Modelling Rare Species

Flavien Collart

2023-08-11

a. Modelling rare species

In this section, we will use the ecospat package to employ the method called "ensemble of small models (ESM)", which was developped by (Lomba et al. 2010; F. T. Breiner et al. 2015; Frank T. Breiner et al. 2018), which is particularly suitable for rare species. One of the major problem to model rare species is that the number of occurrences is scarce. Although some studies reported that species could be accurately modeled with very low sample size (e.g. 3 occurrences in (Proosdij et al. 2015)), sample size is problematic for the modelling procedure, limiting the number of predictors that can be put in a model, with a general rule of thumb that not more than one predictor term should be used per 10 occurrences. Thus to address this issue, ESMs compute bivariate models and then combine all possible bivariate models into an ensemble‘. By averaging simple small models to an ensemble, ESMs avoid overfitting without losing explanatory power through reducing the number of predictor variables, and were shown to perform significantly better than standard SDMs with species having a low number of occurrences (F. T. Breiner et al. (2015)). For this section, we will be focusing on modelling the ecological niche of *Veronica alpina* in the Western Swiss Alps.

i. Pre-Modelling

As the ESM core of the ecospat package relies on biomod2, we need to first format our data by using the function 'BIOMOD_FormatingData', where species occurrences and associated coordinates, the environmental conditions and the name of the species of interest need to be given. In this example we have presence-absence data for 300 plots, where *Veronica alpina* is present in 12 locations and for which 5 environmental variables were collected. These variables are the growing degree days (with a 0°C threshold), moisture index over the growing season (average values for June to August in mm day-1), the annual sum of radiation (in kJ m-2 year-1), Slope (in degrees), and the topographic position .

```r
# Load the needed packages
library(ecospat)
library(biomod2)
library(terra)
```

```r
set.seed(123)
data("ecospat.testData")

# coordinates of the plots
xy <- ecospat.testData[,2:3]
# species presences and absences
sp_occ <- ecospat.testData$Veronica_alpina
sum(sp_occ) ## Number of occurrences
```

```
## [1] 12
```

```
# environmental data
env <- rast(system.file("extdata/ecospat.testEnv.tif", package="ecospat"))
```

```
# Formatting the data with the BIOMOD_FormatingData() function from the package biomod2
myBiomodData <- biomod2::BIOMOD_FormatingData(resp.var =
                                                as.numeric(sp_occ),
                                              expl.var = env,
                                              resp.xy = xy,
                                              resp.name = "Veronica.Alpina",
                                              filter.raster = TRUE)
```

ii. Core-Modelling

The function *ecospat.ESM.Modeling* is used to model the ecological niche of the species by generating bivariate models.

The argument *data* is for the formatted dataset, notably containing species occurrence/absence, plot coordinates and environmental predictors.

ESM can fit 11 different algorithms: Generalized Linear Model (‘GLM’), Gradient Boosted Machine (‘GBM’), Generalized Additive Models (‘GAM’), ‘CTA’, Artificial Neural Network (‘ANN’), ‘SRE’, ‘FDA’, ‘MARS’, ‘RF’, Maximum entropy (‘MAXENT’, using the java software or ‘MAXNET’ from the maxnet package). The desired algorithms can be provided in the argument *models*. Model parameters can be adapted via the argument *models.options* by giving the object from the function BIOMOD_ModellingOptions() of the biomod2 package. A tuning to obtain the optimal parameters for the model can be realized with the argument *tune*. *Prevalence* can be set to build a “weighted response”. If NULL, each observation (presence or absence) will have the same weight. You can also give weight to the response point via the argument *Yweights*.

To evaluate the models, the function can perform a repeated split-sampling cross-validations when using the arguments *DataSplit* and *NbRunEval*. *DataSplit* corresponds to the percentage of occurrences and absences used to calibrate the models. *NbRunEval* indicates the number of replicates that you need. The function also allows performing user defined cross-validations by giving a logical matrix in the argument *DataSplitTable*, where each column corresponds to a run. A value TRUE means that this point will be used for model calibration while a FALSE is for model evaluation.

weighting.score corresponds to the evaluation metric used to weight single bivariate models into an ensemble. The available evaluation metrics are: ‘AUC’, ‘SomersD’ (2xAUC-1), ‘Kappa’, ‘TSS’ or ‘Boyce’.

which.biva allows to split model runs, e.g. if which.biva is 1:3, only the three first bivariate variable combinations will be modeled. This allows to run different bivariate splits on different computers. However, it is better not to use this option if all models are run on a single computer. *NB* Make sure to give each of your biva runs a unique *modeling.id.* and avoid space characters.

