Species distribution modeling with R

Robert J. Hijmans and Jane Elith

August 13, 2011

Chapter 1

Introduction

This document provides an introduction to species distribution modeling with R . Species distribution modeling (SDM) is also known under other names including climate envelope-modeling, habitat modeling, and (environmental or ecological) niche-modeling. In SDM, the following steps are usually taken: (1) locations of occurrence (and perhaps non-occurrence) of a species (or other phenomenon) are compiled. (2) values of environmental predictor variables (such as climate) at these locations are determined. (3) the environmental values are used to fit a model predicting likelihood of presence, or another measure such as abundance for the species. (4) The model is used to predict the likelihood of presence at all locations of an area of interest (and perhaps in a future climate).

We do not provide a general introduction to species distribution modeling itself. We assume that you are familiar with most of the concepts in this field. If in doubt, you could consult Richard Pearson's introduction to the subject: `http://biodiversityinformatics.amnh.org/index.php?section_id=111`, or the book by Janet Franklin (2009). You can also consult a recent review of the field by Elith and Leathwick (2009). One important concept is the interplay of environmental (niche) and geographic (biotope) space – see Colwell and Rangel (2009) for a good overview. SDM is a widely used approach but there is much debate on when and how to best use this method. This document does not provide a critical discussion of these issues. Rather, our objective is to provide a practical help to allow you to take the basic steps in SDM. We leave it to the reader to use other sources to determine the appropriate methods for their research; and to use the ample opportunities provided by the R environment to improve existing approaches and to develop new ones.

We also assume that you are already familiar with the R language and environment. It would be particularly useful if you already had some experience with statistical model fitting (e.g. the `glm` function) and with the '`raster`' package. If you are not experienced with these, we recommend you first familiarize with these. See, for instance, the Documentation section on the CRAN webpage (http://cran.r-project.org/) and the vignette for the '`raster`' package. When we present code we will give some hints on how to understand the code, if we

think it might be confusing. We will do more of this earlier on in this document, so if you are relatively inexperienced with R and would like to ease into it, read in the presented order.

SDM have been implemented in R in many different ways. Here we focus on the functions in the '`dismo`' and the '`raster`' packages (but we also refer to other packages). If you want to test, or build on, some of the examples presented here, make sure you have the latest versions of these libraries, and their dependencies, installed. If you are using a recent version of R , you can do that with:

```
install.packages(c('raster', 'rgdal', 'dismo', 'rJava'))
```

This document consists of 4 main parts. Part I is concerned with data preparation. This is often the most time consuming part of a species distribution modeling project. You need to collect a sufficient number of occurrence records that document presence (and perhaps absence or abundance) of the species of interest. You also need to have accurate and relevant spatial predictor variables at a sufficiently high spatial resolution. We first discuss some aspects of assembling and cleaning species records, followed by a discussion of aspects of choosing and using the predictor variables. A particularly important concern in species distribution modeling is that the species occurrence data adequately represent the species' distribution. For instance, the species should be correctly identified, the coordinates of the location data need to be accurate enough to allow the general species/environment to be established, and the sample unbiased, or accompanied by information on known biases such that these can be taken into account. Part II introduces the main steps in SDM: fitting a model, making a prediction, and evaluating the result. Part III introduces different modeling methods in more detail (profile methods, regression methods, machine learning methods, and geographic methods). In Part IV we discuss a number of applications (e.g. predicting the effect of climate change), and a number of more advanced topics.

Part I

Data preparation

Chapter 2

Species occurrence data

Importing occurrence data into R is easy. But collecting, georeferencing, and cross-checking coordinate data is tedious. Discussions about species distribution modeling often focus on comparing modeling methods, but if you are dealing with species with few and uncertain records, your focus probably ought to be on improving the quality of the occurrence data. All methods do better if your occurrence data is unbiased and free of error (Graham *et al.*, 2007) and you have a relatively large number of records (Wisz *et al.*, 2008). While we'll show you some useful data preparation steps you can do in R , it is necessary to use additional tools as well. For example, Quantum GIS, `http://www.qgis.org/`, is a very useful program for interactive editing of point data sets.

2.1 Importing occurrence data

In most cases you will have a file with point locality data representing the known distribution of a species. Below is an example of using `read.table` to read records that are stored in a text file. The R commands used are in *italics* and preceded by a '>'. Comments are preceded by a hash (#). We are using an example file that is installed with the '`dismo`' package, and for that reason we use a complex way to construct the filename, but you can replace that with your own filename. (remember to use forward slashes in the path of filenames!). `system.file` inserts the file path to where dismo is installed. If you haven't used the `paste` function before, it's worth familiarizing yourself with it (type `?paste` in the command window). It's very useful.

```
> # loads the dismo library
> library(dismo)
> filename <- paste(system.file(package="dismo"), '/ex/bradypus.csv', sep="")
> # this is the filename we will use:
> filename

[1] "C:/soft/R/R-2.13.1/library/dismo/ex/bradypus.csv"
```

```
> bradypus <- read.table(filename,  header=TRUE,  sep=',')
> # let's inspect the values of the file
> # first rows
> head(bradypus)

              species      lon      lat
1 Bradypus variegatus -65.4000 -10.3833
2 Bradypus variegatus -65.3833 -10.3833
3 Bradypus variegatus -65.1333 -16.8000
4 Bradypus variegatus -63.6667 -17.4500
5 Bradypus variegatus -63.8500 -17.4000
6 Bradypus variegatus -64.4167 -16.0000

> # we only need columns 2 and 3:
> bradypus <- bradypus[,2:3]
> head(bradypus)

       lon      lat
1 -65.4000 -10.3833
2 -65.3833 -10.3833
3 -65.1333 -16.8000
4 -63.6667 -17.4500
5 -63.8500 -17.4000
6 -64.4167 -16.0000
```

You can also read such data directly out of Excel or from a database (see e.g. the `RODBC` package). No matter how you do it, the objective is to get a matrix (or a `data.frame`) with at least 2 columns to hold the coordinates. Coordinates are typically longitude and latitude, but they could also be Easting and Northing in UTM or another coordinate reference system (map projection). The convention used here is to organize the coordinates columns so that longitude is first and latitude the second column (think x and y axes in a graph; longitude is x, latitude is y); they often are in the reverse order, leading to undesired results. In many cases you will have additional columns, e.g., a column to indicate the species if you are modeling multiple species; and a column to indicate whether this is a 'presence' or an 'absence' record (a much used convention is to code presence with a 1 and absence with a 0).

If you do not have any species distribution data you can get started by downloading data from the Global Biodiversity Inventory Facility (GBIF) (`http://www.gbif.org/`). In the dismo package there is a function '`gbif`' that you can use for this. The data used below were downloaded using the `gbif` function like this:

```
acaule = gbif('solanum', 'acaule', geo=FALSE)
```

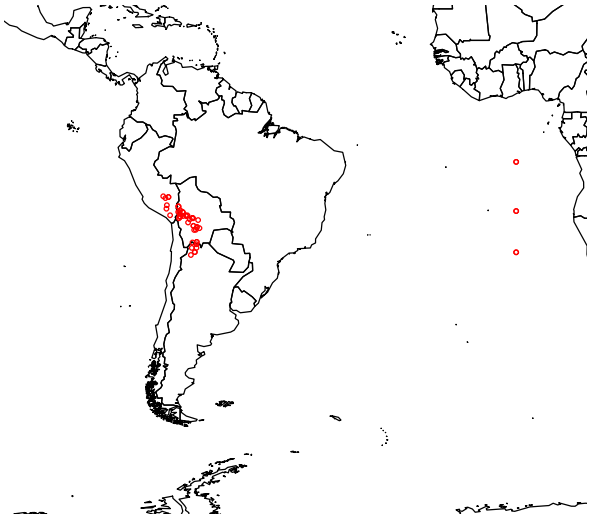
If you want to understand the order of the arguments given here to `gbif` or find out what other arguments you can use with this command, check out the help file (remember you can't access help files if the library is not loaded).

Many records may not have coordinates. Out of the 699 records that gbif returned (March 2010), there were only 54 records with coordinates.

```
> data(acaule)
> dim(acaule)

[1] 699  23

> #select the records that have longitude and latitude data
> acgeo <- subset(acaule, !is.na(lon) & !is.na(lat))
> dim(acgeo)

[1] 54 23

> acgeo[1:4, c(1:5,7:10)]

                  species continent   country   adm1    adm2      lat    lon
13         Solanum acaule      <NA>       BOL   <NA>    <NA> -18.8167 -65.90
426 Solanum acaule Bitter   America Argentina  Jujuy         -22.9000 -66.24
428 Solanum acaule Bitter   America   Bolivia La Paz Pacajes -17.4200 -68.85
429 Solanum acaule Bitter   America   Bolivia La Paz Pacajes -17.1200 -68.77
    coordUncertaintyM  alt
13                 NA 3960
426                NA 4050
428                NA 3811
429                NA 3800
```

Below is a simple way to make a map of the occurrence localities of *Solanum acaule*. It is important to make such maps to assure that the points are, at least roughly, in the right location.

```
> library(maptools)
> data(wrld_simpl)
> plot(wrld_simpl, xlim=c(-100,10), ylim=c(-60,10))
> points(acgeo$lon, acgeo$lat, col='red', cex=0.5)
```



The "wrld_simpl" dataset contains rough country outlines. You can use other datasets of polygons (or lines or points) as well. For example, you can download higher resolution data country and subnational administrative boundaries data with the `getData` function of the `raster` package. You can also read your own shapefile data into `R` using the `readOGR` function in the `rgdal` package or the `readShapePoly` function in the `maptools` package.

2.2 Data cleaning

Data 'cleaning' is particularly important for data sourced from species distribution data warehouses such as GBIF. Such efforts do not specifically gather data for the purpose of species distribution modeling, so you need to understand the data and clean them appropriately, for your application. Here we provide an example.

Solanum acaule is a species that occurs in the higher parts of the Andes mountains of Peru and Bolivia. Do you see any errors on the map? There are three records that have plausible latitudes, but longitudes that are clearly wrong, as they are in the Atlantic Ocean, south of West Africa. It looks like they have a longitude that is zero (because they appear to be exactly South of London). In many data-bases you will find values that are 'zero' where 'no data' was intended. The `gbif` function (with default arguments) sets coordinates that are (0, 0) to `NA`, but not if one of the coordinates is zero. Let's see if we find

them by searching for records with longitudes of zero.

Let's have a look at these records:

```
> lonzero = subset(acgeo, lon==0)
> # show all records, only the first 13 columns
> lonzero[, 1:13]
```

```
                                 species continent country adm1 adm2
544 Solanum acaule Bitter subsp. acaule      <NA>     BOL <NA> <NA>
551 Solanum acaule Bitter subsp. acaule      <NA>     BOL <NA> <NA>
567 Solanum acaule Bitter subsp. acaule      <NA>     PER <NA> <NA>
638 Solanum acaule Bitter subsp. acaule      <NA>     PER <NA> <NA>
640 Solanum acaule Bitter subsp. acaule      <NA>     ARG <NA> <NA>
641 Solanum acaule Bitter subsp. acaule      <NA>     ARG <NA> <NA>
                                           locality        lat lon
544                                           Llave -16.083333   0
551                                           Llave -16.083333   0
567                    km 205 between Puno and Cuzco  -6.983333   0
638                    km 205 between Puno and Cuzco  -6.983333   0
640 between Quelbrada del Chorro and Laguna Colorada -23.716667   0
641 between Quelbrada del Chorro and Laguna Colorada -23.716667   0
    coordUncertaintyM  alt institution collection catalogNumber
544                NA 3900         IPK  WKS 30050        304711
551                NA 3900         IPK         GB     WKS 30050
567                NA 4250         IPK  WKS 30048        304709
638                NA 4250         IPK         GB     WKS 30048
640                NA 3400         IPK  WKS 30027        304688
641                NA 3400         IPK         GB     WKS 30027
```

The records are from Bolivia (BO), Peru (PE) and Argentina (AR), confirming that coordinates are in error (it could have been that the coordinates were correct for a location in the Ocean, perhaps referring to a location a fish was caught rather than a place where *S. acaule* was collected).

