Package 'treebalance'

October 17, 2021

Title Computation of Tree (Im)Balance Indices

Version 1.1.0

Description The aim of the 'R' package 'treebalance' is to provide functions for the computation of a large variety of (im)balance indices for rooted trees. The package accompanies the manuscript "Tree balance indices: a comprehensive survey" by M. Fischer, L. Herbst, S. Kersting, L. Kuehn and K. Wicke (2021) <arXiv:2109.12281>, which gives a precise definition for the terms 'balance index' and 'imbalance index' (Section 3) and provides an overview of the terminology in this manual (Section 2). For further information on (im)balance indices, see also Fischer et al. (2021) <https://treebalance.wordpress.com>.
Considering both established and new (im)balance indices, 'treebalance' provides (among others) functions for calculating the following 18 established indices: the average leaf depth, the B1 and B2 index, the Colijn-Plazzotta rank, the normal, corrected, quadratic and equal weights Colless index, the family of Colless-like indices, the family of I-based indices, the Rogers J index, the Furnas rank, the rooted quartet index, the s-shape statistic, the Sackin index, the symmetry nodes index, the total cophenetic index and the variance of leaf depths. Additionally, we include 5 tree shape statistics that satisfy the definition of an (im)balance index but have not been thoroughly analyzed in terms of tree balance in the literature yet. These are: the maximum width, the maximum difference in widths, the maximal depth, the stairs1 and the stairs2 index.
As input, most functions of 'treebalance' require a rooted (phylogenetic) tree in 'phylo' format (as introduced in 'ape' 1.9 in November 2006). 'phylo' is used to store (phylogenetic) trees with no vertices of out-degree one. For further information on the format we kindly refer the reader to E. Paradis (2012) <http://ape-package.ird.fr/misc/FormatTreeR_24Oct2012.pdf>.

Author Mareike Fischer [aut],
Lina Herbst [aut],
Sophie Kersting [aut],
Luise Kuehn [aut, cre],
Kristina Wicke [aut]

Maintainer Luise Kuehn <treebalanceindices@gmail.com>

Depends R (>= 3.5.0)

Imports ape, memoise

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2021-10-17 15:10:02 UTC

R topics documented:

`areaPerPairI` *Calculation of the area per pair index for rooted trees*

Description

This function calculates the area per pair index $APP(T)$ for a given rooted tree T. The tree must not necessarily be binary. $APP(T)$ is defined as

$$APP(T) = \frac{2}{n \cdot (n-1)} \cdot \sum_{1 \leq i < j \leq n} d_T(i,j)$$

in which n denotes the number of leaves in T, and $d_T(i,j)$ denotes the number of edges on the path between the two leaves i and j. Note that $APP(T)$ can also be computed from the Sackin index $S(T)$ and the total cophenetic index $TCI(T)$ of T as $APP(T) = \frac{2}{n} \cdot S(T) - \frac{4}{n(n-1)} \cdot TCI(T)$ enabling efficient computation.

The area per pair index does not fulfill the definition of an (im)balance index given in "Tree balance indices: a comprehensive survey" (Fischer et al., 2021).

Usage

```
areaPerPairI(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`areaPerPairI` returns the area per pair index of the given tree.

Author(s)

Luise Kuehn

References

T. Araújo Lima, F. M. D. Marquitti, and M. A. M. de Aguiar. Measuring Tree Balance with Normalized Tree Area. arXiv e-prints, art. arXiv:2008.12867, 2020.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
areaPerPairI(tree)
```

avgLeafDepI — *Calculation of the average leaf depth index for rooted trees*

Description

This function calculates the average leaf depth $N(T)$ for a given rooted tree T. The tree must not necessarily be binary. $N(T)$ is defined as

$$N(T) = \frac{1}{n} \cdot \sum_{u \in V_{in}(T)} n_u$$

in which n denotes the number of leaves in T, $V_{in}(T)$ denotes the set of inner nodes of T and n_u denotes the number of leaves in the pending subtree that is rooted at the inner node u. Note that $N(T)$ can also be computed from the Sackin index $S(T)$ as $N(T) = \frac{1}{n} \cdot S(T)$. The average leaf depth is an imbalance index.

For $n = 1$ the function returns $N(T) = 0$ and a warning.

Usage

```
avgLeafDepI(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`avgLeafDepI` returns the average leaf depth of the given tree.

Author(s)

Luise Kuehn

References

M. J. Sackin. "Good" and "Bad" Phenograms. Systematic Biology, 21(2):225-226, 1972. doi: 10.1093/sysbio/21.2.225.

K.-T. Shao and R. R. Sokal. Tree Balance. Systematic Zoology, 39(3):266, 1990.
doi: 10.2307/2992186.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
avgLeafDepI(tree)
```

B1I *Calculation of the B1 index for rooted trees*

Description

This function calculates the $B1$ index $B1(T)$ for a given rooted tree T. The tree must not necessarily be binary. $B1(T)$ is defined as

$$B1(T) = \sum_{u \in V_{in}(T) \setminus \{\rho\}} h(T_u)^{-1}$$

in which $V_{in}(T) \setminus \{\rho\}$ denotes the set of inner vertices of T without the root, and $h(T_u)$ denotes the height of the pending subtree rooted at u. When restricted to binary trees, the $B1$ index is a balance index. For arbitrary trees it does not fulfill the definition of an (im)balance index.

For $n = 1$ the function returns $B1(T) = 0$ and a warning.

Usage

```
B1I(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`B1I` returns the B1 index of the given tree.

Author(s)

Sophie Kersting

References

K.-T. Shao and R. R. Sokal. Tree Balance. Systematic Zoology, 39(3):266, 1990.
doi: 10.2307/2992186.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
B1I(tree)
```

`B2I` *Calculation of the B2 index for rooted trees*

Description

This function calculates the B2 index $B2(T)$ for a given rooted tree T. The tree must not necessarily be binary. $B2(T)$ is defined as

$$B2(T) = -\sum_{x \in V_L(T)} p_x \cdot log(p_x)$$

in which $V_L(T)$ denotes the leaf set of T, and in which

$$p_x = \prod_{v \in anc(x)} \frac{1}{|child(v)|}$$

denotes the probability of reaching leaf x when starting at the root and assuming equiprobable branching at each vertex $v \in anc(x)$ with $anc(x)$ denoting the set of ancestors of x excluding x. $child(v)$ denotes the set of children of the inner vertex v.
The $B2$ index is a balance index.

