Package: phybase (via r-universe)

September 6, 2024

Type Package

Title Basic functions for phylogenetic analysis

Version 1.1

Date 2008-03-25

Author Liang Liu

Maintainer Liang Liu <lliu@oeb.harvard.edu>

Depends Matrix, ape

Description This package provides functions to read, write, manipulate, estimate, and summarize phylogenetic trees including species trees which contain not only the topology and branch lengths but also population sizes. The input/output functions can read tree files in which trees are presented in parenthetic format. The trees are read in as a string and then transformed to a matrix which describes the relationship of nodes and branch lengths. The nodes matrix provides an easy access for developers to further manipulate the tree, while the tree string provides interface with other phylogenetic R packages such as ``ape''. The input/output functions can also be used to change the format of tree files between NEXUS and PHYLIP. Some basic functions have already been established in the package for manipulating trees such as deleting and swapping nodes, rooting and unrooting trees, changing the root of the tree. The package also includes functions such as ``consensus'', ``coaltime, "popsize``, "treedist" for summarizing phylogenetic trees, calculating the coalescence time, population size, and tree distance. The function maxtree is built in the package to esimtate the species tree from multiple gene trees.

License GPL (>= 2)

Date/Publication 2012-10-29 08:59:26

Repository https://phylotastic.r-universe.dev

RemoteUrl https://github.com/cran/phybase

RemoteRef HEAD

RemoteSha d8ea06b72e176a07c0164e7e9f7ea66966b9f467

Contents

`phybase-package` *Basic functions for Phylogenetic trees*

Description

This package provides functions to read, write, manipulate, estimate, and summarize phylogenetic trees including species trees which contains not only the topology and branch lengths but also population sizes. The input/output functions can read tree files in which trees are presented in parenthetic format. The trees are read in as a string and then transformed to a matrix which describes the relationship of nodes and branch lengths. The nodes matrix provides an easy access for developers to further manipulate the tree, while the tree string provides interface with other phylogenetic R packages such as "ape". The input/output functions can also be used to change the format of tree files between NEXUS and PHYLIP. Some basic functions have already been established in the package for manipulating trees such as deleting and swapping nodes, rooting and unrooting trees, changing the root of the tree. The package also includes functions such as "consensus", "coaltime, "popsize", "treedist" for summarizing phylogenetic trees, calculating the coalescence time, population size,

and tree distance. The function maxtree is built in the package to esimtate the species tree from multiple gene trees.

Details

Package:	PhyBase
Type:	Package
Version:	1.1
Date:	2008-03-25
License:	GPL (>=2.0.0)

Author(s)

Liang Liu

Maintainer: Liang Liu <lliu@oeb.harvard.edu>

`ancandtime` *Get ancestors and their divergence times*

Description

This function returns the ancestors of a node and their divergence times.

Usage

```
ancandtime(inode, nodematrix, nspecies)
```

Arguments

`inode`	a node in the tree.
`nodematrix`	the tree matrix.
`nspecies`	number of species (taxa) in the tree.

Author(s)

Liang Liu

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
nodematrix<-read.tree.nodes(treestr)$nodes
inode<-6
ancandtime(inode,nodematrix,nspecies=5)
```

ancestor *Find the ancestral nodes of a node*

Description

The function returns the ancestral nodes of inode including inode itself.

Usage

```
ancestor(inode, nodematrix)
```

Arguments

inode	the node number
nodematrix	the tree node matrix. it must be a rooted tree.

Value

The function returns a vector of ancestoral nodes of inode including inode itself.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

mrca.2nodes, mrca.nodes

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
nodematrix<-read.tree.nodes(treestr)$nodes
ancestor(6,nodematrix)
```

bootstrap *Bootstrap sequences*

Description

This function can be used to bootstrap sequences.

Usage

```
bootstrap(sequence)
```

Arguments

sequence	sequence matrix.

Details

In the sequences matrix, the columns are "Taxa" and the rows are "sites". The function will bootstrap the rows.

Value

the function returns a sequence matrix with sites randomly sampled from the original matrix with replacement.

Author(s)

Liang Liu

Examples

```
#construct the DNA sequences of three taxa
seq <- matrix("A",ncol=4,nrow=3)
rownames(seq)<-c("taxa1","taxa2","taxa3")
seq[,2]<-"G"
seq[,3]<-"C"
seq[,4]<-"T"
bootstrap(seq)
```

---

bootstrap.mulgene *Bootstrap sequences from multiple loci*

Description

The function bootstraps sequence columns for each locus sampled from the original multilocus data. It consists of two step. First, it bootstraps loci. Then it bootstraps sequences for each locus.

Usage

```
bootstrap.mulgene(sequence,gene,name,boot,outfile="")
```

Arguments

sequence	data matrix
gene	location of each locus
name	taxa names of sequences
boot	the number of bootstrap samples
outfile	output file

Details

In the sequences matrix, the rows are "Taxa" and the columns are "sites".

Value

The function generates a data file in phylip format.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

bootstrap

Examples

```
#construct the DNA sequences of three taxa
seq <- matrix("A",ncol=4,nrow=3)
rownames(seq)<-c("taxa1","taxa2","taxa3")
seq[,2]<-"G"
seq[,3]<-"C"
seq[,4]<-"T"

name<-rownames(seq) #taxa names of the sequences

#construct two loci. The first two nucleotides represent the first locus, while nucleotide 3 and 4 represent the sec
gene<-matrix(0,ncol=2,nrow=2)
gene[1,]<-c(1,2)
gene[2,]<-c(3,4)
gene
bootstrap.mulgene(seq,gene,name,boot=2,outfile="bootdata.txt") #the output file is saved at "bootdata.txt"
```

---

change.root *Change tree root*

Description

The function changes the tree root.

Usage

```
change.root(nodematrix, newroot)
```

Arguments

nodematrix	the tree node matrix
newroot	the node number of the new root

Details

The function always returns an unrooted tree. Use the function `link{root.tree}` to root the unrooted tree if you need a rooted tree.

