ModOperating Manual for BIOMOD

Wilfried Thuiller
Bruno Lafourcade, Miguel Araujo

June 11, 2008

Contents

0.1 Introduction

BIOMOD is an acronym for BIOdiversity MODelling. BIOMOD has been originally developed at the Centre d'Ecologie Fonctionnelle et Evolutive of the CNRS in Montpellier (France) and was partly funded by the FP5 ATEAM European Project. The package was developed for species distribution modelling but it can be used for modelling any kinds of distributions. The only restriction is that the dependent variable should be coded in a presence-absence binary format. BIOMOD is a platform for ensemble forecasting of species distributions, enabling the explicit treatment of model uncertainties and the examination of species-environment relationships. BIOMOD includes the ability to model species distributions with several techniques, test models with a wide range of approaches, project species distributions into the future using different climate scenarios and dispersal functions, assess species temporal turnover, plot species response curves, and test the strength of species interactions with predictor variables. Computationally, BIOMOD a collection of functions running within the R (CRAN) software (programmed in R language) and allows the user to apply a range of statistical models to several dependent variables using a set of independent variables.

0.2 Installation

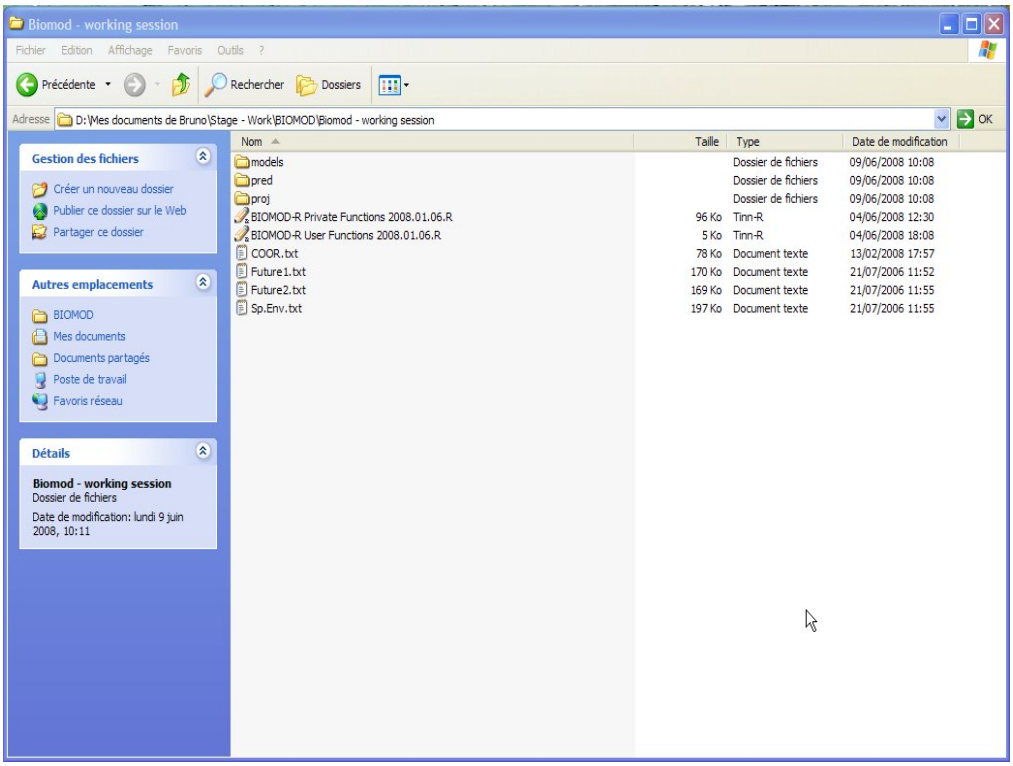
To run BIOMOD, please use the latest version of R. A large number of libraries are also required: rpart, MASS, gbm, gam, nnet, mgcv, mda, randomForest, Design, Hmisc, reshape) before attempting to run BIOMOD. Two scripts are given to the user:
- BIOMOD-R Private Functions 200X.XX.XX (the date depending on the actual update).
- BIOMOD-R User Functions 200X.XX.XX

BIOMOD-R Private Functions 200X.XX.XX contains all the functions BIOMOD needs to run. A new user does not need to open it, while more experienced users can open it and modify some functions or internal parameters if they want to (at their own risks!) BIOMOD-R User Functions aims to help the user to run BIOMOD in optimal conditions. This script presents pre-formatted calls to prepare the datasets, initialize BIOMOD, and run the different models. This is the script recommended to use all the time.

The recommended procedure is to first create a folder called BIOMOD and then paste the two scripts inside this folder. The suggestion is not to modify or move these scripts. They will be the backup. Then, create a new folder where to store the datasets, run the models and save the outputs and results. This is called the "workspace". In this workspace, paste also the two scripts (BIOMOD-R Private Functions 200X.XX.XX, BIOMOD-R User Functions 200X.XX.XX).

It is from this folder that the files will be read and written. In the latter version of BIOMOD, the results will be stored outside R's workspace to counter the memory storage limitations of the software. For this reason you should create 3 folders named *models, pred* and *proj*. As you might have guessed, they will respectively contain the models, the current predictions and the projections produced by BIOMOD. If one of these folders is missing, trying to run the *Models* function will inevitably end up in an error message and you won't be able to go any further.

Your folder should look like this and contain the appropriate objects (do mind that the R files you will use will have a different name).



You also need to give the root on your computer of this working folder. Below are two examples of it. Notice that the orientation of the slash is of the utmost importance. If you want to do them the other way round (as it seems easier to copy the root from the folder's header) then you have to double them like in the second example.

```
> root <- "D:/My documents/BIOMOD/Working session"
> root <- "D:\\My documents\\BIOMOD\\Working session"
```

0.3 Getting Started

0.3.1 Data preparation

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. Once the tutorial is completed the user should be able to run BIOMOD with his own data.
Open the script: BIOMOD-R User Functions 200X.XX.XX

The first thing to do is to compile the different functions required to run BIOMOD.

```
> source("BIOMOD-R Private Functions 2008.06.01.R")
```

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

```
> save.image()
```

Now, BIOMOD is ready to run and the user can import the species and the environmental data. For practical reasons, we can store them in the same file and load the species/environment dataset provided with BIOMOD. The text is already pre-formatted in the script BIOMOD-R User Functions 200X.XX.XX

```
> Sp.Env<-read.table("Sp.Env.txt", h=T, sep="\t")
> CoorXY <- read.table("COOR.txt", h=T, sep='\t')[-(1770:1794),]
> Sp.Env[1:5,]
```

```
  Id   Var1 Var2  Var3  Var4  Var5  Var6  Var7 Species1 Species2
1  1 0.6683 4296 770.1 39.33 295.1 16.74 10.87        0        0
2  2 0.7596 4174 928.1 57.32 348.7 16.41 10.51        0        0
3  3 0.7424 4173 870.3 50.05 330.0 16.41 10.50        0        0
4  4 0.5543 4264 620.0 24.99 239.1 16.66 10.93        0        0
5  5 0.5489 4169 622.3 25.16 241.0 16.40 11.28        0        0
  Species3 Species4 Species5 Species6 Species7
1        0        0        1        0        1
2        0        0        0        0        1
3        0        0        0        0        0
4        0        0        1        0        0
5        0        0        1        0        0
```

- Id: An Id to keep track of the row numbers
- Var1 to Var7: Environmental variables (bioclimatic in that case)
- Species1 to Species7: Presence/absence of 7 species.

Type *?read.table* for more details and other possible extensions.
Here the separator was a tabulation but it could be a comma. Your file might also be in a csv format; in that case you should use the *read.csv* function to correctly load your data.
BIOMOD does not recognise geographical coordinates nor does it order the data according to these. The user should ensure that all datasets are kept in the same order.

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

0.3.2 Initialisation

First, we need to set up the dataset in a correct format for BIOMOD by means of the *Initial.State* function.
Reminder of the syntax - rows are specified before the comma and columns after with a semi-colon separating the start and end column.

The syntax in the *Initial.State* function is the following:
Response: The response variables to model. In our example, the species occurrences are located from the column 9 to 15.
Explanatory: The explanatory or independent variables. In our example, the environmental variables, called Var1 to Var6 are located in the columns 2 to 8.
IndependentResponse: Truly independent response variables. This is used to evaluate the predictive accuracy of the models. If no truly independent data are available, add nothing
IndependentExplanatory: Truly independent explanatory variables. This should be used to evaluate the predictive accuracy of the models. If no truly independent data are available, add nothing

Ideally, one should always evaluate the predictive performance of a species distribution model using independent data. If the data are available, BIOMOD should calibrate the models on the calibration

data and evaluate them using the independent data. If no independent data for model evaluation exist, two alternatives are available. First, we can use a random data splitting procedure into, e.g., 70/30 % as commonly used (Araújo, et al. 2005b, Guisan and Thuiller 2005), where the models are calibrated on a random subset of 70 % of the data and evaluated on the remaining 30 %. Secondly, a multiple cross-validation approach is available, where BIOMOD replicates the data splitting procedure N times, run the models, record the predictive performance and then provide the mean of the cross-validation. It gives a more robust estimate of the predictive performance of each selected model and it also provides an assessment of the sensitivity of the model to the initial conditions, i.e., to the species distribution data. Of course, it takes longer to run on basic personal computers.

In our example, we do not have truly independent data.

```
> Initial.State(Response = Sp.Env[,c(10,11)], Explanatory = Sp.Env[,2:8],
 IndependentResponse = NULL, IndependentExplanatory = NULL)
> ls()
```

```
 [1] "BinaryTransformation"     "Biomod.Models"
 [3] "Biomod.RangeSize"         "Biomod.Turnover"
 [5] "CoorXY"                   "create"
 [7] "create.projection.matrix" "CurrentPred"
 [9] "CutOff.Optimised"         "CVnnet"
[11] "CVnnet1"                  "DataBIOMOD"
[13] "Dispersal.Limit"          "Ensemble.Forecasting"
[15] "FilteringTransformation"  "functionkeep"
[17] "Initial.State"            "KappaRepet"
[19] "KappaSRE"                 "KappaStat"
[21] "Migration"                "Models"
[23] "NbSpecies"                "NbVar"
[25] "plot.level"               "plot.map"
[27] "plot.response"            "PredictionBestModel"
[29] "printcp"                  "ProbDensFunc"
[31] "Projection"               "ProjectionBestModel"
[33] "pseudo.abs"               "Rescaler2"
[35] "root"                     "SampleMat2"
[37] "scope"                    "scopeExpSyst"
[39] "scopeGAM"                 "Sp.Env"
[41] "species.names"            "sre"
[43] "testnull"                 "TSS.Stat"
```

It creates one or several databases: DataBIOMOD and DataEvalBIOMOD, if you have independent data. The latter will be used during the testing of the models. Make sure to always keep these datasets unchanged and never delete them.

```
> DataBIOMOD[1:5,]
```

```
    Var1 Var2  Var3  Var4  Var5  Var6  Var7 Species2 Species3
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
```

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 of the models.

0.4 Models

0.4.1 Short description

The function "Models" runs the different models implemented in BIOMOD, as well as their evaluation using three different techniques (kappa statistic, True Skill Statistics and ROC curve).

Nine different models are currently implemented:
- Generalised Linear Models (GLM)
- Generalised Additive Models (GAM)
- Classification Tree Analysis (CTA)
- Artificial Neural Networks (ANN)
- Surface Range Envelope (SRE)
- Generalised Boosting Model (GBM)
- Breiman and Cutler's random forest for classification and regression (randomForest)
- Mixture Discriminant Analysis (MDA)
- Multiple Adaptive Regression Splines (MARS)

The selection of each model is made by typing T (TRUE) or F (FALSE). There are also various parameters that needs setting up for some of the models. See below for the explanation.
All the selected models (= T) will run for each species and automatically on the calibration dataset. Below you can find a short explanation of each model. Note that they are not explained in the order they appear in the Models function. A more extensive description of the models can be found at the end of the Manual (see section "Models' description").

- *GLM = T, TypeGLM = "poly", Test = "BIC"*: Run a stepwise GLM (TRUE), using linear ("simple"), quadratic ("quad") or polynomial ("poly") terms. The stepwise procedure either uses the AIC or BIC criteria.

- *GBM = T, No.trees = 3000, CV.gbm = 5*: Run a generalised boosting model (GBM) (= boosted regression trees). The maximum number of trees can be user defined (default=3000). A cross-validation procedure to select the optimal number of trees is implemented. The defaul number of cross-validation is 5.

- *GAM = T, Spline = 4*: Run a generalised additive model (GAM) with a spline function with a degree of smoothing of 4 (similar to a polynomial of degree 3).

- *CTA = T, CV.tree = 50*: Run a classification tree analysis (CTA). The optimal length of the tree is estimated using cross-validation (default=50).

- *ANN = T, CV.ann = 2*: Run an artificial neural network (ANN). As different runs can provide different results, the best amount of weight decay and the number of units in the hidden layer is selected by using N-fold cross-validation (3 by default). The user can also select the number of cross-validations.

- *SRE = T, Perc025=T, Perc05=F*: Run an rectilinear surface range envelop (=BIOCLIM) using the percentile 0.025 or 0.05 as recommended by Nix or Busby.

- *MDA = T*: Run a mixture discriminant analysis using the MARS function for the regression part of the model.

- *MARS = T*: Run a multivariate adaptive regression spline.

- *RF = T*: Run a random forest model.

- *NbRunEval*: a new way of calibrating / evaluating the models: To be reliable, predictions must be validated using independent data. As this information is often unavailable, an alternative is to par-

tition randomly the data into a calibration (or training) and an evaluation (or testing) dataset. The calibration data is used during the fitting or learning process of the models, while the evaluation data ae used to estimate the prediction ability of the model. Within BIOMOD, this evaluation step can be done according to three different techniques (Kappa, ROC, TSS). However, this classic splitting method can add variability in the predictions when several runs are made: because the splitting of the data is made at random, each run will have its own calibration and evaluation datasets resulting in inevitable differences during the calibration of models and the subsequent predictions. To address this problem, BIOMOD allows evaluation of model performance on different data split runs and then allows using 100 % of the data to make a final calibration of the models for prediction. In this case, the evaluation is more reliable and the predictions are not influenced by the random splitting of the data. It does, however, require more time to go through all the evaluation runs than when using a single run of the classic but necessarily biased splitting procedure. An extra evaluation of the final model can also be done if independent data are available.

- *DataSplit*: the ratio used for splitting the original database in calibration and evaluation subsets during the evaluation procedure mentioned above. *Note that the function ensures that prevalence (ratio between the total number of presences and the total number of points) is conserved in the calibration and evaluation datasets*

- *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 in R.

- $ROC = T$: Evaluate the models using the Area Under the ROC (receiver operating characteristic curve) Curve (AUC)

- $Optimized.Threshold.ROC = T$: ROC is a threshold independent method. However, if the user wants to find the optimal threshold optimising the percentage of presence and absence correctly predicted, this threshold can be used to transform the probabilities of occurrence from models into presence and absence.

- $Kappa = T$: Evaluate the models using the Cohen's Kappa statistic. The treshold optimising the Kappa is kept.

- $TSS = T$: Evaluate the models using the True Skill Statistic (TSS). The treshold optimising the TSS is kept.

- *KeepindependentPred*: If TRUE (and the truly independent data has been provided), theen the prediction on the independent data will be saved. If FALSE, only the predictive accuracy on the independent data are conserved (obtained by ROC, TSS and/or Kappa).

- *VarImport*: if True, this parameter enables a direct comparison of the explanatory variable importance across models. Once the models are trained (i.e. calibrated), a standard prediction is made. Then, one of the variables is randomized and a new prediction is made. The correlation score between that new prediction and the standard prediction is calculated and is considered to give an estimation of the variable importance in the model : if there is a good correlation score, i.e., the predictions do only slightly differ across the two methods, then the randomized variable does not influence the model in its prediction. This step is repeated n times for each variable independently.

NOTE : when the *VarImportance* function is run, the output is giving 1 minus the mean correlation for each variable. Therefore, a high score means a high importance. The results can also be given as a relative importance, i.e., the values are no longer related to the correlation scores but give a ranking of the variable importance (which sums up to one).