For $n = 1$ the function returns $B2(T) = 0$ and a warning.

Usage

```
B2I(tree, logbase = 2)
```

Arguments

`tree`	A rooted tree in phylo format.
`logbase`	The base that shall be used for the logarithm. For binary trees it is common to use base 2.

Value

`B2I` returns the B2 index of the given tree.

Author(s)

Sophie Kersting, Luise Kuehn

References

K.-T. Shao and R. R. Sokal. Tree Balance. Systematic Zoology, 39(3):266, 1990.
doi: 10.2307/2992186.

P.-M. Agapow and A. Purvis. Power of Eight Tree Shape Statistics to Detect Nonrandom Diversification: A Comparison by Simulation of Two Models of Cladogenesis. Systematic Biology,

51(6):866-872, 2002.doi: 10.1080/10635150290102564.
URL https://doi.org/10.1080/10635150290102564.

M. Hayati, B. Shadgar, and L. Chindelevitch. A new resolution function to evaluate tree shape statistics. PLOS ONE, 14(11):e0224197, 2019. doi: 10.1371/journal.pone.0224197.
URL https://doi.org/10.1371/journal.pone.0224197.

M. Kirkpatrick and M. Slatkin. Searching for evolutionary patterns in the shape of a phylogenetic tree. Evolution, 47(4):1171-1181, 1993. doi: 10.1111/j.1558-5646.1993.tb02144.x.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
B2I(tree)
```

---

cherryI *Calculation of the cherry index for rooted trees*

Description

This function calculates the cherry index $ChI(T)$ for a given rooted tree T. The tree must not necessarily be binary. $ChI(T)$ is defined as the number of cherries in the tree. A cherry is a pair of leaves that have the same direct ancestor. Note, if a vertex u has x leaves as direct descendants, the number of cherries induced by u is $binom(x, 2)$.

The cherry index does not fulfill the definition of an (im)balance index given in "Tree balance indices: a comprehensive survey" (Fischer et al., 2021).

Usage

```
cherryI(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`cherryI` returns the cherry index of the given tree.

Author(s)

Sophie Kersting

References

A. McKenzie and M. Steel. Distributions of cherries for two models of trees. Mathematical Biosciences, 164(1):81-92, 2000. doi: 10.1016/s0025-5564(99)00060-7.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
cherryI(tree)
tree <- ape::read.tree(text="((,),((((,),),),(,)));")
cherryI(tree)
tree <- ape::read.tree(text="((,,,),(,,));")
cherryI(tree)
```

`collessI` *Calculation of the Colless index for rooted binary trees*

Description

This function calculates variants of the Colless index for a given rooted binary tree T. All of them are imbalance indices.

The original Colless index $C(T)$ is defined as

$$C(T) = \sum_{u \in V_{in}(T)} |n_{u_a} - n_{u_b}|$$

in which $V_{in}(T)$ denotes the set of all inner vertices of T, and in which n_{u_a} and n_{u_b} denote the number of leaves in the two pending subtrees that are rooted at the direct descendants of u.

The corrected Colless index $I_C(T)$ of T is defined as $I_C(T) = 0$ for $n = 1$ and $n = 2$ and for $n > 2$ as

$$I_C(T) = \frac{2 \cdot C(T)}{(n-1) \cdot (n-2)}$$

in which n denotes the total number of leaves in T.

The quadratic Colless index $QC(T)$ of T is defined as

$$QC(T) = \sum_{u \in V_{in}(T)} |n_{u_a} - n_{u_b}|^2$$

Special cases: For $n = 1$ the function returns $C(T) = I_C(T) = QC(T) = 0$ and a warning.

Usage

```
collessI(tree, method = "original")
```

Arguments

`tree`	A rooted binary tree in phylo format.
`method`	A character string specifying the version that shall be computed. It can be one of the following: "original", "corrected", "quadratic"

Value

`collessI` returns the Colless index of the given tree according to the chosen method.

Author(s)

Luise Kuehn and Sophie Kersting

References

D. Colless. Review of Phylogenetics: the theory and practice of phylogenetic systematics. Systematic Zoology, 1982. ISSN 00397989.

T. M. Coronado, M. Fischer, L. Herbst, F. Rosselló, and K. Wicke. On the minimum value of the Colless index and the bifurcating trees that achieve it. Journal of Mathematical Biology, 2020.doi: 10.1007/s00285-020-01488-9.

S. B. Heard. Patterns in tree balance among cladistic, phenetic, and randomly generated phylogenetic trees. Evolution, 1992. doi: 10.1111/j.1558-5646.1992.tb01171.x.

K. Bartoszek, T. M. Coronado, A. Mir, and F. Rosselló. Squaring within the Colless index yields a better balance index. Mathematical Biosciences, 331:108503, 2021. doi: 10.1016/j.mbs.2020.108503.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
collessI(tree, method="original")
collessI(tree, method="corrected")
collessI(tree, method="quadratic")
```

`collesslikeI` *Calculation of the Colless-like indices for rooted trees*

Description

This function calculates the Colless-like index for a given rooted tree T according to the chosen weight function f and dissimilarity D. The Colless-like index $CL(T)$ relative to D and f is the sum of the (D, f)-balance values over all inner vertices of the tree. More precisely,

$$CL(T) = \sum_{v \in V_{in}(T)} bal_{D,f}(v)$$

where $V_{in}(T)$ is the set of inner vertices of T. The (D, f)-balance value of v with children $v_1, ..., v_k$ is computed as

$$bal_{D,f}(v) = D(fs(T_{v_1}), ..., fs(T_{v_k}))$$

with D denoting the dissimilarity and fs denoting the f.size.
The f.size $fs(T)$ of a tree T uses the function f, which maps any integer to a non-negative real number, to build a weighted sum of the out-degrees of all vertices in T. More precisely,

$$fs(T) = \sum_{v \in V(T)} f(deg+(v))$$

where $V(T)$ is the set of all vertices of T and $deg+(v)$ denotes the out-degree (i.e. the number of children) of the vertex v. The f-functions that are already implemented are $f(x) = e^x$ and $f(x) = ln(x + e)$.
The dissimilarity $D(x_1, ..., x_k)$ of a vector $x_1, ..., x_k$ assigns a non-negative value to the vector, is independent of the order of the vector entries and equals zero if and only if $x_1 = ... = x_k$. In this implementation the following dissimilarity functions are already built-in: mean deviation from the median (mdm), the sample variance (var) and the sample standard deviation (sd).
`collesslikeI` also allows the use of other functions for the weight function f and the dissimilarity D.