2.2.1 duplicate records

Interestingly, another data quality issue is revealed above: each record occurs twice. This could happen because plant samples are often split and send to multiple herbariums. But in this case it seems that the data from IPK are duplicated in the GBIF database. Duplicates can be removed with the *duplicated* function.

To do: provide code for checking for duplicates. Two issues: exact duplicates (lat / long identical) and duplicates on a per grid cell basis.

```
> # which records are duplicates (only considering the first 10 columns)?
> dups <- duplicated(lonzero[, 1:10])
> # remove duplicates
```

```
> lonzero  <-  lonzero[dups, ]
> lonzero[,1:13]
```

```
                               species continent country adm1 adm2
551 Solanum acaule Bitter subsp. acaule      <NA>     BOL <NA> <NA>
638 Solanum acaule Bitter subsp. acaule      <NA>     PER <NA> <NA>
641 Solanum acaule Bitter subsp. acaule      <NA>     ARG <NA> <NA>
                                          locality        lat lon
551                                          Llave -16.083333   0
638                   km 205 between Puno and Cuzco  -6.983333   0
641 between Quelbrada del Chorro and Laguna Colorada -23.716667   0
    coordUncertaintyM  alt institution collection catalogNumber
551                NA 3900         IPK         GB     WKS 30050
638                NA 4250         IPK         GB     WKS 30048
641                NA 3400         IPK         GB     WKS 30027
```

2.3 cross-checking

It is important to cross-check coordinates by visual and other means. One approach is to compare the country (and lower level administrative subdivisions) of the site as specified by the records, with the country implied by the coordinates (Hijmans *et al.*, 1999). In the example below we use the 'coordinates' function from the 'sp' package to create a SpatialPointsDataFrame, and then the 'overlay' function, also from 'sp', to do a point-in-polygon query with the countries polygons.

```
> library(sp)
> # make a SpatialPointsDataFrame
> coordinates(acgeo) = ~lon+lat
> class(acgeo)
```

```
[1] "SpatialPointsDataFrame"
attr(,"package")
[1] "sp"
```

```
> # use the coordinates to do a spatial query of the polygons in wrld_simpl, wich is an
> # object of class SpatialPolygonsDataFrame
> class(wrld_simpl)
```

```
[1] "SpatialPolygonsDataFrame"
attr(,"package")
[1] "sp"
```

```
> ov <- overlay(acgeo, wrld_simpl)
> # ov has, for each point, the record number of wrld_simpl
> # we use the record number to extract the country name
> # (the variable 'NAME' in the data slot of the wrld_simpl)
> names(wrld_simpl@data)
```

```
 [1] "FIPS"      "ISO2"      "ISO3"      "UN"        "NAME"      "AREA"
 [7] "POP2005"   "REGION"    "SUBREGION" "LON"       "LAT"
```

```
> head(wrld_simpl@data)
```

```
    FIPS ISO2 ISO3 UN                NAME   AREA  POP2005 REGION SUBREGION
ATG   AC   AG  ATG 28 Antigua and Barbuda     44    83039     19        29
DZA   AG   DZ  DZA 12             Algeria 238174 32854159      2        15
AZE   AJ   AZ  AZE 31          Azerbaijan   8260  8352021    142       145
ALB   AL   AL  ALB  8             Albania   2740  3153731    150        39
ARM   AM   AM  ARM 51             Armenia   2820  3017661    142       145
AGO   AO   AO  AGO 24              Angola 124670 16095214      2        17
        LON     LAT
ATG -61.783  17.078
DZA   2.632  28.163
AZE  47.395  40.430
ALB  20.068  41.143
ARM  44.563  40.534
AGO  17.544 -12.296
```

```
> cntr <- as.character(wrld_simpl@data$NAME[ov])
> # which points (identified by their record numbers) do not match
> # any country (i.e. are in an ocean)
> i <- which(is.na(cntr))
> i
```

```
[1] 43 44 45 46 47 48
```

```
> # these are the same records, with longitude=0, as identified above:
> acgeo@data[i,'catalogNumber']
```

```
[1] "304711"    "WKS 30050" "304709"    "WKS 30048" "304688"    "WKS 30027"
```

```
> # which points has coordinates that are in a different country than
> #  listed in the 'country' field of the gbif record
> j <- which(cntr != acgeo@data$country)
> j
```

```
[1]  1 49 50 51 52
```

```
> # for the mismatches, bind the country names of the polygons and points
> cbind(cntr, acgeo@data$country)[j,]
```

```
     cntr
[1,] "Bolivia" "BOL"
[2,] "Bolivia" "BOL"
[3,] "Peru"    "PER"
[4,] "Peru"    "PER"
[5,] "Peru"    "PER"
```

```
> # fortunately the mismatch is simply because of the use of abbreviations
> # instead of full country names in these records.
```

See the sp package for more information on the `overlay` function and the related function textttover. At first it may be confusing that it returns indices (row numbers). These indices, stored in variables `i` and `j` were used to get the relevant records. Note that the polygons that we used in the example above are not very precise, and they should not be used in a real analysis (see `http://www.gadm.org/` for more detailed administrative division files, or use the '`getData`' function from the raster package (e.g. `getData('gadm', country='PER', level=0)` to get the national borders of Peru.

```
> # now let's remove the records that have a longitude of 0
> acgeo  <- acgeo[ coordinates(acgeo)[,'lon'] != 0,  ]
```

2.4 Georeferencing

If you have records with locality descriptions but no coordinates, you should consider georeferencing these. Not all the records can be georeferenced. Sometimes even the country is unknown (country=="UNK"). Here we select only records that do not have coordinates, but that do have a locality description.

```
> georef <- subset(acaule, (is.na(lon) | is.na(lat)) & ! is.na(locality) )
> dim(georef)

[1] 89 23

> georef[1:3,1:13]

                                                    species continent country
30 Solanum acaule Bitter subsp. acaule (Juz.) Hawkes & Hjert.      <NA>     PER
42 Solanum acaule Bitter subsp. acaule (Juz.) Hawkes & Hjert.      <NA>     BOL
81 Solanum acaule Bitter subsp. acaule (Juz.) Hawkes & Hjert.      <NA>     ARG
   adm1 adm2                      locality lat lon coordUncertaintyM  alt
30 <NA> <NA> km 205 between Puno and Cuzco  NA  NA                NA 4250
42 <NA> <NA>                         Llave  NA  NA                NA 3900
81 <NA> <NA>                       da Pena  NA  NA                NA   NA
   institution collection catalogNumber
30      DEU159        DEU     WKS 30048
42      DEU159        DEU     WKS 30050
81      DEU159        DEU     WKS 30417
```

Among the first records is an old acquaintance. The record, with catalog number WKS 30048 was also in the set of records that had a longitude of zero degrees.

We recommend using a tool like BioGeomancer: `http://bg.berkeley.edu/latest` (Guralnick *et al.*, 2006) to georeference textual locality descriptions. An

important feature of BioGeomancer is that it attempts to capture the uncertainty associated with each georeference (Wieczorek *et al.*, 2004). The dismo package has a function `biogeomancer` that you can use for this, and that we demonstrate below, but its use is generally not recommended because you really need a detailed map interface for accurate georeferencing.

Here is an example for one of the records with longitude = 0. We put the biogeomancer function into a 'try' function, to assure elegant error handling if the computer is not connected to the Internet.

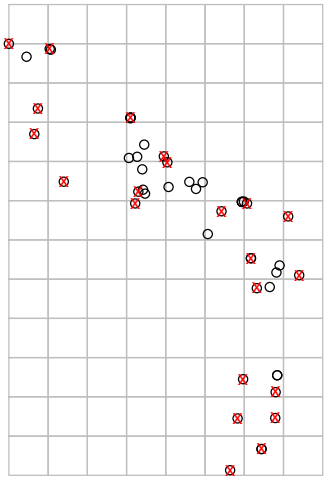
```
> args(biogeomancer)

function (country = "", adm1 = "", adm2 = "", locality = "",
    singleRecord = TRUE, progress = "text")
NULL

> # the first locality:
> lonzero$locality[1]

[1] "Llave"

> b <-  try(  biogeomancer('Peru', locality=lonzero$locality[1], progress='')  )
> b

  id lon lat coordUncertaintyM
1  1  NA  NA                NA

> # there was no longitude in the downloaded data, but
> # is the latitude similar to what we had?
> lonzero$lat[1]

[1] -16.08333
```

2.5 Sampling bias

Sampling bias is frequently present in occurrence records (e.g., Hijmans *et al.*, 2001). One can attempt to remove some of the bias by subsampling records, and this is illustrated below. However, subsampling reduces the number of records, and it cannot correct the data for areas that have not been sampled at all. It also suffers from the problem that locally dense records might be a true reflection of the relative suitable of habitat. See Phillips *et al.*, 2009) for an approach with MaxEnt to deal with bias in occurrence records used in SDM. The example below illustrates how one could go about subsampling. This is not a general recommendation to subsample, or to subsample in this way.

```
> # create a RasterLayer with the extent of acgeo
> r <- raster(acgeo)
> # set the resolution of the cells to 1 degrees
```

```
> res(r) <- 1
> # expand the RasterLayer a little
> r <- expand(r, extent(r)+1)
> # get the cell number for each point
> cell <- cellFromXY(r, acgeo)
> dup <- duplicated(cell)
> # select the records that are not duplicated
> # a random selection within duplicates might be better (but more elaborate)
> acsel <- acgeo[!dup, ]
> # display the results
> p <- rasterToPolygons(r)
> plot(p, border='gray')
> points(acgeo)
> # selected points in red
> points(acsel, cex=1, col='red', pch='x')
```



Chapter 3

Absence and background points

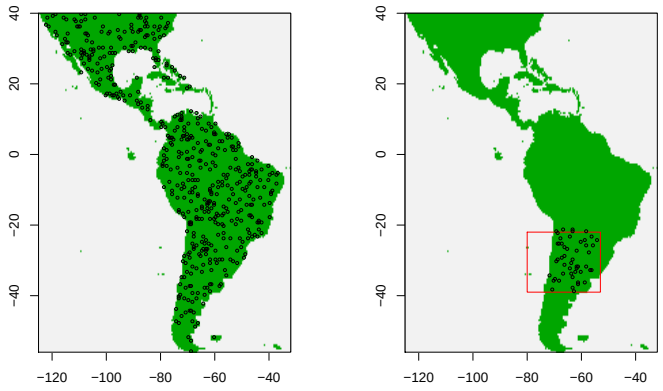
Some of the early species distribution model algorithms, such as Bioclim and Domain only use 'presence' data in the modeling process. Other methods also use 'absence' data or 'background' data. Logistic regression is the classical approach to analyzing presence and absence data (and it is still much used, often implemented in a generalized linear modeling (GLM) framework). If you have a large dataset with presence/absence from a well designed survey, you should use a method that can use these data (i.e. do not use a modeling method that only considers presence data). If you only have presence data, you can still use a method that needs absence data, by substituting absence data with background data.