0.4.2 Running the models

We can now run the different models on our species. It takes a few minutes. Note that one can run one species at a time with all the models being put to true. 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

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).

```
> 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, Perc025=T, Perc05=F, MDA = T,
    MARS = T, RF = T, NbRunEval = 0, DataSplit = 80, Yweights=NULL, ROC = T, Optimized.Threshold.ROC = T,
    Kappa = T, TSS=T, KeepPredIndependent = F, VarImport=5)


 Warning : The models will be evaluated on the calibration data only
         it could lead to over-optimistic predictive performances.

#####                          Species2                          #####
Model=Artificial Neural Network
         2 Fold Cross Validation + 3 Repetitions
Calibration and evaluation phase: Nř of cross-validations:  1
Evaluating Predictor Contributions in  ANN ...
Model=Classification tree
         50 Fold Cross-Validation
Evaluating Predictor Contributions in  CTA ...
Model=GAM spline
         lambda 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  MDA ...
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 ...
#####                          Species3                          #####
Model=Artificial Neural Network
         2 Fold Cross Validation + 3 Repetitions
Calibration and evaluation phase: Nř of cross-validations:  1
Evaluating Predictor Contributions in  ANN ...
Model=Classification tree
         50 Fold Cross-Validation
Evaluating Predictor Contributions in  CTA ...
Model=GAM spline
         lambda 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  MDA ...
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 ...
```

0.4.3 Analysing the outputs

There are now various objects stored in the workspace:

```
> ls()
```

```
 [1] "algo"                      "algo.choice"
 [3] "BinaryTransformation"      "Biomod.Models"
 [5] "Biomod.RangeSize"          "Biomod.Turnover"
 [7] "CoorXY"                    "create"
 [9] "create.projection.matrix" "CurrentPred"
[11] "CutOff.Optimised"          "CVnnet"
[13] "CVnnet1"                   "DataBIOMOD"
[15] "Dispersal.Limit"           "Ensemble.Forecasting"
[17] "Evaluation.results.Kappa" "Evaluation.results.Roc"
[19] "Evaluation.results.TSS"   "FilteringTransformation"
[21] "functionkeep"              "g.pred"
[23] "Initial.State"             "KappaRepet"
[25] "KappaSRE"                  "KappaStat"
[27] "Migration"                 "Models"
[29] "Models.information"        "NbSpecies"
[31] "NbVar"                     "plot.level"
[33] "plot.map"                  "plot.response"
[35] "PredictionBestModel"       "printcp"
[37] "ProbDensFunc"              "Projection"
[39] "ProjectionBestModel"       "pseudo.abs"
[41] "Rescaler2"                 "root"
[43] "SampleMat2"                "scope"
[45] "scopeExpSyst"              "scopeGAM"
[47] "Sp.Env"                    "species.names"
[49] "sre"                       "testnull"
[51] "TSS.Stat"                  "VarImportance"
```

Some objects are functions, some are usefull values to keep in memory and some are dataframes. We can easily recognise the datasets (Sp.Env, DataBIOMOD).

The models themselves are now stored out of the R workspace directly on the computers' hard disk. They are named after the model and the species' names, i.e. species61_ MDA for example.

Each model (excepted SRE) generates an object storing the different parameterisation, the importance of each variable (for GBM, GAM, randomForest), and the ANOVA for variable significance (GLM, GAM), and so on. This output is essential as it allows generating predictions, to know which variable has been selected and so on.

Back loading the models is very straightforward : simply use the *load* function to have the model restored in the R workspace, with the same name, and directly usable. This is also the case with the other outputs stored outside of R (predictions and projections). Let's see an example with the GLM for the first species modelled (the syntax is not very straight forward but it will become natural after a while):

```
> load(paste(root, "/models/Species3_GLM", sep=""))
> Species3_GLM


Call:  glm(formula = Species3 ~ poly(Var6, 3) + poly(Var4, 3) + poly(Var1,      2) + poly(Var7, 3) + poly(Var2,

Coefficients:
   (Intercept)  poly(Var6, 3)1  poly(Var6, 3)2  poly(Var6, 3)3
     -4.37e+00       -7.57e+02        4.00e+02        2.46e+02
poly(Var4, 3)1  poly(Var4, 3)2  poly(Var4, 3)3  poly(Var1, 2)1
      2.03e+01       -9.13e+00       -1.49e+01       -3.15e+01
poly(Var1, 2)2  poly(Var7, 3)1  poly(Var7, 3)2  poly(Var7, 3)3
     -4.05e+01        1.67e+02       -5.14e+01       -7.66e+01
poly(Var2, 3)1  poly(Var2, 3)2  poly(Var2, 3)3  poly(Var5, 2)1
      3.20e+02       -4.55e+02        4.12e+01       -9.10e+01
poly(Var5, 2)2            Var3
      1.77e+01        7.16e-03

Degrees of Freedom: 2263 Total (i.e. Null);  2246 Residual
Null Deviance:          3140
Residual Deviance: 1320         AIC: 1360
```

It shows 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 . A description of the outputs of each model is provided below (cf. OUPUTS and INTERPRETATION).

Different objects are produced by the *Models* function and are thus present in the workspace in addition to the models store on the hard disk.

These are :

- Evaluation.results.Roc (if Roc selected)
- Evaluation.results.Kappa (if Kappa selected)
- Evaluation.results.TSS (if TSS selected)
- VarImportance
- Models.information.

The 3 first ones contain the scores of the evaluation procedure and the cutoffs for each model and for each species. *VarImportance* is rather explicit and contains the results of the variables' contribution analysis. *Models.information* is of little interest for the user, but this object contains essential informations to be used directly by the models.

The predictions on the original dataset 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 selected model.
The same objects are produced for the independent data (if any).

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

For instance, we examine the probability of occurrence of the first species, modelled with GBM. Here we just display the first 50 rows (or sites)

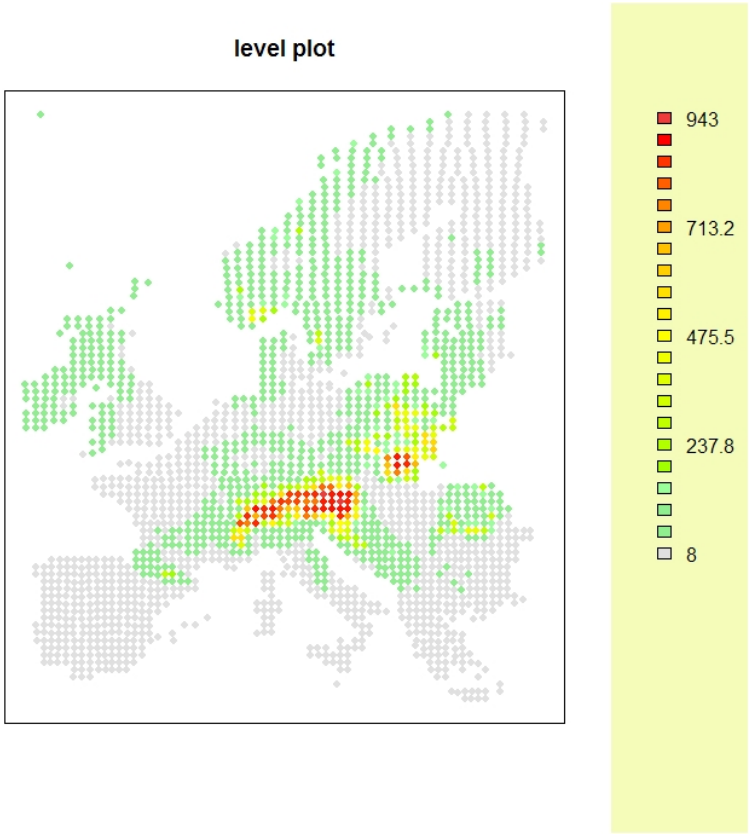
```
> load(paste(root, "/pred/Pred_Species2", sep=""))
> Pred_Species2[450:500,"GBM"]
```

```
450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466
 25  28  28   9   9   9   9   9   9   9   9   9   9   9  47  79  23
467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483
 26  18  19  18  21   9   9   9  89   9  48   9  29 256  26 146  18
484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500
 18   9   9  21  22  20  22  18  21  20  22   9  14   9  18   9   9
```

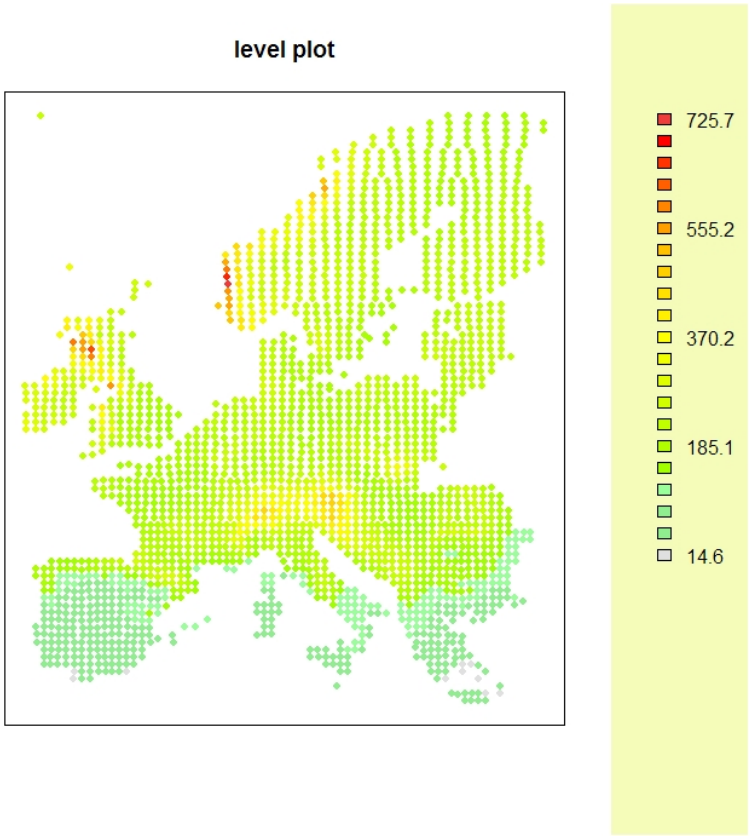
Because GBM contains a stochastic component, the user might end up with slightly different values.

To plot the predictions, use the *plot.level* 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. Here are two examples of its possible use, first plotting the prediction of the GBM for the first species (partially displayed above), and second showing the values of one the variables used to calibrate the models.

```
> plot.level(Pred_Species2[,"GBM"], CoorXY)
```

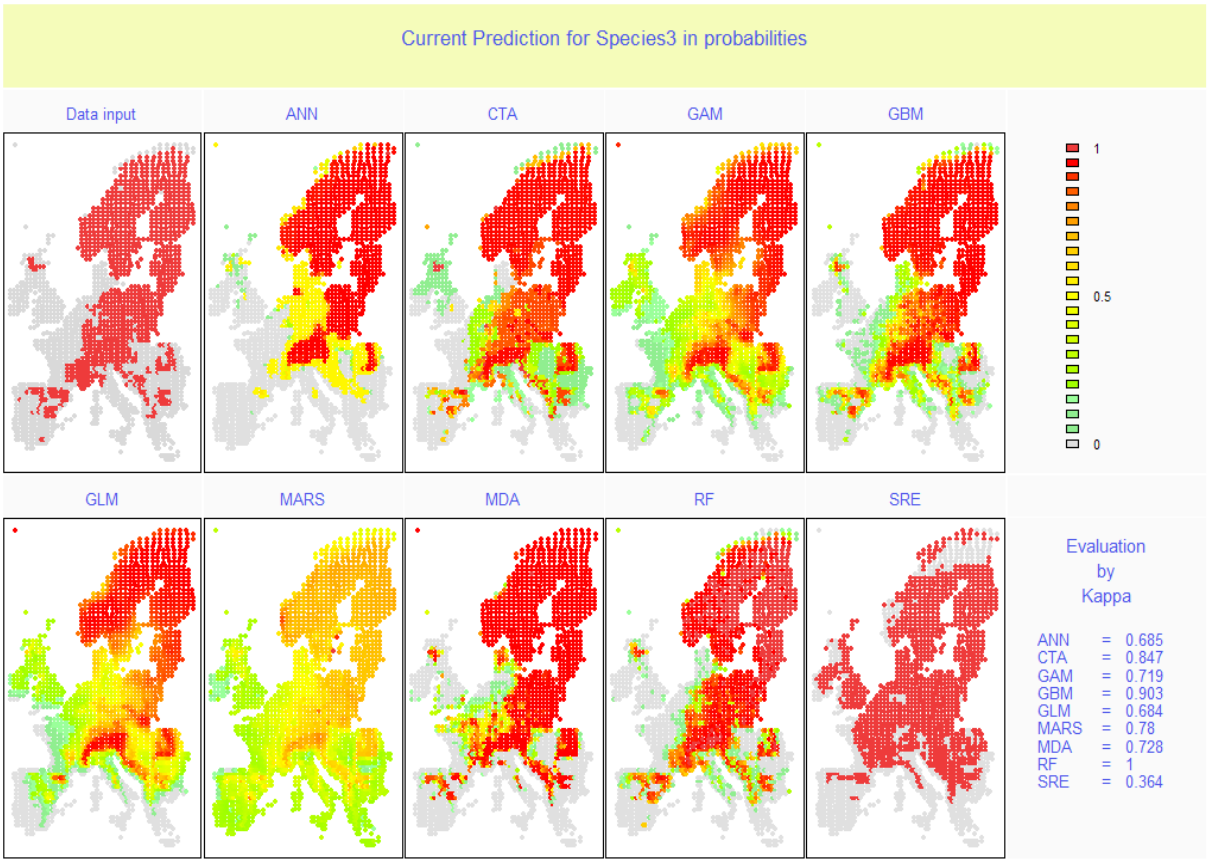


```
> plot.level(Sp.Env[,5], CoorXY)
```



A more fancy version of the *plot.level* function is the *plot.map* function. It works exclusively within BIOMOD and enables a facilitated visualisation and comparison of the different outputs of the model. It requires the model and the species for which you want the plot, but also the format you want your predictions in, and with which method (if a method is needed, for instance having a binary or filtered format which requires a threshold). For example, here we plot the predictions on original data in probabilities for all the models and for the first species.

```
> plot.map(Sp=1, model='all', method='Kappa', format.type='probs', wanted='prediction')
```



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

0.5 Output and interpretation

0.5.1 Interpretation and use of GLM

Depending on what options have been selected during the model setup, the GLM model can contain the various components: the GLM object; This is a reference to the memory of the calibration process. It provides an explanation of the dependent variables selected by the stepwise procedure as well as the residual and null deviances of the model.

```
> load(paste(root, "/models/Species2_GLM", sep=""))
> Species2_GLM


Call:  glm(formula = Species2 ~ poly(Var7, 3) + poly(Var5, 3) + poly(Var3,      3) + poly(Var1, 2) + poly(Var2,

Coefficients:
   (Intercept)  poly(Var7, 3)1  poly(Var7, 3)2  poly(Var7, 3)3
         -15.4           310.3          -137.3           220.4
poly(Var5, 3)1  poly(Var5, 3)2  poly(Var5, 3)3  poly(Var3, 3)1
        -773.3          -608.2          -375.8           516.7
poly(Var3, 3)2  poly(Var3, 3)3  poly(Var1, 2)1  poly(Var1, 2)2
         156.7           205.9           481.3          -269.5
poly(Var2, 3)1  poly(Var2, 3)2  poly(Var2, 3)3
        -223.0          -152.3           -63.9

Degrees of Freedom: 2263 Total (i.e. Null);  2249 Residual
Null Deviance:            1050
Residual Deviance: 491         AIC: 521


> summary(Species2_GLM)


Call:
glm(formula = Species2 ~ poly(Var7, 3) + poly(Var5, 3) + poly(Var3,
    3) + poly(Var1, 2) + poly(Var2, 3), family = binomial, data = DataBIOMOD[sp,
    ], weights = Yweights[sp, i])

Deviance Residuals:
      Min         1Q     Median         3Q        Max
-2.37e+00  -1.69e-01  -3.64e-02  -1.22e-06   3.89e+00

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)     -15.41       2.79   -5.52  3.5e-08 ***
poly(Var7, 3)1   310.26      71.44    4.34  1.4e-05 ***
poly(Var7, 3)2  -137.27      41.77   -3.29  0.00102 **
poly(Var7, 3)3   220.39      47.72    4.62  3.9e-06 ***
poly(Var5, 3)1  -773.33     128.23   -6.03  1.6e-09 ***
poly(Var5, 3)2  -608.24     159.90   -3.80  0.00014 ***
poly(Var5, 3)3  -375.84      83.01   -4.53  6.0e-06 ***
poly(Var3, 3)1   516.65      70.33    7.35  2.0e-13 ***
poly(Var3, 3)2   156.68      80.51    1.95  0.05164 .
poly(Var3, 3)3   205.91      55.19    3.73  0.00019 ***
poly(Var1, 2)1   481.30     271.69    1.77  0.07648 .
poly(Var1, 2)2  -269.50     104.52   -2.58  0.00992 **
poly(Var2, 3)1  -222.95      92.06   -2.42  0.01544 *
poly(Var2, 3)2  -152.31      77.41   -1.97  0.04913 *
poly(Var2, 3)3   -63.88      34.96   -1.83  0.06765 .
---
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: 1050.47  on 2263  degrees of freedom
Residual deviance:  490.55  on 2249  degrees of freedom
AIC: 520.5

Number of Fisher Scoring iterations: 13
```

