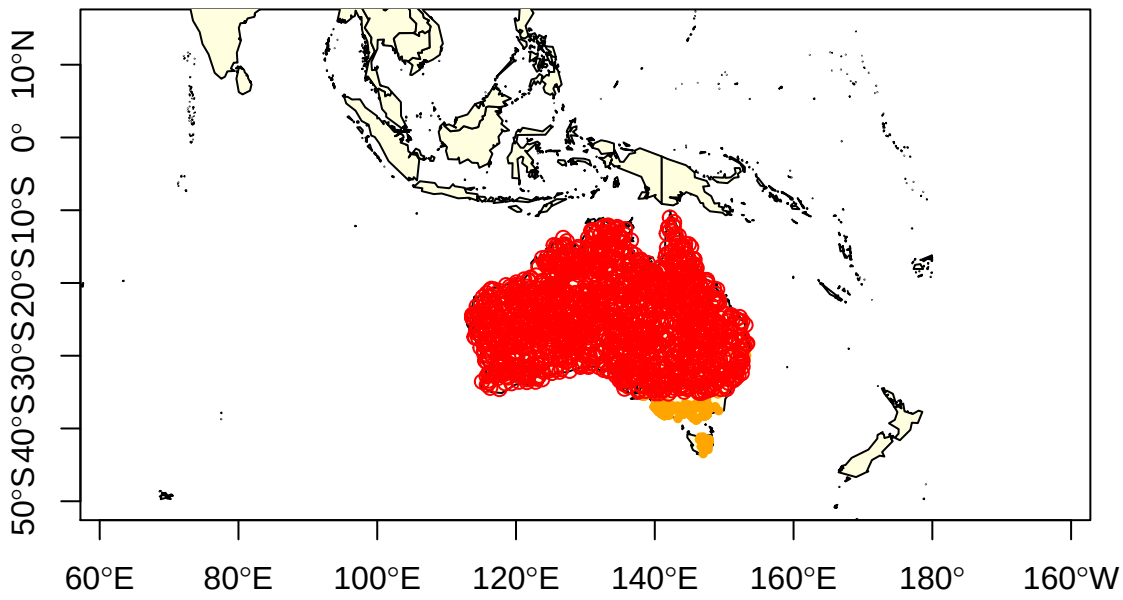



```
# BTreeCreeper.sdm = extractPredictorsValues(BTreeCreeper.c,BTreeCreeper.bg,predictors)
```

```
CommonBeard.c = read.csv("CommonBeardheathC.csv") #Reading the data in; 2 columns containing the longit
CommonBeard.c = CommonBeard.c[,c(2:3)]

CommonBeard.bg_extent = extent(100, 160, sort(CommonBeard.c$lat, TRUE)[7] + 0.5, -10) #Coordinates arou
plot(wrld_simpl, xlim = c(125, 160), ylim= c(-50,-10), axes = TRUE)
abline(h = (sort(CommonBeard.c$lat, TRUE)[7]) + 0.5, col = 'orange')
points(CommonBeard.c, col = 'red')
```

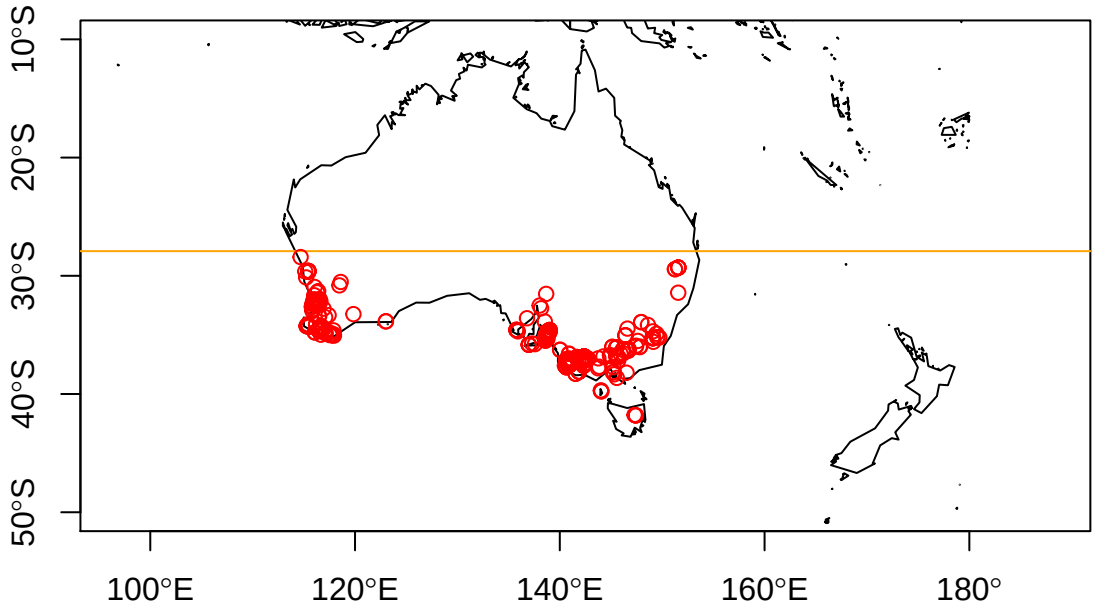



```
#Returns dataframe containing lon/lat values
CommonBeard.bg = generateRandomPoints(CommonBeard.c, CommonBeard.bg_extent)

CommonBeard.extent = getExtent(CommonBeard.c) #extent of observation points