Background data (e.g. Phillips *et al.* 2009) are not attempting to guess at absence locations, but rather to characterize environments in the study region. In this sense, background is the same, irrespective of where the species has been found. Background data establishes the environmental domain of the study, whilst presence data should establish under which conditions a species is more likely to be present than on average. A closely related but different concept, that of "pseudo-absences", is also used for generating the non-presence class for logistic models. In this case, researchers sometimes try to guess where absences might occur – they may sample the whole region except at presence locations, or they might sample at places unlikely to be suitable for the species. We prefer the background concept because it requires fewer assumptions and has some coherent statistical methods for dealing with the "overlap" between presence and background points (e.g. Ward et al. 2009; Phillips and Elith, 2011). 'True' absence data has value. In conjunction with presence records, it establishes where surveys have been done, and the prevalence of the species given the survey effort. That information is lacking for presence-only data, a fact that can cause substantial difficulties for modeling presence-only data well. However, absence data can also be biased and incomplete, as discussed in the

literature on detectability (e.g., Kéry et al., 2010).

`dismo` has a function to sample random points (background data) from a study area. You can use a 'mask' to exclude area with no data `NA`, e.g. areas not on land. You can use an 'extent' to further restrict the area from which random locations are drawn. In the example below, we first get the list of filenames with the predictor raster data (discussed in detail in the next chapter) use that as a mask such that the background points are from the same geographic area, and only for places where there are values (land, in our case).

```
> files <- list.files(path=paste(system.file(package="dismo"), '/ex', sep=''),
+                     pattern='grd', full.names=TRUE )
> mask <- raster(files[[1]])
> # select 500 random points
> bg <- randomPoints(mask, 500 )
> # set up the plotting area for two maps
> par(mfrow=c(1,2))
> plot(!is.na(mask), legend=FALSE)
> points(bg, cex=0.5)
> # now limiting the area of sampling by spatial extent
> e <- extent(-80, -53, -39, -22)
> bg2 <- randomPoints(mask, 50, ext=e)
> plot(!is.na(mask), legend=FALSE)
> plot(e, add=TRUE, col='red')
> points(bg2, cex=0.5)
```



To do: add example for sampling within radius of presence points (VanDerWal et al., 2009) . Mention sampling according to cell area (Elith et al., 2011)

Chapter 4

Environmental data

4.1 Raster data

In species distribution modeling, predictor variables are typically organized as raster (grid) type files. Each predictor should be a 'raster' representing a variable of interest. Variables can include climatic, soil and terrain, vegetation, land use, and other variables. These data are typically stored in files in some kind of GIS format. Almost all relevant formats can be used (including ESRI grid, geoTiff, netCDF, IDRISI, and ASCII). Avoid ASCII files if you can, as they tend to considerably slow down processing speed. For any particular study the layers all should have the same spatial extent, resolution, origin, and projection. If necessary, use functions like `crop, expand, aggregate, resample, and projectRaster` from the '`raster`' package to prepare your predictor variable data. See the help files and the vignette of the raster package for more info on how to do this. The set of predictor variables (rasters) can be used to make a '`RasterStack`', which is a collection of '`RasterLayer`' objects (see the `raster` package for more info).

Here we make a list of files that are installed with the dismo package and then create a rasterStack from these, show the names of each layer, and finally plot them all.

```
> files <- list.files(path=paste(system.file(package="dismo"),
+                          '/ex', sep=''), pattern='grd', full.names=TRUE )
> # The above finds all the files with extension "grd" in the examples
> # ("ex") directory of the dismo package. You do not need such a complex
> # statement to get your own files.
> files

[1] "C:/soft/R/R-2.13.1/library/dismo/ex/bio1.grd"
[2] "C:/soft/R/R-2.13.1/library/dismo/ex/bio12.grd"
[3] "C:/soft/R/R-2.13.1/library/dismo/ex/bio16.grd"
[4] "C:/soft/R/R-2.13.1/library/dismo/ex/bio17.grd"
```

```
[5] "C:/soft/R/R-2.13.1/library/dismo/ex/bio5.grd"
[6] "C:/soft/R/R-2.13.1/library/dismo/ex/bio6.grd"
[7] "C:/soft/R/R-2.13.1/library/dismo/ex/bio7.grd"
[8] "C:/soft/R/R-2.13.1/library/dismo/ex/bio8.grd"
[9] "C:/soft/R/R-2.13.1/library/dismo/ex/biome.grd"
```

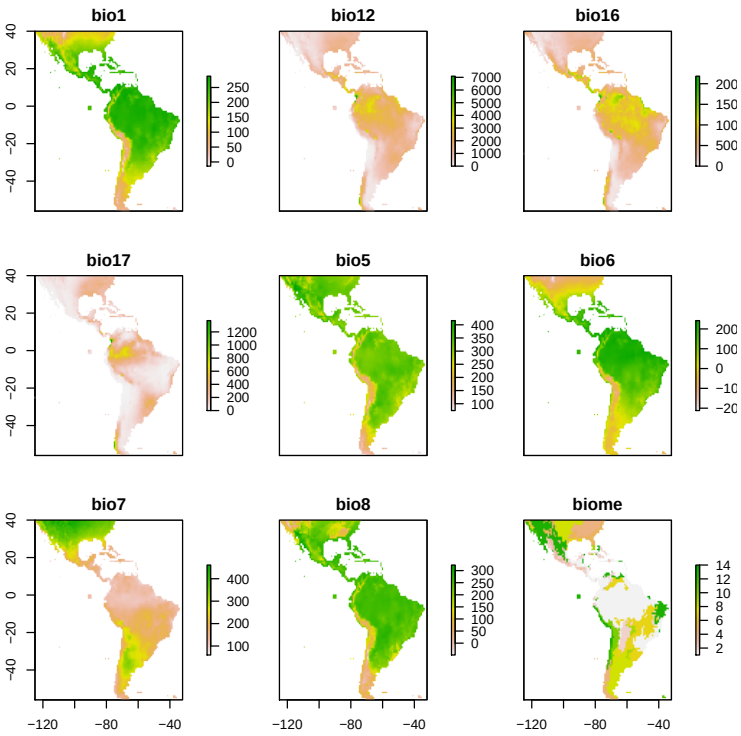
```
> predictors <- stack(files)
> predictors
```

```
class       : RasterStack
dimensions  : 192, 186, 9  (nrow, ncol, nlayers)
resolution  : 0.5, 0.5  (x, y)
extent      : -125, -32, -56, 40  (xmin, xmax, ymin, ymax)
coord. ref. : +proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0
min values  :  -23    0    0    0   61 -212   60  -66    1
max values  :  289 7682 2458 1496  422  242  461  323   14
```

```
> layerNames(predictors)
```

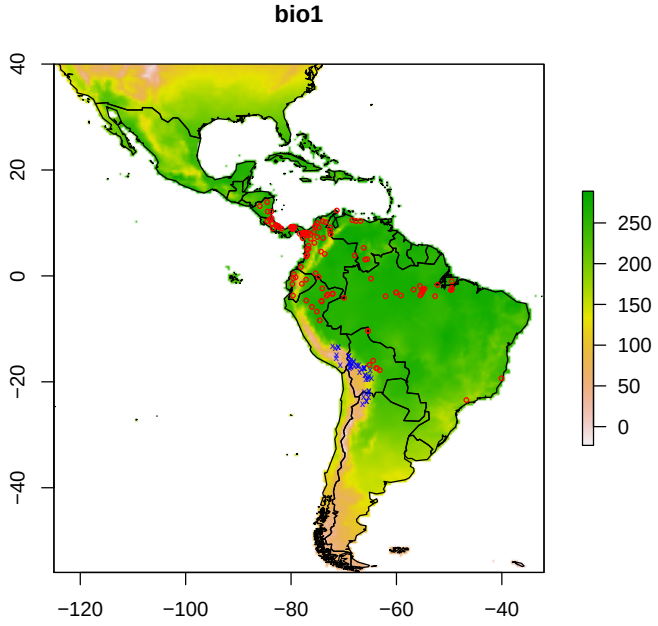
```
[1] "bio1"  "bio12" "bio16" "bio17" "bio5"  "bio6"  "bio7"  "bio8"  "biome"
```

```
> plot(predictors)
```



We can also make a plot of a single layer in a RasterStack, and plot some additional data on top of it:

```
> plot(predictors, 1)
> plot(wrld_simpl, add=TRUE)
> points(bradypus, col='red', cex=0.5)
> points(acgeo, col='blue', pch='x', cex=0.5)
```



The example above uses data representing 'bioclimatic variables' from the WorldClim database (`http://www.worldclim.org`, Hijmans *et al.*, 2004) and 'terrestrial biome' data from the WWF. (`http://www.worldwildlife.org/science/data/item1875.html`, Olsen *et al.*, 2001). You can go to these websites if you want higher resolution data. You can also use the `getData` function from the `raster` package to download WorldClim climate data. Predictor variable selection can be important, particularly if the objective of a study is explanation. See, e.g., Austin and Smith (1987), Austin (2002), Mellert *et al.*, (2011). The early applications of species modeling tended to focus on explanation (Elith and Leathwick 2009). Nowadays, the objective of SDM tends to be prediction. For prediction within the same geographic area, variable selection might arguably be relatively less important, but for many prediction tasks (e.g. to new times or places, see below) variable selection is critically important. In all cases it is important to use variables that are relevant to the ecology of the species (rather than with the first data that can be found on the web!). In some cases it can be useful to develop new, more ecologically relevant, predictor variables from existing data. For example, one could use land cover data and the `focal` function in the `raster` package to create a new variable that indicates

how much forest area is available within x km of a grid cell, for a species that might have a home range of x.

4.2 Extracting values from rasters

We now have a set of predictor variables (rasters) and occurrence points. The next step is to extract the values of the predictors at the locations of the points. (This step can be skipped for the modeling methods that are implemented in the dismo package). This is a very straightforward thing to do using the 'extract' function from the raster package. In the example below we use that function first for the *Bradypus* occurrence points, then for 500 random background points. We combine these into a single `data.frame` in which the first column (variable 'pb') indicates whether this is a presence or a background point. 'biome' is categorical variable (called a 'factor' in R) and it is important to explicitly define it that way (so that it won't be treated like any other numerical variable).

