Package ‘phybase’
August 1, 2016

Type Package
Title Basic functions for phylogenetic analysis
Version 1.5
Date 2016-7-25
Author Liang Liu
Depends R (>= 3.0.0), ape (>= 3.0), Matrix
Maintainer Liang Liu <liu@uga.edu>

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 estimate the species tree from multiple gene trees.

License GPL (>= 2)
Archs i386, x64

R topics documented:

phybase-package ......................................................... 3
ancandtime ............................................................... 4
ancestor ................................................................. 4
bootstrap ................................................................. 5
bootstrap.mulgene ....................................................... 6
change.root ............................................................ 7
ChangeBrlen ............................................................. 8
coal.spree ............................................................... 8
coaltime ................................................................. 9
concatData .............................................................. 10
control.mpest .......................................................... 10
<table>
<thead>
<tr>
<th>R topic documented:</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>del.Brlens</td>
<td>11</td>
</tr>
<tr>
<td>del.Comments</td>
<td>11</td>
</tr>
<tr>
<td>del.node</td>
<td>12</td>
</tr>
<tr>
<td>FindSpnodeDownGenenode</td>
<td>13</td>
</tr>
<tr>
<td>genetree.vector</td>
<td>13</td>
</tr>
<tr>
<td>getcoaltime</td>
<td>14</td>
</tr>
<tr>
<td>getncoal</td>
<td>14</td>
</tr>
<tr>
<td>is.clock</td>
<td>15</td>
</tr>
<tr>
<td>is.rootedtree</td>
<td>16</td>
</tr>
<tr>
<td>loglikeSP</td>
<td>16</td>
</tr>
<tr>
<td>maxxtree</td>
<td>17</td>
</tr>
<tr>
<td>mrca.2nodes</td>
<td>18</td>
</tr>
<tr>
<td>mrca.nodes</td>
<td>19</td>
</tr>
<tr>
<td>mutation_exp</td>
<td>20</td>
</tr>
<tr>
<td>name2node</td>
<td>21</td>
</tr>
<tr>
<td>nancdist</td>
<td>21</td>
</tr>
<tr>
<td>NJst</td>
<td>22</td>
</tr>
<tr>
<td>noclock2clock</td>
<td>23</td>
</tr>
<tr>
<td>node.height</td>
<td>24</td>
</tr>
<tr>
<td>node2name</td>
<td>24</td>
</tr>
<tr>
<td>offspring.nodes</td>
<td>25</td>
</tr>
<tr>
<td>offspring.nodes.string</td>
<td>26</td>
</tr>
<tr>
<td>offspring.species</td>
<td>26</td>
</tr>
<tr>
<td>output.mpest</td>
<td>27</td>
</tr>
<tr>
<td>pair.dist</td>
<td>28</td>
</tr>
<tr>
<td>pair.dist.dna</td>
<td>28</td>
</tr>
<tr>
<td>pair.dist.mulseq</td>
<td>29</td>
</tr>
<tr>
<td>partition.tree</td>
<td>30</td>
</tr>
<tr>
<td>plottree</td>
<td>31</td>
</tr>
<tr>
<td>popsize</td>
<td>32</td>
</tr>
<tr>
<td>populationMutation</td>
<td>32</td>
</tr>
<tr>
<td>postdist.tree</td>
<td>33</td>
</tr>
<tr>
<td>rank.nodes</td>
<td>34</td>
</tr>
<tr>
<td>rdirichlet</td>
<td>35</td>
</tr>
<tr>
<td>read.dna.seq</td>
<td>35</td>
</tr>
<tr>
<td>read.tree.nodes</td>
<td>36</td>
</tr>
<tr>
<td>read.tree.string</td>
<td>37</td>
</tr>
<tr>
<td>root.tree</td>
<td>38</td>
</tr>
<tr>
<td>root.trees</td>
<td>39</td>
</tr>
<tr>
<td>rooted.tree</td>
<td>39</td>
</tr>
<tr>
<td>rootoftere</td>
<td>40</td>
</tr>
<tr>
<td>sctree</td>
<td>40</td>
</tr>
<tr>
<td>sim.coaltree</td>
<td>41</td>
</tr>
<tr>
<td>sim.coaltree.sp</td>
<td>42</td>
</tr>
<tr>
<td>sim.coaltree.sp.mu</td>
<td>43</td>
</tr>
<tr>
<td>sim.dna</td>
<td>44</td>
</tr>
<tr>
<td>simnucleotide</td>
<td>45</td>
</tr>
<tr>
<td>simSeqfromSp</td>
<td>45</td>
</tr>
<tr>
<td>site.pattern</td>
<td>46</td>
</tr>
<tr>
<td>sortmat</td>
<td>47</td>
</tr>
<tr>
<td>species.name</td>
<td>47</td>
</tr>
<tr>
<td>spstructure</td>
<td>48</td>
</tr>
</tbody>
</table>
Description

