Multi-species distribution modelling with `biomod2`

`biomod2` version : 1.0
R version 2.15.1 (2012-06-22)

Damien Georges & Wilfried Thuiller

July 5, 2012

Contents

1 Introduction

This vignette illustrates how to make a multi-species modelling using `biomod2`. Almost all `biomod2` functions are built to work with monospecific species. We decided to built it like that to facilitate parallelized computation.
In this document we are going to show you that multi-species computation has advantages and is not difficult to do.

Classical multi-species computation and a short example of parallelised one will be present here.

NOTE 1 :
We plane to develop in the near future some tools to help users to deal with these multi-species modelling

2 Multiple Species Comupation

Multiple species functions are not implemented yet. You thus have to create a 'loop' around Initialisation, Modeling and optionally EnsembleModeling, Projection, EnsembleForecasting ... The best way depends on your input data class. Let's have an example with `biomod2` data.

2.1 Loading the data

First, we are going to load our species occurances data. Here, species occurances have been previously extracted and stored as table (.csv format).

```
R input
# 1. loading species occurances data
library(biomod2)
mySpeciesOcc <- read.csv( system.file(
                          "external/species/species_occ.csv",
                          package="biomod2"))
head(mySpeciesOcc)
```

```
R output
       x     y CapraIbex MelesMeles MyocastorCoypus
1 21.667 68.33        NA          1              NA
2 25.000 68.33        NA          1              NA
3 28.333 68.33        NA          1              NA
4 31.667 68.33        NA          1              NA
5  8.333 65.00        NA          1              NA
6 11.667 65.00        NA          1              NA
```

NOTE 2 :
The loaded table contains only 1 and NA's. Later, we will conider only the occurances (i.e 1) for selecting pseudo-absences into our explanatory variables (RasterStack). More details about input data supported in the 'SupportedInputData' vignette.

In this example, our explanatory variables are stored in a RasterStack.

```
R input
# 2. loading environemental data

# Environmental variables extracted from Worldclim (bio_3, bio_4,
# bio_7, bio_11 & bio_12)
require(raster)
myExpl = stack( system.file( "external/climat/current/bio3.grd",
                             package="biomod2"),
                system.file( "external/climat/current/bio4.grd",
                             package="biomod2"),
                system.file( "external/climat/current/bio7.grd",
                             package="biomod2"),
                system.file( "external/climat/current/bio11.grd",
                             package="biomod2"),
                system.file( "external/climat/current/bio12.grd",
                             package="biomod2"))
```

We are going to modelised 2 species niches caracterised by their names :

- MelesMeles
- MyocastorCoypus

For each species, we are going to sequencialy :

1. Selecting data correspunding to a species
2. Putting this data in the `biomod2` format and selecting some pseudo-absences (i.e. BIOMOD_FormatingData)
3. Building 'individual models' (i.e.v BIOMOD_Modeling)
4. Building ensemble-models (i.e. BIOMOD_EnsembleModeling)
5. Making models projections (i.e. BIOMOD_Projection BIOMOD_EnsembleForecasting)

NOTE 3 :
The modelling steps applied to each species is the same than which is described in 'GettingStarted' vignette.

R input
```
# define the species you wanted names
sp.names <- c("MelesMeles", "MyocastorCoypus")
# loop on species == aplying the sames functions to each species
for(sp.n in sp.names){

   cat('\n',sp.n,'modelling...')
   ### definition of data for this run
   ## i.e keep only the column of our species
   myResp <- as.numeric(mySpeciesOcc[,sp.n])
   # get NAs id
   na.id <- which(is.na(myResp))
   # remove NAs to enforce PA sampling to be done on explanatory rasters
   myResp <- myResp[-na.id]

   myRespCoord = mySpeciesOcc[-na.id,c('x','y')]

   myRespName = sp.n

   ### Initialisation
```

```
myBiomodData <- BIOMOD_FormatingData(resp.var = myResp,
                                     expl.var = myExpl,
                                     resp.xy = myRespCoord,
                                     resp.name = myRespName,
                                     PA.nb.rep = 2,
                                     PA.nb.absences = 10*sum(myResp==1,
                                                        na.rm=TRUE),
                                     PA.strategy = 'random')


### Options definition
myBiomodOption <- BIOMOD_ModelingOptions()

### Modelling
myBiomodModelOut <- BIOMOD_Modeling(
                           myBiomodData,
                           models = c('SRE','CTA','RF','MARS','FDA'),
                           models.options = myBiomodOption,
                           NbRunEval=1,
                           DataSplit=80,
                           Yweights=NULL,
                           VarImport=3,
                           models.eval.meth = c('TSS','ROC'),
                           SaveObj = TRUE,
                           rescal.all.models = TRUE)

### Building ensemble-models
myBiomodEM <- BIOMOD_EnsembleModeling(
                     modeling.output = myBiomodModelOut,
                     chosen.models = 'all',
                     eval.metric = c('TSS'),
                     eval.metric.quality.threshold = c(0.85),
                     prob.mean = T,
                     prob.cv = T,
                     prob.ci = T,
                     prob.ci.alpha = 0.05,
                     prob.median = T,
                     committee.averaging = T,
                     prob.mean.weight = T,
                     prob.mean.weight.decay = 'proportional' )

### Do projections on current varaiable
myBiomomodProj <- BIOMOD_Projection(
                         modeling.output = myBiomodModelOut,
                         new.env = myExpl,
                         proj.name = 'current',
                         selected.models = 'all',
                         binary.meth= 'ROC',
                         compress = 'xz',
                                    clamping.mask = F)

    ### Do ensemble-models projections on current varaiable
    myBiomodEF <- BIOMOD_EnsembleForecasting(
                           projection.output = myBiomomodProj,
                           EM.output = myBiomodEM,
                           binary.meth = 'TSS',
                           total.consensus = TRUE)
  }
```