#Visualize these points
#Orange = Prescene, red = pesuedo
plot_(CommonBeard.c, CommonBeard.bg, extent = extent_australia)
```

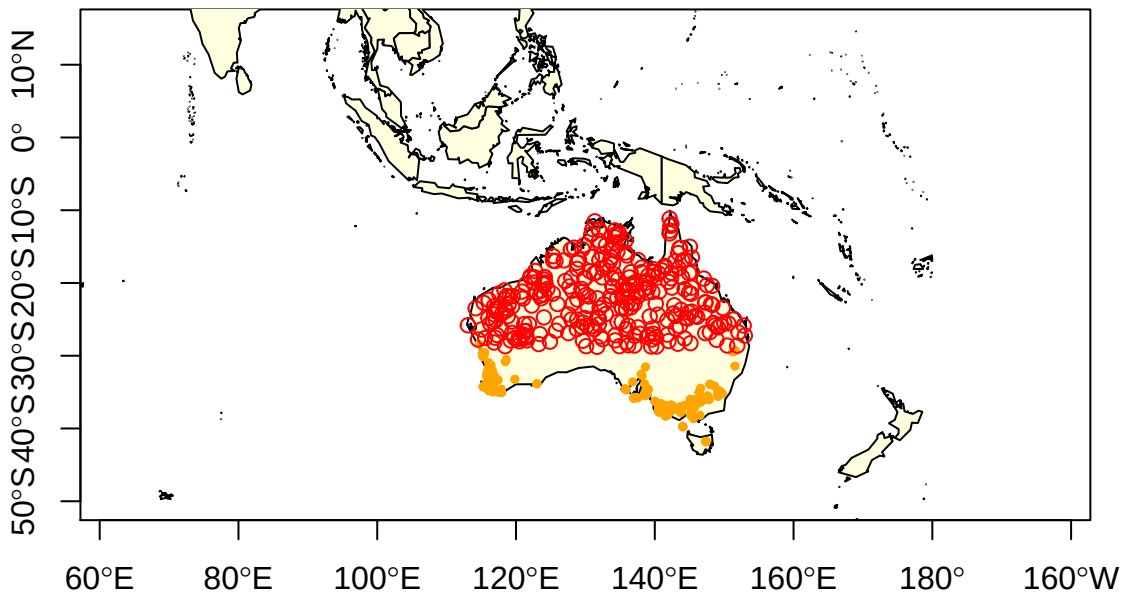



```
# CommonBeard.sdm = extractPredictorsValues(CommonBeard.c,CommonBeard.bg,predictors)
```

```
Smallplant.c = read.csv("SmallTriggerplantC.csv") #Reading the data in; 2 columns containing the longit

Smallplant.c = Smallplant.c[,c(2:3)]

Smallplant.bg_extent = extent(100, 160, sort(Smallplant.c$lat, TRUE)[1] + 0.5, -10) #Coordinates around
plot(wrld_simpl, xlim = c(125, 160), ylim= c(-50,-10), axes = TRUE)
abline(h = (sort(Smallplant.c$lat, TRUE)[1]) + 0.5, col = 'orange')
points(Smallplant.c, col = 'red')
```

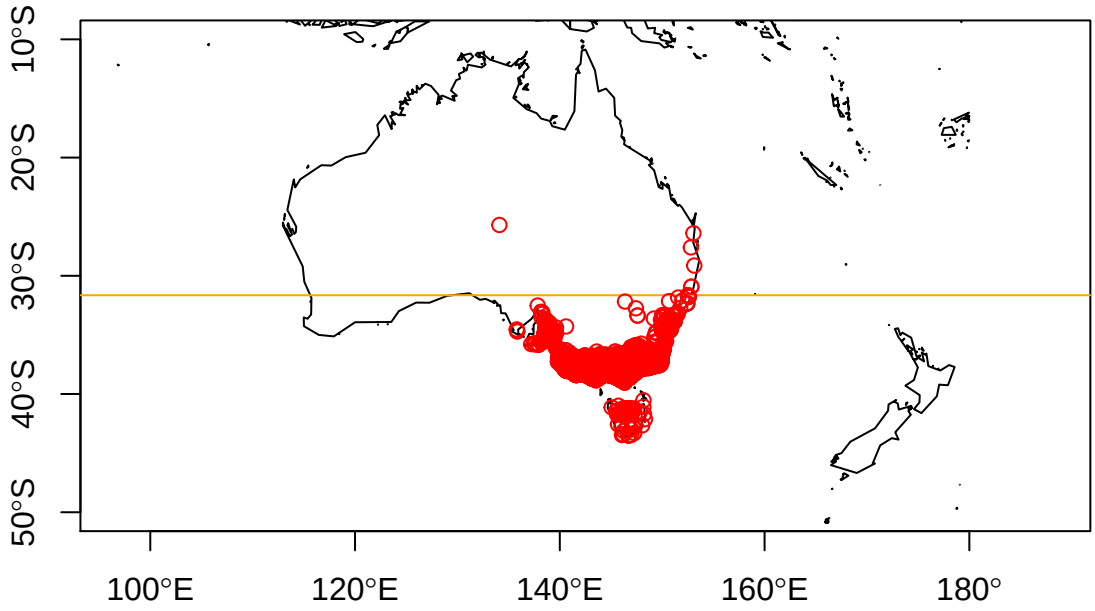



```
#Returns dataframe containing lon/lat values