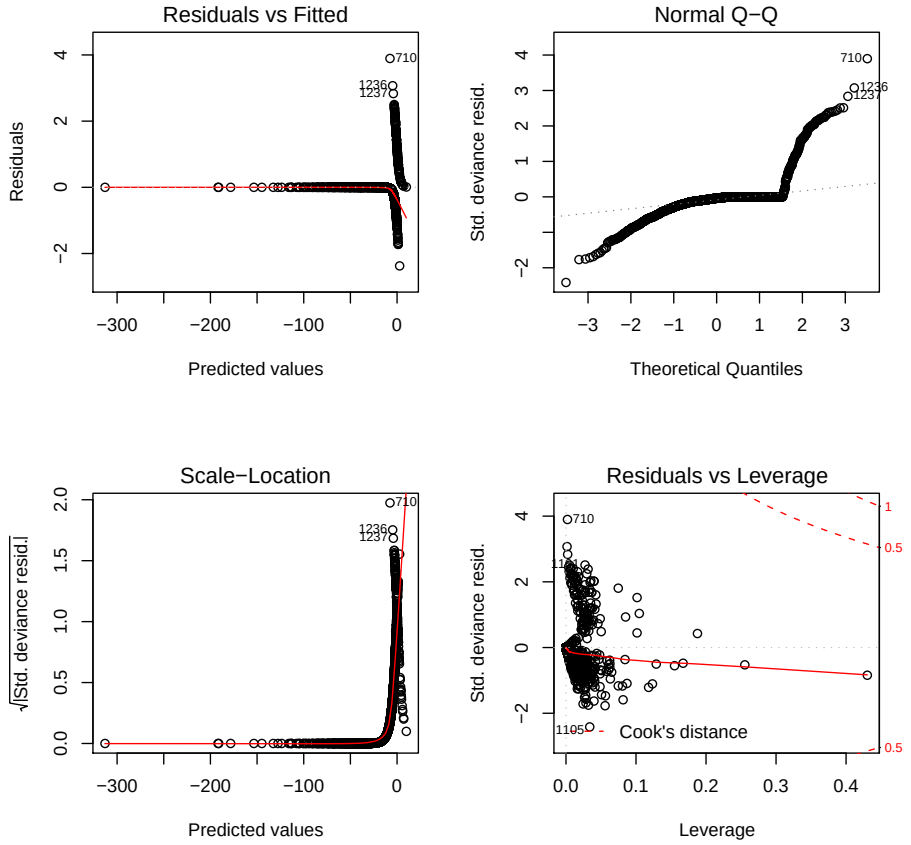
The next call obtains the anova results and the details of the stepwise procedure type. Note that the independent variables are ranked by their AIC importance.

```
> Species2_GLM$anova
```

```
Stepwise Model Path
Analysis of Deviance Table

Initial Model:
Species2 ~ 1

Final Model:
Species2 ~ poly(Var7, 3) + poly(Var5, 3) + poly(Var3, 3) + poly(Var1,
    2) + poly(Var2, 3)


             Step Df Deviance Resid. Df Resid. Dev    AIC
1                                  2263     1050.5 1052.5
2 + poly(Var7, 3)  3  241.431      2260      809.0  817.0
3 + poly(Var4, 2)  2  141.358      2258      667.7  679.7
4 + poly(Var5, 3)  3   91.682      2255      576.0  594.0
5 + poly(Var3, 3)  3   50.359      2252      525.6  549.6
6 + poly(Var1, 2)  2   26.919      2250      498.7  526.7
7 + poly(Var2, 3)  3   11.112      2247      487.6  521.6
8 - poly(Var4, 2)  2    2.944      2249      490.5  520.5
```

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(Species2_GLM)
```



0.5.2 Interpretation and use of GBM

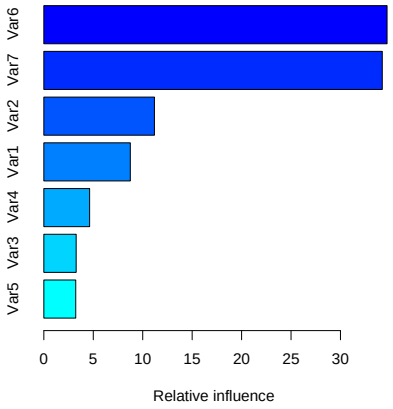
The same kind of outputs obtained for GLM can be extracted for GBM and for the others models. Here we willjust present the outputs of GBM.

The function 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.
Make sure the GBM library is uploaded.

```
> load(paste(root, "/models/Species3_GBM", sep=""))
> summary(Species3_GBM)
```

```
   var rel.inf
1 Var6  34.712
2 Var7  34.228
3 Var2  11.186
4 Var1   8.746
5 Var4   4.632
```

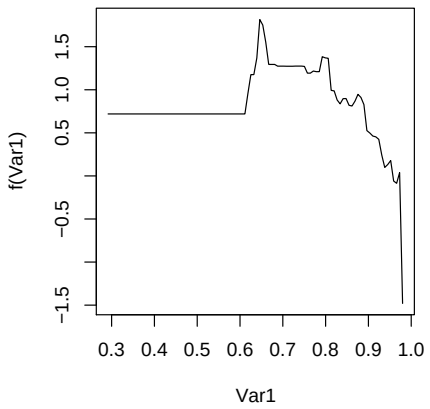
```
6 Var3   3.270
7 Var5   3.227
```



Response curves.

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.

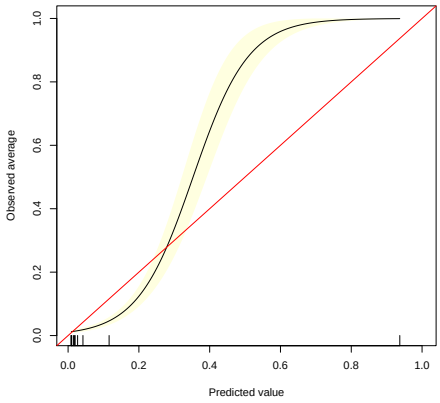
```
> plot(Species3_GBM, i.var=1)
```



The user can also use the custom *plot.response* function presented for GLM.

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

```
> library(gbm)
> calibrate.plot(DataBIOMOD[,8], Pred_Species2[,"GBM"]/1000)
```



The function requires the observed presence-absence of the selected species and the predictions. Note that this function can also be used with any models in R-BIOMOD.

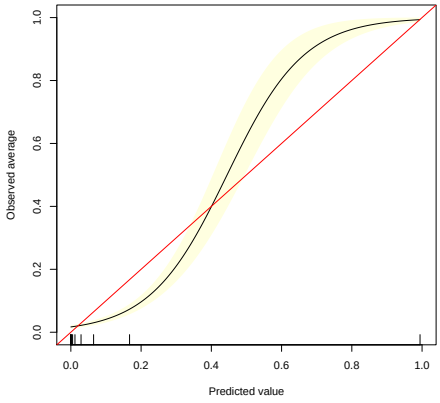
0.5.3 Interpretation and use of GAM

The outputs are very similar than for GLM.

Response curves can be plotted easily with an internal function from the *gam*.

All the tools provided by R to examine GAM results are available (make sure the GAM library is uploaded). As shown for GBM outputs, the user can use the "calibrate.plot" function of the library gbm to plot the accuracy of the model:

```
> calibrate.plot(DataBIOMOD[,8], Pred_Species2[,"GAM"]/1000)
```



0.5.4 Interpretation and use of 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(paste(root, "/models/Species2_CTA", sep=""))
> names(Species2_CTA)
```

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

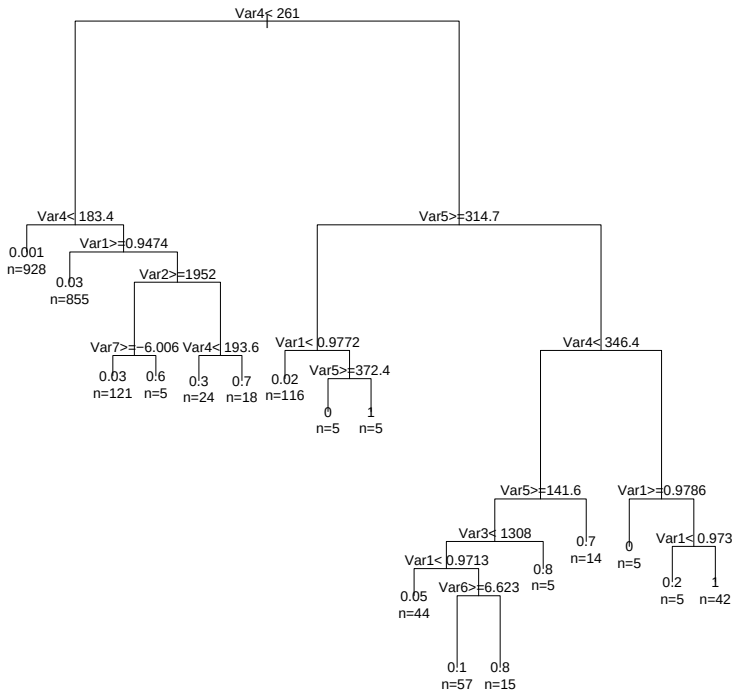
```
> Species2_CTA$frame
```

```
      var    n   wt      dev     yval complexity ncompete nsurrogate
1    Var4 2264 2264 131.3428 0.061837  0.1179606        4          4
2    Var4 1951 1951  54.3926 0.028703  0.0234078        4          5
4  <leaf>  928  928   0.9989 0.001078  0.0007573        0          0
5    Var1 1023 1023  52.0430 0.053763  0.0234078        4          5
10 <leaf>  855  855  27.0830 0.032749  0.0095381        0          0
11   Var2  168  168  22.6607 0.160714  0.0234078        4          5
22   Var7  126  126   6.6111 0.055556  0.0117505        4          1
44 <leaf>  121  121   3.8678 0.033058  0.0016083        0          0
45 <leaf>    5    5   1.2000 0.600000  0.0010000        0          0
23   Var4   42   42  10.4762 0.476190  0.0145173        4          5
46 <leaf>   24   24   4.9583 0.291667  0.0066591        0          0
47 <leaf>   18   18   3.6111 0.722222  0.0052873        0          0
3    Var5  313  313  61.4569 0.268371  0.0793455        4          4
6    Var1  126  126   6.6111 0.055556  0.0176850        4          1
12 <leaf>  116  116   1.9655 0.017241  0.0035443        0          0
13   Var5   10   10   2.5000 0.500000  0.0176850        4          5
26 <leaf>    5    5   0.0000 0.000000  0.0010000        0          0
27 <leaf>    5    5   0.0000 1.000000  0.0010000        0          0
7    Var4  187  187  45.2941 0.411765  0.0793455        4          1
14   Var5  135  135  25.9259 0.259259  0.0246231        4          2
28   Var3  121  121  19.8347 0.206612  0.0217885        4          1
56   Var1  116  116  17.1983 0.181034  0.0217885        4          4
112 <leaf>   44   44   1.9091 0.045455  0.0026994        0          0
113  Var6   72   72  13.9861 0.263889  0.0217885        4          2
226 <leaf>   57   57   6.1404 0.122807  0.0076311        0          0
227 <leaf>   15   15   2.4000 0.800000  0.0091364        0          0
57 <leaf>    5    5   0.8000 0.800000  0.0010000        0          0
29 <leaf>   14   14   2.8571 0.714286  0.0058492        0          0
15   Var1   52   52   8.0769 0.807692  0.0274765        4          3
30 <leaf>    5    5   0.0000 0.000000  0.0010000        0          0
31   Var1   47   47   4.4681 0.893617  0.0204952        4          2
62 <leaf>    5    5   0.8000 0.200000  0.0010000        0          0
63 <leaf>   42   42   0.9762 0.976190  0.0013415        0          0
```

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(Species2_CTA, margin=0.05)
> text(Species2_CTA, use.n=T)
```



Even with CTA, the *plot.response* function allows to plot the response curves. This shows the categorical rule-based approach of CTA which makes sharp relationships.

0.5.5 Interpretation and use of ANN

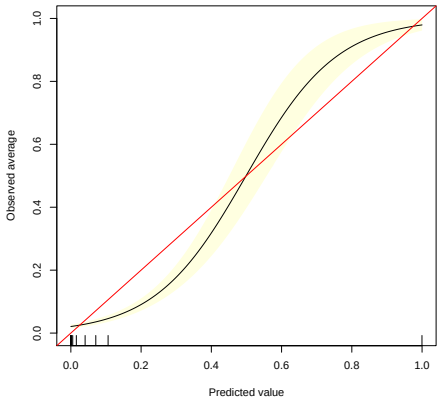
Similarly to GLM, GAM, or CTA, we can plot the response curves of the species to the selected environmental variables using the *plot.response* function.

The user can also plot the relative goodness of fit of the model, using the *calibrate.plot* function from the library(gbm):

```
> load(paste(root, "/models/Species2_ANN", sep=""))
> names(Species2_ANN)
```

```
 [1] "n"           "nunits"      "nconn"       "conn"
 [5] "nsunits"     "decay"       "entropy"     "softmax"
 [9] "censored"    "value"       "wts"         "fitted.values"
[13] "residuals"   "lev"         "call"        "terms"
[17] "coefnames"   "xlevels"
```

```
> calibrate.plot(DataBIOMOD[,8], Pred_Species2[,"ANN"]/1000)
```



0.5.6 Interpretation and use of SRE

There are no models here as this is simply a rectilinear envelop. Like GLM, GAM, CTA, and ANN, SRE.list contains for each species the predictions.
Note also that there is no ROC evaluation available, since SRE does not provide probability values but only the presence-absence of the species. Only TSS and Kappa are available.

0.5.7 Interpretation and use of MDA

Depending on what options have been selected during the model setup, MDA.list can contain the following components: For instance for the first species:

```
> load(paste(root, "/models/Species2_MDA", sep=""))
> names(Species2_MDA)
```

```
 [1] "percent.explained" "values"            "means"
 [4] "theta.mod"         "dimension"         "sub.prior"
 [7] "fit"               "call"              "weights"
[10] "prior"             "assign.theta"      "deviance"
[13] "confusion"         "terms"
```

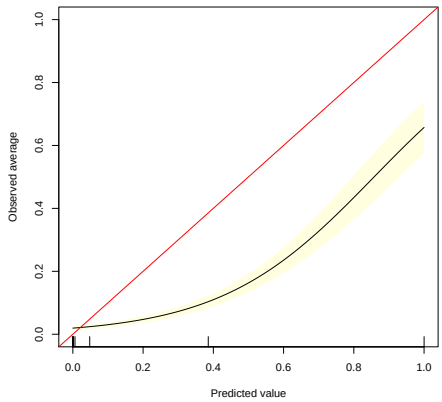
```
> Species2_MDA
```

```
Call:
mda(formula = eval(parse(text = paste(species.names[i], paste(scopeExpSyst(DataBIOMOD[1:10,
    1:NbVar], "MDA"), collapse = "")))), data = DataBIOMOD[sp,
    ], method = mars)

Dimension: 5

Percent Between-Group Variance Explained:
    v1     v2     v3     v4     v5
 59.91  84.68  97.07  99.22 100.00

Training Misclassification Error: 0.07067 ( N = 2264 )

Deviance: 1298
```

Similarly to the previous models, we can also plot the response curves of the species against the selected environmental variables using the *plot.response* function.

To plot the relative goodness of fit using the *calibrate.plot* function of the library(gbm).

