A very brief introduction to species distribution models in R

Jeff Oliver

15 February, 2018

Outline:

- What they are (introductory stuff + description)
- What do we need?
  - Data
    - * Occurrence
    - * Bioclim
  - Dependencies
- Preparing data?
- Quick plot of points on map
- Pseudo-absence points
- Models
  - Quick glance at loadings?
- Maps
  - Do just probability map, without any pseudo-absence points
  - Do threshold map, with pseudo-absence points

[INTRODUCTORY SENTENCE]

Learning objectives

1. Install packages for species distribution modeling
2. Run species distribution models using `bioclim` approach
3. Visualize model predictions on a map

[DESCRIPTION OR MOTIVATION; 2-4 sentences that would be used for an announcement]

Getting started

Before we do anything, we will need to set up our workspace, download example data, and install additional packages that are necessary to run the models and visualize their output.

Workspace organization

So, to start, create a pair of folders in your workspace:

```r
dir.create(path = "data")
dir.create(path = "output")
```

It is good practice to keep input (i.e. the data) and output separate. Furthermore, any work that ends up in the `output` folder should be completely disposable. That is, the combination of data and the code we write should allow us (or anyone else, for that matter) to reproduce any output.

Example data

The data we are working with are observations of the Saguaro, *Carnegiea gigantea*. We are using a subset of records available from GBIF, the Global Biodiversity Information Facility. You can download the data from ; save it in the `data` folder that you created in the step above.

Install additional R packages

Next, there are *five* additional R packages that will need to be installed:

- dismo
- maptools
- rgdal
- raster
- sp

To install these, run:

```r
install.packages("dismo")
install.packages("maptools")
install.packages("rgdal")
install.packages("raster")
install.packages("sp")
```

Ingredient list

The basic idea behind species distribution models is to take two sources of information to model the conditions in which a species is expected to occur. The two sources of information are:

1. Occurrence data: these are usually latitude and longitude geographic coordinates where the species of interest has been observed. These are known as 'presence' data. Some models also make use of 'absence' data, which are geographic coordinates of locations where the species is known to *not* occur. Absence data are a bit harder to come by, but are required by some modeling approaches. For this lesson, we will use the occurrence data of the Saguaro that you downloaded earlier.
2. Environmental data: these are descriptors of the environment, and can include abiotic measurements of temperature and precipitation as well as biotic factors, such as the presence or absence of other species (like predators, competitors, or food sources). In this lesson we will focus on the 19 abiotic variables available from WorldClim. Rather than downloading the data from WorldClim, we'll use functions from the `dismo` package to download these data (see below).

Data and quality control

We'll start our script by loading those five libraries we need. And of course adding a little bit of information at the very top of our script that says what the script does and who is responsible!

```r
# Species distribution modeling for Saguaro
# Jeff Oliver
# jcoliver@email.arizona.edu
# 2018-02-27

library("sp")
library("raster")
library("maptools")
library("rgdal")
library("dismo")
```

There is a good chance you might have seen some red messages print out to the screen, especially when loading the `maptools` or `rgdal` libraries. This is normal, and as long as none of the messages include "ERROR", you can just hum right through those messages. If loading the libraries *does* result in an ERROR message, check to see that the libraries were installed properly.

Now that we have those packages loaded, we can download the bioclimatic variable data with the `getData` function:

```r
bioclim.data <- getData(name = "worldclim",
                        var = "bio",
                        res = 2.5,
                        path = "data/")
```

We're giving `getData` four critical pieces of information:

1. `name = "worldclim"`: This indicates the name of the data set we would like to download
2. `var = "bio"`: This tells `getData` that we want to download all 19 of the bioclimatic variables, rather than individual temperature or precipitation measurements
3. `res = 2.5`: This is the resolution of the data we want to download; in this case, it is 2.5 minutes of a degree. For other resolutions, you can check the documentation by typing `?getData` into the console.
4. `path = "data/"`: Finally, this sets the location to which the files are downloaded. In our case, it is the `data` folder we created at the beginning.

