Tutorial for SDMPlay: 3/ Importance of model calibration

2021-02-26

Species distribution modelling (SDM) has been developed for several years to address conservation issues, to assess the direct impact of human activities on ecosystems and to predict the potential distribution shifts of invasive species (see Elith et al. 2006, Pearson 2007, Elith and Leathwick 2009). SDM relates species occurrences with environmental information and can predict species distribution on their entire occupied space. **This approach has been increasingly applied to Southern Ocean case studies, but requires corrections in such a context, due to the broad scale area, the limited number of presence records available and the spatial and temporal aggregations of these datasets.**

SDMPlay is a pedagogic package that will allow you to compute SDMs, to understand the overall method, and to produce model outputs. The package, along with its associated vignettes, highlights the different steps of model calibration and describes how to choose the best method to generate accurate and relevant outputs. SDMPlay proposes codes to apply a popular machine learning approach, BRT (Boosted Regression Trees) and introduces MaxEnt (Maximum Entropy). It contains occurrences of marine species and environmental descriptor datasets as examples associated with several vignette tutorials.

Objectives of tutorial #3/ Importance of model calibration When generating a model, an ensemble of elements should be checked to properly implement the model and generate the most accurate predictions. This implies to perform several tests to ensure that the method is appropriate to each modelled case study. This tutorial aims at illustrating the influence of the number of presence records on predictions and the influence of georeferencing errors that may be contained within presence datasets. It also highlights the importance of selecting the number and location of background data accordingly to each case study. An example of the influence of spatial resolution of environmental descriptors on model outputs is also given. Finally, an analysis to illustrate the consequences of the choice of BRT parameters is presented.

See also...

- **Tutorial #1/ Compute Species Distribution Models**
  Focusses on data structure, data preparation and general model computing.
- **Tutorial #2/ SDM outputs**
  Presents the main outputs you can generate with your SDM.
- **Tutorial #4/ Spatial cross-validation**
  Cross-validation is a method to validate your model. When working with presence data spatially aggregated, the cross-validation procedure should be adapted. This tutorial provides some elements to apply this method (refering to Guillaumot et al. 2019).
- **Tutorial #5/ Spatial extrapolation**
  Models can extrapolate when projected on broad scale areas. This tutorial provides codes to calculate extrapolation scores and generate extrapolation maps that could be associated to SDM maps (refering to Guillaumot et al. 2020).

Let's start with Tutorial #3 !...

Influence of the number of presence data and georeferencing errors on model outputs

The number of presence records used to generate a SDM is crucial since it characterises the potential of the dataset to describe the species occupied space (i.e. the ability of accurately representing the species ecology).

The following example highlights the influence of the number of presence-only records on model outputs. The distribution of the sea urchin *Ctenocidaris nutrix* is modelled using either the entire dataset available (125 records) or a subset of this dataset (60% of these data = 75 records).

```
library(SDMPlay)
#open occurrence records
data("ctenocidaris.nutrix") # Species distributed on the Kerguelen Plateau,
                            # table with longitude and latitude only
all_data <- ctenocidaris.nutrix[,c("decimal.Longitude","decimal.Latitude")]

#subset a part of the dataset
data_60percent <- all_data[sample(nrow(all_data), nrow(all_data)*60/100, replace=F),]
```

```
# open environmental layers
library(raster)
data("worldmap")
data(predictors2005_2012)
names(predictors2005_2012)
```

```
##  [1] "depth"
##  [2] "seasurface_temperature_mean_2005_2012"
##  [3] "seasurface_temperature_amplitude_2005_2012"
##  [4] "seafloor_temperature_mean_2005_2012"
##  [5] "seafloor_temperature_amplitude_2005_2012"
##  [6] "seasurface_salinity_mean_2005_2012"
##  [7] "seasurface_salinity_amplitude_2005_2012"
##  [8] "seafloor_salinity_mean_2005_2012"
##  [9] "seafloor_salinity_amplitude_2005_2012"
## [10] "chlorophyla_summer_mean_2005_2012"
## [11] "geomorphology"
## [12] "sediments"
## [13] "slope"
## [14] "seafloor_oxygen_mean_2005_2012"
## [15] "roughness"
```

```
# create your input matrices
SDMtable_all_data<- SDMPlay:::SDMtab(xydata=all_data,predictors=predictors2005_2012,
                                     unique.data=FALSE,same=TRUE)

SDMtable_data_60percent<- SDMPlay:::SDMtab(xydata=data_60percent,
                                           predictors=predictors2005_2012,
                                           unique.data=FALSE,same=TRUE)
```

```
# Launch models
Model_all_data <- SDMPlay:::compute.brt(x=SDMtable_all_data,
                                        proj.predictors=predictors2005_2012,
                                        tc = 2, lr = 0.001, bf = 0.75, n.trees = 500)
```

```
Model_data_60percent <- SDMPlay:::compute.brt(x=SDMtable_data_60percent,
                                              proj.predictors=predictors2005_2012,
                                              tc = 2, lr = 0.001, bf = 0.75, n.trees = 500)
```

```r
# Compare maps (Fig. 1)
palettecolor <- colorRampPalette(c("deepskyblue", "darkseagreen","lightgreen","green",
                                   "yellow","gold","orange", "red","firebrick"))(100)

par(mfrow=c(1,2))
plot(Model_all_data$raster.prediction,col=palettecolor, zlim=c(0,1),
     main="Projection for all data",
     cex.axis= 0.7,
     legend.width=0.5, legend.shrink=0.5,
     legend.args=list(text='Distribution probability', side=3, font=2))
points(worldmap, type="l")

plot(Model_data_60percent$raster.prediction,col=palettecolor, zlim=c(0,1),
     main="Projection for 60% of data",
     cex.axis= 0.7, legend=FALSE)
points(worldmap, type="l")
```