```
> calibrate.plot(DataBIOMOD[,8], Pred_Species2[,"MDA"]/1000)
```



0.5.8 Interpretation and use of MARS

Depending on what options have been selected during the model setup, MDA.list can contain the following components: For instance for the first species

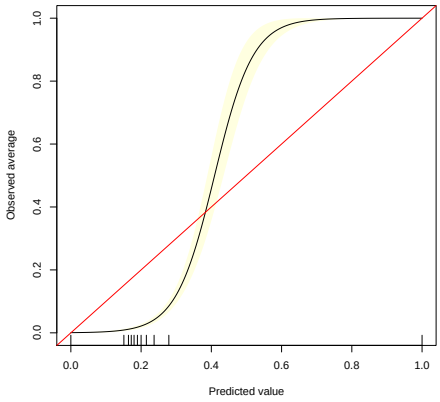
```
> load(paste(root, "/models/Species2_MARS", sep=""))
> names(Species2_MARS)
```

```
 [1] "call"           "all.terms"      "selected.terms"
 [4] "penalty"        "degree"         "nk"
 [7] "thresh"         "gcv"            "factor"
[10] "cuts"           "residuals"      "fitted.values"
[13] "lenb"           "coefficients"   "x"
```

Like for the previous models, we can also plot the response curves of the species against the selected environmental variables using the *plot.response* function.

To plot the relative goodness of fit using the *calibrate.plot* function of the library(gbm):

```
> calibrate.plot(DataBIOMOD[,8], (Pred_Species2[,"MARS"]/1000))
```



0.5.9 Interpretation and use of RF

```
> load(paste(root, "/models/Species2_RF", sep=""))
> names(Species2_RF)
```

```
 [1] "call"            "type"            "predicted"
 [4] "err.rate"        "confusion"       "votes"
 [7] "oob.times"       "classes"         "importance"
[10] "importanceSD"    "localImportance" "proximity"
[13] "ntree"           "mtry"            "forest"
[16] "y"               "test"            "inbag"
```

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

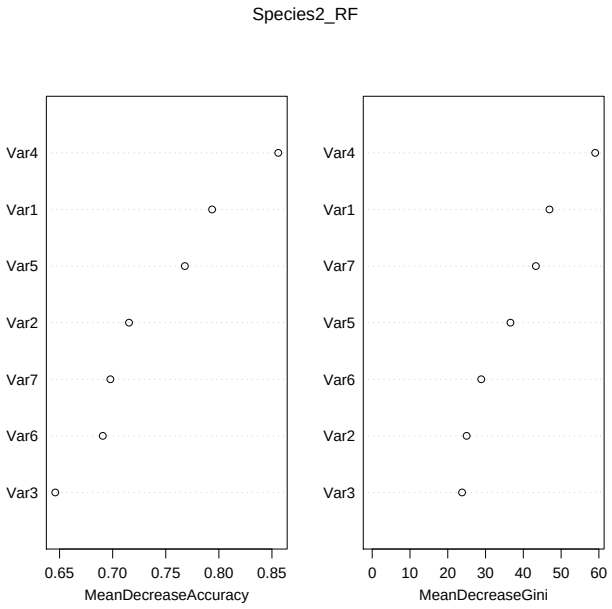
```
> Species2_RF$importance
```

```
            0         1 MeanDecreaseAccuracy MeanDecreaseGini
Var1 0.05461  0.108879              0.05795            46.92
Var2 0.04246 -0.000873              0.03979            24.99
Var3 0.02967  0.034438              0.02995            23.79
Var4 0.06358  0.258261              0.07549            59.02
Var5 0.03862 -0.009182              0.03568            36.59
Var6 0.06531  0.040275              0.06378            28.87
Var7 0.04208  0.126647              0.04725            43.32
```

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.

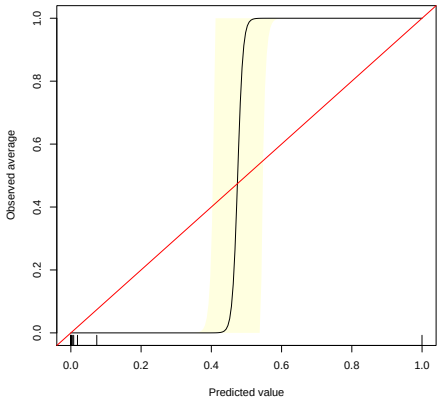
```
> varImpPlot(Species2_RF)
```



An assesment of the importance of each variable using the permutation process (similar for all the models) is also available if this option was selected. Similarly, the response curves using the *plot.response* can be plotted.

To plot the relative goodness-of-fit using the *calibrate.plot* function of the library(gbm).

```
> calibrate.plot(DataBIOMOD[,8], (Pred_Species2[,"RF"]/1000))
```



0.5.10 Evaluation of the predictive performance

There are three available techniques for making an assessment of a model's performance (c.f. 0.14 Predictive Performance description). 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.

A summary table of the type "Evaluation.results.method" are produced by the *Models* function containing the predictive performance of the models which is convenient for making comparisons across methods and taxa.

```
> Evaluation.results.Kappa
```

```
$Species2
     Cross.validation indepdt.data Final.model Cutoff Sensitivity
ANN             0.578         none      0.6684  50.00       90.00
CTA             0.706         none      0.6948  50.00       73.57
GAM             0.585         none      0.8159  69.58       95.71
GBM             0.707         none      0.8357  73.96       92.14
GLM             0.599         none      0.8113  49.95       96.43
MARS            0.558         none      0.7836 240.00       92.86
MDA             0.498         none      0.6870  50.00       84.29
RF              1.000         none      1.0000 350.00      100.00
SRE             0.225         none      0.5746  10.00       80.71
     Specificity
ANN        76.84
CTA        95.90
GAM        85.88
GBM        91.43
GLM        84.70
MARS       85.50
MDA        84.42
RF        100.00
SRE        76.74

$Species3
     Cross.validation indepdt.data Final.model Cutoff Sensitivity
ANN             0.685         none      0.6839  390.0       87.89
CTA             0.847         none      0.8472  340.0       92.21
GAM             0.719         none      0.7204  537.8       82.96
GBM             0.821         none      0.8216  490.8       88.84
GLM             0.748         none      0.7490  499.5       85.73
MARS            0.780         none      0.7799  460.0       89.27
MDA             0.728         none      0.7288  760.0       84.34
RF              1.000         none      1.0000  400.0      100.00
SRE             0.364         none      0.3617   10.0       81.83
     Specificity
ANN        80.51
CTA        92.51
GAM        89.08
GBM        93.32
GLM        89.17
MARS       88.72
MDA        88.54
RF        100.00
SRE        54.33
```

If ROC has been selected, a dataframe called *Evaluation.results.Roc* is present: it includes the AUC for the cross validation steps, the evaluation on independent data (if any) and the evaluation of the final complete model on the original data. The AUC value given for the cross validation procedure is the mean value of the AUC of all the cross-validation events.

To display the predictive accuracy of the GLM for the first species modelled

```
> Evaluation.results.Roc[[1]]["GLM",]
```

```
    Cross.validation indepdt.data Final.model Cutoff Sensitivity
GLM            0.957         none       0.957 84.915      89.286
    Specificity
GLM      89.218
```

As you can see the GLM has a high predictive accuracy on this particular species. The fairly small decrease of accuracy from the Calibration to the Evaluation is an indication that the model does not tend to overfit the data.
If the Optimized Threshold by ROC has been selected, a cutoff is available (fourth column) estimated using the model built with all data for calibration. It represents the best probability threshold maximising the percentage of presence and absence correctly predicted for the evaluation data. The sensitivity and specifity associated with that threshold are given in the last two columns. This threshold value will be used later to transform probabilities into presence-absence (binary format) or filtered values.

As for the probabilities, the thresholds are scalled from 0 to 1000.

The Kappa and TSS methods follow the same logic as that for the ROC assessment.

0.5.11 Importance of each variable

It is always difficult to compare predictions from different models as they do not rely on the same algorithms, techniques and assumptions about the expected relationship between species distributions and the environment. With a permutation procedure, BIOMOD can extract a measure of relative importance of each variable that is independent of the model. As for the predictive accuracy, the results are stored individually per species and per model. It might be more convenient to extract the results in a summary table.
Running the *Models* function will produce an object called "VarImportance" (only if VarImp was put higher than 0 in the function call). Let's have a look at it.

```
> VarImportance
```

```
$Species2
      Var1  Var2  Var3  Var4  Var5  Var6  Var7
ANN  0.003 0.567 1.052 0.201 0.814 0.410 0.216
CTA  0.750 0.149 0.028 0.852 0.315 0.084 0.070
GAM  0.102 0.000 0.974 0.000 0.724 0.124 0.091
GBM  0.155 0.015 0.004 0.874 0.237 0.027 0.199
GLM  0.216 0.226 0.911 0.000 0.718 0.000 0.542
MARS 0.563 0.614 0.000 0.894 0.100 1.013 0.708
MDA  0.000 0.635 0.737 0.459 0.439 1.019 0.526
RF   0.301 0.089 0.055 0.645 0.128 0.125 0.209
SRE  0.034 0.014 0.016 0.107 0.045 0.011 0.205

$Species3
      Var1  Var2  Var3  Var4  Var5  Var6  Var7
ANN  0.000 0.139 0.505 0.499 0.307 0.029 0.055
CTA  0.220 0.155 0.056 0.000 0.070 0.549 0.422
GAM  0.056 0.287 0.364 0.000 0.462 1.385 0.044
GBM  0.044 0.066 0.007 0.024 0.009 0.171 0.334
GLM  0.072 0.472 0.243 0.030 0.278 1.288 0.168
MARS 0.545 1.106 0.000 0.563 0.000 1.014 0.277
MDA  0.111 0.431 0.000 0.071 0.000 0.743 0.174
RF   0.105 0.202 0.022 0.056 0.029 0.203 0.382
SRE  0.034 0.039 0.008 0.030 0.021 0.017 0.074
```

Remember that the importance of each variable is one minus the correlation score between the original prediction and the prediction made with a permuted variable. The range of value is therefore scaled between 0 and 1. A value of 1 means high importance whereas a value is 0 means no

importance.

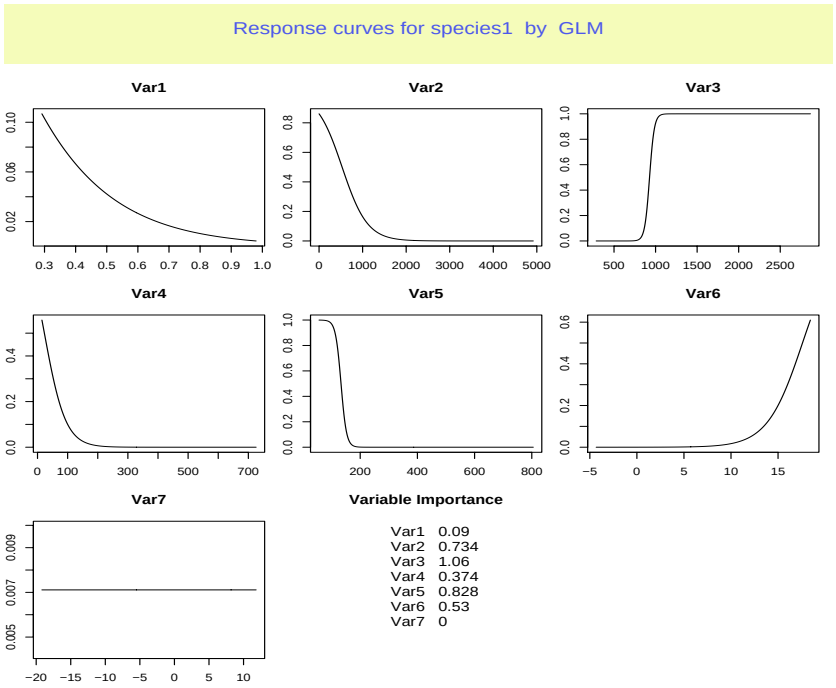
NOTE : The original correlation can be negative. We consider these cases to represent an even bigger influence of the permutated variable on the prediction than with a correlation of 0. The variable importance estimation will therefore still be given as 1 minus the correlation score and, as a consequence, turn into values higher than 1. These cases are not so rare.

0.5.12 Response curves

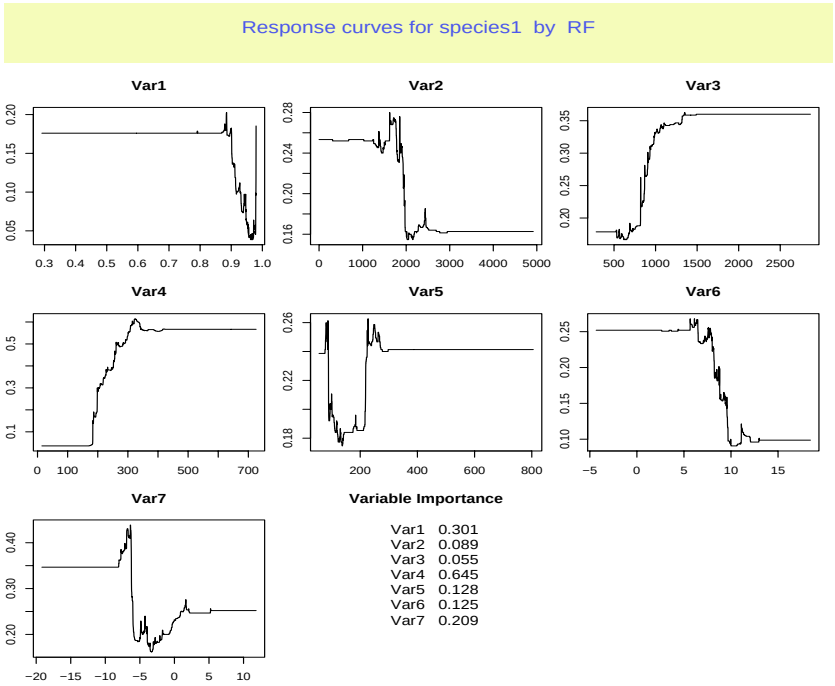
BIOMOD allows plotting the response curves of every model in the good scale. The *plot.response* function must be used to this matter. This function requires a selected model and a selected species to plot the response curves.

Here are two examples of the GLM and RF for the first species modelled:

```
> plot.response("GLM", Sp=1)
```




```
> plot.response("RF", Sp=1)
```



For this, 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 the variable's range. Variation in predictions, made to these 100 cells, only reflects the effects of variation 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.

0.5.13 Predictions on the original dataset

The predictions made by each model for each species are stored inside the *pred* folder. We considered it to be more convenient to have a matrix with the predictions by species for all the models.

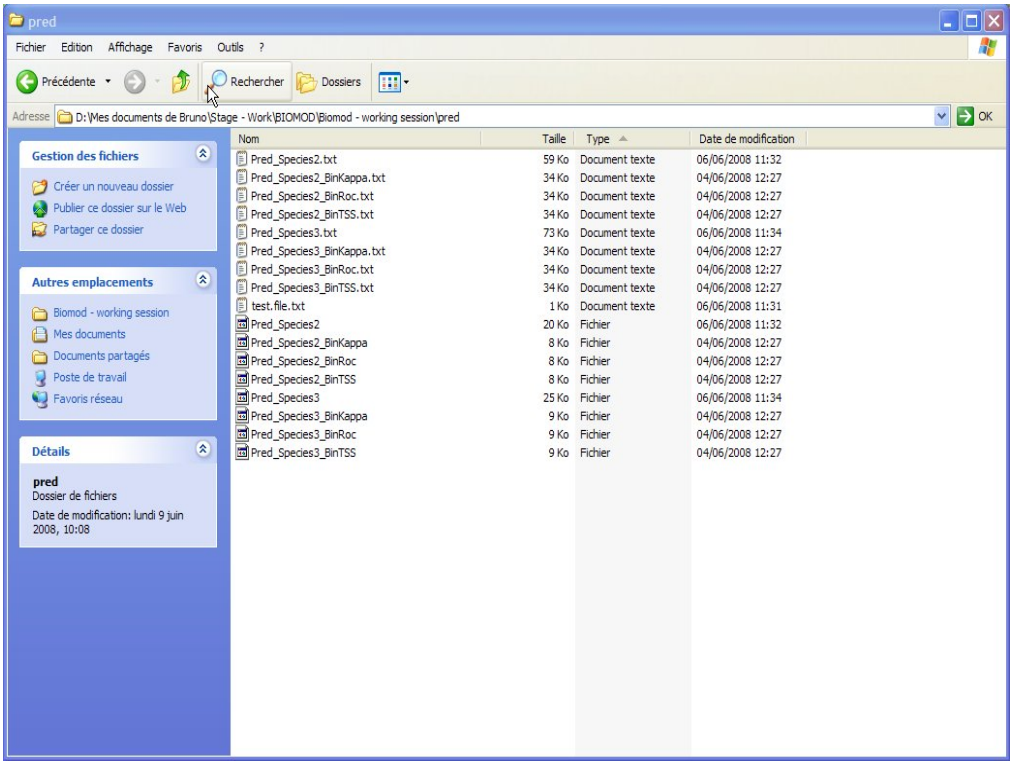
```
> CurrentPred(GLM=T, GBM=T, GAM=T, CTA=T, ANN=T, SRE=T, MDA=T, MARS=F, RF=T,
 BinRoc=T, BinKappa=T, BinTSS=T)
```

For each selected model (the models that were not run will be automatically switched off) an new object will be created for each species of the type Pred_Speciesname, Pred_Species4 for example (do keep in mind that they are not stored directly in the workspace and that a loading of the file is necessary). They contain the predictions made by all the models expressed as a probability of occurrence (remember that the scale is between 0 and 1000).

It might be useful to extract the presence/absence predictions. To do so, switch *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.

Additional datasets will be created: Pred_Species3_BinRoc, Pred_Species3_BinKappa, Pred_Species4_FiltTSS, and so on. The files with no extensions are to be read with R only while the text files can easily be

used for other purposes (e.g. load it into another software).



0.6 Models' projection

For all the models currently implemented, BIOMOD is able to project potential distributions of species or land-use classes for other areas, other resolutions or other times. BIOMOD does not utilise the geographical coordinates nor does it perform a re-ordering of the data for making projections. The user should ensure that all datasets are kept in the same order in order to allow unmistaken comparisons between observed and predicted maps.

To make the projections, use the function *Projection*.

Two hypothetical future climate databases are provided with BIOMOD. They are simply called *Future1* and *Future2*. Note that the labels of the columns should be exactly the same as those of the explanatory variables (or independent variables) of the calibration datasets.

The syntax is very similar to previous functions. First add the new data (e.g. climate change scenario), then the prefix name of the output, and then the models for which the projections have to be made. Then, the user can select if the data needs to be transformed into a binary presence/absence format, or be filtered by a threshold (Kappa, ROC or TSS).