Special cases: For $n = 1$ the function returns $CL(T) = 0$ and a warning.

Usage

```
collesslikeI(tree, f.size, dissim)
```

Arguments

`tree`	A rooted binary tree in phylo format.
`f.size`	A character string specifying the function f that shall be used to compute the f.size. It can be one of the following: "exp", "ln" or the name of a function as a string.
`dissim`	A character string specifying the dissimilarity that shall be used. It can be one of the following: "mdm", "var", "sd" or the name of a function as a string.

Value

`collesslikeI` returns the Colless-like index of the given tree according to the chosen function and dissimilarity.

Author(s)

Luise Kuehn, Sophie Kersting

References

A. Mir, L. Rotger, and F. Rosselló. Sound Colless-like balance indices for multifurcating trees. PLOSONE, 13(9):e0203401, 2018. doi: 10.1371/journal.pone.0203401

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
collesslikeI(tree, f.size="exp", dissim="mdm")
collesslikeI(tree, f.size="exp", dissim="var")
collesslikeI(tree, f.size="ln", dissim="sd")
myfsize <- function(x) return(x+1)
mydissim <- function(x) return (var(x))
collesslikeI(tree, f.size="myfsize",dissim = "mydissim")
```

`colPlaLab` *Calculation of the Colijn-Plazzotta rank for rooted trees*

Description

This function calculates the Colijn-Plazzotta rank $CP(T)$ for a given rooted tree T.

For a binary tree T, the Colijn-Plazzotta rank $CP(T)$ is recursively defined as $CP(T) = 1$ if T consists of only one leaf and otherwise

$$CP(T) = \frac{1}{2} \cdot CP(T_1) \cdot (CP(T_1) - 1) + CP(T_2) + 1$$

with $CP(T_1) \geq CP(T_2)$ being the ranks of the two pending subtrees rooted at the children of the root of T. This rank of T corresponds to its position in the lexicographically sorted list of (i, j): (1),(1,1),(2,1),(2,2),(3,1),... The Colijn-Plazzotta rank of binary trees has been shown to be an imbalance index.

For an arbitrary tree T whose maximal number of children of any vertex is l, the Colijn-Plazzotta rank $CP(T)$ is recursively defined as $CP(T) = 0$ if T is the empty tree (with no vertices), $CP(T) = 1$ if T consists of only one leaf and otherwise $CP(T) = \sum_{i=1}^{l} binom(CP(T_i) + i - 1, i)$ (with $CP(T_1) \leq ... \leq CP(T_l)$). If there are only $k < l$ pending subtrees rooted at the children of the root of T, then $T_1, ..., T_{l-k}$ are empty trees, i.e. $CP(T_1) = ... = CP(T_{l-k}) = 0$, and $CP(T_{l-k+1}), ..., CP(T_l)$ are the increasingly ordered CP-ranks of the k pending subtrees rooted at the children of the root of T. Note that if $k = l$ there are no empty trees.

For $n = 1$ the function returns $CP(T) = 1$ and a warning.

Note that problems can sometimes arise even for trees with small leaf numbers due to the limited range of computable values (ranks can reach INF quickly).

Usage

```
colPlaLab(tree, method)
```

Arguments

`tree`	A rooted tree in phylo format.
`method`	The method must be one of: "binary" or "arbitrary". Note that (only) in the arbitrary case vertices of out-degree 1 are allowed.

Value

`colPlaLab` returns the Colijn-Plazotta rank of the given tree according to the chosen method.

Author(s)

Sophie Kersting, Luise Kuehn

References

C. Colijn and G. Plazzotta. A Metric on Phylogenetic Tree Shapes. Systematic Biology, doi: 10.1093/sysbio/syx046.

N. A. Rosenberg. On the Colijn-Plazzotta numbering scheme for unlabeled binary rooted trees. Discrete Applied Mathematics, 2021. doi: 10.1016/j.dam.2020.11.021.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
colPlaLab(tree, method="binary")
tree <- ape::read.tree(text="(((,),(,)),(,),(,));")
colPlaLab(tree, method="arbitrary")
```

`colPlaLab_inv` *Generation of the rooted binary tree corresponding to a given Colijn-Plazzotta rank*

Description

This function generates the unique rooted binary tree T (in phylo format) that corresponds to the given Colijn-Plazzotta rank $CP(T)$. It is the inverse function of colPlaLab() with the method 'binary'.

`colPlaLab()`: For a given rooted binary tree T, $CP(T)$ is recursively defined as $CP(T) = 1$ if T consists of only one vertex and otherwise $CP(T) = \frac{1}{2} \cdot CP(T_1) \cdot (CP(T_1) - 1) + CP(T_2) + 1$ with $CP(T_1) \geq CP(T_2)$ being the ranks of the two pending subtrees rooted at the children of T. The rank $CP(T)$ of T corresponds to its position in the lexicographically sorted list of (i, j): (1),(1,1),(2,1),(2,2),(3,1),...

`colPlaLab_inv()`: For a given rank CP the corresponding tree T can be reconstructed by starting from one vertex ρ (labelled CP) and recursively splitting vertices whose labels h are greater than 1 into two children with the labels:

$$i = \left\lceil \frac{1 + \sqrt{8 \cdot h - 7}}{2} \right\rceil - 1$$

and

$$j = h - \frac{i \cdot (i-1)}{2} - 1$$

until there are no more vertices to split.
For $CP = 1$ the function returns the smallest possible tree in the phylo format: the tree consisting of a single edge.

Note that problems can arise for extremely high input values (>10e+18).

Usage

```
colPlaLab_inv(rank)
```

Arguments

`rank`	An integer denoting the Colijn-Plazzotta rank of the sought tree.

Value

`colPlaLab_inv` returns the unique rooted binary tree for the given rank.

Author(s)

Sophie Kersting

References

C. Colijn and G. Plazzotta. A Metric on Phylogenetic Tree Shapes. Systematic Biology, 67(1):113-126,2018. doi: 10.1093/sysbio/syx046.

N. A. Rosenberg. On the Colijn-Plazzotta numbering scheme for unlabeled binary rooted trees. Discrete Applied Mathematics, 291:88-98, 2021. doi: 10.1016/j.dam.2020.11.021.

