Package 'MSCquartets'

January 7, 2021

Title Analyzing Gene Tree Quartets under the Multi-Species Coalescent

Version 1.1.0

Description Methods for analyzing and using quartets displayed on a collection of gene trees, primarily to make inferences about the species tree or network under the multi-species coalescent model. These include quartet hypothesis tests for the model, as developed by Mitchell et al. (2019) <doi:10.1214/19-EJS1576>, simplex plots of quartet concordance factors as presented by Allman et al. (2020) <doi:10.1101/2020.02.13.948083>, species tree inference methods
based on quartet distances of Rhodes (2019) <doi:10.1109/TCBB.2019.2917204> and Yourdkhani and Rhodes (2019) <doi:10.1007/s11538-020-00773-4>, and the NANUQ algorithm for inference of level-1
species networks of Allman et al. (2019) <doi:10.1186/s13015-019-0159-2>. Software announcement by
Rhodes et al. (2020) <doi:10.1093/bioinformatics/btaa868>.

License MIT + file LICENSE

Imports RandomFieldsUtils, zipfR, graphics, stats, Rdpack, foreach, doParallel

RdMacros Rdpack

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Depends R (>= 3.2.0), ape (>= 5.0), phangorn

Author Elizabeth Allman [aut],
Hector Banos [aut],
Jonathan Mitchell [aut],
John Rhodes [aut, cre] (<https://orcid.org/0000-0001-9921-1091>)

Maintainer John Rhodes <j.rhodes@alaska.edu>

Repository CRAN

Date/Publication 2021-01-07 21:00:02 UTC

R topics documented:

`MSCquartets-package` *Multispecies Coalescent Model Quartet Package*

Description

A package for analyzing quartets displayed on gene trees, under the multispecies coalescent (MSC) model.

Details

This package contains routines to analyze a collection of gene trees through the displayed quartets on them.

A quartet count concordance factor (qcCF) for a set of 4 taxa is the triple of counts of the three possible resolved quartet trees on those taxa across some set of gene trees. The major routines in this package can:

1. Tabulate all qcCFs for a collection of gene trees.
2. Perform hypothesis tests of whether one or more qcCFs are consistent with the MSC model on a species tree (Mitchell et al. 2019).
3. Produce simplex plots showing all estimated CFs as well as results of hypothesis tests (Allman et al. 2020).
4. Infer a species tree using the qcCFs via the QDC and WQDC methods (Rhodes 2020; Yourdkhani and Rhodes 2020).
5. Infer a level-1 species network via the NANUQ method (Allman et al. 2019).

As discussed in the cited works, the inference methods for species trees and networks are statistically consistent under the MSC and Network MSC respectively.

This package, and the theory on which it is based, allows gene trees to have missing taxa (i.e., not all gene trees display all the taxa). It does require that each subset of 4 taxa is displayed on at least one of the gene trees.

Several gene tree data sets, simulated and empirical, are included.

In publications please cite the software announcement (Rhodes et al. 2020), as well as the appropriate paper(s) developing the theory behind the routines you used.

References

Rhodes JA, Baños H, Mitchell JD, Allman ES (2020). "MSCquartets 1.0: Quartet methods for species trees and networks under the multispecies coalescent model in R." *Bioinformatics*. doi: 10.1093/bioinformatics/btaa868, https://doi.org/10.1093/bioinformatics/btaa868.

Mitchell J, Allman ES, Rhodes JA (2019). "Hypothesis testing near singularities and boundaries." *Electron. J. Statist.*, **13**(1), 2150–2193. doi: 10.1214/19EJS1576, https://doi.org/10.1214/19-EJS1576.

Allman ES, Mitchell JD, Rhodes JA (2020). "Gene tree discord, simplex plots, and statistical tests under the coalescent." *bioRxiv*. doi: 10.1101/2020.02.13.948083, https://doi.org/10.1101/2020.02.13.948083.

Rhodes JA (2020). "Topological metrizations of trees, and new quartet methods of tree inference." *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **17**(6), 2107–2118. doi: 10.1109/TCBB.2019.2917204, https://doi.org/10.1109/TCBB.2019.2917204.

Yourdkhani S, Rhodes JA (2020). "Inferring metric trees from weighted quartets via an intertaxon distance." *Bul. Math. Biol.*, **82**(97). doi: 10.1007/s11538020007734, https://doi.org/10.1007/s11538-020-00773-4.

Allman ES, Baños H, Rhodes JA (2019). "NANUQ: A method for inferring species networks from gene trees under the coalescent model." *Algorithms Mol. Biol.*, **14**(24), 1–25. doi: 10.1186/s13015-01901592, https://doi.org/10.1186/s13015-019-0159-2.

`dataGeneTreeSample` *Simulated gene tree dataset from species tree*

Description

A text file dataset containing 1000 gene trees on 9 taxa simulated under the MSC on a species tree

Format

A text file with 1000 metric Newick gene trees on the taxa t1-t9

Details

This simulated dataset was produced by SimPhy (Mallo et al. 2016), using the species tree

((((t5:5000,t6:5000):5000,t4:10000):2500,t7:12500):7500,((t8:3000,t9:3000):5000, ((t1:4000,t2:4000):2500,t3:6500):1500):12000);

with a population size of 10,000 throughout the tree.

File is accessed as `system.file("extdata","dataGeneTreeSample",package="MSCquartets")`, for example via the ape command:

```
gts=read.tree(file = system.file("extdata","dataGeneTreeSample",package="MSCquartets")
)
```

References

Mallo D, De Oliveira Martins L, Posada D (2016). "SimPhy: Phylogenomic Simulation of Gene, Locus, and Species Trees." *Syst. Biol.*, **65**(2), 334–344. doi: 10.1093/sysbio/syv082, https://doi.org/10.1093/sysbio/syv082.

dataHeliconiusMartin *Heliconius gene tree dataset*

Description

A text file dataset for Heliconius butterflies containing 2909 gene trees on 7 taxa, with 4 individuals sampled for each of 3 of the taxa, for a total of 16 leaves per gene tree. This is a subset of the data of Martin et al. (2013).

Format

A text file with 2909 metric Newick gene trees each with 16 leaves labelled:
chioneus.553, chioneus.560, chioneus.564, chioneus.565,
ethilla.67, hecale.273, melpomeneFG.13435, melpomeneFG.9315,
melpomeneFG.9316, melpomeneFG.9317, pardalinus.371, rosina.2071,
rosina.531, rosina.533, rosina.546, sergestus.202

Details

File is accessed as `system.file("extdata","dataHeliconiusMartin",package="MSCquartets")`, for example via the ape command:

```
gts = read.tree(file=system.file("extdata","dataHeliconiusMartin",package="MSCquartets"))
```

Source

doi: 10.5061/dryad.dk712

References

Martin SH, K.K. D, Nadeau NJ, Salazar C, Walters JR, Simpson F, Blaxter M, Manica A, Mallet J, Jiggins CD (2013). "Genome-wide evidence for speciation with gene flow in Heliconius butterflies." *Genome Res*, **23**, 1817–1828.

dataYeastRokas *Yeast gene tree dataset*

Description

A text file dataset for Yeast containing 106 gene trees on 8 taxa (7 Saccharomyces and 1 Candida outgroup). This is a subset of the data of Rokas et al. (2003).

Format

A text file with 106 topological Newick gene trees on the taxa: Sbay, Scas, Scer, Sklu, Skud, Smik, Spar, and Calb (outgroup).

Details

File is accessed as `system.file("extdata","dataYeastRokas",package="MSCquartets")`, for example via the ape command:

```
gts=read.tree(file = system.file("extdata","dataYeastRokas",package="MSCquartets"))
```

Source

https://wiki.rice.edu/confluence/download/attachments/8898533/yeast.trees?version=1&modificationDate=1360603275797&api=v2

References

Rokas A, Williams B, Carrol S (2003). "Genome-scale approaches to resolving incongruence in molecular phylogenies." *Nature*, **425**, 798–804.

