Practical 1 for BIOMOD
Building Models

LECA-CNRS, Université Joseph Fourier, Grenoble, France
BIOMOD : Wilfried Thuiller, Bruno Lafourcade, Robin Engler

July 9, 2010

Contents

0.1 Installation

In order to facilitate the learning of BIOMOD, a tutorial is provided here with artificial data. It is recommended that the user follows each step and run the models on these artificial datasets, or at least in parallel with runs on its own data. The completion of the tutorial should bring sufficient answers as for the usage of BIOMOD on other datasets.

To run BIOMOD, please use the latest version of R. A certain number of libraries are also required (rpart, MASS, gbm, gam, nnet, fda, randomForest, Design, Hmisc, plyr) and are also to be downloaded rom Rcran before attempting to run BIOMOD. Note that BIOMOD now enables to build projections directly on rasters. This recent innovation requires several more packages, even if you will not be using rasters with your own work. These are : foreign, sp, rgdal, raster, maptools, some of which are on Rcran and others on the R-forge website.

BIOMOD is a developping R package that is to be downloaded from this web page :

http://r-forge.r-project.org/R/?group_id=302

It is advised to check relatively frequently for updates.

The recommended procedure is to first create a working directory, for example called BIOMOD. Then, create a new folder where to store the datasets, run the models and save the outputs and results. In our examples, we will create and use the directory called Biomod_runs.

It is from this folder that the files will be read and written. For example, you need to put a copy of your datasets in order to be able to open them once the working directory in R is set to this workspace.

In the latter version of BIOMOD, the results are stored outside R's workspace to counter the memory storage limitations of the software. While running BIOMOD, you will realise that additional folders will be created. First, the *Models()* function will create 2 folders named *models* and *pred*. As you might have guessed, they will respectively contain the models and the current predictions. Then, the *Projection()* function will create a folder to store the outputs for each projection scenario that is run (refer to the second practical : 'Making Projections').

Once R is opened, the first thing to do is to load the BIOMOD package. It will load all the functions required to run BIOMOD as well as the examples files to be used in this practical.

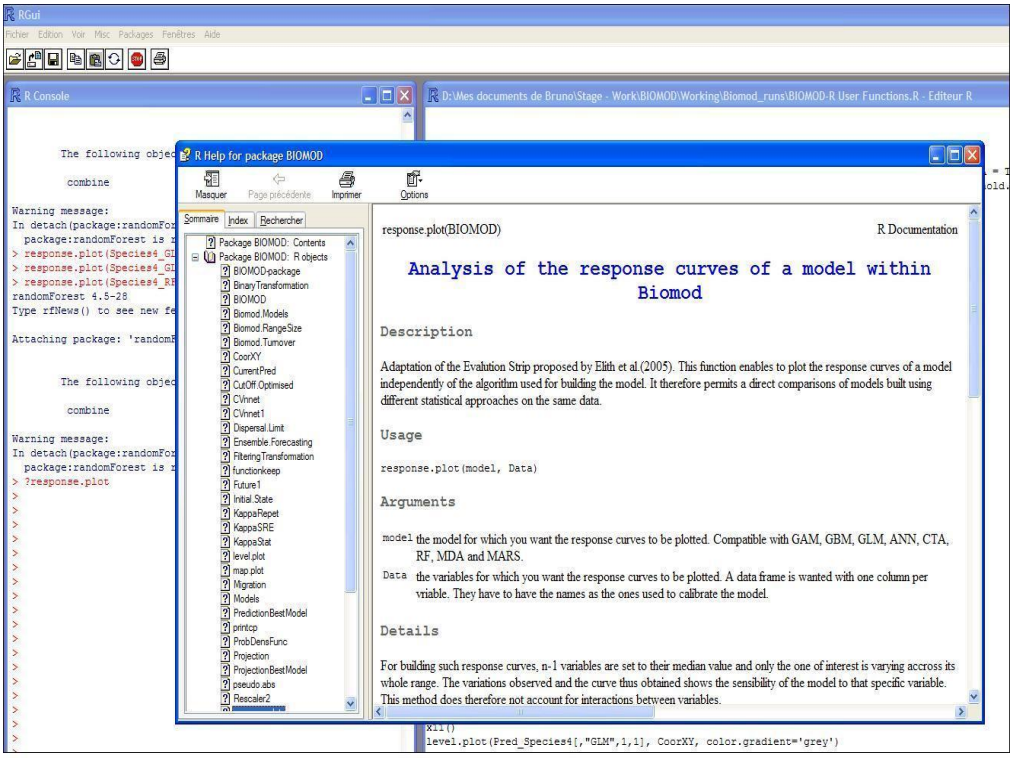
```
> library(BIOMOD)
```

```
raster version 1.0.0-3 (20-March-2010)
```

If you obtain the following error message : *Error in library(BIOMOD) : no package named 'BIOMOD' has been found*, or anything similar, then the package might not be located at the right place (see section above). You can also encounter errors if one or more dependency packages are missing from R's library. If everything seems right, then R knows the different BIOMOD functions.

You can type any function name with a question mark in front to access to the help files. You will find a general explanation of what the function does, an explanation for the use of each parameter to be set and some examples. As a general trend, the examples are more varied but also less detailed in the help files than in this present document.

```
> ?response.plot
```



You can also open the Biomod pdfs directly from R :

```
> Biomod.Manual()
```

0.1.1 Biomod Contents

BIOMOD is composed of a series of functions that enables to do our species modelling :

running BIOMOD
Initial.State
Models
Projection
Ensemble.Forecasting

further BIOMOD steps
CurrentPred
PredictionBestModel
ProjectionBestModel
Biomod.Turnover
Biomod.RangeSize
Migration

plotting functions
level.plot
multiple.plot
response.plot

ProbDensFunc calculates density probabilities
pseudo.abs generating pseudo-absences
BiomodManual opens the pdf manual and practicals from R

We will mainly focus here on the *Models* function as it contains all the options for calibrating and evaluating the models and look at how it can lead to significant variablity in prediction making. This function runs the models and evaluation technics presented in the Presentation Manual of BIOMOD (see *Biomod.Manual('Presentation')*).

0.1.2 R basics

Saving an R session

Here we present some basics to know about R and its functioning. Going through these simple examples will help you out in many cases that you will certainly come accross when using R. For those already aware of the usage of the software, you can jump to the next step.

You already know how to load a package in R using the *library()* function The *ls()* function enables you to see which objects are present in your R session. The *save.image()* function enables you to save them in a .RData object in your workspace.

```
> #we have nothing
> ls()
```

```
[1] "obj"
```

Let's create some objects. Remember that R is case sensitive, thus 'y' does not equal 'Y'.

```
> #create objects by assigning a value to it
> x = 123
> y <- "hello"
> x*2 -> Y
> #read them
> x
```

```
[1] 123
```

```
> y
```

```
[1] "hello"
```

```
> Y
```

```
[1] 246
```

```
> ls()
```

```
[1] "obj" "x"   "y"   "Y"
```

```
> #Objects can also be deleted with the rm() command
> #delete just the ones selected
> rm(x,Y)
> ls()
```

```
[1] "obj" "y"
```

```
> #delete all the objects
> rm(list=ls())
> ls()
```

```
character(0)
```

All the information stored in the memory of the R software can be saved as a work session (or workspace). When beginning a new work session within R, you can load any previously saved work session, which will load all the functions, objects, results obtained in a previous session thus enabling you to continue exactly from where you left it.

```
> obj <- 45^2
> save.image("Practical_1_start.RData")
> rm(list=ls())
> ls()
```

```
character(0)
```

```
> #load data in an other R session
> load("Practical_1_start.RData")
> ls()
```

```
[1] "obj"
```

```
> obj
```

```
[1] 2025
```

You can also choose to save just one object and save it on disk in binary code (only to be read in R) using the *save()* function.

GOOD HINT : When setting a new work session in a new directory, open R and save an image of your session before doing anything. It will create a file called *.RData* in this directory (containing nothing). The point of this is that double clicking on this file will open an R session directly correlated to your working directory.

```
> #in our case it will contain the object 'obj' we have in memory
> save.image()
```

Different type of objects

There can be a lot of different types of objects that you can use and create. In many cases the format of an object can be a limitation to using some functions and operations.

An example of the possibilities :

```
> #matrices
> mat <- matrix(1:24, nc=4, byrow=T)
> mat
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
[4,]   13   14   15   16
[5,]   17   18   19   20
[6,]   21   22   23   24
```

```
> #data.frames
> dataf <- data.frame(0, nc=5, nr=2)
> dataf
```

```
  X0 nc nr
1  0  5  2
```

```
> #conversion from one to the other
> W <- as.matrix(dataf)
> class(W)
```

```
[1] "matrix"
```

```
> W
```

```
     X0 nc nr
[1,]  0  5  2
```

```
> #arrays which are matrices of more than one dimension
> Arr <- array(c(mat, mat*2, mat*3), dim=c(6,4,3))
> dim(Arr)
```

```
[1] 6 4 3
```

```
> Arr
```

```
, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
[4,]   13   14   15   16
[5,]   17   18   19   20
[6,]   21   22   23   24

, , 2

     [,1] [,2] [,3] [,4]
[1,]    2    4    6    8
[2,]   10   12   14   16
[3,]   18   20   22   24
[4,]   26   28   30   32
[5,]   34   36   38   40
[6,]   42   44   46   48

, , 3

     [,1] [,2] [,3] [,4]
[1,]    3    6    9   12
[2,]   15   18   21   24
[3,]   27   30   33   36
[4,]   39   42   45   48
[5,]   51   54   57   60
[6,]   63   66   69   72
```

```
> #characters
> Goodmorning <- "hi"
> Goodmorning
```

```
[1] "hi"
```

```
> #lists (very usefull for storage)
> mylist <- list(mat, mat*2, 1:56, Goodmorning)
> mylist
```

```
[[1]]
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
[4,]   13   14   15   16
[5,]   17   18   19   20
[6,]   21   22   23   24

[[2]]
     [,1] [,2] [,3] [,4]
[1,]    2    4    6    8
[2,]   10   12   14   16
[3,]   18   20   22   24
[4,]   26   28   30   32
[5,]   34   36   38   40
[6,]   42   44   46   48

[[3]]
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22
[23] 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44
[45] 45 46 47 48 49 50 51 52 53 54 55 56

[[4]]
[1] "hi"


> #vectors
> bob <- c(132, 6:-2, "jack", 15^2)
> bob


 [1] "132"  "6"    "5"    "4"    "3"    "2"    "1"    "0"    "-1"
[10] "-2"   "jack" "225"


> #Note that because of "jack", all the values in "bob" are considered as characters (-> in between "")
> #To counter it :
> bob[1:4]


[1] "132" "6"   "5"   "4"


> as.numeric(bob[1:4])


[1] 132   6   5   4


> #functions
> MyFunc <- function(x){ x*2 }
> MyFunc


function(x){ x*2 }


> MyFunc(6)


[1] 12
```

Generally speaking, the square brackets enable to navigate in an object (but not always : double them for lists, use the @ sign for rasters). Reminder of the syntax : rows are specified before the comma and columns after. All dimensions need to be specified, even in blank

```
> mat[1:3,]
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
```

```
> mat[,4]
```

```
[1]  4  8 12 16 20 24
```

```
> mat[2,2]
```

```
[1] 6
```

```
> Arr[,4,1]
```

```
[1]  4  8 12 16 20 24
```

```
> Arr[,,2]
```

```
     [,1] [,2] [,3] [,4]
[1,]    2    4    6    8
[2,]   10   12   14   16
[3,]   18   20   22   24
[4,]   26   28   30   32
[5,]   34   36   38   40
[6,]   42   44   46   48
```

```
> mylist[[2]]
```

```
     [,1] [,2] [,3] [,4]
[1,]    2    4    6    8
[2,]   10   12   14   16
[3,]   18   20   22   24
[4,]   26   28   30   32
[5,]   34   36   38   40
[6,]   42   44   46   48
```

```
> bob[7:11]
```

```
[1] "1"    "0"    "-1"   "-2"   "jack"
```

0.1.3 Loading data

We need to import the species and the environmental data for our modelling. In our example the same file holds the two datasets.

```
> #Loading the example datasets
> #For practical reasons, species and environment datasets are stored together
> data(Sp.Env)

> #load the coordinates
> data(CoorXY)

> #Visualisation of our data (show first six rows)
> head(Sp.Env)

  Idw      X     Y   Var1 Var2  Var3  Var4  Var5  Var6  Var7 Sp281
1  73 -9.288 38.62 0.6683 4296 770.1 39.33 295.1 16.74 10.87     0
2  74 -9.292 39.52 0.7596 4174 928.1 57.32 348.7 16.41 10.51     0
3  75 -9.290 39.07 0.7424 4173 870.3 50.05 330.0 16.41 10.50     0
4  76 -8.715 37.72 0.5543 4264 620.0 24.99 239.1 16.66 10.93     0
5  77 -8.717 37.27 0.5489 4169 622.3 25.16 241.0 16.40 11.28     0
6  78 -8.148 37.72 0.5363 4206 591.8 25.74 222.9 16.49 10.13     0
  Sp290 Sp277 Sp164 Sp163 Sp177 Sp185 Sp191
1     1     0     0     1     0     0     1
2     1     0     0     1     0     0     1
3     0     0     0     1     0     0     1
4     0     0     0     0     0     0     0
5     0     0     0     0     0     0     0
6     0     0     0     0     0     0     0
```

- Idw: An Id to keep track of the row numbers
- X and Y: longitude and latitude of our sites (for plots, not needed for the modelling in itself)
- Var1 to Var7: Environmental variables (bioclimatic in that case)
- Sp281 to Sp191: Presence/absence of 8 species.

BIOMOD does not read the coordinates and does not recognise any geographical information when proceeding the modelling. The user should ensure that all datasets are kept in the same order, i.e. each species information (presence or absence) is correctly associated to the explanatory variables. Any mismatch will not be recognised by BIOMOD and the influence on the different outputs and results will be unnoticeable but real.

To load your own data from a text file, use the *read.table()* function:

```
> #Loading from a text file
> My.Data <- read.table("my_data.txt", h=T, sep="\t")
```

Type *?read.table* to get to the help file for more details and other possible extensions.

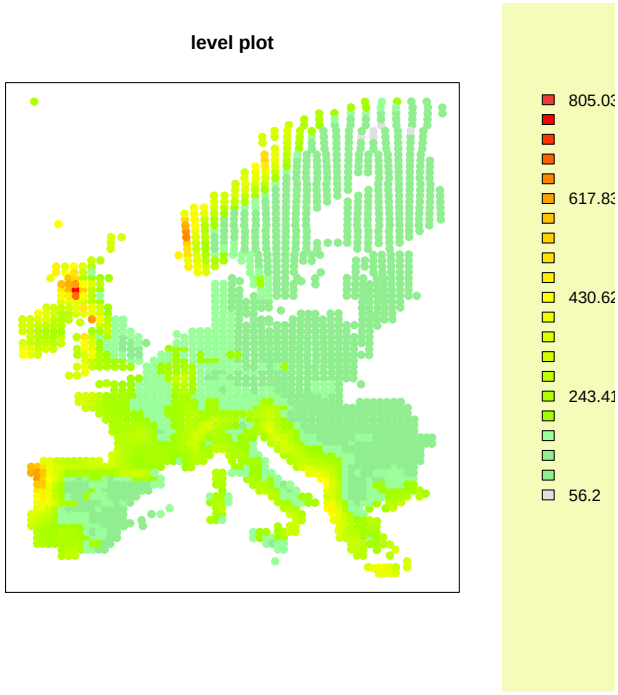
Here the separator was a tabulation but it could be a comma (then type sep=","). Your file might also be in a csv format; in that case you should use the *read.csv* function to correctly load your data.

NOTE: Missing values are not permitted in BIOMOD and will result in an error.

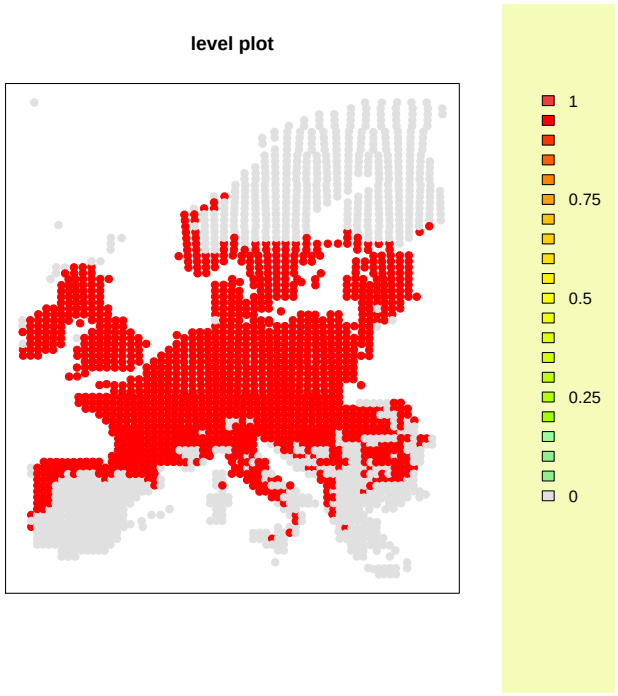
Ploting our data

The *level.plot* function requires two inputs : the vector of values that you want to plot and the coordinates of your data points. It works with any type of data.

```
> level.plot(Sp.Env[,8], CoorXY[,1:2])
```

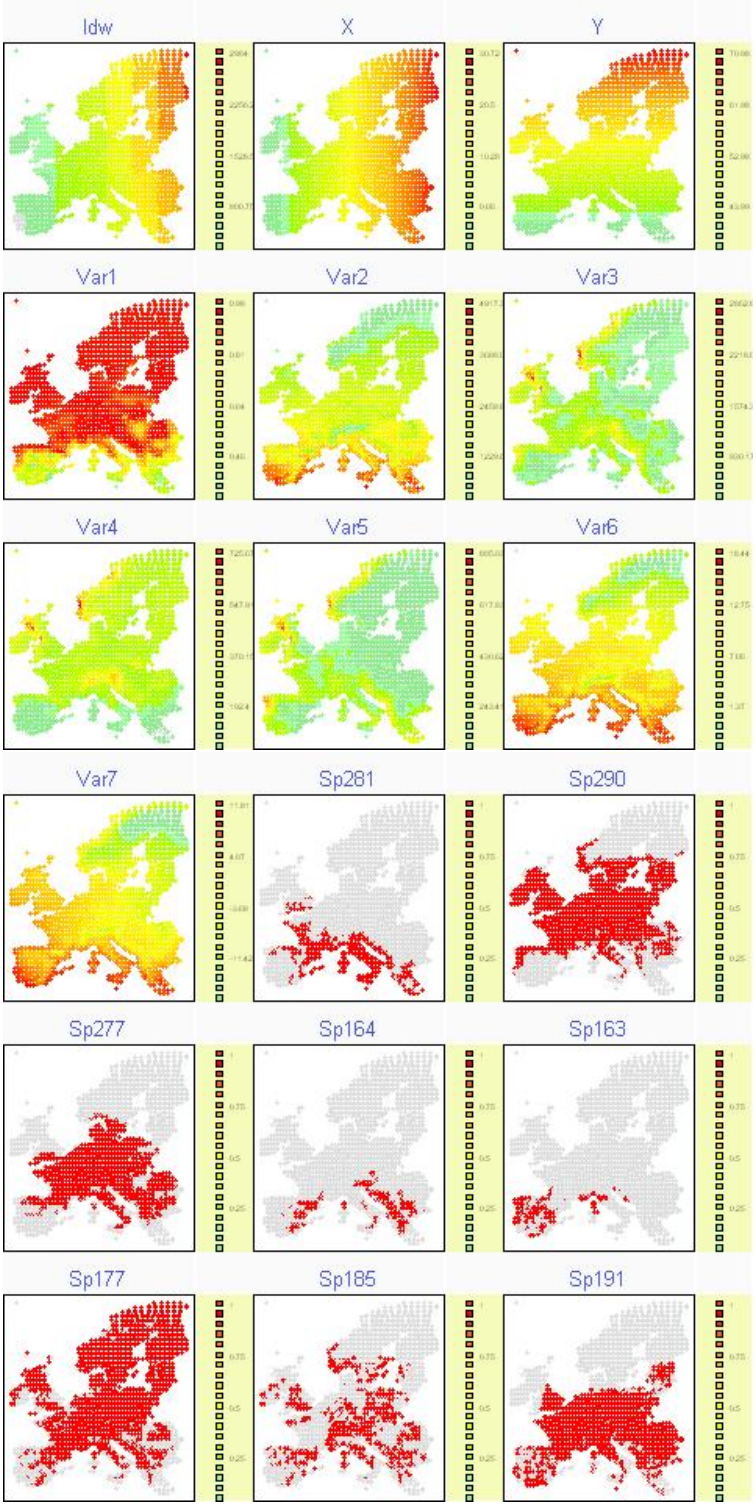



```
> level.plot(Sp.Env[,12], CoorXY[,1:2])
```



Let's take a general view of our data with the multiple.plot function :

```
> multiple.plot(Sp.Env, CoorXY[,1:2], cex=0.8)
```



You can modify the color gradient by setting the *color.gradient* argument to either *red* (the default), *blue* or *grey*.