This package provides functions to read, write, manipulate, simulate, 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 includes functions such as "consensus", "coaltime", "popsize", "treedist" for summarizing phylogenetic trees, calculating the coalescence time, population size, and tree distance. The function maxtree, star.sptree, and steac.sptree are built in the package to estimate the species tree from multiple gene trees. The package offers function to simulate DNA sequences from gene trees under substitution models.

Details

Package: PhyBase
Type: Package
Version: 1.1
Date: 2008-03-25
License: GPL (>=2.0.0)
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
odematrix 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 <liliu@uga.edu>

See Also

`mrca.2nodes`, `mrca.nodes`

Examples

treestr<"(((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,0:0.01635):0.1,W:0.11635);"
node <- read.tree.nodes(treestr)$nodes
ancestor(6,node)

<table>
<thead>
<tr>
<th>bootstrap</th>
<th>Bootstrap sequences</th>
</tr>
</thead>
</table>

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

```r
# 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)
```
Examples

```r
# 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 second locus.
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")
```

---

**change.root**  
*Change tree root*

**Description**

The function changes the tree root.

**Usage**

```r
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 <liu@uga.edu>

**See Also**

`root.tree`, `rootoftree`
Examples

treestr<-"(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
change.root(nodematrix,6)

**ChangeBr1en**  
*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)

coitme 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):0.00929,0:0.01635):0.1,W:0.11635);"
taxanname<species.name(treestr)
nodematrix<-read.tree.nodes(treestr,name=taxanname)$nodes
coaltime(1,2,nodematrix,5) #the coalescence time of taxa H (1) and C (2).
**concatData**

*concatenate sequences from multiple files*

**Description**

This function concatenates sequences from multiple files.

**Usage**

```r
concatData(file, spname)
```

**Arguments**

- **file**
  a list of files from which the sequences are concatenated
- **spname**
  a complete list of species’ names. Some files may have missing sequences

**Author(s)**

Liang Liu

---

**control.mpest**

*generate a control file for mpest*

**Description**

This function can generate a control file for mpest

**Usage**

```r
control.mpest(genetreefile, ngene, randomseed=-1, nrun, speciesnames, outputfile)
```

**Arguments**

- **genetreefile**
  the gene tree file
- **ngene**
  the number of genes
- **randomseed**
  the default is -1; otherwise, a random seed will be generated
- **nrun**
  the number of runs; each run has a different starting point, and mp-est will find the tree with the maximum likelihood score across all runs
- **speciesnames**
  the names of species
- **outputfile**
  the name of the control file

**Author(s)**

Liang Liu
**del.Brlens**

*Delete branch lengths from trees*

**Description**

This function deletes branch lengths from trees.

**Usage**

`del.Brlens(tree)`

**Arguments**

- `tree`  
  trees in the newick format

**Author(s)**

Liang Liu

---

**del.Comments**

*Delete comments*

**Description**

This function deletes comments in the data file.

**Usage**

`del.Comments(X)`

**Arguments**

- `X`  
  a vector of strings as the data file is read using scan

**Author(s)**

Liang Liu
**del.node**  
*Delete a node from the tree*

**Description**

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

**Usage**