Smallplant.bg = generateRandomPoints(Smallplant.c, Smallplant.bg_extent)

Smallplant.extent = getExtent(Smallplant.c) #extent of observation points

#Visualize these points
#Orange = Prescene, red = pesuedo
plot_(Smallplant.c, Smallplant.bg, extent = extent_australia)
```

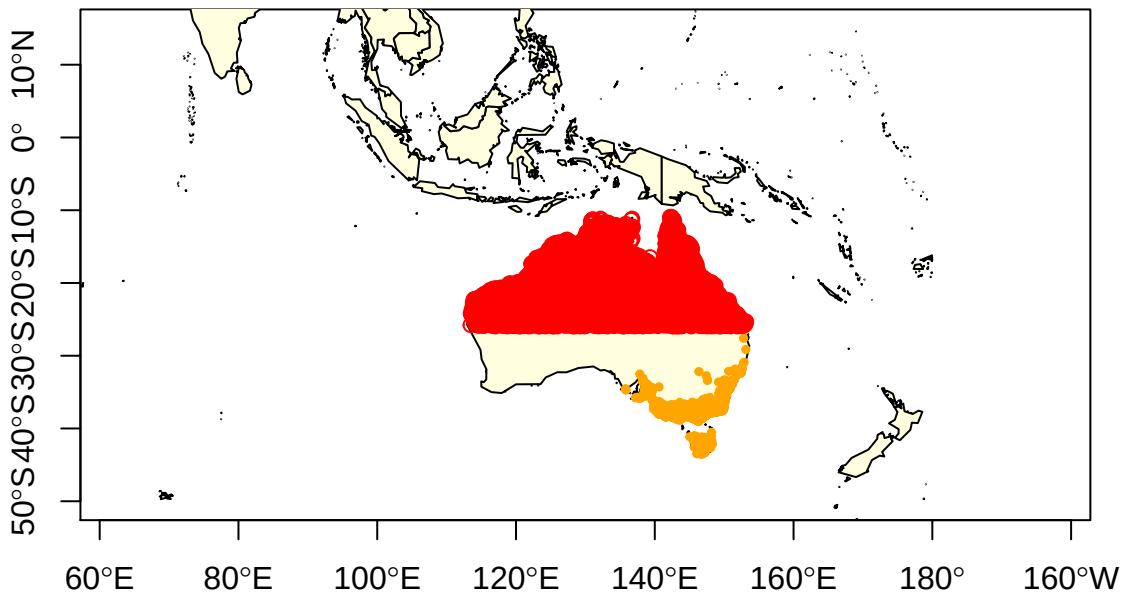



```
# Smallplant.sdm = extractPredictorsValues(Smallplant.c,Smallplant.bg,predictors)
```

```
SouthernFrog.c = read.csv("SouthernBrownTreeFrogC.csv")

SouthernFrog.c = SouthernFrog.c[,c(2:3)]

SouthernFrog.bg_extent = extent(100, 160, sort(SouthernFrog.c$lat, TRUE)[1] + 0.5, -10) #Coordinates ar
plot(wrld_simpl, xlim = c(125, 160), ylim= c(-50,-10), axes = TRUE)
abline(h = (sort(SouthernFrog.c$lat, TRUE)[15]) + 0.5, col = 'orange')
points(SouthernFrog.c, col = 'red')