---

estimateEdgeLengths *Estimate edge lengths on a species tree from gene tree quartet counts*

Description

Estimate edge lengths, in coalescent units, on an unrooted species tree from a table of resolved quartet counts from a collection of gene trees.

Usage

```
estimateEdgeLengths(tree, rqt, terminal = 1, method = "simpleML", lambda = 1/2)
```

Arguments

`tree`	a phylo object, giving a resolved tree on which to estimate edge lengths
`rqt`	a resolved quartet table, as from `quartetTableResolved`, in which all taxa on `tree` appear
`terminal`	an edge length to assign to terminal edges, whose lengths cannot be estimated
`method`	`"simpleML"` or `"simpleBayes"`
`lambda`	a positive parameter for the `"simpleBayes"` method

Details

While the argument `tree` may be supplied as rooted or unrooted, metric or topological, only its unrooted topology will be used for determining new metric estimates.

Counts of quartets for all those quartets which define a single edge on the tree (i.e., whose internal edge is the single edge on the unrooted input tree) are summed, and from this an estimate of the branch length is computed. If `method="simpleML"` this is the maximum likelihood estimate. If `method="simpleBayes"` this is the Bayesian estimate of Theorem 2 of Sayyari and Mirarab (2016), using parameter `lambda`. Using `lambda=1/2` gives a flat prior on [1/3,1] for the probability of the quartet displayed on the species tree.

These methods are referred to as 'simple' since they use only the quartets defining a single edge of the species tree. Quartets with central edges composed of several edges in the species tree are ignored.

Note that branch length estimates may be 0 (if the count for the quartet displayed on the input tree is not dominant), positive, or `Inf` (if the counts for quartet topologies not displayed on the input tree are all 0, and `method="simpleML"`).

Value

an unrooted metric tree with the same topology as `tree`, of type phylo

References

Sayyari E, Mirarab S (2016). "Fast Coalescent-Based Computation of Local Branch Support from Quartet Frequencies." *Mol. Biol. Evol.*, **33**(7), 1654–1668.

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
taxanames=taxonNames(gtrees)
QT=quartetTable(gtrees,taxanames[1:6])
RQT=quartetTableResolved(QT)
DQT=quartetTableDominant(RQT)
tree=QDS(DQT)
write.tree(tree)
plot(tree)
metricMTree=estimateEdgeLengths(tree,RQT,method="simpleML")
write.tree(metricMTree)
plot(metricMTree)
metricBTree=estimateEdgeLengths(tree,RQT,method="simpleBayes")
write.tree(metricBTree)
plot(metricBTree)
```

---

expectedCFs — *Produce table of expected quartet concordance factors for a species tree*

Description

Compiles table of expected quartet concordance factors (CFs) for gene trees under the MSC model on a metric species tree.

Usage

```
expectedCFs(speciestree, plot = TRUE, model = "T1", cex = 1)
```

Arguments

speciestree	phylo or character object specifying un/rooted metric species tree
plot	TRUE (default) to produce simplex plot of CFs, or FALSE to omit plot
model	"T1" or "T3" specifying model for plot
cex	scaling factor for size of plotted symbols

Details

The species tree may be rooted or unrooted, but must have edge lengths given in coalescent units. Note that while the MSC requires a rooted metric species tree parameter, as shown in (Allman et al. 2011) the quartet CFs are independent of the root.

In the returned table, columns are labeled by taxon names and quartet names ("12|34", etc.). 1s and 0s in taxon columns indicate the taxa in a quartet. Quartet 12|34 means the first and second indicated taxa form a cherry, 13|24 means the first and third form a cherry, and 14|23 means the first and fourth form a cherry. Unresolved quartets always have expectation 0 under the MSC.

If a simplex plot is produced, for the T1 model all CFs will lie on the vertical model line, and many choices of 4 taxa will give the same CFs. For model T3 the model lines the CFs are plotted on depend on taxon names and are essentially arbitrary.

Value

an (n choose 4)x(n+3) matrix encoding 4 taxon subsets of taxa and expectation of each of the quartets 12|34, 13|24, 14|23 across gene trees

References

Allman ES, Degnan JH, Rhodes JA (2011). "Identifying the rooted species tree from the distribution of unrooted gene trees under the coalescent." *Journal of Mathematical Biology*, **62**(6), 833–862. doi: 10.1007/s0028501003557, https://doi.org/10.1007/s00285-010-0355-7.

See Also

quartetTable, quartetTableResolved

Examples

```
stree="((((t5:5000,t6:5000):5000,t4:10000):2500,t7:12500):7500,((t8:3000,t9:3000):5000,
((t1:4000,t2:4000):2500,t3:6500):1500):12000);"
st=read.tree(text=stree)
st$edge.length=st$edge.length/10000
expectedCFs(st)
```

`HolmBonferroni` *Apply Holm-Bonferroni method to adjust for multiple tests*

Description

Apply the Holm-Bonferroni method to adjust for multiple hypothesis tests performed on quartets from a data set of gene trees.

Usage

```
HolmBonferroni(pTable, model, alpha = 0.05)
```

Arguments

`pTable`	a table of quartets with p-values, as computed by `quartetTreeTestInd` or `quartetStarTestInd`
`model`	one of `"T1"`, `"T3"`, or `"star"`, where `pTable` contains a column `p_model` of p-values
`alpha`	a critical value, for rejection of adjusted p-values less than or equal to `alpha`

Details

When p-values are computed for each quartet using `quartetTreeTestInd` or `quartetStarTestInd`, multiple comparisons are being done for one dataset. The Holm-Bonferroni method (Holm 1979) adjusts these p-values upward, controlling the familywise error rate. The probability of at least one false discovery (rejection of the null hypothesis) is no more than the significance level.

The Holm-Bonferroni method does not require that test hypotheses are independent, which is important for its application to quartet counts presumed to have arisen on a single species tree.

This can be a low power test (often failing to reject when the null hypothesis is false). In particular for the T1 and T3 tests, it does not consider the relationships between edge lengths for different sets of four taxa.

Value

the same table, with rows reordered, and 2 new columns of 1) adjusted p-values, and 2) "Y" or "N" for indicating "reject" or "fail to reject"

References

Holm S (1979). "A simple sequentially rejective multiple test procedure." *Scand. J. Statist.*, **6**(2), 65–70.

See Also

`quartetTreeTestInd`, `quartetStarTestInd`

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
taxanames=taxonNames(gtrees)
QT=quartetTable(gtrees,taxanames[1:6])
RQT=quartetTableResolved(QT)
pTable=quartetTreeTestInd(RQT,"T3")
pTable[1:10,]
HBpTable=HolmBonferroni(pTable,"T3",.05)
HBpTable[1:10,]
```

NANUQ *Apply NANUQ network inference algorithm to gene tree data*

Description

Apply the NANUQ algorithm of Allman et al. (2019) to infer a hybridization network from a collection of gene trees, under the level-1 network multispecies coalescent (NMSC) model.

Usage

```
NANUQ(
  genedata,
  outfile = "NANUQdist",
  omit = FALSE,
  epsilon = 0,
  alpha = 0.05,
  beta = 0.95,
  taxanames = NULL,
  plot = TRUE
)
```

Arguments

genedata	gene tree data that may be supplied in any of 3 forms: 1. as a character string giving the name of a file containing Newick gene trees, 2. as a multiPhylo object containing the gene trees, or 3. as a table of quartets on the gene trees, as produced by a previous call to NANUQ or quartetTableResolved, which has columns only for taxa, resolved quartet counts, and possibly p_T3 and p_star
outfile	a character string giving an output file name stub for saving a NANUQ distance matrix in nexus format; to the stub outfile will be appended an alpha and beta value and ".nex"; if NULL then then no file is written
omit	FALSE to treat unresolved quartets as 1/3 of each resolution; TRUE to discard unresolved quartet data; ignored if gene tree data given as quartet table

epsilon	minimum for branch lengths to be treated as non-zero; ignored if gene tree data given as quartet table
alpha	a value or vector of significance levels for judging p-values testing a null hypothesis of no hybridization vs. an alternative of hybridization, for each quartet; a smaller value applies a less conservative test for a tree (more trees), hence a stricter requirement for desciding in favor of hybridization (fewer reticulations)
beta	a value or vector of significance levels for judging p-values testing a null hypothesis of a star tree (polytomy) for each quartet vs. an alternative of anything else; a smaller value applies a less conservative test for a star tree (more polytomies), hence a stricter requirement for deciding in favor of a resolved tree or network; if vectors, alpha and beta must have the same length
taxanames	if genedata is a file or a multiPhylo object, a vector of a subset of the taxa names on the gene trees to be analyzed, if NULL all taxa on the first gene tree are used; if genedata is a quartet table, this argument is ignored and all taxa in the table are used
plot	TRUE produces simplex plots of hypothesis test results, FALSE omits plots

Details

This function

1. counts displayed quartets across gene trees to form quartet count concordance factors (qcCFs),
2. applies appropriate hypothesis tests to judge qcCFs as representing putative hybridization, resolved trees, or unresolved (star) trees using alpha and beta as significance levels,
3. produces a simplex plot showing results of the hypothesis tests for all qcCFs
4. computes the appropriate NANUQ distance table, writing it to a file.