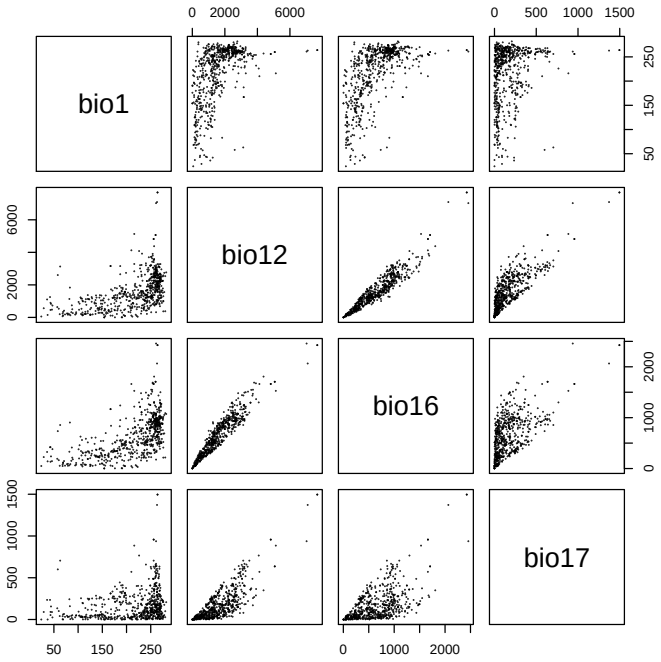
```
> presvals <- extract(predictors, bradypus)
> backgr <- randomPoints(predictors, 500)
> absvals <- extract(predictors, backgr)
> pb <- c(rep(1, nrow(presvals)), rep(0, nrow(absvals)))
> sdmdata <- data.frame(cbind(pb, rbind(presvals, absvals)))
> sdmdata[,'biome'] = as.factor(sdmdata[,'biome'])
> head(sdmdata)

  pb bio1 bio12 bio16 bio17 bio5 bio6 bio7 bio8 biome
1  1  263  1639   724    62  338  191  147  261     1
2  1  263  1639   724    62  338  191  147  261     1
3  1  253  3624  1547   373  329  150  179  271     1
4  1  243  1693   775   186  318  150  168  264     1
5  1  243  1693   775   186  318  150  168  264     1
6  1  252  2501  1081   280  326  154  172  270     1

> tail(sdmdata)

    pb bio1 bio12 bio16 bio17 bio5 bio6 bio7 bio8 biome
611  0  179   996   345   139  320   64  256  236     9
612  0  240  1647   855    20  313  157  156  244     1
613  0  197  1565   448   320  330   44  285  268     7
614  0  223    42    31     0  311  149  163  252    13
615  0   47   732   381    25  120  -54  174   57    13
616  0  242   982   374    93  349  131  218  276     2

> summary(sdmdata)

       pb              bio1           bio12            bio16
 Min.   :0.0000   Min.   : 24.0   Min.   :   2.0   Min.   :   2.0
 1st Qu.:0.0000   1st Qu.:186.0   1st Qu.: 835.2   1st Qu.: 330.5
```

```
Median :0.0000   Median :242.0   Median :1440.5   Median : 617.0
Mean   :0.1883   Mean   :216.5   Mean   :1616.3   Mean   : 648.7
3rd Qu.:0.0000   3rd Qu.:261.0   3rd Qu.:2266.8   3rd Qu.: 919.0
Max.   :1.0000   Max.   :282.0   Max.   :7682.0   Max.   :2458.0

    bio17             bio5            bio6             bio7
Min.   :   0.0   Min.   :110.0   Min.   :-158.00   Min.   : 81
1st Qu.:  40.0   1st Qu.:303.0   1st Qu.:  56.75   1st Qu.:118
Median : 116.0   Median :319.0   Median : 157.00   Median :157
Mean   : 171.9   Mean   :309.8   Mean   : 123.85   Mean   :186
3rd Qu.: 242.0   3rd Qu.:333.0   3rd Qu.: 202.00   3rd Qu.:230
Max.   :1496.0   Max.   :402.0   Max.   : 231.00   Max.   :432

     bio8           biome
Min.   :-16.0   1      :290
1st Qu.:219.8   13     : 71
Median :251.5   7      : 67
Mean   :228.8   2      : 56
3rd Qu.:262.0   8      : 49
Max.   :311.0   5      : 28
                (Other): 55
```

There are alternative approaches possible here. For example, one could extract multiple points in a radius as a potential means for dealing with mismatch between location accuracy and grid cell size. If one would make 10 datasets that represent 10 equally valid "samples" of the environment in that radius, that could be then used to fit 10 models and explore the effect of uncertainty in location.

To visually investigate colinearity in the environmental data (at the occurrence points) you can use a pairs plot. See Dormann et al. (2011) for a discussion of methods to remove colinearity.

```
> # pairs plot of the values of the climate data at the bradypus occurrence sites.
> pairs(sdmdata[,2:5], cex=0.1, fig=TRUE)
```


bio1
bio12
bio16
bio17
0 2000 6000
0 500 1000 1500
50 150 250
0 2000 6000
0 1000 2000
0 500 1000 1500
50 150 250
0 1000 2000

Part II

Model fitting, prediction, and evaluation

Chapter 5

Model fitting

Model fitting is technically quite similar across the modeling methods that exist in R . Most methods take a '`formula`' identifying the dependent and independent variables, accompanied with a `data.frame` that holds these variables. Details on specific methods are provided further down on this document, in the sections on specific modeling methods.

A simple formula could look like: `y ~ x1 + x2 + x3`, i.e. y is a function of x1, x2, and x3. Another example is `y ~ .`, which means that y is a function of all other variables in the `data.frame` provided to the function. See `help('formula')` for more details about the formula syntax. In the example below, the function '`glm`' is used to fit generalized linear models. `glm` returns a model object.

```
> m1 = glm(pb ~ bio1 + bio5 + bio12, data=sdmdata)
> class(m1)

[1] "glm" "lm"

> summary(m1)

Call:
glm(formula = pb ~ bio1 + bio5 + bio12, data = sdmdata)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.66962  -0.22695  -0.10583   0.08412   0.90126

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.760e-01  1.116e-01   1.577 0.115406
bio1         1.563e-03  4.101e-04   3.811 0.000153 ***
bio5        -1.676e-03  4.888e-04  -3.428 0.000648 ***
bio12        1.196e-04  1.662e-05   7.195 1.84e-12 ***
```

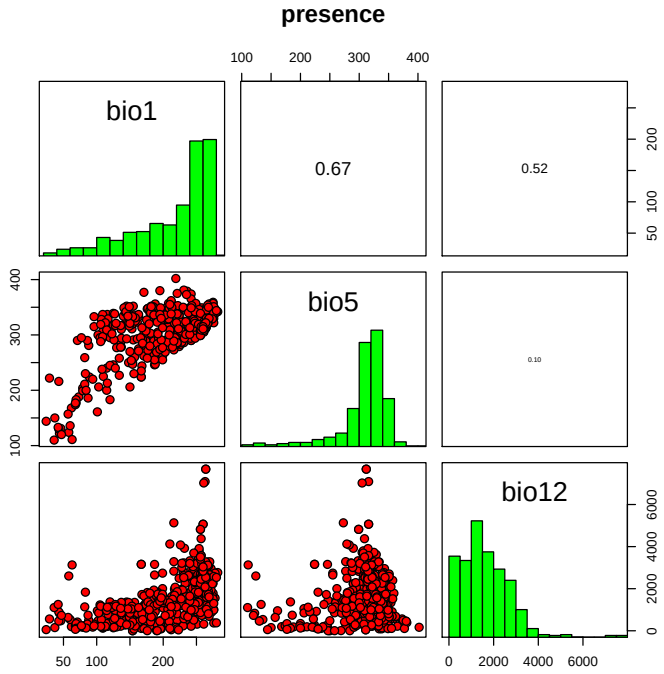
```
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.1217365)

    Null deviance: 94.156  on 615  degrees of freedom
Residual deviance: 74.503  on 612  degrees of freedom
AIC: 456.89

Number of Fisher Scoring iterations: 2

> m2 = glm(pb ~ ., data=sdmdata)
> m2

Call:  glm(formula = pb ~ ., data = sdmdata)

Coefficients:
(Intercept)         bio1        bio12        bio16        bio17         bio5
  0.2884444   -0.0029854    0.0004096   -0.0005504   -0.0008353   -0.0024214
       bio6         bio7         bio8       biome2       biome3       biome4
  0.0043807    0.0023776    0.0008630   -0.1315969   -0.0702524   -0.0951886
     biome5       biome7       biome8       biome9      biome10      biome12
 -0.0642400   -0.2172687   -0.0509056    0.0144005   -0.0525539   -0.0599098
    biome13      biome14
  0.0075382    0.4534979

Degrees of Freedom: 615 Total (i.e. Null);  596 Residual
Null Deviance:            94.16
Residual Deviance: 68.73         AIC: 439.2
```

Models that are implemented in dismo do not use a formula (and most models only take presence points). For example:

```
> bc = bioclim(sdmdata[,c('bio1', 'bio5', 'bio12')])
> class(bc)

[1] "Bioclim"
attr(,"package")
[1] "dismo"

> bc

class    : Bioclim

variables: bio1 bio5 bio12
```

```
presence points: 616
   bio1 bio5 bio12
1   263  338  1639
2   263  338  1639
3   253  329  3624
4   243  318  1693
5   243  318  1693
6   252  326  2501
7   240  317  1214
8   275  335  2259
9   271  327  2212
10  274  329  2233
  (... ...  ...)
```

```
> pairs(bc)
```



Chapter 6

Model prediction

Different modeling methods return different type of 'model' objects (typically they have the same name as the modeling method used). All of these 'model' objects, irrespective of their exact class, can be used to with the `predict` function to make predictions for any combination of values of the independent variables. This is illustrated in the example below where we make predictions with model object 'm1' for three records with values for variables bio1, bio5 and bio12 (the variables used in the example above to create object m1)

```
> bio1 = c(40, 150, 200)
> bio5 = c(60, 115, 290)
> bio12 = c(600, 1600, 1700)
> pd = data.frame(cbind(bio1, bio5, bio12))
> pd

  bio1 bio5 bio12
1   40   60   600
2  150  115  1600
3  200  290  1700

> predict(m1, pd)

        1         2         3
0.2097133 0.4090173 0.2058267

> predict(bc, pd)

[1] 0.000000000 0.006493506 0.347402597
```

Chapter 7

Model evaluation

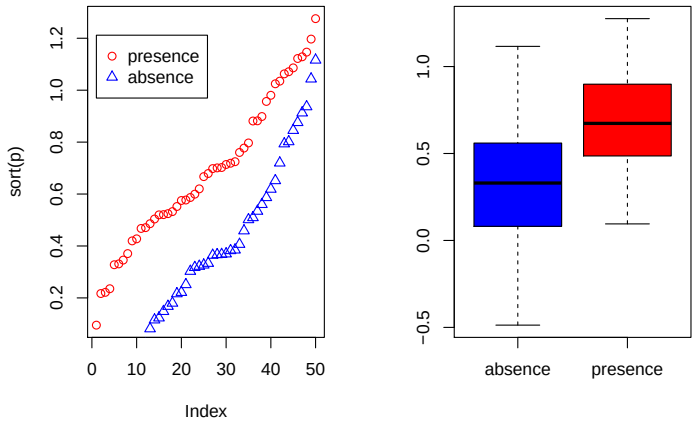
Many model types have measures that help you to assess model fit. It is worth becoming familiar with these and understanding their role, because they help you to assess whether there is anything substantially wrong with your model. Most statistics or machine learning texts will provide some details. For instance, for a GLM one can look at how much deviance is explained, whether there are patterns in the residuals, whether there are points with high leverage and so on. However, since many models are to be used for prediction, much evaluation is focused on how well the model predicts to points not used in model training (see following section on data partitioning). Before we start to give some examples of statistics used for this evaluation, it is worth considering what else can be done to evaluate a model. Useful questions include: • does the model seem sensible, ecologically? • do the fitted functions (the shapes of the modeled relationships) make sense? • do the predictions seem reasonable? (map them, and think about them) • are there any spatial patterns in model residuals? (see Leathwick and Whitehead 2001 for an interesting example)

Most modelers rely on cross-validation. This consists of creating a model with one 'training' data set, and testing it with another data set of known occurrences. Typically, training and testing data are created through random sampling (without replacement) from a single data set. Only in a few cases, e.g. Elith *et al.*, 2006, training and test data are from different sources and predefined. Different measures can be used to evaluate the quality of a prediction (Fielding and Bell, 1997, Liu et al., 2011; and Potts and Elith (2006) for abundance data), perhaps depending on the goal of the study. Many measures for evaluating models based on presence-absence or presence-only data are 'threshold dependent'. That means that a threshold must be set first (e.g., 0.5, though 0.5 is rarely a sensible choice – e.g. see Lui et al. 2005). Predicted values above that threshold indicate a prediction of 'presence', and values below the threshold indicate 'absence'. Some measures emphasize the weight of false absences; others give more weight to false presences. Cohen's *kappa* is an example of a threshold dependent model evaluation statistic.