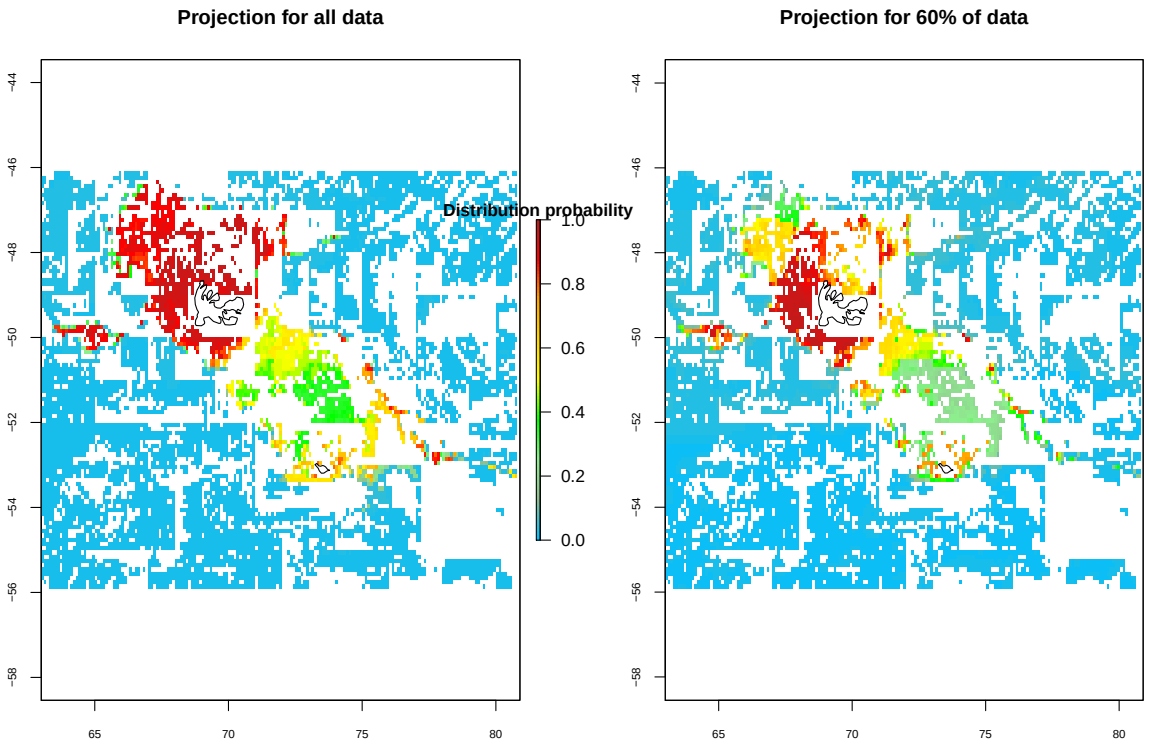

Figure 1: Comparison of model predictions for Ctenocidaris nutrix (2005-2012 environmental conditions), using all presence data available (left) or a subset (60% of data, right).

We can see that model predictions are different when using a different number of presence records to generate the model (Fig. 1). These shifts are expected to be even more pronounced if the presence-only records would not be as aggregated in a same area as they are in this example.

Now, let's generate a second analysis where georeferencing errors are introduced in the dataset (15% of the total presence records are modified, i.e. 19 presence records). Model outputs are compared (Fig. 2). The differents steps are similar to the previous analysis (and skipped from screen).

```
# Generate random errors in the presence records dataset
all_data <- ctenocidaris.nutrix[,c("decimal.Longitude","decimal.Latitude")]
idx <- sample.int(nrow(all_data), size= round(nrow(all_data)*15/100), replace=F )
data_15percent <- all_data[idx,]
remains <- all_data[-idx,]

perc15_data_errors <- matrix(data = NA, ncol=2, nrow=nrow(data_15percent))
colnames(perc15_data_errors) <- colnames(all_data)

for (i in 1:nrow(data_15percent)){
  perc15_data_errors[i,1] <- data_15percent[i,1]+runif(1,1,4) # random changes
                                # between 1 and 4° in latitude or longitude
  perc15_data_errors[i,2] <- data_15percent[i,2]+runif(1,1,4)
}
all_data_errors <- rbind(perc15_data_errors,remains)
```

```
# Create your input matrices
SDMtable_data_errors<- SDMPlay:::SDMtab(xydata=all_data_errors,predictors=predictors2005_2012,
                                        unique.data=FALSE,same=TRUE)

# Launch models
Model_data_errors <- SDMPlay:::compute.brt(x=SDMtable_data_errors,
                                           proj.predictors=predictors2005_2012,
                                           tc = 2, lr = 0.001, bf = 0.75, n.trees = 500)
```

```
# Compare maps (Fig. 2)
par(mfrow=c(1,2))
plot(Model_all_data$raster.prediction,col=palettecolor, zlim=c(0,1),
     main="Projection for all data",
     cex.axis= 0.7,
     legend.width=0.5, legend.shrink=0.25,
     legend.args=list(text='Distribution probability', side=3, font=2))
points(worldmap, type="l")

plot(Model_data_errors$raster.prediction,col=palettecolor, zlim=c(0,1),
     main="With 15% of georeferencing errors",
     cex.axis= 0.7, legend=FALSE)
points(worldmap, type="l")
```

These examples highlight the importance of gathering an exhaustive presence dataset and to check for georeferecing errors within these datasets (because they are often a combinaison between several sources). Model outputs can be more or less influenced by these biases.

Influence of the number and distribution of background data on model outputs

In the context of calibrating SDM based on presence-only records, background records are used to represent the environmental boundaries on which the model will be projected. The choice of the number of presence records has an important influence on model outputs and contrasts between the chosen algorithm (Barbet-Massin et al. 2012). For BRT, it is adviced to use a number of background records as close as possible as the number of available presence records (Barbet-Massin et al. 2012). We will illustrate contrasts between models outputs for models calibrated with 125 or 500 background records.