The distance table file can then be opened in the external software SplitsTree (Huson and Bryant 2006) (recommended) or within R using the package phangorn to obtain a circular split system under the Neighbor-Net algorithm, which is then depicted as a splits graph. The splits graph should be interpreted via the theory of Allman et al. (2019) to infer the level-1 species network, or to conclude the data does not arise from the NMSC on such a network.

If alpha and beta are vectors, they must have the same length k. Then the i-th entries are paired to produce k plots and k output files. This is equivalent to k calls to NANUQ with scalar values of alpha and beta.

A call of NANUQ with genedata given as a table previously output from NANUQ is equivalent to a call of NANUQdist. If genedata is a table previously output from quartetTableResolved which lacks columns of p-values for hypothesis tests, these will be appended to the table output by NANUQ.

If plots are produced, each point represents an empirical quartet concordance factor, color-coded to represent test results.

In general, alpha should be chosen to be small and beta to be large so that most quartets are interpreted as resolved trees.

Usually, an initial call to NANUQ will not give a good analysis, as values of alpha and beta are likely to need some adjustment based on inspecting the data. Saving the returned table from NANUQ will allow for the results of the time-consuming computation of qcCFs to be saved, along with p-values, for input to further calls of NANUQ with new choices of alpha and beta.

See the documentation for quartetNetworkDist for an explanation of a small, rarely noticeable, stochastic element of the algorithm.

For data sets of many gene trees, user time may be reduced by using parallel code for counting displayed quartets. See quartetTableParallel, where example commands are given.

Value

a table of quartets and p-values for judging fit to the MSC on quartet trees (returned invisibly); this table can be used as input to NANUQ or NANUQdist with new choices of alpha and beta, without re-tallying quartets on gene trees; a distance table to be used as input for SplitsTree is written to a nexus file

References

Allman ES, Baños H, Rhodes JA (2019). "NANUQ: A method for inferring species networks from gene trees under the coalescent model." *Algorithms Mol. Biol.*, **14**(24), 1–25. doi: 10.1186/s13015-01901592, https://doi.org/10.1186/s13015-019-0159-2.

Huson DH, Bryant D (2006). "Application of Phylogenetic Networks in Evolutionary Studies." *Molecular Biology and Evolution*, **23**(2), 254–267.

See Also

quartetTable, quartetTableParallel, quartetTableDominant, quartetTreeTestInd, quartetStarTestInd, NANUQdist, quartetTestPlot, pvalHist, quartetNetworkDist

Examples

```
pTable=NANUQ(system.file("extdata", "dataYeastRokas",package="MSCquartets"),
   alpha=.0001, beta=.95, outfile = file.path(tempdir(), "NANUQdist"))
NANUQ(pTable, alpha=.05, beta=.95,outfile = file.path(tempdir(), "NANUQdist"))
# The distance table was written to an output file for opening in SplitsTree.
# Alternately, to use the experimental phangorn implementation of NeighborNet
# within R, enter the following additional lines:
dist=NANUQdist(pTable, alpha=.05, beta=.95,outfile = file.path(tempdir(), "NANUQdist"))
nn=neighborNet(dist)
plot(nn,"2D")
```

NANUQdist *Compute NANUQ distance and write to file*

Description

Computes the quartet distance tables for the NANUQ algorithm of Allman et al. (2019), using precomputed p-values for quartets, for each of several levels specified. Distance tables are written to files, in nexus format.

Usage

```
NANUQdist(
  pTable,
  outfile = "NANUQdist",
  alpha = 0.05,
  beta = 0.95,
  plot = TRUE
)
```

Arguments

pTable	a table of resolved quartets and p-values, as previously computed by NANUQ, or by both quartetTreeTestInd and quartetStarTestInd, with columns "p_T3" and "p_star"
outfile	a character string giving an output file name stub for saving a NANUQ distance matrix in nexus format; to the stub outfile will be appended an alpha and beta value and ".nex"; if NULL then a distance matrix is not computed
alpha	a value or vector of significance levels for judging p-values testing a null hypothesis of no hybridization for each quartet; a smaller value applies a more liberal test for a tree (more trees), hence a stricter requirement for suspecting hybridization (fewer reticulations)
beta	a value or vector of significance levels for judging p-values testing a null hypothesis of a star tree for each quartet; a smaller value applies a more liberal test for a star tree (more polytomies), hence a stricter requirment for suspecting a resolved tree; if vectors, alpha and beta must have the same length
plot	TRUE produces simplex plots of hypothesis tests, FALSE omits plots

Details

If plots are produced, each point represents an empirical quartet concordance factor, color-coded to represent test results giving interpretation as network, resolved tree, or star tree.

If alpha and beta are vectors, they must be of the same length k. Then the i-th entries are paired to produce k plots and k distance tables/output files. This is equivalent to k calls to NANUQdist with paired scalar values from the vectors of alpha and beta.

See the documentation for quartetNetworkDist for an explanation of a small, rarely noticeable, stochastic element of the algorithm.

Value

a NANUQ distance table, or a list of such tables if alpha and beta are vectors (returned invisibly)

References

Allman ES, Baños H, Rhodes JA (2019). "NANUQ: A method for inferring species networks from gene trees under the coalescent model." *Algorithms Mol. Biol.*, **14**(24), 1–25. doi: 10.1186/s13015-01901592, https://doi.org/10.1186/s13015-019-0159-2.

See Also

NANUQ, quartetTreeTestInd, quartetStarTestInd

Examples

```
pTable=NANUQ(system.file("extdata","dataYeastRokas",package="MSCquartets"),
      alpha=.0001, beta=.95, outfile = file.path(tempdir(), "NANUQdist"))
NANUQdist(pTable, alpha=.05, beta=.95, outfile = file.path(tempdir(), "NANUQdist"))
```

---

nexusDist *Write a distance table to a file in nexus format*

Description

Write a distance table to a file in nexus format.