Parallel computing can be enabled with the argument *parallel*

The following step is to combine all the bivariate models into an ensemble. To so, we can use the function *ecospat.ESM.EnsembleModeling* which will need the object returned by *ecospat.ESM.Modeling*, the evaluation metric used to weight the bivariate models (*weighting.score*) and a *threshold* to remove poor performing models. The argument *models* allows to select one or several algorithms to realize the ensemble.

```
my.ESM.EF <- ecospat.ESM.EnsembleModeling(ESM.modeling.output = my.ESM,
                                          weighting.score = "SomersD",
                                          threshold = 0,
                                          models = NULL)
```

ESM performances resulted from the cross-validations can be observed in the object returned by *ecospat.ESM.EnsembleModeling*.

```
my.ESM.EF$ESM.evaluations
```

Table 1: Table X: ESM performances based on a mean or standard deviations across bivariate model performances of a same run

model	threshold	sensitivity	specificity	Kappa	AUC	sensitivity.sd	specificity.sd	Kappa.sd	AUC.sd	TSS	SomersD	MPA	Boyce	technique	RUN
RUN1_GLM	175	1	0.671	0.155	0.771	0	0.051	0.071	0.059	0.671	0.541	0.234	-0.138	GLM	RUN1
RUN2_GLM	525	1	0.847	0.332	0.926	0	0.039	0.127	0.036	0.847	0.853	0.616	0.474	GLM	RUN2
RUN3_GLM	570	1	0.776	0.238	0.859	0	0.045	0.100	0.056	0.776	0.718	0.612	-0.429	GLM	RUN3

However, because a minimum sample size is needed to evaluate models (see (Jiménez-Valverde 2020)), it is recommended to evaluate ESMs using the pooling evaluation ((Collart and Guisan 2023)). The function *ecospat.ESM.EnsembleEvaluation* uses this approach, which consists to pool the suitability values of the hold-out data (evaluation dataset) across replicates. As the same data point (presence or absence or background point) is presumably sampled in several replicates, the suitability values for each data point is consequently averaged across replicates where they were sampled. This procedure generates a series of independent suitability values with a size approximately equal (as some data points may not have been sampled by chance in any of the n replicates) to that of the number of data point. This function can compute several metrics, which can be selected with the argument *metrics*. If needed, *EachSmallModels* allows to evaluate each bivariate models via the pooling evaluation

```
my.ESM.EF.eval <- ecospat.ESM.EnsembleEvaluation(
                              ESM.modeling.output = my.ESM,
                              ESM.EnsembleModeling.output =
                                my.ESM.EF,
                              metrics = c("SomersD","AUC","
                                         MaxTSS","MaxKappa",
                                         "Boyce"),
                              EachSmallModels = FALSE)

## Evaluation dataset obtained by the pooling evaluation
pred.test <- as.data.frame(my.ESM.EF.eval$ESM.fit)

## Evaluation scores of the ESM based on the pooling evaluation
my.ESM.EF.eval$ESM.evaluations
```

Table 2: Table X: ESM performances based on the pooling evaluation

	AUC	SomersD	Boyce	MaxKappa
GLM	0.8469203	0.6938406	0.73	0.2496419

Ecospat package also have numerous functions to compute model performances on your own. By providing model predictions and species observation.