Note also that after the files are downloaded to the `data` folder, the are read into memory and stored in the variable called `bioclim.data`

```r
# Read in saguaro observations
obs.data <- read.csv(file = "data/Carnegiea-gigantea-GBIF.csv")

# Check the data to make sure it loaded correctly
summary(obs.data)
```

```
##      gbifid             latitude       longitude
##  Min.   :2.021e+08   Min.   :26.78   Min.   :-114.0
##  1st Qu.:1.453e+09   1st Qu.:32.17   1st Qu.:-111.4
##  Median :1.571e+09   Median :32.28   Median :-111.1
##  Mean   :1.567e+09   Mean   :32.16   Mean   :-111.3
##  3rd Qu.:1.677e+09   3rd Qu.:32.38   3rd Qu.:-111.0
##  Max.   :1.806e+09   Max.   :34.80   Max.   :-109.3
##                      NA's   :3       NA's   :3
```

Notice that there are three `NA` values in the `latitude` and `longitude` columns. Those records will not be of any use to us, so we can remove them from our data frame:

```r
# Notice NAs - drop them before proceeding
obs.data <- obs.data[!is.na(obs.data$latitude), ]
```

```r
# Make sure those NA's went away
summary(obs.data)
```

```
##      gbifid             latitude       longitude
##  Min.   :8.910e+08   Min.   :26.78   Min.   :-114.0
##  1st Qu.:1.453e+09   1st Qu.:32.17   1st Qu.:-111.4
##  Median :1.571e+09   Median :32.28   Median :-111.1
##  Mean   :1.575e+09   Mean   :32.16   Mean   :-111.3
##  3rd Qu.:1.677e+09   3rd Qu.:32.38   3rd Qu.:-111.0
##  Max.   :1.806e+09   Max.   :34.80   Max.   :-109.3
```

When we look at the `obs.data` data frame now there are no `NA` values, so we are ready to proceed.

To make species distribution modeling more streamlined, it is useful to have an idea of how widely our species is geographically distributed. We are going to find general latitudinal and longitudinal boundaries and store this information for later use:

```r
# Determine geographic extent of our data
max.lat = ceiling(max(obs.data$latitude))
min.lat = floor(min(obs.data$latitude))
max.lon = ceiling(max(obs.data$longitude))
min.lon = floor(min(obs.data$longitude))
geographic.extent <- extent(x = c(min.lon, max.lon, min.lat, max.lat))
```

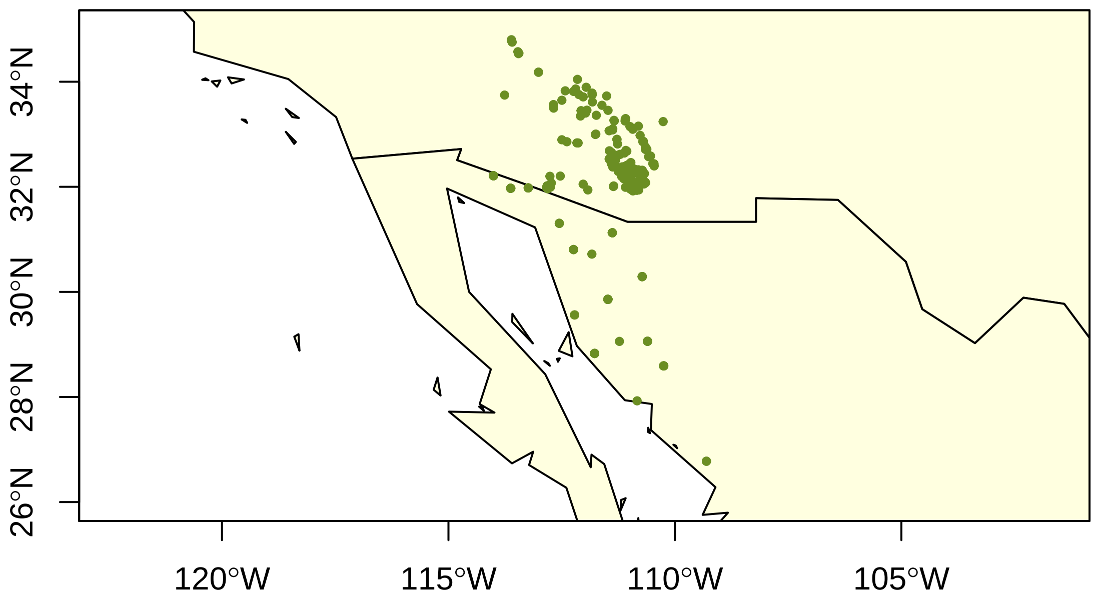
Before we do any modeling, it is also a good idea to run a reality check on your occurrence data by plotting the points on a map.

```r
# Load the data to use for our base map
data(wrld_simpl)

# Plot the base map
plot(wrld_simpl,
     xlim = c(min.lon, max.lon),
     ylim = c(min.lat, max.lat),
     axes = TRUE,
     col = "lightyellow")

# asp = 0 will restrict xlim, but RStudio stretches out the map

# Add the points for individual observation
points(x = obs.data$longitude,
       y = obs.data$latitude,
       col = "olivedrab",
       pch = 20,
       cex = 0.75)
# And draw a little box around the graph
box()
```