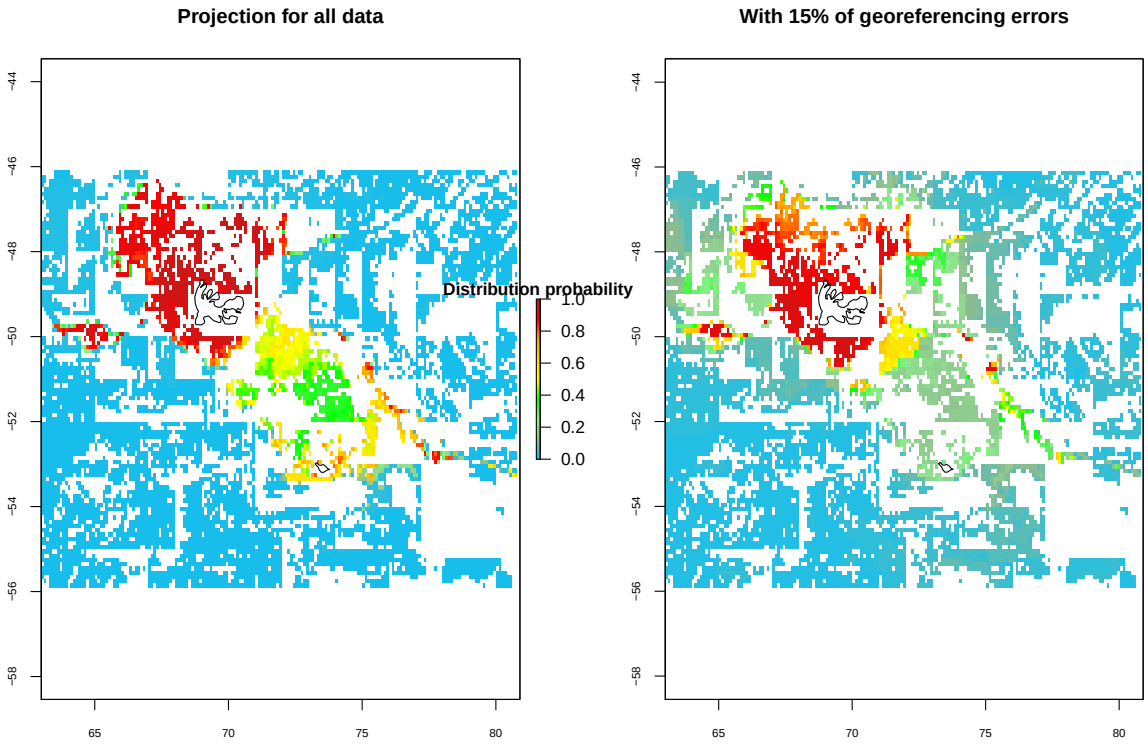

Figure 2: Comparison of model predictions for Ctenocidaris nutrix (2005-2012 environmental conditions), using the original dataset (left) or a dataset with 15% georeferencing errors (right)

```
# Create your input matrices
SDMtable_125bgr<- SDMPlay:::SDMtab(xydata=all_data,
                                   predictors=predictors2005_2012,
                                   unique.data=FALSE, same=TRUE)

SDMtable_500bgr<- SDMPlay:::SDMtab(xydata=all_data,
                                   predictors=predictors2005_2012,
                                   unique.data=FALSE, same=FALSE, background.nb=500)
```

```
# Launch models
Model_125bgr <- SDMPlay:::compute.brt(x=SDMtable_125bgr,
                                      proj.predictors=predictors2005_2012,
                                      tc = 2, lr = 0.001, bf = 0.75, n.trees = 500)
```

```
Model_500bgr <- SDMPlay:::compute.brt(x=SDMtable_500bgr,
                                      proj.predictors=predictors2005_2012,
                                      tc = 2, lr = 0.001, bf = 0.75, n.trees = 500)
```

```
# Compare maps (Fig. 3)
par(mfrow=c(1,2))
plot(Model_125bgr$raster.prediction,col=palettecolor, zlim=c(0,1),
     main="Model 125 background records",
     cex.axis= 0.7,
     legend.width=0.5, legend.shrink=0.25,
     legend.args=list(text='Distribution probability', side=3, font=2))
points(worldmap, type="l")

plot(Model_500bgr$raster.prediction,col=palettecolor, zlim=c(0,1),
     main="Model 500 background records",
     cex.axis= 0.7, legend=FALSE)
points(worldmap, type="l")
```

Another point: background data sampling can be methodologically used to reduce the influence of spatial aggregation of presence records on model outputs (Phillips et al. 2009). Several methods have been developed so far (Phillips et al. 2009, Guillaumot et al. 2018). One of them consists in building a kernel density estimation (KDE) layer to represent the spatial sampling bias, based on visited areas for example (Venables and Ripley 2002). Background data are then sampled according to the weighting scheme of the KDE layer, in order to reduce discrepancies between presence-only records and background data (Phillips et al. 2009, Barbet-Massin et al. 2012, Guillaumot et al. 2018). The comparison between models generated with and without 'targeted' background data samples is illustrated below.

```
# Create a KDE layer
# As explained above, you can use a set of data that describes the pixels
# that have been visited so far or use the distribution of another species
# to generate your KDE.
# We will provide here a simple construction based on the species presence records

KDE_layer <- raster(MASS::kde2d(all_data$decimal.Longitude,
                                all_data$decimal.Latitude,
                                n=c(ncol(predictors2005_2012),
                                    nrow(predictors2005_2012)),
                                lims=c( 63, 81, -56,-46))) # number of pixels and
```