For example, the Boyce index which only requires presences can be calculated with the function *ecospat.boyce* The argument *obs* should contains the model prediction for the presences while *fit* should contain the predictions of background points. Correlation measurement can be changed via the argument *method*

```
boyce.index <- ecospat.boyce(fit = pred.test$Fit_GLM,
                             obs = pred.test$Fit_GLM[pred.test$resp==1],
                             PEplot = FALSE,
                             method = "spearman")
boyce.index$cor
```

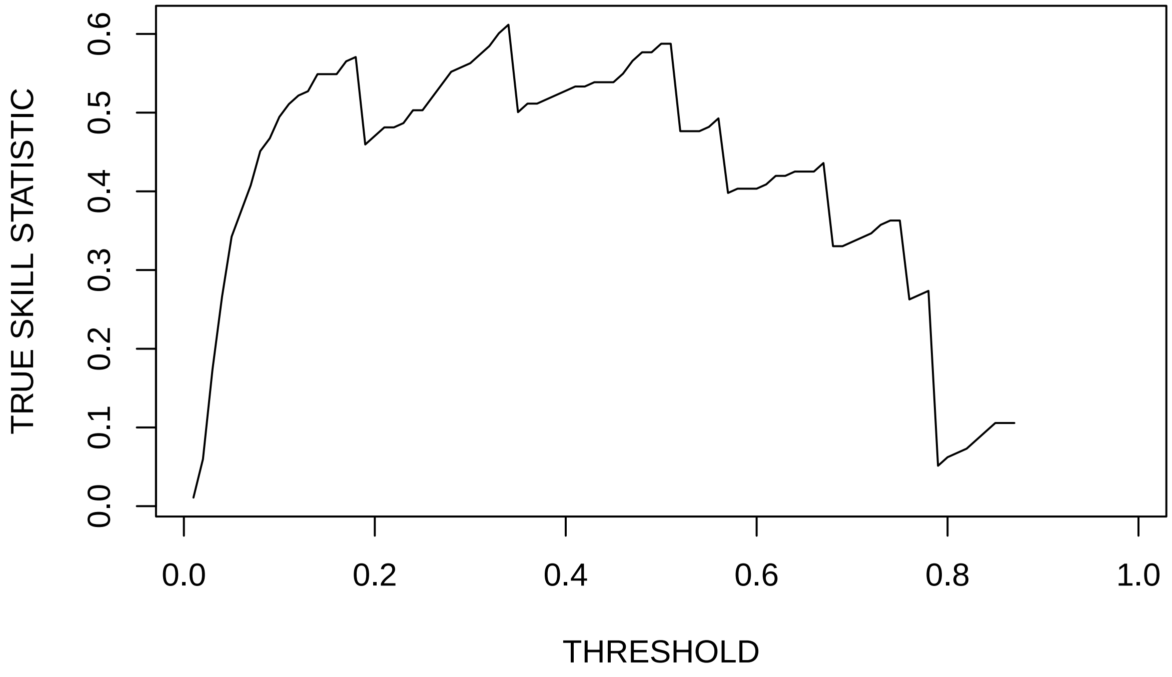
```
## [1] 0.73
```

MaxTSS and MaxKappa can be estimated via the functions *ecospat.max.tss* and *ecospat.max.kappa* and the variations of TSS and Kappa metric on a threshold can be done with *ecospat.plot.tss* adn *ecospat.plot.kappa*

```
MaxTSS <- ecospat.max.tss(Pred=pred.test$Fit_GLM,
                          Sp.occ = pred.test$resp)
MaxTSS$max.TSS
```

```
## [1] 0.611715
```