Building a model and visualizing results

Now that our occurrence data look OK, we can use the bioclimatic variables to create a model. The first thing we want to do though is limit our consideration to a reasonable geographic area. That is, for our purposes we are not looking to model Saguaro habitat suitability *globally*, but rather to the general southwest region. So we can restrict the biolimatic variable data to the geographic extent of our occurrence data:

```
# Crop bioclim data to geographic extent of saguaro
bioclim.data <- crop(x = bioclim.data, y = geographic.extent)

# Build species distribution model
bc.model <- bioclim(x = bioclim.data, p = obs.data)
```

```
## Error in .xyValues(x, as.matrix(y), ...): xy should have 2 columns only.
## Found these dimensions: 400, 3
```

Uh oh. That's not good. It looks like the data we passed to `bioclim` is not in the right format. The clue comes in the second line of the error message: `## Found these dimensions: 400, 3`. This is referring to the `obs.data` data frame, which does indeed have 400 rows and three columns. From the documentation from `bioclim` (see for yourself via `?bioclim` in the console):

> **Usage**
> bioclim(x, p, ...)
> **Arguments**
> x Raster* object or matrix
> p two column matrix or SpatialPoints* object
> ... Additional arguments

So whatever we pass to `p` should only have **two** columns. Let's modify the `obs.data` so it only has two columns. The first column is the GBIF identifier, which we will not need, so we drop it using the negation operator (i.e. the minus sign). Then we can run the species distribution model.

```
# Drop unused column
obs.data <- obs.data[, -1]
```

```r
# Build species distribution model
bc.model <- bioclim(x = bioclim.data, p = obs.data)
```

```
## Error in bioclim(data.frame(m), ...): insufficient records
```

What the…? OK, this error message is tougher to figure out. But let's consider what our `obs.data` data frame looks like now:

```r
head(obs.data)
```

```
##   latitude longitude
## 1 32.33556 -110.8980
## 2 32.28267 -110.9028
## 3 30.29105 -110.7213
## 4 32.05413 -110.6837
## 5 32.25111 -110.7169
## 6 32.19404 -111.0198
```

The first column is latitude and the second column is longitude, which seems fine. That is, until we think about how R generally deals with coordinates. When we plot something, we generally use syntax like this:

```r
plot(x, y)
```