```
> Future1<-read.table("Future1.txt", h=T, sep="\t")
> Projection(Proj = Future1[,2:8], Proj.name='Future1', GLM = T, GBM = T, GAM = T,
    CTA = T, ANN = T, SRE = T, Perc025=T, Perc05=F, MDA =T, MARS = T, RF = T,
    BinRoc = T, BinKappa = T, BinTSS = T, FiltRoc = T, FiltKappa = T, FiltTSS = T)
```

```
Species2
Species3
```

Let's check the future projections made by GLM:

```
> load(paste(root, "/proj/Proj_Future1_Species2", sep=""))
> load(paste(root, "/proj/Proj_Future1_Species2_BinRoc", sep=""))
> Proj_Future1_Species2[740:760, ]
```

```
    ANN CTA GAM GBM GLM MARS  MDA  RF  SRE
740 697 122 137 211 277  227    0 345    0
741  56  33   3  34   7  628    0   6    0
742 283 122  44  54  77  652    0 154    0
743  27  33   2  26   2  631    0   8    0
744 617 976 591 598 654  552 1000 570 1000
745 986 976 625 508 474  509  632 324 1000
746 222 976 245 836 670  647  999 910    0
747 347 800 152 262 268  597    0 425 1000
748 108 122  13  49  25   44    0 118    0
749 536 976 243 334 213  280    0 324 1000
750 227 122  46  80  63  210    0 121 1000
751 348 800 204 556 275  508  905 622 1000
752   0   1   0   9   0  612    0   0    0
753   0   1   0   9   0  616    0   0    0
754   0   1   0   9   0  615    0   0    0
755   0   1   0   9   0  615    0   0    0
756   2   1   0  10  45  623    0  14    0
757   0   1   0   9  22  609    0  30    0
758  10   1   0   9  16  625    0  26    0
759   0   1   0   9   3  616    0   2    0
760   7   1   0   9   0  617    0   0    0
```

```
> Proj_Future1_Species2_BinRoc[740:760,]
```

	ANN	CTA	GAM	GBM	GLM	MARS	MDA	RF	SRE
740	1	1	1	1	1	0	0	0	0
741	0	1	0	0	0	1	0	0	0
742	1	1	0	0	0	1	0	0	0
743	0	1	0	0	0	1	0	0	0
744	1	1	1	1	1	1	1	0	1000
745	1	1	1	1	1	1	1	0	1000
746	1	1	1	1	1	1	1	1	0
747	1	1	1	1	1	1	0	0	1000
748	1	1	0	0	0	0	0	0	0
749	1	1	1	1	1	1	0	0	1000
750	1	1	0	1	0	0	0	0	1000
751	1	1	1	1	1	1	1	1	1000
752	0	0	0	0	0	1	0	0	0
753	0	0	0	0	0	1	0	0	0
754	0	0	0	0	0	1	0	0	0
755	0	0	0	0	0	1	0	0	0
756	0	0	0	0	0	1	0	0	0
757	0	0	0	0	0	1	0	0	0
758	0	0	0	0	0	1	0	0	0
759	0	0	0	0	0	1	0	0	0
760	0	0	0	0	0	1	0	0	0

0.7 Models' optimisation

0.7.1 Predictions on the original datasets

BIOMOD has been programmed to allow direct comparisons between models during the process. This provides a flexible way to derive optimised predictions.

The function *PredictionBestModel* will check, iteratively for each species, which model has the highest predictive accuracy according to the selected method (Roc, Kappa or TSS). Type T (TRUE) or F (FALSE) for each model you want for the optimisation. Note that if you have run the *Models* function using all models, it is not necessary to run the optimisation on all the models, but only the one which might be of interest.

The function will create new datasets prefixed *PredBestModelByXXX* (with XXX being replaced by the evaluation method used, Kappa, ROC or TSS) where the predictions on the original dataset will be stored according to the model selected. For instance, the first species could be predicted using GLM, while the second one by GAM. The selected model, the predictive accuracy, the associated threshold as well as the sensitivity and specificity of the selected models are stored in the new dataset: *BestModelByRoc*. One could choose only the optimisation run on only one evaluation method (e.g. *method='Kappa'*), or all (e.g. *method='all'*). Two additional options can also be selected : as the previous option generates probability values, users who want binary transformation can type: *Bin.trans = T*. In this case, new datasets will be created depending on the evalution method used, e.g. *PredBestModelByRoc.BinRoc*.

If users want probability values above the threshold used to predict presences to be kept (i.e., only probabilities below the threshold are set to zero, the others are left as they were), then type: *Filt.trans = T*.

In our example, we could compare all the models we run for the different species using the three different evaluation methods available. We also transform the probabilities into the presence/absence and filtered probabilities.

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

Multimodel comparison according to the TSS statistic:

```
> BestModelByTSS
```

```
         Best.Model Cal Eval Tot CutOff  Se  Sp
Species2         RF   1 none   1    350 100 100
Species3         RF   1 none   1    400 100 100
```

Multimodel comparison according to the ROC:

```
> BestModelByRoc
```

```
         Best.Model Cal Eval Tot CutOff  Se  Sp
Species2         RF   1 none   1    612 100 100
Species3         RF   1 none   1    577 100 100
```

Multimodel predictions according to the Kappa statistic

```
> PredBestModelByKappa[740:760,]
```

```
    Species2 Species3
740      714      870
741       18      108
742      689      930
743        1      109
744      998     1000
745      980      997
746      946      986
747      752     1000
748       57      966
749      938      988
750      774      977
751      993     1000
752        0       50
753        0        0
754        0      116
755        0       20
756        4      780
757       54      649
758        9      161
759       14      102
760        6       40
```

Multimodel predictions according to the ROC, transformed in binary presence/absence

```
> PredBestModelByRoc.Bin[740:760,]
```

```
    Species2 Species3
740        1        1
741        0        0
742        1        1
743        0        0
744        1        0
745        1        1
746        1        1
747        1        0
748        0        1
749        1        1
750        1        1
751        1        0
752        0        0
753        0        0
754        0        0
755        0        0
756        0        1
757        0        1
758        1        0
759        0        0
760        0        0
```

0.7.2 Projections onto the future or other areas

Depending on the model that has been selected as the *best model* into the PredictionBestModel function, optimisation for the future can also be performed using the functions *ProjectionBestModel* according to the selected evaluation method (ROC, Kappa or TSS).

The syntax is the same than in the *Projections* function. The user only needs to give the name of the climatic that will be used when running the *projection* function. Similarly to the *PredictionBestModel* function, the user can also specify if he wants the optimised-projections transformed into presence-absence or filtered, respectively typing: BinRoc=T and FiltRoc=T.

```
> ProjectionBestModel(Proj.name='Future1', Bin.trans=T, Filt.trans=T, method='all')
> Proj.Future1.BestModelByTSS[740:760,]
```

```
    Species2 Species3
740      345      444
741        6      213
742      154      218
743        8      201
744      570      992
745      324      536
746      910      977
747      425      978
748      118      277
749      324      456
750      121      322
751      622      990
752        0      176
753        0        0
754        0      228
755        0       40
756       14      448
757       30      346
758       26      278
759        2      329
760        0      158
```

```
> Proj.Future1.BestModelByTSS.Bin[740:760,]
```

```
    Species2 Species3
740        0        1
741        0        0
742        0        0
743        0        0
744        1        1
745        0        1
746        1        1
747        1        1
748        0        0
749        0        1
750        0        0
751        1        1
752        0        0
753        0        0
754        0        0
755        0        0
756        0        1
757        0        0
758        0        0
759        0        0
760        0        0
```

Note that it is necessary to have run the PredictionBestModel before running the ProjectionBestModel.

Let's check all the new objects that we now have:

```
> ls()
```

```
 [1] "algo"
 [2] "algo.choice"
 [3] "BestModelByKappa"
 [4] "BestModelByRoc"
 [5] "BestModelByTSS"
 [6] "BinaryTransformation"
 [7] "Biomod.Models"
 [8] "Biomod.RangeSize"
 [9] "Biomod.Turnover"
[10] "CoorXY"
[11] "create"
[12] "create.projection.matrix"
[13] "CurrentPred"
[14] "CutOff.Optimised"
[15] "CVnnet"
[16] "CVnnet1"
[17] "DataBIOMOD"
[18] "Dispersal.Limit"
[19] "Ensemble.Forecasting"
[20] "Evaluation.results.Kappa"
[21] "Evaluation.results.Roc"
[22] "Evaluation.results.TSS"
[23] "FilteringTransformation"
[24] "functionkeep"
[25] "Future1"
[26] "g.pred"
[27] "Initial.State"
[28] "KappaRepet"
[29] "KappaSRE"
[30] "KappaStat"
[31] "Migration"
[32] "Models"
[33] "Models.information"
[34] "NbSpecies"
[35] "NbVar"
[36] "plot.level"
[37] "plot.map"
[38] "plot.response"
[39] "Pred_Species2"
[40] "PredBestModelByKappa"
[41] "PredBestModelByKappa.Bin"
[42] "PredBestModelByKappa.Filt"
[43] "PredBestModelByRoc"
[44] "PredBestModelByRoc.Bin"
[45] "PredBestModelByRoc.Filt"
[46] "PredBestModelByTSS"
[47] "PredBestModelByTSS.Bin"
[48] "PredBestModelByTSS.Filt"
[49] "PredictionBestModel"
[50] "printcp"
[51] "ProbDensFunc"
[52] "proj.choice"
[53] "Proj.Future1.BestModelByKappa"
[54] "Proj.Future1.BestModelByKappa.Bin"
[55] "Proj.Future1.BestModelByKappa.Filt"
[56] "Proj.Future1.BestModelByRoc"
[57] "Proj.Future1.BestModelByRoc.Bin"
[58] "Proj.Future1.BestModelByRoc.Filt"
[59] "Proj.Future1.BestModelByTSS"
[60] "Proj.Future1.BestModelByTSS.Bin"
[61] "Proj.Future1.BestModelByTSS.Filt"
[62] "proj.length"
[63] "Proj_Future1_Species2"
```

```
[64] "Proj_Future1_Species2_BinRoc"
[65] "Projection"
[66] "ProjectionBestModel"
[67] "pseudo.abs"
[68] "Rescaler2"
[69] "root"
[70] "SampleMat2"
[71] "scope"
[72] "scopeExpSyst"
[73] "scopeGAM"
[74] "Sp.Env"
[75] "species.names"
[76] "Species2_ANN"
[77] "Species2_CTA"
[78] "Species2_GLM"
[79] "Species2_MARS"
[80] "Species2_MDA"
[81] "Species2_RF"
[82] "Species3_GBM"
[83] "Species3_GLM"
[84] "sre"
[85] "testnull"
[86] "TSS.Stat"
[87] "VarImportance"
```

0.8 Ensemble Forecasting

One difficulty with the use of species distribution models is that the number of techniques available is large and is increasing steadily, making it difficult for 'non-aficionados' to select the most appropriate methodology for their needs ((Elith, J. et al. 2006, Heikkinen, R. et al. 2006)). Recent analyses have also demonstrated that discrepancies between different techniques can be very large, making the choice of the appropriate model even more difficult. This is particularly true when models are used to project distributions of species into independent situations, which is the case of projections of species distributions under future climate change scenarios ((Pearson, R. G. et al. 2006, Thuiller, W. 2004)). A solution for this inter-model variability is to fit ensembles of forecasts by simulating across more than one set of initial conditions, model classes, model parameters, and boundary conditions (for a review see Araújo & New 2007) and analyse the resulting range of uncertainties with bounding box, consensus and probabilistic methodologies rather than lining up with a single modelling outcome ((Araújo, M. B. and New, M. 2007, Thuiller, W. 2007)). BIOMOD offers such a platform for ensemble forecasting.

Several approaches are available for combining ensembles of models in BIOMOD. Here is an example of the use of the *Ensemble.Forecasting* function as well as some details of the different strategies:

```
> Ensemble.Forecasting(pond.method='Kappa', decay=1.6, PCA.median=T, binary=T, mean.bin.method='TSS')
```

Four straightforward means of 'committee averaging' (giving the same weight to all the elements) are done across all the models :

- on the probabilities
- on the binary projection according to the Roc method,
- on the binary projection according to the Kappa method,
- on the binary projection according to the TSS method.

A weighted approach is also available that ranks the models using their evaluation score.

Making a mean on the 0-1 projections gives some sort of probability of presence. For example, for a given site and with the TSS method, 6 projections give a "1" and 2 give a "0". The mean will be 0.75. It is extracted from binary projection and it is therefore not possible to determine a prior threshold. Conversion into binary is nevertheless possible (see *binary* below).

The median value is also calculated on the probabilities given by the models. It is considered to be more reliable because it less influenced by extreme values. A weighting is not possible, nor the determination of a threshold.

Options:
pond.method: the method for ranking the models according to their predictive performance. The decay gives the relative importance of the weights. The default weight *decay* is 1.6; See the example below.

models	GAM	GBM	GLM	ANN	RF	MARS	CTA	MDA
score with Roc	0.96	0.92	0.90	0.88	0.87	0.75	0.72	0.68
decay of 1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
decay of 1.2	0.217	0.181	0.151	0.126	0.105	0.087	0.073	0.061
decay of 1.6	0.384	0.240	0.150	0.094	0.059	0.037	0.023	0.014
decay of 2	0.502	0.251	0.125	0.063	0.031	0.016	0.008	0.004

You can type in any value (it has however to be higher than 1) depending on the strength of discrimination that you want. A decay of 1 is equivalent to a committee averaging (i.e. same weights given to all elments).

PCA.median: this is an alternative approach for obtaining a weighting of models in an ensemble that does not depend on the performance of each modelling technique.
A PCA is run with projected probabilities of all of the models selected. In the current version of BIOMOD, the consensus model is the model whose projection is the most correlated with the first axis of the PCA. However, the PCA approach can be used in several ways. It can be used to select one single consensus model (as currently implemented in BIOMOD), but it can also be used to allow committee averaging across consensus models (models with high loads in the first axis of PCA), or be used to allow committee averaging across models ranking high in different axes of the PCA. Implementations of these methods can be found in Thuiller (2004), Araújo et al. (2005), and Araújo et al. (2006).
In the current version of BIOMOD no output is produced for this option, just the selected model will be informed in the global function's output.

binary: by setting this argument to True, the ensemble forecasting function will also render the consensus projections in a binary format. The thresholds used differ from one method to the other:

- mean on probabilities: converted in binary format by a mean threshold (thus giving 3 possibilities - Roc, Kappa or TSS; you need to set it in the *mean.bin.method* argument),

- weighted mean on probabilities: converted in binary by a weighted mean threshold (using the same method than for ranking),

- Roc-Kappa-TSS means: an arbitrary value of 0.5 is used, meaning that a site is considered suitable if at least half of the projections have projected a presence.

OUTPUTS

This function will be run for all the species at once. It will produces a series of objects named after the consensus method used to render a final projection. These are :

consensus.mean
consensus.mean.pond
consensus.median
consensus.Roc.mean
consensus.Kappa.mean
consensus.TSS.mean.

if binary is set to True, the same names are used with a terminal *.bin* containing the consensus results in binary format.

A list is also produced giving the following details for each species: the predictive performance of each method when applied to current predictions; the weights awarded to the models in the weighting process; the model selected by the PCA.median method (if set to True). The list is stored under the name *consensus.results*.

0.9 Migration

This function allows the inclusion of a very simple migration process when projecting species distributions into the future. The function constraints the projection to occur in a delimited perimeter around the current distribution. The delimited perimeter has to be decided by the user.
The function uses two datasets: the current species distributions and the the future (assuming by default unlimited migration).
The latitude and longitude of the datasets need to be specified in order to calculate the distances allowed for migration.
Note that to be able to use this function, both current and future datasets must be ordered in the same way and have the same coordinates and the same resolution.

Then the migration rate has to be specified. Two options are available. Either the user can specify the same rate for all the species modelled (a number must be given) or specify a different rate for every species modelled (a vector must be given).
For the generic migration rate, type the maximum distance the species could migrate according to the time slice modelled.

For the species-specific migration rate, create a vector (number of rows = number of species) containing for each species the maximum distance the species could migrate.

Finally, give the name where the projections using limited migration will be stored.
For instance *Future1.Migration.1km.per.year*.

Note that the rate of migration should be given in degrees. For instance for a species with a maximum of 1 minute (1.6km) by 10 years. If we project its distribution in 50 years: Rate = $1x0.16667x5$ (where 0.01667 is the conversion from minute to degree).

For projection in 2080: Rate = $1x0.16667x8$.

For a maximum rate of 3 minutes per 10 years (4.8km) in 2080: Rate = $3x0.16667x8$

```
> Migration(CurrentPred = PredBestModelByRoc.Bin, FutureProj =Proj.Future1.BestModelByRoc.Bin,
 X=CoorXY[,1], Y=CoorXY[,2], MaxMigr=5*0.16667*8, Pred.Save="Future1.Migration")
```

..