Value

`nodes`	the tree node matrix after changing the tree root
`rootnode`	the node number of the new root

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

`root.tree`, `rootoftree`

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
change.root(nodematrix,6)
```

ChangeBrlen *Change the branch length*

Description

for internal use only

---

coal.sptree *Estimating species trees using average coalescence times*

Description

For a given set of gene trees, the UPGMA tree is constructed from the distance matrix based on the average coalescence times among taxa.

Usage

```
coal.sptree(trees, speciesname, nspecies, outgroup=1)
```

Arguments

trees	a vector of trees in newick format
speciesname	species names
nspecies	number of species
outgroup	the node number of the species used to root the tree

Details

If the gene trees are not clocklike trees, they are first converted to clock trees using function noclock2clock and then construct a distance matrix in which the entries are twice the coalescence times among species. The distance matrix is used to build an UPGMA tree as the estimate of the species tree. This function is different from steac.sptree in that steac.sptree uses nucleotide distances to construct distance matrix.

Value

The function returns the tree node matrix and the estimate of the species tree.

Author(s)

Liang Liu

See Also

See also to steac.sptree

Examples

```
data(rooted.tree)
genetrees<-rooted.tree
spname<-species.name(genetrees[1])
coal.sptree(genetrees,spname,nspecies=4,outgroup=4)
```

---

coaltime *Coalescence time of two nodes*

Description

The function computes the coalescence time of two nodes.

Usage

```
coaltime(inode, jnode, nodematrix, nspecies)
```

Arguments

inode	the first node, it could be an internode.
jnode	the second node, it could be an internode.
nodematrix	the tree node matrix
nspecies	the number of species

Value

the function returns the coalescence time of inode and jnode.

Author(s)

Liang Liu

See Also

popsize

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
taxaname<-species.name(treestr)
nodematrix<-read.tree.nodes(treestr,name=taxaname)$nodes
coaltime(1,2,nodematrix,5) #the coalescence time of taxa H (1) and C (2).
```

---

consense *Consensus tree*

Description

The function returns a consensus tree from multiple gene trees.

Usage

```
consense(treestr, name,type="freq")
```

Arguments

treestr	a vector of tree strings
name	the species names
type	if type="freq", the frequency of each clade in the consensus tree is presented at the node of the clade. if type="prop", the proportion of each clade is presented at the node of the clade"

Value

The function returns the consensus tree and species names.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

maxtree, partition.tree

Examples

```
treestr<-c("((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);","((((H:0.00402,G:0.00
name<-species.name(treestr[1])
consense(treestr,name)


###unrooted trees
data(unrooted.tree)
name<-paste("S",1:29,sep="")
consense(unrooted.tree,name)
```

---

del.node *Delete a node from the tree*

Description

This function deletes a node (and its descendant nodes) from the tree.

Usage

```
del.node(inode, name, nodematrix)
```

Arguments

inode	the node to be deleted
name	the species names
nodematrix	the tree node matrix

Details

The species names are those defined in the original tree before deleting the node inode. No need to delete the species name of inode! If inode is an internode, the whole subtree below inode will be deleted.

Value

nodes	the tree node matrix after deleting inode
treestr	the tree string of the tree after deleting inode.

Author(s)

Liang Liu

See Also

change.root, swap.nodes

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
spname<-read.tree.nodes(treestr)$names
nodematrix<-read.tree.nodes(treestr)$nodes
del.node(6,spname,nodematrix)

##unrooted tree
data(unrooted.tree)
nodematrix<-read.tree.nodes(unrooted.tree[1])$nodes
name<-paste("S",1:29,sep="")
del.node(6,name,nodematrix)
```

FindSpnodeDownGenenode *Internal function*

Description

for internal use only

---

genetree.vector *Construct gene tree vectors from multiple loci*

Description

This function constructs gene tree vectors from gene trees across loci. The gene tree vectors can be used to construct maximum tree by the function maxtree.

Usage

```
genetree.vector(filenames,outputfile)
```

Arguments

filenames	the gene tree files
outputfile	the output file

Value

The function returns a matrix of gene trees. Each row represents a gene tree vector. The gene tree vector consists of trees from multiple gene tree files.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

References

Liu, L. and D.K. Pearl. Species trees from gene trees: reconstructing Bayesian posterior distributions of a species phylogeny using estimated gene tree distributions. Systematic Biology, 2007, 56:504-514.

Edwards, S.V., L. Liu., and D.K. Pearl. High resolution species trees without concatenation. PNAS, 2007, 104:5936-5941.

See Also

maxtree

---

getcoaltime *Get coalescence times*

Description

This function can get gene coalescence times in the species tree.

Usage

```
getcoaltime(genetree, sptree, ntax, nspecies, species.structure)
```

Arguments

genetree	a genetree matrix
sptree	a species tree matrix
ntax	number of taxa in the gene tree
nspecies	number of species in the species tree
species.structure	sequence-species relationship

Value

The function returns a two-column matrix, the first column is the ancestral node in the species tree, the second column is the gene coalescence time at the corresponding ancestral node in the species tree.

Author(s)

Liang Liu

Examples

```
genetree<-"(((A:1,B:1):3,C:4):2,D:6);"
sptree<-"(((A:0.5,B:0.5):1,C:1.5):1,D:2.5);"
name<-c("A","B","C","D")

genetree<-read.tree.nodes(genetree,name)$nodes
sptree<-read.tree.nodes(sptree,name)$nodes

ntax<-length(name)
nspecies<-length(name)
species.structure<-matrix(0,nrow=nspecies,ncol=ntax)
diag(species.structure)<-1

getcoaltime(genetree,sptree,ntax,nspecies,species.structure)
```

---

getncoal *internal function*

Description

This is an internal function for calculating the rannala and yang's formula

---

is.clock *Is a clock tree or not*

Description

This function checks the tree to see if the branch lengths satisfy the molecular clock assumption. For each node, the lengths of the left lineage and right lineage are compared. If they are not equal to each other and the difference is greater than `threshold`, the function will return FALSE. This function does not perform statistical test for the molecular clock assumption.

Usage

```
is.clock(nodematrix, nspecies,threshold)
```

Arguments

`nodematrix`	the tree node matrix
`nspecies`	the number of species
`threshold`	the critical value for the difference between the length of the left decendant lineage and that of the right decendant lieage of an internode. The difference below the threshold is treated as no difference.

Value

The function returns TRUE for a clock tree and FALSE for a non-clock tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

is.rootedtree

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00705):0.00929,O:0.01635):0.1,W:0.11635);"
nodematrix<-read.tree.nodes(treestr)$nodes

##if the threshold is set to be large, the tree is a clock tree
is.clock(nodematrix,5,0.0001)
##[1] TRUE

##if the threshold is a small number, the tree is not a clock tree.
is.clock(nodematrix,5,0.00001)
##[1] FALSE
```

---

is.rootedtree *Is the tree rooted or not*

Description

This function can test if the tree is rooted.

Usage

```
is.rootedtree(tree)
```

Arguments

tree	tree string or tree node matrix

Value

The function returns TRUE if the tree is a rooted tree. Otherwise, it returns FALSE.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

is.clock

Examples

```
data(unrooted.tree)
nodematrix<-read.tree.nodes(unrooted.tree[1])$nodes
is.rootedtree(nodematrix)

data(rooted.tree)
is.rootedtree(rooted.tree[1])
```

---

maxtree *Maximum Tree*

Description

The function computes the Maximum Tree from multiple gene trees.

