Running our first ENM/SDM

Emilio Berti

We can now model the ecological niche and the distribution of the species *Podarcis muralis*.

Species niche

We start by loading the species and climate data that we prepared in the previous section.

```r
d <- read.csv("../data/occurrences.csv")
head(d)
```

```
  wc2.1_10m_bio_1 wc2.1_10m_bio_12 wc2.1_10m_bio_13 wc2.1_10m_bio_14
1       14.709969              510               60               20
2       11.454646              775               89               44
3        8.854438              933              106               60
4       10.599990              633               61               44
5       10.302510              632               58               45
6        2.760979              549               68               27
  wc2.1_10m_bio_15 wc2.1_10m_bio_4 wc2.1_10m_bio_5 wc2.1_10m_bio_6   cell
1        25.683231        686.3660        32.04650         2.11375 620998
2        21.440331        589.9808        25.82825        -0.33375 621014
3        18.089415        578.0707        22.92975        -2.12900 618853
4         9.118695        567.0883        23.99325         0.50650 538936
5         8.751935        560.5479        23.76875         0.34775 534618
6        30.307224        617.9191        15.89150        -8.37175 562745
           x        y occ
1 -0.4166667 42.08333   1
2  2.2500000 42.08333   1
3  2.0833333 42.25000   1
4  2.5833333 48.41667   1
5  2.9166667 48.75000   1
6 10.7500000 46.58333   1
```

The column names `wc2.1_10m_bio_<id>` means that this is data from WorldClim (`wc`) at a resolution of 10 arc-minutes (`10m`) for bioclimatic variables (`<id>`). Bioclimatic variables are generally highly correlated with each others and only a subset of them should be used for train an ecological niche model. Variable selection can be performed with the usual statistical tricks or, even better, can be informed by the biology of the species. Because I do not know the biology of the species, I build several competing ENMs and test which one is best using the Akaike information criterion (AIC).

The first model considers only average temperature (`BIO01`) and total precipitation (`BIO12`) to train an ENM using `glm()`.

```r
enm_01_12 <- glm(
  occ ~ poly(wc2.1_10m_bio_1, 2, raw = TRUE) + poly(wc2.1_10m_bio_12, 2, raw = TRUE),
  data = d,
  family = "binomial"
)
```

We can test how well these variables explain the distribution of the species by building another ENM with different variables and comparing it with the model above. For example, we can use the minimum temperature of the coldest month (`BIO06`) and the precipitation of the driest month (`BIO14`) instead.

```r
enm_06_14 <- glm(
  occ ~ poly(wc2.1_10m_bio_6, 2, raw = TRUE) + poly(wc2.1_10m_bio_14, 2, raw = TRUE),
  data = d,
  family = "binomial"
)
```

We can compared the two models by AIC, with the best most having the lowest AIC.

```r
AIC(enm_01_12, enm_06_14)
```

```
          df      AIC
enm_01_12  5 4289.915
enm_06_14  5 4030.867
```

The model with the second set of variables explain the distribution of the species better than the first model.

Statistical dredging (optional and dangerous)

