

mulTree manual

Thomas Guillerme

January 21, 2016

This package is based on the MCMCglmm package and runs a MCMCglmm analysis on multiple trees.

1 Installation

You can install the latest released version (1.2) directly from GitHub using the following:

```
if(!require(devtools)) install.packages("devtools")
install_github("TGuillerme/mulTree", ref = "release")
```

2 Quick go through

Note that this section will be developed in the future. Stay tuned!

```
## Loading the package
library(mulTree)

## Loading required package: caper
## Loading required package: MASS
## Loading required package: mvtnorm
## Loading required package: coda
## Loading required package: hdbcde
## hdbcde 3.1 loaded
## Loading required package: MCMCglmm
## Loading required package: Matrix

## Loading the lifespan data
data(lifespan)
```

2.1 Combining phylogenies: tree.bind

This function allows to combine phylogenies, in this example, we are going to combine the `trees_mammalia` and the `trees_aves` randomly at a root.age of 250 million years ago.

```
## The 2 mammalian trees
trees_mammalia

## 2 phylogenetic trees

## Number of tips in both
unlist(lapply(trees_mammalia, Ntip))
```

```
## [1] 134 134

## The 2 aves trees
trees_aves

## 2 phylogenetic trees

## Number of tips in both
unlist(lapply(trees_aves, Ntip))

## [1] 58 58

## Combining them
combined_trees <- tree.bind(trees_mammalia, trees_aves, sample = 2,
                             root.age = 250)
combined_trees

## 2 phylogenetic trees

## Number of tips in both
unlist(lapply(combined_trees, Ntip))

## [1] 192 192
```

2.2 Preparing the mulTree data: as.mulTree

This function allows to combine the trees to some trait data and get it in the right format to be passed to the mulTree function.

```
## The trait data
head(lifespan_volant)

##           species      class longevity      mass    volant
## 1 Dolichotis_patagonum Mammalia -0.1490041  1.0875446 nonvolant
## 2      Eidolon_helvum Mammalia  0.4686111 -0.2748337  volant
## 3      Elephas_maximus Mammalia  2.1071286  3.1220340 nonvolant
## 4      Equus_asinus Mammalia  1.6128024  2.0352764 nonvolant
## 5      Equus_burchellii Mammalia  1.2962194  2.2295299 nonvolant
## 6      Equus_caballus Mammalia  1.9001076  2.2548716 nonvolant

## Creating the mulTree object
mulTree_data <- as.mulTree(data = lifespan_volant, tree = combined_trees,
                           taxa = "species")

## This object is classified as "mulTree" and contains different elements to be
## passed to the mulTree function
class(mulTree_data) ; names(mulTree_data)

## [1] "mulTree"
## [1] "phy"          "data"          "random.terms" "taxa.column"
```

2.3 Running the MCMCglmm on multiple trees: mulTree

This function intakes the normal arguments form the MCMCglmm function alongside with the mulTree object.

WARNING: this part of the code does currently not run on knitr and is therefore skipped. However, this part works normally if copy / pasted into the R console. Be warned however, that this part of the code can take several minutes to run!

```
## The glmm formula
my_formula <- longevity ~ mass + volant

## The MCMC parameters (generations, sampling, burnin)
my_parameters <- c(100000, 10, 1000)

## The MCMCglmm priors
my_priors <- list(R = list(V = 1/2, nu = 0.002),
  G = list(G1 = list(V = 1/2, nu = 0.002)))

## Running the MCMCglmm on multiple trees
mulTree(mulTree.data = mulTree_data, formula = my_formula, priors = my_priors,
  parameters = my_parameters, output = "longevity_example", ESS = 50,
  chains = 2)
```

2.4 Reading the models: read.mulTree

The models were written out of the R environment in your current directory. To reinput them in the R environment and analysis the results, we can use the read.mulTree function.

WARNING: this part of the code does currently not run on knitr and is therefore skipped. However, this part works normally if copy / pasted into the R console.

```
## Reading only one specific model
one_model <- read.mulTree("longevity_example-tree1_chain1", model = TRUE)
## This model is a normal MCMCglmm object that has been ran on one single tree
class(one_model) ; names(one_model)

## Reading the convergence diagnosis test to see if the two chains converged for
## each tree
read.mulTree("longevity_example", convergence = TRUE)
## As indicated here, the chains converged for both chains!

## Reading all the models to perform the MCMCglmm analysis on multiple trees
all_models <- read.mulTree("longevity_example")
str(all_models)
## This object contains 39600 estimations of the Intercept and the terms!

## Removing the chains from the current directory
file.remove(list.files(pattern="longevity_example"))

## To temporarily remedy the problem with knitr described above we can load pre
## calculated MCMCglmm.
data(lifespan.mcmc)

## NOTE HOWEVER THAT IF YOU ARE RUNNING THE CODE OF THIS VIGNETTE IN THE R
## CONSOLE, YOU WON'T NEED THIS STEP!

## Summarizing all the chains
all_models <- lifespan.mcmc
```

2.5 Summarising the results: summary.mulTree

It is possible to summarise the results of the glmm on all chains using the `summary.mulTree` function.

```
## Summarising the results by estimating the highest density regions
## and their associated 95 and 50 confidence intervals (default)
summarised_results <- summary(all_models)
summarised_results

##           Estimates(mode hdr) lower.CI(2.5) lower.CI(25) upper.CI(75) upper.CI(97.5)
## Intercept      -0.06999296   -1.13207196  -0.44486942   0.29070469   1.01296766
## mass           0.51833987    0.38949753   0.47446665   0.56227442   0.64498855
## volancy        0.98967086    0.39699309   0.78557528   1.18069787   1.55656195
## phy.var        0.91505855    0.62186635   0.81016578   1.03538270   1.28537645
## res.var        0.04438445    0.02238516   0.03521298   0.05303724   0.07522887
## attr("class")
## [1] "matrix" "mulTree"

## Summarising the results using the quick 'n' dirty way along with some options
## i.e just measuring the distributions quantiles
## note that there is a S3 method for "mulTree" objects allowing to just use
## summary()
summary(all_models, use.hdr = FALSE, cent.tend = mean, prob = c(75, 25))

##           Estimates(mean) lower.CI(12.5) lower.CI(37.5) upper.CI(62.5) upper.CI(87.5)
## Intercept      -0.06726806   -0.69609640   -0.24328020   0.10553401   0.56463730
## mass           0.51757042    0.44244907   0.49679614   0.53879559   0.59259546
## volancy        0.97726166    0.63844718   0.88624230   1.07205171   1.31429889
## phy.var        0.94630374    0.75547035   0.88405672   0.99072104   1.14261125
## res.var        0.04758039    0.03238856   0.04198954   0.05061299   0.06364383
## attr("class")
## [1] "matrix" "mulTree"
```

2.6 Visualising the results: plot.mulTree

Finally it is possible to simply plot the results from the MCMCglmm analysis on multiple trees using the S3 method for `plot.mulTree`. Here are two examples:

```
## Graphical options
quartz(width = 10, height = 5) ; par(mfrow = (c(1,2)), bty = "n")

## Plotting using the default options
plot(summarised_results)

## Plotting using some more pretty options
plot(summarised_results, horizontal = TRUE, ylab = "", cex.coef = 0.8,
      main = "Posterior distributions", ylim = c(-2,2), cex.terms = 0.5,
      terms = c("Intercept", "Body Mass", "Volancy", "Phylogeny", "Residuals"),
      col = "red", cex.main = 0.8)
abline(v = 0, lty = 3)
```