Examples

```
colPlaLab_inv(22)
```

`ewCollessI` *Calculation of the equal weights Colless index for rooted binary trees*

Description

This function calculates the equal weights Colless index $I_2(T)$ for a given rooted binary tree T. $I_2(T)$ is defined as

$$I_2(T) = \frac{1}{n-2} \cdot \sum_{u \in V_{in}(T), n_u > 2} \frac{|n_{u_a} - n_{u_b}|}{n_u - 2}$$

in which $V_{in}(T)$ denotes the set of all inner vertices of T, and in which n_u, n_{u_a} and n_{u_b} denote the number of leaves in the pending subtrees that are rooted at u and the two direct descendants of u. The equal weights Colless index is an imbalance index.

For $n = 1$ and $n = 2$ the function returns $I_2(T) = 0$ and a warning.

Usage

```
ewCollessI(tree)
```

Arguments

`tree`	A rooted binary tree in phylo format.

Value

`ewCollessI` returns the equal weights Colless index of the given tree.

Author(s)

Luise Kuehn

References

A. O. Mooers and S. B. Heard. Inferring Evolutionary Process from Phylogenetic Tree Shape. The Quarterly Review of Biology, 72(1), 1997. doi: 10.1086/419657.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
ewCollessI(tree)
```

`furnasI` *Calculation of the Furnas rank for rooted binary trees*

Description

This function calculates the Furnas rank $F(T)$ for a given rooted binary tree T. $F(T)$ is the unique rank of the tree T among all rooted binary trees with n leaves in the left-light rooted ordering. For details on the left-light rooted ordering as well as details on how the Furnas rank is computed, see "The generation of random, binary unordered trees" by G.W. Furnas (1984) or "Tree balance indices: a comprehensive survey" by Fischer et al. (2021). The Furnas rank is a balance index.

The concept of assigning each rooted binary tree a unique tuple $(rank, n)$ allows to store many trees with minimal storage use. When the tree gets too big, the function returns Inf.

Usage

```
furnasI(tree)
```

Arguments

`tree`	A rooted binary tree in phylo format.

Value

`furnasI` returns the unique Furnas rank of the given tree, i.e. the rank of the tree among all rooted binary trees with n leaves in the left-light rooted ordering.

Author(s)

Luise Kuehn, Lina Herbst

References

G. W. Furnas. The generation of random, binary unordered trees. Journal of Classification, 1984. doi: 10.1007/bf01890123. URL https://doi.org/10.1007/bf01890123.

M. Kirkpatrick and M. Slatkin. Searching for evolutionary patterns in the shape of a phylogenetic tree. Evolution, 1993. doi: 10.1111/j.1558-5646.1993.tb02144.x.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
furnasI(tree)
```

`furnasI_inv` *Calculation of rooted binary tree for tuple (rank, leaf number)*

Description

This function calculates the unique tree T (in phylo format) for two given integer values r and n, with n denoting the number of leaves of T and r denoting the rank of T in the left-light rooted ordering of all rooted binary trees with n leaves. It is the inverse function of `furnasI()`. For details on how to calculate T (including algorithm) see "The generation of random, binary unordered trees" by G.W. Furnas (1984) or "Tree balance indices: a comprehensive survey" by Fischer et al. (2021).

`furnasI_inv` can be used e.g. to generate random rooted binary trees with a certain number of leaves. Also, the concept of assigning each rooted binary tree a unique tuple $(rank, n)$ allows to store many trees with minimal storage use.

Usage

```
furnasI_inv(rank, n)
```

Arguments

`rank`	An integer denoting the rank of the sought tree among all rooted binary trees with n leaves.
`n`	An integer denoting the number of leaves of the sought tree.

Value

`furnasI_inv` returns the unique tree (in phylo format) for the given leaf number and rank.

Author(s)

Sophie Kersting

References

G. W. Furnas. The generation of random, binary unordered trees. Journal of Classification, 1984. doi: 10.1007/bf01890123. URL https://doi.org/10.1007/bf01890123.

Examples

```
furnasI_inv(rank=6,n=8)
```

`getDescMatrix` *Auxiliary functions*

Description

`getDescMatrix` - Creates a matrix that contains the descendants of node i in row i.

`getAncVec` - Creates a vector that contains the parent (direct ancestor) of node i at position i.

`getNodesOfDepth` - Creates a matrix that contains the nodes of depth i in row i.

`symBucketLexicoSort` - Sorts the pairs of numbers lexicographically and returns ranking. Uses bucket sort.

`getAllAncestors` - Returns all ancestors of v including v itself.

`cPL_inv` - Returns the binary tree that belongs to the input label in an incomplete Newick format.

`maxDepthLeaf` - Returns the maximal depth of a leaf in the subtree that is rooted at v.

`get.subtreesize` - Creates a vector that contains at the i-th position the number of leaves in the pending subtree rooted at i.

`getlca` - Returns the name of the lowest common ancestor of the two input vertices v and w.

`we_eth` - Returns the Wedderburn-Etherington number $we(n)$ for a given non-negative integer n.

`getfurranks` - Returns for each vertex i the Furnas rank of the subtree rooted at i.

`getsubtree` - Returns the pending subtree (in phylo format) that is rooted at the input vertex. If the input vertex is a leaf, the function returns the standard tree for $n = 1$ (with 1 edge).

`is_binary` - Returns TRUE if the input tree is binary and FALSE otherwise.

`is_phylo` - Tests all requirements of the phylo format, and returns TRUE if the tree is correctly formatted, else FALSE with detailed feedback on the features that are not met.

`tree_decomposition` - Returns a list of length two, which contains the two pending subtrees that are rooted at the children of the root of the input tree. The smaller one (according to the number of leaves) is stated first.

`tree_merge` - Returns a rooted tree T in phylo format, which contains the input trees $tree1$ and $tree2$ as "left" and "right" maximal pending subtrees.

`treenumber` - Returns the unique tree number $tn(T)$ of the given tree. $tn(T)$ is the rank of the tree T among all rooted binary trees in the left-light rooted ordering. It can be calculated as follows:

$$tn(T) = F(T) + \sum_{i=1}^{n-1} we(i)$$

in which n is the number of leaves in T, $F(T)$ is the Furnas rank of T, i.e. the rank of T in the left-light rooted ordering of all rooted binary trees with n leaves, and $we(i)$ is the Wedderburn-Etherington number of i. The concept of assigning each rooted binary tree a unique tree number allows to store many trees with minimal storage use. For $n = 1$ the function returns $tn(T) = 1$ and a warning.