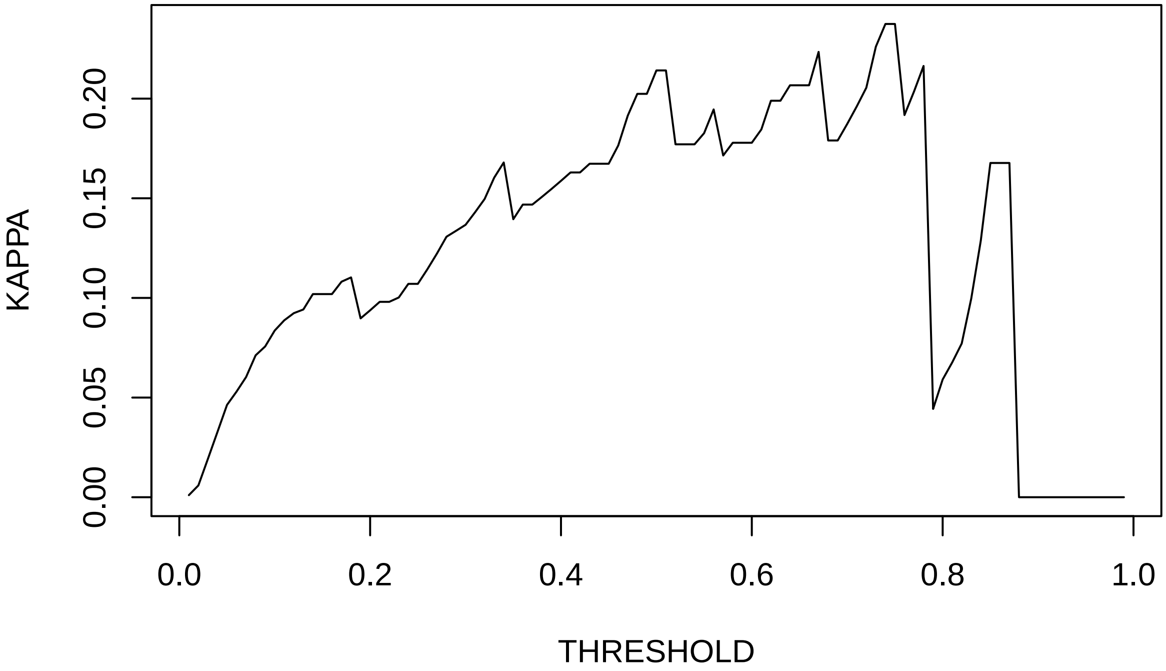
```
ecospat.plot.tss(Pred=pred.test$Fit_GLM,
                 Sp.occ = pred.test$resp)
```



```
MaxKappa <- ecospat.max.kappa(Pred=pred.test$Fit_GLM,
                              Sp.occ = pred.test$resp)
MaxKappa$max.Kappa
```

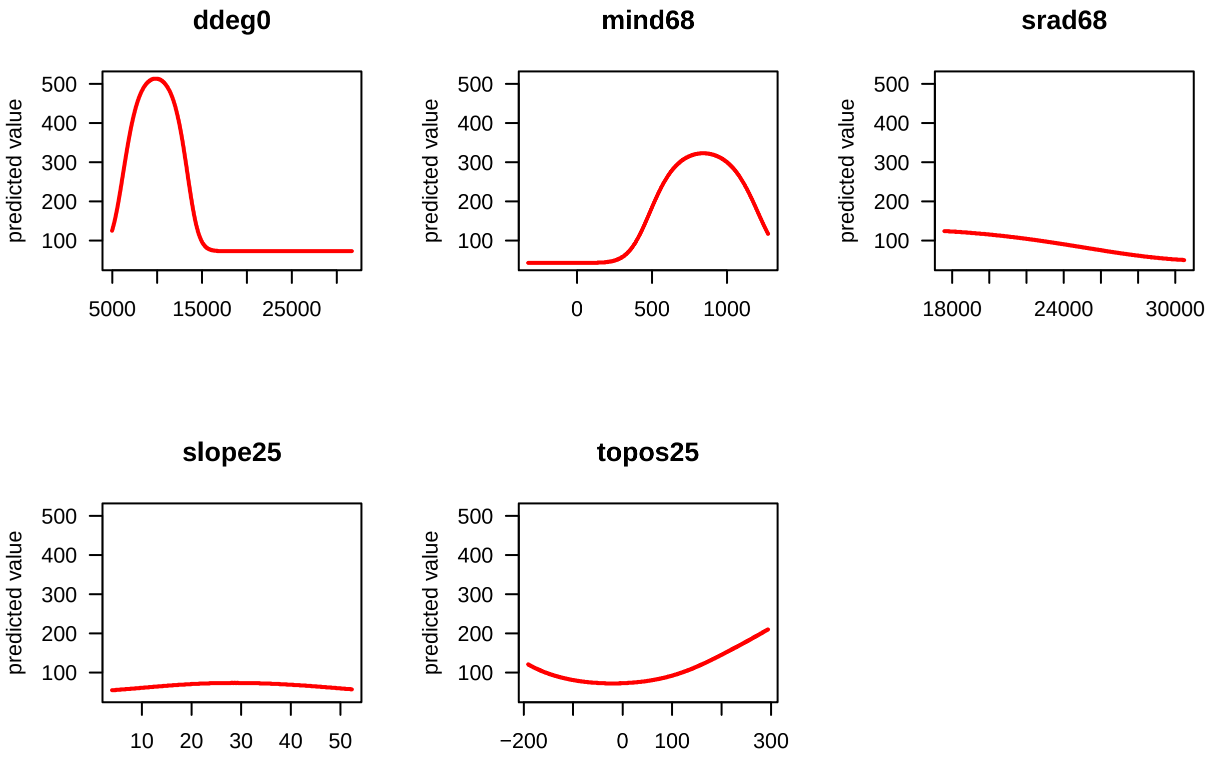
```
## [1] 0.2496419
```

```
ecospat.plot.kappa(Pred=pred.test$Fit_GLM,
                   Sp.occ = pred.test$resp)
```