There are packages to run each possible model starting from a full model and rank them according to AIC. The most popular package is `MuMIn`, which contains the function `dredge()` to perform this steps. Start with the full model, i.e. the most complex model, and pass this to `dredge()`.

```
# full model
full_enm <- glm(
  occ ~ poly(wc2.1_10m_bio_1, 2, raw = TRUE) +
   poly(wc2.1_10m_bio_12, 2, raw = TRUE) +
   poly(wc2.1_10m_bio_13, 2, raw = TRUE) +
   poly(wc2.1_10m_bio_14, 2, raw = TRUE) +
   poly(wc2.1_10m_bio_15, 2, raw = TRUE) +
   poly(wc2.1_10m_bio_5, 2, raw = TRUE) +
   poly(wc2.1_10m_bio_6, 2, raw = TRUE),
  data = d,
  family = "binomial"
)

# dredging
options(na.action = "na.fail")   # required by MuMIn, don't think about it
dr <- MuMIn::dredge(full_enm)
head(dr)
```

```
Global model call: glm(formula = occ ~ poly(wc2.1_10m_bio_1, 2, raw = TRUE) + poly(wc2.1_10m
    2, raw = TRUE) + poly(wc2.1_10m_bio_13, 2, raw = TRUE) +
    poly(wc2.1_10m_bio_14, 2, raw = TRUE) + poly(wc2.1_10m_bio_15,
    2, raw = TRUE) + poly(wc2.1_10m_bio_5, 2, raw = TRUE) + poly(wc2.1_10m_bio_6,
    2, raw = TRUE), family = "binomial", data = d)
---
Model selection table
     (Int) ply(wc2.1_10m_bio_1,2,T) ply(wc2.1_10m_bio_12,2,T)
128 -17.89                        +                         +
126 -19.23                        +
120 -20.91                        +                         +
118 -21.79                        +
124 -22.34                        +                         +
116 -23.99                        +                         +
    ply(wc2.1_10m_bio_13,2,T) ply(wc2.1_10m_bio_14,2,T)
128                         +                         +
126                         +                         +
120                         +
118                         +
124                                                   +
116
    ply(wc2.1_10m_bio_15,2,T) ply(wc2.1_10m_bio_5,2,T) ply(wc2.1_10m_bio_6,2,T)
128                         +                        +                        +
126                         +                        +                        +
120                         +                        +                        +
118                         +                        +                        +
124                         +                        +                        +
116                         +                        +                        +
    df    logLik   AICc delta weight
128 15 -1673.297 3376.7  0.00  0.983
126 13 -1679.477 3385.1  8.32  0.015
120 13 -1682.201 3390.5 13.77  0.001
118 11 -1684.682 3391.4 14.70  0.001
124 13 -1708.921 3444.0 67.21  0.000
116 11 -1718.051 3458.2 81.44  0.000
Models ranked by AICc(x)
```

This table contains all possible sub-models of `full_enm` (including also `full_enm`) as rows ranked by AICc. A model contains the variable if there is a `+` at that column. The column `weight` quantifies the support of that model. We see that the model in first row (lowest AICc) is strongly supported against all others and we can select only that as our ENM. This model is actually the full model as it contains all bioclimatic variables specified.

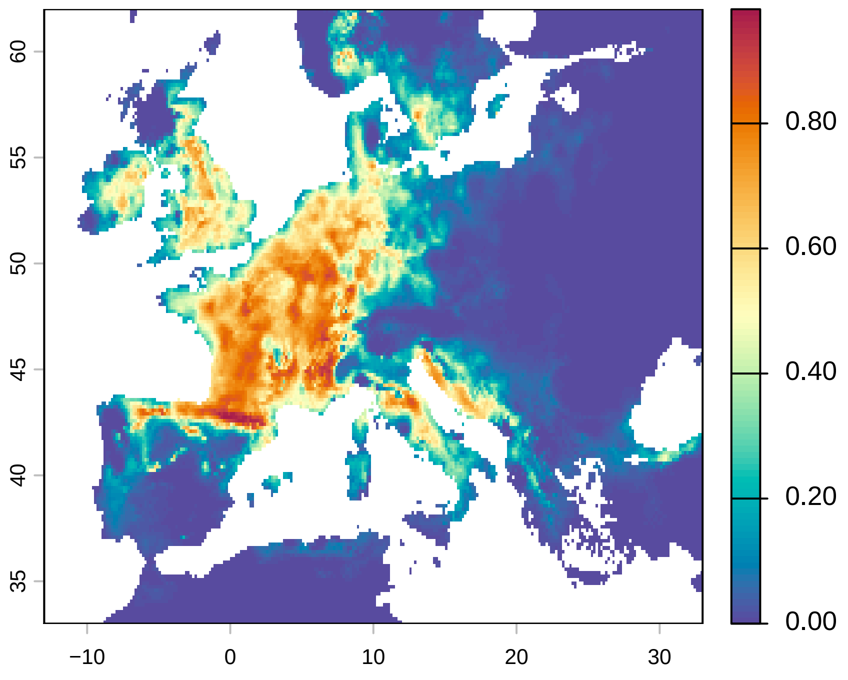
Species distribution

When using `glm()`, `terra` makes it extremely easy to produce a SDM. We first need to load the raster layers of the bioclimatic variables.

```r
library(terra)

ff <- list.files("../data", pattern = ".tif") # all files with .tif extension
r <- rast(file.path("..", "data", ff))
roi <- ext(-13, 33, 33, 62) # roi of Europe
r <- crop(r, roi) # crop to Europe
```

The `terra` function `predict(<raster>, model)` is all we need.

```r
sdm <- predict(r, full_enm, type = "response")
plot(sdm, col = hcl.colors(100, "Spectral", rev = TRUE))
```



Binary projections

And we obtained the projected suitability of the species for Europe. Note that this is a continuous value, in this case representing the probability of detecting the species given climate. If we are interested in a binary map, e.g. showing the climatic range of the species, we need to binarize this continuous value into `0/1`. There are several approaches to achieve this, but here we consider only the approach using the true skill statistics (TSS), which is one of the most widely used.