`treenumber_inv` - Returns the unique tree (in phylo format) for the given tree number.

`auxE_l_X` - Returns the sum of all products of l different values in X.

Usage

```
getDescMatrix(tree)

getAncVec(tree)

getNodesOfDepth(mat, root, n)

symBucketLexicoSort(workLabs)

getAllAncestors(tree, v)

cPL_inv(label)

maxDepthLeaf(tree, v = length(tree$tip.label) + 1)

get.subtreesize(tree)

getlca(tree, v, w)

we_eth(n)

getfurranks(tree)

getsubtree(tree, subroot)

is_binary(tree)

is_phylo(tree)

tree_decomposition(tree)

tree_merge(tree1, tree2)

treenumber(tree)

treenumber_inv(treenum)

auxE_l_X(subX, Xset)
```

Arguments

`tree`	A rooted tree in phylo format, >= 2 leaves
`mat`	Descendants matrix from `getDescMatrix`
`root`	Number (label) of the root of the tree
`n`	Number of leaves of the tree
`workLabs`	numeric matrix (2 columns)
`v`	A vertex of the tree.
`label`	A Colijn-Plazotta label of desired tree, a positive integer.
`w`	A vertex of the tree.
`subroot`	A vertex of the tree. It is not recommended to use leaves as subroots.
`tree1`	A rooted tree in phylo format.
`tree2`	A rooted tree in phylo format.
`treenum`	An integer denoting the tree number of the sought tree.
`subX`	integer >=1, size of the subsets of X.
`Xset`	Vector (multiset) of numeric values.

Value

`desc_mat` numeric matrix

`anc_vec` numeric vector

`nodes_of_depth` numeric matrix

`ranking` numeric vector

`vectorWithAncs` numeric vector

Author(s)

Sophie Kersting, Luise Kuehn and Lina Herbst

Examples

```
mat <- cbind(c(7,7,6,5,5,6),c(1,2,3,4,6,7))
tree <- list(edge=mat, tip.label=c("","","",""), Nnode=3)
getDescMatrix(tree)
mat <- cbind(c(5,5,5,5),c(1,2,3,4))
tree <- list(edge=mat, tip.label=c("","","",""), Nnode=1)
getDescMatrix(tree)
getAncVec(tree)
getNodesOfDepth(mat=getDescMatrix(tree),root=length(tree$tip.label)+1,
n=length(tree$tip.label))
myWorkLabs <- cbind(c(0,1,2,3,1,0),c(0,2,2,4,1,0))
symBucketLexicoSort(myWorkLabs)
getAllAncestors(tree,v=6)
cPL_inv(label=6)
maxDepthLeaf(tree,v=6)
```

```
get.subtreesize(tree)
getlca(tree,1,2)
we_eth(5)
getfurranks(tree)
getsubtree(tree,4)
is_binary(ape::read.tree(text="((((,),),(,)),(((,),),(,)));"))
is_phylo(ape::read.tree(text="((((,),),(,)),(((,),),(,)));"))
tree_decomposition(ape::read.tree(text="((((,),),(,)),(((,),),(,)));"))
treeA <- ape::read.tree(text="(((,),),(,));")
treeB <- ape::read.tree(text="((,),);")
tree_merge(treeA, treeB)
treenumber(ape::read.tree(text="((((,),),(,)),(((,),),(,)));"))
treenumber_inv(192)
auxE_l_X(subX=3,Xset=c(1,1,2,2))
```

`IbasedI` *Calculation of the I-based indices for rooted trees*

Description

This function calculates I-based indices $I(T)$ for a given rooted tree T. Note that the leaves of the tree may represent single species or groups of more than one species. Thus, a vector is required that contains for each leaf the number of species that it represents. The tree may contain few polytomies, which are not allowed to concentrate in a particular region of the tree (see p. 238 in Fusco(1995)).

Let v be a vertex of T that fulfills the following criteria: a) The number of descendant (terminal) species of v is $k_v > 3$ (note that if each leaf represents only one species k_v is simply the number of leaves in the pending subtree rooted at v), and b) v has exactly two children.

Then, we can calculate the I_v value as follows:

$$I_v = \frac{k_{v_a} - \lceil \frac{k_v}{2} \rceil}{k_v - 1 - \lceil \frac{k_v}{2} \rceil}$$

in which k_{v_a} denotes the number of descendant (terminal) species in the bigger one of the two pending subtrees rooted at v.

As the expected value of I_v under the Yule model depends on k_v, Purvis et al. (2002) suggested to take the corrected values I'_v or I^w_v instead.
The I'_v value of v is defined as follows: $I'_v = I_v$ if k_v is odd and $I'_v = \frac{k_v-1}{k_v} \cdot I_v$ if k_v is even.
The I^w_v value of v is defined as follows:

$$I^w_v = \frac{w(I_v) \cdot I_v}{mean_{V'(T)} w(I_v)}$$

where $V'(T)$ is the set of inner vertices of T that have precisely two children and $k_v \geq 4$, and $w(I_v)$ is a weight function with $w(I_v) = 1$ if k_v is odd and $w(I_v) = \frac{k_v-1}{k_v}$ if k_v is even and $I_v > 0$, and $w(I_v) = \frac{2\cdot(k_v-1)}{k_v}$ if k_v is even and $I_v = 0$.

The I-based index of T can now be calculated using different methods. Here, we only state the version for the I' correction method, but the non-corrected version or the I_v^w corrected version works analoguously.

1. root: The I' index of T equals the I'_v value of the root of T, i.e. $I'(T) = I'_\rho$, provided that the root fulfills the two criteria. Note that this method does not fulfil the definition of an (im)balance index.
2. median: The I' index of T equals the median I'_v value of all vertices v that fulfill the two criteria.
3. total: The I' index of T equals the summarised I'_v values of all vertices v that fulfill the two criteria.
4. mean: The I' index of T equals the mean I'_v value of all vertices v that fulfill the two criteria.
5. quartile deviation: The I' index of T equals the quartile deviation (half the difference between third and first quartile) of the I'_v values of all vertices v that fulfill the two criteria.