Much used statistics that are threshold independent are the correlation co-

efficient and the Area Under the Receiver Operator Curve (AUROC, generally further abbreviated to AUC). AUC is a measure of rank-correlation. If it is high, it indicates that high predicted scores tend to be areas of known presence and locations with lower model prediction scores tend to areas where the species is known to be absent (or a random point). An AUC score of 0.5 means that the model is as good as a random guess. Below we illustrate the computation of the correlation coefficient, AUC with two random variables. p (presence) represents the predicted value for 50 known cases (locations) where the species is present, and a (absence) represents the predicted value for 50 known cases (locations) where the species is absent.

```
> p <- rnorm(50, mean=0.7, sd=0.3)
> a <- rnorm(50, mean=0.4, sd=0.4)
> par(mfrow=c(1, 2))
> plot(sort(p), col='red', pch=21)
> points(sort(a), col='blue', pch=24)
> legend(1, 0.95 * max(a,p), c('presence', 'absence'), pch=c(21,24), col=c('red', 'blue'))
> comb = c(p,a)
> group = c(rep('presence', length(p)), rep('absence', length(a)))
> boxplot(comb~group, col=c('blue', 'red'))
```



We created two variables with random normally distributed values, but with different mean and standard deviation. The two variables clearly have different distributions, and the values for 'presence' tend to be higher than for 'absence'. Below we compute the correlation coefficient and the AUC:

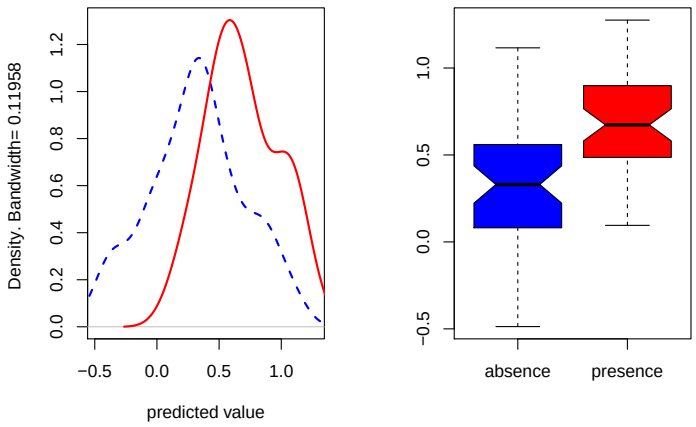
```
> group = c(rep(1, length(p)), rep(0, length(a)))
> cor.test(comb, group)$estimate
```

```
      cor
0.4763127
```

```
> mv <- wilcox.test(p,a)
> auc <- as.numeric(mv$statistic) / (length(p) * length(a))
> auc

[1] 0.7756
```

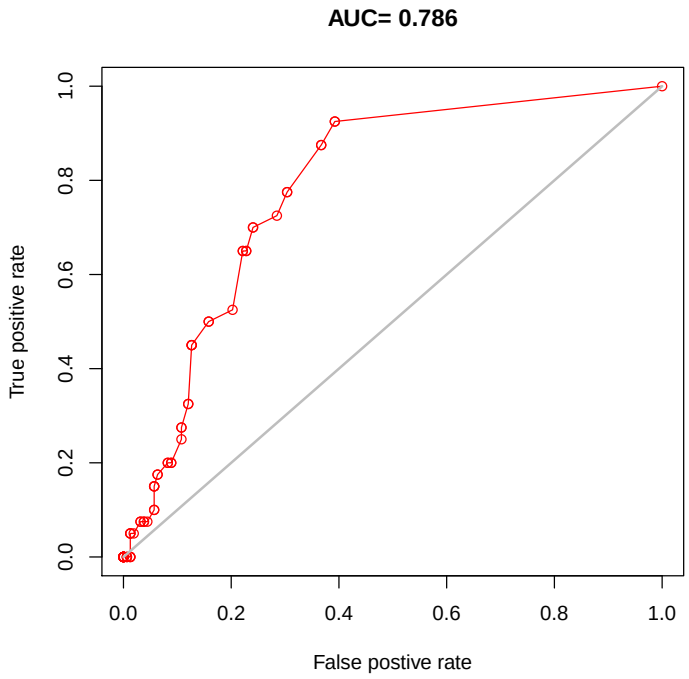
Below we show how you can compute these, and other statistics, with the dismo package. See ?evaluate for info on additional evaluation measures that are available. ROC/AUC can also be computed with the `ROCR` package.

```
> e = evaluate(p=p, a=a)
> class(e)

[1] "ModelEvaluation"
attr(,"package")
[1] "dismo"

> e

class            : ModelEvaluation
n presences      : 50
n absences       : 50
AUC              : 0.7756
cor              : 0.4763127
TPR+TNR threshold: 0.408

> par(mfrow=c(1, 2))
> density(e)
> boxplot(e, col=c('blue', 'red'))
```



Now back to some real data, presence-only in this case. We'll divide the data in two random sets, one for training a Bioclim model, and one for evaluating the model. (to do: comment on what AUC means for presence-only; Phillips et al, 2006)

```
> rand <- round(0.75 * runif(nrow(sdmdata)))
> traindata <- sdmdata[rand==0,]
> traindata <- traindata[traindata[,1] == 1, 2:9]
> testdata <- sdmdata[rand==1,]
> bc <- bioclim(traindata)
> e <- evaluate(testdata[testdata==1,], testdata[testdata==0,], bc)
> e

class            : ModelEvaluation
n presences      : 37
n absences       : 190
AUC              : 0.816643
cor              : 0.3107902
TPR+TNR threshold: 0.038

> plot(e, 'ROC')
```



Chapter 8

Data partitioning

The `kfold` function facilitates data partitioning. It creates a vector that assigns each row in the data matrix to a a group (between 1 to k).

Let's first create presence and background data.

```
> pres <- sdmdata[sdmdata[,1] == 1, 2:9]
> back <- sdmdata[sdmdata[,1] == 0, 2:9]
```

The background data will only be used for model testing and does not need to be partitioned. We now partition the data into 5 groups.

```
> k <- 5
> group <- kfold(pres, k)
> group[1:10]

 [1] 4 3 4 2 4 2 1 3 2 3

> unique(group)

[1] 4 3 2 1 5
```

Now we can fit and test our model five times. In each run, the records corresponding to one of the five groups is only used to evaluate the model, while the other four groups are only used to fit the model. The results are stored in a list called 'e'.

```
> e <- list()
> for (i in 1:k) {
+         train <- pres[group != i,]
+         test <- pres[group == i,]
+         bc <- bioclim(train)
+         e[[i]] <- evaluate(p=test, a=back, bc)
+ }
```

We can extract several things from the objects in 'e', but let's restrict ourselves to the AUC values and the "maximum of the sum of the sensitivity (true positive rate) and specificity (true negative rate)" (this is sometimes uses as a threshold for setting cells to presence or absence).

```
> auc <- sapply( e, function(x){slot(x, 'auc')} )
> auc

[1] 0.8328696 0.7371739 0.7670833 0.7552174 0.7653043

> mean(auc)

[1] 0.7715297

> sapply( e, function(x){ x@t[which.max(x@TPR + x@TNR)] } )

[1] 0.054 0.013 0.040 0.022 0.022
```

Part III

Modeling methods

Chapter 9

Types of algorithms & data used in examples

A large number of algorithms has been used in species distribution modeling. They can be classified as 'profile', 'regression', and 'machine learning' methods. Profile methods only consider 'presence' data, not absence or background data. Regression and machine learning methods use both presence and absence or background data. The distinction between regression and machine learning methods is not sharp, but it is perhaps still useful as way to classify models. Below we discuss examples of these different types of models. [[add geographic models]]

We will use the same data to illustrate all models, except that some models cannot use categorical variables. So for those models we drop the categorical variables from the predictors stack.

```
> pred_nf <- dropLayer(predictors, 'biome')
```

We'll use the *Bradypus* data for presence of a species. Let's make a training and a testing set.

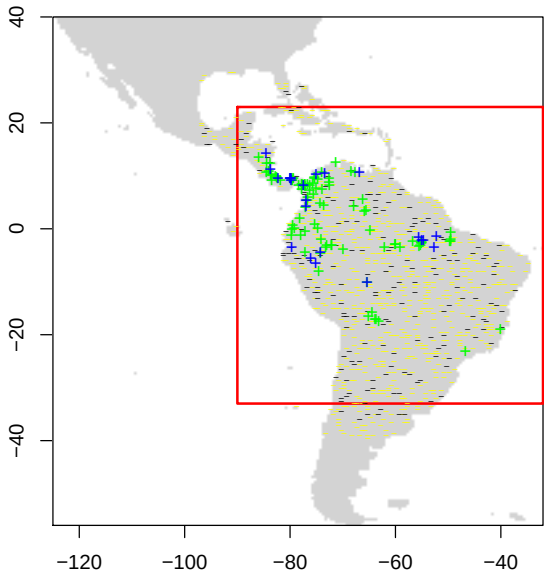
```
> group <- kfold(bradypus, 5)
> pres_train <- bradypus[group != 1, ]
> pres_test <- bradypus[group == 1, ]
```

To speed up processing, let's restrict the predictions to a more restricted area (defined by a rectangular extent):

```
> ext = extent(-90, -32, -33, 23)
```

Background data for training and a testing set. The first layer in the RasterStack is used as a 'mask'. That ensures that random points only occur within the spatial extent of the rasters, and within cells that are not NA, and that there is only a single absence point per cell. Here we further restrict the background points to be within 15% of our specified extent 'ext'.

```
> backg <- randomPoints(pred_nf, n=1000, ext=ext, extf = 1.25)
> colnames(backg) = c('lon', 'lat')
> group <- kfold(backg, 5)
> backg_train <- backg[group != 1, ]
> backg_test <- backg[group == 1, ]

> r = raster(pred_nf, 1)
> plot(!is.na(r), col=c('white', 'light grey'), legend=FALSE)
> plot(ext, add=TRUE, col='red', lwd=2)
> points(backg_train, pch='-', cex=0.5, col='yellow')
> points(backg_test, pch='-',  cex=0.5, col='black')
> points(pres_train, pch= '+', col='green')
> points(pres_test, pch='+', col='blue')
```



Chapter 10

Profile methods

The three methods described here, Bioclim, Domain, and Mahal. These methods are implemented in the dismo package, and the procedures to use these models are the same for all three.

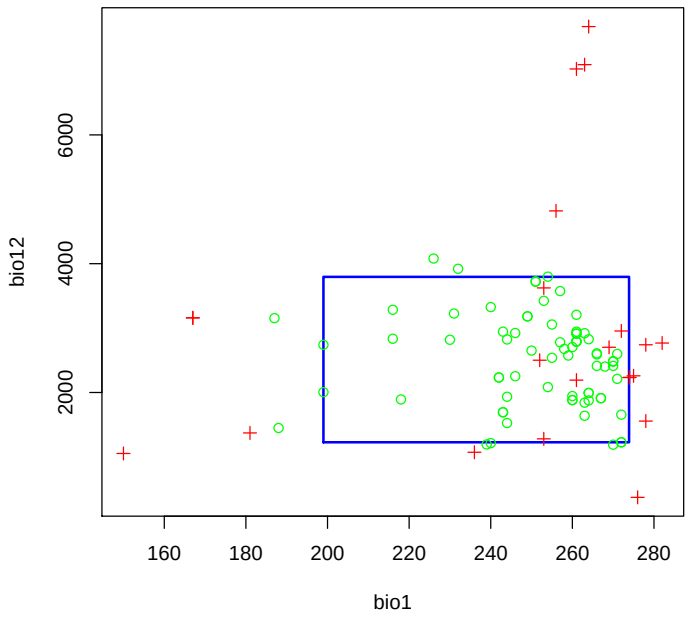
10.1 Bioclim

The BIOCLIM algorithm has been extensively used for species distribution modeling. BIOCLIM is a classic 'climate-envelope-model'. Although it generally does not perform as good as some other modeling methods (Elith *et al.* 2006), particularly in the context of climate change (Hijmans and Graham, 2006), it is still used, among other reasons because the algorithm is easy to understand and thus useful in teaching species distribution modeling. The BIOCLIM algorithm computes the similarity of a location by comparing the values of environmental variables at any location to a percentile distribution of the values at known locations of occurrence ('training sites'). The closer to the 50th percentile (the median), the more suitable the location is. The tails of the distribution are not distinguished, that is, 10 percentile is treated as equivalent to 90 percentile. In the 'dismo' implementation, the values of the upper tail values are transformed to the lower tail, and the minimum percentile score across all the environmental variables is used (i.e., BIOCLIM uses an approach like Liebig's law of the minimum). This value is subtracted from 1 and then multiplied with two so that the results are between 0 and 1. The reason for scaling this way is that the results become more like that of other distribution modeling methods and are thus easier to interpret. The value 1 will rarely be observed as it would require a location that has the median value of the training data for all the variables considered. The value 0 is very common as it is assigned to all cells with a value of an environmental variable that is outside the percentile distribution (the range of the training data) for at least one of the variables.