0.2 Initialisation of Biomod

First, we need to set up the dataset in a correct format for BIOMOD by means of the *Initial.State* function. The syntax in the function is the following:

- Response: The response variables to model.
- Explanatory: The explanatory or independent variables.

Additional arguments (see the *Presentation* pdf for explanation) :
- IndependentResponse: Truly independent response variables.
- IndependentExplanatory: Truly independent explanatory variables.

These are used to evaluate the predictive accuracy of the models.

So our call looks like :

```
> Initial.State(Response = Sp.Env[,c(11,13)], Explanatory = Sp.Env[,4:10])
```

But we will inform anyway the 2 optional arguments with the same information. The point is to have an example of predictions on our full database as we are going to use pseudo-absences for the purpose of the example (hence BIOMOD will only produce predictions on partial data).

So instead we have :

```
> Initial.State(Response = Sp.Env[,c(11,13)], Explanatory = Sp.Env[,4:10],
 IndependentResponse = Sp.Env[,c(11,13)], IndependentExplanatory = Sp.Env[,4:10])
> ls()
```

```
 [1] "Arr"             "Biomod.material" "bob"
 [4] "CoorXY"          "DataBIOMOD"      "DataEvalBIOMOD"
 [7] "dataf"           "Goodmorning"     "mat"
[10] "MyFunc"          "mylist"          "obj"
[13] "Sp.Env"          "W"
```

It creates 'DataBIOMOD' our reference database, and DataEvalBIOMOD if you have given independent information. The latter will be used during the testing of the models. Make sure to always keep these datasets unchanged and never delete them.

```
> head(DataBIOMOD)
```

```
    Var1 Var2  Var3  Var4  Var5  Var6  Var7 Sp281 Sp277
1 0.6683 4296 770.1 39.33 295.1 16.74 10.87     0     0
2 0.7596 4174 928.1 57.32 348.7 16.41 10.51     0     0
3 0.7424 4173 870.3 50.05 330.0 16.41 10.50     0     0
4 0.5543 4264 620.0 24.99 239.1 16.66 10.93     0     0
5 0.5489 4169 622.3 25.16 241.0 16.40 11.28     0     0
6 0.5363 4206 591.8 25.74 222.9 16.49 10.13     0     0
```

DataBIOMOD contains the environmental variables in the first columns, followed by the species occurrences. DataEvalBIOMOD has the same structure but it contains the data for testing the

models.

An object called Biomod.material is also produced which contains information that has been extracted from the datasets like the number of variables, the number of species, etc.. Most of the functions will refer to this object to obtain some necessary values, so make sure to keep it unchanged.

```
> Biomod.material


$NbVar
[1] 7

$VarNames
[1] "Var1" "Var2" "Var3" "Var4" "Var5" "Var6" "Var7"

$NbSpecies
[1] 2

$species.names
[1] "Sp281" "Sp277"
```

0.3 Settings in Models()

The *Models()* function will run the different models available in BIOMOD and described in the *Presentation* manual. In this pdf, most of the available options in the function have already been explained to lighten this present document and concentrate on the practical issues of running the functions. Please don't hesitate to go back to it if you need further understanding of what the function does and for setting the options correctly. There are two main issues to consider :

- which models to select
- what calibration/evaluation procedure to choose

These two parameters should be thoroughly examined before throwing a good bunch of models and seeing what you get. It can bring sufficient randomness in the modelling to obtain one thing and another leading to possibly opposite conclusions, all from the same datasets in the first place. Also, in order to avoid either crashing your computer repeatedly either sitting there with R running for weeks, it is thus advised to carefully choose how many models can be reasonnably selected. This is more specifically an issue that can be dealt with choosing the correct settings of the calibration procedure.

Let's first have a look at the options to be set in the *Models()* function (arguments are presented with there default values):

Models(
Setting the models to TRUE or FALSE (to run them or not) and their associated options (please refer to the Presentation Manual)
GLM=FALSE, TypeGLM="simple", Test="AIC",
GBM=FALSE, No.trees= 5000,
GAM=FALSE, Spline=3,
CTA=FALSE, CV.tree=50,
ANN=FALSE, CV.ann=5,
SRE=FALSE, quant=0.025,
FDA=FALSE,
MARS=FALSE,
RF=FALSE,

The calibration procedure options
NbRunEval=1, DataSplit=100,
NbRepPA=0, strategy="sre", coor=NULL, distance=0, nb.absences=NULL,
Yweights=NULL,

The evaluation procedure options
VarImport=0,
Roc=FALSE, Optimized.Threshold.Roc=FALSE, Kappa=FALSE, TSS=FALSE,
KeepPredIndependent=FALSE)

)

Note that one of the calibration option is also determinant for the evaluation procedure (DataSplit option). Also, note that the various models' specific options will directly influence **the inner** calibration procedure of the models, whereas the calibration options below (NbRunEval, DataSplit) determine **the general trend** of the calibration which will be applied to all the models in the same way.

0.3.1 Algorithms options

Off course, one could advise that running all the available models sounds like the most promising approach to broaden the range of different modelling algorithm used and to assess for uncertainty in prediction making. Undoubtfull of the reliability of this idea, building models can be a real pain considering the data you are using in paralell with the computer power available. Not considering their robustness, some models are light to run and others can be pretty heavy. This can be even more true considering the settings you choose to go for with each algorithm (the number of trees for the GBM or number of cross validation of the ANN for example).
Consider also that as far as ensemble forecasts are concerned, having a mass of models isn't necessarily the best approach. It is believed that better forecasts can be obtained from models that are more reliable in the first place. This can be obtained by carefully considering the settings of each model. Please refer to the *Presentation manual*.

0.3.2 Calibration and evaluation procedure

Two aspects are then to be seriously considered with varying consequences on the forecoming modelling : **the calibration running time** and **the calibration quality** of the modelling.

0.3.3 Pseudo-absences

Pseudo-absences will be selected from the datasets given in input of the *Initial.State()* function. Please refer to the *Presentation manual* for further explanations.

- NbRepPA = 0: This will set the use of a pseudo-absences selection if higher than 0. Various repetitions of this procedure can be done, *multiplying the total number of runs* that are to be done for each model.

- strategy = 'random': the strategy to use for selecting pseudo absences. Can be either "circles", "squares", "per", "random" or "sre".

- coor = CoorXY: a two columned matrix giving the coordinates of your data points. It is needed for the "per", "circles" and "squares" strategies.

- distance = 3: a value giving the distance to use for the "per", "circles" and "squares" strategy of the pseudo absences selection. The value depends on the scale of your coordinates.

- nb.absences = 2000: the number of pseudo absences wanted to run the models with. They are randomly selected from the pool of pseudo absences available selected by the given strategy.

0.3.4 Weights

- Yweights: Weights that the user can set for the response variables (a matrix with N columns for the N species). This is similar to an index of detectability for each site, which allows users to give stronger weights to more reliable presences or absences. It can be scaled up and put as a weight in the modeling process. For more information, see how *weights* is working for each of the algorithm in R.

0.3.5 Evaluation parameters

- Roc, Kappa and TSS : the technics set to TRUE will be used for evaluating the performance of the models.
- Optimized.Threshold.Roc : enables to define a threshold using the Roc technic.
- VarImport : enables to define a threshold using the Roc technic.
- KeepPredIndependent : if TRUE, the evaluation technics will not be run on the remaining data that hasn't been used for calibration but on the independent data given in *Initial.State* (if any was given).

0.3.6 Running the models

We can now run the different models on our species. It takes only a few moments for each model to run. All the selected models (= TRUE) will run for each species. Here we will have 9(models selected)*4(3 repetitions + final model)*2(PA repetitions) which makes 72 models per species, it will thus take several minutes.

It might be appropriate to fraction the modelling effort on basic personal computers (i.e. laptops), especially if your data has tens of thousands of rows (requiring longer calculation time). One can run one species at a time with all the models being put to true.
Note that in contrast with earlier versions of BIOMOD, it is unwise to run one model at a time as the results are now stored per species. Making several runs with different models will bring unwelcome trouble for analysing the outputs.

Please, be also aware that the *NbRunEval* and *NbRepPA* arguments can considerably enlarge your calculation time by multiplying the number of runs to be made for each species. Do not enter excessively high values for these two arguments **unless** you have sufficient patience and/or reasonable calculation power.

```
> Models(GLM = T, TypeGLM = "poly", Test = "AIC", GBM = T, No.trees = 2000, GAM = T,
    Spline = 3, CTA = T, CV.tree = 50, ANN = T, CV.ann = 2, SRE = T, quant=0.025, FDA = T,
    MARS = T, RF = T, NbRunEval = 3, DataSplit = 80, Yweights=NULL, Roc = T, Optimized.Threshold.Roc = T,
    Kappa = T, TSS=T, KeepPredIndependent = T, VarImport=5, NbRepPA=2, strategy="circles",
    coor=CoorXY, distance=2, nb.absences=1000)


-----------------------------------
Modelling summary
-----------------------------------
Number of species modelled :                2
Sp281, Sp277

numerical variables :                        Var1, Var2, Var3, Var4, Var5, Var6, Var7


number of evaluation repetitions :         3
number of pseudo-absences runs :         2
models selected :                         ANN, CTA, GAM, GBM, GLM, MARS, FDA, RF, SRE
total number of model runs :                144
-----------------------------------


#####                         Sp281                           #####
#####                   pseudo-absence run 1                        #####
Model=Artificial Neural Network
         2 Fold Cross Validation + 3 Repetitions
Calibration and evaluation phase: Nb of cross-validations:  3
Evaluating Predictor Contributions in  ANN ...
Model=Classification tree
         50 Fold Cross-Validation
Evaluating Predictor Contributions in  CTA ...
Model=GAM spline
         3  Degrees of smoothing
Evaluating Predictor Contributions in  GAM ...
Model=Generalised Boosting Regression
         2000 maximum different trees and lambda Fold Cross-Validation
Evaluating Predictor Contributions in  GBM ...
Model=GLM polynomial + quadratic          Stepwise procedure using AIC criteria
Evaluating Predictor Contributions in  GLM ...
Model=Multiple Adaptive Regression Splines
```

```
Evaluating Predictor Contributions in  MARS ...
Model=Mixture Discriminant Analysis
Evaluating Predictor Contributions in  FDA ...
Model=Breiman and Cutler's random forests for classification and regression
Evaluating Predictor Contributions in  RF ...
Model=Surface Range Envelop
Evaluating Predictor Contributions in  SRE ...
#####                  pseudo-absence run 2                       #####
Model=Artificial Neural Network
        2 Fold Cross Validation + 3 Repetitions
Calibration and evaluation phase: Nb of cross-validations:  3
Evaluating Predictor Contributions in  ANN ...
Model=Classification tree
        50 Fold Cross-Validation
Evaluating Predictor Contributions in  CTA ...
Model=GAM spline
        3  Degrees of smoothing
Evaluating Predictor Contributions in  GAM ...
Model=Generalised Boosting Regression
        2000 maximum different trees and lambda Fold Cross-Validation
Evaluating Predictor Contributions in  GBM ...
Model=GLM polynomial + quadratic         Stepwise procedure using AIC criteria
Evaluating Predictor Contributions in  GLM ...
Model=Multiple Adaptive Regression Splines
Evaluating Predictor Contributions in  MARS ...
Model=Mixture Discriminant Analysis
Evaluating Predictor Contributions in  FDA ...
Model=Breiman and Cutler's random forests for classification and regression
Evaluating Predictor Contributions in  RF ...
Model=Surface Range Envelop
Evaluating Predictor Contributions in  SRE ...
#####                        Sp277                          #####
#####                  pseudo-absence run 1                       #####
Model=Artificial Neural Network
        2 Fold Cross Validation + 3 Repetitions
Calibration and evaluation phase: Nb of cross-validations:  3
Evaluating Predictor Contributions in  ANN ...
Model=Classification tree
        50 Fold Cross-Validation
Evaluating Predictor Contributions in  CTA ...
Model=GAM spline
        3  Degrees of smoothing
Evaluating Predictor Contributions in  GAM ...
Model=Generalised Boosting Regression
        2000 maximum different trees and lambda Fold Cross-Validation
Evaluating Predictor Contributions in  GBM ...
Model=GLM polynomial + quadratic         Stepwise procedure using AIC criteria
Evaluating Predictor Contributions in  GLM ...
Model=Multiple Adaptive Regression Splines
Evaluating Predictor Contributions in  MARS ...
Model=Mixture Discriminant Analysis
Evaluating Predictor Contributions in  FDA ...
Model=Breiman and Cutler's random forests for classification and regression
Evaluating Predictor Contributions in  RF ...
Model=Surface Range Envelop
Evaluating Predictor Contributions in  SRE ...


---------
 completed
```

load("RUN.RData")

For the purpose of the example (even though the data does not ask for it) we used 2 pseudo-absences (PA) runs. Note that there has only been one PA run for Sp277 because too little absences were available compared to the ones wanted. The nb.absences argument was set to 1000, but:

```
> #the number of data selected by the pseudo-absences procedure
> length(Biomod.PA.data$Sp277)
```

```
[1] 1791
```

```
> #the number of presences for Sp277
> sum(Sp.Env[,"Sp277"])
```

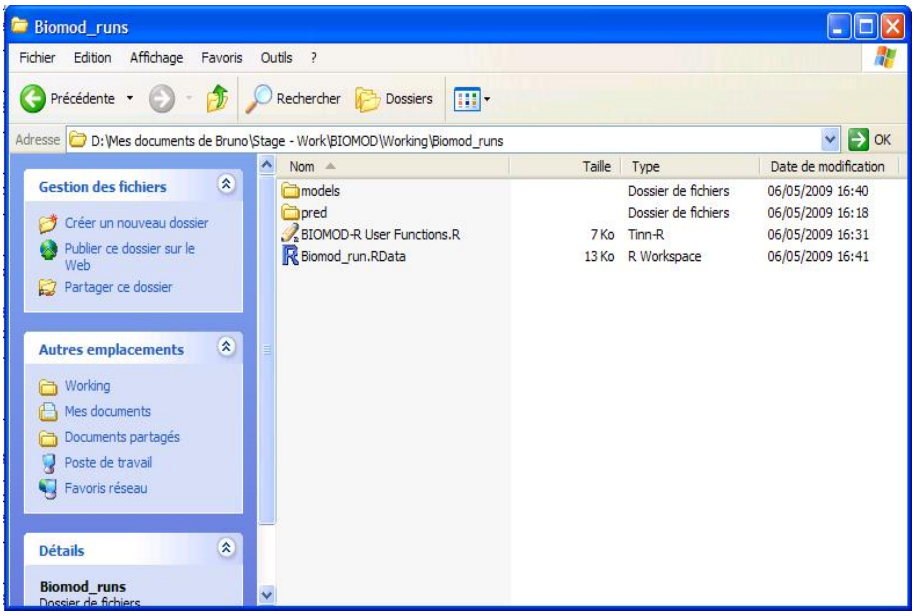
```
[1] 1080
```

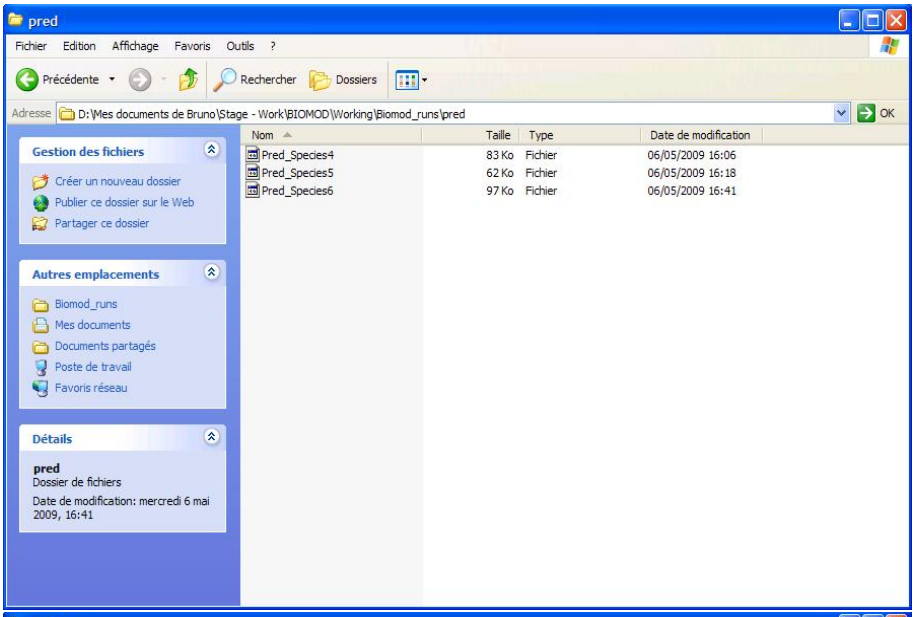
```
> #Hence, the number of absences available for calibration
> length(Biomod.PA.data$Sp277) - sum(Sp.Env[,"Sp277"])
```

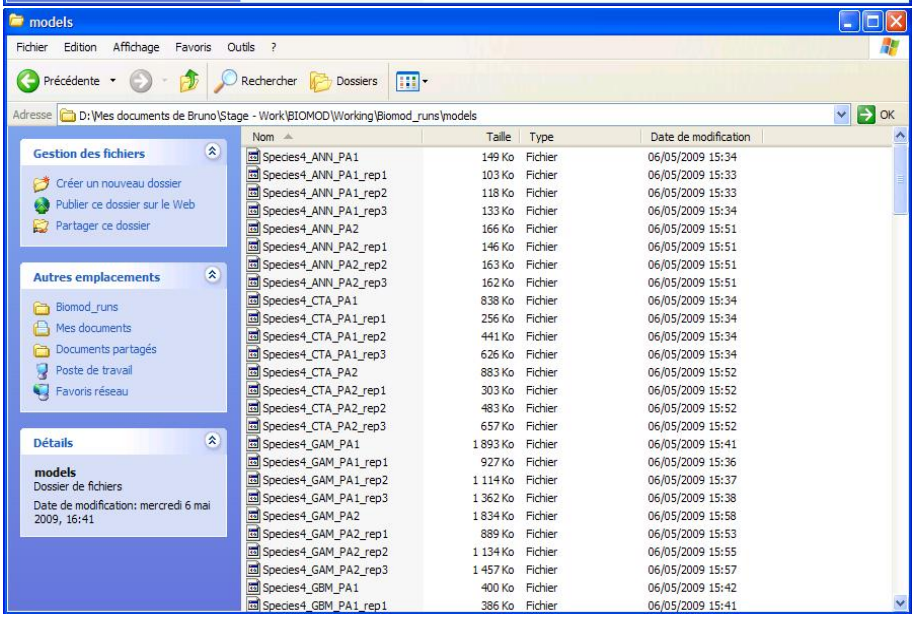
```
[1] 711
```

Too little absences are available. In this case, a single pseudo-absences run is made using all the absences available.