```

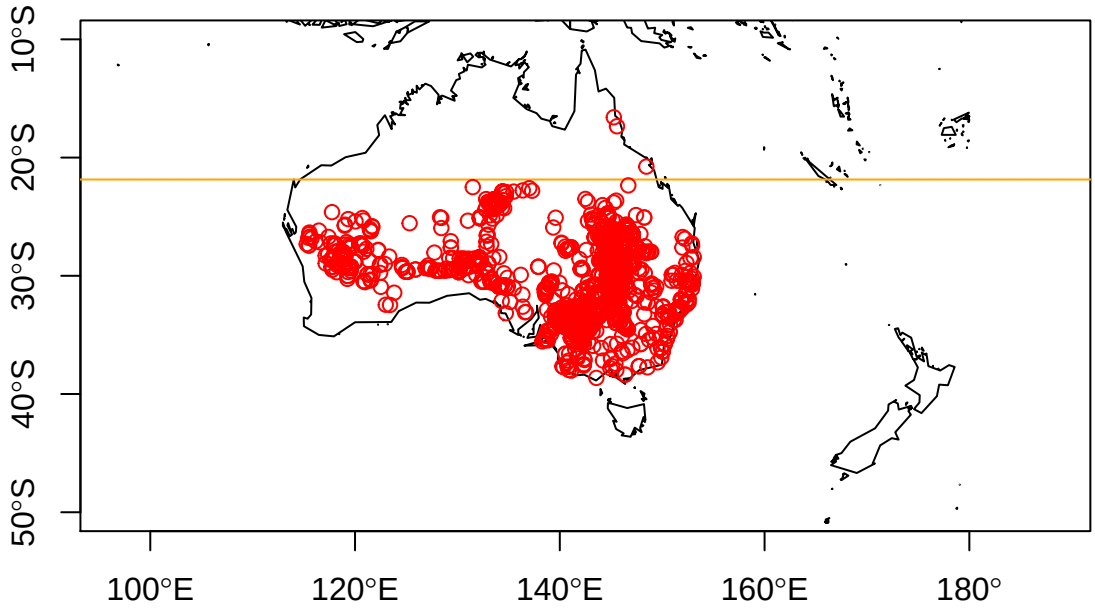



```
#Returns dataframe containing lon/lat values
SouthernFrog.bg = generateRandomPoints(SouthernFrog.c, SouthernFrog.bg_extent)

SouthernFrog.extent = getExtent(SouthernFrog.c) #extent of observation points

#Visualize these points
#Orange = Prescene, red = pesuedo
plot_(SouthernFrog.c, SouthernFrog.bg, extent = extent_australia)
```

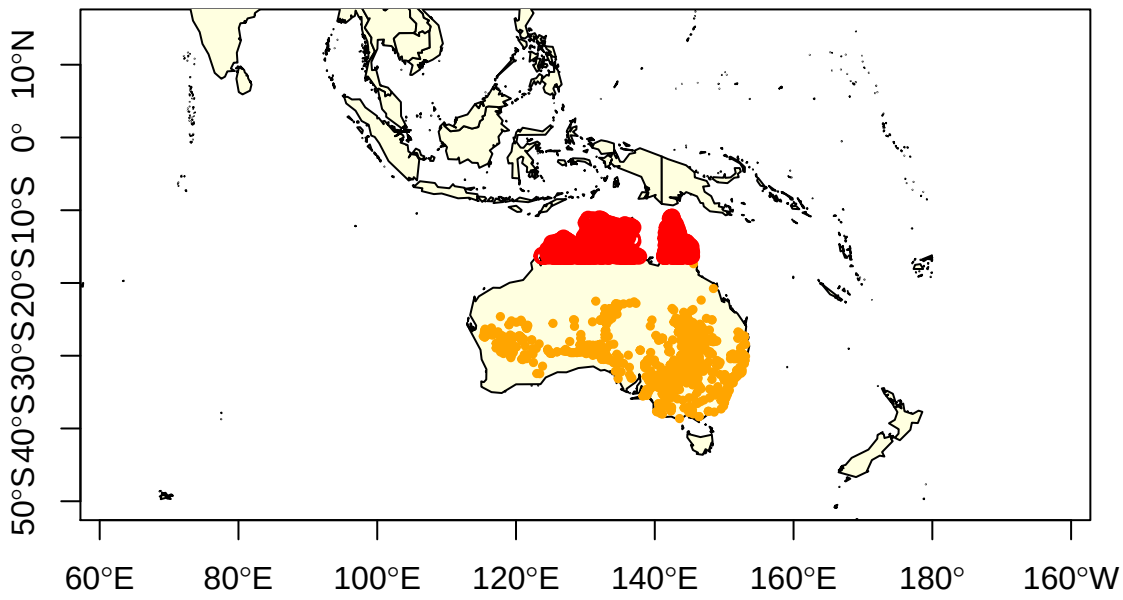



```
# SouthernFrog.sdm = extractPredictorsValues(SouthernFrog.c,SouthernFrog.bg,predictors)
#
```

```
WhiteTreeCreeper.c = read.csv("WhitebrowedTreecreeperC.csv")

WhiteTreeCreeper.c = WhiteTreeCreeper.c[,c(2:3)]

WhiteTreeCreeper.bg_extent = extent(100, 160, sort(WhiteTreeCreeper.c$lat, TRUE)[1] + 0.5, -10) #Coordi
plot(wrld_simpl, xlim = c(125, 160), ylim= c(-50,-10), axes = TRUE)
abline(h = (sort(WhiteTreeCreeper.c$lat, TRUE)[4]) + 0.5, col = 'orange')
points(WhiteTreeCreeper.c, col = 'red')
```




```
#Returns dataframe containing lon/lat values
WhiteTreeCreeper.bg = generateRandomPoints(WhiteTreeCreeper.c, WhiteTreeCreeper.bg_extent)

WhiteTreeCreeper.extent = getExtent(WhiteTreeCreeper.c) #extent of observation points

#Visualize these points
#Orange = Prescene, red = pesuedo
plot_(WhiteTreeCreeper.c, WhiteTreeCreeper.bg, extent = extent_australia)
```




```
# WhiteTreeCreeper.sdm = extractPredictorsValues(WhiteTreeCreeper.c,WhiteTreeCreeper.bg,predictors)
```

```
save.session(file='Modelling.Rda')
```

```
## Warning in save.session(file = "Modelling.Rda"): Open graphics devices will
## not be saved or restored.
```