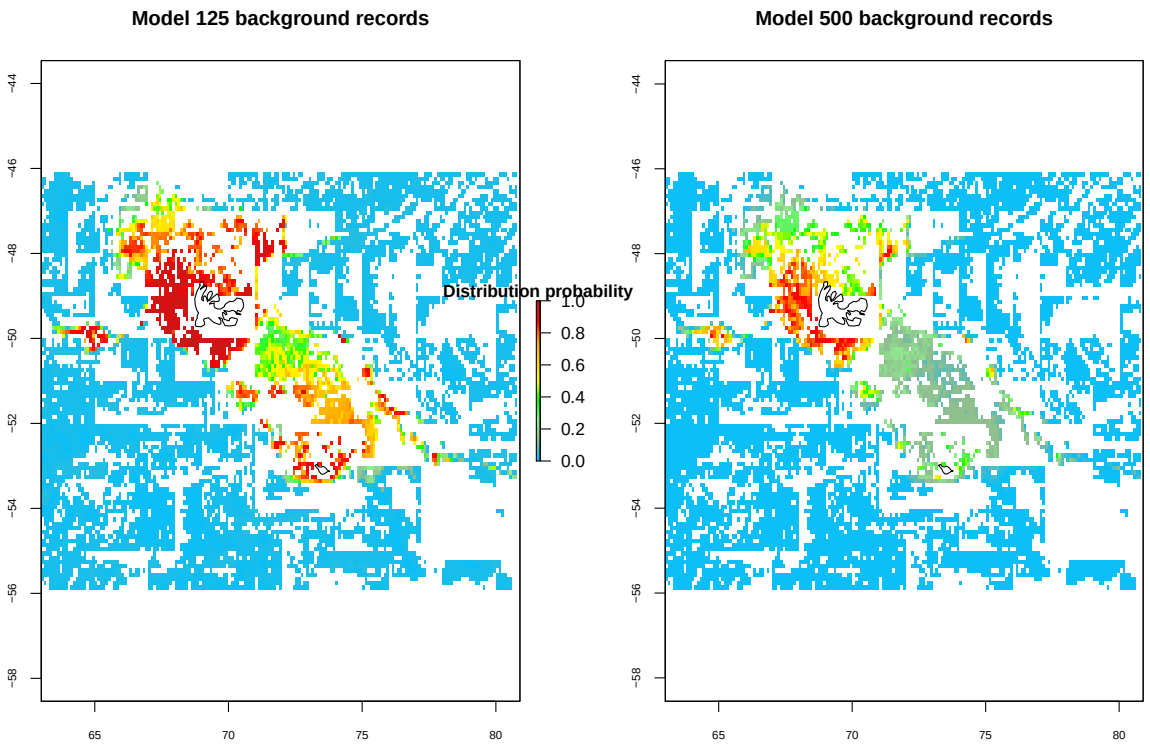

Figure 3: Comparison of model predictions for Ctenocidaris nutrix (2005-2012 environmental conditions), using 125 or 500 background records to calibrate the model. Contrasts between model predictions are clearly important.

```
                                        #extent (latitude/longitude) of the projection area

extent(KDE_layer) <- extent(predictors2005_2012)
KDE_layer <- mask(KDE_layer, subset(predictors2005_2012,1)) # mask the KDE layer by
                                                            # NA values (continental areas)
# Plot the KDE layer (Fig.4)
plot(KDE_layer)
points(all_data, pch=20)
points(worldmap, type="l")
```

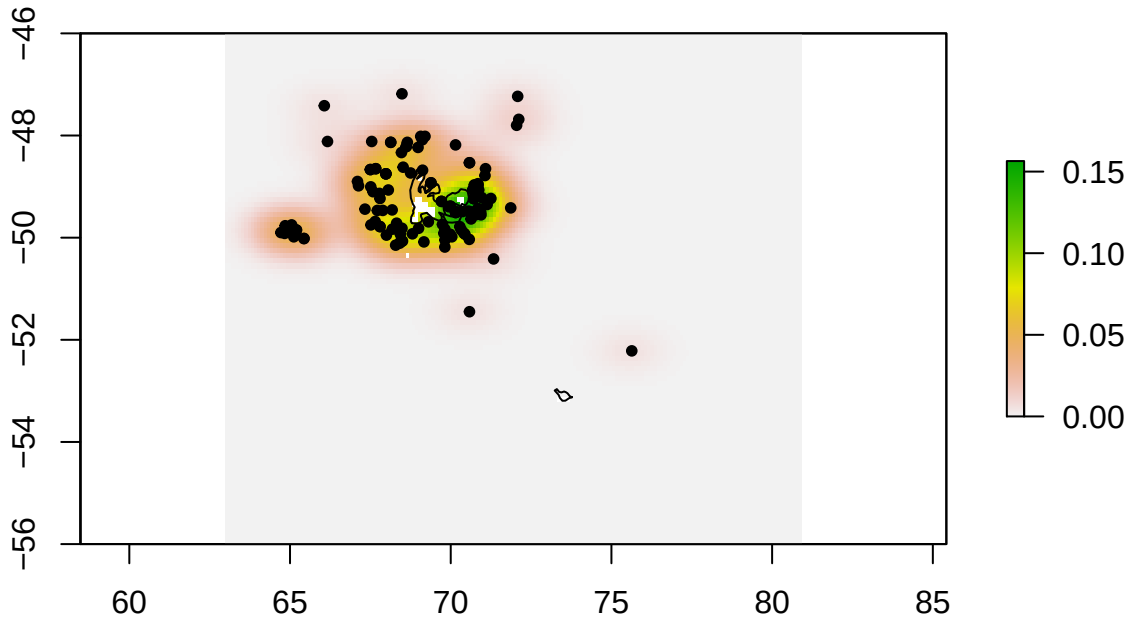

Figure 4: KDE layer

```
# Create your matrice and generate your model
library(SDMPlay)
library(raster)
library(dismo)

SDMtable_KDE<- SDMPlay:::SDMtab(xydata=all_data, predictors=predictors2005_2012,
                                unique.data=FALSE,same=TRUE, KDE=KDE_layer)
background_detail_KDE <- subset(SDMtable_KDE, SDMtable_KDE$id==0)[,c(2,3)]
background_detail <- subset(SDMtable_all_data,SDMtable_all_data$id==0)[,c(2,3)]

Model_KDE <- SDMPlay:::compute.brt(x=SDMtable_KDE,
                                   proj.predictors=predictors2005_2012,
```

```
                                    tc = 2, lr = 0.001, bf = 0.75, n.trees = 500)
```

```
# Compare background records samples (Fig. 5)
bluepalette<-colorRampPalette(c("blue4","blue","dodgerblue", "deepskyblue","lightskyblue"))(800)

par(mfrow=c(1,2))
plot(subset(predictors2005_2012,1), col=bluepalette, main="Background sampling without KDE",
     cex.axis= 0.7)
points(worldmap, type="l")
points(background_detail, pch=20)

plot(subset(predictors2005_2012,1), col=bluepalette, main="Background sampling with KDE",
     cex.axis= 0.7)
points(worldmap, type="l")
points(background_detail_KDE, pch=20)
```