Earlier on, we fitted a Bioclim model using data.frame with each row representing the environmental data at known sites of presence of a species. Here we

fit a bioclim model simply using the predictors and the occurrence points (the function will do the extracting for us).

```
> bc <- bioclim(pred_nf, pres_train)
> plot(bc, a=1, b=2, p=0.85)
```



We evaluate the model in a similar way, by providing presence and background (absence) points, the model, and a RasterStack:

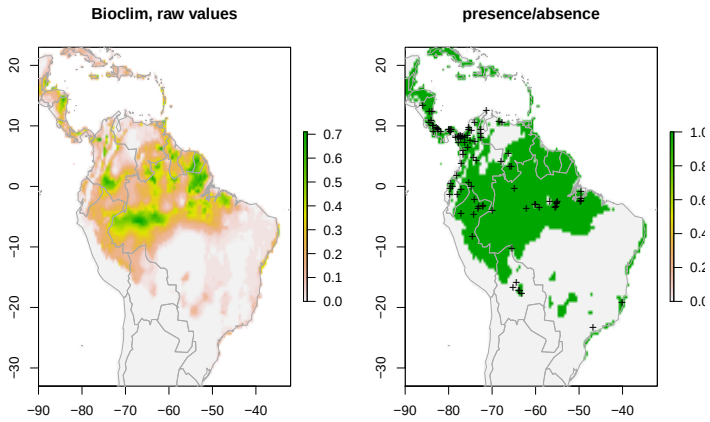
```
> e <- evaluate(pres_test, backg_test, bc, pred_nf)
> e

class            : ModelEvaluation
n presences      : 23
n absences       : 200
AUC              : 0.8020652
cor              : 0.2857707
TPR+TNR threshold: 0.06
```

And we use the RasterStack with predictor variables to make a prediction to a RasterLayer:

```
> pb <- predict(pred_nf, bc, ext=ext, progress='')
> pb
```

```
class       : RasterLayer
dimensions  : 112, 116, 12992  (nrow, ncol, ncell)
resolution  : 0.5, 0.5  (x, y)
extent      : -90, -32, -33, 23  (xmin, xmax, ymin, ymax)
coord. ref. : +proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0
values      : in memory
min value   : 0
max value   : 0.7096774
```

```
> par(mfrow=c(1,2))
> plot(pb, main='Bioclim, raw values')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> threshold <- e@t[which.max(e@TPR + e@TNR)]
> plot(pb > threshold, main='presence/absence')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> points(pres_train, pch='+')
```



Please note the order of the arguments in the predict function. In the example above, we used `predict(pred_nf, bc)` (first the RasterStack, then the model object), which is little bit less efficient than predict(bc, pred_nf) (first the model, than the RasterStack). The reason for using the order we have used, is that this will work for all models, whereas the other option only works for the models defined in the dismo package, such as Bioclim, Domain, and Maxent, but not for models defined in other packages (random forest, boosted regression trees, glm, etc.).

10.2 Domain

The Domain algorithm (Carpenter *et al.* 1993) has been extensively used for species distribution modeling. It did not perform very well in a model

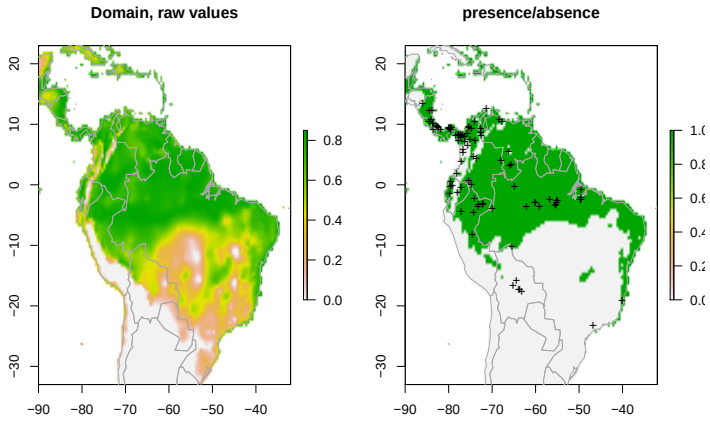
comparison (Elith *et al.* 2006) and very poorly when assessing climate change effects (Hijmans and Graham, 2006). The Domain algorithm computes the Gower distance between environmental variables at any location and those at any of the known locations of occurrence ('training sites').

The distance between the environment at point A and those of the known occurrences for a single climate variable is calculated as the absolute difference in the values of that variable divided by the range of the variable across all known occurrence points (i.e., the distance is scaled by the range of observations). For each variable the minimum distance between a site and any of the training points is taken. The Gower distance is then the mean of these distances over all environmental variables. The algorithm assigns to a place the distance to the closest known occurrence (in environmental space).

To integrate over environmental variables, the distance to any of the variables is used. This distance is subtracted from one, and (in this R implementation) values below zero are truncated so that the scores are between 0 (low) and 1 (high).

Below we fit a domain model, evaluate it, and make a prediction. We map the prediction, as well as a map subjectively classified into presence / absence.

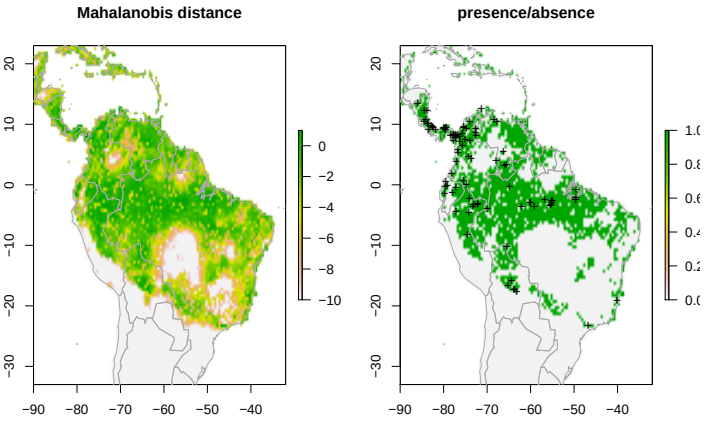
```
> dm <- domain(pred_nf, pres_train)
> e <- evaluate(pres_test, backg_test, dm, pred_nf)
> e

class            : ModelEvaluation
n presences      : 23
n absences       : 200
AUC              : 0.7934783
cor              : 0.2889847
TPR+TNR threshold: 0.64

> pd = predict(pred_nf, dm, ext=ext, progress='')
> par(mfrow=c(1,2))
> plot(pd, main='Domain, raw values')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> threshold <- e@t[which.max(e@TPR + e@TNR)]
> plot(pd > threshold, main='presence/absence')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> points(pres_train, pch='+')
```



10.3 Mahalanobis

The `mahal` function implements a species distribution model based on the Mahalanobis distance (Mahalanobis, 1936). Mahalanobis distance takes into account the correlations of the variables in the data set, and it is not dependent on the scale of measurements.

```
> mm <- mahal(pred_nf, pres_train)
> e <- evaluate(pres_test, backg_test, mm, pred_nf)
> e

class            : ModelEvaluation
n presences      : 23
n absences       : 200
AUC              : 0.8952174
cor              : 0.1337541
TPR+TNR threshold: -1.952

> pm = predict(pred_nf, mm, ext=ext, progress='')
> par(mfrow=c(1,2))
> pm[pm < -10] <- -10
> plot(pm, main='Mahalanobis distance')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> threshold <- e@t[which.max(e@TPR + e@TNR)]
> plot(pm > threshold, main='presence/absence')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> points(pres_train, pch='+')
```


Mahalanobis distance
presence/absence
20
10
0
−10
−20
−30
−90
−80
−70
−60
−50
−40
0
−2
−4
−6
−8
−10

Chapter 11

Regression models

The remaining models need to be fit presence textifand absence (background) data. With the exception of 'maxent', we cannot fit the model with a RasterStack and points. Instead, we need to extract the environmental data values ourselves, and fit the models with these values.

```
> train <- rbind(pres_train, backg_train)
> pb_train <- c(rep(1, nrow(pres_train)), rep(0, nrow(backg_train)))
> envtrain <- extract(predictors, train)
> envtrain <- data.frame( cbind(pa=pb_train, envtrain) )
> envtrain[,'biome'] = factor(envtrain[,'biome'], levels=1:14)
> head(envtrain)
```

```
  pa bio1 bio12 bio16 bio17 bio5 bio6 bio7 bio8 biome
1  1  263  1639   724    62  338  191  147  261     1
2  1  253  3624  1547   373  329  150  179  271     1
3  1  243  1693   775   186  318  150  168  264     1
4  1  243  1693   775   186  318  150  168  264     1
5  1  252  2501  1081   280  326  154  172  270     1
6  1  240  1214   516   146  317  150  168  261     2
```

```
> testpres <- data.frame( extract(predictors, pres_test) )
> testbackg <- data.frame( extract(predictors, backg_test) )
> testpres[ ,'biome'] = factor(testpres[ ,'biome'], levels=1:14)
> testbackg[ ,'biome'] = factor(testbackg[ ,'biome'], levels=1:14)
```

11.1 Generalized Linear Models

A generalized linear model (GLM) is a generalization of ordinary least squares regression. Models are fit using maximum likelihood and by allowing the linear model to be related to the response variable via a link function and by allowing the magnitude of the variance of each measurement to be a function of its

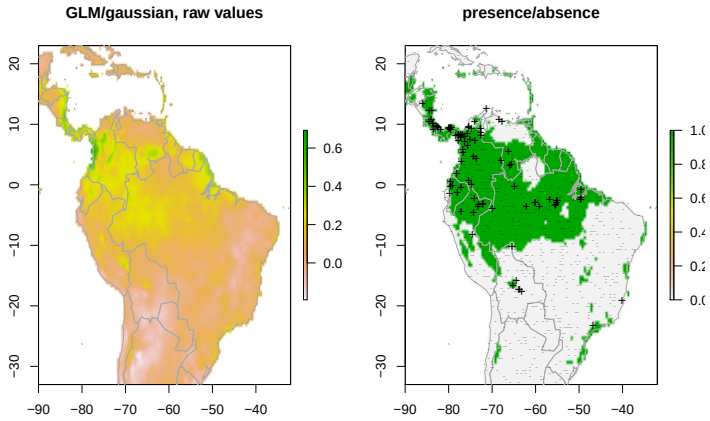
predicted value. Depending on how a GLM is specified it can be equivalent to (multiple) linear regression, logistic regression or Poisson regression. See Guisan *et al* (2002) for an overview of the use of GLM in species distribution modeling.