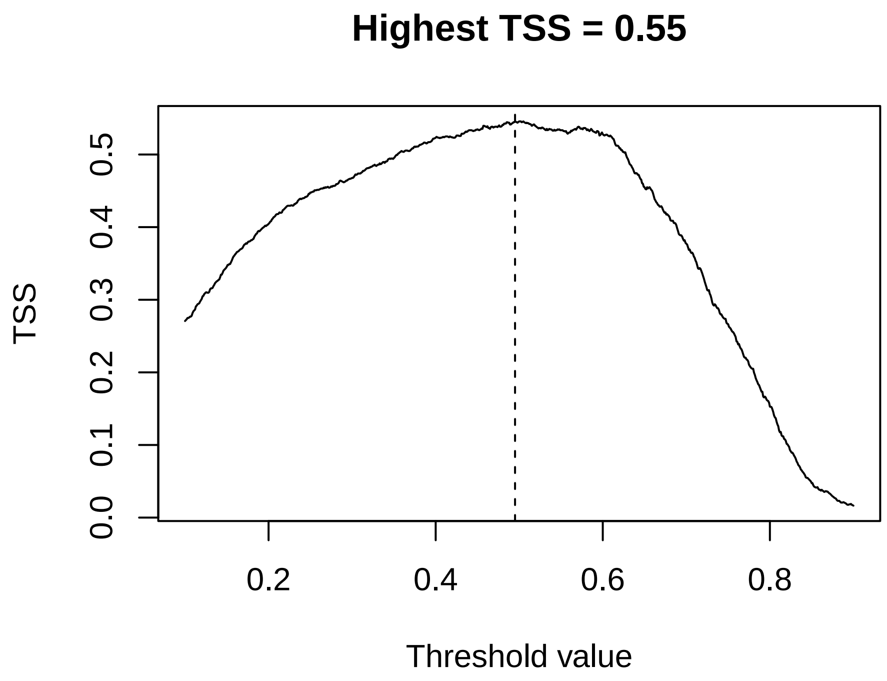
The main idea of the TSS approach is to pick a threshold value and set the cells of the map above to `0` if their values if less than this threshold and to `1` otherwise. These `0/1` values are then compared to the known occurrence of the species to calculate

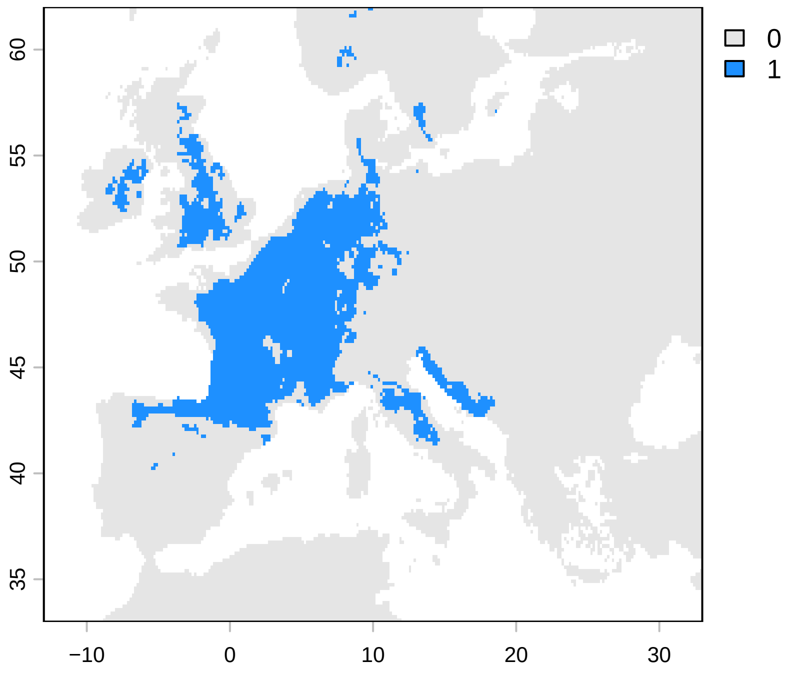
- The number of occurrences correctly predicted as presences (true positives, TP).
- The number of occurrences incorrectly predicted as absences (false negatives, FN).
- The number of occurrences correctly as absences (true negatives, TN).
- The number of occurrences incorrectly presences (false positives, FP).