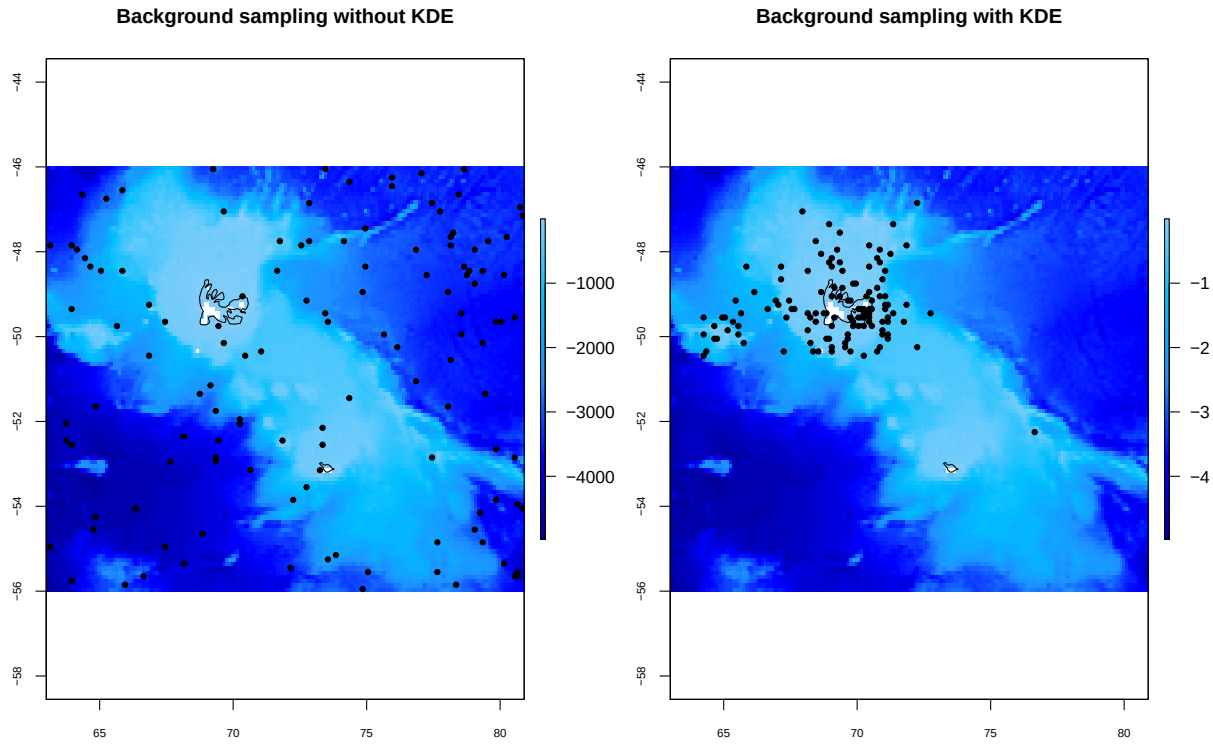

Figure 5: Comparison of background data samplings without (left) or with (right) KDE sampling.

```
# Compare SDM maps (Fig. 6)
par(mfrow=c(1,2))
plot(Model_all_data$raster.prediction,col=palettecolor, zlim=c(0,1), main="Model without KDE",
     cex.axis= 0.7,
     legend.width=0.5, legend.shrink=0.25,
     legend.args=list(text='Distribution probability', side=3, font=2))
points(worldmap, type="l")
```

```
plot(Model_KDE$raster.prediction,col=palettecolor, zlim=c(0,1), main="Model with KDE ",
     cex.axis= 0.7, legend=FALSE)
points(worldmap, type="l")
```

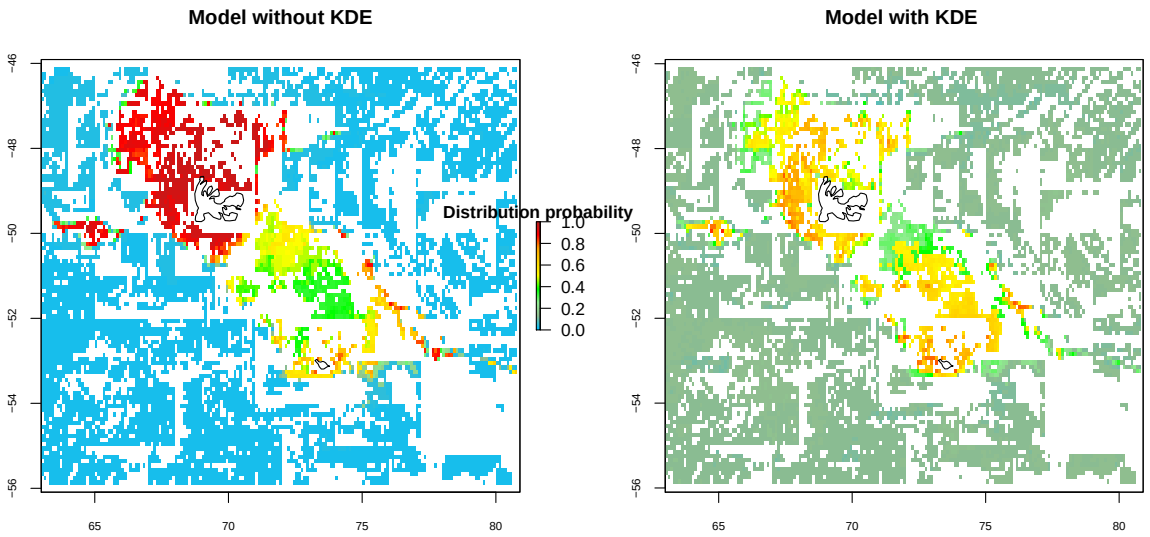

Figure 6: Comparison of model predictions, with background data sampled without (left) or with (right) a KDE scheme

Influence of the spatial resolution of environmental descriptors on model outputs