Usage

```
maxtree(genetreevector,spname,taxaname,species.structure)
```

Arguments

genetreevector	a vector of gene trees
spname	the species names
taxaname	the names of taxa
species.structure	the correspondence between species and taxa

Value

The function returns the node matrix and tree string of the maximum tree. It also returns the species names.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

References

Liu, L. and D.K. Pearl. Species trees from gene trees: reconstructing Bayesian posterior distributions of a species phylogeny using estimated gene tree distributions. Systematic Biology, 2007, 56:504-514.

Edwards, S.V., L. Liu., and D.K. Pearl. High resolution species trees without concatenation. PNAS, 2007, 104:5936-5941.

See Also

consense, genetree.vector

Examples

```
genetreevector<-c("((((H:0.00302,C:0.00302):0.00304,G:0.00605):0.01029,O:0.01635):0.1,W:0.11635);","((((H:0.00
species.structure<-matrix(0,5,5)
diag(species.structure)<-1
name<-species.name(genetreevector[1])
maxtree(genetreevector,name,name,species.structure)
```

---

mrca.2nodes *Find the most recent common ancestor of two nodes*

Description

The function can find the most recent common ancestor of two nodes inode and jnode

Usage

```
mrca.2nodes(inode, jnode, nodematrix)
```

Arguments

inode	the node inode
jnode	the node jnode
nodematrix	the tree node matrix

Value

anc	the node number of the most recent common ancestor of inode and jnode.
dist	the distance between the two nodes.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

mrca.nodes, coaltime, popsize

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
mrca.2nodes(1,2,nodematrix)
```

mrca.nodes *Find the most recent common ancestor of multiple nodes*

Description

The function can find the most recent common ancestor of multiple nodes specified in nodevector

Usage

```
mrca.nodes(nodevector, nodematrix)
```

Arguments

nodevector	a set of nodes
nodematrix	the tree node matrix

Value

The function returns the node number of the most recent common ancestor of the nodes in nodevector.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

mrca.2nodes, coaltime, popsize

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
mrca.nodes(c(1,2,3),nodematrix)
```

mutation_exp *Generate mutation rates for populations in the species tree*

Description

In the non-clock species tree model (Liu, et.al), the lineages (populations) in the species tree are allowed to have variable mutation rates. This function is used to simulate mutation rates for the non-clock species tree model. There are many other ways to simulate variable mutation rates across populations in the species tree.

Usage

```
mutation_exp(sptree, root, inode, nspecies,alpha)
```

Arguments

sptree	the species tree matrix
root	the root of the species tree
inode	the root of the species tree
nspecies	the number of species in the species tree
alpha	the parameter in the gamma distribution used to generate mutation rates.

Details

mutation rates are generated from gamma (alpha, alpha/w) where w is the mutation rate of the parent population of the current node. Thus the mean of the mutation rate of the current node equals to the mutation rate of its parent population.

Value

The function returns a species tree matrix with mutation rates in the last column.

Author(s)

Liang Liu

Examples

```
sptree<-"((((H:0.00402#0.01,C:0.00402#0.01):0.00304#0.01,G:0.00707#0.01):0.00929#0.01,O:0.01635#0.01):0.1#0.01
nodematrix<-read.tree.nodes(sptree)$nodes
mutation_exp(nodematrix, root=9, inode=9, nspecies=5, alpha=5)
```

---

name2node *Replace species names by their node numbers*

Description

This function replaces the species names in the tree string with their node numbers.

Usage

```
name2node(treestr,name="")
```

Arguments

treestr	the tree string
name	the species names

Details

If species names are not given, the function will use the sorted species names in the tree string.

Value

The function returns the tree string with the species names replaced by the node numbers.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

subtree.length, node2name

Examples

```
treestr<-"(((H:4.2,C:4.2):3.1,G:7.3):6.3,O:13.5);"
name<-c("H","G", "C", "O")
name2node(treestr,name)
```

---

noclock2clock *Convert a non-clocklike tree to a clocklike tree*

Description

This function converts a non-clocklike tree to a clocklike tree using an ad-hoc approach described in the paper Liu et al 2007.

Usage

```
noclock2clock(inode, treematrix, nspecies)
```

Arguments

inode	root of the tree
treematrix	tree node matrix
nspecies	the number of species in the tree

Value

The function returns the tree node matrix of the clocklike tree.

Author(s)

Liang Liu

References

~put references to the literature/web site here ~

Examples

```
treestr<-"(((H:1,C:3):2,G:6):2,O:10);"
name<-species.name(treestr)
treenode<-read.tree.nodes(treestr,name)$nodes
noclock2clock(7,treenode,4)
```

---

node.height *Calculate node height*

Description

The function calculates the height of a node. The tree is assumed to be an ultramatric tree.

Usage

```
node.height(inode, nodematrix, nspecies)
```

Arguments

inode	the node number
nodematrix	the tree node matrix
nspecies	the number of species in the tree

Value

The function returns the height of inode.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

subtree.length

Examples

```
tree.string<-"(((H:4.2,C:4.2):3.1,G:7.3):6.3,O:13.5);"
nodematrix<-read.tree.nodes(tree.string)$nodes
node.height(6,nodematrix,4)
```

node2name *Replace node numbers by species names in a tree string*

Description

This function replaces node numbers in a tree string by species names.

Usage

```
node2name(treestr,name="")
```

Arguments

treestr	a tree string
name	species names

Value

The function returns the tree string with the node numbers replaced by the species names.

Author(s)

Liang Liu

See Also

subtree.length, name2node

Examples

```
treestr<-"(((1:4.2,2:4.2):3.1,3:7.3):6.3,4:13.5);"
name<-c("H","C", "G", "O")
node2name(treestr,name)
```

offspring.nodes *Find the offspring nodes*

Description

The function returns the offspring nodes of inode.

Usage

```
offspring.nodes(inode, nodematrix, nspecies)
```

Arguments

inode	the node of which the the offspring nodes will be found by the function.
nodematrix	the tree node matrix.
nspecies	the number of species.