A series of objects have been produced in the workspace and also on the hardrive of your computer. Your working folder should now look like this.




pred
Fichier Edition Affichage Favoris Outils ?
Précédente Rechercher Dossiers
Adresse D:\Mes documents de Bruno\Stage - Work\BIOMOD\Working\Biomod_runs\pred
OK
Gestion des fichiers
Créer un nouveau dossier
Publier ce dossier sur le Web
Partager ce dossier
Autres emplacements
Biomod_runs
Mes documents
Documents partagés
Poste de travail
Favoris réseau
Détails
pred
Dossier de fichiers
Date de modification: mercredi 6 mai 2009, 16:41
Nom Taille Type Date de modification
Pred_Species4 83 Ko Fichier 06/05/2009 16:06
Pred_Species5 62 Ko Fichier 06/05/2009 16:18
Pred_Species6 97 Ko Fichier 06/05/2009 16:41


models
Fichier Edition Affichage Favoris Outils ?
Précédente Rechercher Dossiers
Adresse D:\Mes documents de Bruno\Stage - Work\BIOMOD\Working\Biomod_runs\models
OK
Gestion des fichiers
Créer un nouveau dossier
Publier ce dossier sur le Web
Partager ce dossier
Autres emplacements
Biomod_runs
Mes documents
Documents partagés
Poste de travail
Favoris réseau
Détails
models
Dossier de fichiers
Date de modification: mercredi 6 mai 2009, 16:41
Nom Taille Type Date de modification
Species4_ANN_PA1 149 Ko Fichier 06/05/2009 15:34
Species4_ANN_PA1_rep1 103 Ko Fichier 06/05/2009 15:33
Species4_ANN_PA1_rep2 118 Ko Fichier 06/05/2009 15:33
Species4_ANN_PA1_rep3 133 Ko Fichier 06/05/2009 15:34
Species4_ANN_PA2 166 Ko Fichier 06/05/2009 15:51
Species4_ANN_PA2_rep1 146 Ko Fichier 06/05/2009 15:51
Species4_ANN_PA2_rep2 163 Ko Fichier 06/05/2009 15:51
Species4_ANN_PA2_rep3 162 Ko Fichier 06/05/2009 15:51
Species4_CTA_PA1 838 Ko Fichier 06/05/2009 15:34
Species4_CTA_PA1_rep1 256 Ko Fichier 06/05/2009 15:34
Species4_CTA_PA1_rep2 441 Ko Fichier 06/05/2009 15:34
Species4_CTA_PA1_rep3 626 Ko Fichier 06/05/2009 15:34
Species4_CTA_PA2 883 Ko Fichier 06/05/2009 15:52
Species4_CTA_PA2_rep1 303 Ko Fichier 06/05/2009 15:52
Species4_CTA_PA2_rep2 483 Ko Fichier 06/05/2009 15:52
Species4_CTA_PA2_rep3 657 Ko Fichier 06/05/2009 15:52
Species4_GAM_PA1 1 893 Ko Fichier 06/05/2009 15:41
Species4_GAM_PA1_rep1 927 Ko Fichier 06/05/2009 15:36
Species4_GAM_PA1_rep2 1 114 Ko Fichier 06/05/2009 15:37
Species4_GAM_PA1_rep3 1 362 Ko Fichier 06/05/2009 15:38
Species4_GAM_PA2 1 834 Ko Fichier 06/05/2009 15:58
Species4_GAM_PA2_rep1 889 Ko Fichier 06/05/2009 15:53
Species4_GAM_PA2_rep2 1 134 Ko Fichier 06/05/2009 15:55
Species4_GAM_PA2_rep3 1 457 Ko Fichier 06/05/2009 15:57
Species4_GBM_PA1 400 Ko Fichier 06/05/2009 15:42
Species4_GBM_PA1_rep1 386 Ko Fichier 06/05/2009 15:41

0.4 Analysing the outputs

There are now various objects stored in the workspace. First, we can have a look at what is present in our R session and check what has been produced by the *Models()* function.

```
> ls()
```

```
 [1] "Arr"                       "Biomod.material"
 [3] "Biomod.PA.data"            "Biomod.PA.sample"
 [5] "bob"                       "CoorXY"
 [7] "DataBIOMOD"                "DataEvalBIOMOD"
 [9] "dataf"                     "Evaluation.results.Kappa"
[11] "Evaluation.results.Roc"    "Evaluation.results.TSS"
[13] "GBM.list"                  "GBM.perf"
[15] "Goodmorning"               "mat"
[17] "MyFunc"                    "mylist"
[19] "obj"                       "Sp.Env"
[21] "VarImportance"             "W"
```

Some of them result from the run while others were already there. Let's clear the objects we created earlier and that are not related with our modelling campaign.

```
> rm(Arr, bob, dataf, mat, mylist, obj, rand, store, W)
> ls()
```

```
 [1] "Biomod.material"           "Biomod.PA.data"
 [3] "Biomod.PA.sample"          "CoorXY"
 [5] "DataBIOMOD"                "DataEvalBIOMOD"
 [7] "Evaluation.results.Kappa" "Evaluation.results.Roc"
 [9] "Evaluation.results.TSS"    "GBM.list"
[11] "GBM.perf"                  "Goodmorning"
[13] "MyFunc"                    "Sp.Env"
[15] "VarImportance"
```

So, we have the outputs generated by *Initial.State* and the original datasets : - Sp.Env
- CoorXY
- DataBIOMOD
- Biomod.material

We also have the objects produced by the *Models()* function in the workspace (additional objects are stored on the hard disk). These are : - Evaluation.results.Roc
- Evaluation.results.Kappa
- Evaluation.results.TSS
- VarImportance.

And we get the following if NbRepPA is higher than 0 : - Biomod.PA.data
- Biomod.PA.sample
- SpNoName.circles.2 (or something close)

0.4.1 Objects in the workspace

Evaluation of the predictive performance

There are three available techniques for making an assessment of a model's performance. If ROC, Kappa and/or TSS is selected, the correspondant technique will be run on the cross-validation step

models (if any cross- validation are wanted) and on the final model calibrated on 100% of the data. Performance measures are stored indidividually for each species and model, and for each run.

A summary table of the type "Evaluation.results.method" are produced by the *Models* function containing the predictive performance of each model which is convenient for making comparisons across methods and taxa. The same structure is kept for Roc, Kappa and TSS methods.

```
> #Here we only display the info for the first species modelled
> Evaluation.results.Kappa[1:8]


$Sp281_PA1
     Cross.validation indepdt.data total.score Cutoff Sensitivity
ANN             0.908          0.6      0.8985  398.3       97.70
CTA             0.852        0.627      0.9360  209.6       99.74
GAM             0.849        0.666      0.8796  679.3       91.84
GBM             0.873        0.576      0.8908  503.5       95.92
GLM             0.854        0.554      0.7766  748.5       84.18
MARS            0.906        0.692      0.9289  459.5       94.64
FDA             0.891        0.627      0.9158  599.6       92.60
RF              0.927        0.765      1.0000  340.0      100.00
SRE             0.632        0.394      0.6560   10.0       83.42
     Specificity
ANN         95.0
CTA         96.4
GAM         96.4
GBM         95.3
GLM         93.6
MARS        98.1
FDA         98.2
RF         100.0
SRE         86.0

$Sp281_PA1_rep1
     Cross.validation indepdt.data total.score Cutoff Sensitivity
ANN             0.909         none      0.9127  556.0       93.11
CTA             0.886         none      0.9170  246.7       98.21
GAM             0.885         none      0.8519  579.4       94.13
GBM             0.871         none      0.8846  510.6       94.13
GLM             0.872         none      0.8982  699.3       91.84
MARS            0.893         none      0.9174  449.6       92.35
FDA             0.900         none      0.9149  323.7       94.13
RF              0.921         none      0.9841  370.0       99.23
SRE             0.638         none      0.6527   10.0       83.16
     Specificity
ANN         97.8
CTA         95.9
GAM         93.7
GBM         95.7
GLM         97.5
MARS        98.4
FDA         97.5
RF          99.4
SRE         85.9

$Sp281_PA1_rep2
     Cross.validation indepdt.data total.score Cutoff Sensitivity
ANN             0.901         none      0.9462  570.7       94.64
CTA             0.892         none      0.9248  250.0       96.94
GAM             0.896         none      0.8526  728.5       87.50
GBM             0.911         none      0.8845  496.2       96.43
GLM             0.888         none      0.8349  839.2       81.63
MARS            0.926         none      0.9231  459.5       93.37
FDA             0.955         none      0.9269  334.5       94.13
RF              0.973         none      0.9929  360.0       99.23
SRE             0.674         none      0.6210   10.0       83.42
     Specificity
ANN         99.1
```

```
CTA         96.9
GAM         96.7
GBM         94.7
GLM         98.3
MARS        98.3
FDA         98.2
RF          99.9
SRE         83.5

$Sp281_PA1_rep3
     Cross.validation indepdt.data total.score Cutoff Sensitivity
ANN             0.913         none      0.9387  163.8       97.70
CTA             0.777         none      0.8667  720.8       96.43
GAM             0.767         none      0.8489  578.8       94.39
GBM             0.836         none      0.8733  518.3       92.35
GLM             0.803         none      0.8809  579.4       95.92
MARS            0.898         none      0.9185  659.3       91.07
FDA             0.818         none      0.8805  154.7       95.41
RF              0.888         none      0.9771  330.0       99.49
SRE             0.585         none      0.6148   10.0       84.95
     Specificity
ANN         97.4
CTA         93.6
GAM         93.4
GBM         95.8
GLM         94.7
MARS        99.0
FDA         94.9
RF          98.9
SRE         82.2

$Sp281_PA2
     Cross.validation indepdt.data total.score Cutoff Sensitivity
ANN             0.907        0.595      0.9037  365.6       98.21
CTA             0.888        0.618      0.9343  190.0       99.49
GAM             0.893        0.683      0.8795  589.4       96.17
GBM             0.912        0.561      0.8939  507.2       95.66
GLM             0.888        0.651      0.8468  829.2       83.16
MARS            0.937        0.689      0.9276  349.7       95.66
FDA             0.922        0.643      0.9270  390.2       94.39
RF              0.932         0.76      1.0000  320.0      100.00
SRE             0.648        0.394      0.6661   10.0       83.42
     Specificity
ANN         95.1
CTA         96.4
GAM         94.5
GBM         95.6
GLM         98.3
MARS        97.6
FDA         98.1
RF         100.0
SRE         86.7

$Sp281_PA2_rep1
     Cross.validation indepdt.data total.score Cutoff Sensitivity
ANN             0.892         none      0.9195  550.0       93.11
CTA             0.895         none      0.9197  720.0       96.94
GAM             0.900         none      0.8823  739.3       88.52
GBM             0.965         none      0.9003  501.8       97.96
GLM             0.853         none      0.8554  799.2       84.95
MARS            0.945         none      0.9284  519.5       93.62
FDA             0.900         none      0.9015  246.7       91.58
RF              0.956         none      0.9911  410.0       99.49
SRE             0.679         none      0.6043   10.0       85.71
     Specificity
ANN         98.2
CTA         96.6
GAM         98.0
GBM         95.0
GLM         98.0
MARS        98.5
FDA         97.8
```

```
RF          99.7
SRE         81.0

$Sp281_PA2_rep2
     Cross.validation indepdt.data total.score Cutoff Sensitivity
ANN             0.884         none      0.9270  430.8       94.39
CTA             0.872         none      0.9046  719.8       93.88
GAM             0.833         none      0.8740  749.2       88.78
GBM             0.875         none      0.8971  502.7       95.41
GLM             0.896         none      0.9175  639.4       95.92
MARS            0.912         none      0.9381  399.6       96.17
FDA             0.929         none      0.9209  180.2       95.92
RF              0.896         none      0.9788  350.0       98.98
SRE             0.624         none      0.6603   10.0       83.42
     Specificity
ANN         98.1
CTA         97.0
GAM         97.4
GBM         95.9
GLM         96.9
MARS        98.0
FDA         97.1
RF          99.2
SRE         86.3

$Sp281_PA2_rep3
     Cross.validation indepdt.data total.score Cutoff Sensitivity
ANN             0.946         none      0.9421  261.4       97.70
CTA             0.896         none      0.9007  230.0       98.72
GAM             0.946         none      0.8869  688.6       92.60
GBM             0.896         none      0.8886  510.0       95.15
GLM             0.917         none      0.8493  849.1       82.14
MARS            0.955         none      0.9303  559.4       93.88
FDA             0.938         none      0.9274  283.6       95.15
RF              0.946         none      0.9893  620.0       98.98
SRE             0.640         none      0.6145   10.0       84.18
     Specificity
ANN         97.6
CTA         94.7
GAM         96.5
GBM         95.5
GLM         98.9
MARS        98.5
FDA         97.8
RF          99.8
SRE         82.6
```

Taking the example of the first PA run : there are 4 different matrices, one for each run (3 repetitions with a 80-20% partitioning and the final 100% model). For the first repetition, (Sp277_PA1_rep1), the first column is the score on the remaining 20% of the data after calibration of the model. The last four columns are determined with the 80% used for calibration and the 20% leftovers combined. Like for the probabilities, the thresholds (column 4) are scalled from 0 to 1000. Thiese threshold values will be used later to transform probabilities into presence-absence (binary format) or filtered values. The sensitivity and specifity associated with that threshold are given in the last two columns.

For the final model (Sp277_PA1), the first column is the average of the cross- validation of all the repetitions. The second one is the score when the model is evaluated on independent data if any is available, and the four following colums are results obtained from the final model itself.

You can explore and see that the PA2 runs for Sp277 are empty matrices. That's because there has only been 1 PA run for that species.

Evaluation of the importance of each variable

.

With a permutation procedure, BIOMOD proposes another way to examine the importance of the variables in the models. We extract a measure of relative importance of each variable that is independent of the model (Please look up the *Presentation Manual* for further explanations).

Running the *Models* function will produce an object called "VarImportance" (only if VarImp was put higher than 0 in the function call). The results are stored individually per species and per model. Let's look at the results we have :

```
> VarImportance
```

```
$Sp281
      Var1  Var2  Var3  Var4  Var5  Var6  Var7
ANN  0.003 0.980 0.719 0.653 0.390 0.392 0.760
CTA  0.245 0.126 0.170 0.057 0.163 0.020 0.663
GAM  0.403 1.161 0.631 0.147 0.204 0.413 1.232
GBM  0.174 0.011 0.045 0.029 0.013 0.007 0.720
GLM  0.437 0.068 0.721 0.213 0.184 0.000 0.216
MARS 0.581 0.163 0.059 0.151 0.105 0.000 0.640
FDA  0.395    NA 0.454 0.168 0.109    NA 1.196
RF   0.137 0.055 0.088 0.069 0.039 0.039 0.418
SRE  0.075 0.038 0.002 0.021 0.072 0.015 0.097

$Sp277
      Var1  Var2  Var3  Var4  Var5  Var6  Var7
ANN  0.000 0.690 0.882 0.802 0.703 0.251 0.760
CTA  0.107 0.302 0.006 0.092 0.079 0.365 0.312
GAM  0.056 0.722 0.591 0.191 0.497 0.143 0.302
GBM  0.023 0.352 0.000 0.157 0.013 0.208 0.064
GLM  0.047 0.466 0.312 0.203 0.253 0.640 0.598
MARS 0.000 0.193 0.507 0.382 0.367 0.453 0.312
FDA  0.094 1.082 0.150 0.173 0.184 0.400 0.810
RF   0.043 0.273 0.011 0.045 0.024 0.208 0.127
SRE  0.019 0.008 0.020 0.018 0.037 0.001 0.091
```

Values should be considered independently for each model. For instance, the SRE shows a generally low value for all the variable when the is generally high. The goal is nevertheless to identify which variable is of the most importance. A good example with the GLM for Sp281, only 2 variables seem to have a significance in the predictions.