When generating a SDM, one of the first step is to collect and select the list of environmental descriptors that will be used for the modelling analysis (see Tutorial #1). The choice of the grid-cell pixel resolution has its importance for prediction accuracy, relevance and interpretation. Let's compare two model outputs if calibrated with environmental descriptors at 0.1° resolution or 10 times coarser.

```
# Create environmental predictors with a spatial resolution 10 times coarser (Fig. 7)
predictors2005_2012_10X <- raster::aggregate(predictors2005_2012, 10, na.rm=T)
plot(subset(predictors2005_2012_10X, c(1:3)))
```

```
# Create the SDMtab matrix
SDMtable_10X<- SDMPlay:::SDMtab(xydata=all_data,
                                predictors=predictors2005_2012_10X,
                                unique.data=FALSE,same=TRUE)
SDMtable_10X[is.nan(as.matrix(SDMtable_10X))]<- NA
```

```
# Launch the model
Model_10X <- SDMPlay:::compute.brt(x=SDMtable_10X,
                                   proj.predictors=predictors2005_2012_10X,
                                   tc = 2, lr = 0.001, bf = 0.75, n.trees = 500)
```

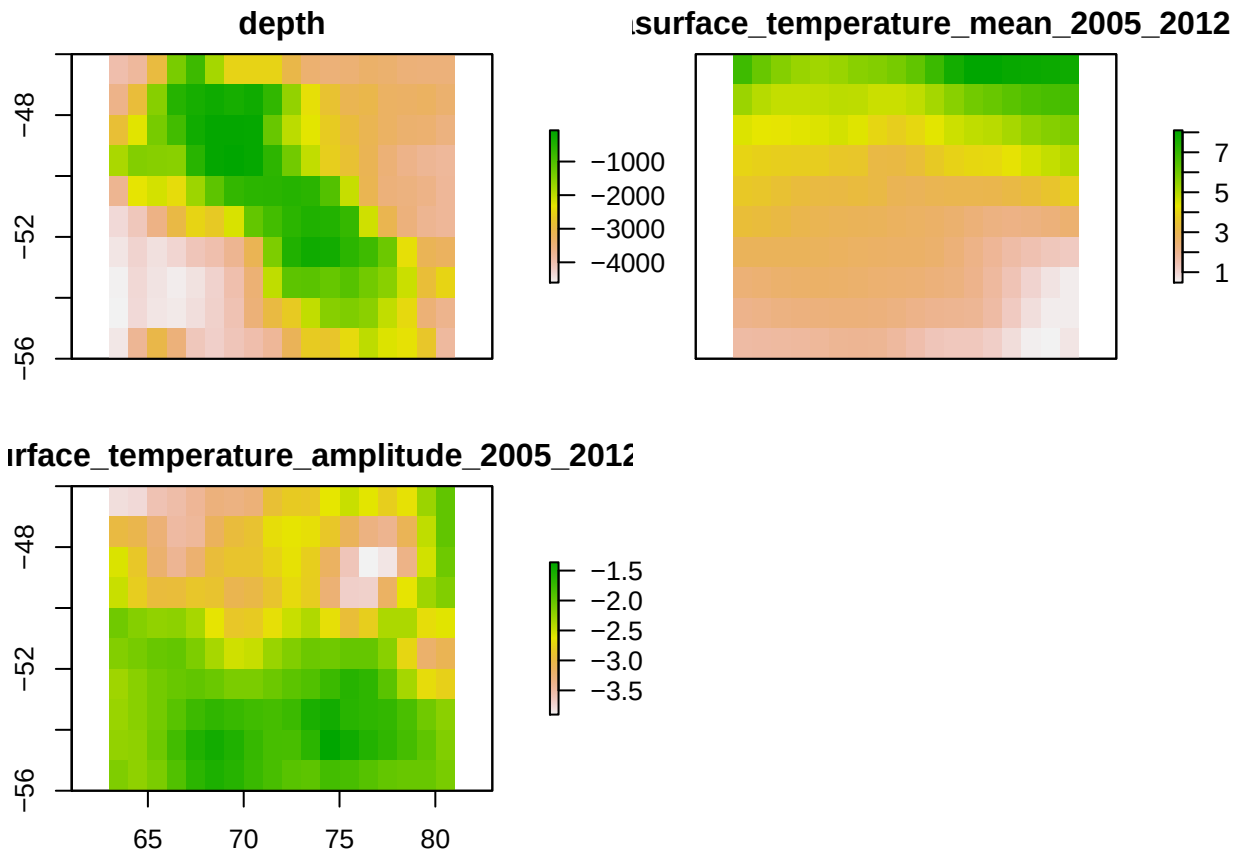

Figure 7: Environmental descriptors, resolution 10 times coarser (1°)

```
# Compare SDM maps (Fig. 8)
par(mfrow=c(1,2))
plot(Model_all_data$raster.prediction, col=palettecolor, zlim=c(0,1),
     main="Model at 0.1° resolution", cex.axis= 0.7,
legend.width=0.5, legend.shrink=0.25,
legend.args=list(text='Distribution probability', side=3, font=2))
points(worldmap, type="l")

plot(Model_10X$raster.prediction, col=palettecolor, zlim=c(0,1),
     main="Model at 1° resolution", cex.axis= 0.7, legend=FALSE)
points(worldmap, type="l")
```