A folder by species was created in your working diractory. You are now able to work on created output and combine them as you want.
To illustrate it let's create an α-diversity map (which is in fact just a sum of binary maps). We choose to considered the binaries of 'total consensus ensemble-models projections' and focus our attention on 'mean of probability' ensemble-models.

R input

```
# load the first speces binary maps which will define the mask
alphaMap <- get(load(paste(sp.names[1],"/proj_current/",
                            sp.names[1],"_TotalConsensus.bin.TSS",
                            sep="")))[[1]]
# free space
rm(list=paste(sp.names[1],"_TotalConsensus.bin.TSS", sep=""))
# # add all other species map
for(sp.n in sp.names[-1]){
   # add layer
   alphaMap <- alphaMap + get(load(paste(sp.n,"/proj_current/",
                                           sp.n,"_TotalConsensus.bin.TSS",
                                           sep="")))[[1]]
   # free space
   rm(list=paste(sp.n,"_TotalConsensus.bin.TSS", sep=""))
 }
# summary of created raster
alphaMap
```

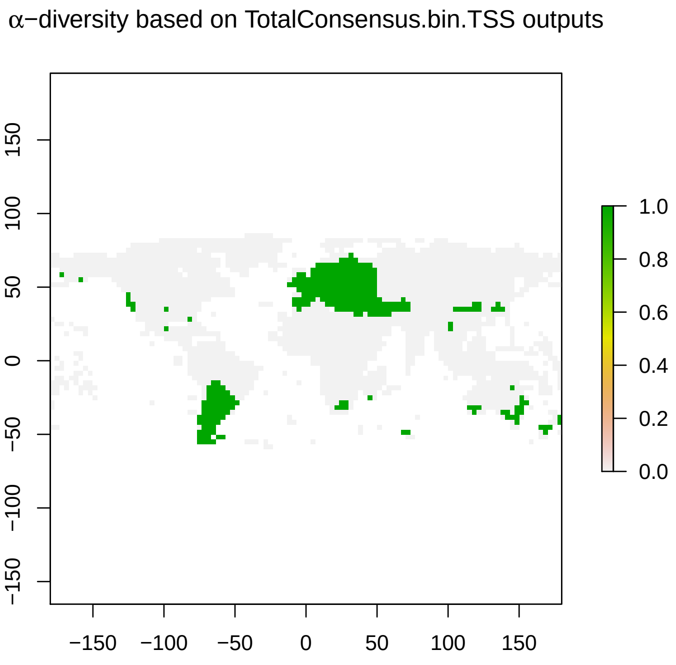
R output

```
class       : RasterLayer
dimensions  : 45, 108, 4860  (nrow, ncol, ncell)
resolution  : 3.333, 3.333  (x, y)
extent      : -180, 180, -60, 90  (xmin, xmax, ymin, ymax)
coord. ref. : +proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs +towgs84=0,0,0
values      : in memory
min value   : 0
max value   : 1
```

R input
```
```

Lets visualise our α-diversity map.

R input
```
plot(alphaMap, main = expression( paste(alpha, "-diversity based on",
                                  " TotalConsensus.bin.TSS outputs")))
```



NOTE 4 :
Here our two species seems to never coexist, that's why the α-diversity is never higher than 1.