Value

The function returns the offspring nodes of inode.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

offspring.species

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
offspring.nodes(7,nodematrix,5)
```

---

offspring.nodes.string

Find offspring nodes (internal use only)

Description

The function returns a string of offspring nodes of inode.

Usage

```
offspring.nodes.string(inode, nodematrix, nspecies)
```

Arguments

inode	the node of which the the offspring nodes will be found by the function.
nodematrix	the tree node matrix
nspecies	the number of species

Value

The function returns a string of offspring nodes of inode.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

`offspring.species` *Find the species nodes*

Description

The function returns the descendant species of `inode`.

Usage

```
offspring.species(inode, nodematrix, nspecies)
```

Arguments

`inode`	the node.
`nodematrix`	the tree node matrix
`nspecies`	the number of species

Value

This function returns the descendant species of `inode`, while the function `offspring.nodes` returns all the descendant nodes of `inode` including internal nodes in the tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

`offspring.nodes`

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
offspring.species(7,nodematrix,5)
```

pair.dist *Calculate all pairwise distances among taxa in the tree*

Description

The function computes all pairwise distances among taxa in the tree.

Usage

```
pair.dist(nodematrix, nspecies)
```

Arguments

nodematrix	the tree node matrix
nspecies	the number of taxa in the tree

Value

The function returns a distance matrix.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

treedist, upgma, maxtree

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00705):0.00929,O:0.01635):0.1,W:0.11635);"
nodematrix<-read.tree.nodes(treestr)$nodes
pair.dist(nodematrix,5)
```

pair.dist.dna *Calculate pairwise distances among DNA sequences*

Description

Calculate pairwise distances among DNA sequences. The DNA sequences are coded as 1:A, 2:G, 3:C, 4:T.

Usage

```
pair.dist.dna(sequences, nst = 0)
```

Arguments

sequences	DNA sequences
nst	substitution model. 0:no model, 1:JC

Details

If nst=0, the distance is equal to the proportion of sites having different nucleotides between two sequences.

Value

The function returns a distance matrix.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

References

Jukes, TH and Cantor, CR. 1969. Evolution of protein molecules. Pp. 21-123 in H. N. Munro, ed. Mammalian protein metabolism. Academic Press, New York.

See Also

upgma

Examples

```
tree<-"(((H:0.00402#0.01,C:0.00402#0.01):0.00304#0.01,G:0.00707#0.01):0.00929#0.01,O:0.01635#0.01)#0.01;"
nodematrix<-read.tree.nodes(tree)$nodes
sequences<-sim.dna(nodematrix,10000,model=1)
pair.dist.dna(sequences,nst=1)
```

---

pair.dist.mulseq *Calculate pairwise distances among species*

Description

If some species have multiple taxa, the pairwise distance between two species is equal to the average of the distances between all pairs of taxa in the two species. This functions returns the pairwise distances among species (average over all taxa in the species).

Usage

```
pair.dist.mulseq(dist, species.structure)
```

Arguments

dist	the distance matrix of taxa
species.structure	a matrix with rows representing species and columns representing taxa. 1: the species (row) has the taxon at the corresponding column. see the example.

Value

This functions returns the distance matrix of species.

Author(s)

Liang Liu

See Also

See Also as pair.dist

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00705):0.00929,O:0.01635):0.1,W:0.11635);"
nodematrix<-read.tree.nodes(treestr)$nodes
dist<-pair.dist(nodematrix,5)
species.structure<-matrix(0,nrow=2,ncol=5) #2 species and 5 taxa
species.structure[1,]<-c(1,1,1,0,0)        #taxa 1,2,3 belong to the first species
species.structure[2,]<-c(0,0,0,1,1)        #taxa 4,5 belong to the second species
pair.dist.mulseq(dist,species.structure)
```

---

partition.tree *partition a tree*

Description

partition a tree.

Usage

```
partition.tree(tree,nspecies)
```

Arguments

tree	the tree node matrix
nspecies	the number of species

Value

The function returns a matrix. Each row represents a particular partition of the tree. The position of "1" in the matrix indicates the presence of the corresponding species in the partition. The last number at each row is the frequency of that partition. This function returns the partition matrix for only one tree. For multiple trees, the partitions and their frequencies can be obtained by the function consense.

Author(s)

Liang Liu

See Also

consense

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
partition.tree(nodematrix,5)
#
#     [,1] [,2] [,3] [,4] [,5] [,6]
#[1,]    1    0    1    0    0    1
#[2,]    1    1    1    0    0    1
#[3,]    1    1    1    1    0    1
#
#The last number of each row is the frequency of the corresponding partition. For example, the frequency of the
#first partition (1 0 1 0 0) is 1. The first partition includes species 1 and 3 as indicated by the position of 1 in th
#Each row represens a partition and its frequency.
```

---

plottree *Write a tree file*

Description

The function plots phylogenetic trees.

Usage

```
plottree(tree)
```

Arguments

tree	a phylogenetic tree in newrick format

Author(s)

use the function "plot.phylo" in package APE to plot phylogenetic trees.

See Also

write.subtree, read.tree.string

Examples

```
treestr<-"(((H:4.2,C:4.2):3.1,G:7.3):6.3,O:13.5);"
plottree(treestr)
```

---

popsize *Population size of the most recent common ancestor of two nodes*

Description

This function computes the population size of the most recent common ancestor of two nodes.

Usage

```
popsize(inode, jnode, nodematrix)
```

Arguments

inode	the first node, it could be an internode.
jnode	the second node, it could be an internode.
nodematrix	the tree node matrix

Value

The function returns the population size of the most recent common ancestor of inode and jnode.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

coaltime

Examples

```
treestr<-"((((H:0.00402,C:0.00402#0.035):0.00304,G:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
nodematrix<-read.tree.nodes(treestr)$nodes
popsize(1,2,nodematrix)
#[1] -9   ##this tree does not have values for population size.

popsize(1,1,nodematrix)
#[1] 0.035       ##the population size for the species C is 0.035
```

populationMutation *Change branch lengths of a gene tree in the non-clocklike species tree model (internal use only)*

Description

This function changes branch lengths of a gene tree with the mutation rates in the species tree.

Usage

```
populationMutation(sptree, spnodedepth, genetree, genenodedepth, speciesmatrix)
```

Arguments

sptree	the species tree
spnodedepth	depth of the species tree
genetree	a gene tree
genenodedepth	depth of the gene tree
speciesmatrix	tree node matrix of the species tree

Value

It returns a gene tree.

Author(s)

Liang Liu

postdist.tree *Calculate posterior probabilities of trees*

Description

The function summarize a set of trees by calculating the proportion of each tree in the tree set.

Usage

```
postdist.tree(trees,name)
```

Arguments

trees	a vector of tree strings
name	the species names

Value

trees	a vector of tree
prob	the probability associated with each tree in the vector tree

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

See Also as read.tree.nodes

Examples

```
library(phybase)
tree<-"(((H:0.005 , C:0.005 ) : 0.00025 #.01, G:0.00525):0.00025 #0.01 , O:0.0055) #.01;"
name<-species.name(tree)
nodematrix<-read.tree.nodes(tree,name)$nodes
rootnode<-7
seq<-rep(1,4)
nsim<-100
str<-rep(0,nsim)

for(i in 1:nsim){
 str[i]<-sim.coaltree.sp(rootnode,nodematrix,4,seq,name=name)$gt
}
postdist.tree(str,name)
```

---

rank.nodes *Node ranks (internal use only)*

Description

The function returns the rank of each node in the tree.

Usage