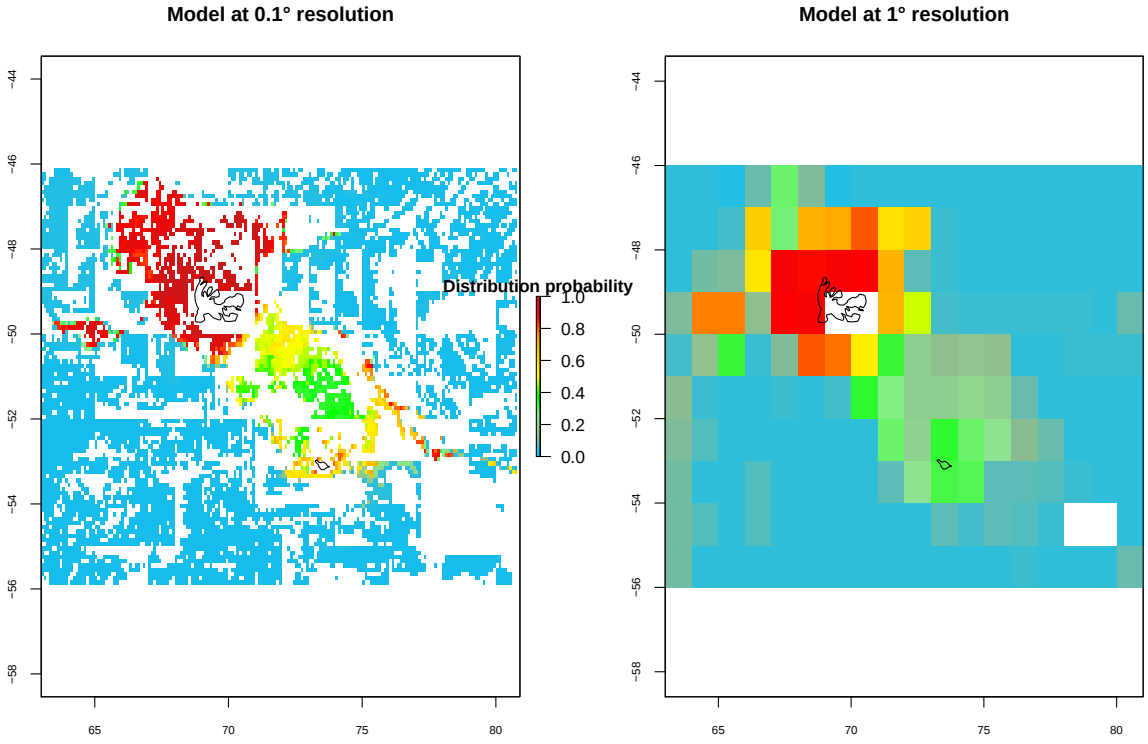

Figure 8: Compare predictions of Ctenocidaris nutrix with environmental descriptors at 0.1° resolution (left) and 1° resolution (right)

Influence of the choice of BRT parameters on model outputs

In Elith et al.(2008), a tutorial is provided to explain the importance of each BRT parameter in model construction (bag fraction bg, learning rate lr, tree complexity tc). A simple analysis can be run during model calibration in order to select the best combination of BRT parameters to generate the most accurate model.

```
library(dismo)
library(gbm)
```

```
# In order to perform this analysis, we need to go into details of the BRT calculations
# (that are made simpler when you use the compute.brt function). To do so, we will
# rather use the gbm.step function of the dismo package.

# Do several models with several BRT parameters
model1 <- dismo::gbm.step(data = SDMtable_all_data, gbm.x = 4:ncol(SDMtable_all_data),
                          gbm.y = 1, tree.complexity = 2, learning.rate = 0.001,
                          bag.fraction = 0.75, n.trees = 500, step.size = 500)

model2 <- dismo::gbm.step(data = SDMtable_all_data, gbm.x = 4:ncol(SDMtable_all_data),
                          gbm.y = 1, tree.complexity = 3, learning.rate = 0.0001,
                          bag.fraction = 0.75, n.trees = 500, step.size = 500)

model3 <- dismo::gbm.step(data = SDMtable_all_data, gbm.x = 4:ncol(SDMtable_all_data),
                          gbm.y = 1, tree.complexity = 3, learning.rate = 0.0005,
                          bag.fraction = 0.75, n.trees = 500, step.size = 500)

model4 <- dismo::gbm.step(data = SDMtable_all_data, gbm.x = 4:ncol(SDMtable_all_data),
                          gbm.y = 1, tree.complexity = 4, learning.rate = 0.0005,
                          bag.fraction = 0.85, n.trees = 500, step.size = 500)

tree.list1 <- seq(100, model1$gbm.call$best.trees, by = 100)

tree.list2 <- seq(100, model2$gbm.call$best.trees, by = 100)

tree.list3 <- seq(100, model3$gbm.call$best.trees, by = 100)

tree.list4 <- seq(100, model4$gbm.call$best.trees, by = 100)

pred1 <- predict.gbm(model1,SDMtable_all_data, n.trees = tree.list1, "response")
pred2 <- predict.gbm(model2,SDMtable_all_data, n.trees = tree.list2, "response")
pred3 <- predict.gbm(model3,SDMtable_all_data, n.trees = tree.list3, "response")
pred4 <- predict.gbm(model4,SDMtable_all_data, n.trees = tree.list4, "response")

# comparison of predictions with observed values : measure of predicted deviance
graphe.deviance1 <- rep(0,max(tree.list1)/100)
for (i in 1:length(graphe.deviance1)) {
  graphe.deviance1 [i] <- calc.deviance(SDMtable_all_data$id, pred1[,i],calc.mean=T)
}
graphe.deviance2 <- rep(0,max(tree.list2)/100)
for (i in 1:length(graphe.deviance2)) {
  graphe.deviance2 [i] <- calc.deviance(SDMtable_all_data$id, pred2[,i],calc.mean=T)
}
graphe.deviance3 <- rep(0,max(tree.list3)/100)
for (i in 1:length(graphe.deviance3)) {
  graphe.deviance3 [i] <- calc.deviance(SDMtable_all_data$id, pred3[,i],calc.mean=T)
}
graphe.deviance4 <- rep(0,max(tree.list4)/100)
for (i in 1:length(graphe.deviance4)) {
  graphe.deviance4 [i] <- calc.deviance(SDMtable_all_data$id, pred4[,i],calc.mean=T)
}
```

Finally, plot the predicted deviance of each of the case study according to the number of trees necessary to reach the lowest deviance (Fig. 9). This plot will serve as the decision criteria for the best set of parameter to choose: the set for which the predicted deviance is the lowest for the lowest number of trees will be the best (Elith et al. 2008).

```
# Figure 9
plot(tree.list1,graphe.deviance1,xlim = c(-100,10000),
     ylim=c(0,1),type='l', xlab = "Number of trees",
     ylab = "Predictive deviance", cex.lab = 1.5, col="black")
lines(tree.list2,graphe.deviance2,col="blue")
lines(tree.list3,graphe.deviance3,col="red")
lines(tree.list4,graphe.deviance4,col="green",
      xlab = "Number of trees",
      ylab = "Predictive deviance")
legend("topright",legend=c("tc2 lr0.001 bf=0.75","tc3 lr0.0001 bf=0.75",
                           "tc3 lr0.0005 bf=0.75","tc4 lr0.0005 bf=0.85"),
       col=c("black","blue","red", "green"),pch=16,cex=1.5)
```