Usage

```
nexusDist(distMatrix, outfilename)
```

Arguments

distMatrix	a square matrix giving a distance table, with rows and columns labeled by taxon names
outfilename	the name of an output file

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
DQT=quartetTableDominant(RQT)
Dist=quartetDist(DQT)
nexusDist(Dist,outfile = file.path(tempdir(), "NANUQdist"))
```

powerDivStat *Power divergence statistic of Cressie & Read*

Description

Computes any of the family of power-divergence statistics for multinomial data of Cressie and Read (1984), to compare observed and expected counts. Includes Likelihood Ratio and Chi-squared statistics as special cases.

Usage

```
powerDivStat(obs, expd, lambda)
```

Arguments

obs	observation vector
expd	expected vector
lambda	statistic parameter (e.g., 0=Likelihood Ratio, 1=Chi-squared)

Value

value of statistic

References

Cressie N, Read TRC (1984). "Multinomial Goodness-Of-Fit Tests." *J. Royal Stat. Soc. B*, **46**(3), 440–464.

Examples

```
obs=c(10,20,30)
expd=c(20,20,20)
powerDivStat(obs,expd,0)
```

pvalHist *Plot histogram of log p-values in table*

Description

Graphical exploration of extreme p-values from quartet hypothesis tests, to aid in choosing critical values for hypothesis tests. Log base 10 of p-values exceeding some minimum are plotted, to explore gaps in the tail of the distribution.

Usage

```
pvalHist(pTable, model, pmin = 0)
```

Arguments

pTable	a quartet table with p-values, such as from NANUQ, quartetTreeTestInd, or quartetStarTestInd
model	one of "T1", "T3", or "star", where pTable contains a column p_model of p-values
pmin	include only p-values above pmin in the histogram

Details

Since logarithms are plotted, p-values close to 0 will appear as negative numbers of large magnitude, putting the tail of the distribution to the left in the histogram.

When exploring possible critical values for the hypothesis tests in the NANUQ algorithm, use model= "T3" to choose values for alpha which distinguish treelikeness from hybridization, and model= "star" to choose values for beta which distinguish polytomies from resolved trees. In general, alpha should be chosen to be small and beta to be large so that most quartets are interpreted as resolved trees.

See Also

NANUQ, NANUQdist

Examples

```
pTable=NANUQ(system.file("extdata","dataYeastRokas",package="MSCquartets"),
       alpha=0.05, beta=.95, outfile = file.path(tempdir(), "NANUQdist"))
pvalHist(pTable,"T3")
```

QDC *Compute Quartet Distance Consensus tree from gene tree data*

Description

Compute the Quartet Distance Consensus (Rhodes 2020) estimate of an unrooted topological species tree from gene tree data.

Usage

```
QDC(
  genetreedata,
  taxanames = NULL,
  method = fastme.bal,
  omit = FALSE,
  metric = FALSE
)
```

Arguments

genetreedata	gene tree data that may be supplied in any of 3 forms: 1. a character string giving the name of a file containing gene trees in Newick, 2. a multiPhylo object containing gene trees, or 3. a resolved quartet table, such as produced by quartetTableResolved
taxanames	if genetreedata is a file or a multiPhylo object, a vector of a subset of the taxa names on the gene trees to be analyzed, if NULL all taxa on the first gene tree are used; if genetreedata is a quartet table, this argument is ignored and all taxa in the table are used
method	a distance-based tree building function, such as fastme.bal or nj
omit	TRUE ignores unresolved quartets; FALSE treats them as 1/3 of each resolution; ignored if genetreedata is supplied as a quartet table
metric	if FALSE return topological tree; if TRUE return metric tree with internal edge lengths estimated by estimateEdgeLengths with lambda=0, and terminal branches of length 1

Details

This function is a wrapper which performs the steps of reading in a collection of gene trees, tallying quartets, computing the quartet distance between taxa, building a tree which consistently estimates the unrooted species tree topology under the MSC, and then possibly estimating edge lengths using the "simpleML" method.

Value

an unrooted tree of type phylo

References

Rhodes JA (2020). "Topological metrizations of trees, and new quartet methods of tree inference." *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **17**(6), 2107–2118. doi: 10.1109/TCBB.2019.2917204, https://doi.org/10.1109/TCBB.2019.2917204.

See Also

quartetTable, quartetTableResolved, quartetTableDominant, quartetDist, QDS, WQDC, WQDCrecursive estimateEdgeLengths

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
stree=QDC(gtrees,tnames[1:6])
write.tree(stree)
plot(stree)
streeMetric=QDC(gtrees, tnames[1:6],metric=TRUE)
write.tree(streeMetric)
plot(streeMetric)
```

QDS *Compute Quartet Distance Supertree*

Description

Apply the Quartet Distance Supertree method of Rhodes (2020) to a table specifying a collection of quartets on n taxa.

Usage

```
QDS(dqt, method = fastme.bal)
```

Arguments

dqt	an (n choose 4) x n (or n+1) matrix of form output by quartetTableDominant; (Note: If present, the n+1th column of dqt is ignored)
method	tree building function (e.g., fastme.bal, nj)

Details

This function is a wrapper which runs quartetDist and then builds a tree.

Value

an unrooted metric tree of type phylo. Edge lengths are not in interpretable units

References

Rhodes JA (2020). "Topological metrizations of trees, and new quartet methods of tree inference." *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **17**(6), 2107–2118. doi: 10.1109/TCBB.2019.2917204, https://doi.org/10.1109/TCBB.2019.2917204.

See Also

quartetTableDominant, quartetDist, QDC, WQDS, WQDC, WQDCrecursive

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
DQT=quartetTableDominant(RQT)
tree=QDS(DQT)
write.tree(tree)
plot(tree)
```

quartetDist *Compute quartet distance between taxa*

Description

Compute the Quartet Distance of Rhodes (2020) from a table specifying a collection of quartets on n taxa.

Usage

```
quartetDist(dqt)
```

Arguments

dqt	an (n choose 4) x n (or n+1) matrix of form output by quartetTableDominant; (Note: If present, the n+1th column of dqt is ignored.)

Value

a pairwise distance matrix on n taxa

References

Rhodes JA (2020). "Topological metrizations of trees, and new quartet methods of tree inference." *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **17**(6), 2107–2118. doi: 10.1109/TCBB.2019.2917204, https://doi.org/10.1109/TCBB.2019.2917204.

See Also

quartetTableDominant, QDS, QDC, quartetWeightedDist

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
DQT=quartetTableDominant(RQT)
Dist=quartetDist(DQT)
tree=NJ(Dist)
write.tree(tree)
plot(tree)
```

`quartetNetworkDist` *Compute network quartet distance between taxa*

Description

Produce network quartet distance table for the NANUQ algorithm, from a table of quartets and p-values, and specified levels of quartet hypothesis tests. The network quartet distance, which is described more fully by Allman et al. (2019), generalizes the quartet distance of Rhodes (2020).

Usage

```
quartetNetworkDist(pTable, alpha, beta)
```

Arguments

`pTable`	a table of quartets and p-values, as computed by NANUQ, or `quartetTreeTestInd` and `quartetStarTestInd`
`alpha`	a scalar significance level for judging p-values `p_T3` indicating hybridization on quartet; smaller value gives fewer hybridization decisions
`beta`	a scalar significance level for judging p-values `p_star` indicating quartet star tree; smaller value gives fewer resolved tree decisions

Details

In case of a triple of quartet counts with the two largest equal and the third slighltly smaller, along with `alpha` and `beta` leading to a star quartet being rejected and a tree not being rejected, this function chooses a resolved quartet topology uniformly at random from the two largest counts. This is the only stochastic element of the code, and its impact is usually negligable.

Value

a distance table

References

Allman ES, Baños H, Rhodes JA (2019). "NANUQ: A method for inferring species networks from gene trees under the coalescent model." *Algorithms Mol. Biol.*, **14**(24), 1–25. doi: 10.1186/s13015-01901592, https://doi.org/10.1186/s13015-019-0159-2.

Rhodes JA (2020). "Topological metrizations of trees, and new quartet methods of tree inference." *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **17**(6), 2107–2118. doi: 10.1109/TCBB.2019.2917204, https://doi.org/10.1109/TCBB.2019.2917204.

See Also

NANUQ, NANUQdist

Examples

```
pTable=NANUQ(system.file("extdata", "dataYeastRokas",package="MSCquartets"), alpha=.0001,
       beta=.95,outfile = file.path(tempdir(), "NANUQdist"))
dist=quartetNetworkDist(pTable, alpha=.05, beta=.95)
dist
```

`quartetStarTest` *Hypothesis test for quartet counts fitting a star tree under the MSC*

Description

Perform hypothesis test for star tree for a vector of quartet counts to fit expected frequencies of (1/3,1/3,1/3). The test performed is a standard Chi-square with 2 degrees of freedom.

Usage

```
quartetStarTest(obs)
```

Arguments

`obs`	vector of 3 counts of resolved quartet frequencies

Value

p-value

Examples

```
obs=c(16,72,12)
quartetStarTest(obs)
```

`quartetStarTestInd` *Multiple independent hypothesis tests for gene quartet counts fitting a species quartet star tree under the MSC*

Description

Perform hypothesis tests for a species quartet star tree vs. any alternative for all quartet counts in an input table, as if the quartets are independent.