Model performances can also be checked by observing species response curves to each environmental predictors. To do so, The function *ecospat.ESM.responsePlot* can be used. This function is an adaptation of the Evaluation Strip method proposed by Elith et al.(2005, *Ecological Modelling*) and needs the objects returned by *ecospat.ESM.Modeling* and *ecospat.ESM.EnsembleModeling*. The statistic used to keep constant the other predictor while generated the response curve for a predictor can be changed via the argument *fixed.var.metric*

```
response.plots <- ecospat.ESM.responsePlot(ESM.EnsembleModeling.output =
                                              my.ESM.EF,
                                           ESM.modeling.output = my.ESM,
                                           fixed.var.metric = 'median')
```



To check the contribution of each variable, you can use the function *ecospat.ESM.VarContrib*. This function computes the ratio between sum of weights of bivariate models where a focal variable was used and sum of weights of bivariate models where the focal variable was not used. The ratio is corrected for the number of models with or without the focal variable. This ratio gives an indication on the proportional contribution of the variable in the final ensemble model. A value of higher than 1 indicate that the focal variable has a higher contribution than average. For ensemble model, a weighted mean is apply among model algorithms.

```
var.contrib <- ecospat.ESM.VarContrib(ESM.modeling.output = my.ESM,
                                      ESM_EF.output = my.ESM.EF)
var.contrib
```

Table 3: Table X: Variable contributions to ESMs

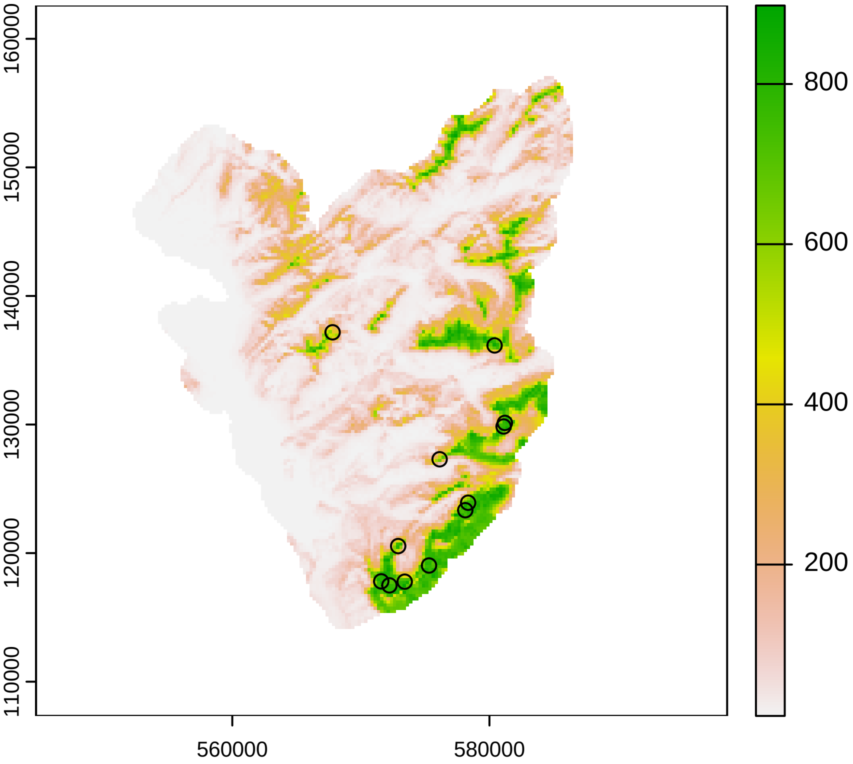
	GLM
ddeg0	1.5956454
mind68	1.3260098
srad68	0.8728915
slope25	0.5631219
topos25	0.8981928