```
rank.nodes(treenode, inode, ntaxa, start, rank)
```

Arguments

treenode	tree node matrix
inode	the tree root
ntaxa	the number of taxa in the tree
start	the maximum rank
rank	a dummy vector

Value

The function returns a vector of ranks for the nodes in the tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

mrca.2nodes, mrca.nodes

---

rannalandyang *Rannala and Yang's formula*

Description

This function calculates the likelihood of a vector of gene trees given the species tree using the Rannala and Yang's formula

Usage

```
rannalandyang(gtree, stree, taxaname,spname,species.structure)
```

Arguments

gtree	a collection of gene trees
stree	a species tree in newick format
taxaname	the names of taxa
spname	the names of species
species.structure	define which sequence belong to which species

Value

The function returns the log likelihood score.

Author(s)

Liang Liu

References

Rannala, B. and Z. Yang. 2003. Bayes estimation of species divergence times and ancestral population sizes using DNA sequences from multiple loci. Genetics 164: 1645-1656.

Examples

```
gtree<-"(((A:1,B:1):3,C:4):2,D:6);"
stree<-"(((A:0.5,B:0.5):1#0.1,C:1.5):1#0.1,D:2.5)#0.1;"
taxaname<-c("A","B","C","D")
spname<-taxaname
ntax<-length(taxaname)
nspecies<-length(spname)
species.structure<-matrix(0,nrow=nspecies,ncol=ntax)
diag(species.structure)<-1
rannalandyang(gtree,stree,taxaname,spname,species.structure)
```

---

rdirichlet *Generate random numbers from the dirichlet distribution*

Description

This function can generate random numbers from a dirichlet distribution.

Usage

```
rdirichlet(n,a)
```

Arguments

n	the number of random numbers to be generated
a	shape parameters of the dirichlet distribution

Value

The function returns random numbers from a dirichlet distribution.

Author(s)

Code is taken from Greg's Miscellaneous Functions (gregmisc). His code was based on code posted by Ben Bolker to R-News on Fri Dec 15 2000.

Examples

```
rdirichlet(1,c(3,3,3))
```

read.dna.seq *Read sequences from files*

Description

The function reads sequences from files in the nexus or phylip format.

Usage

```
read.dna.seq(file="", format="nexus")
```

Arguments

file	the input file name
format	nexus or phylip

Value

seq	sequences
gene	partitions on the sequences. Each partition represents a gene or a locus.

Author(s)

Liang Liu

read.tree.nodes *Read tree nodes*

Description

Read a tree string in parenthesic format and output tree nodes, species names and whether the tree is rooted

Usage

```
read.tree.nodes(str, name = "")
```

Arguments

str	a tree string in the parenthetical format
name	species names

Details

This function reads a tree string into a matrix that describes the relationships among nodes and corresponding branch lengths. Each row in the matrix represents a node. The first n rows contain the information of the nodes at the tips of the tree. The order of the first n nodes is identical to the alphabetic order of the species names given by name. If name is null, the names will be extracted from the tree str and the first n nodes are in the same order as the species names appear in the tree str from left to right.

The numbers after ":" are branch lengths. The numbers after pound signs are population sizes. The numbers after "

Value

`nodes`	nodes is a matrix that describes the relationships among nodes and corresponding branch lengths and population sizes if the tree is a species tree. Each row corresponds a node in the tree. The matrix has 5 columns. The first column is the father of the current node. The following columns are left son, right son, branch length, and population size. The value -9 implies that the information does not exist. The last row is the root of the tree. If the tree is unrooted, the first number of the root node is -8, while it is -9 for a rooted tree.
`names`	species names in the same order of the first n nodes.
`root`	TRUE for a rooted tree, FALSE for an unrooted tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

`read.tree.string`, `species.name`

Examples

```
##read an unrooted tree
data(unrooted.tree)
tree<-read.tree.nodes(unrooted.tree[1])
tree$nodes
tree$names
tree$root

#read a rooted tree
data(rooted.tree)
tree<-read.tree.nodes(rooted.tree[1])
tree$nodes
tree$names
tree$root
```

read.tree.string *Read tree strings from a tree file*

Description

This function reads tree strings in the parenthetical format from a tree file. The output of the function is a vector of tree strings that can be converted to a matrix of nodes by the function read.tree.nodes.

Usage

```
read.tree.string(file = "", format="nexus")
```

Arguments

file	the tree file that contains trees in the parenthetical format.
format	phylip or nexus

Details

The function can read NEXUS and PHYLIP tree files. It works for other types of tree files as long as the trees in the tree files are parenthetical trees. This function combining with write.tree.string can change the tree file format.

Value

tree	a vector of tree strings.
names	species names.
root	TRUE for rooted trees, FALSE for unrooted trees

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

write.tree.string, read.tree.nodes

Examples

```
##read rooted trees in PHYLIP format
cat("(((H:4.2,C:4.2):3.1,G:7.3):6.3,O:13.5);",file = "phylip.tre", sep = "\n")
tree.string<-read.tree.string("phylip.tre")
tree.string

##read unrooted trees in NEXUS format
cat("#NEXUS
[ID: 4045516090]
```

```
begin trees;
   translate
       1 WW_7A03_1,
       2 WW_7H06_2,
       3 WW_7H05_1,
       4 WW_N03__5,
       5 WW_Snnr_1,
       6 WW_7P10__1,
       7 WW_7A05_1,
       8 WW_B03__1,
       9 WW_B04_1,
      10 WW_D07_9,
      11 WW_7K01_1,
      12 WW_7K04_1,
      13 WW_7N13_1,
      14 WW_M02_1,
      15 WW_N04_1,
      16 WW_UK6_1,
      17 WW_7A04_1,
      18 Pfuscatus_PF2_4,
      19 Pfuscatus_PF1_1,
      20 Pfuscatus_PF3_2,
      21 PCabietinus_331_1,
      22 PCabietinus_333_6,
      23 PCabietinus_336_1,
      24 PCcollybita_GB_1,
      25 PCtristis_GB_1,
      26 PCbrehmii_GB_1,
      27 Psibilatrix_GB_1,
      28 Pbonelli_GB_1,
      29 PTviridanus_1;
  tree rep.1 = ((((((((((((((16:0.100000,2:0.100000):0.100000,20:0.100000):0.100000,21:0.100000):0.100000,8:0.
  tree rep.100 = ((((((((((((((16:0.100000,2:0.100000):0.100000,20:0.100000):0.100000,21:0.100000):0.100000,8:
  tree rep.200 = ((((((((((((((16:0.100000,2:0.100000):0.100000,20:0.100000):0.100000,21:0.100000):0.100000,8:
end;",file="tree.nexus")
tree.string<-read.tree.string("tree.nexus")
tree.string
```

`root.tree` *Root a tree*

Description

Root a tree.

Usage