Usage

```
quartetStarTestInd(rqt)
```

Arguments

rqt	Table of resolved quartet counts, as produced by quartetTableResolved, or quartetTreeTestInd

Details

This function assumes all quartets are resolved. The test performed is described in quartetStarTest.

Value

the same table as the input rqt with column "p_star" appended, containing p-values for judging fit to MSC on a star tree

See Also

quartetStarTest, quartetTreeTest, quartetTreeTestInd, quartetTableResolved, quartetTestPlot

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
pTable=quartetStarTestInd(RQT)
quartetTablePrint(pTable[1:6,])
```

---

quartetTable *Produce table of counts of quartets displayed on trees*

Description

Compiles table of quartet count concordance factors (qcCFs) for topological quartets displayed on a collection of trees.

Usage

```
quartetTable(trees, taxonnames = NULL, epsilon = 0, random = 0)
```

Arguments

trees	multiphylo object containing un/rooted metric/topological trees
taxonnames	vector of n names of taxa of interest; if NULL then taken from taxa on trees[[1]]
epsilon	minimum for branch lengths to be treated as non-zero
random	number of random subsets of 4 taxa to consider; if 0, use all n choose 4 subsets

Details

The names in taxonnames may be any subset of those on the trees. Branch lengths of non-negative size less than or equal to epsilon are treated as zero, giving polytomies.

In the returned table, columns are labeled by taxon names and quartet names ("12|34", etc.). 1s and 0s in taxon columns indicate the taxa in a quartet. Quartet 12|34 means the first and second indicated taxa form a cherry, 13|24 means the first and third form a cherry, 14|23 means the first and fourth form a cherry, and 1234 means the quartet is unresolved.

An error occurs if any branch length is negative. Warnings are given if some of taxonnames are missing on some trees, or if some 4-taxon set is not on any tree.

If random>0, then for efficiency random should be much smaller then the number of possible 4 taxon subsets.

Value

an (n choose 4)x(n+4) matrix (or (random)x(n+4) matrix) encoding 4 taxon subsets of taxonnames and counts of each of the quartets 12|34, 13|24, 14|23, 1234 across the trees

See Also

quartetTableParallel, quartetTableResolved, quartetTableDominant, taxonNames

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
DQT=quartetTableDominant(RQT)
```

---

quartetTableCollapse *Reduce quartet table by combining some taxa*

Description

Form a smaller resolved quartet table by lumping some taxa into a composite taxon.

Usage

```
quartetTableCollapse(rqt, taxaA, taxaB)
```

Arguments

rqt	a resolved quartet table, as from quartetTableResolved
taxaA	a vector of taxon names in rqt to be included in the output table
taxaB	a vector of taxon names in rqt to form new composite taxon in the output table

Details

This function is needed for the recursive calls in `WQDSrec`. It should only be applied to a resolved quartet table which includes counts for all possible quartets on the taxa (though counts can be zero).

The sets `taxaA` and `taxaB` must be disjoint. Their union need not be all taxa in `rqt`.

Value

a resolved quartet table with `length(taxaA)+1` taxa; the composite taxon is named as the concatenation of the sorted names in `taxaB`

See Also

`WQDCrecursive`

`quartetTableDominant` *Produce table of dominant quartets, with estimates of internal edge lengths*

Description

Converts table of counts of resolved quartets on `n` taxa to show only dominant one, with maximum likelihood estimate of internal edge weight under the MSC.

Usage

```
quartetTableDominant(rqt, bigweights = "infinite")
```

Arguments

`rqt`	a table, as produced by `quartetTableResolved` of size (n choose 4)x(n+3);
`bigweights`	`"infinite"` or `"finite"`, to indicate whether the weight (internal edge length) of a quartet for which only one topology appears is given as `Inf` or a finite, but large, numerical value

Details

If `bigweights="finite"`, when for a set of 4 taxa the quartet counts are (m,0,0) then the edge weight is computed as if the relative frequency of the dominant topology were m/(m+1).

Value

an (n choose 4)x(n+1) array with dominant quartet topology encoded by 1,1,-1,-1 in taxon columns, with signs indicating cherries; the (n+1)th column `"weight"` contains the maximum likelihood estimates, under MSC on a 4-taxon tree, of the quartets' central edge lengths, in coalescent units

See Also

`quartetTable`, `quartetTableResolved`

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
RQT[1:6,]
DQT=quartetTableDominant(RQT)
DQT[1:6,]
```

`quartetTableParallel` *Produce table of counts of quartets displayed on trees, in parallel for large data sets*

Description

Compiles table of quartet count concordance factors (qcCFs) for topological quartets displayed on a collection of trees. Gives the same output as `quartetTable`, but operates in parallel.

Usage

```
quartetTableParallel(
  trees,
  taxonnames = NULL,
  epsilon = 0,
  random = 0,
  numCores
)
```

Arguments

`trees`	multiphylo object containing un/rooted metric/topological trees
`taxonnames`	vector of n names of taxa of interest; if `NULL` then taken from taxa on `trees[[1]]`
`epsilon`	minimum for branch lengths to be treated as non-zero
`random`	number of random subsets of 4 taxa to consider; if 0, use all n choose 4 subsets
`numCores`	number of cores to use for parallel calls

Details

The number of available cores can be determined by `parallel::detectCores()`. With overhead, tabulating quartets for a large data set (many taxa and/or many gene trees) on a 4-core computer using `numCores=4` may require less than half the elapsed time of the sequential `quartetTable`.

The names in `taxonnames` may be any subset of those on the trees. Branch lengths of non-negative size less than or equal to `epsilon` are treated as zero, giving polytomies.

In the returned table, columns are labeled by taxon names and quartet names ("12|34", etc.). 1s and 0s in taxon columns indicate the taxa in a quartet. Quartet 12|34 means the first and second

indicated taxa form a cherry, 13|24 means the first and third form a cherry, 14|23 means the first and fourth form a cherry, and 1234 means the quartet is unresolved.

An error occurs if any branch length is negative. Warnings are given if some of `taxonnames` are missing on some trees, or if some 4-taxon set is not on any tree.

If `random`>0, then for efficiency `random` should be much smaller then the number of possible 4 taxon subsets.

If the quartet counts are to be used for NANUQ, or any other routines requiring resolved quartet counts, `quartetTableResolved` must be run following `quartetTableParallel`. See example below.

Value

an (n choose 4)x(n+4) matrix (or (`random`)x(n+4) matrix) encoding 4 taxon subsets of `taxonnames` and counts of each of the quartets 12|34, 13|24, 14|23, 1234 across the trees

See Also

`quartetTable`, `quartetTableResolved`, `quartetTableDominant`, `taxonNames`

Examples

```
gtrees=read.tree(file=system.file("extdata","dataHeliconiusMartin",package="MSCquartets"))
N <- parallel::detectCores()
QT=quartetTableParallel(gtrees,numCores=N)
RQT=quartetTableResolved(QT)
pTable=NANUQ(RQT,alpha=1e-40, beta=1e-30, outfile = file.path(tempdir(), "NANUQdist"))
```

---

quartetTablePrint *Print a quartet table with nice formatting*

Description

Print a quartet table with the taxa in each quartet shown by name.

Usage

```
quartetTablePrint(qt)
```

Arguments

`qt`	a table such as returned by `quartetTable`, `quartetTableResolved`, or `quartetTableDominant`, possibly with extra columns added by other functions

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
QT[1:6,]
quartetTablePrint(QT[1:6,])
RQT=quartetTableResolved(QT)
RQT[1:6,]
quartetTablePrint(RQT[1:6,])
pTable=quartetTreeTestInd(RQT,"T3")
pTable[1:6,]
quartetTablePrint(pTable[1:6,])
DQT=quartetTableDominant(RQT)
DQT[1:6,]
quartetTablePrint(DQT[1:6,])
```

`quartetTableResolved` *Modify quartet table to show only resolved quartets*

Description

Converts table of all quartet counts, including unresolved ones, by either dropping unresolved ones, or distributing them uniformly among the three resolved counts.