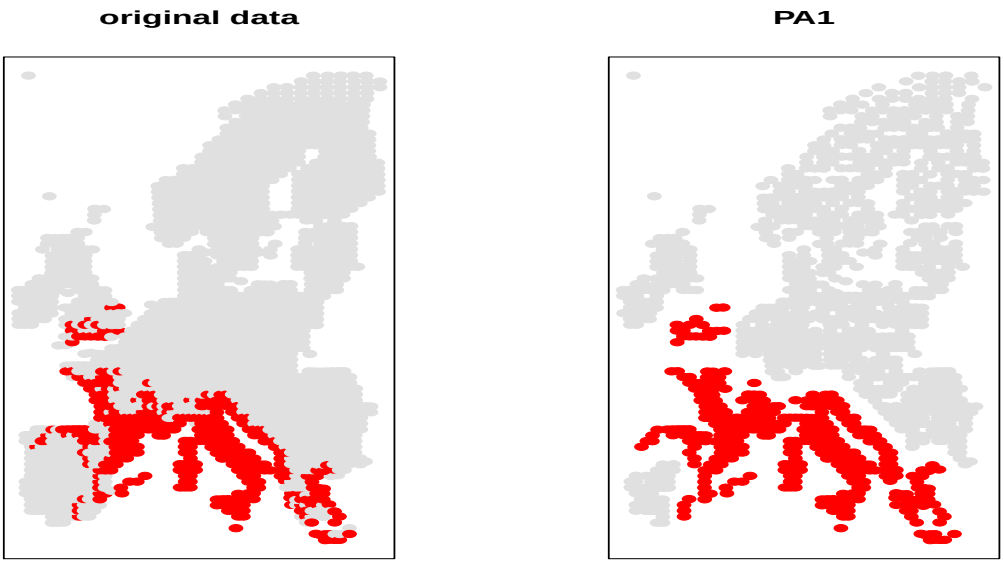
Note also that this technic only accounts for the direct effects of the variables and doesn't enable to identify combined effect of variables or anything as such. It should mainly be considered as an informational tool, not an absolute reliable measure of the variables' contributions to the models.

PA data generated

Biomod.PA.data contains the amount of data available after the inner run of the pseudo-absence function. Biomod.PA.sample contains the rows to take from DataBIOMOD to get the data that has been used for the calibration of each species for each PA run. The last object is a result of the pesudo-absence function inner run and is of no importance here (but see the "Pseudo-absences" section of the Presentation manual for explanations).

For example, let's see what data has been used for the calibration of the run PA1 :

```
> our.lines <- Biomod.PA.sample$Sp281$PA1
> par(mfrow=c(1,2))
> level.plot(DataBIOMOD[, "Sp281"], CoorXY, title='original data', show.scale=F)
> level.plot(DataBIOMOD[our.lines, "Sp281"], CoorXY[our.lines,], title='PA1', show.scale=F)
```



0.4.2 Objects stored on the hard drive : The Models

Each algorithm (excepted SRE) generates an object storing the different parameterisation, the importance of each variable for the model and other statistics. This output is essential as it allows generating predictions.
These objects, the models themselves, are now stored out of the R workspace directly on the computers' hard disk. They are named after the algorithm used and the species' names, i.e. Sp164_FDA for example. There is also extensions of the names concerning the repetitions and the pseudo-absences runs, so that one of our models will be Sp164_FDA_PA1_rep2.

Back loading the models and having them directly usable is very straightforward : simply use the *load()* function to have the model restored in the R workspace, with the same name plus the directory root. This is also the case with the other outputs stored outside of R (predictions and projections). The syntax is not always handy but easy to pick up :

```
> #Example of the GLM
> load("models/Sp277_GLM_PA1")
> ls()


 [1] "Biomod.material"           "Biomod.PA.data"
 [3] "Biomod.PA.sample"          "CoorXY"
 [5] "DataBIOMOD"                "DataEvalBIOMOD"
 [7] "Evaluation.results.Kappa" "Evaluation.results.Roc"
 [9] "Evaluation.results.TSS"    "GBM.list"
[11] "GBM.perf"                  "Goodmorning"
[13] "MyFunc"                    "our.lines"
[15] "Sp.Env"                    "Sp277_GLM_PA1"
[17] "VarImportance"


> Sp277_GLM_PA1


Call:  glm(formula = Sp277 ~ poly(Var2, 3) + poly(Var7, 3) + poly(Var5,      3) + poly(Var1, 3) + poly(Var3, 3)

Coefficients:
   (Intercept)  poly(Var2, 3)1  poly(Var2, 3)2  poly(Var2, 3)3
         0.344        -598.966        -693.252         -39.381
poly(Var7, 3)1  poly(Var7, 3)2  poly(Var7, 3)3  poly(Var5, 3)1
      -135.059        -700.546          83.745        -132.451
poly(Var5, 3)2  poly(Var5, 3)3  poly(Var1, 3)1  poly(Var1, 3)2
        67.663         204.751         -14.084         -36.087
poly(Var1, 3)3  poly(Var3, 3)1  poly(Var3, 3)2  poly(Var3, 3)3
       -81.726         187.879        -405.403        -243.330
poly(Var4, 3)1  poly(Var4, 3)2  poly(Var4, 3)3  poly(Var6, 3)1
       112.524          48.763         282.348        1018.986
poly(Var6, 3)2  poly(Var6, 3)3
      1149.032         244.607

Degrees of Freedom: 1790 Total (i.e. Null);  1769 Residual
Null Deviance:            2990
Residual Deviance: 306         AIC: 396


> summary(Sp277_GLM_PA1)


Call:
glm(formula = Sp277 ~ poly(Var2, 3) + poly(Var7, 3) + poly(Var5,
    3) + poly(Var1, 3) + poly(Var3, 3) + poly(Var4, 3) + poly(Var6,
    3), family = binomial, data = DataBIOMOD[calib.lines, ],
    weights = Yweights[calib.lines, i])
```

```
Deviance Residuals:
      Min         1Q     Median         3Q        Max
-2.91e+00  -2.67e-04   8.32e-05   1.87e-02   2.58e+00

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)      0.344      1.052    0.33   0.7437
poly(Var2, 3)1 -598.966    340.449   -1.76   0.0785 .
poly(Var2, 3)2 -693.252    125.512   -5.52  3.3e-08 ***
poly(Var2, 3)3  -39.381     20.200   -1.95   0.0512 .
poly(Var7, 3)1 -135.059    136.929   -0.99   0.3240
poly(Var7, 3)2 -700.546     84.270   -8.31  < 2e-16 ***
poly(Var7, 3)3   83.745     72.402    1.16   0.2474
poly(Var5, 3)1 -132.451     69.866   -1.90   0.0580 .
poly(Var5, 3)2   67.663     45.108    1.50   0.1336
poly(Var5, 3)3  204.751     36.798    5.56  2.6e-08 ***
poly(Var1, 3)1  -14.084     31.784   -0.44   0.6577
poly(Var1, 3)2  -36.087     13.295   -2.71   0.0066 **
poly(Var1, 3)3  -81.726     13.569   -6.02  1.7e-09 ***
poly(Var3, 3)1  187.879    104.949    1.79   0.0734 .
poly(Var3, 3)2 -405.403     90.675   -4.47  7.8e-06 ***
poly(Var3, 3)3 -243.330     48.658   -5.00  5.7e-07 ***
poly(Var4, 3)1  112.524     59.176    1.90   0.0572 .
poly(Var4, 3)2   48.763     43.591    1.12   0.2633
poly(Var4, 3)3  282.348     47.353    5.96  2.5e-09 ***
poly(Var6, 3)1 1018.986    422.797    2.41   0.0159 *
poly(Var6, 3)2 1149.032    209.588    5.48  4.2e-08 ***
poly(Var6, 3)3  244.607     54.432    4.49  7.0e-06 ***
---
Signif. codes:  0 Ś***Š 0.001 Ś**Š 0.01 Ś*Š 0.05 Ś.Š 0.1 Ś Š 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2994.40  on 1790  degrees of freedom
Residual deviance:  306.11  on 1769  degrees of freedom
AIC: 395.7

Number of Fisher Scoring iterations: 10
```

A series of commands enables you to navigate in the object and to extract usefull information from it. Here are a few example that can be used for all algorithms.

```
> #simply type its name
> Sp277_GLM_PA1


Call:  glm(formula = Sp277 ~ poly(Var2, 3) + poly(Var7, 3) + poly(Var5,      3) + poly(Var1, 3) + poly(Var3, 3)

Coefficients:
   (Intercept)  poly(Var2, 3)1  poly(Var2, 3)2  poly(Var2, 3)3
         0.344        -598.966        -693.252         -39.381
poly(Var7, 3)1  poly(Var7, 3)2  poly(Var7, 3)3  poly(Var5, 3)1
      -135.059        -700.546          83.745        -132.451
poly(Var5, 3)2  poly(Var5, 3)3  poly(Var1, 3)1  poly(Var1, 3)2
        67.663         204.751         -14.084         -36.087
poly(Var1, 3)3  poly(Var3, 3)1  poly(Var3, 3)2  poly(Var3, 3)3
       -81.726         187.879        -405.403        -243.330
poly(Var4, 3)1  poly(Var4, 3)2  poly(Var4, 3)3  poly(Var6, 3)1
       112.524          48.763         282.348        1018.986
poly(Var6, 3)2  poly(Var6, 3)3
      1149.032         244.607

Degrees of Freedom: 1790 Total (i.e. Null);  1769 Residual
Null Deviance:          2990
Residual Deviance: 306          AIC: 396


> names(Sp277_GLM_PA1)
```

```
 [1] "coefficients"      "residuals"         "fitted.values"
 [4] "effects"           "R"                 "rank"
 [7] "qr"                "family"            "linear.predictors"
[10] "deviance"          "aic"               "null.deviance"
[13] "iter"              "weights"           "prior.weights"
[16] "df.residual"       "df.null"           "y"
[19] "converged"         "boundary"          "model"
[22] "call"              "formula"           "terms"
[25] "data"              "offset"            "control"
[28] "method"            "contrasts"         "xlevels"
[31] "anova"
```

```
> str(Sp277_GLM_PA1)
```

```
List of 31
 $ coefficients     : Named num [1:22] 0.344 -598.966 -693.252 -39.381 -135.059 ...
  ..- attr(*, "names")= chr [1:22] "(Intercept)" "poly(Var2, 3)1" "poly(Var2, 3)2" "poly(Var2, 3)3" ...
 $ residuals        : Named num [1:1791] -1 -1 -1 -1 -1 ...
  ..- attr(*, "names")= chr [1:1791] "1" "2" "3" "4" ...
 $ fitted.values    : Named num [1:1791] 7.90e-12 1.31e-07 8.52e-09 2.22e-16 2.22e-16 ...
  ..- attr(*, "names")= chr [1:1791] "1" "2" "3" "4" ...
 $ effects          : Named num [1:1791] 1.941 -1.077 2.606 0.986 0.576 ...
  ..- attr(*, "names")= chr [1:1791] "(Intercept)" "poly(Var2, 3)1" "poly(Var2, 3)2" "poly(Var2, 3)3" ...
 $ R                : num [1:22, 1:22] -6.76 0 0 0 0 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:22] "(Intercept)" "poly(Var2, 3)1" "poly(Var2, 3)2" "poly(Var2, 3)3" ...
  .. ..$ : chr [1:22] "(Intercept)" "poly(Var2, 3)1" "poly(Var2, 3)2" "poly(Var2, 3)3" ...
 $ rank             : int 22
 $ qr               :List of 5
  ..$ qr   : num [1:1791, 1:22] -6.76 6.61e-05 1.68e-05 2.72e-09 2.72e-09 ...
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : chr [1:1791] "1" "2" "3" "4" ...
  .. .. ..$ : chr [1:22] "(Intercept)" "poly(Var2, 3)1" "poly(Var2, 3)2" "poly(Var2, 3)3" ...
  ..$ rank : int 22
  ..$ qraux: num [1:22] 1 1 1 1 1 ...
  ..$ pivot: int [1:22] 1 2 3 4 5 6 7 8 9 10 ...
  ..$ tol  : num 1e-11
  ..- attr(*, "class")= chr "qr"
 $ family           :List of 12
  ..$ family    : chr "binomial"
  ..$ link      : chr "logit"
  ..$ linkfun   :function (mu)
  ..$ linkinv   :function (eta)
  ..$ variance  :function (mu)
  ..$ dev.resids:function (y, mu, wt)
  ..$ aic       :function (y, n, mu, wt, dev)
  ..$ mu.eta    :function (eta)
  ..$ initialize:  expression({     if (NCOL(y) == 1) {         if (is.factor(y))              y <- y != levels(
  ..$ validmu   :function (mu)
  ..$ valideta  :function (eta)
  ..$ simulate  :function (object, nsim)
  ..- attr(*, "class")= chr "family"
 $ linear.predictors: Named num [1:1791] -25.6 -15.8 -18.6 -32.8 -36.5 ...
  ..- attr(*, "names")= chr [1:1791] "1" "2" "3" "4" ...
 $ deviance         : num 306
 $ aic              : num 396
 $ null.deviance    : num 2994
 $ iter             : int 10
 $ weights          : Named num [1:1791] 1.20e-11 1.99e-07 1.29e-08 3.37e-16 3.37e-16 ...
  ..- attr(*, "names")= chr [1:1791] "1" "2" "3" "4" ...
 $ prior.weights    : Named num [1:1791] 1.52 1.52 1.52 1.52 1.52 ...
  ..- attr(*, "names")= chr [1:1791] "1" "2" "3" "4" ...
 $ df.residual      : int 1769
 $ df.null          : int 1790
 $ y                : Named num [1:1791] 0 0 0 0 0 0 0 0 0 0 ...
  ..- attr(*, "names")= chr [1:1791] "1" "2" "3" "4" ...
 $ converged        : logi TRUE
 $ boundary         : logi FALSE
 $ model            :'data.frame':        1791 obs. of  9 variables:
  ..$ Sp277        : int [1:1791] 0 0 0 0 0 0 0 0 0 0 ...
```

```
..$ poly(Var2, 3): poly [1:1791, 1:3] 0.0646 0.0615 0.0615 0.0638 0.0614 ...
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : NULL
.. .. ..$ : chr [1:3] "1" "2" "3"
.. ..- attr(*, "degree")= int [1:3] 1 2 3
.. ..- attr(*, "coefs")=List of 2
.. .. ..$ alpha: num [1:3] 1773 2525 2509
.. .. ..$ norm2: num [1:5] 1.00 1.79e+03 1.52e+09 2.78e+15 3.23e+21
.. ..- attr(*, "class")= chr [1:2] "poly" "matrix"
..$ poly(Var7, 3): poly [1:1791, 1:3] 0.049 0.0478 0.0477 0.0492 0.0505 ...
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : NULL
.. .. ..$ : chr [1:3] "1" "2" "3"
.. ..- attr(*, "degree")= int [1:3] 1 2 3
.. ..- attr(*, "coefs")=List of 2
.. .. ..$ alpha: num [1:3] -3 -5.43 -2.97
.. .. ..$ norm2: num [1:5] 1.00 1.79e+03 8.01e+04 5.55e+06 3.32e+08
.. ..- attr(*, "class")= chr [1:2] "poly" "matrix"
..$ poly(Var5, 3): poly [1:1791, 1:3] 0.0291 0.0424 0.0378 0.0151 0.0156 ...
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : NULL
.. .. ..$ : chr [1:3] "1" "2" "3"
.. ..- attr(*, "degree")= int [1:3] 1 2 3
.. ..- attr(*, "coefs")=List of 2
.. .. ..$ alpha: num [1:3] 178 343 469
.. .. ..$ norm2: num [1:5] 1.00 1.79e+03 1.61e+07 5.05e+11 1.90e+16
.. ..- attr(*, "class")= chr [1:2] "poly" "matrix"
..$ poly(Var1, 3): poly [1:1791, 1:3] -0.0469 -0.0294 -0.0327 -0.0687 -0.0697 ...
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : NULL
.. .. ..$ : chr [1:3] "1" "2" "3"
.. ..- attr(*, "degree")= int [1:3] 1 2 3
.. ..- attr(*, "coefs")=List of 2
.. .. ..$ alpha: num [1:3] 0.913 0.599 0.642
.. .. ..$ norm2: num [1:5] 1.00 1.79e+03 2.73e+01 5.67e-01 1.87e-02
.. ..- attr(*, "class")= chr [1:2] "poly" "matrix"
..$ poly(Var3, 3): poly [1:1791, 1:3] -0.00244 0.00957 0.00517 -0.01386 -0.01368 ...
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : NULL
.. .. ..$ : chr [1:3] "1" "2" "3"
.. ..- attr(*, "degree")= int [1:3] 1 2 3
.. ..- attr(*, "coefs")=List of 2
.. .. ..$ alpha: num [1:3] 802 1479 1794
.. .. ..$ norm2: num [1:5] 1.00 1.79e+03 1.73e+08 7.14e+13 2.94e+19
.. ..- attr(*, "class")= chr [1:2] "poly" "matrix"
..$ poly(Var4, 3): poly [1:1791, 1:3] -0.047 -0.0421 -0.044 -0.0509 -0.0508 ...
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : NULL
.. .. ..$ : chr [1:3] "1" "2" "3"
.. ..- attr(*, "degree")= int [1:3] 1 2 3
.. ..- attr(*, "coefs")=List of 2
.. .. ..$ alpha: num [1:3] 212 314 358
.. .. ..$ norm2: num [1:5] 1.00 1.79e+03 1.35e+07 5.34e+11 2.07e+16
.. ..- attr(*, "class")= chr [1:2] "poly" "matrix"
..$ poly(Var6, 3): poly [1:1791, 1:3] 0.0476 0.0459 0.0459 0.0471 0.0458 ...
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : NULL
.. .. ..$ : chr [1:3] "1" "2" "3"
.. ..- attr(*, "degree")= int [1:3] 1 2 3
.. ..- attr(*, "coefs")=List of 2
.. .. ..$ alpha: num [1:3] 7.52 6.01 8.45
.. .. ..$ norm2: num [1:5] 1 1791 37583 1366318 39616625
.. ..- attr(*, "class")= chr [1:2] "poly" "matrix"
..$ (weights)    : num [1:1791] 1.52 1.52 1.52 1.52 1.52 ...
..- attr(*, "terms")=Classes 'terms', 'formula' length 3 Sp277 ~ poly(Var2, 3) + poly(Var7, 3) + poly(Var5, 3)
.. .. ..- attr(*, "variables")= language list(Sp277, poly(Var2, 3), poly(Var7, 3), poly(Var5, 3), poly(Var1,
.. .. ..- attr(*, "factors")= int [1:8, 1:7] 0 1 0 0 0 0 0 0 0 0 ...
.. .. .. ..- attr(*, "dimnames")=List of 2
.. .. .. .. ..$ : chr [1:8] "Sp277" "poly(Var2, 3)" "poly(Var7, 3)" "poly(Var5, 3)" ...
.. .. .. .. ..$ : chr [1:7] "poly(Var2, 3)" "poly(Var7, 3)" "poly(Var5, 3)" "poly(Var1, 3)" ...
.. .. ..- attr(*, "term.labels")= chr [1:7] "poly(Var2, 3)" "poly(Var7, 3)" "poly(Var5, 3)" "poly(Var1, 3)" .
.. .. ..- attr(*, "order")= int [1:7] 1 1 1 1 1 1 1
```

```
  .. .. ..- attr(*, "intercept")= int 1
  .. .. ..- attr(*, "response")= int 1
  .. .. ..- attr(*, ".Environment")=<environment: 0x0421455c>
  .. .. ..- attr(*, "predvars")= language list(Sp277, poly(Var2, 3, coefs = structure(list(alpha = c(1772.86179(
  .. .. ..- attr(*, "dataClasses")= Named chr [1:9] "numeric" "nmatrix.3" "nmatrix.3" "nmatrix.3" ...
  .. .. .. ..- attr(*, "names")= chr [1:9] "Sp277" "poly(Var2, 3)" "poly(Var7, 3)" "poly(Var5, 3)" ...
 $ call              : language glm(formula = Sp277 ~ poly(Var2, 3) + poly(Var7, 3) + poly(Var5,      3) + poly(V
 $ formula           :Class 'formula' length 3 Sp277 ~ poly(Var2, 3) + poly(Var7, 3) + poly(Var5, 3) + poly(Var1,
  .. ..- attr(*, ".Environment")=<environment: 0x0421455c>
 $ terms             :Classes 'terms', 'formula' length 3 Sp277 ~ poly(Var2, 3) + poly(Var7, 3) + poly(Var5, 3) +
  .. ..- attr(*, "variables")= language list(Sp277, poly(Var2, 3), poly(Var7, 3), poly(Var5, 3), poly(Var1,
  .. ..- attr(*, "factors")= int [1:8, 1:7] 0 1 0 0 0 0 0 0 0 0 ...
  .. .. ..- attr(*, "dimnames")=List of 2
  .. .. .. ..$ : chr [1:8] "Sp277" "poly(Var2, 3)" "poly(Var7, 3)" "poly(Var5, 3)" ...
  .. .. .. ..$ : chr [1:7] "poly(Var2, 3)" "poly(Var7, 3)" "poly(Var5, 3)" "poly(Var1, 3)" ...
  .. ..- attr(*, "term.labels")= chr [1:7] "poly(Var2, 3)" "poly(Var7, 3)" "poly(Var5, 3)" "poly(Var1, 3)" ...
  .. ..- attr(*, "order")= int [1:7] 1 1 1 1 1 1 1
  .. ..- attr(*, "intercept")= int 1
  .. ..- attr(*, "response")= int 1
  .. ..- attr(*, ".Environment")=<environment: 0x0421455c>
  .. ..- attr(*, "predvars")= language list(Sp277, poly(Var2, 3, coefs = structure(list(alpha = c(1772.861790383
  .. ..- attr(*, "dataClasses")= Named chr [1:9] "numeric" "nmatrix.3" "nmatrix.3" "nmatrix.3" ...
  .. .. ..- attr(*, "names")= chr [1:9] "Sp277" "poly(Var2, 3)" "poly(Var7, 3)" "poly(Var5, 3)" ...
 $ data              :'data.frame':        1791 obs. of  9 variables:
  ..$ Var1 : num [1:1791] 0.668 0.76 0.742 0.554 0.549 ...
  ..$ Var2 : num [1:1791] 4296 4174 4173 4264 4169 ...
  ..$ Var3 : num [1:1791] 770 928 870 620 622 ...
  ..$ Var4 : num [1:1791] 39.3 57.3 50.1 25 25.2 ...
  ..$ Var5 : num [1:1791] 295 349 330 239 241 ...
  ..$ Var6 : num [1:1791] 16.7 16.4 16.4 16.7 16.4 ...
  ..$ Var7 : num [1:1791] 10.9 10.5 10.5 10.9 11.3 ...
  ..$ Sp281: int [1:1791] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Sp277: int [1:1791] 0 0 0 0 0 0 0 0 0 0 ...
 $ offset            : NULL
 $ control           :List of 3
  ..$ epsilon: num 1e-08
  ..$ maxit  : num 25
  ..$ trace  : logi FALSE
 $ method            : chr "glm.fit"
 $ contrasts         : NULL
 $ xlevels           : list()
 $ anova             :Classes 'ŚAnovaŠ and 'data.frame':        8 obs. of  6 variables:
  ..$ Step      : Factor w/ 8 levels "","+ poly(Var1, 3)",..: 1 3 8 6 2 4 5 7
  ..$ Df        : num [1:8] NA 3 3 3 3 3 3 3
  ..$ Deviance  : num [1:8] NA 1657 509 205 114 ...
  ..$ Resid. Df : num [1:8] 1790 1787 1784 1781 1778 ...
  ..$ Resid. Dev: num [1:8] 2994 1337 828 623 509 ...
  ..$ AIC       : num [1:8] 3471 1547 974 745 607 ...
  ..- attr(*, "heading")= chr [1:7] "Stepwise Model Path \nAnalysis of Deviance Table" "\nInitial Model:" "Sp277
 - attr(*, "class")= chr [1:2] "glm" "lm"
```

```
> #summary
> summary(Sp277_GLM_PA1)


Call:
glm(formula = Sp277 ~ poly(Var2, 3) + poly(Var7, 3) + poly(Var5,
    3) + poly(Var1, 3) + poly(Var3, 3) + poly(Var4, 3) + poly(Var6,
    3), family = binomial, data = DataBIOMOD[calib.lines, ],
    weights = Yweights[calib.lines, i])

Deviance Residuals:
      Min         1Q     Median         3Q        Max
-2.91e+00  -2.67e-04   8.32e-05   1.87e-02   2.58e+00

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)       0.344      1.052    0.33   0.7437
poly(Var2, 3)1 -598.966    340.449   -1.76   0.0785 .
poly(Var2, 3)2 -693.252    125.512   -5.52  3.3e-08 ***
poly(Var2, 3)3  -39.381     20.200   -1.95   0.0512 .
```

```
poly(Var7, 3)1 -135.059    136.929   -0.99   0.3240
poly(Var7, 3)2 -700.546     84.270   -8.31  < 2e-16 ***
poly(Var7, 3)3   83.745     72.402    1.16   0.2474
poly(Var5, 3)1 -132.451     69.866   -1.90   0.0580 .
poly(Var5, 3)2   67.663     45.108    1.50   0.1336
poly(Var5, 3)3  204.751     36.798    5.56  2.6e-08 ***
poly(Var1, 3)1  -14.084     31.784   -0.44   0.6577
poly(Var1, 3)2  -36.087     13.295   -2.71   0.0066 **
poly(Var1, 3)3  -81.726     13.569   -6.02  1.7e-09 ***
poly(Var3, 3)1  187.879    104.949    1.79   0.0734 .
poly(Var3, 3)2 -405.403     90.675   -4.47  7.8e-06 ***
poly(Var3, 3)3 -243.330     48.658   -5.00  5.7e-07 ***
poly(Var4, 3)1  112.524     59.176    1.90   0.0572 .
poly(Var4, 3)2   48.763     43.591    1.12   0.2633
poly(Var4, 3)3  282.348     47.353    5.96  2.5e-09 ***
poly(Var6, 3)1 1018.986    422.797    2.41   0.0159 *
poly(Var6, 3)2 1149.032    209.588    5.48  4.2e-08 ***
poly(Var6, 3)3  244.607     54.432    4.49  7.0e-06 ***
---
Signif. codes:  0 Ś***Š 0.001 Ś**Š 0.01 Ś*Š 0.05 Ś.Š 0.1 Ś Š 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2994.40  on 1790  degrees of freedom
Residual deviance:  306.11  on 1769  degrees of freedom
AIC: 395.7

Number of Fisher Scoring iterations: 10
```