```
> Future1.Migration[740:760,]
```

```
    V1 V2
740  0  0
741  0  0
742  0  0
743  0  0
744  0  1
745  0  0
746  1  1
747  0  1
748  0  0
749  0  0
750  0  0
751  1  1
752  0  0
753  0  0
754  0  0
755  0  0
756  0  0
757  0  0
758  0  0
759  0  0
760  0  0
```

0.10 Species Turnover

This function allows to estimate species loss, gained, and turnover by pixel for the time slice considered. The function uses two datasets: the current species distributions and the future one (for instance after accounting for migration). Note that predictions for current and future must be in a binary (presence and absence) format. Finally, give the name where the turnover summaries will be stored.

In the stored database, 10 columns are created.

The first four columns are relative numbers: Disa represents the number of species predicted to disappear from the given pixel. Stable0 is the number of species which are currently not in the given pixel and not predicted to migrate. Stable1 represents the number of species currently occurring in the given pixel, and predicted to remains into the future. Gain represent the number of species which are currently absent but predicted to migrate in the given pixel.

PercLoss, PercGain and Turnover are the related percentage estimated as the following:
- PercLoss = 100 x L/(SR)
- PercGain = 100 x G/(SR)
- Turnover = 100 x (L+G)/(SR+G)
Where SR is the current species richness.

CurrentSR represent the current modelled species richness in the given pixel.
FutureSR0Disp represents the future modelled species richness assuming no migration of species
FutureSR1Disp represents the future modelled species richness assuming migration (depending on the datasets given in input, if Migration has been used or not).

```
> Biomod.Turnover(CurrentPred = PredBestModelByRoc.Bin, FutureProj = Future1.Migration,
    Turnover.Save= "Turnover.2050")
> Turnover.2050[740:760,]
```

	Disa	Stable0	Stable1	Gain	PercLoss	PercGain	Turnover	CurrentSR
740	2	0	0	0	100	0	100	2
741	0	2	0	0	NaN	NaN	NaN	0
742	2	0	0	0	100	0	100	2
743	0	2	0	0	NaN	NaN	NaN	0
744	1	0	0	1	100	100	100	1
745	2	0	0	0	100	0	100	2
746	0	0	2	0	0	0	0	2
747	1	0	0	1	100	100	100	1
748	1	1	0	0	100	0	100	1
749	2	0	0	0	100	0	100	2
750	2	0	0	0	100	0	100	2
751	0	0	1	1	0	100	50	1
752	0	2	0	0	NaN	NaN	NaN	0
753	0	2	0	0	NaN	NaN	NaN	0
754	0	2	0	0	NaN	NaN	NaN	0
755	0	2	0	0	NaN	NaN	NaN	0
756	1	1	0	0	100	0	100	1
757	1	1	0	0	100	0	100	1
758	1	1	0	0	100	0	100	1
759	0	2	0	0	NaN	NaN	NaN	0
760	0	2	0	0	NaN	NaN	NaN	0

	FutureSR.0Disp	FutureSR.1Disp
740	0	0
741	0	0
742	0	0
743	0	0
744	0	1
745	0	0
746	2	2
747	0	1
748	0	0
749	0	0
750	0	0
751	1	2
752	0	0
753	0	0

```
754           0           0
755           0           0
756           0           0
757           0           0
758           0           0
759           0           0
760           0           0
```

0.11 Species Range Change

This function allows to estimate the proportion and relative number of pixels (or habitat) lost, gained and stable for the time slice considered.
The function uses two datasets. The current species distributions and the future one. Note that predictions for current and future must be in a binary (presence and absence) format. Finally, give the name where the species range change summaries will be stored.

```
> Biomod.RangeSize(CurrentPred = PredBestModelByRoc.Bin, FutureProj = Future1.Migration,
    SpChange.Save="SpChange.2050")
```

A list of two datasets is created: Compt.By.Species and Diff.By.Pixel

Diff.By.Pixel stores useful information for each species. The species are in columns and the pixel in rows. For each species, a pixel could have four different values:

-2 if the given pixel is predicted to be lost by the species.
-1 if the given pixel is predicted to be stable for the species.
0 is the given pixel was not occupied, and will not be into the future.
1 if the given pixel was not occupied, and is predicted to be into the future.

```
> SpChange.2050$Diff.By.Pixel[740:760,]
```

```
    V1 V2
740 -2 -2
741  0  0
742 -2 -2
743  0  0
744 -2  1
745 -2 -2
746 -1 -1
747 -2  1
748  0 -2
749 -2 -2
750 -2 -2
751 -1  1
752  0  0
753  0  0
754  0  0
755  0  0
756  0 -2
757  0 -2
758 -2  0
759  0  0
760  0  0
```

This table could be easily plotted into GIS software in order to represent the pattern of change for the selected species.

Compt.By.Species stores the summary of range change for each species (by rows).
The first four columns are relative numbers: Disa represents the number of pixels predicted to be lost by the given species. Stable0 is the number of pixels which are not currently occupied by the given species and not predicted to be. Stable1 represents the number of pixels currently occupied by the given species, and predicted to remain occupied into the future. Gain represent the number of pixels which are currently not occupied by the given species but predicted to be into the future. PercLoss, PercGain and SpeciesRangeChange are the related percentage estimating as the following:
- CurrentRangeSize represent the modelled current range size (number of pixels occupied) of the given species.
- FutureRangeSize0Disp represents the future modelled range size assuming no migration of the given species.
- FutureRangeSize1Disp represents the future modelled range size assuming migration of the given species (depending on the datasets given in input, if Migration has been used or not).

```
> SpChange.2050$Compt.By.Species
```

```
         Disa Stable0 Stable1 Gain PercLoss PercGain
Species2  271    1974      17    2    94.10   0.6944
Species3  660     946     280  378    70.21  40.2128
         SpeciesRangeChange CurrentRangeSize FutureRangeSize.0Disp
Species2              -93.4              288                    17
Species3              -30.0              940                   280
         FutureRangeSize.1Disp
Species2                    19
Species3                   658
```

0.12 Other Functionalities

This section presents a series of functionalities that are not directly related to the functioning of BIOMOD. These are to be used on any datasets, considering that they follow the specific requirements. Thus, you do not need to run BIOMOD to use them.

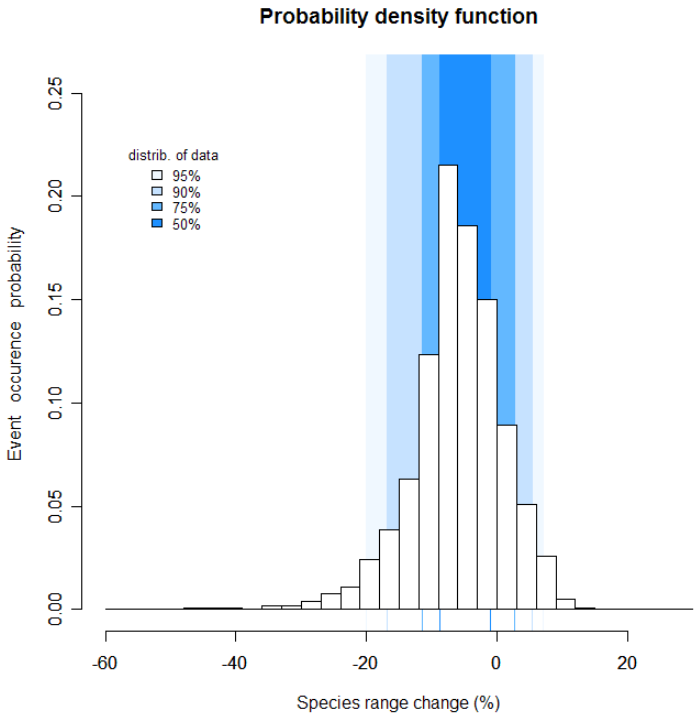
However, do NOT copy the lines presented here and try to run them in an R console. This will inevitably end up in an error message. The code here is used as an example.

0.12.1 Probability Density Function

Using a variety of parameters in modelling will inevitably bring variability in predictions, especially when it comes to making future predictions. This function enables an overall viewing of the future predictions range per species and gives the likelihood of range shift estimations.

The future range changes are calculated as a percentage of the species' present state. For example, if a species currently occupies 100 cells and is estimated by a model to cover 120 cells in the future, the range change will be + 20%.

```
> ProbDensFunc (initial=Sp.Env[,9], projections=Proj[,1:120], distrib=T, cvsn=T, groups=gp, resolution=5)
```



initial: a vector in a binary format (ones and zeros) representing the current distribution of a species which will be used as a reference for the range change calculations.

projection: a matrix grouping all the predictions where each column is a single prediction. MAke sure you keep projections in the same order as the initial vector (line1=site1, line2=site2, etc.).

distrib: if true, the optimal way for condensing 50, 75, 90 and 95% of the data will be calculated and shown on the graph.

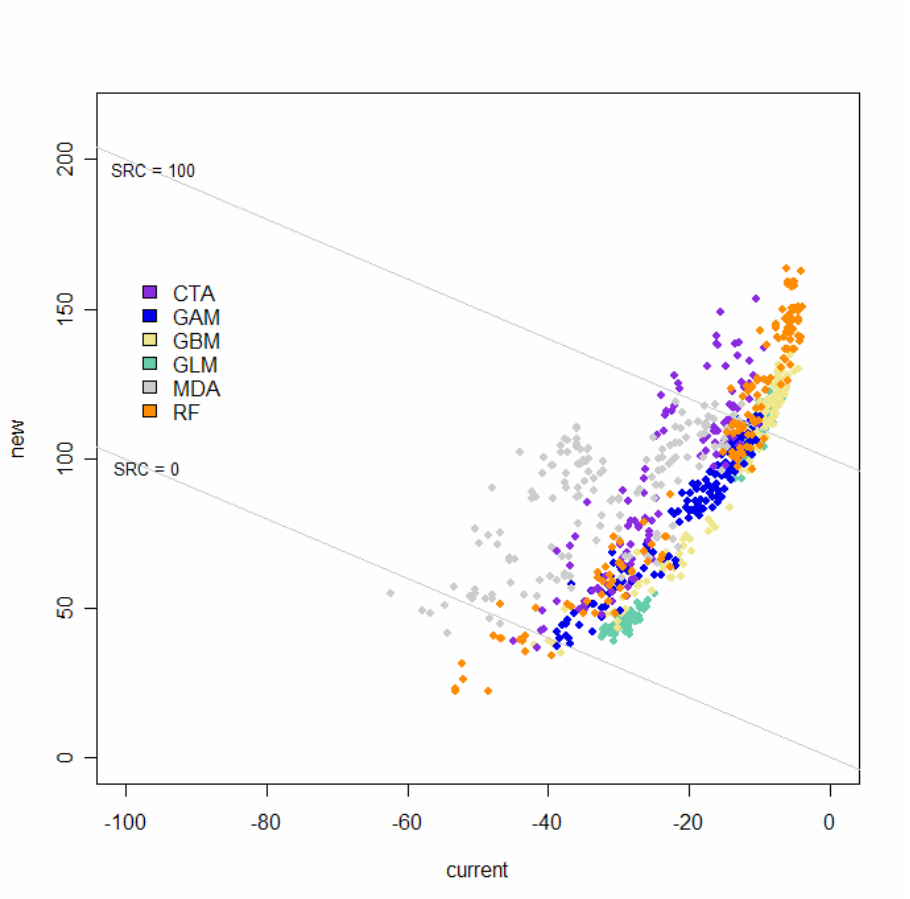
Resolution: the step used for classes of prediction in graphics. The default value is 5.

NOTE: modifying the resolution will directly influence the probability scale. Bigger classes will cumulate a greater number of predictions and therefore represent a greater fraction of the total predictions. The probability is in fact that of the class and not of isolated events.

cvsn: stands for current vs new. If true, the range change calculations will be of two types: the percentage of cells currently occupied by the species to be lost, and the relative percentage of cells currently unoccupied but projected to be, namely 'new' cells, compared to current surface range.

With the example above where the species will have 120 suitable sites in the future whilst only 100 at present, this might be the result of different events. A case could be that the 100 present cells are kept and an additional 20 new sites makes the 120 cells. Another possibility is that the 100 current cells are predicted to be lost with 120 new cells, also giving 120 total cells in future.

These two cases bring the same SRC calculations results, but whilst the first case does not imply much as in survival strategies (the current populations will still be in good conditions in future, plus even having new potential territories to explore and colonise), the second case, however, implies a strong migrating effort for the populations to stay in suitable environments. Those two cases and all in-between possibilities are distinguishable with this method.



Here, each dot is a projection. For example, the one furthest on the left gives the following information: approximately -60% of the current sites will be lost and 50% of new sites will be gained. The SRC is very simply the addition of these two values : -10%. See how this single value does not reflect every thing that is going on: it does not tell that more than half of current habitats are projected to be lost, which would surely lead to different management decisions.

The two lines represent where the SRC value is 0 (no absolute change in the number of suitable sites) and +100% (the species will double its current potential distribution size). Along those line, you have all the possibilities for giving that one value (-10+10=0 ; -40+40=0 ; ...).

An extra feature on this graph is the colours. They enable to differenciate groups of projections with the present example of the models. It enables to view where the variability in projection comes from (see the description of *groups* below). You will have as many as these graphs as lines that you have in the *groups* matrix.

groups: an option for ungrouping the projections enabling a separated visualisation of the prediction range per given group. A matrix is expected where each column is a single prediction and each line is giving details of one parameter. For example, if you have 9 different projections, with 3 models and 3 threshold possibilities, your matrix could look like this:

```
     [,1]  [,2]    [,3]  [,4]  [,5]    [,6]  [,7]  [,8]    [,9]
[1,] "GAM" "GAM"   "GAM" "CTA" "CTA"   "CTA" "RF"  "RF"    "RF"
[2,] "Roc" "Kappa" "TSS" "Roc" "Kappa" "TSS" "Roc" "Kappa" "TSS"
```

or like this:

```
     [,1]  [,2]  [,3]  [,4]    [,5]    [,6]    [,7]  [,8]  [,9]
[1,] "GAM" "CTA" "RF"  "GAM"   "CTA"   "RF"    "GAM" "CTA" "RF"
[2,] "Roc" "Roc" "Roc" "Kappa" "Kappa" "Kappa" "TSS" "TSS" "TSS"
```

Do keep in mind that this matrix represents the projections the way you have put them into the *projection* argument. Sort your matrix the way you have sorted your projections!

Uncertainty Estimation

This function enables an assessment of the variability in predictions. The PDF plot permits a visual assessment but a calculated estimation of each parameter's role in variability is also possible.

uncertainty: if True, the variability due to each parameter entered in the *groups* argument will be calculated. For 3 or less parameters (i.e. 3 lines in the *groups* matrix) a table is given. Here is an example with 3 parameters: 9 models, 3 threshold methods, 5 future climate scenarios. The output will be in a standard R format but this presentation is just for making it easier to read.

	Roc	Kap	Prev	Sc1	Sc2	Sc3	Sc4	Sc5
GLM	0.047	0.050	0.046	0.106	0.107	0.109	0.113	0.109
GBM	0.072	0.092	0.070	0.115	0.118	0.119	0.119	0.118
GAM	0.068	0.074	0.067	0.100	0.098	0.098	0.100	0.097
CTA	0.168	0.167	0.175	0.175	0.184	0.182	0.185	0.181
ANN	0.205	0.225	0.173	0.196	0.206	0.197	0.210	0.200
MDA	0.138	0.136	0.140	0.139	0.154	0.149	0.144	0.150
MARS	0.329	0.271	0.211	0.387	0.368	0.379	0.366	0.374
RF	0.108	0.122	0.095	0.164	0.172	0.168	0.169	0.173
Roc	NA	NA	NA	0.265	0.275	0.273	0.266	0.278
Kap	NA	NA	NA	0.234	0.266	0.249	0.233	0.263
Prev	NA	NA	NA	0.314	0.317	0.313	0.316	0.318

.

You can identify 4 boxes: model/threshold (top left), model/scenarios (top right), threshold/scenarios(bottom right) and threshold/threshold (bottom left which contains NA values meaning 'not available').

Let's take the first value on the top left corner of the matrix. The way to read is as follows: only the projections concerning the GLM with the Roc evaluation method are taken into account. That makes 5 projections, one for each scenario. The standard deviation is measured for each line (i.e. site) of the data across these projections. The value printed in the matrix is the mean of the standard deviations across all lines. It represents the variation due to the different scenarios.

You can see that the effect of the scenarios is more or less constant considering different threshold methods (i.e. the 3 first values of each line) but is more varying across models (i.e. the first 8

values of each column). The impact of different scenarios is the strongest for the MARS, a model known for showing significant discrepancies when making projections. For this model, the threshold method seems to have an even bigger influence.

NOTE: do keep in mind that standard deviations are influenced by how many values you use for the calculation. The more you have, the bigger the chance to have a smoothing of the differences. Also, for example, using extreme future scenarios will bring greater variations than with several middle ones. Be careful when interpeting these values.

0.12.2 Pseudo-absences

The majority of models need information about presences and absences for being able to determine the suitable conditions for a given species. Some data sets, however, do not contain absences but only presences and the construction of virtual absences is therefore needed. This is, for example, the case of bird datasets where determining an absence can be rather tricky. The assumed absences are called pseudo-absences for there is no field verification of this generated information.

These pseudo-absences are created by considering any point where the species was not recorded and where the environmental conditions are known to cause potential absence. Feeding the models with exceeding numbers of absences can significantly disturb the ability of models to discriminate meaningful relationships between climate and species distributions. Moreover, running models on such heavy databases is incredibly time consuming.

In addition, some of the chosen absences might unfortunately represent true presences (this is particularlyl likely in the case of incomplete samples) and therefore the pseudo-absence data gives false information for the estimation of the species-climate relationship. Hence, we propose various strategies that seek to remove the spurious effects of using poorly seelcted pseudo-absences **before** running the models.

Use the *pseudo.abs* function as in the example below.