Usage

```
quartetTableResolved(qt, omit = FALSE)
```

Arguments

`qt`	table, as produced by `quartetTable` for `n` taxa, with `n+4` columns
`omit`	`TRUE` deletes unresolved quartets column; `FALSE` deletes the column but redistributes unresolved counts as (1/3,1/3,1/3) to resolved counts

Value

a table of quartet counts similar to `qt`, but with columns showing only resolved quartet counts

See Also

`quartetTable`, `quartetTableDominant`

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
QT[1:6,]
RQT=quartetTableResolved(QT)
RQT[1:6,]
```

---

quartetTestPlot *Produce simplex plot with results of quartet hypothesis tests*

Description

Plot a 2-d probability simplex, with points for all quartet count vectors. Colors indicate rejection or failure to reject for tests at specified levels.

Usage

```
quartetTestPlot(pTable, test, alpha = 0.05, beta = 1, cex = 1)
```

Arguments

pTable	table of quartets and p-values, as produced by quartetTreeTestInd, quartetStarTestInd, or NANUQ
test	model to use, for tree null hypothesis; options are "T1" or "T3"
alpha	significance level for tree test with null hypothesis given by test
beta	significance level for test with null hypothesis star tree; test results plotted only if beta<1 and "p_star" column present in pTable
cex	scaling factor for size of plotted symbols

Details

The first argument of this function is a table of quartets and p-values. The plot may show results of either the T1 or T3 test, with or without a star tree test (depending on whether a "p_star" column is in the table and/or beta =1). The p-values must be computed by previous calls to quartetTreeTestInd (for "T1" or "T3" p-values) and quartetStarTestInd (for "star" p-values). The NANUQ and NANUQdist functions include calls to these tree test functions.

See Also

quartetTreeTestInd, quartetStarTestInd, NANUQ, NANUQdist

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=c("t1","t2","t3","t4","t5","t6")
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
stree="(((t5,t6),t4),((t1,t2),t3));"
pTable=quartetTreeTestInd(RQT,"T1",speciestree=stree)
pTable=quartetStarTestInd(pTable)
quartetTestPlot(pTable, "T1", alpha=.05, beta=.95)
```

---

quartetTreeErrorProb *Bayesian posterior probability of error in 4-taxon unrooted species tree topology estimate*

Description

From a gene quartet count concordance factor (qcCF), computes Bayesian posterior probabilities of the three 4-taxon species tree topologies and the Bayesian posterior probability that the assumed topology is incorrect, under the assumption that the counts arise from the MSC on some species tree.

Usage

```
quartetTreeErrorProb(obs, model = "T3")
```

Arguments

`obs`	vector of counts for 3 topologies
`model`	`"T3"` or `"T1"`, for the models of Mitchell et al. (2019) describing an unspecified species tree topology (`"T3"`), or the topology whose count is the first entry of `obs` (`"T1"`)

Details

The Jeffreys prior is used for internal branch length, along with the uniform prior on the resolved topology.

Value

`(error.prob,top.probs)` where `error.prob` is the species tree error probability and `top.probs` is a vector of the three species tree topology probabilities in the order of `obs`; for model `"T1"` the species tree used is the one corresponding to the first count; for model `"T3"` the species tree is the one corresponding to the largest count

References

Mitchell J, Allman ES, Rhodes JA (2019). "Hypothesis testing near singularities and boundaries." *Electron. J. Statist.*, **13**(1), 2150–2193. doi: 10.1214/19EJS1576, https://doi.org/10.1214/19-EJS1576.

Examples

```
obs <- c(28,32,30)
quartetTreeErrorProb(obs,model="T1")
quartetTreeErrorProb(obs,model="T3")
```

`quartetTreeTest` *Hypothesis test for quartet counts fitting a tree under the MSC*

Description

Test the hypothesis H_0= T1 or T3 model of Mitchell et al. (2019), vs. H_1 = everything else. T1 is for a specific species quartet topology, and T3 for any species quartet topology.

Usage

```
quartetTreeTest(
  obs,
  model = "T3",
  lambda = 0,
  method = "MLest",
  smallcounts = "approximate",
  bootstraps = 10^4
)
```

Arguments

`obs`	vector of 3 counts of resolved quartet frequencies
`model`	`"T1"` or `"T3"`, for the models of Mitchell et al. (2019)
`lambda`	parameter for power-divergence statistic (e.g., 0 for likelihood ratio statistic, 1 for Chi-squared statistic)
`method`	`"MLtest"`,`"conservative"`, or `"bootstrap"`
`smallcounts`	`"bootstrap"` or `"approximate"`, method of obtaining p-value when some counts are small
`bootstraps`	number of samples for bootstrapping

Details

This function implements two of the versions of the test given by Mitchell et al. (2019) as well as parametric boostrapping, with other procedures for when some expected counts are small. When the topology and/or the internal quartet branch length is not specified by the null hypothesis these are more accurate tests than, say, a Chi-square with one degree of freedom, which is not theoretically justified near the singularities and boundaries of the models.

If `method="MLtest"`, this uses the test by that name described in Section 7 of Mitchell et al. (2019). For both the T1 and T3 models the test is slightly anticonservative over a small range of true internal edges of the quartet species tree. Although the test generally performs well in practice, it lacks a uniform asymptotic guarantee over the full parameter space for either T1 or T3.

If `method="conservative"`, a conservative test described by Mitchell et al. (2019) is used. For model T3 this uses the Chi-square distribution with 1 degree of freedom (the "least favorable" approach), while for model T1 it uses the Minimum Adjusted Bonferroni, based on precomputed values from simulations with n=1e+6. These conservative tests are asymptotically guaranteed to reject the null hypothesis at most at a specified level, but at the expense of increased type II errors.

If `method="bootstrap"`, then parametric bootstrapping is performed, based on parameter estimates of the quartet topology and internal edge length. The bootstrap sample size is given by the `bootstrap` argument.

When some expected topology counts are small, the methods `"MLest"` and `"conservative"` are not appropriate. The argument `smallcounts` determines whether bootstrapping or a faster approximate method is used. These both involve estimates of the quartet topology and internal edge length. The approximate approach returns a precomputed p-value, found by replacing the largest observed count with 1e+6 and performing 1e+8 bootstraps for the model T3. When n is sufficiently large (at least 30) and some expected counts are small, the quartet tree error probability is small and the bootstrap p-value is approximately independent of the choice of T3 or T1 and of the largest observed count.

For model T1, the first entry of `obs` is treated as the count of gene quartets concordant with the species tree.

The returned p-value should be taken with caution when there is a small sample size, e.g. less than 30 gene trees. The returned value of `bl` is a consistent estimator, but not the MLE, of the internal edge length in coalescent units. Although consistent, the MLE for t is biased. Our consistent estimator is still biased, but with less bias than the MLE. See Mitchell et al. (2019) for more discussion on dealing with the bias of parameter estimates in the presence of boundaries and/or singularities of parameter spaces.

Value

`(p-value,bl)` where `bl` is a consistent estimator of the internal edge length in coalescent units, possibly `Inf`.

References

Mitchell J, Allman ES, Rhodes JA (2019). "Hypothesis testing near singularities and boundaries." *Electron. J. Statist.*, **13**(1), 2150–2193. doi: 10.1214/19EJS1576, https://doi.org/10.1214/19-EJS1576.

See Also

quartetTreeTestInd

Examples

```
quartetTreeTest(c(17,72,11),"T3")
quartetTreeTest(c(17,72,11),"T1")
quartetTreeTest(c(72,11,17),"T1")
quartetTreeTest(c(11,17,72),"T1")
```

---

quartetTreeTestInd *Multiple independent hypothesis tests for quartet counts fitting a species tree under the MSC*

Description

Perform a tree hypothesis test for all quartet counts in an input table, as if the counts for different choices of 4 taxa are independent.