```
root.tree(nodematrix,outgroup)
```

Arguments

nodematrix	the tree node matrix
outgroup	the node used as outgroup

Value

The function returns a rooted tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

rootoftree, is.rootedtree

Examples

```
data(unrooted.tree)
nodematrix<-read.tree.nodes(unrooted.tree[1])$nodes
root.tree(nodematrix,23)
```

---

rooted.tree *An example of rooted trees*

Description

An example of rooted trees

Usage

```
data(rooted.tree)
```

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

Examples

```
data(rooted.tree)
read.tree.nodes(rooted.tree[1])
```

rootoftree *Root of a tree*

Description

This function can be used to find the root of a tree.

Usage

```
rootoftree(nodematrix)
```

Arguments

nodematrix	the tree node matrix

Value

The function returns the root of the tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

rootoftree, root.tree

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
spname<-read.tree.nodes(treestr)$names
rootoftree(nodematrix)
```

sctree *Shallowest Coalescence Tree*

Description

The function computes the shallowest coalescence tree from multiple gene trees.

Usage

```
sctree(genetreevector,spname,taxaname,species.structure)
```

Arguments

genetreevector	a vector of gene trees
spname	the species names
taxaname	the names of taxa
species.structure	the correspondence between species and taxa

Value

The function returns the node matrix and tree string of the maximum tree. It also returns the species names.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

References

Maddison, W. P., and L. L. Knowles. 2006. Inferring phylogeny despite incomplete lineage sorting. Syst. Biol. 55:21-30.

See Also

consense, genetree.vector

Examples

```
genetreevector<-c("((((H:0.00302,C:0.00302):0.00304,G:0.00605):0.01029,O:0.01635):0.1,W:0.11635);","((((H:0.00
species.structure<-matrix(0,5,5)
diag(species.structure)<-1
name<-species.name(genetreevector[1])
sctree(genetreevector,name,name,species.structure)
```

---

sim.coaltree *Simulate a coalescence tree*

Description

This function can simulate a coalescence tree from a single population with parameter theta. The coalescence times in the tree have exponential distributions. theta is equal to 4uNe where Ne is the effective population size and u is the mutation rate.

Usage

```
sim.coaltree(nspecies,theta)
```

Arguments

nspecies	the number of species
theta	the population parameter

Details

theta is the population parameter theta=4N*mu.

Value

The function returns the simulated coalescence tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

References

John Wakeley, Coalescent theory: An introduction.

See Also

sim.coaltree.sp

Examples

```
sim.coaltree(5,theta=0.2)
##[1] "((5:0.55696,(1:0.34858,3:0.34858):0.20838):2.99874,(2:0.97896,4:0.97896):2.57674)"
```

---

sim.coaltree.sp *simulate a gene tree from the species tree*

Description

The function simulates a gene tree from the species tree using Rannala and Yang's formula

Usage

```
sim.coaltree.sp(rootnode, nodematrix, nspecies, seq, name)
```

Arguments

rootnode	the root node of the species tree
nodematrix	the tree node matrix of the species tree
nspecies	the number of species
seq	a vector of number of sequences in each species
name	species names used in the simulated gene tree

Value

gt	the gene tree generated from the species tree
height	the tree height of the gene tree

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

References

Rannala, B. and Z. Yang. 2003. Bayes estimation of species divergence times and ancestral population sizes using DNA sequences from multiple loci. Genetics 164: 1645-1656.

See Also

sim.coaltree

Examples

```
tree<-"(((H:0.00402#0.01,C:0.00402#0.01):0.00304#0.01,G:0.00707#0.01):0.00929#0.01,O:0.01635#0.01)#0.01;"
nodematrix<-read.tree.nodes(tree)$nodes
rootnode<-7
spname<-species.name(tree)
##define the vector seq as [2,2,2,2] which means that there are 2 sequences in each species
seq<-rep(2,4)
str<-sim.coaltree.sp(rootnode,nodematrix,4,seq,name=spname)$gt
```

---

sim.coaltree.sp.mu *Simulate a gene tree from the non-clock species tree model*

Description

The function generates a random gene tree from the species tree under the non-clock species tree model.

Usage

```
sim.coaltree.sp.mu(sptree, spname, seq, numgenetree,method="dirichlet",alpha=5.0)
```

Arguments

sptree	species tree
spname	species names
seq	the species-sequences struction, i.e., which sequence belongs to which species
numgenetree	the number of gene trees to be generated
alpha	the parameter in the gamma distribution. see also mutation_exp
method	either gamma or dirichlet

Value

gt	the simulated gene tree
st	the node matrix of the species tree
seqname	the names of sequences

Author(s)

Liang Liu

Examples

```
sptree<-"(((A:0.5,B:0.5):1#0.1,C:1.5):1#0.1,D:2.5)#0.1;"
spname<-c("A","B","C","D")
seq<-c(1,1,1,1) #each species has only one sequence.
sim.coaltree.sp.mu(sptree, spname, seq, numgenetree=1,method="dirichlet",alpha=5.0)
```

---

sim.dna *Simulate DNA sequences from substitution models*

Description

Simulate DNA sequences from a tree using substitution model

Usage

```
sim.dna(nodematrix, seqlength, model, kappa=2, rate=c(1,1,1,1,1,1), frequency=c(1/4,1/4,1/4,1/4))
```

Arguments

nodematrix	the tree node matrix
seqlength	sequence length
model	1 JC, 2 H2P, 3 HKP, 4 GTR
kappa	the transition/transversion ratio
rate	the six rates used in GTR model
frequency	frequencies of four types of nucleotides

Value

The function returns DNA sequences simulated from the gene tree `nodematrix`. The sequences are coded as 1:A, 2:C, 3:G, 4:T.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

References

Jukes, TH and Cantor, CR. 1969. Evolution of protein molecules. Pp. 21-123 in H. N. Munro, ed. Mammalian protein metabolism. Academic Press, New York.

See Also

sim.coaltree

Examples

```
tree<-"(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635);"
nodematrix<-read.tree.nodes(tree)$nodes
sim.dna(nodematrix,100, model=2, kappa=4)
```

---

simnucleotide *Intrinsic function used in sim.dna*

Description

The function simulates DNA sequences from a tree using the Jukes-Cantor model.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

---

simSeqfromSp *simulate DNA sequences from a species tree*

Description

The function simulates sequences from a species tree.