The thing to note is that the first argument we pass is data for the **x-axis** and the second argument is for the **y-axis**. The `bioclim` function is looking for data in the *same order*. That is, it looks at whatever we passed to `p` and assumes the first column is for the x-axis and the second column is for the y-axis. But our data is in the opposite order: the first column is *latitude*, essentially the y-axis data, and our second column is longitude, corresponding to x-axis data. So we need to reverse the column order before we pass `obs.data` to `bioclim`:

```r
# Reverse order of columns
obs.data <- obs.data[, c(2, 1)]

# Build species distribution model
bc.model <- bioclim(x = bioclim.data, p = obs.data)
```

```r
library("sp") # required by raster
library("raster") # extent
library("maptools") # wrld_simpl
library("rgdal")
library("dismo")

obs.data <- read.csv(file = "data/Carnegiea-gigantea-GBIF.csv")
summary(obs.data)

# Determine geographic extent of our data
max.lat = ceiling(max(obs.data$latitude))
min.lat = floor(min(obs.data$latitude))
max.lon = ceiling(max(obs.data$longitude))
min.lon = floor(min(obs.data$longitude))
geographic.extent <- extent(x = c(min.lon, max.lon, min.lat, max.lat))

# Reality check - plot points
data(wrld_simpl)
# Debugging note: Have to run multiple successive plotting lines all at once for it to work
```

```r
plot(wrld_simpl, xlim = c(min.lon, max.lon), ylim = c(min.lat, max.lat), axes = TRUE, col = "lightyellow
points(obs.data$longitude, obs.data$latitude, col = "olivedrab", pch = 20, cex = 0.75)
box()

# Get the biolim data
bioclim.data <- getData(name = "worldclim",
                        var = "bio",
                        res = 2.5,
                        path = "data/")
bioclim.data <- crop(x = bioclim.data, y = geographic.extent)

# TODO: run this without re-ordering first
# Have to swap order of latitude & longitude, as bioclim method
# expects first column to be longitude, second to be latitude
obs.data <- obs.data[, c(2:1)]

# Do species distribution model and draw a plot
bc.model <- bioclim(x = bioclim.data, p = obs.data)
predict.presence <- predict(x = bioclim.data, object = bc.model, ext = geographic.extent, progress = "")

# TODO: May need some more graphics massaging here
plot(predict.presence, xlim = c(min.lon, max.lon), ylim = c(min.lat, max.lat), main = "BIOCLIM, raw")
plot(wrld_simpl, add = TRUE, border = "dark grey")
points(obs.data$longitude, obs.data$latitude, col = "olivedrab", pch = "+", cex = 0.75)
box()

# Create pseudo-absence points
bil.files <- list.files(path = "data/wc2-5",
                          pattern = "*.bil$",
                          full.names = TRUE)
mask <- raster(bil.files[1])

# Random points (same number as our observed points)
bg <- randomPoints(mask = mask, n = nrow(obs.data), ext = geographic.extent, extf = 1.25)

# Do a quick plot to see the points
plot(wrld_simpl, xlim = c(min.lon, max.lon), ylim = c(min.lat, max.lat), axes = TRUE, col = "light yell
points(bg, col = "grey30", pch = "+", cex = 0.75)
points(obs.data, col = "olivedrab", pch = 20, cex = 0.75)
box()

# Now re-run model with only the training data

# Data for observation sites (presence and background)
presence.values <- extract(x = bioclim.data, y = obs.data)
absence.values <- extract(x = bioclim.data, y = bg)

# Separate training & testing data
group.presence <- kfold(x = obs.data, k = 5) # kfold is in dismo package
testing.group <- 1
presence.train <- obs.data[group.presence != testing.group, ]
presence.test <- obs.data[group.presence == testing.group, ]
group.bg <- kfold(x = bg, k = 5)
```

```r
bg.train <- bg[group.bg != testing.group, ]
bg.test <- bg[group.bg == testing.group, ]

# Do species distribution model and draw a plot
bc.model <- bioclim(x = bioclim.data, p = presence.train)


bc.eval <- evaluate(p = presence.test, a = bg.test, model = bc.model, x = bioclim.data)
bc.threshold <- threshold(x = bc.eval, stat = "spec_sens")
predict.presence <- predict(x = bioclim.data, object = bc.model, ext = geographic.extent, progress = ""
plot(predict.presence > bc.threshold, main = "Presence/Absence")
plot(wrld_simpl, add = TRUE, border = "dark grey")
box()
```

Additional resources

- Vignette for `dismo` package
- Fast and flexible Bayesian species distribution modelling using Gaussian processes
- Species distribution models in R
- Run a range of species distribution models
- SDM polygons on a Google map
- R package 'maxnet' for functionality of Java maxent package
- A study on the effect of pseudo-absences in SDMs (Barbet-Massin et al. 2012)
- A PDF version of this lesson

Back to learn-r main page

Questions? e-mail me at jcoliver@email.arizona.edu.