```r
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**

```r
treestr<-
```

```r
spname<-read.tree.nodes(treestr)$names
nodematrix<-read.tree.nodes(treestr, spname)$nodes
del.node(6, spname, nodematrix)
```

```r
## unrooted tree
data(unrooted.tree)
name<-paste("S",1:29,sep="")
nodematrix<-read.tree.nodes(unrooted.tree[1])$nodes
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@uga.edu>

References


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, speciesNstructure)

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
- speciesNstructure: 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
```r
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)
speciesNstructure <- matrix(0,nrow=nspecies,ncol=ntax)
diag(speciesNstructure) <- 1

getchoaltime(genetree,sptree,ntax,nspecies,speciesNstructure)
```

getncoal

internal function

Description
This is an internal function for calculating the rannala and yang’s formula.
is.clock  

**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**

```r
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 lineage 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 <liu@uga.edu>

**See Also**

- `is.rootedtree`

**Examples**

```r
treestr<"(((H:0.00402,C:0.00402):0.00304,G:0.00705):0.00929,0:0.01635):0.1,W:0.11635);"

odematrix<-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 <liu@uga.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])

loglikeSP loglikelihood of the species tree, i.e., Rannala and Yang formula

Description
This function calculates the loglikelihood of a species tree from a set of gene trees using the Rannala and Yang formula.

Usage
loglikeSP(gtreetree, sptree, taxaname, spname, species.structure, strict=T)
**maxtree**

**Arguments**
- `gtree` a collection of gene trees
- `sptree` 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
- `strict` whether or not to check the result

**Value**
The function returns the log likelihood score.

**Author(s)**
Liang Liu

**References**

**Examples**
```r
sptree<-"(((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
loglikeSP(gtreet, sptree, taxaname, spname, species.structure)
```

**Description**
The function computes the Maximum Tree from multiple gene trees.

**Usage**
```r
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 <liu@uga.edu>

References

Examples
```r
genetreevector<-c("(((H:0.00302,C:0.00302):0.00304,G:0.00605):0.01029,0:0.01635):0.1,W:0.11635);", "(((H:0.00402,G:0.00402):0.00304,C:0.00705):0.00929,0:0.01635):0.1,W:0.11635);")
species.structure<-matrix(c(0.0,5,5)
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

```r
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 <liu@uga.edu>
mrca.nodes

**See Also**

mrca.nodes, coalt ime, popsize

**Examples**

```r
# Example 1
treestr<-
"(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
mrca.2nodes(1,2,nodematrix)
```

```r
# Example 2
mrca.nodes
treamer<-
"(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
mrca.nodes(c(1,2,3),nodematrix)
```


**Description**

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

**Usage**

```r
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 <liu@uga.edu>

**See Also**

mrca.2nodes, coalt ime, popsize

**Examples**

```r
# Example 1
treestr<-
"(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
mrca.2nodes(1,2,nodematrix)
```

```r
# Example 2
mrca.nodes
treamer<-
"(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
mrca.nodes(c(1,2,3),nodematrix)
```
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,W:0.12@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**

```r
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 <liu@uga.edu>

**See Also**
`subtree.length`, `node2name`

**Examples**

```r
name<="H", "G", "C", "O")
name2node(treestr, name)
```

---

**nancdist**

*Get ancestors and their divergence times*

**Description**
This function calculates the distance of two sequences on the basis of number of ancestors between two sequences.

**Usage**

```r
nancdist(tree, taxaname)
```
**Arguments**

```
tree          a tree in the Newick format
taxaname     taxa names
```

**Author(s)**

Liang Liu

**Examples**

```
treestr<"(((H:0.1,C:0.1):0.1,G:0.1):0.1,O:0.1):0.1,W:0.1);"
taxaname<-species.name(treestr)
nancdist(treestr, taxaname)
```

---

**NJst**

*calculate the NJst tree*

---

**Description**

This function can estimate species trees from a set of unrooted gene trees.

**Usage**

```
NJst(genetrees, taxaname, spname, species.structure)
```

**Arguments**

```
genetrees  a set of unrooted gene trees
taxaname   names of taxa
spname      names of species
species.structure  the taxaname-spname table
```

**Author(s)**

Liang Liu

**Examples**

```
sptree<"(A:0.4,(B:0.3,(C:0.2,(D:0.1,E:0.1):0.1):0.1):0.1));"
sptree<-species.name(sptree)
nspecies<-length(spname)
rootnode<-9
odematrix<-read.tree.nodes(sptree, spname)$node
seq<-rep(1,nspecies)
species.structure<-matrix(0,nspecies,nspecies)
diag(species.structure)<-1

##population size, theta
node(nodematrix[,5]<-0.1
ngene<5
```
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);"
names<species.name(treestr)
treenode<read.tree.nodes(treestr,name)$nodes
noclock2clock(7,treenode,4)
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

odematrix the tree node matrix

nspecies the number of species in the tree

Value

The function returns the height of inode.