In R , GLM is implemented in the 'glm' function, and the link function and error distribution are specified with the 'family' argument. Examples are:

```
family = binomial(link = "logit")
family = gaussian(link = "identity")
family = poisson(link = "log")
```

Here we fit two basic glm models. All variables are used, but without interaction terms.

```
> gm1 <- glm(pa ~ bio1 + bio5 + bio6 + bio7 + bio8 + bio12 + bio16 + bio17,
+                          family = binomial(link = "logit"), data=envtrain)
> gm2 <- glm(pa ~ bio1 + bio5 + bio6 + bio7 + bio8 + bio12 + bio16 + bio17,
+                          family = gaussian(link = "identity"), data=envtrain)
> e1 = evaluate(testpres, testbackg, gm1)
> e2 = evaluate(testpres, testbackg, gm2)
> e1

class            : ModelEvaluation
n presences      : 23
n absences       : 200
AUC              : 0.8234783
cor              : 0.2923691
TPR+TNR threshold: -2.965

> e2

class            : ModelEvaluation
n presences      : 23
n absences       : 200
AUC              : 0.791087
cor              : 0.3238194
TPR+TNR threshold: 0.088

> pg <- predict(predictors, gm2, ext=ext)
> par(mfrow=c(1,2))
> plot(pg, main='GLM/gaussian, raw values')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> threshold <- e2@t[which.max(e@TPR + e@TNR)]
> plot(pg > threshold, main='presence/absence')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> points(pres_train, pch='+')
> points(backg_train, pch='-', cex=0.25)
```



11.2 Generalized Additive Models

Generalized additive models (GAMs; Hastie and Tibshirani, 1990; Wood, 2006) are an extension to GLMs. In GAMs, the linear predictor is the sum of smoothing functions. This makes GAMs very flexible, and they can fit very complex functions. It also makes them very similar to machine learning methods. In R , GAMs are implemented in the 'mgcv' package. The 'grasp' package implements species distribution modeling with gam (Lehman *et al.*, 2002).

Chapter 12

Machine learning methods

There is a variety of machine learning (sometimes referred to data mining) methods in R . For a long time there have been packages to do Artifical Neural Networks (ANN) and Classification and Regression Trees (CART). More recent methods include Random Forests, Boosted Regression Trees, and Support Vector Machines. Through the dismo package you can also use the Maxent program, that implements the most widely used method (maxent) in species distribution modeling. Breiman (2001a) provides a accessible introduction to machine learning, and how it contrasts with 'classical statistics' (model based probabilistic inference). Hastie *et al.*, 2009 provide what is probably the most extensive overview of these methods.

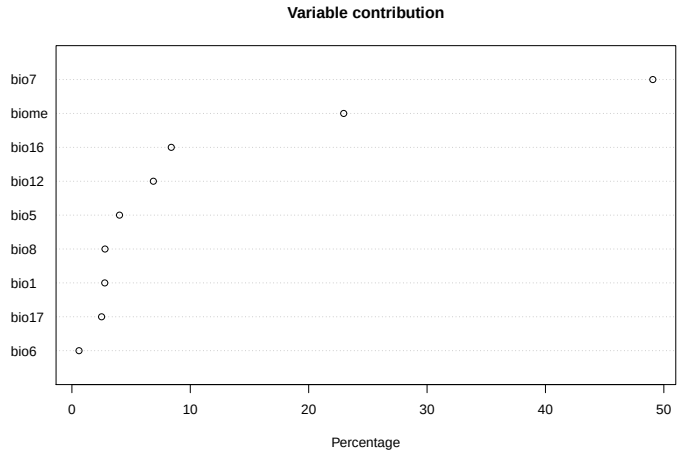
All the model fitting methods discussed here can be tuned in several ways. We do not explore that here, and only show the general approach. If you want to use one of the methods, then you should consult the R help pages (and other sources) to find out how to best implement the model fitting procedure.

12.1 Maxent

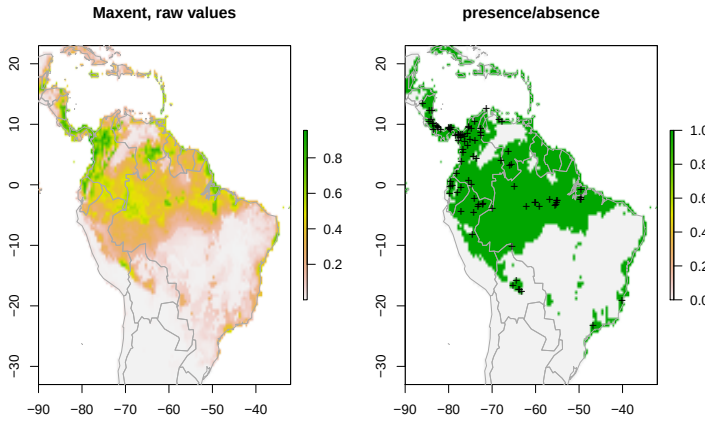
MaxEnt (Maximum Entropy; Phillips *et al.*, 2004, 2006) is the most widely used SDM algorithm. Elith *et al.*, 2010,) provide an explanation of the algorithm (and software) geared towards ecologists. MaxEnt is available as a stand-alone Java program. Dismo has a function 'maxent' that communicates with this program. To use it you must first download the program from `http://www.cs.princeton.edu/~schapire/maxent/`. Put the file 'maxent.jar' in the 'java' folder of the 'dismo' package. That is the folder returned by `system.file("java", package="dismo")`. Please note that this program (`maxent.jar`) cannot be redistributed or used for commercial purposes.

Because MaxEnt is implemented in dismo you can fit it like the profile methods (e.g. Bioclim). That is, you can provide presence points and a RasterStack. However, you can also first fit a model, like with the other methods such as glm. But in the case of MaxEnt you cannot use the formula notation.

```
> # checking if the jar file is present. If not, skip this bit
> jar <- paste(system.file(package="dismo"), "/java/maxent.jar", sep='')
> if (file.exists(jar)) {
+         xm <- maxent(predictors, pres_train, factors='biome')
+         plot(xm)
+ } else {
+     cat('cannot run this example because maxent is not available on this system')
+         plot(1)
+ }
```




```
> if (file.exists(jar)) {
+         e <- evaluate(pres_test, backg_test, xm, predictors)
+         e
+         px = predict(predictors, xm, ext=ext, progress='')
+         par(mfrow=c(1,2))
+         plot(px, main='Maxent, raw values')
+         plot(wrld_simpl, add=TRUE, border='dark grey')
+         threshold <- e@t[which.max(e@TPR + e@TNR)]
+         plot(px > threshold, main='presence/absence')
+         plot(wrld_simpl, add=TRUE, border='dark grey')
+         points(pres_train, pch='+')
+ } else {
+         plot(1)
+ }
```



12.2 Boosted Regression Trees

Boosted Regression Trees (BRT) is, unfortunately, known by a large number of different names. It was developed by Friedman (2001), who referred to it as a "Gradient Boosting Machine" (GBM). It is also known as "Gradient Boost", "Stochastic Gradient Boosting", "Gradient Tree Boosting". The method is implemented in the '`gbm`' package in R .

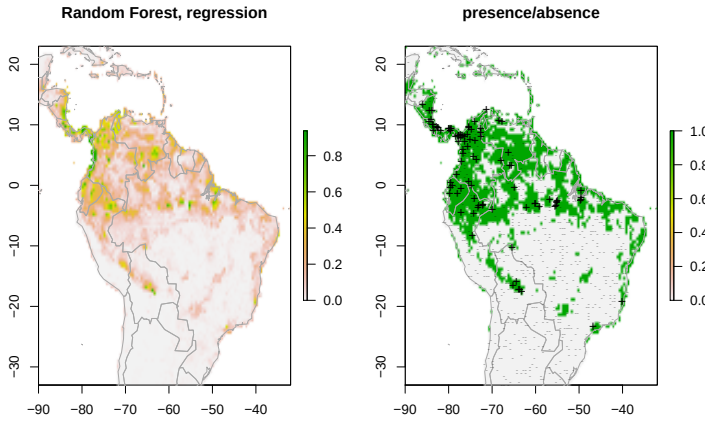
The article by Elith, Leathwick and Hastie (2009) describes the use of BRT in the context of species distribution modeling. Their article is accompanied by a number of R functions and a tutorial that have been slightly adjusted and incorporated into the 'dismo' package. These functions extend the functions in the '`gbm`' package, with the goal to make these easier to apply to ecological data, and to enhance interpretation. The adapted tutorial is available as a vignette to the dismo package. You can access it via the index of the help pages, or with this command: `vignette('gbm', 'dismo')`

12.3 Random Forest

The Random Forest (Breiman, 2001b) method is an extension of Classification and regression trees (CART; Breiman *et al.*, 1984). In R it is implemented in the function 'randomForest' in a package with the same name. The function randomForest can take a formula or, in two separate arguments, a data.frame with the predictor variables, and a vector with the response. If the response variable is a factor (categorical), randomForest will do classification, otherwise it will do regression. Whereas with species distribution modeling we are often interested in classification (species is present or not), it is my experience that using regression provides better results. rf1 does regression, rf2 and rf3 do

classification (they are exactly the same models). See the function tuneRF for optimizing the model fitting procedure.

```
> library(randomForest)
> model <- pa ~ bio1 + bio5 + bio6 + bio7 + bio8 + bio12 + bio16 + bio17
> rf1 <- randomForest(model, data=envtrain)
> model <- factor(pa) ~ bio1 + bio5 + bio6 + bio7 + bio8 + bio12 + bio16 + bio17
> rf2 <- randomForest(model, data=envtrain)
> rf3 <- randomForest(envtrain[,1:8], factor(pb_train))
> e = evaluate(testpres, testbackg, rf1)
> e

class            : ModelEvaluation
n presences      : 23
n absences       : 200
AUC              : 0.8141304
cor              : 0.4226258
TPR+TNR threshold: 0.116

> pr <- predict(predictors, rf1, ext=ext)
> par(mfrow=c(1,2))
> plot(pr, main='Random Forest, regression')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> threshold <- e@t[which.max(e@TPR + e@TNR)]
> plot(pr > threshold, main='presence/absence')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> points(pres_train, pch='+')
> points(backg_train, pch='-', cex=0.25)
```

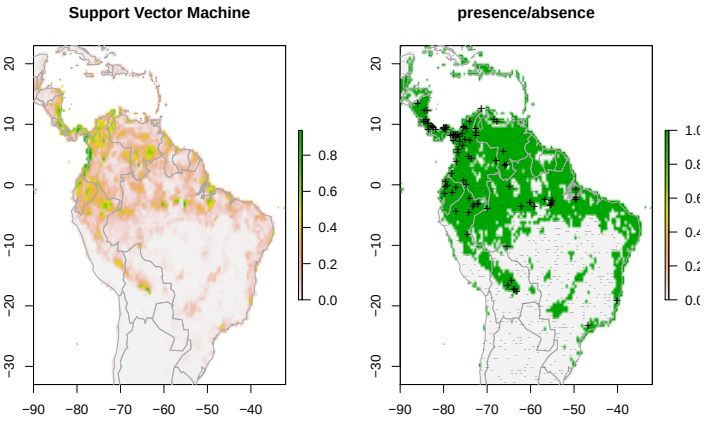


12.4 Support Vector Machines

Support Vector Machines (SVMs; Vapnik, 1998) apply a simple linear method to the data but in a high-dimensional feature space non-linearly related to the input space, but in practice, it does not involve any computations in that high-dimensional space. This simplicity combined with state of the art performance on many learning problems (classification, regression, and novelty detection) has contributed to the popularity of the SVM (Karatzoglou *et al.*, 2006). They were first used in species distribution modeling by Guo *et al.* (2005).