```
> pseudo.abs(coor=data[,1:2], status=data[,3], strategy='per', env=data[,4:16], distance=10000, plot=F,
    species.name= 'Sp1', acol='grey80', pcol='red', add.pres=T)
```

coor: a 2 columns matrix giving the coordinates of the points - presences and the whole set of potential absences.
status: a vector containing the presence-absence (1-0) information for the *coor* data. Any point for which a "1" is not given will be taken as zero by default, thus considered as an absence.
strategy: (examples on the figure below)
- random: the absences will be taken at random from the whole set of potential absences
- per: stands for the perimeter to be drawn around the presences as a whole.
- perind: same as *per* but the perimeter is drawn individually around each presence. For this strategy, information is needed on the distance wanted (*distance* argument)
- sre: sites where the environment is considered to be possibly favourable to the species (according to the SRE model) are unselected as candidate sites for drawing pseudo-absences. For this

strategy, the *env* argument must be given.

distance: only used for the "perind" strategy. The unit is the one of the *coor* data.

env: needed for the "sre" strategy. A matrix giving information on the environment as a set of variables (just like the one needed to run any model).

species.name: The output will be stored under the name given by this argument, plus the strategy chosen separated by a dot. For example, if you give "larix" in this argument and choose the sre strategy, then the output is stored in a new object named: "larix.sre".

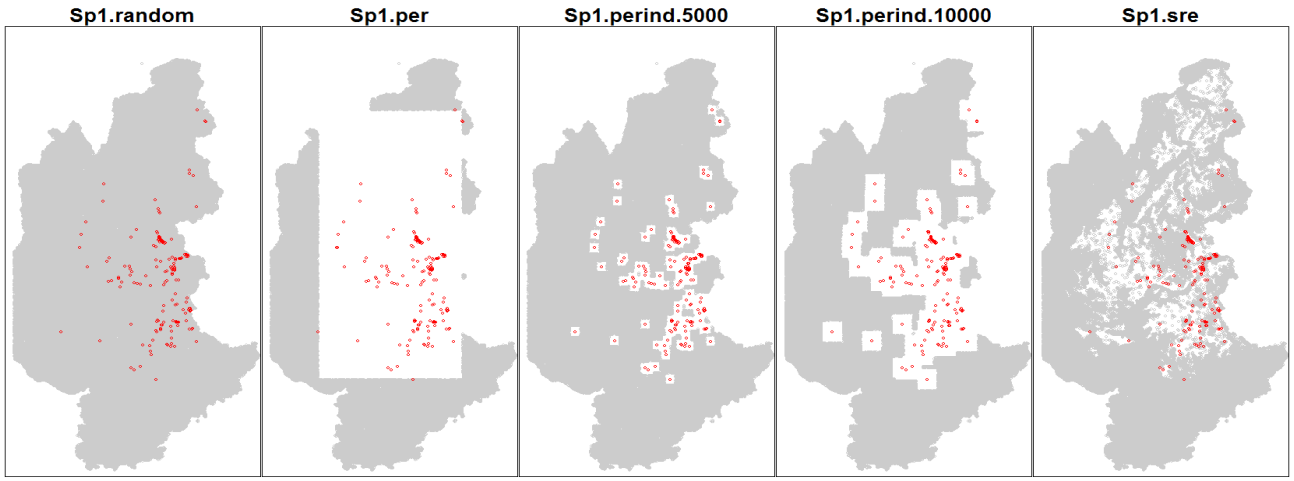
nb.points: an option for selecting only a limited number of absences at random. The default (nb.points=NULL) keeps all the possible absences according to the strategy selected.

add.pres: if True, the output will be an object also containing the presence information (see section below for further explanations).

plot: an option for plotting the outup set of presences and absences obtained.

acol and *pcol*: the colours wanted to plot the absences and presences respectively.

Example of the 4 available strategies in the region of the French Alps for *Larix decidua miller*. The presences are in red and the pseudo-absences selected by each strategy are in grey.



How to correctly use the *pseudo.abs* function output

The output of this function is an object containing the rows of the absences selected by a strategy (and presences if *add.pres* was set to True) from the original full presence-absence dataset. Mind that it will only contain a limited number of absences if you have used the *nb.points* argument. The way to use the output correctly is the following.

Let's say your original full data is stored in an object called "fulldata" and you want to use the sre strategy for selecting pseudo-absences. Run the pseudo.abs function:

```
> pseudo.abs(coor=data[,1:2], status=data[,3], strategy='sre', env=data[,4:16],
    species.name= 'first.species', add.pres=T)
```

An object called "first.species.sre" will be produced containing all the possible absences but also the presences (because I asked for it in the function call). The new data set will be called by:

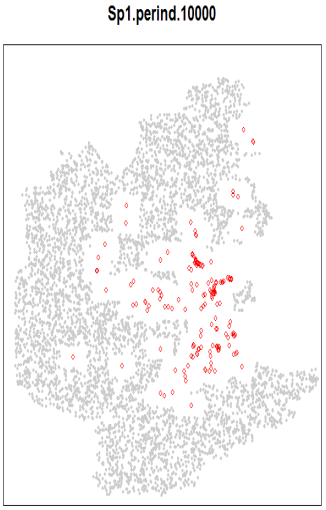
```
> new.data.set <- fulldata[first.species.sre, ]
```

The appropriate lines of the original dataset are called, building a new dataset that was here store under a new name. If you want to pick only 5,000 points from the absences strategy-selected (supposedly that you have more available) or you don't want the presences, the way to proceed is exactly the same by setting the arguments with the appropriate values.

An example :