Usage

```
simSeqfromSp(sptree, spname, ntaxasp, ngene, theta=0, noclock=0, simsequence=1, murate="Dirichlet",alp
```

Arguments

sptree	A species tree which must be a rooted tree.
spname	species names
ntaxasp	a vector of the number of individuals in each species
ngene	number of genes
theta	population size
noclock	0: clocklike species tree 1: nonclocklike species tree

simsequence	1: simulate sequences and gene trees, 0: simulate gene trees
murate	distribution of mutation rates
alpha	the shape parameter of dirichlet distribution
seqlength	the number of nucleotides along the sequences
model	substitution model
kappa	transition/transversion ratio
rate	rates
frequency	nucleotide frequency
outfile	the full path of the output file
format	either "phylip" or "nexus"

Value

The function writes sequences into a file.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

References

Felsenstein, J. The Newick tree format. http://evolution.genetics.washington.edu/phylip/newicktree.html

See Also

write.subtree, read.tree.string

Examples

```
#read the species tree from a data file
data(sptree)
outfile<-"out.txt"
spname <- paste("S",1:20,sep="")
outgroup <- "S20"
ntaxasp <- rep(2,length(spname))
ntaxasp[length(spname)]<-1
ngene<-2
seqlength<-100
simSeqfromSp(sptree,spname,ntaxasp,noclock=1,ngene=ngene,seqlength=seqlength,model=1,outfile=outfile)
simSeqfromSp(sptree,spname,ntaxasp,noclock=0,ngene=ngene,simsequence=0,seqlength=seqlength,model=1,outfile=out
```

site.pattern *Site patterns*

Description

The function returns site patterns.

Usage

```
site.pattern(seq)
```

Arguments

seq	DNA sequences with rows representing taxa and columns representing sites

Value

The function returns a matrix. Each row in the matrix represents a site pattern and the last number at each row is the frequency of the site pattern appeared in the DNA sequences.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

mrca.2nodes, mrca.nodes

Examples

```
seq<- matrix("A",nrow=4,ncol=5)
seq[1,]<-c("A","A","G","C","C")
seq[2,]<-c("A","G","G","C","C")
seq[3,]<-c("T","A","G","C","C")
seq[4,]<-c("A","A","G","T","T")
site.pattern(seq)
```

sortmat *Sort a matrix*

Description

The function returns a sorted matrix

Usage

```
sortmat(mat, columns)
```

Arguments

mat	a matrix
columns	the columns upon which the matrix is sorted

Value

The function returns a sorted matrix.

See Also

del.node

Examples

```
mat<-matrix(1:9,ncol=3)
sortmat(mat,1)
```

species.name *Species names in a tree string*

Description

The function can be used to obtain species names from a tree string.

Usage

```
species.name(str)
```

Arguments

str	a tree string in the parenthetical format

Details

The function returns the species names. If the tree string contains only the node number instead of species names, the function will return the node numbers.

Value

The function returns the species names.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

read.tree.string

Examples

```
tree.string<-"(((H:4.2,C:4.2):3.1,G:7.3):6.3,O:13.5);"
species.name(tree.string)
```

---

spstructure *Create a sequence-species relationship*

Description

This function can create a matrix to present the sequence-species relationship.

Usage

```
spstructure(numsgenenodes)
```

Arguments

numsgenenodes	number of sequences for each species

Details

The matrix created by this function can be used as species.structure.

Author(s)

Liang Liu

Examples

```
numsgenenodes<-c(1,1,1,1,1,2,2,1,1,1,1,2,3,2,2,2,1,1,1,2,1,8,2,2,2,1,1,1)
species.structure<-spstructure(numsgenenodes)
```

sptree *A species tree*

Description

a species trees

Usage

```
data(sptree)
```

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

Examples

```
data(sptree)
read.tree.nodes(sptree)
```

star.sptree *Build a STAR tree*

Description

The function can build a STAR tree from a set of gene trees.

Usage

```
star.sptree(trees, speciesname, taxaname, species.structure,outgroup,method="nj")
```

Arguments

trees	the gene tree vector
speciesname	species names
taxaname	taxa names
species.structure	a matrix defining the species-taxa relationship
outgroup	outgroup
method	UPGMA or NJ

Value

The function returns a STAR tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

mrca.2nodes, mrca.nodes

Examples

```
#create three gene trees
treestr<-rep("",4)
treestr[1]<-"((((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
treestr[2]<-"((((H:0.00402,G:0.00402):0.00304,C:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
treestr[3]<-"((((O:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,H:0.01635):0.1,W:0.11635);"
treestr[4]<-"((((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"

speciesname<-species.name(treestr[1])
taxaname<-speciesname
species.structure<-matrix(0,ncol=5,nrow=5)
diag(species.structure)<-1

star.sptree(treestr, speciesname, taxaname, species.structure,outgroup="W",method="nj")
```

---

steac.sptree *Build a STEAC tree*

Description

The function can build a STEAC tree from a set of gene trees.

Usage

```
steac.sptree(trees, speciesname, taxaname, species.structure,outgroup,method="nj")
```

Arguments

trees	the gene tree vector
speciesname	species names
taxaname	taxa names
species.structure	a matrix defining the species-taxa relationship
outgroup	outgroup
method	UPGMA or NJ

Value

The function returns a STEAC tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

mrca.2nodes, mrca.nodes

Examples

```
#create three gene trees
treestr<-rep("",4)
treestr[1]<-"((((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
treestr[2]<-"((((H:0.00402,G:0.00402):0.00304,C:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
treestr[3]<-"((((O:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,H:0.01635):0.1,W:0.11635);"
treestr[4]<-"((((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"

speciesname<-species.name(treestr[1])
taxaname<-speciesname
species.structure<-matrix(0,ncol=5,nrow=5)
diag(species.structure)<-1

steac.sptree(treestr, speciesname, taxaname, species.structure,outgroup="W",method="nj")
```

---

subtree *Subtree*

Description

The function returns the subtree under the node inode

Usage

```
subtree(inode, name, nodematrix)
```

Arguments

inode	the root node of the subtree
name	the species names
nodematrix	the tree node matrix

Value

The function returns the tree string of the subtree.