After checking the model performances and the response curves, models can be projected using two functions: *ecospat.ESM.Projection* which projects each bivariate models and *ecospat.ESM.EnsembleProjection* which generates the ensemble of these bivariate models. *new.env* argument allows to project models onto a new data.frame, *SpatRaster* of *RasterStack* of ecological values while *name.env* allows to give a name to the projection. Parallel computing can be enabled with the argument *parallel*

```
### Projection of simple bivariate models into new space
my.ESM.proj.current <- ecospat.ESM.Projection(ESM.modeling.output = my.ESM,
                                              new.env = env,
                                              name.env = "currentTime",
                                              parallel = FALSE,
                                              cleanup = FALSE)

### Projection of calibrated ESMs into new space
my.ESM.EF.proj.current <- ecospat.ESM.EnsembleProjection(
                                                  ESM.prediction.output =
                                                    my.ESM.proj.current,
                                                  ESM.EnsembleModeling.output =
                                                    my.ESM.EF,
                                                  chosen.models = "all")
```

```
## projected ESM of Veronica alpina at present time
plot(my.ESM.EF.proj.current)
points(xy[sp_occ==1,],col="black")
```



iii. Post-Modelling

ESM projections can be afterwards binarized. To binarize these maps, diverse thresholds can be computed via the function *ecospat.ESM.threshold*. This function also provides evaluation scores for the full model (thus, evaluating the fit of the model but not the transferability).

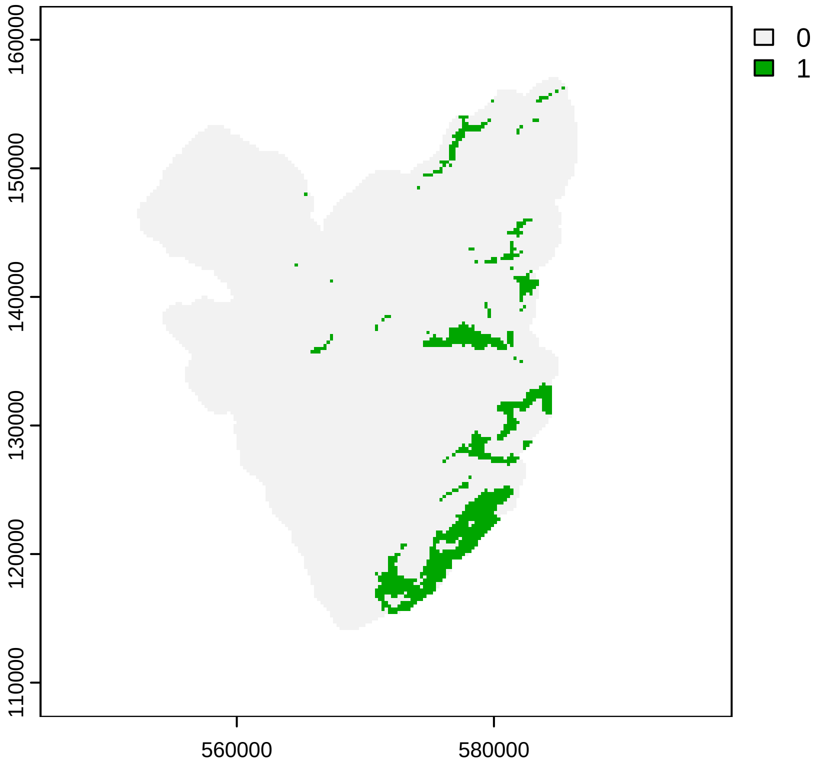
```
Thr <- ecospat.ESM.threshold(ESM.EnsembleModeling.output = my.ESM.EF,
                             PEplot = FALSE)
```

