Package 'HyPhy'

February 19, 2015

Type Package

Title Macroevolutionary phylogentic analysis of species trees and gene trees

Version 1.0

Date 2012-07-25

Author Nathaniel Malachi Hallinan

Maintainer Nathaniel Malachi Hallinan <nmhallinan@gmail.com>

Depends ape, R.utils

Description A Bay Area high level phylogenetic analysis package mostly using the birth-death process. Analysis of species tree branching times and simulation of species trees under a number of different time variable birth-death processes. Analysis of gene tree species tree reconciliations and simulations of gene trees in species trees.

License GPL-2

Repository CRAN

Date/Publication 2012-07-30 04:09:01

NeedsCompilation no

R topics documented:

HyPhy-package *Macroevolutionary phylogentic analysis of species trees and gene trees*

Description

A Bay Area high level phylogenetic analysis package mostly using the birth-death process. Analysis of species tree branching times and simulation of species trees under a number of different time variable birth-death processes. Analysis of gene tree species tree reconciliations and simulations of gene trees in species trees under the birth-death process.

Details

Package:	HyPhy
Type:	Package
Version:	1.0
Date:	2012-07-25
License:	GPL-2

Author(s)

Nathaniel Malachi Hallinan <nmhallinan@gmail.com>

Internal R functions *Internal Functions*

Description

These functions are used by other functions in the HyPhy package and not by users

Usage

```
get.recon.1(phy, phy.sub, reconcile)
```

Arguments

phy	The species tree to be reconciled of class "phylo"
phy.sub	The gene tree(s) to be reconciled in class "phylo" or "multiPhylo"
reconcile	A description of the relationship between the gene tree and the species tree: either NULL; a vector of positive integers of length length(phy.sub$tip.label); or a vector of integers and NA of length max(phy.sub$edge). (See details)

Author(s)

Nathaniel Malachi Hallinan

`recon.score` *Duplications and losses for gene tree in a species tree*

Description

Calculates the minimum number of gene duplications and gene losses necessary to reconcile a gene tree with a species tree

Usage

```
recon.score(phy, phy.sub, reconcile = NULL)
```

Arguments

`phy`	The species tree to be reconciled of class "phylo"
`phy.sub`	The gene tree(s) to be reconciled in class "phylo" or "multiPhylo"
`reconcile`	A description of the relationship between the gene tree and the species tree: either `NULL`; a vector of positive integers of length `length(phy.sub$tip.label)`; or a vector of integers and NA of length `max(phy.sub$edge)`. (See details)

Details

`reconcile` is a vector in which the *ith* element of the vector gives the position on `phy` of the node of the gene tree labeled *i* in `phy$sub.edge`. A positive *n* value places the *ith* `ph.sub` node on the node of the species tree labeled *n* in `phy$edge`, a negative *n* value places the *ith* `ph.sub` node on the branch of the species tree defined by the *nth* row in `phy$edge`, and a 0 places the *ith* `ph.sub` node on the root of the species tree. Elements for which `is.na(reconcile[i])` or `length(reconcile)<i` are undefined, so if `reconcile=NULL`, then all elements are undefined. The first `length(phy.sub$tip.label)` elements of `reconcile` refer to the tips of `phy.sub`; if the *ith* element is undefined, then the function will place the *ith* tip of `phy.sub` in a tip of `phy`, such that `phy$tip.label==phy.sub$tip.label[i]`. The remaining elements of `reconcile` refer to the internal nodes of `phy.sub`; if the *ith* element is undefined, then the function will place the *ith* node of `phy.sub` at its maximum parsimony position given the position of the two nodes above it.

To make a long story short, if the labels in `phy.sub$tip.label` match the labels in `phy$tip.label`, as they would for trees produced by `rgenetree`, and you want a maximum parsimony reconciliation, then set `reconcile=NULL`. On the other hand, if the labels do not match and you want a maximum parsimony reconciliation, then `reconcile` should be a vector of positive integers assigning the tips of `phy.sub` to the tips of `phy`. Finally, if you want a non maximum parsimony reconciliation, then `reconcile` should be of length `max(phy$edge)`, and the nodes of `sub.phy` should be assigned to branches of `phy` as described above. Be warned that any internal node of `sub.phy` can only be assigned to certain nodes and branches of `phy`, and there is no mechanism to check the logical consistancy of `reconcile`, so make sure that you assign nodes to appropriate branches. You can always leave a node as `NA`, and it will end up in its maximum parsimony position.