There are a number of implementations of svm in R . The most useful implementations in our context are probably function 'ksvm' in package 'kernlab' and the 'svm' function in package 'e1071'. 'ksvm' includes many different SVM formulations and kernels and provides useful options and features like a method for plotting, but it lacks a proper model selection tool. The 'svm' function in package 'e1071' includes a model selection tool: the 'tune' function (Karatzoglou *et al.*, 2006)

```
> library(kernlab)
> svm <- ksvm(pa ~ bio1 + bio5 + bio6 + bio7 + bio8 + bio12 + bio16 + bio17, data=envtrain)

Using automatic sigma estimation (sigest) for RBF or laplace kernel

> e = evaluate(testpres, testbackg, svm)
> e

class            : ModelEvaluation
n presences      : 23
n absences       : 200
AUC              : 0.7882609
cor              : 0.374769
TPR+TNR threshold: 0.033

> ps <- predict(predictors, rf1, ext=ext)
> par(mfrow=c(1,2))
> plot(ps, main='Support Vector Machine')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> threshold <- e@t[which.max(e@TPR + e@TNR)]
> plot(ps > threshold, main='presence/absence')
> plot(wrld_simpl, add=TRUE, border='dark grey')
> points(pres_train, pch='+')
> points(backg_train, pch='-', cex=0.25)
```


Support Vector Machine
presence/absence
20
10
0
−10
−20
−30
−90
−80
−70
−60
−50
−40
0.8
0.6
0.4
0.2
0.0

Chapter 13

Geographic models

The 'geographic models' described here are not commonly used in species distribution modeling. They are an attempt to formalize methods to draw 'expert range maps'. They can also be interpreted as null-models. To be completed. Colwel and Rangel, 2009, duality. Sampling background

13.1 Distance

13.2 Convex hulls

13.3 Circles

13.4 Inverse distance

Add indicator kriging

13.5 Voronoi hulls

Part IV

Additional topics

Chapter 14

Model transfer in space and time

14.1 Transfer in space

14.2 Transfer in time: climate change

Chapter 15

More things

There are many sophistications that are required by the realities that (a) there are multiple end uses of models, and (b) there are numerous issues with ecological data that mean that the assumptions of the standard methods don't hold. Could include:

- spatial autocorrelation
- imperfect detection
- mixed models (for nested data, hierarchical stuff)
- Bayesian methods
- resource selection functions?
- measures of niche overlap, linked to thoughts about niche conservatism?
- anything to do with phylogeography?
- advanced topics on predictors including remote sensing variables, thinking about extremes etc?
- species that don't "mix" with grids – freshwater systems etc..
- quantile regression
- model selection literature (AIC etc etc)

multispecies modeling: Mars, gdm

SDMTools

Model averaging See the BIOMOD for on multi-model inference.

Dealing with uncertainty

talk about how to target the "important" uncertainties (will vary with the application), an example of partial plots with standard errors, and predicting the upper and lower bounds; the idea of testing sensitivity to decisions made in the modeling process (including dropping out points etc etc).

Part V

References

Austin MP, 2002. Spatial prediction of species distribution: an interface between ecological theory and statistical modelling. Ecological Modelling 157:101-18.

Austin, M.P., and T.M. Smith, 1989. A new model for the continuum concept. Vegetatio 83:35-47.

Breiman, L., 2001a. Statistical Modeling: The Two Cultures. Statistical Science 16: 199-215.

Breiman, L., 2001b. Random Forests. Machine Learning 45: 5-32.

Breiman, L., J. Friedman, C.J. Stone and R.A. Olshen, 1984. Classification and Regression Trees. Chapman & Hall/CRC.

Carpenter G., A.N. Gillison and J. Winter, 1993. Domain: a flexible modelling procedure for mapping potential distributions of plants and animals. Biodiversity Conservation 2:667-680.

Colwell R.K. and T.F. Rangel, 2009. Hutchinson's duality: The once and future niche. Proceedings of the National Academy of Sciences 106:19651-19658.

Dormann C.F., Elith J., Bacher S., Buchmann C., Carl G., Carré G., Diekötter T., García Marquéz J., Gruber B., Lafourcade B., Leitão P.J., Münkemüller T., McClean C., Osborne P., Reineking B., Schröder B., Skidmore A.K., Zurell D., Lautenbach S. (2011 in review) Collinearity: a review of methods to deal with it and a simulation study evaluating their performance.

Elith, J. and J.R. Leathwick, 2009. Species Distribution Models: Ecological Explanation and Prediction Across Space and Time. Annual Review of Ecology, Evolution, and Systematics 40: 677-697. `http://dx.doi.org/10.1146/annurev.ecolsys.110308.120159`

Elith, J., C.H. Graham, R.P. Anderson, M. Dudik, S. Ferrier, A. Guisan, R.J. Hijmans, F. Huettmann, J. Leathwick, A. Lehmann, J. Li, L.G. Lohmann, B. Loiselle, G. Manion, C. Moritz, M. Nakamura, Y. Nakazawa, J. McC. Overton, A.T. Peterson, S. Phillips, K. Richardson, R. Scachetti-Pereira, R. Schapire, J. Soberon, S. Williams, M. Wisz and N. Zimmerman, 2006. Novel methods improve prediction of species' distributions from occurrence data. Ecography 29: 129-151. `http://dx.doi.org/10.1111/j.2006.0906-7590.04596.x`

Elith, J., S.J. Phillips, T. Hastie, M. Dudik, Y.E. Chee, C.J. Yates, 2011. A statistical explanation of MaxEnt for ecologists. Diversity and Distributions 17:43-57. `http://dx.doi.org/10.1111/j.1472-4642.2010.00725.x`

Elith, J., J.R. Leathwick and T. Hastie, 2009. A working guide to boosted regression trees. Journal of Animal Ecology 77: 802-81

Ferrier, S. and A. Guisan, 2006. Spatial modelling of biodiversity at the community level. Journal of Applied Ecology 43:393-40

Fielding, A.H. and J.F. Bell, 1997. A review of methods for the assessment of prediction errors in conservation presence/absence models. Environmental Conservation 24: 38-49

Franklin, J. 2009. Mapping Species Distributions: Spatial Inference and Prediction. Cambridge University Press, Cambridge, UK.

Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. The Annals of Statistics 29: 1189-1232. `http://www-stat.`

stanford.edu/~jhf/ftp/trebst.pdf)

Graham, C.H., J. Elith, R.J. Hijmans, A. Guisan, A.T. Peterson, B.A. Loiselle and the NCEAS Predicting Species Distributions Working Group, 2007. The influence of spatial errors in species occurrence data used in distribution models. Journal of Applied Ecology 45: 239-247

Guisan, A., Thomas C. Edwards, Jr, and Trevor Hastie, 2002. Generalized linear and generalized additive models in studies of species distributions: setting the scene. Ecological Modelling 157: 89-100.

Guo, Q., M. Kelly, and C. Graham, 2005. Support vector machines for predicting distribution of Sudden Oak Death in California. Ecological Modeling 182:75-90

Guralnick, R.P., J. Wieczorek, R. Beaman, R.J. Hijmans and the BioGeomancer Working Group, 2006. BioGeomancer: Automated georeferencing to map the world's biodiversity data. PLoS Biology 4: 1908-1909. http://dx.doi.org/10.1371/journal.pbio.0040381

Hastie, T.J. and R.J. Tibshirani, 1990. Generalized Additive Models. Chapman & Hall/CRC.

Hastie, T., R. Tibshirani and J. Friedman, 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Second Edition) http://www-stat.stanford.edu/~tibs/ElemStatLearn/

Hijmans R.J., and C.H. Graham, 2006. Testing the ability of climate envelope models to predict the effect of climate change on species distributions. Global change biology 12: 2272-2281. http://dx.doi.org/10.1111/j.1365-2486.2006.01256.x

Hijmans, R.J., M. Schreuder, J. de la Cruz and L. Guarino, 1999. Using GIS to check coordinates of germplasm accessions. Genetic Resources and Crop Evolution 46: 291-296.

Hijmans, R.J., S.E. Cameron, J.L. Parra, P.G. Jones and A. Jarvis, 2005. Very high resolution interpolated climate surfaces for global land areas. International Journal of Climatology 25: 1965-1978. http://dx.doi.org/10.1002/joc.1276

Karatzoglou, A., D. Meyer and K. Hornik, 2006. Support Vector Machines in R . Journal of statistical software 15(9). http://www.jstatsoft.org/v15/i09/

Kéry M., Gardner B., Monnerat C. (2010) Predicting species distributions from checklist data using site-occupancy models. J. Biogeogr. 37:1851–1862

Lehmann, A., J. McC. Overton and J.R. Leathwick, 2002. GRASP: Generalized Regression Analysis and Spatial Predictions. Ecological Modelling 157: 189-207.

Leathwick J., Whitehead D. (2001) Soil and atmospheric water deficits and the distribution of New Zealand's indigenous tree species. Functional Ecology 15:233–242.

Liu C., White M., Newell G. (2011) Measuring and comparing the accuracy of species distribution models with presence–absence data. Ecography 34:232-243.

Liu C., Berry P.M., Dawson T.P., Pearson R.G. (2005) Selecting thresholds of occurrence in the prediction of species distributions. Ecography 28:385-393.

Mahalanobis, P.C., 1936. On the generalised distance in statistics. Proceedings of the National Institute of Sciences of India 2: 49-55.

Mellert K.H., Fensterer V., Küchenhoff H., Reger B., Kölling C., Klemmt H.J., Ewald J. (2011 in press) Hypothesis-driven species distribution models for tree species in the Bavarian Alps. J. Veg. Sci.

Nix, H.A., 1986. A biogeographic analysis of Australian elapid snakes. In: Atlas of Elapid Snakes of Australia. (Ed.) R. Longmore, pp. 4-15. Australian Flora and Fauna Series Number 7. Australian Government Publishing Service: Canberra.

Olson, D.M, E. Dinerstein, E.D. Wikramanayake, N.D. Burgess, G.V.N. Powell, E.C. Underwood, J.A. D'amico, I. Itoua, H.E. Strand, J.C. Morrison, C.J. Loucks, T.F. Allnutt, T.H. Ricketts, Y. Kura, J.F. Lamoreux, W.W.Wettengel, P. Hedao, and K.R. Kassem. 2001. Terrestrial Ecoregions of the World: A New Map of Life on Earth. BioScience 51:933-938

Phillips S.J., Elith J. (2011 in press) Logistic methods for resource selection functions and presence-only species distribution models, AAAI (Association for the Advancement of Artificial Intelligence), San Francisco, USA.

Phillips, S.J., R.P. Anderson, R.E. Schapire, 2006. Maximum entropy modeling of species geographic distributions. Ecological Modelling 190:231-259.

Phillips, S. J., M. Dudik, J. Elith, C. H. Graham, A. Lehmann, J. Leathwick, and S. Ferrier. 2009. Sample selection bias and presence-only distribution models: implications for background and pseudo-absence data. Ecological Applications 19:181-197.

Potts J., Elith J. (2006) Comparing species abundance models. Ecol. Model. 199:153-163.

Vapnik, V., 1998. Statistical Learning Theory. Wiley, New York.

VanDerWal J., Shoo L.P., Graham C., Williams S.E., 2009. Selecting pseudo-absence data for presence-only distribution modeling: how far should you stray from what you know? Ecol. Model. 220:589-594.

Ward G., Hastie T., Barry S.C., Elith J., Leathwick J.R., 2009. Presence-only data and the EM algorithm. Biometrics 65:554-563.

Wieczorek, J., Q. Guo and R.J. Hijmans, 2004. The point-radius method for georeferencing point localities and calculating associated uncertainty. International Journal of Geographic Information Science 18: 745-767.

Wisz, M.S., R.J. Hijmans, J. Li, A.T. Peterson, C.H. Graham, A. Guisan, and the NCEAS Predicting Species Distributions Working Group, 2008. Effects of sample size on the performance of species distribution models. Diversity and Distributions 14: 763-773.

Wood, S., 2006. Generalized Additive Models: An Introduction with R . Chapman & Hall/CRC.