Usage

```
quartetTreeTestInd(
  rqt,
  model = "T3",
  lambda = 0,
  method = "MLest",
  smallcounts = "approximate",
  bootstraps = 10^4,
  speciestree = NULL
)
```

Arguments

rqt	table of resolved quartet counts, as produced by quartetTableResolved, or quartetStarTestInd
model	"T1" for a specific species tree topology, or "T3" for any species tree topology, with these models explained more fully by Mitchell et al. (2019)
lambda	power divergence statistic parameter (e.g., 0 for likelihood ratio statistic, 1 for Chi-squared statistic)
method	"MLest", "conservative", or "bootstrap"; see quartetTreeTest for explanation
smallcounts	"bootstrap" or "approximate", method of obtaining p-value when some counts are small, so the chosen method is inappropriate
bootstraps	number of samples for bootstrapping
speciestree	species tree, in Newick as text, to determine quartet for T1 test; required for model="T1", ignored for model="T3"

Details

This function assumes all quartets are resolved. The test performed and the arguments are described more fully in `QuartetTreeTest`.

Value

if `model="T3"`, a copy of `rqt` with a new column `"p_T3"` appended with p-values for each quartet; if `model="T1"`, a copy of `rqt` with 2 columns appended: `"p_T1"` with p-values, and `"qindex"` giving index of quartet consistent with specified species tree, i.e., 1 if 12|34 on species tree, 2 if 13|24, 3 if 14|23

References

Mitchell J, Allman ES, Rhodes JA (2019). "Hypothesis testing near singularities and boundaries." *Electron. J. Statist.*, **13**(1), 2150–2193. doi: 10.1214/19EJS1576, https://doi.org/10.1214/19-EJS1576.

See Also

`quartetTreeTest`, `quartetTestPlot`, `quartetStarTestInd`, `quartetTableResolved`

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=c("t1","t2","t3","t4","t5","t6")
QT=quartetTable(gtrees,tnames)
RQT=quartetTableResolved(QT)
stree="(((t5,t6),t4),((t1,t2),t3));"
pTable3=quartetTreeTestInd(RQT,"T3")
quartetTablePrint(pTable3[1:6,])
stree="((((t5,t6),t4),t7),((t8,t9),((t1,t2),t3)));"
pTable1=quartetTreeTestInd(RQT,"T1",speciestree=stree)
quartetTablePrint(pTable1[1:6,])
```

---

quartetWeightedDist *Compute the Weighted Quartet Distance between taxa*

Description

Compute the Weighted Quartet Distance between taxa of Yourdkhani and Rhodes (2020) from a table specifying a collection of quartets on `n` taxa and the quartets' internal branch lengths.

Usage

```
quartetWeightedDist(dqt)
```

Arguments

dqt	an (n choose 4) x (n+1) matrix of the form output by quartetTableDominant

Value

a pairwise distance matrix on n taxa

References

Yourdkhani S, Rhodes JA (2020). "Inferring metric trees from weighted quartets via an intertaxon distance." *Bul. Math. Biol.*, **82**(97). doi: 10.1007/s11538020007734, https://doi.org/10.1007/s11538-020-00773-4.

See Also

quartetTableDominant, WQDSAdjustLengths, WQDS, WQDC, WQDCrecursive, quartetWeightedDist

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
DQT=quartetTableDominant(RQT,bigweights="finite")
D=quartetWeightedDist(DQT)
tree=NJ(D)
stree=WQDSAdjustLengths(tree)
write.tree(stree)
```

---

simplexCoords *Convert 3-d coordinates to 2-d probability simplex coordinates*

Description

Convert from 3-d Cartesian coordinates to 2-d coordinates suitable for plotting in the probability simplex.

Usage

```
simplexCoords(v)
```

Arguments

v	vector of 3 non-negative numbers, not summing to 0

Details

Applies an affine coordinate trandformation that maps the centroid (1/3,1/3,1/3) to the origin (0,0), and rescales so that the line segments between (1,0,0), (0,1,0), and (0,0,1) are mapped to segments of length 1.

An input vector v is first normalized so its component sum to 1 before the map is applied.

Value

2-d coordinates to plot normalized point in simplex

See Also

simplexLabels, simplexPoint, simplexPrepare, simplexSegment, simplexText

Examples

```
simplexCoords(c(15,65,20))
```

---

simplexLabels *Label vertices of 2-d probability simplex*

Description

Add labels to vertices of the probability simplex.

Usage

```
simplexLabels(top = "", left = "", right = "")
```

Arguments

top	label for top
left	label for left bottom
right	label for right bottom

See Also

simplexPoint, simplexPrepare, simplexSegment, simplexText, simplexCoords

Examples

```
simplexPrepare("T3","Example Plot")
simplexLabels("ab|cd","ac|bd","ad|bc")
```

simplexPoint *Plot point in 2-d probability simplex*

Description

Normalizes a point given in 3-d non-normalized coordinates, then plots it in the 2-d probability simplex.

Usage

```
simplexPoint(v, ...)
```

Arguments

v	a 3-d point in non-negative orthant, coordinates not summing to 0
...	other options to pass to graphics::points function

See Also

simplexLabels, simplexPrepare, simplexSegment, simplexText, simplexCoords

Examples

```
simplexPrepare("T3","Example Plot")
simplexPoint(c(15,65,20),pch=3,col="blue")
```

simplexPrepare *Draw 2-d probability simplex, with model lines for T3 or T1 model*

Description

Outline the 2-d probability simplex, and draw the T1 or T3 model points for quartet frequencies. The models "T1" and "T3" are described more fully by Mitchell et al. (2019).

Usage

```
simplexPrepare(model = "T3", maintitle = NULL, titletext = NULL)
```

Arguments

model	"T1" or "T3", for 1-tree or 3-tree model
maintitle	main title for plot
titletext	additional text for title

References

Mitchell J, Allman ES, Rhodes JA (2019). "Hypothesis testing near singularities and boundaries." *Electron. J. Statist.*, **13**(1), 2150–2193. doi: 10.1214/19EJS1576, https://doi.org/10.1214/19-EJS1576.

See Also

simplexLabels, simplexPoint, simplexSegment, simplexText, simplexCoords

Examples

```
simplexPrepare("T3",maintitle="Main title",titletext="further text")
```

`simplexSegment` *Plot line segment in 2-d probability simplex*

Description

Normalizes two points in 3-d, and draws line segment between them in 2-d probability simplex.

Usage

```
simplexSegment(v, w, ...)
```

Arguments

`v, w`	3-d endpoints of line segment in non-negative orthant, coords not summing to 0
`...`	other options to pass to graphics::segments function

See Also

simplexLabels, simplexPoint, simplexPrepare, simplexText, simplexCoords

Examples

```
simplexPrepare("T3","Example Plot")
simplexSegment(c(15,65,20),c(15,70, 15),col="green")
```

simplexText *Add text at a point in 2-d probability simplex*

Description

Add text to a 2-d probability simplex plot, at specified location.

Usage

```
simplexText(v, label = "", ...)
```

Arguments

v	a 3-d point in non-negative orthant, coordinates not summing to 0
label	text to add to plot
...	other options to pass to graphics::text function

See Also

simplexLabels, simplexPoint, simplexPrepare, simplexSegment, simplexCoords

Examples

```
simplexPrepare("T3","Example Plot")
simplexText(c(15,65,20),"tree ac|bd")
```

T1density *Probability density function for Model T1 of Mitchell et al. (2019), Proposition 5.2*

Description

Value of probability density function for Model T1 of Mitchell et al. (2019), Proposition 5.2.

Usage

```
T1density(x, mu0)
```

Arguments

x	statistic value (e.g., likelihood ratio statistic, or other power divergence statistic)
mu0	parameter of density function

Value

value of density function

References

Mitchell J, Allman ES, Rhodes JA (2019). "Hypothesis testing near singularities and boundaries." *Electron. J. Statist.*, **13**(1), 2150–2193. doi: 10.1214/19EJS1576, https://doi.org/10.1214/19-EJS1576.

See Also

T3density

`T3density` *Probability density function for Model T3 of Mitchell et al. (2019), Proposition 4.2*

Description

Value of probability density function for Model T3 of Mitchell et al. (2019), Proposition 4.2.