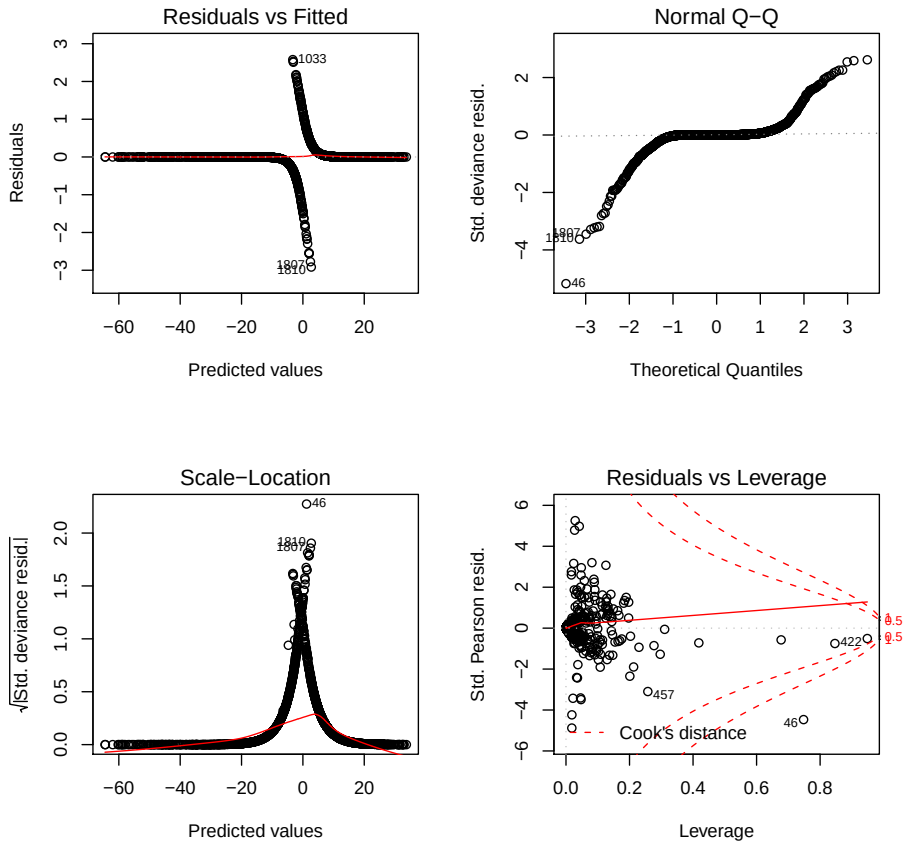
It shows the information stored, like the different variables retained in the final model.
The outputs also give the different coefficient values, the degrees of freedom, the residual deviance and the AIC of the final model. Of course, each model's outputs will not give the same information, as it depends on its specificity.
For the GBM, the summary computes the relative influence of each variable in the gbm object. This returns the reduction attributable to each variable in sum of squared error in predicting the gradient on each iteration. It describes the relative influence of each variable in reducing the loss function. It returns a data frame where the first component is the variable name and the second is the computed relative influence, normalized to sum up to 100.

The next call obtains the anova results and the details of the stepwise procedure type. Note that the independent variables are ranked by their AIC importance.

```
> Sp277_GLM_PA1$anova


Stepwise Model Path
Analysis of Deviance Table

Initial Model:
Sp277 ~ 1

Final Model:
Sp277 ~ poly(Var2, 3) + poly(Var7, 3) + poly(Var5, 3) + poly(Var1,
    3) + poly(Var3, 3) + poly(Var4, 3) + poly(Var6, 3)


             Step Df Deviance Resid. Df Resid. Dev    AIC
1                                  1790     2994.4 3470.5
2 + poly(Var2, 3)  3  1657.42      1787     1337.0 1547.0
3 + poly(Var7, 3)  3   509.05      1784      827.9  974.0
4 + poly(Var5, 3)  3   204.71      1781      623.2  745.3
5 + poly(Var1, 3)  3   114.01      1778      509.2  607.0
6 + poly(Var3, 3)  3    88.43      1775      420.8  511.0
7 + poly(Var4, 3)  3    63.39      1772      357.4  444.6
8 + poly(Var6, 3)  3    51.28      1769      306.1  395.7
```

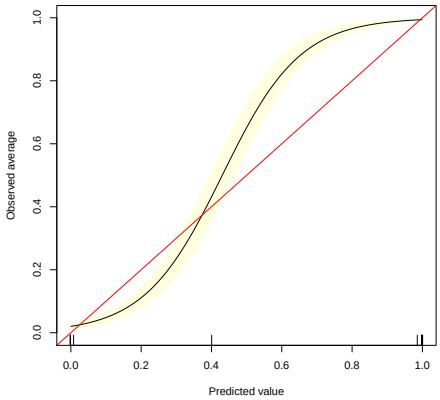
The function *plot* of R will give the basic and usual outputs for GLM. They are useful but not entirely relevant in the case of the logistic regression.

```
> par(mfrow=c(2,2))
> plot(Sp277_GLM_PA1)
```



The *gbm* library also provides an experimental diagnostic tool that plots the fitted values versus the actual average values. Uses gam to estimate $E(y|p)$. Well-calibrated predictions imply that $E(y|p) = p$. The plot also includes a pointwise 95 band.
This method can be applied to all models to visualise the relative goodness of fit of the model. The function requires the observed presence-absence of the selected species and the predictions. Hence, you will need top load the predictions for this.

```
> library(gbm)
> load("pred/Pred_Sp277")
> #let's store the data that was used for calibration of the
> #first PA run for Sp277 to simplify the code
> data.used <- DataBIOMOD[Biomod.PA.sample$Sp277$PA1,"Sp277"]
> calibrate.plot(data.used, Pred_Sp277[,"GLM",1,1]/1000)
```

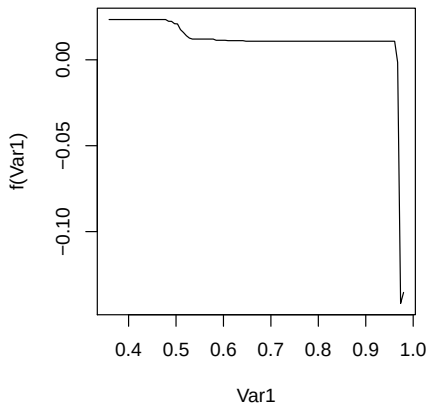


Model Specificities

GBM

The GBM library allows to plot the response curves of the species against the environmental variables selected by the models.
i.var: a vector of indices or the names of the variables to plot. If using indices, the variables are indexed in the same order as they appear in the initial 'gbm' formula. For instance, here BIOMOD will plot the first variable in the model.

```
> load("models/Sp281_GBM_PA1")
> plot(Sp281_GBM_PA1, i.var=1)
```



CTA

There are several useful outputs in CTA models. A critical one is *frame* which gives the details of the node, the explained deviance by each node (dev) and the probability of occurrences (yval).

```
> load("models/Sp277_CTA_PA1")
> names(Sp277_CTA_PA1)
```

```
 [1] "frame"     "where"     "call"      "terms"     "cptable"
 [6] "splits"    "method"    "parms"     "control"   "functions"
[11] "y"         "ordered"
```

```
> Sp277_CTA_PA1$frame
```

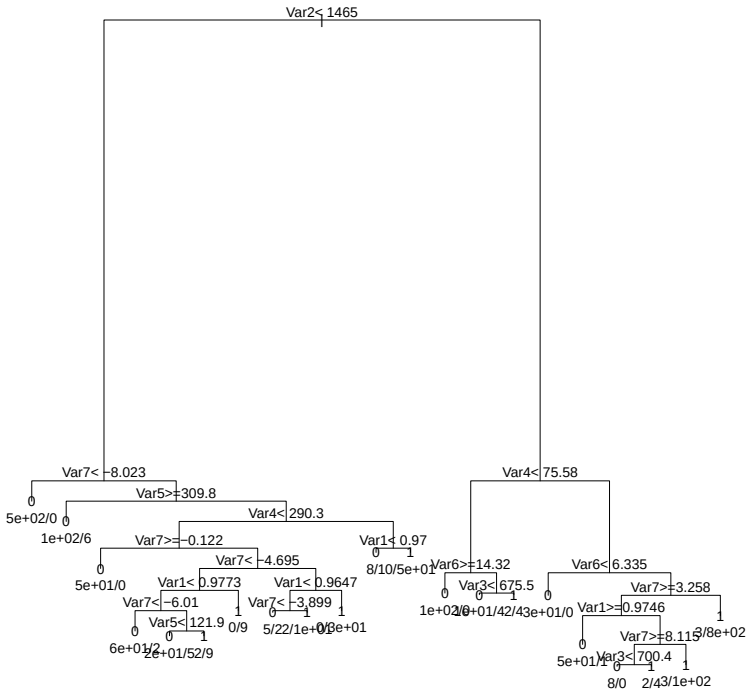
```
       var    n       wt      dev yval complexity ncompete nsurrogate
1     Var2 1791 2160.000 1080.000    1   0.661041        4          5
2     Var7  669  953.924  120.000    1   0.012779        4          5
4   <leaf>  362  549.873    0.000    1   0.001000        0          0
5     Var5  307  404.051  120.000    1   0.012779        4          5
10  <leaf>  101  150.304    6.000    1   0.000000        0          0
11    Var4  206  253.747  114.000    1   0.012779        4          4
22    Var7  152  197.152   65.000    1   0.012779        4          4
44  <leaf>   30   45.570    0.000    1   0.001000        0          0
45    Var7  122  151.582   65.000    1   0.012779        4          5
90    Var1   78  105.506   25.000    1   0.008333        4          1
180   Var7   69   96.506   16.000    1   0.003463        4          5
360 <leaf>   41   61.241    2.000    1   0.000000        0          0
361   Var5   28   35.266   14.000    1   0.003463        4          5
722 <leaf>   18   24.747    5.000    1   0.000000        0          0
723 <leaf>   10   10.519    1.519    2   0.000000        0          0
181 <leaf>    9    9.000    0.000    2   0.001000        0          0
91    Var1   44   46.076    6.076    2   0.001184        4          5
182   Var7   16   18.076    6.076    2   0.001184        4          5
364 <leaf>    5    6.557    2.000    1   0.001000        0          0
365 <leaf>   11   11.519    1.519    2   0.000000        0          0
183 <leaf>   28   28.000    0.000    2   0.001000        0          0
23    Var1   54   56.595    7.595    2   0.006106        4          2
46  <leaf>    6    8.595    1.000    1   0.001000        0          0
47  <leaf>   48   48.000    0.000    2   0.001000        0          0
3     Var4 1122 1206.076  246.076    2   0.131833        4          5
6     Var6  107  158.380    8.000    1   0.001149        4          3
12  <leaf>   89  135.190    0.000    1   0.001000        0          0
13    Var3   18   23.190    8.000    1   0.001149        4          4
26  <leaf>   13   17.671    4.000    1   0.000000        0          0
27  <leaf>    5    5.519    1.519    2   0.001000        0          0
7     Var6 1015 1047.696   95.696    2   0.032349        4          1
14  <leaf>   23   34.937    0.000    1   0.001000        0          0
15    Var7  992 1012.759   60.759    2   0.020634        4          5
30    Var1  189  208.722   57.722    2   0.020634        4          5
60  <leaf>   31   46.570    1.000    1   0.001000        0          0
61    Var7  158  162.152   12.152    2   0.004735        4          5
122   Var3   10   13.114    4.000    1   0.002297        4          5
244 <leaf>    5    7.595    0.000    1   0.001000        0          0
245 <leaf>    5    5.519    1.519    2   0.001000        0          0
123 <leaf>  148  149.038    3.038    2   0.000000        0          0
31  <leaf>  803  804.038    3.038    2   0.000000        0          0
      yval2.1   yval2.2   yval2.3   yval2.4   yval2.5
1   1.000e+00 1.080e+03 1.080e+03 5.000e-01 5.000e-01
2   1.000e+00 8.339e+02 1.200e+02 8.742e-01 1.258e-01
4   1.000e+00 5.499e+02 0.000e+00 1.000e+00 0.000e+00
5   1.000e+00 2.841e+02 1.200e+02 7.030e-01 2.970e-01
10  1.000e+00 1.443e+02 6.000e+00 9.601e-01 3.992e-02
11  1.000e+00 1.397e+02 1.140e+02 5.507e-01 4.493e-01
22  1.000e+00 1.322e+02 6.500e+01 6.703e-01 3.297e-01
44  1.000e+00 4.557e+01 0.000e+00 1.000e+00 0.000e+00
45  1.000e+00 8.658e+01 6.500e+01 5.712e-01 4.288e-01
90  1.000e+00 8.051e+01 2.500e+01 7.630e-01 2.370e-01
180 1.000e+00 8.051e+01 1.600e+01 8.342e-01 1.658e-01
360 1.000e+00 5.924e+01 2.000e+00 9.673e-01 3.266e-02
361 1.000e+00 2.127e+01 1.400e+01 6.030e-01 3.970e-01
722 1.000e+00 1.975e+01 5.000e+00 7.980e-01 2.020e-01
723 2.000e+00 1.519e+00 9.000e+00 1.444e-01 8.556e-01
181 2.000e+00 0.000e+00 9.000e+00 0.000e+00 1.000e+00
91  2.000e+00 6.076e+00 4.000e+01 1.319e-01 8.681e-01
182 2.000e+00 6.076e+00 1.200e+01 3.361e-01 6.639e-01
364 1.000e+00 4.557e+00 2.000e+00 6.950e-01 3.050e-01
365 2.000e+00 1.519e+00 1.000e+01 1.319e-01 8.681e-01
183 2.000e+00 0.000e+00 2.800e+01 0.000e+00 1.000e+00
```

```
23  2.000e+00 7.595e+00 4.900e+01 1.342e-01 8.658e-01
46  1.000e+00 7.595e+00 1.000e+00 8.837e-01 1.163e-01
47  2.000e+00 0.000e+00 4.800e+01 0.000e+00 1.000e+00
3   2.000e+00 2.461e+02 9.600e+02 2.040e-01 7.960e-01
6   1.000e+00 1.504e+02 8.000e+00 9.495e-01 5.051e-02
12  1.000e+00 1.352e+02 0.000e+00 1.000e+00 0.000e+00
13  1.000e+00 1.519e+01 8.000e+00 6.550e-01 3.450e-01
26  1.000e+00 1.367e+01 4.000e+00 7.736e-01 2.264e-01
27  2.000e+00 1.519e+00 4.000e+00 2.752e-01 7.248e-01
7   2.000e+00 9.570e+01 9.520e+02 9.134e-02 9.087e-01
14  1.000e+00 3.494e+01 0.000e+00 1.000e+00 0.000e+00
15  2.000e+00 6.076e+01 9.520e+02 5.999e-02 9.400e-01
30  2.000e+00 5.772e+01 1.510e+02 2.765e-01 7.235e-01
60  1.000e+00 4.557e+01 1.000e+00 9.785e-01 2.147e-02
61  2.000e+00 1.215e+01 1.500e+02 7.494e-02 9.251e-01
122 1.000e+00 9.114e+00 4.000e+00 6.950e-01 3.050e-01
244 1.000e+00 7.595e+00 0.000e+00 1.000e+00 0.000e+00
245 2.000e+00 1.519e+00 4.000e+00 2.752e-01 7.248e-01
123 2.000e+00 3.038e+00 1.460e+02 2.038e-02 9.796e-01
31  2.000e+00 3.038e+00 8.010e+02 3.778e-03 9.962e-01
```