Usage

```
IbasedI(
  tree,
  specnum = rep(1, length(tree$tip.label)),
  method = "mean",
  correction = "none",
  logs = TRUE
)
```

Arguments

`tree`	A rooted tree in phylo format (with possibly few polytomies).
`specnum`	A vector whose i-th entry is the number of species that the i-th leaf represents. (default is 1,...,1)
`method`	A character string specifying the method that shall be used to calculate $I(T)$. It can be one of the following: "root", "median", "total", "mean", "quartdev"
`correction`	A character string specifying the correction method that shall be applied to the I values. It can be one of the following: "none", "prime", "w"
`logs`	Boolean value, (default true), determines if the number of suitable nodes (i.e. nodes that fulfill the criteria) and polytomies in the tree should be printed

Value

`IbasedI` returns an I-based balance index of the given tree according to the chosen (correction and) method.

Author(s)

Luise Kuehn and Sophie Kersting

References

G. Fusco and Q. C. Cronk. A new method for evaluating the shape of large phylogenies. Journal of Theoretical Biology, 1995. doi: 10.1006/jtbi.1995.0136. URL https://doi.org/10.1006/jtbi.1995.0136.

A. Purvis, A. Katzourakis, and P.-M. Agapow. Evaluating Phylogenetic Tree Shape: Two Modifications to Fusco & Cronks Method. Journal of Theoretical Biology, 2002. doi: 10.1006/jtbi.2001.2443. URL https://doi.org/10.1006/jtbi.2001.2443.

Examples

```
tree <- ape::read.tree(text="((((((,),),),),),);")
IbasedI(tree, method="mean")
IbasedI(tree, method="mean", correction="prime", specnum=c(1,1,2,1,1,1))
```

---

maxDelW *Calculation of the maximal difference in widths for a rooted tree*

Description

This function calculates the maximal difference in widths $maxDelW(T)$ for a given rooted tree T. The tree must not necessarily be binary. $maxDelW(T)$ is defined as

$$maxDelW(T) = \max_{i=0,\ldots,h(T)-1} w(i+1) - w(i)$$

in which $h(T)$ denotes the height of the tree T and $w(i)$ denotes the number of vertices in T that have depth i. The maximal difference in widths is a balance index.

Usage

```
maxDelW(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`maxDelW` returns the maximal difference in widths of a tree.

Author(s)

Sophie Kersting, Luise Kuehn

References

C. Colijn and J. Gardy. Phylogenetic tree shapes resolve disease transmission patterns. Evolution, Medicine, and Public Health, 2014(1):96-108, 2014. ISSN 2050-6201. doi: 10.1093/emph/eou018.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
maxDelW(tree)
tree <- ape::read.tree(text="((,),((((,),),),(,)));")
maxDelW(tree)
```

`maxDepth` *Calculation of the maximal depth of the tree*

Description

This function calculates the maximal depth of any vertex in a rooted tree T, which is at the same time its height $h(T)$. The tree must not necessarily be binary. Formally, $h(T)$ is defined as

$$h(T) = \max_{v \in V(T)} \delta(v)$$

with $\delta(v)$ being the depth of the vertex v. The maximal depth is an imbalance index.

For $n = 1$ the function returns $h(T) = 0$ and a warning.

Usage

```
maxDepth(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`maxDepth` returns the maximal depth, i.e. height, of a tree.

Author(s)

Luise Kuehn, Sophie Kersting

References

C. Colijn and J. Gardy. Phylogenetic tree shapes resolve disease transmission patterns. Evolution, Medicine, and Public Health, 2014(1):96-108, 2014. ISSN 2050-6201. doi: 10.1093/emph/eou018.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
maxDepth(tree)
tree <- ape::read.tree(text="((,),((((,),),),(,)));")
maxDepth(tree)
```

`maxWidth` *Calculation of the maximal width of the tree*

Description

This function calculates the maximal width $maxWidth(T)$ for a given rooted tree T. The tree must not necessarily be binary. $maxWidth(T)$ is defined as

$$maxWidth(T) = \max_{i=0,\ldots,h(T)} w(i)$$

in which $h(T)$ denotes the height of the tree T and $w(i)$ denotes the number of vertices in T that have depth i. The maximal width is a balance index.

Usage

```
maxWidth(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`maxWidth` returns the maximal width of a tree.

Author(s)

Sophie Kersting

References

C. Colijn and J. Gardy. Phylogenetic tree shapes resolve disease transmission patterns. Evolution, Medicine, and Public Health, 2014(1):96-108, 2014. ISSN 2050-6201. doi: 10.1093/emph/eou018.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
maxWidth(tree)
tree <- ape::read.tree(text="((,),((((,),),),(,)));")
maxWidth(tree)
```

rogersI *Calculation of the Rogers J index for rooted binary trees*

Description

This function calculates the Rogers J index $J(T)$ for a given rooted binary tree T. It is defined as the number of inner vertices whose balance value is unequal to zero, more precisely

$$J(T) = \sum_{u \in V_{in}(T)} (1 - I(n_{u_a} = n_{u_b}))$$

in which $V_{in}(T)$ denotes the set of all inner vertices of T, and in which n_{u_a} and n_{u_b} denote the number of leaves in the two pending subtrees that are rooted at the direct descendants of u.
Special cases: For $n = 1$, the function returns $J(T) = 0$ and a warning.

Usage

```
rogersI(tree)
```

Arguments

`tree`	A rooted binary tree in phylo format.

Value

`rogersI` returns the Rogers J index of the given tree.

Author(s)