Table 4: Table X: Various threshold and fit performances of ESM

model	sensitivity	specificity	Kappa	AUC	sensitivity.sd	specificity.sd	Kappa.sd	AUC.sd	SomersD	Boyce	TSS	TSS.th	MPA1.0	MPA0.95	MPA0.90	Boyce.th.min	Boyce.th.max
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.386	0.737	0.776	[illegible]	[illegible]

Model projections can be afterwards binarized with the function *ecospat.binary.model* which need in the arguments *Pred*, a spatial grid and *Threshold*, the value of the threshold.

```
### Binarization of the projected ESM based on the threshold maximizing the TSS
my.ESM.EF.proj.current.bin <- ecospat.binary.model(Pred = my.ESM.EF.proj.current,
                                                   Threshold = (Thr$TSS.th)*1000)
plot(my.ESM.EF.proj.current.bin)
```

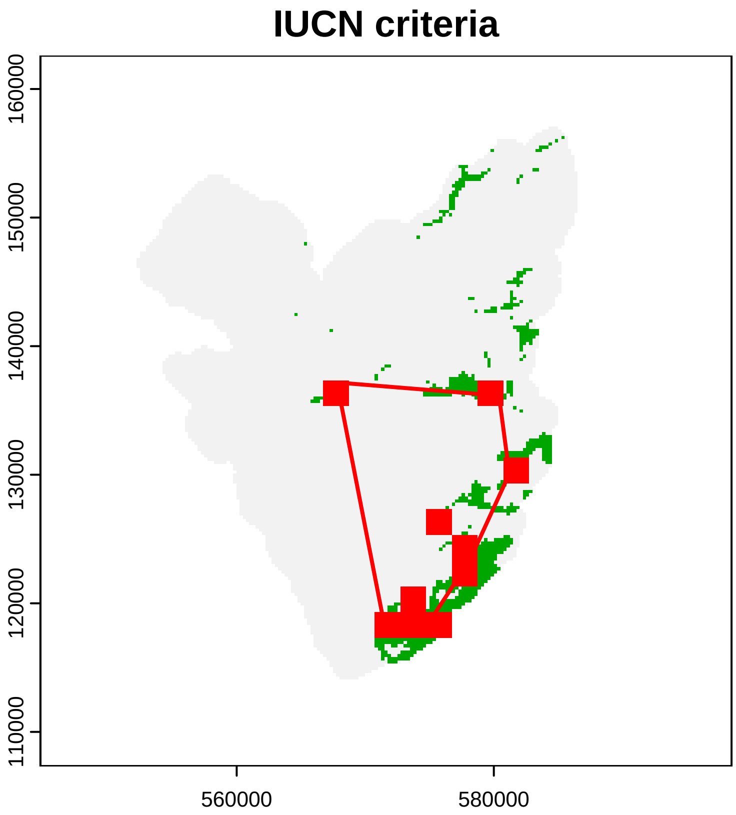


After binarizing maps, one could quantify the species range size or the occupied patches from ESM maps and IUCN criteria. In the ecospat package, the function *ecospat.rangesize* and *ecospat.occupied.patch* are made for these purposes.

More precisely, *ecospat.rangesize* allows quantifying Area of Occupancy AOO and the Extent of Occurence EOO. Numerous parameters are available and are describe when running in R ?ecospat.rangesize