This table is easier to read by plotting the tree in the same time.
Make sure the *rpart* library is loaded.

Note that the *plot* function does not display the label and text by default. The user must use the *text* function to add the text

```
> plot(Sp277_CTA_PA1, margin=0.05)
> text(Sp277_CTA_PA1, use.n=T)
```



RF

The importance of each variable, as produced by random Forest, can be extracted.

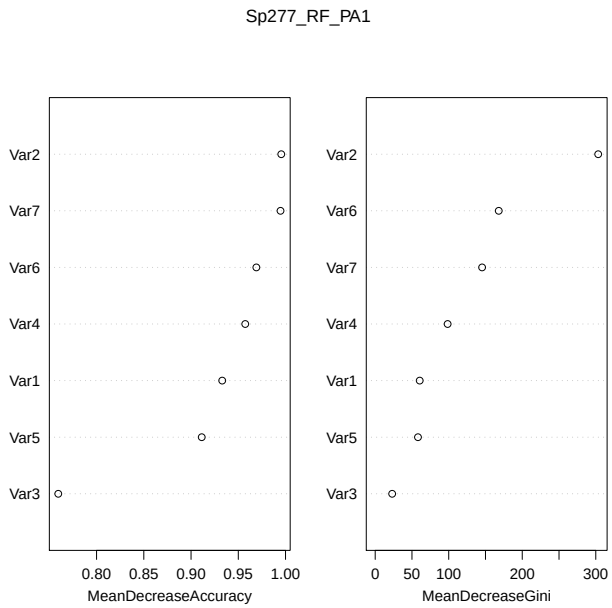
```
> load("models/Sp277_RF_PA1")
> Sp277_RF_PA1$importance
```

```
           0       1 MeanDecreaseAccuracy MeanDecreaseGini
Var1 0.07039 0.03431              0.04855            60.55
Var2 0.17893 0.20870              0.19657           303.28
Var3 0.03847 0.01520              0.02440            23.09
Var4 0.07477 0.02437              0.04432            98.52
Var5 0.06070 0.01789              0.03482            58.19
Var6 0.17244 0.19320              0.18491           168.02
Var7 0.19162 0.07762              0.12269           145.39
```

Here are the definitions of the variables' importance measures.
- Mean Decrease Accuracy: For each tree, the prediction accuracy on the out-of-bag portion of the data is recorded. Then the same is done after permuting each predictor variable. The difference between the two accuracies are then averaged across all trees, and normalized by the standard error.
- Mean Decrease Gini: The second measure is the total decrease in node impurities from splitting on the variable, averaged over all trees. For classification, the node impurity is measured by the Gini index.

Similarly, a dotchart of variable importance as measured by a Random Forest can be plotted.

```
> varImpPlot(Sp277_RF_PA1)
```

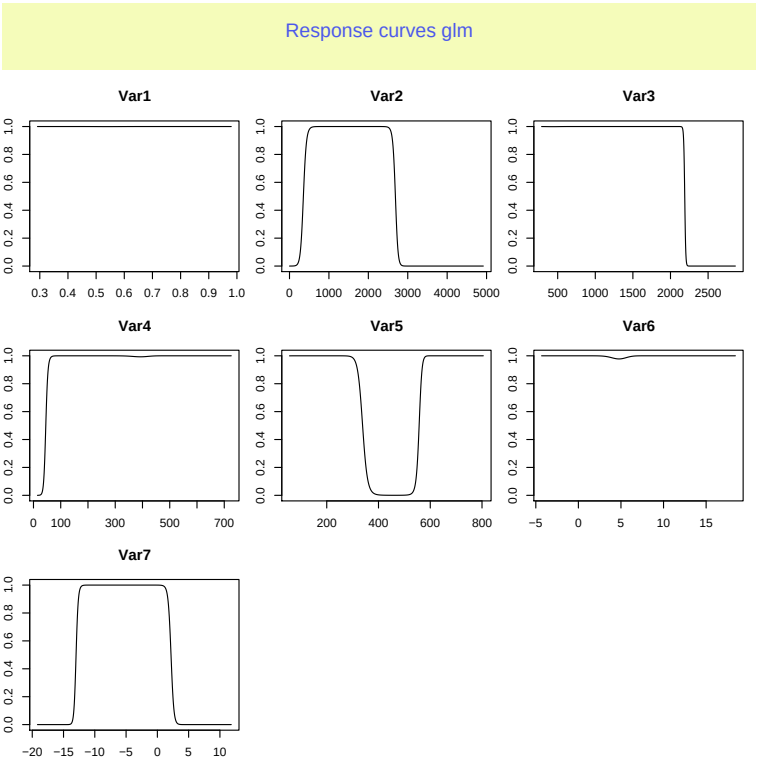


Response Curves

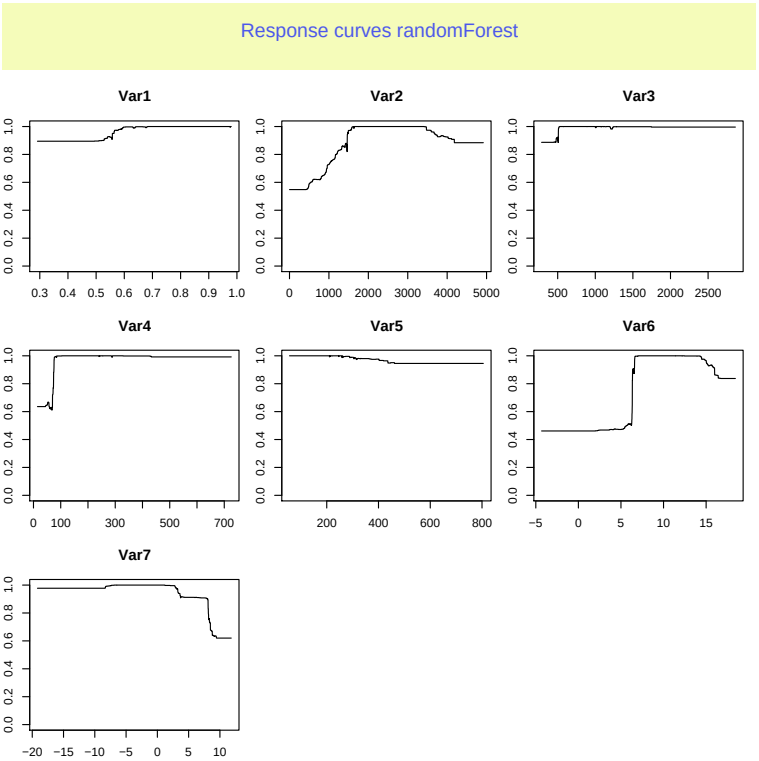
BIOMOD allows plotting the response curves of every model in the good scale. The *response.plot* function must be used to this matter. This function requires a model and a related set of variables to plot the response curves.

Here are two examples of the GLM and RF for the first species modelled. You need to load the model, type its name in the first argument, then give the variables for which you want to see the curves. Note the you can choose to only show some of the variables with the *show.variables* argument.

```
> #this one has already been loaded in a prior call
> response.plot(Sp277_GLM_PA1, Sp.Env[4:10])
```




```
> load("models/Sp277_RF_PA1")
> response.plot(Sp277_RF_PA1, Sp.Env[4:10])
```



The response curves are generated following this calculation : N-1 variables are held constant at their mean value whilst the variable of interest contains 100 points varying across the maximum and the minimum of its range. Variation in predictions, made to these 100 cells, only reflects the effects of the one selected variable. Thus, a plot of these predictions allows visualisation of the modelled response to the variable of interest, contingent on the other variables being held constant. This is done subsequently for all the selected variables.

In our examples, the variable 6 doesn't seem to have a great influence for the GLM (very few variations in the prediction staying close to 1) when it shows a non negligeable influence in the predictions of the RF approximately scaling from 0.4 to 1.

0.4.3 Objects stored on the hard drive : The Predictions

The predictions made by each model on the original dataset are stored inside the *pred* folder. They are stored independently for each species in an object following a 'Pred.Speciesname' logic and contains the probability of occurrence (habitat suitability index) for each run (if several runs) of the selected models. The same objects are produced for the independent data (if any) and the same logic is respected for the projections.

NOTE: for calculation and memory storage purposes, this index is on a scale between 0 and 1000. To obtain a true probability of occurrence, rescaled between 0 and 1, simply divide each value by a thousand.

```
> load("pred/Pred_Sp277")
```

The trick is that these objects are no longer matrices but arrays (multiple dimensions) with 4 dimensions. The dimensions can be visualised as follows :

The first two build up a matrix where each column is the prediction of one of the models. The number of rows corresponds to the amount of data used for building those models.

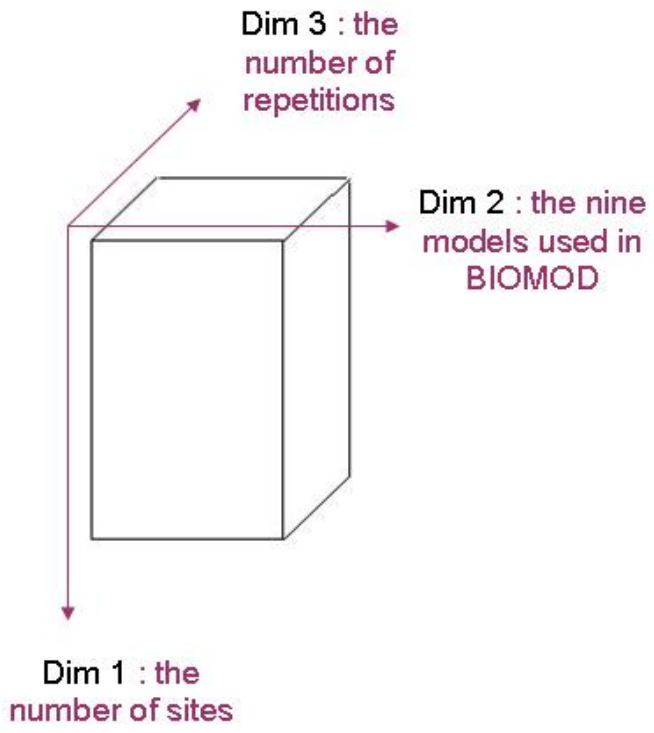
```
> dim(Pred_Sp277)
```

```
[1] 1791    9    4    1
```

```
> Pred_Sp277[1:20,,1,1]
```

```
   ANN CTA GAM GBM GLM MARS FDA RF SRE
1  896   0   2 456   0    1  38  4   0
2  896   0  34 456   0   18  38  1   0
3  896   0  22 456   0    8  38  2   0
4  896   0   0 456   0    0  37  0   0
5  896   0   0 456   0    0  37  0   0
6  896   0   1 456   0    0  37  0   0
7  896   0   0 456   0    0  37  0   0
8  896   0   1 456   0    0  37  6   0
9  896   0   0 456   0    0  37  0   0
10 896   0   4 456   0    1  38  2   0
11 896   0   1 456   0    0  37  0   0
12 896   0 137 456   0   52  38  1   0
13 896   0   8 456   0    1  38  1   0
14 896   0  93 456   0    9  38  2   0
15 896   0  10 456   0    2  38  9   0
16 896   0 286 456   0   97  38  5   0
17 896   0   3 456   0    0  37  0   0
18 896   0   1 456   0    0  37  0   0
19 896   0  13 456   0    1  38  0   0
20 896   0   1 456   0    0  37  0   0
```

Now, the third dimensions consists of a collection of 2-D matrices, one behind another, corresponding to the prediction produced by each repetition. The minimum for this dimension is 1. Considering that BIOMOD always produces a final model calibrated with 100% of the data given, the length of this third dimension is the value of the NbRunEval argument + 1. For example, with NbRunEval=10, you have 11 layers.



Note that the firts layer is always the final model, then come the repetitions.

```
> #the final model
> Pred_Sp277[1:15,,1,1]
```

	ANN	CTA	GAM	GBM	GLM	MARS	FDA	RF	SRE
1	896	0	2	456	0	1	38	4	0
2	896	0	34	456	0	18	38	1	0
3	896	0	22	456	0	8	38	2	0
4	896	0	0	456	0	0	37	0	0
5	896	0	0	456	0	0	37	0	0
6	896	0	1	456	0	0	37	0	0
7	896	0	0	456	0	0	37	0	0
8	896	0	1	456	0	0	37	6	0
9	896	0	0	456	0	0	37	0	0
10	896	0	4	456	0	1	38	2	0
11	896	0	1	456	0	0	37	0	0
12	896	0	137	456	0	52	38	1	0
13	896	0	8	456	0	1	38	1	0
14	896	0	93	456	0	9	38	2	0
15	896	0	10	456	0	2	38	9	0

```
> #the first repetition model
> Pred_Sp277[1:15,,2,1]
```

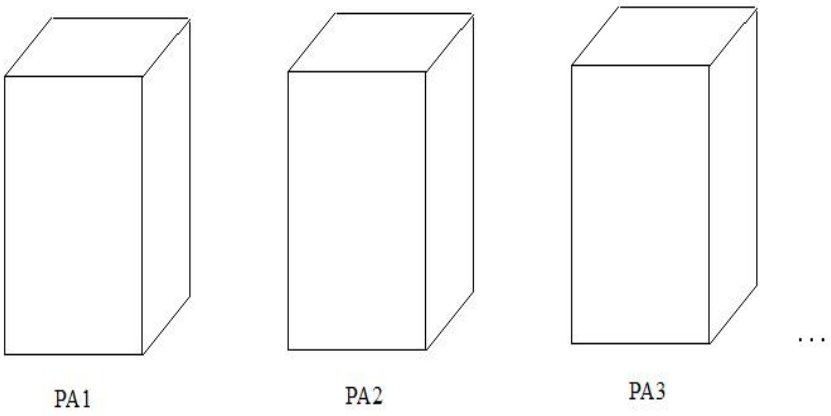
	ANN	CTA	GAM	GBM	GLM	MARS	FDA	RF	SRE
1	15	0	1	455	0	1	39	4	0
2	15	0	37	455	0	17	39	6	0
3	15	0	22	455	0	8	39	2	0
4	15	0	0	455	0	0	39	0	0
5	15	0	0	455	0	0	39	2	0
6	15	0	0	455	0	0	39	0	0

```
7   15   0   0 455   0    0  39  0   0
8   15   0   0 455   0    0  39  0   0
9   15   0   0 455   0    0  39  0   0
10  15   0   3 455   0    1  39  0   0
11  15   0   0 455   0    0  39  0   0
12  15   0 184 455   0   53  41  9   0
13  15   0   6 455   0    1  39  8   0
14  15   0  93 455   0   10  41  4   0
15  15   0   7 455   0    2  39  9   0
```

```
> #the second repetition model
> Pred_Sp277[1:15,,3,1]
```

```
   ANN CTA GAM GBM GLM MARS FDA RF SRE
1   37  52   2 456   0    3  27 13   0
2   38  52  29 458   0   44  27 22   0
3   37  52  18 456   0   16  27 17   0
4   37  52   0 456   0    0  27  0   0
5   37  52   0 456   0    0  27  0   0
6   37  52   1 456   0    0  27  1   0
7   37  52   0 456   0    0  27  0   0
8   37  52   1 456   0    0  27  5   0
9   37  52   1 456   0    0  27  2   0
10  37  52   5 456   0    1  27  4   0
11  37  52   2 456   0    0  27  0   0
12  39  52 129 458   0   97  27 17   0
13  37  52   6 456   0    3  27 13   0
14  37  52  75 456   0   17  27 21   0
15  37  52   9 456   0    3  27 12   0
```

The fourth dimension represents the number of pseudo-absences repetitions that have been made. In the case where NbRepPA=0, the dimension is simply 1 (not 0).