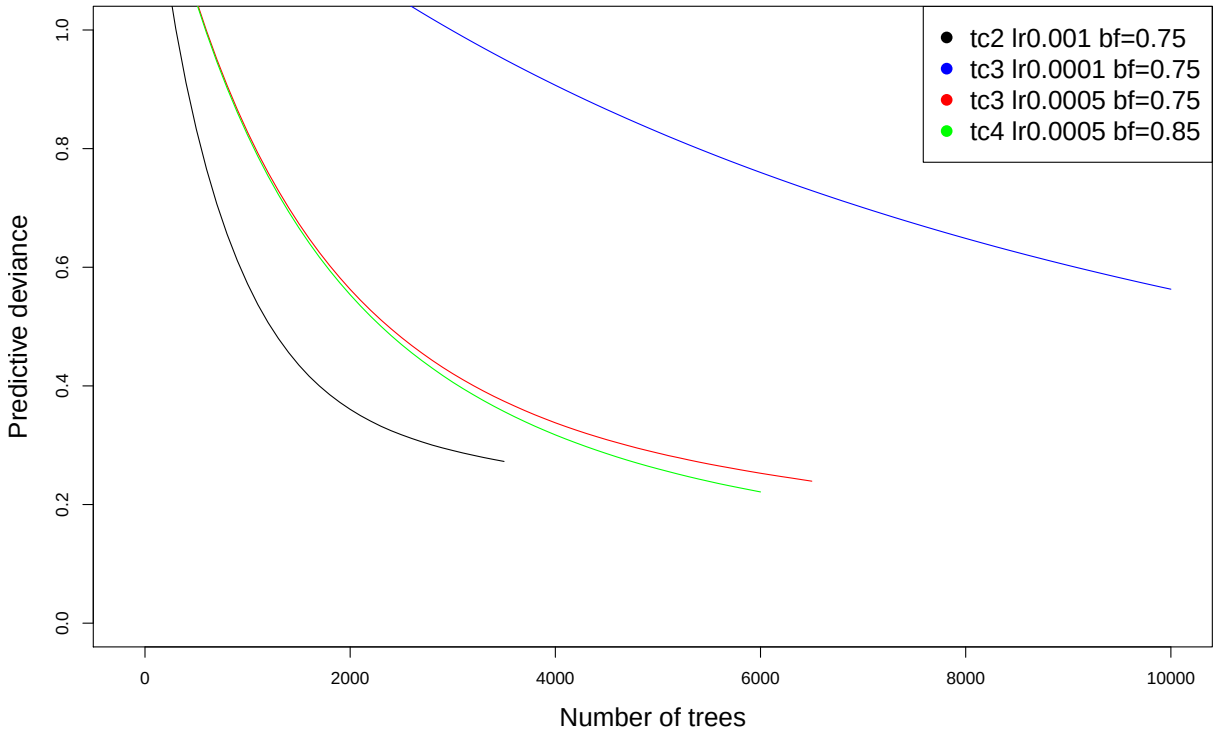

Figure 9: Procedure to choose the best set of BRT parameters (Elith et al. 2008). The set of parameters for which the predictive deviance is the lowest for the minimal number of trees is the optimal

Of course, you should generate more possibilities than these 4 models to figure out the best set of parameters. In this example, the blue line (Fig. 9) has the highest predictive deviance and consequently is not interesting, it also needs a (too) high number of trees to explain model deviance. That is because of the learning rate that has a very low value. Among the three other lines, they all reach more or less the same predicted deviance threshold (around 0.3) but the black line requires less trees to reach it. The best set of BRT parameters

(bg, tc, lr) is therefore the one corresponding to this black line (choosing the green one can also be justified as the predictive deviance is a bit lower). Refer to Elith et al. (2008) for more technical details about the influence of each BRT parameter.

REFERENCES

Barbet-Massin, M., Jiguet, F., Albert, C.H. & Thuiller, W. (2012). Selecting pseudo-absences for species distribution models: how, where and how many? *Methods in Ecology and Evolution*, 3(2), 327-338.

Elith, J., Anderson, R., Dudík, M., Ferrier, S., Guisan, A., J Hijmans, R., Huettmann, F., ... & A Loiselle, B. (2006). Novel methods improve prediction of species' distributions from occurrence data. *Ecography*, 29(2), 129-151.

Elith, J., Leathwick, J.R. & Hastie, T. (2008). A working guide to boosted regression trees. *Journal of Animal Ecology*, 77(4), 802-813.

Elith, J. & Leathwick, J. R. (2009). Species distribution models: ecological explanation and prediction across space and time. *Annual Review of Ecology, Evolution, and Systematics*, 40, 677-697.

Guillaumot, C., Martin, A., Eléaume, M. & Saucède, T. (2018). Methods for improving species distribution models in data-poor areas: example of sub-Antarctic benthic species on the Kerguelen Plateau. *Marine Ecology Progress Series*, 594, 149-164.

Guillaumot, C., Artois, J., Saucède, T., Demoustier, L., Moreau, C., Eléaume, M., ... & Danis, B. (2019). Broad-scale species distribution models applied to data-poor areas. *Progress in Oceanography*, 175, 198-207.

Guillaumot, C., Moreau, C., Danis, B. & Saucède, T. (2020). Extrapolation in species distribution modelling. Application to Southern Ocean marine species. *Progress in Oceanography*, 188, 102438.

Pearson, R.G. (2007). Species' distribution modeling for conservation educators and practitioners. *Synthesis. American Museum of Natural History*, 50.

Phillips, S.J., Dudík, M., Elith, J., Graham, C.H., Lehmann, A., Leathwick, J. & Ferrier, S. (2009). Sample selection bias and presence-only distribution models: implications for background and pseudo-absence data. *Ecological Applications*, 19(1), 181-197.

Venables, W.N. & Ripley, B.D. (2002). Random and mixed effects. *In Modern applied statistics with S (pp. 271-300). Springer*, New York, NY.