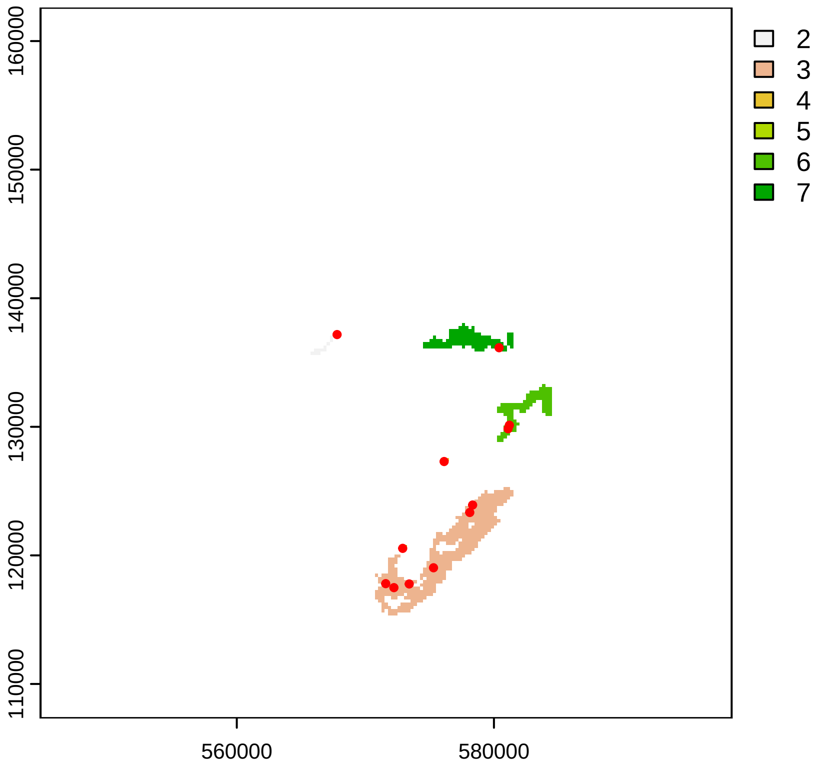
```
rangesize <- ecospat.rangesize(my.ESM.EF.proj.current.bin,
                               xy=xy[sp_occ==1,],
                               AOO.circles = TRUE,
                               lonlat =F)

plot(my.ESM.EF.proj.current.bin,legend=FALSE, main="IUCN criteria")
plot(rangesize$RangeObjects$AOO,add=TRUE, col="red",legend=FALSE)
plot(rangesize$RangeObjects$EOO@polygons,add=TRUE, border="red", lwd=2)
```



ecospat.occupied.patch quantified the number of patches where species occupied based on species distribution predictions, species occurrences and a buffer value (in meter) around species occurrences.

```
ocp  <- ecospat.occupied.patch(my.ESM.EF.proj.current.bin,
                               xy[sp_occ==1,],
                               buffer = 500)
plot(ocp) ## occupied patches: green area
points(xy[sp_occ==1,],col="red",cex=0.5,pch=19)
```



Breiner, F. T., A. Guisan, A. Bergamini, and M. P. Nobis. 2015. “Overcoming Limitations of Modelling Rare Species by Using Ensembles of Small Models.” *Methods in Ecology and Evolution* 6 (10): 1210–18.

Breiner, Frank T., Michael P. Nobis, Ariel Bergamini, and Antoine Guisan. 2018. “Optimizing Ensembles of Small Models for Predicting the Distribution of Species with Few Occurrences.” Edited by Nick Isaac. *Methods in Ecology and Evolution* 9 (4): 802–8. https://doi.org/10.1111/2041-210x.12957.

Collart, Flavien, and Antoine Guisan. 2023. “Small to Train, Small to Test: Dealing with Low Sample Size in Model Evaluation.” *Ecological Informatics* 75 (July): 102106. https://doi.org/10.1016/j.ecoinf.2023.102106.

Jiménez-Valverde, Alberto. 2020. “Sample Size for the Evaluation of Presence-Absence Models.” *Ecological Indicators* 114 (July): 106289. https://doi.org/10.1016/j.ecolind.2020.106289.

Lomba, A., L. Pellissier, C. Randin, J. Vicente, F. Moreira, J. Honrado, and A. Guisan. 2010. “Overcoming the Rare Species Modelling Paradox: A Novel Hierarchical Framework Applied to an Iberian Endemic Plant.” *Biological Conservation* 143 (11): 2647–57. https://doi.org/10.1016/j.biocon.2010.07.007.

Proosdij, André S. J. van, Marc S. M. Sosef, Jan J. Wieringa, and Niels Raes. 2015. “Minimum Required Number of Specimen Records to Develop Accurate Species Distribution Models.” *Ecography* 39 (6): 542–52. https://doi.org/10.1111/ecog.01509.