Author(s)

Liang Liu <liu@uga.edu>

See Also

subtree.length

Examples

tree.string<-"(((H:4.2,C:4.2):3.1,G:7.3):6.3,0:13.5);

odematrix<-read.tree.nodes(tree.string)$nodes

node.height(6,odematrix,4)

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
**offspring.nodes**

**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**

```r
treestr <- "(((1:2,2:4,2):3,1,3:7,3):6,3,4,13,5);"
name <- c("H", "C", "G", "O")
node2name(treestr, name)
```

---

**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@uga.edu>

**See Also**

offspring.species

**Examples**

```r
treestr <- "(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):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 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 <liu@uga.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.
output.mpest

Author(s)
Liang Liu <liu@uga.edu>

See Also
offspring.nodes

Examples

treestr<-"(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):0.1,W:0.12);"
nodematrix<-read.tree.nodes(treestr)$nodes
offspring.species(7,nodematrix,5)

Description
This function can find the mpest tree with the maximum likelihood score generated from multiple runs by mpest 1.5

Usage
output.mpest(mpestfile)

Arguments

mpestfile the name of the mpest output file

Value
The function returns the mpest tree

Author(s)
Liang Liu <liu@uga.edu>
pair.dist 

**Calculate all pairwise distances among taxa in the tree**

**Description**

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

**Usage**

```r
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 <liu@uga.edu>

**See Also**

`treedist`, `upgma`, `maxtree`

**Examples**

```r
treestr<-"((H:0.00402,C:0.00402):0.00304,G:0.00705):0.00929,0:0.01635):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**

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

**Arguments**

- `sequences`: DNA sequences
- `nst`: substitution model. 0:no model, 1:JC
pair.dist.mulseq

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 <liu@uga.edu>

References


See Also

upgma

Examples

tree=paste("(((H:0.00402#0.01,C:0.00402#0.01):0.00304#0.01,G:0.00704#0.01):0.00929#0.01,O:0.01625#0.01):0.01;",
node.matrix=read.tree.nodes(tree)$nodes
sequences=sim.dna(node.matrix,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

```r
treestr <- "((H:0.00402,C:0.00402):0.00304,G:0.00705):0.00929,0:0.01635):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)
```

Description

partition a tree.

Usage

```r
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.

Author(s)

Liang Liu
Examples

```r
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,]  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 the partition.
# Each row represents a partition and its frequency.
```

---

plottree  

Write a tree file

Description

The function plots phylogenetic trees.

Usage

```r
plottree(tree)
```

Arguments

- `tree` - a phylogenetic tree in newick format

Author(s)

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

See Also

`write.subtree, read.tree.string`

Examples

```r
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**

\[
\text{popsize}(\text{inode}, \text{jnode}, \text{nodematrix})
\]

**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 <liu@uga.edu>

**See Also**

`coaltime`

**Examples**

```
treestr<"((((H:0.00402,C:0.00402#0.035):0.00304,G:0.00786):0.00929,D: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**

\[
\text{populationMutation}(\text{sptree}, \text{spnodedepth}, \text{genetree}, \text{genenodedepth}, \text{speciesmatrix})
\]
postdist.tree

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

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 <liliu@uga.edu>

See Also

See Also as read.tree.nodes
Examples

```r
code
library(phybase)
tree<-"(((H:0.005 , C:0.005 ) : 0.00025 #.01 , G:0.00525):0.00025 #.01 : O:0.0055) #.01;"
names<-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

| 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 <liu@uga.edu>

See Also

`mrca.2nodes, mrca.nodes`
**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.
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 string and the first n nodes are in the same order as the species names appear in the tree string from left to right.

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

Value

nodes  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.

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 Newick 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**

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

**Arguments**

- `file` the tree file that contains trees in Newick format.
- `format` format = "nexus" or format = "phylip"

**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 in Newick format. 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 <liu@uga.edu>
root.tree

See Also

write.tree.string, read.tree.nodes

Examples