```
## Saving search path..
## Saving list of loaded packages..
## Saving all data...
## Done.
```

Creating the models - DONE ONE BY ONE FOR EACH SPECIES CAN TAKE SEVERAL HOURS

```
#Read in reliable known points; we are letting model generate psuedo abscenes
species.c = read.csv("speciesC_PA.csv", stringsAsFactors=FALSE)
species.c$X = NULL
species.c_pres = species.c[species.c$reliability == 1,] #only prescene points

curr1 =  species.c_pres[species.c_pres$species == "Agile antechinus",]
curr2 =  species.c_pres[species.c_pres$species == "Brown Treecreeper",]
curr3 =  species.c_pres[species.c_pres$species == "Common Beard-heath",]
curr4 =  species.c_pres[species.c_pres$species == "Small Triggerplant",]
curr5 =  species.c_pres[species.c_pres$species == "Southern Brown Tree Frog",]
curr6 =  species.c_pres[species.c_pres$species == "White-browed Treecreeper",]

#For each species, generate RF model, k-fold, 100 trees
#Refer to name parameter below to see what model it belongs to
```

```
A = suppressWarnings(ensemble_modelling(
  c("RF"),
  Occurrences = curr1,
  Env = crop(predictors, extent_australia),
  Xcol = "lon",
  Ycol = "lat",
  Pcol = NULL,
  save = TRUE,
  rep = 1,
  uncertainty = TRUE,
  cv = "k-fold",
  cv.param = c(5,1),
  name = "Agile antechinus",
  trees = 100
  ))


Bt = suppressWarnings(ensemble_modelling(
  c("RF"),
  Occurrences = curr2,
  Env = crop(predictors, extent_australia),
  Xcol = "lon",
  Ycol = "lat",
  Pcol = NULL,
  save = TRUE,
  rep = 1,
  uncertainty = TRUE,
  cv = "k-fold",
  cv.param = c(5,1),
  name = "Brown Treecreeper",
  trees = 100
  ))


Cb = suppressWarnings(ensemble_modelling(
  c("RF"),
  Occurrences = curr3,
  Env = crop(predictors, extent_australia),
  Xcol = "lon",
  Ycol = "lat",
  Pcol = NULL,
  save = TRUE,
  rep = 1,
  uncertainty = TRUE,
  cv = "k-fold",
  cv.param = c(5,1),
  name = "Common Beard-heath",
  trees = 100
  ))

Sm = suppressWarnings(ensemble_modelling(
  c("RF"),
  Occurrences = curr4,
```

```
  Env = crop(predictors, extent_australia),
  Xcol = "lon",
  Ycol = "lat",
  Pcol = NULL,
  save = TRUE,
  rep = 1,
  uncertainty = TRUE,
  cv = "k-fold",
  cv.param = c(5,1),
  name = "Small Triggerplant",
  trees = 100
  ))

S = suppressWarnings(ensemble_modelling(
  c("RF"),
  Occurrences = curr5,
  Env = crop(predictors, extent_australia),
  Xcol = "lon",
  Ycol = "lat",
  Pcol = NULL,
  save = TRUE,
  rep = 1,
  uncertainty = TRUE,
  cv = "k-fold",
  cv.param = c(5,1),
  name = "Southern Brown Tree Frog",
  trees = 100
  ))


W = suppressWarnings(ensemble_modelling(
  c("RF"),
  Occurrences = curr6,
  Env = crop(predictors, extent_australia),
  Xcol = "lon",
  Ycol = "lat",
  Pcol = NULL,
  save = TRUE,
  rep = 1,
  uncertainty = TRUE,
  cv = "k-fold",
  cv.param = c(5,1),
  name = "White-browed Treecreeper",
  trees = 100
  ))




#Save session into R object, then you reload session ,all models
#will be loaded into memory
save.session("Modelling.Rda")

))
```

Open saved models for each species and stack them into one 'super' model

```
restore.session('Modelling.Rda') #Restoring the session with
#the models loaded in memory


#unique(species.c_pres$species)

#Stacking all 6 ensembles modells for all 6 species into one 'super model'

SSDM = stacking(W,
                S,
                Sm,
                Cb,
                A,
                Bt,
                name = "Choose name")
saveRDS(SSDM, "SSDM")
plot(SSDM)
```