Value

If `class(phy.sub)=="phylo"`, the output is a vector with two elements, the number of duplications and the number of losses. If `class(phy.sub)=="multiPhylo"`, the output is a matrix with two colums, the number of duplications and the number of losses, and every row representing a different tree in `phy.sub`..

Author(s)

Nathaniel Malachi Hallinan

References

M. Goodman, J. Czelusniak, G. Moore, A. Romero-Herrera, G. Matsuda, Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences, Syst. Zool. 28 (1979) 132-163.

See Also

`rgenetree`,`plot.recon`(coming soon)

Examples

```
##First we need a simple species tree
spec<-read.tree(text="((A:0.5,B:0.5):0.5,C:1);")
##Now let's simulate a bunch of gene family trees
genes<-rgenetree(10,spec,0.5,0.5,3,10,TRUE)
##Let's look at those trees
##Note that all their tips are labeled A, B or C, just like spec
plot(genes)
##Therefore we can calculate the counts for all trees without any other info
recon.score(spec,genes)
##On the other hand, if we make our own gene tree with different labels
gene<-read.tree(text="((A1,(A2,B1)),(B2,(C1,C2)));")
##We must generate a reconcile vector
##to do so we must know the positions of the tip labels in both phylogenies
spec$tip.label
gene$tip.label
reconcile<-c(1,1,2,2,3,3)
recon.score(spec,gene,reconcile)
##To force the node at the base of A2 and B1 down to the branch below A and B
##we must look at both edge matrices to learn how each node and branch are labeled
spec$edge
gene$edge
reconcile<-c(1,1,2,2,3,3,NA,NA,-1,NA,NA)
recon.score(spec,gene,reconcile)
```

`requiprobable` *Sample trees from an equiprobable labeled, semi-labeled or unlabeled distribution*

Description

An equiprobable distribution assumes that all tree topologies are equally likely. The degree of tip labeling affects the number of possible tree topologies. In labeled trees all the tips are different, in unlabeled trees all the tips are the same and in semi-labeled trees the tips fall into groups in which any individual tip may or may not be the same as any other.

Usage

```
requiprobable(n, labels, label.counts = rep(1, length(labels)))
```

Arguments

`n`	An integer. The number of trees to sample.
`labels`	A character vector. The different labels for the tips of the trees.
`label.counts`	An integer vector of length `length(labels)`. The number of tips using each label.

Details

To sample from labeled trees, use the default for `label.counts`, where every value is 1. To sample from unlabeled trees use a `labels` of length 1 and set `label.counts` as a single integer equal to the number of tips in the tree. To sample semilabeled trees `label.counts` should be the number of tips of the tree for each tip label from `labels`.

Value

If `n=1`, a tree of class "phylo". Otherwise, multiple trees in a list of class "multiPhylo".

Author(s)

Nathaniel Malachi Hallinan

References

For labeled trees:

G. Furnas, The generation of random, binary unordered trees, J. Classif. 1 (1984) 187-233.

For unlabeled trees:

L. Cavalli-Sforza, A. Edwards, Phylogenetic analysis: Models and esti- mation procedures, Am. J. Hum. Genet. 19 (1967) 233-257.