```r
## read rooted trees in PHYLIP format
tree.string<-read.tree.string("phylip.tre",format="phylip")
```

---

## root.tree

**Root a tree**

Description

Root a tree.

Usage

```r
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 <liu@uga.edu>

See Also

rootof.tree, is.rooted.tree

Examples

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

**Description**  
This function can root phylogenetic trees using the outgroup.

**Usage**  
`root.trees(trees, outgroup)`

**Arguments**  
- `trees`: trees in the newick format
- `outgroup`: a list of possible outgroups; the first outgroup that is available in the tree will be used as the outgroup to root that tree

**Author(s)**  
Liang Liu

---

**rooted.tree**  

**Description**  
An example of rooted trees

**Usage**  
`data(rooted.tree)`

**Author(s)**  
Liang Liu <liu@uga.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**

```r
rootoftree(nodematrix)
```

**Arguments**

- `nodematrix` : the tree node matrix

**Value**

The function returns the root of the tree.

**Author(s)**

Liang Liu <liu@uga.edu>

**See Also**

`rootoftree`, `root.tree`

**Examples**

```r
treestr<-"(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):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**

```r
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 <liu@uga.edu>

References


Examples

genetreervector<-c("(((H:0.2,C:0.2):0.3,G:0.5):0.9,0:1.4):0.1,W:1.5);","(((H:0.2,G:0.2):0.4,C:0.6):0.9,0:1.5):0.1,W:1.6);");
species.structure<-matrix(0,5,5)
diag(species.structure)<-1
name<-species.name(genetreervector[1])
sctree(genetreervector,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 <liu@uga.edu>
References


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
odematrix    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. the order of the names must be consistent with that in "nodematrix"

Value

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

Author(s)

Liang Liu <liu@uga.edu>

References


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#0.01,0:0.01635#0.01)#0.01;"
spname<-species.name(tree)
nodematrix<-read.tree.nodes(tree, spname)$nodes
rootnode<-7
# define the vector seq as [2,2,2,2] which means that there are 2 sequences in each species
seq<-rep(2,4)
str<-sim.coaltree.sp(rootnode, nodematrix, 4, seq, name=spname)$gt
```

---

**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)
```
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),
  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 HKY, 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@uga.edu>

References


See Also

sim.coaltree

Examples

```r
tree<="(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,O:0.01635);"
node<-read.tree.nodes(tree)$nodes
sim.dna(node,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 <liu@uga.edu>

---

**simSeqfromSp**  
simulate DNA sequences from a species tree

---

**Description**

The function simulates sequences from a species tree. The function assumes that seq-gen has been installed.

**Usage**

```
simSeqfromSp(sptree, spname, ntaxasp, ngene, theta=0, noclock=0, simsequence=1, murate="Dirichlet", alpha=5, seqlength=100, rate=c(1,1,1,1,1,1), frequency=c(1/4,1/4,1/4,1/4), outfile, format="phylip", concat=TRUE)
```

**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
- **rate**: rates
- **frequency**: nucleotide frequency
- **outfile**: the full path of the output file
- **format**: either "phylip" or "nexus"
- **concat**: save the concatenated sequences or save single-gene sequences as different data in the same file
The function writes sequences into a file.

Author(s)
Liang Liu <liliu@uga.edu>

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 <liliu@uga.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**

```r
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**

```r
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**

```r
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 <liu@uga.edu>

See Also
read.tree.string

Examples
```
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**

```
umsgenenodes<-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

da species trees

Usage

data(sptree)

Author(s)

Liang Liu <liu@uga.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. Although STAR can handle missing sequences, it requires that all possible pairs of species (n choose 2) should appear in at least one gene tree. Otherwise, STAR cannot calculate the pairwise distances among species.