Sophie Kersting

References

J. S. Rogers. Central Moments and Probability Distributions of Three Measures of Phylogenetic Tree Imbalance. Systematic Biology, 45(1):99-110, 1996. doi: 10.1093/sysbio/45.1.99.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
rogersI(tree)
```

`rQuartetI` *Calculation of the rooted quartet index for rooted trees*

Description

This function calculates the rooted quartet index $rQI(T)$ for a given rooted tree T. The tree must not necessarily be binary.

Let T be a rooted tree, whose leaves are $1, ..., n$. Let P_4 denote the set of all subsets of $\{1, ..., n\}$ that have cardinality 4. Let $T(Q)$ denote the rooted quartet on $Q \in P_4$ that is obtained by taking the subgraph of T that is induced by Q and supressing its outdegree-1 vertices. $T(Q)$ can have one of the five following shapes:

- Q_0^*: This is the caterpillar tree shape on 4 leaves, i.e. "`(,(,(,)));`" in Newick format. It has 2 automorphisms.
- Q_1^*: This is the tree shape on 4 leaves that has three pending subtrees rooted at the children of the root of T, one of them being a cherry and the other two being single vertices, i.e. "`((,),,);`" in Newick format. It has 4 automorphisms.
- Q_2^*: This is the tree shape on 4 leaves that has two pending subtrees rooted at the children of the root of T, one of them being a star tree shape on 3 leaves and the other one being a single vertex, i.e. "`((,,),);`" in Newick format. It has 6 automorphisms.
- Q_3^*: This is the fully balanced binary tree shape on 4 leaves, i.e. "`((,),(,));`" in Newick format. Its has 8 automorphisms.
- Q_4^*: This is the star tree shape on 4 leaves, i.e. "`(,,,);`" in Newick format. It has 24 automorphisms.

$T(Q)$ is assigned an rQI-value based on its shape, i.e. $rQI(T(Q)) = q_i$ if $T(Q)$ has the shape Q_i^*. The values $q_0, ..., q_4$ are chosen in such a way that they increase with the symmetry of the shape as measured by means of its number of automorphisms. Coronado et al. (2019) suggested the values $q_0 = 0$ and $q_i = i$ or $q_i = 2^i$ for $i = 1, ..., 4$.
The rooted quartet index $rQI(T)$ of the tree T is then defined as the sum of the rQI-values of its rooted quartets:

$$rQI(T) = \sum_{Q \in P_4} rQI(T(Q))$$

The rooted quartet index is a balance index.

Usage

```
rQuartetI(tree, shapeVal = c(0, 1, 2, 3, 4))
```

Arguments

`tree`	A rooted tree in phylo format.
`shapeVal`	A vector of length 5 containing the shape values $q_0, ..., q_4$. Default is $(q_0, q_1, q_2, q_3, q_4) = (0, 1, 2, 3, 4)$.

Value

`rQuartetI` returns the rooted quartet index of the given tree based on the chosen shape values (see description for details).

Author(s)

Sophie Kersting

References

T. M. Coronado, A. Mir, F. Rosselló, and G. Valiente. A balance index for phylogenetic trees based on rooted quartets. Journal of Mathematical Biology, 79(3):1105-1148, 2019. doi: 10.1007/s00285-019-01377-w. URL https://doi.org/10.1007/s00285-019-01377-w.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
rQuartetI(tree)
```

`sackinI` *Calculation of the Sackin index for rooted trees*

Description

This function calculates the Sackin index $S(T)$ for a given rooted tree T. The tree must not necessarily be binary. $S(T)$ is defined as

$$S(T) = \sum_{x \in V_L(T)} \delta(x) = \sum_{u \in V_{in}(T)} n_u$$

in which $V_L(T)$ denotes the leaf set of T, $\delta(x)$ denotes the depth of the leaf x, $V_{in}(T)$ denotes the set of inner vertices in T and n_u denotes the number of leaves in the pending subtree that is rooted at u. The Sackin index is an imbalance index.

For $n = 1$ the function returns $S(T) = 0$ and a warning.

Usage

```
sackinI(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`sackinI` returns the Sackin index of the given tree.

Author(s)

Luise Kuehn

References

M.J. Sackin. "Good" and "Bad" Phenograms. Systematic Biology, 21(2):225-226, 1972. doi: 10.1093/sysbio/21.2.225.

K.-T. Shao and R.R. Sokal. Tree Balance. Systematic Zoology, 39(3):266, 1990. doi: 10.2307/2992186.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
sackinI(tree)
```

`sShapeI` *Calculation of the s-shape statistic for rooted trees*

Description

This function calculates the s-shape statistic $sShape(T)$ for a given rooted tree T. The tree must not necessarily be binary, however $sShape$ only fulfils the definition of an imbalance index on the space of binary trees. $sShape(T)$ is defined as

$$sShape(T) = \sum_{u \in V_{in}(T)} log(n_u - 1)$$

in which $V_{in}(T)$ denotes the set of inner vertices of T and n_u denotes the number of leaves in the pending subtree that is rooted at u. An arbitrary logarithm base can be used (for binary trees it is common to use base 2).

For $n = 1$ the function returns $sShape(T) = 0$ and a warning.

Usage

```
sShapeI(tree, logbase = 2)
```

Arguments

`tree`	A rooted tree in phylo format.
`logbase`	The logarithm base that shall be used.

Value

`sShapeI` returns the s-shape statistic of the given tree.

Author(s)

Luise Kuehn

References

M.G. Blum and O. Francois. Which random processes describe the tree of life? a large-scale study of phylogenetic tree imbalance. Systematic Biology, 2006.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
sShapeI(tree)
```

`stairs1` *Calculation of the stairs1 value for rooted binary trees*

Description

This function calculates the stairs1 value $st1(T)$ for a given rooted binary tree T. It is a modified version of the Rogers J index and is defined as the fraction of inner vertices whose balance value is unequal to zero, more precisely

$$st1(T) = \frac{1}{n-1} \cdot \sum_{u \in V_{in}(T)} (1 - I(n_{u_a} = n_{u_b}))$$

in which $V_{in}(T)$ denotes the set of all inner vertices of T, and in which n_{u_a} and n_{u_b} denote the number of leaves in the two pending subtrees that are rooted at the direct descendants of u. The stairs1 value is an imbalance index.

Special cases: For $n = 1$, the function returns $st1(T) = 0$ and a warning.

Usage

```
stairs1(tree)
```

Arguments

`tree`	A rooted binary tree in phylo format.

Value

`stairs1` returns the stairs1 value of the given tree.

Author(s)

Sophie Kersting

References

M. M. Norström, M. C. Prosperi, R. R. Gray, A. C. Karlsson, and M. Salemi. PhyloTempo: A Set of R Scripts for Assessing and Visualizing Temporal Clustering in Genealogies Inferred from Serially Sampled Viral Sequences. Evolutionary Bioinformatics, 8:EBO.S9738, 2012. ISSN 1176-9343, 1176-9343. doi:10.4137/EBO.S9738.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
stairs1(tree)
```

`stairs2` *Calculation of the stairs2 value for rooted binary trees*

Description

This function calculates the stairs2 value $st2(T)$ for a given rooted binary tree T. It is defined as the mean ratio between the leaf numbers of the smaller and larger pending subtree over all inner vertices, more precisely

$$st2(T) = \frac{1}{n-1} \cdot \sum_{u \in V_{in}(T)} \frac{n_{u_a}}{n_{u_b}}$$

in which $V_{in}(T)$ denotes the set of all inner vertices of T, and in which $n_{u_a} \geq n_{u_b}$ denote the number of leaves in the two pending subtrees that are rooted at the direct descendants of u. The stairs2 value is an imbalance index.