3 How to saving computation time?

If you have a lot of species to study or\$andifyouworkwithhugedataset, computationtimemayrapidlye
Thebestwaytodealwiththiscasesistomakeaparallelisation.Itmeanssplitingspeciesinseveralsmallgr
Inthebestcaseyouhaveanaccesstoclusterofcomputers.Ifnot, youcanhaveseveralcomputersandsplity
CPUsinglecomputer.TheSnowFallpackagewillhelpyoutodothis.$

NOTE 5 :
Parallelizing tools are developping fastly in R so whaterver type of paralleli-

sation you want to do, tools may exists to help you!

4 Introduction to SnowFall library

The snowfall package is usefull to parallelise R jobs. The syntax is almost the same than an 'apply' one.

The first thing to do is to download snowfold library

R input
```
install.packages('snowfall', dependencies=TRUE)
```

R input
```
library(snowfall)
```

The second thing to do is to create the function you are going to parallelise (let's take the same than in the previous section). Here we just put all the loop into MyBiomodSF function.

R input
```
MyBiomodSF <- function(sp.n){

   cat('\n',sp.n,'modelling...')
   ### definition of data for this run
   ## i.e keep only the column of our species
   myResp <- as.numeric(mySpeciesOcc[,sp.n])
   # get NAs id
   na.id <- which(is.na(myResp))
   # remove NAs to enforce PA sampling to be done on explanatory rasters
   myResp <- myResp[-na.id]

   myRespCoord = mySpeciesOcc[-na.id,c('x','y')]

   myRespName = sp.n

   ### Initialisation
   myBiomodData <- BIOMOD_FormatingData(resp.var = myResp,
                                        expl.var = myExpl,
                                        resp.xy = myRespCoord,
                                        resp.name = myRespName,
                                        PA.nb.rep = 2,
                                        PA.nb.absences = 10*sum(myResp==1,
                                                                na.rm=TRUE),
                                        PA.strategy = 'random')


   ### Options definition
```

```
    myBiomodOption <- BIOMOD_ModelingOptions()

    ### Modelling
    myBiomodModelOut <- BIOMOD_Modeling(
                              myBiomodData,
                              models = c('SRE','CTA','RF','MARS','FDA'),
                              models.options = myBiomodOption,
                              NbRunEval=1,
                              DataSplit=80,
                              Yweights=NULL,
                              VarImport=3,
                              models.eval.meth = c('TSS','ROC'),
                              SaveObj = TRUE,
                              rescal.all.models = TRUE)

    ### Building ensemble-models
    myBiomodEM <- BIOMOD_EnsembleModeling(
                        modeling.output = myBiomodModelOut,
                        chosen.models = 'all',
                        eval.metric = c('TSS'),
                        eval.metric.quality.threshold = c(0.85),
                        prob.mean = T,
                        prob.cv = T,
                        prob.ci = T,
                        prob.ci.alpha = 0.05,
                        prob.median = T,
                        committee.averaging = T,
                        prob.mean.weight = T,
                        prob.mean.weight.decay = 'proportional' )

    ### Do projections on current varaiable
    myBiomomodProj <- BIOMOD_Projection(
                            modeling.output = myBiomodModelOut,
                            new.env = myExpl,
                            proj.name = 'current',
                            selected.models = 'all',
                            binary.meth= 'ROC',
                            compress = 'xz',
                            clamping.mask = F)

    ### Do ensemble-models projections on current varaiable
    myBiomodEF <- BIOMOD_EnsembleForecasting(
                         projection.output = myBiomomodProj,
                         EM.output = myBiomodEM,
                         binary.meth = 'TSS',
                         total.consensus = TRUE)

  }
```

Then you have to give to snowfall all the variables and libraries that are used in the function.

R input

```
## Init snowfall
library(snowfall)
sfInit(parallel=TRUE, cpus=2 )
## Export packages
sfLibrary('biomod2', character.only=TRUE)
## Export variables
sfExport('mySpeciesOcc')
sfExport('myExpl')
sfExport('sp.names')
# you may also use sfExportAll() to exprt all your workspace variables


## Do the run
mySFModelsOut <- sfLapply( sp.names, MyBiomodSF)
## stop snowfall
sfStop( nostop=FALSE )
```

You may obtain exacly the same folder structure than if you worked in serial so you can compute your 'Comunities' analyses axactly as previously.
NOTE 6 :
In case of computation on several computers, you have to merge all your species folders in a lone folder, then you will get the same files than in a serial computation case.