TSS is defined as $TSS = \frac{TPTN - FPFN}{(TP+FN)(TN+FP)}$, which is a statistic balancing how well the model performs in predicting both presences and absences. TSS ranges from 0, for a model not better than random, to 1, for a model with perfect predictions.

If we pick several threshold and calculate the TSS for each of them, the best threshold is the one that has highest TSS, which is also the TSS of our model predictions.

```r
# extract the values from the continuous map
suit <- extract(sdm, d[, c("x", "y")], ID = FALSE)[, 1]

# generate a gradient of threshold values
threshold <- seq(0.1, 0.9, by = 0.001)
tss <- rep(NA, length(threshold)) # empty vector for storage

# iterate over threshold values
for (i in seq_along(threshold)) {
  p <- ifelse(suit > threshold[i], 1, 0)
  TP <- sum(p == 1 & d$occ == 1)
  FP <- sum(p == 1 & d$occ == 0)
  FN <- sum(p == 0 & d$occ == 1)
  TN <- sum(p == 0 & d$occ == 0)
  sens <- TP / (TP + FN)
  spec <- TN / (TN + FP)
  tss[i] <- sens + spec - 1
}

th <- threshold[which.max(tss)] # best threshold

plot(
  threshold, tss,
  type = "l", main = paste0("Highest TSS = ", round(max(tss), 2)),
  xlab = "Threshold value", ylab = "TSS"
)
abline(v = th, lty = 2)

# binarize the continuous map
sdm_bin <- ifel(sdm >= th, 1, 0)
plot(sdm_bin, col = c("grey90", "dodgerblue"))
```

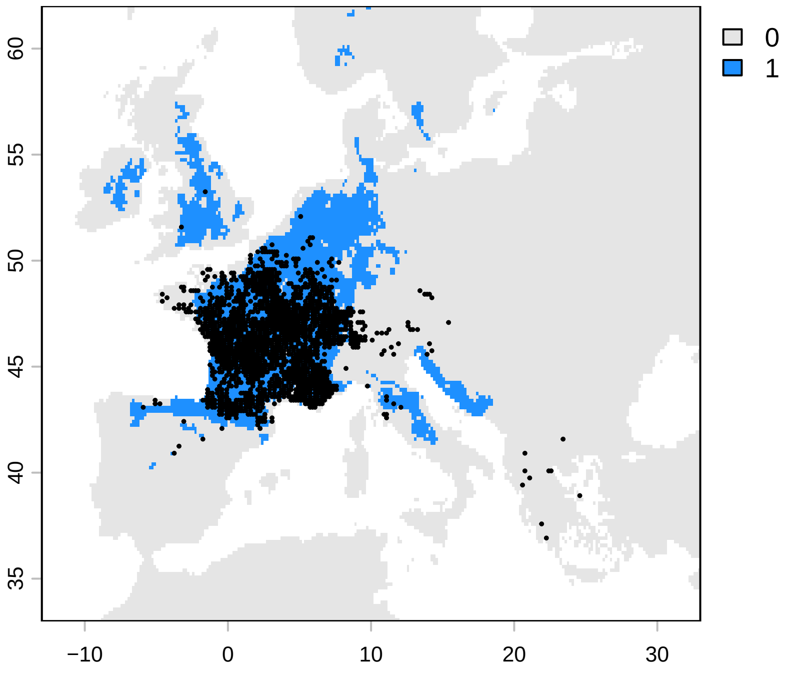




This binary map is quite incorrect for this species. According to the IUCN (https://www.iucnredlist.org/species/61550/12514105), this species is found also in most of Italy, all the Balkans, and part of Turkey, but is not found in the UK, the Low Countries, and most of

Germany. Why do we get such bad projections compared to the known range from IUCN? We will answer this in a next lecture, but my general recommendation is to plot the detection points.

```r
plot(sdm_bin, col = c("grey90", "dodgerblue"))
points(d[d$occ == 1, c("x", "y")], pch = 20, cex = .3)
```



The selected detection records are geographically biased and likely the environmental (climatic) space is not well represented.