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 <liu@uga.edu>
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 <liu@uga.edu>

See Also
mrca.2nodes, mrca.nodes
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 <liu@harvard.edu>

See Also

del.node

Examples

treestr<"(((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,0:0.01635):0.1,W:0.11635);"

```r
treestr[1]<"(((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,0:0.01635):0.1,W:0.11635);"
treestr[2]<"(((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,H:0.01635):0.1,W:0.11635);"
treestr[3]<"(((0:0.00402,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):0.00929,0:0.01635):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.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 <liu@uga.edu>

**See Also**

`node.height`

**Examples**

```r

nodes<-read.tree.nodes(tree.string)$nodes
subtree.length(6,nodes,4)
```
### Description
The function swaps 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 <liu@uga.edu>

### See Also
`del.node`

### Examples
```r
treestr <- "(((H:0.00402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):0.1,W:0.12);"
nodematrix <- read.tree.nodes(treestr)$nodes
spname <- read.tree.nodes(treestr)$names
swap.nodes(1,2,spname,nodematrix)
```
tredist  

**Distance between two trees**

**Description**

This function calculates the distance between two trees.

**Usage**

```
tredist(tree1, tree2)
```

**Arguments**

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

**Value**

The function returns the RF distance of two trees.

**Author(s)**

Liang Liu <liu@uga.edu>

**See Also**

- `pair.dist.partition.tree`

**Examples**

```
treestr1<-"(((H:0.00402,C:0.00402):0.00304,G:0.00706):0.00929,0:0.01635):0.1,W:0.11635);"
treestr2<-"(((H:0.00402,G:0.00402):0.00304,C:0.00706):0.00929,0:0.01635):0.1,W:0.11635);"
name<-species.name(treestr1)
nodematrix1<-read.tree.nodes(treestr1,name)$nodes
odematrix2<-read.tree.nodes(treestr2,name)$nodes
tredist(nodematrix1,nodematrix2)
```

---

**tripleloglike**  

**Loglikelihood 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 loglikelihood of triples.

**Author(s)**

Liang Liu <liu@uga.edu>

**See Also**

`write.subtree`, `read.tree.string`

---

**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 <liu@uga.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 <liu@uga.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 <liu@uga.edu>
unrooted.tree

An example of unrooted trees

Description
An example of unrooted trees

Usage
data(unrooted.tree)

Author(s)
Liang Liu <liu@uga.edu>

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

unroottree
Unroot a tree

Description
unroot a tree.

Usage
unroottree(nodematrix)

Arguments

odematrix the tree node matrix

Value
The function returns an unrooted tree.

Author(s)
Liang Liu <liu@uga.edu>

See Also
rootoftree, root.tree

Examples
treestr="(((H:0.0402,C:0.00402):0.00304,G:0.00707):0.00929,0:0.01635):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**

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

**Arguments**

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

**Value**

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

**Author(s)**

Liang Liu <liu@uga.edu>

**See Also**

`maxtree, pair.dist`

**Examples**

```r
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.seq  

**Write sequences to a Nexus file**

**Description**

write sequences to a Nexus file.

**Usage**

```r
write.dna.seq(sequence, name, file = "", format="nexus",
program="mrbayes", partition=matrix(0,ncol=2,nrow=1),
clock=0, popmupr=0, ngen=1000000,nrun=1,nchain=1,samplefreq=100,
taxa=as.vector,burnin=1000,gamma="(3,0.02)",
outgroup=1,outfile="",append = FALSE)
```
write.seq.phylip

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

Description

This function writes concatenated sequences to a file.

Usage

```r
write.seq.phylip(sequence, name, length, outfile = "", append=FALSE)
```

Arguments

- `sequence` concatenated sequences as strings
- `name` species names
- `length` the length of sequences per line in the output file
- `outfile` output file
- `append` FALSE or TRUE

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, nodematrix, taxaname, root)

Arguments

inode the root node of a sub-tree

nodematrix a tree node matrix

taxaname taxa names

root 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@uga.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$nodes, tree$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 <liu@uga.edu>

References

See Also
write.subtree, read.tree.string
Index

*Topic **IO**
- plotree, 31
- rdirichlet, 35
- read.tree.nodes, 36
- read.tree.string, 37
- tripleloglike, 54
- tripavenum, 55
- triplicable, 56
- write.subtree, 60
- write.tree.string, 61