Author(s)

Liang Liu <lliu@harvard.edu>

See Also

del.node

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
spname<-read.tree.nodes(treestr)$names
subtree(7,spname,nodematrix)
```

---

subtree.length *Calculate total branch length of a tree*

Description

calculate the total branch length of a sub-tree under inode.

Usage

```
subtree.length(inode, nodes, nspecies)
```

Arguments

inode	the root node of the sub-tree
nodes	the tree node matrix
nspecies	the number of species in the tree

Details

The node matrix is the output of the function read.unrooted.nodes or read.rooted.nodes. The function can calculate the total branch length of a tree if inode is set to be the root node. If inode is not the root node, subtree.length calculates the total branch length of a sub-tree.

Value

The function returns the total branch length of a sub-tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

node.height

Examples

```
tree.string<-"(((H:4.2,C:4.2):3.1,G:7.3):6.3,O:13.5);"
nodes<-read.tree.nodes(tree.string)$nodes
subtree.length(6,nodes,4)
```

---

swap.nodes *Swap two nodes*

Description

The function swapps two subtrees.

Usage

```
swap.nodes(inode, jnode, name, nodematrix)
```

Arguments

inode	the root node of the first subtree
jnode	the root node of the second subtree
name	the species names
nodematrix	the tree node matrix

Value

nodes	the tree node matrix after swapping
treestr	the tree string after swapping

Note

The function is unable to swap two overlapped subtrees.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

del.node

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
spname<-read.tree.nodes(treestr)$names
swap.nodes(1,2,spname,nodematrix)
```

`treedist` *Distance between two trees*

Description

This function calculates the distance between two trees.

Usage

```
treedist(tree1,tree2)
```

Arguments

`tree1`	the first tree node matrix
`tree2`	the second tree node matrix

Value

The function returns the distance of two trees.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

`pair.dist`, `partition.tree`

Examples

```
treestr1<-"((((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
treestr2<-"((((H:0.00402,G:0.00402):0.00304,C:0.00706):0.00929,O:0.01635):0.1,W:0.11635);"
name<-species.name(treestr1)
nodematrix1<-read.tree.nodes(treestr1,name)$nodes
nodematrix2<-read.tree.nodes(treestr2,name)$nodes
treedist(nodematrix1,nodematrix2)
```

tripleloglike *Loglikehood of Triples*

Description

The function calculates the loglikelihood for DNA sequences (snip data)

Usage

```
tripleloglike(sptree,spname,dna)
```

Arguments

sptree	species tree
spname	species names
dna	dna sequences

Details

This function is used to calculate the loglikelihood of triples.

Value

The function returns the loglikehood of triples.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

write.subtree, read.tree.string

triplenumber *Internal function*

Description

This is an internal function used to calculate the loglikelihood of triples.

Usage

```
triplenumber(dna)
```

Arguments

dna	DNA sequences

Details

This function is used to calculate triple likelihoods.

Value

The function returns the number of triples.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

write.subtree, read.tree.string

---

triplepara *Internal function*

Description

This is an internal function used to calculate the loglikelihood of triples.

Usage

```
triplepara(inode,jnode,nodematrix,nspecies)
```

Arguments

inode	the decendant node in the triple
jnode	the ancestral node in the triple
nodematrix	the species tree
nspecies	the number of species

Details

This function is used to calculate triple likelihoods.

Value

The function returns the theta and gamma in a triple.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

write.subtree, read.tree.string

---

tripleProb *Probability of a set of rooted triples*

Description

The function calculates the probability of a set of rooted triples.

Usage

```
tripleProb(para)
```

Arguments

para	theta and gamma

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

---

unrooted.tree *An example of unrooted trees*

Description

An example of unrooted trees

Usage

```
data(unrooted.tree)
```

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

Examples

```
data(unrooted.tree)
read.tree.nodes(unrooted.tree[1])
```

unroottree *Unroot a tree*

Description

unroot a tree.

Usage

```
unroottree(nodematrix)
```

Arguments

nodematrix	the tree node matrix

Value

The function returns an unrooted tree.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

rootoftree, root.tree

Examples

```
treestr<-"((((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
spname<-read.tree.nodes(treestr)$names
unroottree(nodematrix)
```

upgma *UPGMA tree*

Description

The function computes the UPGMA tree from multiple gene trees.

Usage

```
upgma(dist, name, method="average")
```

Arguments

dist	a distance matrix
name	the species names
method	the method for recalculate pairwise distances. two options: averge or min.

Value

The function returns a tree node matrix, a tree string and species names.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

maxtree, consense, pair.dist

Examples

```
dist<-matrix(runif(25),5,5)
dist<-(dist+t(dist))/2
diag(dist)<-0
upgma(dist,name=c("H","G","C","O","W"))
```

---

write.dna *Write sequences to a Nexus file*

Description

write sequences to a Nexus file.

Usage

```
write.dna(sequence, name, file = "", format="nexus", program="mrbayes",partition=matrix(0,ncol=2,nrow
```

Arguments

sequence	DNA sequences
name	taxa names
file	output file
program	either mrbayes or best.
format	nexus or phylip
partition	each partition corresponds a gene or a locus.
clock	1:clock, 0:no clock
popmupr	for non-clock species tree model

ngen	number of generations
nrun	number of runs
nchain	number of chains
samplefreq	sampling frequency
taxa	species names if best is defined
burnin	burn in
outgroup	the node number of the outgroup
outfile	output file
append	append or not
gamma	parameters in the inverse gamma distribution as the prior of theta.

Author(s)

Liang Liu

`write.subtree` *Write a sub-tree into a string*

Description

write a tree or a sub-tree into a string in parenthetical format

Usage

```
write.subtree(inode, nodes, nspecies,inodeindex)
```

Arguments

inode	the root node of a sub-tree
nodes	a tree node matrix
nspecies	the number of species
inodeindex	the root node of a sub-tree

Details

If inode is the root of the tree, the function will write the whole tree into a string in parenthetical format. If inode is not the root node, the function will write the sub-tree into a string. The function works for both rooted trees and unrooted trees.

Value

The function returns a tree string in parenthetical format

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

See Also

write.tree.string, read.tree.nodes

Examples

```
data(rooted.tree)
tree<-read.tree.nodes(rooted.tree[1])
tree$nodes
tree$names
write.subtree(7,tree$nodes,length(tree$names),7)
```

---

write.tree.string *Write a tree file*

Description

The function writes tree strings to a file in NEXUS or PHYLIP format.

Usage

```
write.tree.string(X, format = "Nexus", file = "", name = "")
```

Arguments

X	a vector of tree strings
format	tree file format
file	the file name
name	the species names

Details

If name is provided, the function will use name as the species names in the translation block in the NEXUS tree file. Otherwise, the species names will be extracted from the tree strings.

Value

The function returns a tree file in the format of NEXUS or PHYLIP.

Author(s)

Liang Liu <lliu@oeb.harvard.edu>

References

Felsenstein, J. The Newick tree format. http://evolution.genetics.washington.edu/phylip/newicktree.html

See Also

write.subtree, read.tree.string

Index