You will never visualise it this way with R though. It is just an abstract view of how it is sorted. Some usefull functions for not getting lost are *dim()* and *dimnames()*. The first one gives you the number of layers for each dimension, the second will give you their names respectively.

```
> load("pred/Pred_Sp281")
> dim(Pred_Sp281)
```

```
[1] 1392    9    4    2
```

```
> dimnames(Pred_Sp281)
```

```
[[1]]
   [1] "1"    "2"    "3"    "4"    "5"    "6"    "7"    "8"    "9"
  [10] "10"   "11"   "12"   "13"   "14"   "15"   "16"   "17"   "18"
  [19] "19"   "20"   "21"   "22"   "23"   "24"   "25"   "26"   "27"
  [28] "28"   "29"   "30"   "31"   "32"   "33"   "34"   "35"   "36"
  [37] "37"   "38"   "39"   "40"   "41"   "42"   "43"   "44"   "45"
  [46] "46"   "47"   "48"   "49"   "50"   "51"   "52"   "53"   "54"
  [55] "55"   "56"   "57"   "58"   "59"   "60"   "61"   "62"   "63"
  [64] "64"   "65"   "66"   "67"   "68"   "69"   "70"   "71"   "72"
  [73] "73"   "74"   "75"   "76"   "77"   "78"   "79"   "80"   "81"
  [82] "82"   "83"   "84"   "85"   "86"   "87"   "88"   "89"   "90"
  [91] "91"   "92"   "93"   "94"   "95"   "96"   "97"   "98"   "99"
 [100] "100"  "101"  "102"  "103"  "104"  "105"  "106"  "107"  "108"
 [109] "109"  "110"  "111"  "112"  "113"  "114"  "115"  "116"  "117"
 [118] "118"  "119"  "120"  "121"  "122"  "123"  "124"  "125"  "126"
 [127] "127"  "128"  "129"  "130"  "131"  "132"  "133"  "134"  "135"
 [136] "136"  "137"  "138"  "139"  "140"  "141"  "142"  "143"  "144"
 [145] "145"  "146"  "147"  "148"  "149"  "150"  "151"  "152"  "153"
 [154] "154"  "155"  "156"  "157"  "158"  "159"  "160"  "161"  "162"
 [163] "163"  "164"  "165"  "166"  "167"  "168"  "169"  "170"  "171"
 [172] "172"  "173"  "174"  "175"  "176"  "177"  "178"  "179"  "180"
 [181] "181"  "182"  "183"  "184"  "185"  "186"  "187"  "188"  "189"
 [190] "190"  "191"  "192"  "193"  "194"  "195"  "196"  "197"  "198"
 [199] "199"  "200"  "201"  "202"  "203"  "204"  "205"  "206"  "207"
 [208] "208"  "209"  "210"  "211"  "212"  "213"  "214"  "215"  "216"
 [217] "217"  "218"  "219"  "220"  "221"  "222"  "223"  "224"  "225"
 [226] "226"  "227"  "228"  "229"  "230"  "231"  "232"  "233"  "234"
 [235] "235"  "236"  "237"  "238"  "239"  "240"  "241"  "242"  "243"
 [244] "244"  "245"  "246"  "247"  "248"  "249"  "250"  "251"  "252"
 [253] "253"  "254"  "255"  "256"  "257"  "258"  "259"  "260"  "261"
 [262] "262"  "263"  "264"  "265"  "266"  "267"  "268"  "269"  "270"
 [271] "271"  "272"  "273"  "274"  "275"  "276"  "277"  "278"  "279"
 [280] "280"  "281"  "282"  "283"  "284"  "285"  "286"  "287"  "288"
 [289] "289"  "290"  "291"  "292"  "293"  "294"  "295"  "296"  "297"
 [298] "298"  "299"  "300"  "301"  "302"  "303"  "304"  "305"  "306"
 [307] "307"  "308"  "309"  "310"  "311"  "312"  "313"  "314"  "315"
 [316] "316"  "317"  "318"  "319"  "320"  "321"  "322"  "323"  "324"
 [325] "325"  "326"  "327"  "328"  "329"  "330"  "331"  "332"  "333"
 [334] "334"  "335"  "336"  "337"  "338"  "339"  "340"  "341"  "342"
 [343] "343"  "344"  "345"  "346"  "347"  "348"  "349"  "350"  "351"
 [352] "352"  "353"  "354"  "355"  "356"  "357"  "358"  "359"  "360"
 [361] "361"  "362"  "363"  "364"  "365"  "366"  "367"  "368"  "369"
 [370] "370"  "371"  "372"  "373"  "374"  "375"  "376"  "377"  "378"
 [379] "379"  "380"  "381"  "382"  "383"  "384"  "385"  "386"  "387"
 [388] "388"  "389"  "390"  "391"  "392"  "393"  "394"  "395"  "396"
 [397] "397"  "398"  "399"  "400"  "401"  "402"  "403"  "404"  "405"
 [406] "406"  "407"  "408"  "409"  "410"  "411"  "412"  "413"  "414"
 [415] "415"  "416"  "417"  "418"  "419"  "420"  "421"  "422"  "423"
 [424] "424"  "425"  "426"  "427"  "428"  "429"  "430"  "431"  "432"
 [433] "433"  "434"  "435"  "436"  "437"  "438"  "439"  "440"  "441"
 [442] "442"  "443"  "444"  "445"  "446"  "447"  "448"  "449"  "450"
 [451] "451"  "452"  "453"  "454"  "455"  "456"  "457"  "458"  "459"
 [460] "460"  "461"  "462"  "463"  "464"  "465"  "466"  "467"  "468"
 [469] "469"  "470"  "471"  "472"  "473"  "474"  "475"  "476"  "477"
 [478] "478"  "479"  "480"  "481"  "482"  "483"  "484"  "485"  "486"
 [487] "487"  "488"  "489"  "490"  "491"  "492"  "493"  "494"  "495"
 [496] "496"  "497"  "498"  "499"  "500"  "501"  "502"  "503"  "504"
 [505] "505"  "506"  "507"  "508"  "509"  "510"  "511"  "512"  "513"
 [514] "514"  "515"  "516"  "517"  "518"  "519"  "520"  "521"  "522"
 [523] "523"  "524"  "525"  "526"  "527"  "528"  "529"  "530"  "531"
 [532] "532"  "533"  "534"  "535"  "536"  "537"  "538"  "539"  "540"
 [541] "541"  "542"  "543"  "544"  "545"  "546"  "547"  "548"  "549"
 [550] "550"  "551"  "552"  "553"  "554"  "555"  "556"  "557"  "558"
 [559] "559"  "560"  "561"  "562"  "563"  "564"  "565"  "566"  "567"
 [568] "568"  "569"  "570"  "571"  "572"  "573"  "574"  "575"  "576"
 [577] "577"  "578"  "579"  "580"  "581"  "582"  "583"  "584"  "585"
```

```
 [586] "586"  "587"  "588"  "589"  "590"  "591"  "592"  "593"  "594"
 [595] "595"  "596"  "597"  "598"  "599"  "600"  "601"  "602"  "603"
 [604] "604"  "605"  "606"  "607"  "608"  "609"  "610"  "611"  "612"
 [613] "613"  "614"  "615"  "616"  "617"  "618"  "619"  "620"  "621"
 [622] "622"  "623"  "624"  "625"  "626"  "627"  "628"  "629"  "630"
 [631] "631"  "632"  "633"  "634"  "635"  "636"  "637"  "638"  "639"
 [640] "640"  "641"  "642"  "643"  "644"  "645"  "646"  "647"  "648"
 [649] "649"  "650"  "651"  "652"  "653"  "654"  "655"  "656"  "657"
 [658] "658"  "659"  "660"  "661"  "662"  "663"  "664"  "665"  "666"
 [667] "667"  "668"  "669"  "670"  "671"  "672"  "673"  "674"  "675"
 [676] "676"  "677"  "678"  "679"  "680"  "681"  "682"  "683"  "684"
 [685] "685"  "686"  "687"  "688"  "689"  "690"  "691"  "692"  "693"
 [694] "694"  "695"  "696"  "697"  "698"  "699"  "700"  "701"  "702"
 [703] "703"  "704"  "705"  "706"  "707"  "708"  "709"  "710"  "711"
 [712] "712"  "713"  "714"  "715"  "716"  "717"  "718"  "719"  "720"
 [721] "721"  "722"  "723"  "724"  "725"  "726"  "727"  "728"  "729"
 [730] "730"  "731"  "732"  "733"  "734"  "735"  "736"  "737"  "738"
 [739] "739"  "740"  "741"  "742"  "743"  "744"  "745"  "746"  "747"
 [748] "748"  "749"  "750"  "751"  "752"  "753"  "754"  "755"  "756"
 [757] "757"  "758"  "759"  "760"  "761"  "762"  "763"  "764"  "765"
 [766] "766"  "767"  "768"  "769"  "770"  "771"  "772"  "773"  "774"
 [775] "775"  "776"  "777"  "778"  "779"  "780"  "781"  "782"  "783"
 [784] "784"  "785"  "786"  "787"  "788"  "789"  "790"  "791"  "792"
 [793] "793"  "794"  "795"  "796"  "797"  "798"  "799"  "800"  "801"
 [802] "802"  "803"  "804"  "805"  "806"  "807"  "808"  "809"  "810"
 [811] "811"  "812"  "813"  "814"  "815"  "816"  "817"  "818"  "819"
 [820] "820"  "821"  "822"  "823"  "824"  "825"  "826"  "827"  "828"
 [829] "829"  "830"  "831"  "832"  "833"  "834"  "835"  "836"  "837"
 [838] "838"  "839"  "840"  "841"  "842"  "843"  "844"  "845"  "846"
 [847] "847"  "848"  "849"  "850"  "851"  "852"  "853"  "854"  "855"
 [856] "856"  "857"  "858"  "859"  "860"  "861"  "862"  "863"  "864"
 [865] "865"  "866"  "867"  "868"  "869"  "870"  "871"  "872"  "873"
 [874] "874"  "875"  "876"  "877"  "878"  "879"  "880"  "881"  "882"
 [883] "883"  "884"  "885"  "886"  "887"  "888"  "889"  "890"  "891"
 [892] "892"  "893"  "894"  "895"  "896"  "897"  "898"  "899"  "900"
 [901] "901"  "902"  "903"  "904"  "905"  "906"  "907"  "908"  "909"
 [910] "910"  "911"  "912"  "913"  "914"  "915"  "916"  "917"  "918"
 [919] "919"  "920"  "921"  "922"  "923"  "924"  "925"  "926"  "927"
 [928] "928"  "929"  "930"  "931"  "932"  "933"  "934"  "935"  "936"
 [937] "937"  "938"  "939"  "940"  "941"  "942"  "943"  "944"  "945"
 [946] "946"  "947"  "948"  "949"  "950"  "951"  "952"  "953"  "954"
 [955] "955"  "956"  "957"  "958"  "959"  "960"  "961"  "962"  "963"
 [964] "964"  "965"  "966"  "967"  "968"  "969"  "970"  "971"  "972"
 [973] "973"  "974"  "975"  "976"  "977"  "978"  "979"  "980"  "981"
 [982] "982"  "983"  "984"  "985"  "986"  "987"  "988"  "989"  "990"
 [991] "991"  "992"  "993"  "994"  "995"  "996"  "997"  "998"  "999"
[1000] "1000" "1001" "1002" "1003" "1004" "1005" "1006" "1007" "1008"
[1009] "1009" "1010" "1011" "1012" "1013" "1014" "1015" "1016" "1017"
[1018] "1018" "1019" "1020" "1021" "1022" "1023" "1024" "1025" "1026"
[1027] "1027" "1028" "1029" "1030" "1031" "1032" "1033" "1034" "1035"
[1036] "1036" "1037" "1038" "1039" "1040" "1041" "1042" "1043" "1044"
[1045] "1045" "1046" "1047" "1048" "1049" "1050" "1051" "1052" "1053"
[1054] "1054" "1055" "1056" "1057" "1058" "1059" "1060" "1061" "1062"
[1063] "1063" "1064" "1065" "1066" "1067" "1068" "1069" "1070" "1071"
[1072] "1072" "1073" "1074" "1075" "1076" "1077" "1078" "1079" "1080"
[1081] "1081" "1082" "1083" "1084" "1085" "1086" "1087" "1088" "1089"
[1090] "1090" "1091" "1092" "1093" "1094" "1095" "1096" "1097" "1098"
[1099] "1099" "1100" "1101" "1102" "1103" "1104" "1105" "1106" "1107"
[1108] "1108" "1109" "1110" "1111" "1112" "1113" "1114" "1115" "1116"
[1117] "1117" "1118" "1119" "1120" "1121" "1122" "1123" "1124" "1125"
[1126] "1126" "1127" "1128" "1129" "1130" "1131" "1132" "1133" "1134"
[1135] "1135" "1136" "1137" "1138" "1139" "1140" "1141" "1142" "1143"
[1144] "1144" "1145" "1146" "1147" "1148" "1149" "1150" "1151" "1152"
[1153] "1153" "1154" "1155" "1156" "1157" "1158" "1159" "1160" "1161"
[1162] "1162" "1163" "1164" "1165" "1166" "1167" "1168" "1169" "1170"
[1171] "1171" "1172" "1173" "1174" "1175" "1176" "1177" "1178" "1179"
[1180] "1180" "1181" "1182" "1183" "1184" "1185" "1186" "1187" "1188"
[1189] "1189" "1190" "1191" "1192" "1193" "1194" "1195" "1196" "1197"
[1198] "1198" "1199" "1200" "1201" "1202" "1203" "1204" "1205" "1206"
[1207] "1207" "1208" "1209" "1210" "1211" "1212" "1213" "1214" "1215"
[1216] "1216" "1217" "1218" "1219" "1220" "1221" "1222" "1223" "1224"
[1225] "1225" "1226" "1227" "1228" "1229" "1230" "1231" "1232" "1233"
```

```
[1234] "1234" "1235" "1236" "1237" "1238" "1239" "1240" "1241" "1242"
[1243] "1243" "1244" "1245" "1246" "1247" "1248" "1249" "1250" "1251"
[1252] "1252" "1253" "1254" "1255" "1256" "1257" "1258" "1259" "1260"
[1261] "1261" "1262" "1263" "1264" "1265" "1266" "1267" "1268" "1269"
[1270] "1270" "1271" "1272" "1273" "1274" "1275" "1276" "1277" "1278"
[1279] "1279" "1280" "1281" "1282" "1283" "1284" "1285" "1286" "1287"
[1288] "1288" "1289" "1290" "1291" "1292" "1293" "1294" "1295" "1296"
[1297] "1297" "1298" "1299" "1300" "1301" "1302" "1303" "1304" "1305"
[1306] "1306" "1307" "1308" "1309" "1310" "1311" "1312" "1313" "1314"
[1315] "1315" "1316" "1317" "1318" "1319" "1320" "1321" "1322" "1323"
[1324] "1324" "1325" "1326" "1327" "1328" "1329" "1330" "1331" "1332"
[1333] "1333" "1334" "1335" "1336" "1337" "1338" "1339" "1340" "1341"
[1342] "1342" "1343" "1344" "1345" "1346" "1347" "1348" "1349" "1350"
[1351] "1351" "1352" "1353" "1354" "1355" "1356" "1357" "1358" "1359"
[1360] "1360" "1361" "1362" "1363" "1364" "1365" "1366" "1367" "1368"
[1369] "1369" "1370" "1371" "1372" "1373" "1374" "1375" "1376" "1377"
[1378] "1378" "1379" "1380" "1381" "1382" "1383" "1384" "1385" "1386"
[1387] "1387" "1388" "1389" "1390" "1391" "1392"

[[2]]
[1] "ANN"  "CTA"  "GAM"  "GBM"  "GLM"  "MARS" "FDA"  "RF"   "SRE"

[[3]]
[1] "total.data" "rep1"       "rep2"       "rep3"

[[4]]
[1] "PA1" "PA2"


> #you can avoid having the rownames to be printed in the console as they
> #are generally not very usefull
> dimnames(Pred_Sp281)[-1]


[[1]]
[1] "ANN"  "CTA"  "GAM"  "GBM"  "GLM"  "MARS" "FDA"  "RF"   "SRE"

[[2]]
[1] "total.data" "rep1"       "rep2"       "rep3"

[[3]]
[1] "PA1" "PA2"
```

For instance, we examine the probability of occurrence of the first species, modelled with GBM. Here we just display 20 rows (or sites) in the middle.