Usage

```
T3density(x, mu0, alpha0, beta0)
```

Arguments

`x`	statistic value (e.g., likelihood ratio statistic, or other power divergence statistic)
`mu0`	parameter of density function
`alpha0`	parameter of density function
`beta0`	parameter of density function

Value

value of density function

References

Mitchell J, Allman ES, Rhodes JA (2019). "Hypothesis testing near singularities and boundaries." *Electron. J. Statist.*, **13**(1), 2150–2193. doi: 10.1214/19EJS1576, https://doi.org/10.1214/19-EJS1576.

See Also

T1density

taxonNames *Get all taxon names from a collection of trees*

Description

Create a vector of all taxon names appearing on a collection of trees, with no repeats.

Usage

```
taxonNames(trees)
```

Arguments

trees	a multiPhylo object containing a collection of trees

Value

a vector of unique names of taxa appearing on the trees

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
```

WQDC *Compute Weighted Quartet Distance Consensus tree from gene tree data*

Description

Compute the Weighted Quartet Distance Consensus (Yourdkhani and Rhodes 2020) estimate of a species tree from gene tree data. This is a consistent estimator of the unrooted species tree topology and all internal branch lengths.

Usage

```
WQDC(
  genetreedata,
  taxanames = NULL,
  method = fastme.bal,
  omit = FALSE,
  terminal = 1
)
```

Arguments

genetreedata	gene tree data that may be supplied in any of 3 forms: 1. a character string giving the name of a file containing gene trees in Newick 2. a multiPhylo object containing gene trees 3. a resolved quartet table, as produced by quartetTableResolved
taxanames	if genetreedata is a file or a multiPhylo object, a vector of a subset of the taxa names on the gene trees to be analyzed, if NULL all taxa on the first gene tree are used; if genetreedata is a quartet table, this argument is ignored and all taxa in the table are used
method	a distance-based tree building function, such as fastme.bal or nj
omit	TRUE leaves out unresolved quartets, FALSE treats them as 1/3 of each resolution; ignored if genetreedata is given as a resolved quartet table
terminal	non-negative branch length to supply for terminal branches whose length cannot be inferred by WQDC

Details

This function is a wrapper which performs the steps of reading in a collection of gene trees, tallying quartets, estimating quartet internal branch lengths, computing the weighted quartet distance between taxa, building a tree, and adjusting edge lengths, to give a consistent estimate of the metric species tree in coalescent units under the MSC.

If the gene tree data indicates some quartets experienced little to no incomplete lineage sorting, this algorithm tends to be less topologically accurate than QDC (which infers no metric information) or WQDCrecursive (which gives better topologies, and reasonably accurate lengths for short edges, though long edge lengths may still be unreliable).

Value

an unrooted metric tree of type phylo

References

Yourdkhani S, Rhodes JA (2020). "Inferring metric trees from weighted quartets via an intertaxon distance." *Bul. Math. Biol.*, **82**(97). doi: 10.1007/s11538020007734, https://doi.org/10.1007/s11538-020-00773-4.

See Also

quartetTable, quartetTableResolved, quartetTableDominant, quartetWeightedDist, WQDCrecursive, WQDS, QDC

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
stree=WQDC(gtrees,tnames[1:6])
write.tree(stree)
```

```
plot(stree)
```

`WQDCrecursive` *Compute the Recursive Weighted Quartet Distance Consensus tree from gene tree data*

Description

Infer a metric species tree from counts of quartets displayed on a collection of gene trees, as described by Yourdkhani and Rhodes (2020). Edge lengths are in coalescent units.

Usage

```
WQDCrecursive(rqt, method = fastme.bal, stopAt = 2, terminal = 1)
```

Arguments

`rqt`	a resolved quartet table as produced by `quartetTableResolved`
`method`	a distance-based tree building function, such as `fastme.bal` or `nj`
`stopAt`	a non-negative branch length in coalescent units; recursive calls stop when the longest branch in a recursively examined subtree is smaller than this value
`terminal`	non-negative branch length to supply for terminal branches, whose lengths cannot be inferred by `WQDCrecursive`

Details

The algorithm counts quartets displayed on the gene trees, builds a tree using `WQDS`, determines the split corresponding to the longest edge in that tree, and then recursively builds trees on the taxa in each split set together with a 'composite taxon' formed by all taxa in the other split set. This approach is slower than non-recursive `WQDC`, but increases topological accuracy. Shorter branch lengths tend to be more accurately estimated.

This function must be called with its argument a resolved quartet table of size (n choose 4)x(n+3). Its recursive nature requires building smaller resolved quartet tables on split sets with an additional composite taxon.

Value

an unrooted metric tree, of type phylo

References

Yourdkhani S, Rhodes JA (2020). "Inferring metric trees from weighted quartets via an intertaxon distance." *Bul. Math. Biol.*, **82**(97). doi: 10.1007/s11538020007734, https://doi.org/10.1007/s11538-020-00773-4.

See Also

quartetTableResolved,quartetTable, QDC, QDS, quartetTableCollapse

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
stree=WQDCrecursive(RQT)
write.tree(stree)
plot(stree)
```

WQDS *Compute the Weighted Quartet Distance Supertree*

Description

Apply the Weighted Quartet Distance Supertree method of Yourdkhani and Rhodes (2020) to a collection of quartets on n taxa together with internal quartet branch lengths, specified by a table.

Usage

```
WQDS(dqt, method = fastme.bal)
```

Arguments

dqt	an (n choose 4) x n+1) matrix of form output by quartetTableDominant
method	a distance-based tree building function (e.g., fastme.bal, NJ, etc.)

Details

This function is a wrapper which runs quartetWeightedDist, builds a tree, and then adjusts edge lengths with WQDSAdjustLengths.

Value

an unrooted metric tree, of type phylo

References

Yourdkhani S, Rhodes JA (2020). "Inferring metric trees from weighted quartets via an intertaxon distance." *Bul. Math. Biol.*, **82**(97). doi: 10.1007/s11538020007734, https://doi.org/10.1007/s11538-020-00773-4.

See Also

quartetTableDominant, quartetWeightedDist, WQDSAdjustLengths, WQDC, WQDCrecursive, QDS

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
DQT=quartetTableDominant(RQT,bigweights= "finite")
tree=WQDS(DQT)
write.tree(tree)
plot(tree)
```

WQDSAdjustLengths *Adjust edge lengths on tree built from Weighted Quartet distance to estimate metric tree*

Description

Modify edge lengths of a tree built from a distance table produced by quartetWeightedDist, to remove scaling factors related to the size of the split associated to the edge.

Usage

```
WQDSAdjustLengths(tree)
```

Arguments

tree	an unrooted metric tree, of type phylo

Details

As explained by Yourdkhani and Rhodes (2020), a metric tree produced from the weighted quartet distance has edge lengths inflated by a factor dependent on the associated split size. Removing these factors yields a consistent estimate of the metric species tree displaying the weighted quartets, if such a tree exists.

This function should not be used on trees output from WQDS, WQDC, or WQDCrecursive, as their edges are already adjusted. It can be used on trees built from the distance computed by quartetWeightedDist.

Value

an unrooted metric tree, of type phylo

References

Yourdkhani S, Rhodes JA (2020). "Inferring metric trees from weighted quartets via an intertaxon distance." *Bul. Math. Biol.*, **82**(97). doi: 10.1007/s11538020007734, https://doi.org/10.1007/s11538-020-00773-4.

See Also

WQDS, WQDC

Examples

```
gtrees=read.tree(file=system.file("extdata","dataGeneTreeSample",package="MSCquartets"))
tnames=taxonNames(gtrees)
QT=quartetTable(gtrees,tnames[1:6])
RQT=quartetTableResolved(QT)
DQT=quartetTableDominant(RQT,bigweights="finite")
D=quartetWeightedDist(DQT)
tree=NJ(D)
write.tree(tree)
plot(tree)
stree=WQDSAdjustLengths(tree)
write.tree(stree)
plot(stree)
```

Index