```
> pseudo.abs(coor=data[,1:2], status=data[,3], strategy='perind', distance=10000, plot=T,
    species.name= 'Sp1', nb.points=5000, add.pres=T)
```

And your dataset will look like this.



0.13 Models' description

0.13.1 GLM - Generalised Linear Models

This provides a less restrictive form than classic multiple regressions by providing error distributions for the dependent variable other than normal and non-constant variance functions. If the response with a predictor variable is not linear, then a transformation can be included where such polynomial terms allow for the simulation of skewed and bimodal responses, -functions or hierarchical sets of models. The associated shortcoming is that the nature of the relationship between species and environmental gradients has to be known a priori. Furthermore, GLM is not always flexible enough to approximate the true regression surface adequately. To select for the most parsimonious model, BIOMOD uses an automatic stepwise model selection. The stepAIC function of Splus (library MASS) builds models by sequentially adding new terms and testing how much they improve the fit, and by dropping terms that do not degrade the fit to a significant amount. The statistical criteria used for selection of models of increasing fit could be either the Akaike Information Criterion (AIC) or the Bayesian Information Criteria (BIC). The stepwise procedure allows the removal of redundancy in variables and reduces multicolinearity (not always).

Three kinds of GLM can be run:
GLM Simple: Used only linear terms.
$Y1 = X1 + X2 + X3 + (X1 * X2) + (X2 * X3)$
GLM Quad: Used linear, 2nd and 3rd order.
$Y1 = X1 + X1^2 + X1^3 + X2^2 + X3^3$
GLM Poly: Use ordinary polynomial terms.
$Y1 = f(X1 + X1^2 + X1^3) + f(X2 + X2^2 + X2^3) +$

If you select GLM, just type GLM = T inside the function call.
If you want to use polynomial terms, type TypeGLM = "poly", or quadratics, TypeGLM = "quad", or using only linear terms, type TypeGLM = "simple" If you want to use the AIC as a selection criteria, just type Test = "AIC", or if you want to use the BIC, just type Test = "BIC".

Key reference.
McCullagh, P. and Nelder, J.A. (1989) Generalized linear models Chapman and Hall.

Key reference in ecology/biogeography.
Austin, M.P. and Meyers, J.A. (1996) Current approaches to modelling the environmental niche of eucalypts: implication for management of forest biodiversity. Forest Ecology and Management, 85, 95-106.
Elith, J., Graham, C.H., Anderson, R.P., Dudik, M., Ferrier, S., Guisan, A., Hijmans, R.J., Huettman, F., Leathwick, J.R., Lehmann, A., Li, J., Lohmann, L., Loiselle, B.A., Manion, G., Moritz, C., Nakamura, M., Nakazawa, Y., Overton, J.M., Peterson, A.T., Phillips, S., Richardson, K., Schachetti Pereira, R., Schapire, R.E., Soberón, J., Williams, S.E., Wisz, M., and Zimmermann, N.E. (2006) Novel methods improve predictions of species' distributions from occurrence data. Ecography, 29, 129-151.
Guisan, A. and Thuiller, W. (2005) Predicting species distribution: offering more than simple habitat models. Ecology Letters, 8, 993-1009.
Guisan, A. and Zimmermann, N.E. (2000) Predictive habitat distribution models in Ecology. Eco-

logical Modelling, 135, 147-186.
Thuiller, W., Araújo, M.B., and Lavorel, S. (2003) Generalized models versus classification tree analysis: a comparative study for predicting spatial distributions of plant species at different scales. Journal of Vegetation Science, 14, 669-680.

0.13.2 GAM - Generalised Additive Models

This has been recently used in ecology to deal with various species response shapes to environmental variables. GAMs are designed to capitalise on the strengths of GLMs without requiring the problematic steps of postulating a response curve shape or specific parametric response function. They use a class of equations called "smoothers" that attempt to generalise data into smooth curves by local fitting to subsections of the data. GAMs are therefore useful when the relationship between the variables are expected to be of a more complex form, not easily fitted by standard linear or non-linear models, or where there is no a priori reason for using a particular model. The idea is to 'plot' the value of the dependent variables (occurrences) along a single environmental variable, and then to calculate a smooth curve that fits the data as closely as possible while being parsimonious. The algorithm fits a smooth curve to each variable and then combines the results additively.
BIOMOD uses a cubic spline smoother, which is a collection of polynomials of degree less than or equal to 3, defined on subintervals. A separate polynomial is fitted for each neighbourhood, thus enabling the fitted curve to join all of the points. Similarly to GLM, BIOMOD uses an automated stepwise process to select the most significant variables for each species.
$Y = s(X1, 4) + s(X2, 4) + s(X3, 4)$.

The user needs to select the number of degree of freedom. By default, the value is 4. Just type Spline = 4. In order words, 4 degrees of freedom is similar to a polynomial of degree 3.

Key reference.
Hastie, T.J. and Tibshirani, R. (1990) Generalized additive models Chapman and Hall, London.

Key reference in ecology/biogeography.
Austin, M.P. and Meyers, J.A. (1996) Current approaches to modelling the environmental niche of eucalypts: implication for management of forest biodiversity. Forest Ecology and Management, 85, 95-106.
Elith, J., Graham, C.H., Anderson, R.P., Dudik, M., Ferrier, S., Guisan, A., Hijmans, R.J., Huettman, F., Leathwick, J.R., Lehmann, A., Li, J., Lohmann, L., Loiselle, B.A., Manion, G., Moritz, C., Nakamura, M., Nakazawa, Y., Overton, J.M., Peterson, A.T., Phillips, S., Richardson, K., Schachetti Pereira, R., Schapire, R.E., Soberón, J., Williams, S.E., Wisz, M., and Zimmermann, N.E. (2006) Novel methods improve predictions of species' distributions from occurrence data. Ecography, 29, 129-151.
Guisan, A. and Thuiller, W. (2005) Predicting species distribution: offering more than simple habitat models. Ecology Letters, 8, 993-1009.
Guisan, A. and Zimmermann, N.E. (2000) Predictive habitat distribution models in Ecology. Ecological Modelling, 135, 147-186.
Thuiller, W., Araújo, M.B., and Lavorel, S. (2003) Generalized models versus classification tree analysis: a comparative study for predicting spatial distributions of plant species at different scales. Journal of Vegetation Science, 14, 669-680.
Yee, T.W. and Mitchell, N.D. (1991) Generalized additive models in plant ecology. Journal of Vegetation Science, 2, 587-602.

0.13.3 CTA - Classification Tree Analysis

This provides a good alternative to regression techniques. Like GAM, they do not rely on a priori hypotheses about the relationship between independent and dependent variables. This method consists of recursive partitions of the dimensional space defined by the predictors into groups that are as homogeneous as possible in terms of response. The tree is built by repeatedly splitting the data, defined by a simple rule based on a single explanatory variable. At each split, the data are partitioned into two exclusive groups, each of which is as homogeneous as possible. The algorithm seeks to decrease the variance within the subset as much as possible. The heterogeneity of a node can be interpreted as a deviance of a Gaussian model (regression tree) or of a multinomial model (classification tree). The result is a graph representing the deviance function of the cost-complexity parameter. The best tree is a trade-off between a high decrease of deviance and the smallest number of leaves. BIOMOD uses the rpart library to run the classification tree analysis. To control the length of the tree, the program builds a nested sequence of sub-trees by recursively snipping off the less important splits in terms of explained deviance. BIOMOD uses a procedure running X-fold cross-validations to select the best trade-off between the number of leaves of the tree and the explained deviance. The user can specify the number of cross-validation required.
If you want to use classification tree analysis model, just type Tree = TRUE. Then select the number of cross-validation typing CV.tree = 10.
There is no optimal number of cross-validation. Note that high number increases the memory demand.

Key reference.
Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J. (1984) Classification and regression trees Chapman and Hall, New York.

Key reference in ecology/biogeography.
De'Ath, G. and Fabricius, K.E. (2000) Classification and regression trees: a powerful yet simple technique for ecological data analysis. Ecology, 81, 3178-3192.
Thuiller, W., Vaydera, J., Pino, J., Sabaté, S., Lavorel, S., and Gracia, C. (2003) Large-scale environmental correlates of forest tree distributions in Catalonia (NE Spain). Global Ecology and Biogeography, 12, 313-325.
Vayssières, M.P., Plant, R.E., and Allen-Diaz, B.H. (2000) Classification trees: an alternative nonparametric approach for predicting species distributions. Journal of Vegetation Science, 11, 679-694.

0.13.4 ANN - Artifical Neural Networks

Feed forward neural networks provide a flexible way to generalize linear regression functions. They are non-linear regression models but with so many parameters that they are extremely flexible; flexible enough to approximate any smooth function. The accuracy of ANN is mainly controlled by two parameters: the amount of weight decay and the number of hidden unit. BIOMOD uses the library nnet. As different runs can provide different results, the best amount of weight decay and the number of units in the hidden layer [either equals to the number of variables (see Wierenga et Kluytmans, 1994) or 75by using N-fold cross-validation (3 by default). The user can also select the number of cross-validation. Note than ANN is very time-consuming so avoid excessive number of cross-validations.

If you want to use ANN model, simply type ANN = T. Then select the number of cross-validation typing CV.ann = 3.

Key reference.

Ripley, B.D. (1996) Pattern Recognition and Neural Networks Cambridge.

Key references in ecology/biogeography

Lek, S., Delacoste, M., Baran, P., Dimopoulos, I., Lauga, J., and Aulagnier, S. (1996) Application of neural networks to modelling nonlinear relationships in ecology. Ecological Modelling, 90, 39-52.
Luoto, M. and Hjort, J. (2005) Evaluation of current statistical approaches for predictive geomorphological mapping. Geomorphology, 67, 299-315.
Moisen, G.G. and Frescino, T.S. (2002) Comparing five modelling techniques for predicting forest characteristics. Ecological Modelling, 157, 209-225.
Pearson, R.G., Dawson, T.P., Berry, P.M., and Harrison, P.A. (2002) SPECIES: A Spatial Evaluation of Climate Impact on the Envelope of Species. Ecological Modelling, 154, 289-300.
Segurado, P. and Araújo, M.B. (2004) Evaluation of methods for modelling species probabilities of occurrence. Journal of Biogeography, 31, 1555-1568.

0.13.5 MDA - Mixture Discriminant Analysis

MDA is a method for classification (supervised) based on mixture models. It is an extension of the well-known linear discriminant analysis. The mixture of normals is used to obtain a density of estimation for each class. MDA has an implementation in the library mda. Very often, a single Gaussian to model a class, as in LDA, is too restricted. MDA extends to a mixture of Gaussians. Different regression methods can be used in the optimal scaling process. R-BIOMOD used mars (see below) to increase the predictive power of the models.

Key reference.

Hastie, T., Tibshirani, R and Buja, A. (1994) Flexible Disriminant Analysis by Optimal Scoring, JASA, 1255-1270.
Hastie, T. J., Buja, A., and Tibshirani, R. (1995) Penalized Discriminant Analysis. Annals of Statistics.
Hastie, T. and Tibshirani, R. (1996) Discriminant Analysis by Gaussian Mixtures. JRSSB.

Key references in ecology/biogeography

Manel, D., Dias, J. M., Buckton, S. T. and Ormerod, S. J. (1999) Alternative methods for predicting species distribution: an illustration with Himalayan river birds. Journal of Applied Ecology. 36, 734-747.

0.13.6 MARS - Multivariate Adaptive Regression Splines

A major assumption of any linear process is that the coefficients are stable across all levels of the explanatory variables and, in the case of a time series model, across all time periods. The MARS model is a very useful method of analysis when it is suspected that the model's coefficients have different optimal values across different levels of the explanatory variables. There are many theoretical reasons consistent with this possibility occurring in many different applications including energy, finance, economics, social science, and manufacturing. The MARS approach introduced by Friedman (1991) will systematically identify and estimate a model whose coefficients differ based on the levels of the explanatory variables. The breakpoints or thresholds that define a change in a model coefficient is termed a spline knot and can be thought of similar to a piecewise regression. An advantage of the MARS approach is that the spline knots are determined automatically by the procedure. In addition, complex nonlinear interactions between variables can also be specified. The MARS procedure is particularly powerful in situations where there are large numbers of right-hand variables and low-order interaction effects. The equation switching model, in which the slope of the model suddenly changes for a given value of the X variable, is a special case of the MARS model. The MARS procedure can detect and fit models in situations where there are distinct breaks in the model, such as are found if there is a change in the underlying probability density function of the coefficients and where there are complex variable interactions.
R-BIOMOD uses the *mars* function from the mda library programmed by Trevor Hastie and Robert Tibshirani. MARS automatically selects the amount of smoothing required for each predictor as well as the interaction order of the predictors. It is considered a projection method where variable selection is not a concern but the maximum level of interaction needs to be determined. Taking a conservative approach, only two-level interactions are specified into R-BIOMOD (this could be changed easily)
There is no specific parameterisation to modify here. More experience user could have a look at the private functions.

Key reference.

J. Friedman, "Multivariate Additive Regression Splines". Annals of Statistics, 1991

Key references in ecology/biogeography

Elith, J., Graham, C.H., Anderson, R.P., Dudik, M., Ferrier, S., Guisan, A., Hijmans, R.J., Huettman, F., Leathwick, J.R., Lehmann, A., Li, J., Lohmann, L., Loiselle, B.A., Manion, G., Moritz, C., Nakamura, M., Nakazawa, Y., Overton, J.M., Peterson, A.T., Phillips, S., Richardson, K., Schachetti Pereira, R., Schapire, R.E., Soberón, J., Williams, S.E., Wisz, M., and Zimmermann, N.E. (2006) Novel methods improve predictions of species' distributions from occurrence data. Ecography, 29, 129-151.
Luoto, M. and Hjort, J. (2005) Evaluation of current statistical approaches for predictive geomorphological mapping. Geomorphology, 67, 299-315.
Moisen, G.G. and Frescino, T.S. (2002) Comparing five modelling techniques for predicting forest characteristics. Ecological Modelling, 157, 209-225.

0.13.7 GBM - Generalised Boosting Models (or boosting regression trees, BRT)

Explanation adapted from Greg Ridgeway

Boosting: basic explanations Whereas GLM seeks to fit the single most parsimonious model that best explains the relationship between species distribution and a set of ecological predictors, boosting methods fit a large number of relatively simple models whose predictions are then combined to give more robust estimates of the response. The algorithm used by BIOMOD is a boosted regression tree (BRT, Friedman 2001, Ridgeway 1999) where each of the individual models consists of a simple classification or regression trees, i.e. a rule based classifier that consists of recursive partitions of the dimensional space defined by the predictors into groups that are as homogeneous as possible in terms of response. The tree is built by repeatedly splitting the data, defined by a simple rule based on a single explanatory variable. At each split, the data are partitioned into two exclusive groups, each of which is as homogeneous as possible. Ordinary generalised linear models have the form: where the algorithm seeks to estimate the ?j throughout various optimisation procedures (often maximum likelihood estimation). Special cases of basis expansions like generalised additive models (GAM) have also been using the same form: where h(x) is a non parametric function (e.g. spline). These methods have so far fixed the hjs and then found ?j using standard techniques (e.g. ordinary least squares regression - OLS). Regression trees also have this form where the hjs are indicator functions indicating whether x falls into a particular "box" and ? is just the terminal node means. Regression trees do not preselect the hjs nor J, rather they are estimated iteratively through the recursive partitioning algorithm. GBM makes each hj take the form of a regression tree. They are fitted incrementally so that h1(x) is the single best tree, h2(x) is the best tree that predicts the residuals of h1(x), and so on (Friedman, et al. 2000). By this way, the BRT uses an iterative method for developing a final model progressively adding trees to the model, while re-weighting the data to emphasises cased poorly predicted by the previous trees.

In BIOMOD, the user has the possibility to set up the number of cross-validation to identify an optimal number of trees that maximises the ability of a model to make accurate predictions to new, independent sites while avoiding excessive model complexity. The user has also to define the maximum number of trees which are going to be fitted. There is no way to know a priori what is the best. Between 2000 and 5000 is a good compromise. More importantly, BRT allowed the estimation of the relative importance of each variable in the model. BIOMOD uses a permutation method, which randomly permutes each predictor variable independently, and computes the associated reduction in predictive performance.
For more details:
http://www.salford-systems.com/friedmankdd.php
www.i-pensieri.com/gregr/ ModernPrediction/L9boosting.pdf

R-BIOMOD uses the gbm library programmed by Greg Ridgeway. This package implements the generalized boosted modelling framework. This implementation closely follows Friedman's Gradient Boosting Machine (Friedman, 2001). The interaction depth and the learning rate are set-up to 4 and 0.001 respectively (but could be easily changed).

Key reference.

Friedman, J.H. (2001) Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29, 1189-1232.
Friedman, J.H., Hastie, T.J., and Tibshirani, R. (2000) Additive logistic regression: a statistical

view of boosting. Annals of Statistics, 28, 337-374.
Ridgeway, G. (1999) The state of boosting. Computing Science and Statistics, 31, 172-181.

Key references in ecology/biogeography

Elith, J., Graham, C. H., Anderson, R. P., Dudik, M., Ferrier, S., Guisan, A., Hijmans, R. J., Huettman, F., Leathwick, J. R., Lehmann, A., Li, J., Lohmann, L., Loiselle, B. A., Manion, G., Moritz, C., Nakamura, M., Nakazawa, Y., Overton, J. M., Peterson, A. T., Phillips, S., Richardson, K., Schachetti Pereira, R., Schapire, R. E., Soberón, J., Williams, S. E., Wisz, M. and Zimmermann, N. E. (2006) Novel methods improve predictions of species' distributions from occurrence data. Ecography. 29, 129-151.
Leathwick, J.R., Elith, J., Francis, M.P., Hastie, T.J., and Taylor, P. (2006) Variation in demersal fish species richness in the oceans surroundings New Zealand: an analysis using boosted regression trees. Marine Ecology Progress Series, In press.
Thuiller, W., Midgley, G.F., Rouget, M., and Cowling, R.M. (2006) Predicting patterns of plant species richness in megadiverse South Africa. Ecography, 29, 733-744

0.13.8 randomForest - Breiman and Cutler's random forest for classification and regression

The model randomForest implements Breiman's random forest algorithm (based on Breiman and Cutler's original Fortran code) for classification and regression. It is implemented into the "randomForest" library programmed by Andy Liaw and Matthew Wiener.
Random Forests grows many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification, and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).
Each tree is grown as follows:
If the number of cases in the training set is N, sample N cases at random - but with replacement, from the original data. This sample will be the training set for growing the tree. If there are M input variables, a number $m << M$ is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing. Each tree is grown to the largest extent possible. There is no pruning.
In the original paper on random forests, it was shown that the forest error rate depends on two things:
- The correlation between any two trees in the forest. Increasing the correlation increases the forest error rate.
- The strength of each individual tree in the forest. A tree with a low error rate is a strong classifier. Increasing the strength of the individual trees decreases the forest error rate.
Reducing m reduces both the correlation and the strength. Increasing it increases both. Somewhere in between is an "optimal" range of m - usually quite wide. Using the oob error rate (see below) a value of m in the range can quickly be found. This is the only adjustable parameter to which random forests is somewhat sensitive.

Features of Random Forests.
It runs efficiently on large data bases.
It can handle thousands of input variables without variable deletion.
It gives estimates of what variables are important in the classification.
It generates an internal unbiased estimate of the generalization error as the forest building progresses.
It has methods for balancing error in class population unbalanced data sets.
It offers an experimental method for detecting variable interactions.

How random forests work. To understand and use the various options, further information about how they are computed is useful. Most of the options depend on two data objects generated by random forests. When the training set for the current tree is drawn by sampling with replacement, about one-third of the cases are left out of the sample. This oob (out-of-bag) data is used to get a running unbiased estimate of the classification error as trees are added to the forest. It is also used to get estimates of variable importance.

The out-of-bag (oob) error estimate In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error. It is estimated internally, during the run, as follows: Each tree is constructed using a different bootstrap sample from the original data. About one-third of the cases are left out of the bootstrap sample and not used in the construction of the kth tree. Put each case left out in the construction of the kth tree down the kth tree to get a classification. In this way, a test set classification is obtained for each case in about

one-third of the trees. At the end of the run, take j to be the class that got most of the votes every time case n was oob. The proportion of times that j is not equal to the true class of n averaged over all cases is the oob error estimate.

Variable importance In every tree grown in the forest, put down the oob cases and count the number of votes cast for the correct class. Now randomly permute the values of variable m in the oob cases and put these cases down the tree. Subtract the number of votes for the correct class in the variable-m-permuted oob data from the number of votes for the correct class in the untouched oob data. The average of this number over all trees in the forest is the raw importance score for variable m. If the values of this score from tree to tree are independent, then the standard error can be computed by a standard computation. The correlations of these scores between trees have been computed for a number of data sets and proved to be quite low, therefore we compute standard errors in the classical way, divide the raw score by its standard error to get a z-score, ands assign a significance level to the z-score assuming normality. For each case, consider all the trees for which it is oob. Subtract the percentage of votes for the correct class in the variable-m-permuted oob data from the percentage of votes for the correct class in the untouched oob data.

R-BIOMOD uses 500 trees (this could be changed easily in BIOMOD-R Private Functions 200X.XX.XX.R) and extracts the importance of each selected variable.

Key References. Breiman, L. (2001), Random Forests, Machine Learning 45(1), 5-32. Breiman, L (2002), "Manual On Setting Up, Using, And Understanding Random Forests V3.1.

Key References in ecology/biogeography.

Elith, J., Graham, C.H., Anderson, R.P., Dudik, M., Ferrier, S., Guisan, A., Hijmans, R.J., Huettman, F., Leathwick, J.R., Lehmann, A., Li, J., Lohmann, L., Loiselle, B.A., Manion, G., Moritz, C., Nakamura, M., Nakazawa, Y., Overton, J.M., Peterson, A.T., Phillips, S., Richardson, K., Schachetti Pereira, R., Schapire, R.E., Soberón, J., Williams, S.E., Wisz, M., and Zimmermann, N.E. (2006) Novel methods improve predictions of species' distributions from occurrence data. Ecography, 29, 129-151.

Prasad, A.M., Iverson, L.R., and Liaw, A. (2006) Newer classification and regression tree techniques: bagging and random forests for ecological prediction. Ecosystems, 9, 181-199.

0.13.9 SRE - Surface Range Envelops

This is a simple surface range envelop, similar to BioClim. The envelop is defined by identifying maximum and minimum values for each input variable from the set of sites containing an observed species' presence. Any site with all variables falling between these maximum and minimum limits is included within the range. This is the simplest method to model the distribution of species or biomes. The Perc025 and Perc05 allow specifying a broad percentile range (2.5-97.5 % or 5-95 % respectively) based on the chosen predictors. It allows removing the extreme presence (those who are close to be outside the envelop) which might be considered as outliers.

Key reference.
Busby JR (1991) BIOCLIM - a bioclimate analysis and prediction system. In: Margules CR, Austin MP, editors. Nature Conservation: Cost Effective Biological Surveys and Data Analysis. Canberra, Australia: CSIRO. pp. 64-68.

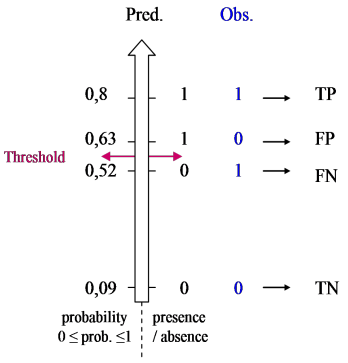
Key References in ecology/biogeography.
Beaumont LJ and Hughes L (2002) Potential changes in the distribution of latitudinally restricted Australian butterfly species in response to climate change. Global Change Biology 8:954-971.

0.14 Predictive performance description

BIOMOD proposes three different evaluation procedures, namely the ROC curve, the True Skill Statistic and the Kappa statistic. Any of them can be used independently but it is advisable to run them all for cross-comparisons.

The accuracy of statistical models is often assessed by studying the agreement between observation and prediction using a confusion matrix (see below). Four fractions can be deduced from this matrix.
- sensitivity (true positive fraction).
- specificity (true negative fraction).
- false positive fraction.
- false negative fraction.

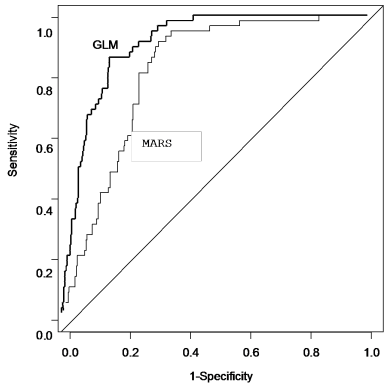
Sensitivity can be described as the ratio of positive sites (presence) correctly predicted over the number of positive sites in the sample. Specificity is the ratio of negatives sites (absence) correctly predicted over the number of negative sites in the sample. False positive and false negative fractions equal 1-specificity and 1-sensitivity respectively. To generate such a matrix and because a very large fraction of the existing models produce predictions as a probability of presence, a probability threshold must be decided to differentiate between a site (or cell) predicted to be occupied and a site (or cell) predicted to be unoccupied.



		Predicted		
		Absence	Presence	Total
Observed	Absence	TN	FP	TN + FP
	Presence	FN	TP	FN + TP
	Total	TN + FN	FP + TP	

BIOMOD is able to compute three different approaches.

Relative Operating Characteristic curve (ROC curve): This is not dependent on the threshold. The ROC curve is a graphical method representing the relationship between the False Positive fraction (1-specificity) and the sensitivity for a range of thresholds. If all predictions were possibly expected by chance, the relation would be a 45ř line. Good model performance is characterised by a curve that maximises sensitivity for low values of (1-specificity), i.e. when the curve passes close to the upper left corner of the plot. The area between the 45ř line and the curve measures discrimination, that is, the ability of the model to correctly classify a species as present or absent in a given plot. This measure is therefore called the area under the curve (AUC). In the example below, the GLM will show a better score than the MARS and is expected to be more reliable.



Cohen's Kappa statistic: This measure expresses the agreement not obtained randomly between two qualitative variables (of which a binary variable is a particular case). Kappa is based on the misclassification matrix which necessitates the calculation of a probability threshold. To do that, BIOMOD calculated Kappa for all thresholds between zero to one. The greatest value was kept as the best Kappa value. This measure expresses the best possible agreement.

The Hanssen-Kuiper Skill Score (KSS) or True Skill Statistic (TSS): This statistic, traditionally used for assessing the accuracy of weather forecasts compares the number of correct forecasts, minus those attributable to random guessing, to that of a hypothetical set of perfect forecasts.
For a 2x2 confusion matrix TSS is defined as:
TSS = sensitivity + specificity - 1

Like kappa, TSS takes into account both omission and commission errors, and success as a result of random guessing, and ranges from -1 to +1, where +1 indicates perfect agreement and values of zero or less indicate a performance no better than random. However, in contrast to kappa, TSS is not affected by prevalence. It can also be seen that TSS is not affected by the size of the validation set, and that two methods of equal performance have equal TSS scores. TSS is a special case of kappa, given that the proportions of presences and absences in the validation set are equal.

Index for classifying model prediction accuracy.

Accuracy	AUC	Kappa/TSS
Excellent or high	0.9 – 1	0.8 – 1
Good	0.8 – 0.9	0.6 – 0.8
Fair	0.7 – 0.8	0.4 – 0.6
Poor	0.6 – 0.7	0.2 – 0.4
Fail or null	0.5 – 0.6	0 – 0.2