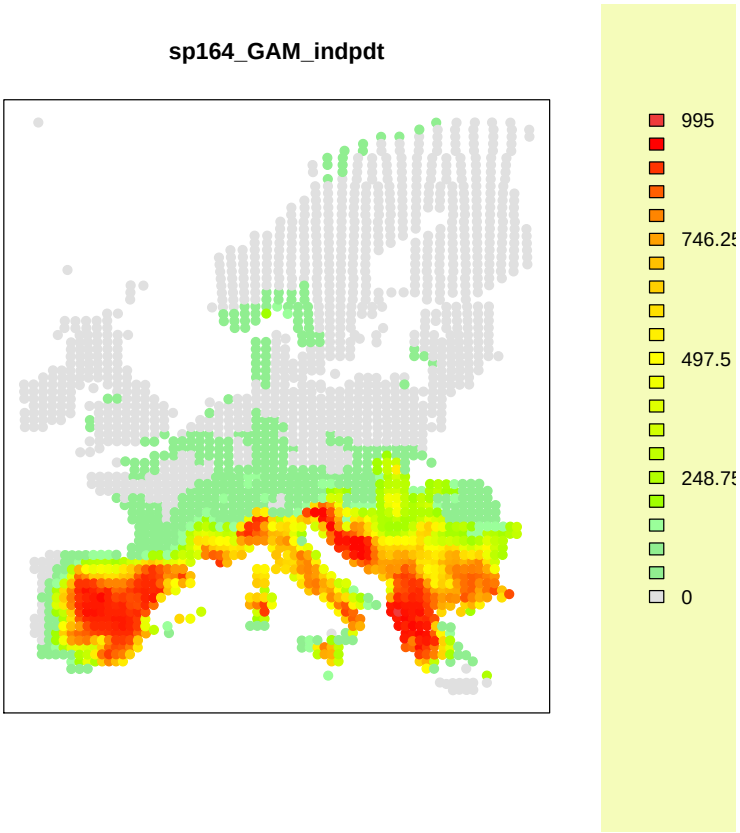
```
> #if you don't inform the 3rd and 4th dimension (you still need commas), you will have all of them
> #at once in a matrix.
> load("pred/Pred_Sp281")
> Pred_Sp281[281:300,"GBM",,]


, , PA1

    total.data rep1 rep2 rep3
281        540  542  535  538
282        537  535  534  530
283        542  543  540  543
284        541  542  538  538
285        541  540  537  537
286        526  521  528  529
287        544  545  545  545
288        542  543  543  545
289        544  545  545  545
290        505  511  501  510
291        534  536  534  524
292        527  529  521  522
293        521  521  504  513
294        541  542  543  541
```

```
295         544  543  544  543
296         544  543  544  544
297         533  536  534  515
298         515  512  525  509
299         538  539  534  540
300         539  539  535  534

, , PA2

    total.data rep1 rep2 rep3
281         509  514  507  511
282         534  524  525  535
283         530  523  515  533
284         526  518  511  517
285         543  540  541  543
286         544  543  544  544
287         544  544  544  544
288         539  523  533  539
289         516  519  491  495
290         541  535  534  537
291         541  535  535  537
292         540  535  534  537
293         541  536  535  539
294         541  535  534  537
295         517  517  494  510
296         542  534  536  538
297         540  533  535  537
298         531  528  505  523
299         539  535  523  534
300         528  525  501  519
```

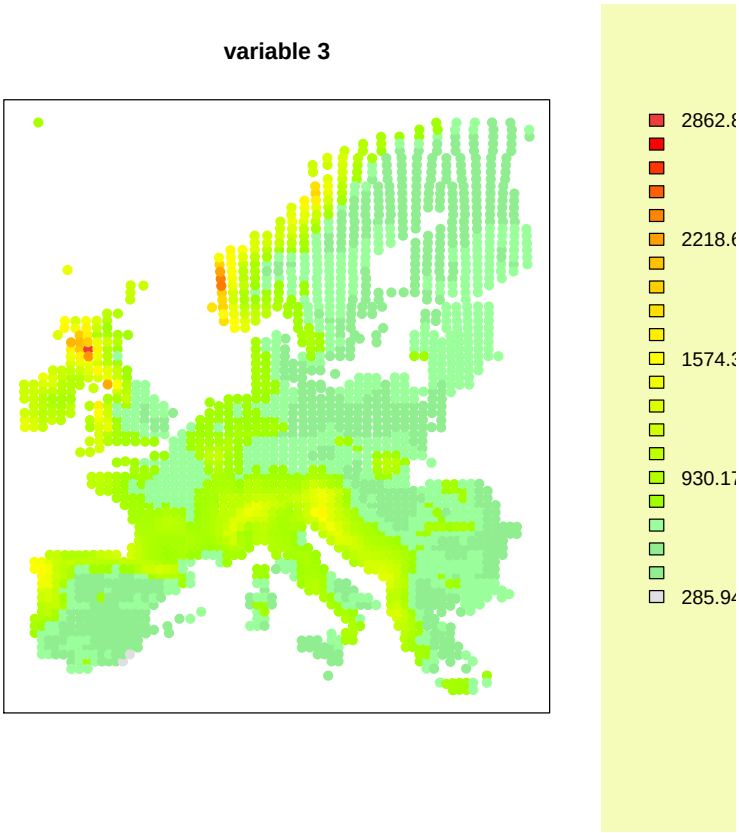
Note that because there is a random selection of the data for calibration, you will end up with slightly different values on these example runs.

To plot the predictions, use the *level.plot* function. It requires two inputs : the vector of values that you want to plot and the coordinates of your data points. This function works with any type of data. Because we have chosen to run the models with pseudo-absence data, plotting the partial predictions is not very convinient. We will plot instead the values of the fake independant data (which is just the full original dataset) for the GAM, and the values of one of the variables used to calibrate the models.

```
> load("pred/Pred_Sp277_indpdt")
> level.plot(Pred_Sp277_indpdt[,"GAM",1,1], CoorXY, title='sp277_GAM_indpdt')
```




```
> #and the level plot for the third variable used
> level.plot(Sp.Env[,6], CoorXY, title='variable 3')
```



Note that the independent predictions are only made on the final 100% model and not on the repetitions. To check it :

```
> Pred_Sp277_indpdt[1:10,,,]


, , total.data

   ANN CTA GAM GBM GLM MARS FDA RF SRE
1  896   0   2 456   0    1  38  4   0
2  896   0  34 456   0   18  38  1   0
3  896   0  22 456   0    8  38  2   0
4  896   0   0 456   0    0  37  0   0
5  896   0   0 456   0    0  37  0   0
6  896   0   1 456   0    0  37  0   0
7  896   0   0 456   0    0  37  0   0
8  896   0   1 456   0    0  37  6   0
9  896   0   0 456   0    0  37  0   0
10 896   0   4 456   0    1  38  2   0

, , rep1

   ANN CTA GAM GBM GLM MARS FDA RF SRE
1   NA  NA  NA  NA  NA   NA  NA NA  NA
2   NA  NA  NA  NA  NA   NA  NA NA  NA
3   NA  NA  NA  NA  NA   NA  NA NA  NA
4   NA  NA  NA  NA  NA   NA  NA NA  NA
5   NA  NA  NA  NA  NA   NA  NA NA  NA
6   NA  NA  NA  NA  NA   NA  NA NA  NA
7   NA  NA  NA  NA  NA   NA  NA NA  NA
8   NA  NA  NA  NA  NA   NA  NA NA  NA
9   NA  NA  NA  NA  NA   NA  NA NA  NA
```

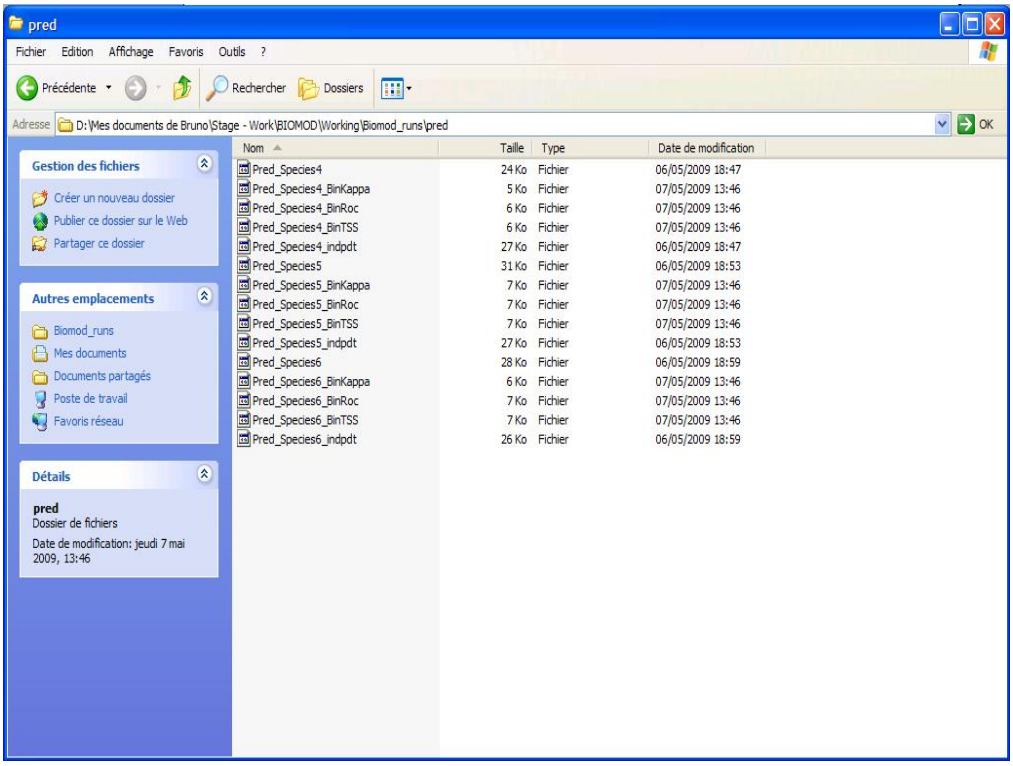
```
10  NA  NA  NA  NA  NA   NA  NA NA  NA

, , rep2

   ANN CTA GAM GBM GLM MARS FDA RF SRE
1   NA  NA  NA  NA  NA   NA  NA NA  NA
2   NA  NA  NA  NA  NA   NA  NA NA  NA
3   NA  NA  NA  NA  NA   NA  NA NA  NA
4   NA  NA  NA  NA  NA   NA  NA NA  NA
5   NA  NA  NA  NA  NA   NA  NA NA  NA
6   NA  NA  NA  NA  NA   NA  NA NA  NA
7   NA  NA  NA  NA  NA   NA  NA NA  NA
8   NA  NA  NA  NA  NA   NA  NA NA  NA
9   NA  NA  NA  NA  NA   NA  NA NA  NA
10  NA  NA  NA  NA  NA   NA  NA NA  NA

, , rep3

   ANN CTA GAM GBM GLM MARS FDA RF SRE
1   NA  NA  NA  NA  NA   NA  NA NA  NA
2   NA  NA  NA  NA  NA   NA  NA NA  NA
3   NA  NA  NA  NA  NA   NA  NA NA  NA
4   NA  NA  NA  NA  NA   NA  NA NA  NA
5   NA  NA  NA  NA  NA   NA  NA NA  NA
6   NA  NA  NA  NA  NA   NA  NA NA  NA
7   NA  NA  NA  NA  NA   NA  NA NA  NA
8   NA  NA  NA  NA  NA   NA  NA NA  NA
9   NA  NA  NA  NA  NA   NA  NA NA  NA
10  NA  NA  NA  NA  NA   NA  NA NA  NA
```

Transforming the predictions on the original dataset

It might be useful to extract the presence/absence predictions. To do so, use the *CurrentPred()* function by switching *BinRoc, BinKappa* and/or *BinTSS* to TRUE and each probability of occurence will be transformed into presence and absence using the cutoff maximising the models accuracy according to Roc, Kappa or TSS.

```
> CurrentPred(GLM=T, GBM=T, GAM=T, CTA=T, ANN=T, SRE=T, FDA=T, MARS=F, RF=T,
 BinRoc=T, BinKappa=T, BinTSS=T, FiltKappa=T)
```

New objects are created for each species containing the predictions in binary and or filtered format using the thresholds produced by the evaluation technics : Pred_Sp277_BinRoc, Pred_Sp277_BinKappa, Pred_Sp277_BinTSS, Pred_Sp277_FiltKappa, and so on.




```
> load("pred/Pred_Sp277")
> load("pred/Pred_Sp277_BinKappa")
> load("pred/Pred_Sp277_FiltKappa")
> #
> #
> Pred_Sp277[260:270,,1,1]
```

```
    ANN CTA GAM GBM GLM MARS FDA   RF  SRE
260 895 979 990 539 999  998 981 1000 1000
261 894 979 981 539 999  995 981 1000 1000
262 895 979 985 540 999  997 981 1000 1000
263 893 996 976 544 999  991 981 1000 1000
264 884 996 981 545 999  990 981 1000 1000
265 886 996 994 545 999  998 981 1000 1000
266  39  39   0 459   0    0  37    0    0
267  39  39   0 459   0    0  38    0    0
268  38  39   0 459   8    0  37   24    0
269  38  39   0 459   0    0  37    0    0
270  45  39   0 459   0    0  37    0    0
```

```
> Pred_Sp277_BinKappa[260:270,,1,1]
```

```
    ANN CTA GAM GBM GLM MARS FDA RF SRE
260   1   1   1   1   1   NA   1  1   1
261   1   1   1   1   1   NA   1  1   1
262   1   1   1   1   1   NA   1  1   1
263   1   1   1   1   1   NA   1  1   1
264   1   1   1   1   1   NA   1  1   1
265   1   1   1   1   1   NA   1  1   1
266   0   0   0   0   0   NA   0  0   0
267   0   0   0   0   0   NA   0  0   0
268   0   0   0   0   0   NA   0  0   0
269   0   0   0   0   0   NA   0  0   0
270   0   0   0   0   0   NA   0  0   0
```

```
> Pred_Sp277_FiltKappa[260:270,,1,1]
```

```
    ANN CTA GAM GBM GLM MARS FDA   RF  SRE
260 895 979 990 539 999   NA 981 1000 1000
261 894 979 981 539 999   NA 981 1000 1000
262 895 979 985 540 999   NA 981 1000 1000
263 893 996 976 544 999   NA 981 1000 1000
264 884 996 981 545 999   NA 981 1000 1000
265 886 996 994 545 999   NA 981 1000 1000
266   0   0   0   0   0   NA   0    0    0
267   0   0   0   0   0   NA   0    0    0
268   0   0   0   0   0   NA   0    0    0
269   0   0   0   0   0   NA   0    0    0
270   0   0   0   0   0   NA   0    0    0
```

Identifying the best model

In our example, we could compare all the models we run for the different species using the three different evaluation methods available. The function *PredictionBestModel* also transforms the probabilities into the presence/absence and filtered formats.

```
> PredictionBestModel(GLM=T,GBM=T, GAM=T, CTA=T, ANN=T, FDA=T, MARS=F, RF=T, SRE=T,
  method='all', Bin.trans = T, Filt.trans = T)
```

Multimodel comparison according to the TSS statistic:

```
> load("pred/BestModelByTSS")
> BestModelByTSS
```

```
$Sp281
         Best.Model Cross.validation indepdt.data total.score Cutoff
PA1              RF            0.942        0.875      1.0000    340
PA1_rep1         RF            0.935         none      0.9869    330
PA1_rep2         RF            0.972         none      0.9924    300
PA1_rep3         RF            0.919         none      0.9839    330
PA2              RF            0.944        0.863      1.0000    320
PA2_rep1         RF            0.959         none      0.9919    410
PA2_rep2         RF            0.925         none      0.9818    350
PA2_rep3         RF            0.947         none      0.9894    350
         Sensitivity Specificity
PA1           100.00       100.0
PA1_rep1       99.49        99.2
PA1_rep2       99.74        99.5
PA1_rep3       99.49        98.9
PA2           100.00       100.0
PA2_rep1       99.49        99.7
PA2_rep2       98.98        99.2
PA2_rep3       99.74        99.2

$Sp277
         Best.Model Cross.validation indepdt.data total.score Cutoff
PA1              RF            0.974        0.787      1.0000    360
PA1_rep1         RF            0.960         none      0.9921    480
PA1_rep2         RF            0.988         none      0.9977    510
PA1_rep3         RF            0.972         none      0.9944    470
         Sensitivity Specificity
PA1           100.00      100.00
PA1_rep1       99.63       99.58
PA1_rep2       99.91       99.86
PA1_rep3       99.72       99.72
```

The RF comes out first each time, let's switch it off : Multimodel comparison according to the TSS statistic:

```
> PredictionBestModel(GLM=T,GBM=T, GAM=T, CTA=T, ANN=T, FDA=T, MARS=F, RF=F, SRE=T,
  method='all', Bin.trans = T, Filt.trans = T)
> load("pred/BestModelByTSS")
> BestModelByTSS


$Sp281
         Best.Model Cross.validation indepdt.data total.score Cutoff
PA1             CTA            0.867         0.79      0.9614  209.6
PA1_rep1        CTA            0.899         none      0.9411  246.7
PA1_rep2        CTA            0.888         none      0.9384  250.0
PA1_rep3        ANN            0.934         none      0.9537  134.2
PA2             CTA            0.902        0.794      0.9589  190.0
PA2_rep1        CTA            0.919         none      0.9422  340.0
PA2_rep2        FDA            0.929         none      0.9302  180.2
PA2_rep3        ANN            0.944         none      0.9530  261.4
         Sensitivity Specificity
PA1            99.74        96.4
PA1_rep1       98.21        95.9
PA1_rep2       96.94        96.9
PA1_rep3       98.47        96.9
PA2            99.49        96.4
PA2_rep1       98.72        95.5
PA2_rep2       95.92        97.1
PA2_rep3       97.70        97.6

$Sp277
         Best.Model Cross.validation indepdt.data total.score Cutoff
PA1             CTA            0.942        0.734      0.9693    310
PA1_rep1        CTA            0.937         none      0.9665    310
PA1_rep2        CTA            0.942         none      0.9633    340
PA1_rep3        CTA            0.947         none      0.9629    310
         Sensitivity Specificity
PA1            98.06       98.87
PA1_rep1       98.33       98.31
PA1_rep2       97.31       99.02
PA1_rep3       97.13       99.16
```

Multimodel comparison according to the ROC:

```
> load("pred/BestModelByRoc")
> BestModelByRoc


$Sp281
         Best.Model Cross.validation indepdt.data total.score  Cutoff
PA1             CTA            0.962        0.931       0.995 835.326
PA1_rep1        ANN            0.993         none       0.994 218.528
PA1_rep2        ANN            0.988         none       0.996  71.285
PA1_rep3        ANN            0.995         none       0.997  173.62
PA2             CTA            0.971        0.931       0.995     848
PA2_rep1        ANN            0.993         none       0.995  164.24
PA2_rep2        ANN            0.987         none       0.994 141.356
PA2_rep3        ANN             0.99         none       0.993  285.16
         Sensitivity Specificity
PA1           96.939        97.1
PA1_rep1      95.663        95.8
PA1_rep2      96.684        96.9
PA1_rep3      97.449        97.4
PA2           96.684        97.1
PA2_rep1      95.408        95.4
PA2_rep2      95.918        95.9
PA2_rep3      97.704        97.6

$Sp277
         Best.Model Cross.validation indepdt.data total.score  Cutoff
PA1             CTA            0.994         0.89       0.997 437.562
PA1_rep1        GLM             0.99         none       0.996 454.545
PA1_rep2        GLM            0.996         none       0.997 423.576
PA1_rep3        GLM            0.997         none       0.997 393.606
         Sensitivity Specificity
```

```
PA1           96.667      96.624
PA1_rep1      96.944      96.906
PA1_rep2      96.667      96.624
PA1_rep3      96.667      96.624
```

.

Multimodel predictions according to the Kappa statistic

```
> load("pred/PredBestModelByKappa")
> PredBestModelByKappa[740:750,,1]
```

```
    PA1 PA1_rep1 PA1_rep2 PA1_rep3 PA2 PA2_rep1 PA2_rep2 PA2_rep3
740   0        0        7        6   0        0        7        4
741   0        0        7        6   0        0        7        4
742   0        0        7        6   0        0        7        4
743   0        0       15        6   0        0        6        4
744   0        0        7        6   0        0        7        4
745   0        0       14        6   0        0        6        4
746   0        0       15        6   0        0        6        4
747   0        0       13        6   0        0        8        5
748   0        0        7        6   0        0        9        4
749   0        0       10        7  52        0        9        4
750   0        0        7        6   0        0        9        4
```

0.5 Save our work session

```
> save.image("Practical_1_finished.RData")
```

Do keep in mind that some information is kept in the file that has just been generated but that a lot our work is also stored in the directories that have been created by BIOMOD. Both will be needed for carrying on the next steps.