For semi-labeled trees:

N. Hallinan. Null models for gene family trees, Math. Biosci. (In review).

See Also

plot.phylo, recon.score

Examples

```
##We are going to sample 10 labeled 10 unlabeled and 10 semi labeled 10 tip trees
labeled<-requiprobable(10,c("A1","A2","B1","B2","B3","C1","C2","C3","C4","C5"))
plot(labeled)

unlabeled<-requiprobable(10,"?",10)
plot(unlabeled)
##If you want to add tip labels to these trees
for (i in 1:10) unlabeled[[i]]$tip.label<-sample(c("A1","A2","B1","B2","B3","C1","C2","C3","C4","C5"))

semilabeled<-requiprobable(10,c("A","B","C"),c(2,3,5))
plot(semilabeled)
```

---

rgenetree *Samples gene family trees that evolved on a species tree with various conditions.*

Description

Samples gene family trees that evolved on the branches of a species tree under the birth-death process. Trees may be conditioned on various priors for the number of reconstructed gene lineages at the root of the species tree, on the total number of genes in the gene tree, on the number of genes in each tip of the species tree, on at least one gene in each tip of the species tree or some combination of those conditions.

Usage

```
rgenetree(n, spec.phy, lams, mus, root = NULL, genetips = NULL, alltips = FALSE)
```

Arguments

n	An integer. The number of trees to sample.
spec.phy	The species tree on which to evolve the gene trees of class "phylo" with branch lengths.
lams	The gene duplication rate in units of duplications per spec.phy branch lengths. Either a single numeric for the whole tree or a vector of length dim(spec.phy$edge)[1] for each branch of the species tree.
mus	The gene loss rate in units of losses per spec.phy branch lengths. Either a single numeric for the whole tree or a vector of length dim(spec.phy$edge)[1] for each branch of the species tree.
root	The prior distribution of reconstructed gene lieages at the root of the species tree. Either NULL, a single positive integer, or a positive numeric vector summing to 1. See details.

genetips	The number of genes at the tips of the species tree. Either NULL, a single positive integer or a vector of positive integers of length length(spec.phy$tip.label). See details.
alltips	A logical. Should the sample only include gene trees with at least one gene in each species. Ignored if length(genetips==length(spec.phy$tip.label).

Details

root sets the prior distribution of the number of reconstructed gene lineages at the root node of the species tree. If is.null(root), then the prior on the number of lineages is flat, but if is.null(genetips) then root must have some value. If root is a single positive integer, then all gene trees will start with root reconstructed gene lineages at the root. To establish a more complex prior root should be a numerical vector for which the *ith* element represents the prior probability that there are *i* gene lineages at the root, so that sum(root)==1. In that case root can be of any length and the prior probability of any number of lineages greater than length(root) will be 0. It should be noted that this is the prior probability assuming is.null(genetips) & !alltips, changes to those values will affect the probabilities (Hallinan 2012?).

genetips sets the conditions on the number of genes in the tips of the spec.phy. If is.null(genetips) then there are no conditions. If genetips is a single integer, then that value will be the total number of genes in each sampled gene tree, although the number of genes in each species may vary from sample to sample. One can set the number of genes in each tip of the spec.phy with a genetips of length length(spec.phy$tip.label), where the *ith* member of genetips will be the number of genes in spec.phy$tip.label[i].

Value

If n==1, a tree of class "phylo". Otherwise, multiple trees in a list of class "multiPhylo".

Author(s)

Nathaniel Malachi Hallinan

References

N. Hallinan. Null models for gene family trees, Math. Biosci. (In review).

See Also

recon.score,plot.phylo,plot.recon(coming soon)

Examples

```
##First we need a simple species tree
spec<-read.tree(text="((A:0.5,B:0.5):0.5,C:1);")
##Now we sample ten gene trees starting with 3 reconstructed gene lineages
phy.all<-rgenetree(10,spec,0.5,0.5,3)
plot(phy.all)

##Now let's make sure that every tip has at least one gene and set an exponential prior on the root
phy.full<-rgenetree(10,spec,0.5,0.5,exp(-(1:20))/sum(exp(-(1:20))),NULL,TRUE)
```

```
plot(phy.full)

##Now lets force the whole gene tree to end in 10 genes and set a flat prior for the root
phy.10<-rgenetree(10,spec,0.5,0.5,NULL,10)
plot(phy.10)

##Now lets start with 3 genes, set the number of genes at each tip of spec and vary mu between the branches of spec
phy.253<-rgenetree(10,spec,0.5,c(0,1,0.2,0.5),3,c(2,5,3))
plot(phy.253)
```

Index