*Topic **datasets**
- rooted.tree, 39
- sptree, 49
- unrooted.tree, 57

*Topic **programming**
- ancandtime, 4
- ancestor, 4
- bootstrap, 5
- bootstrap.mulgene, 6
- change.root, 7
- ChangeBrLen, 8
- coal.sptree, 8
- coaltime, 9
- concatData, 10
- control.mpest, 10
- del.Brlens, 11
- del.Comments, 11
- del.node, 12
- FindSnpNodeDownGenenode, 13
- genetree.vector, 13
- getcoaltime, 14
- getncoal, 14
- is.clock, 15
- is.rootedtree, 16
- loglikeSP, 16
- maxtree, 17
- mrca.2nodes, 18
- mrca.nodes, 19
- mutation_exp, 20
- nancdist, 21
- NJst, 22
- noclock2clock, 23
- offspring.nodes, 25
- offspring.nodes.string, 26
- offspring.species, 26
- output.mpest, 27
- pair.dist, 28
- pair.dist.dna, 28
- pair.dist.mulseq, 29
- partition.tree, 30
- phybase-package, 3
- popsize, 32
- populationMutation, 32
- postdist.tree, 33
- rank.nodes, 34
- read.dna.seq, 35
- root.tree, 38
- root.trees, 39
- rootoftree, 40
- sctree, 40
- sim.coaltree, 41
- sim.coaltree.sp, 42
- sim.coaltree.sp.mu, 43
- sim.dna, 44
- simnucleotide, 45
- simSeqFromSp, 45
- site.pattern, 46
- sortmat, 47
- species.name, 47
- spstructure, 48
- star.sptree, 49
- steac.sptree, 50
- subtree, 51
- swap.nodes, 53
- treedist, 54
- tripleProb, 56
- unroottree, 57
- upgma, 58
- write.dna.seq, 58
- write.seq.phylip, 59

*Topic **univar**
- name2node, 21
- node.height, 24
- node2name, 24
- subtree.length, 52
- ancandtime, 4
ancestor, 4
bootstrap, 5, 6
bootstrapNmulgene, 6
change.root, 7, 12
ChangeBrLen, 8
coil.sptree, 8
coaltime, 9, 19, 32
concatData, 10
control.mpest, 10
del.BrLens, 11
del.Comments, 11
del.node, 12, 47, 51, 53
FindSpNodeDownGenenode, 13
genetree.vector, 13
getcoaltime, 14
getncoal, 14
is.clock, 15, 16
is.rootedtree, 15, 16, 38
loglikeSP, 16
maxtree, 13, 17, 28, 58
mrca.2nodes, 5, 18, 19, 34, 46, 50
mrca.nodes, 5, 19, 19, 34, 46, 50
mutation.exp, 20
name2node, 21, 25
nancdist, 21
NJst, 22
noclock2clock, 23
node.height, 24, 52
node2name, 21, 24
offspring.nodes, 25, 27
offspring.nodes.string, 26
offspring.species, 25, 26
output.mpest, 27
pair.dist, 28, 30, 54, 58
pair.dist.dna, 28
pair.dist.mulseq, 29
partition.tree, 30, 54
PhyBase (phybase-package), 3
phybase-package, 3
plottree, 31
popsize, 9, 19, 32
populationMutation, 32
postdist.tree, 33
rank.nodes, 34
dirichlet, 35
read.dna.seq, 35
read.tree.nodes, 33, 36, 37, 38, 60
read.tree.string, 31, 36, 37, 48, 55, 56, 61
root.tree, 7, 38, 40, 57
root.trees, 39
rooted.tree, 39
rootoftree, 7, 38, 40, 40, 57
sctree, 40
sim.coaltree, 41, 42, 44
sim.coaltree.sp, 42, 42
sim.coaltree.sp.m, 43
sim.dna, 44
simnucleotide, 45
simSeqFromSp, 45
site.pattern, 46
sortmat, 47
species.name, 36, 47
spstructure, 48
sptree, 49
star.sptree, 49
steac.sptree, 8, 50
subtree, 51
subtree.length, 21, 24, 25, 52
swap.nodes, 12, 53
treedist, 28, 54
tripleloglike, 54
triplenumber, 55
triplepara, 56
tripleProb, 56
unrooted.tree, 57
unroottree, 57
upgma, 28, 29, 58
write.dna.seq, 58
write.seq.phylip, 59
write.subtree, 31, 55, 56, 60, 61
write.tree.string, 37, 38, 60, 61