Special cases: For $n = 1$, the function returns $st2(T) = 0$ and a warning.

Usage

```
stairs2(tree)
```

Arguments

`tree`	A rooted binary tree in phylo format.

Value

`stairs2` returns the stairs2 value of the given tree.

Author(s)

Sophie Kersting

References

C. Colijn and J. Gardy. Phylogenetic tree shapes resolve disease transmission patterns. Evolution, Medicine, and Public Health, 2014(1):96-108, 2014. ISSN 2050-6201. doi: 10.1093/emph/eou018.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
stairs2(tree)
```

---

symNodesI *Calculation of the symmetry nodes index for rooted binary trees*

Description

This function calculates the symmetry nodes index $SNI(T)$ for a given rooted binary tree T. $SNI(T)$ is defined as the number of inner vertices v that are not symmetry nodes, i.e. the two pending subtrees rooted at the children of v do not have the same tree shape.

For $n = 1$ the function returns $SNI(T) = 0$ and a warning.

Usage

```
symNodesI(tree)
```

Arguments

`tree`	A rooted binary tree in phylo format.

Value

`symNodesI` returns the symmetry nodes index of the given tree.

Author(s)

Sophie Kersting

References

S. J. Kersting and M. Fischer. Measuring tree balance using symmetry nodes — A new balance index and its extremal properties. Mathematical Biosciences, page 108690, 2021. ISSN 0025-5564. doi:https://doi.org/10.1016/j.mbs.2021.108690

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
symNodesI(tree)
```

totCophI *Calculation of the total cophenetic index for rooted trees*

Description

This function calculates the total cophenetic index $TCI(T)$ of a given rooted tree T. The tree must not necessarily be binary. $TCI(T)$ is defined as

$$TCI(T) = \sum_{1 \leq i < j \leq n} \delta(lca(i,j)) = \sum_{u \in V_{in}(T) \setminus \{\rho\}} binom(n_u, 2)$$

in which $\delta(lca(i,j))$ denotes the depth of the last common ancestor of the two leaves i and j and $V_{in}(T) \setminus \{\rho\}$ denotes the set of all inner vertices exept the root and n_u denotes the number of descendant leaves of u. The second formula is useful for efficient computation of $TCI(T)$. The total cophenetic index is an imbalance index.

For $n = 1$ the function returns $TCI(T) = 0$.

Usage

```
totCophI(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`totCophI` returns the total cophenetic index of the given tree.

Author(s)

Sophie Kersting

References

A. Mir, F. Rosselló, and L. Rotger. A new balance index for phylogenetic trees. Mathematical Bio-sciences, 241(1):125-136, 2013. doi: 10.1016/j.mbs.2012.10.005.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
totCophI(tree)
tree <- ape::read.tree(text="((,),((((,),),),(,)));")
totCophI(tree)
tree <- ape::read.tree(text="((,,,),(,,));")
totCophI(tree)
```

varLeafDepI *Calculation of the variance of leaf depths index for rooted trees*

Description

This function calculates the variance of leaf depths index $VLD(T)$ for a given rooted tree T. The tree must not necessarily be binary. $VLD(T)$ is defined as

$$VLD(T) = \frac{1}{n} \cdot \sum_{x \in V_L(T)} (\delta(x) - N(T))^2$$

in which n denotes the number of leaves of T, $V_L(T)$ denotes the set of leaves of T, $\delta(x)$ denotes the depth of the leaf x and $N(T)$ denotes the average leaf depth of T.

For $n = 1$ the function returns $VLD(T) = 0$ and a warning.

Usage

```
varLeafDepI(tree)
```

Arguments

`tree`	A rooted tree in phylo format.

Value

`varLeafDepI` returns the variance of leaf depths index of the given tree.

Author(s)

Sophie Kersting

References

T. M. Coronado, A. Mir, F. Rosselló, and L. Rotger. On Sackin's original proposal: the variance of the leaves' depths as a phylogenetic balance index. BMC Bioinformatics, 21(1), 2020. doi: 10.1186/s12859-020-3405-1. URL https://doi.org/10.1186/s12859-020-3405-1.

M. J. Sackin. "Good" and "Bad" Phenograms. Systematic Biology, 21(2):225-226, 1972. doi: 10.1093/sysbio/21.2.225.

K.-T. Shao and R. R. Sokal. Tree Balance. Systematic Zoology, 39(3):266, 1990. doi: 10.2307/2992186.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
varLeafDepI(tree)
```

`wedEth` *Wedderburn Etherington numbers (from OEIS)*

Description

Contains a vector of Wedderburn Etherington numbers for $n = 1$ to $n = 2545$.

Usage

```
data(wedEth)
```

Format

numerical vector

Source

OEIS Sequence A001190 available at https://oeis.org/A001190

Examples

```
data(wedEth)
wedEth[5]
```

`weighL1dist` *Calculation of weighted l1 distance index for rooted binary trees*

Description

This function calculates the weighted l1 distance index $D_{l1}(T)$ for a given rooted binary tree T. $D_{l1}(T)$ is defined as

$$D_{l1}(T) = \sum_{z=2}^{n} z \cdot |f_n(z) - p_n(z)|$$

in which n denotes the number of leaves of T, $f_n(z)$ denotes the frequency of pending subtrees of size z in T and $p_n(z)$ is the expected number of pending subtrees of size z under the Yule model, i.e. $p_n(z) = \frac{1}{n-1}$ if $z = n$ and otherwise $\frac{n}{n-1} \cdot \frac{2}{z \cdot (z+1)}$.

For $n = 1$ the function returns $D_{l1}(T) = 0$.

Usage

```
weighL1dist(tree)
```

Arguments

`tree`	A rooted binary tree in phylo format.

Value

`weighL1distI` returns the weighted l1 distance index of the given tree.

Author(s)

Sophie Kersting

References

M. G. Blum and O. François. On statistical tests of phylogenetic tree imbalance: The Sackin and other indices revisited. Mathematical Biosciences, 195(2):141-153, 2005. doi: 10.1016/j.mbs.2005.03.003.

Examples

```
tree <- ape::read.tree(text="((((,),),(,)),(((,),),(,)));")
weighL1dist(tree)
```

Index